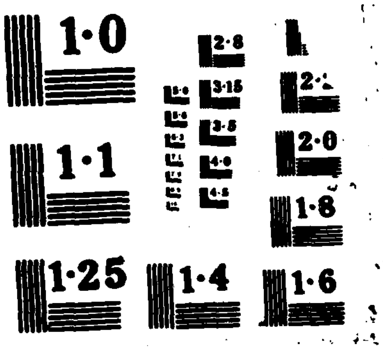
AD-A191 296 COMPUTER SCIENCE AND STATISTICS PROCEEDINGS OF THE SYMPOSIUM ON THE INTER..(U) COLORADO STATE UNIV FORT COLLINS T J BOARDMAN 26 AUG 87 AFOSR-TR-88-0153 1/5

UNCLASSIFIED AFOSR-86-0070 F/G 12/6 NL


1·0
2·8
3·15
2·5
4·0
4·5
2·2
2·0
1·1
1·8
1·25
1·4
1·6

and Statistics:

Proceedings of the 18th Symposium on the Interface

March 19-21, 1986

Thomas J. Boardman

edito

INTERFACE

1986

6c. ADDRESS (*City, State, and ZIP Code*) Fort Collins, CO	7b. ADDRESS (*City, State, and ZIP Code*) AFOSR/NM Bolling AFB DC 20332-6448

8a. NAME OF FUNDING/SPONSORING ORGANIZATION AFOSR/NM	8b. OFFICE SYMBOL (*If applicable*)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER AFOSR-86-0070

8c. ADDRESS (*City, State, and ZIP Code*)
AFOSR/NM
Bolling AFB DC 20332-6448

10. SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT ACCESSION NO.
61102F	2304	A5	

11. TITLE (*Include Security Classification*)
Computer Science and Statistics: 18th Symposium on the Interface

12. PERSONAL AUTHOR(S)
Thomas J. Boradman

13a. TYPE OF REPORT Final	13b. TIME COVERED FROM 2/15/86 TO 2/14/87	14. DATE OF REPORT (*Year, Month, Day*) 87/08/26	15. PAGE COUNT 466

16. SUPPLEMENTARY NOTATION

17. COSATI CODES

FIELD	GROUP	SUB-GROUP

18. SUBJECT TERMS (*Continue on reverse if necessary and identify by block number*)

19. ABSTRACT (*Continue on reverse if necessary and identify by block number*)

This report contains the proceedings of a conference on the interface between computer science and statistics. The conference "Computer Science and Statistics: 18th Symposium on the Interface" was held 19-21 March 1986 at Fort Collins, CO.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION Unclassified

22a. NAME OF RESPONSIBLE INDIVIDUAL Brian Woodruff, Maj, USAF	22b. TELEPHONE (*Include Area Code*) (202) 767-5025	22c. OFFICE SYMBOL AFOSR/NM

2

COMPUTER SCIENCE AND STATISTICS

PROCEEDINGS OF THE 18th SYMPOSIUM ON THE INTERFACE

Fort Collins, Colorado, March 1986

Editor
THOMAS J. BOARDMAN
Colorado State University
Fort Collins, Colorado, U.S.A.

Assistant Editor
IRENE M. STEFANSKI
American Statistical Association
Washington, D.C., U.S.A.

ASA
1986

AMERICAN STATISTICAL ASSOCIATION, WASHINGTON, D.C.

The papers and discussions in this Proceedings volume are reproduced exactly as received from the authors. None of the papers has been submitted to a refereeing process. However, the authors have been encouraged to have their papers reviewed by a colleague prior to final preparation. These presentations are presumed to be essentially as given at the 18th Symposium on the Interface. This Proceedings volume is not copyrighted by the Association; hence, permission for reproduction must be obtained from the author, who holds the rights under the copyright law.

Authors in these Proceedings are encouraged to submit their papers to any journal of their choice. The ASA Board of Directors has ruled that publication in the Proceedings does not preclude publication elsewhere.

American Statistical Association
806 15th Street, N.W., Suite 640
Washington, DC 20005

PRINTED IN THE U.S.A.

PREFACE

The 18th Symposium on the Interface between Statistics and Computer Science follows in a long-standing series designed to provide a forum for numerical analysts, statisticians, and computer scientists to meet, listen, and discuss topics of mutual interest.

The 18th Symposium was held at the University Park Holiday Inn, Fort Collins, Colorado, on March 19-21, 1986, hosted by Colorado State University.

The registrants reflected the international nature of the Symposium. Among the 300-plus registrants were statisticians, computer scientists, numerical analysts, combinations of the preceding, and others from most states in the U.S., several of the provinces in Canada, and six other countries. The organizers were pleased with the distribution and magnitude of the attendance.

The official cosponsors of the 18th Symposium were Colorado State University, the Statistical Computing Section of the American Statistical Association (ASA), the International Association for Statistical Computing, and the Colorado-Wyoming Chapter of ASA. After many years of discussion, the proceedings are being published by ASA and made available the same year as the Symposium.

On March 19, 1986, two short courses were conducted at the Holiday Inn. Peter Lewis presented material on his "Advanced Simulation and Statistics Package" from 9 a.m. to 4 p.m. At the same time several representatives from TCI Software demonstrated the "T^3 Scientific Word Processing System." Registration began at 6 p.m. and continued during the beginning of the Welcoming Reception/Mixer that was held in the Fountain Court of the hotel. The keynote address by John W. Tukey, entitled "The Interface with Computing: In the Small or In the Large," opened the Symposium on Thursday morning. Thereafter, three invited and one contributed sessions continued on Thursday and Friday. The Holiday Inn staff prepared delightful buffet luncheons on Thursday and Friday, which provided the registrants with ample time to pursue other interests and carry on further discussion over lunch. During the conference eleven firms took the opportunity to exhibit their software or materials (see the list on page v). The exhibit room also served as the coffee break location, thus increasing the traffic to the exhibit area. As one might expect, much of the true interface of information occurred during the informal discussions and demonstrations. Our organizing committee made efforts to provide facilities and opportunities for these activities. We encourage future Symposium organizers to expand on these opportunities.

As chairman of the Symposium, I gratefully acknowledge the help provided me by Jim zumBrunnen, the vice chairman, and Marilyn Lesh, our secretary. Without their encouragement and support, the success of the Symposium would have been in question. Then, too, we must reserve special thanks to the program committee who organized the invited sessions; their efforts to secure speakers and topics made the program outstanding. The committee consisted of: Daniel B. Carr, Pacific Northwest Laboratory; Paula Cowley, Pacific Northwest Laboratory; James Dolby (deceased), Los Altos, California; William Eddy, Carnegie-Mellon University; Dennis Friday, National Bureau of Standards; Richard Jones, University of Colorado; William Kennedy, Iowa State University; John Nash, BYTE and University of Ottawa; Wesley Nicholson, Pacific Northwest Laboratory; Robert Schnabel, University of Colorado; G.W. "Pete" Stewart, University of Maryland; Paul Switzer, Stanford University; Mike Tarter, University of California, Berkeley; Bob Teitel, Teitel Data Systems; John Tukey, Princeton University; and Paul Velleman, Cornell University.

Local arrangements were carried out by Colorado State University's Office of Conference Services. Jill Lancaster and her staff did an outstanding job with all of the many details in running our Symposium. The staff at the Holiday Inn under the able leadership of Jane Folsom made everyone feel welcome. As I have learned from Dr. W. Edwards Deming, when people know what their job is and what their customers expect, it is not uncommon to find that quality service can result. The hotel and Conference Service staff knew our needs and exceeded our expectations. We praise them for their efforts.

As indicated on page *v* of these proceedings, financial support for the Symposium was made possible by three organizations. When bringing a diverse group of people together, travel and related expenses must be covered for some of the participants. This support was essential. In addition, this year for the first time we offered student fellowships for graduate students in statistics or computer science. I hope that this model will be continued in the future.

I offer thanks to the many other Colorado State University staff, students, and faculty members who helped to make this Symposium possible. I would like to thank Randall Spoeri, Associate Executive Director of ASA, whose advice and counsel was sought often and was always helpful. Finally on behalf of all of the contributors to this volume, I wish to thank Irene M. Stefanski, ASA Publications Manager, who served as Assistant Editor for these proceedings, and Debra B. Shapiro, Publications Assistant.

Thomas J. Boardman,
Editor

Financial Supporters of the 18th Interface Symposium

U.S. Office of Naval Research
Air Force Office of Scientific Research
Colorado State University

Cosponsoring Organizations

American Statistical Association
Statistical Computing Section
Colorado–Wyoming Chapter

International Association for Statistical Computing

Exhibitors

American Mathematical Society
Aptech Systems, Inc. (GAUSS Software)
BMDP Statistical Software, Inc.
Chapman & Hall
Conceptual Software, Inc.
CSU Statistics Department
IMSL, Inc.
Minitab
Numerical Algorithms
P–STAT, Inc.
SAS Institute Inc.
SYSTAT, Inc.
TCI Software
Wadsworth Brooks

AVAILABILITY OF PROCEEDINGS

15, 16, 17th (1983, 84, 85)	North Holland Publishing Company Amsterdam The Netherlands
13, 14th (1981, 82)	Springer-Verlag New York, Inc. 175 Fifth Avenue New York, NY 10010
12th (1979)	Jane F. Gentleman Statistics Canada Social & Economic Statistics Division Coats Building, 23rd Floor Tunney's Pasture Ottawa, Ontario Canada K1A 0T6
11th (1978)	Institute of Statistics North Carolina State University P.O. Box 5457 Raleigh, North Carolina 27650
10th (1977)	Statistical Engineering Laboratory Applied Mathematics Division National Bureau of Standards U.S. Department of Commerce Washington, DC 20234
9th (1979)	Prindle, Weber, and Schmidt, Inc. 20 Newbury Street Boston, Massachusetts 02116
8th (1975)	Health Sciences Computing Facility, AV-111 Center for Health Sciences University of California Los Angeles, California 90024
7th (1974)	Statistical Numerical Analysis and Data Processing Section 117 Snedecor Hall Iowa State University Ames, Iowa 50010
4, 5, 6th (1971, 72, 73)	Western Periodicals Company 13000 Raymer Street North Hollywood, California 91605

The 19th Symposium on the Interface will be held March 9-11, 1987, in Philadelphia, Pennsylvania. The chair will be Richard Heiberger of Temple University.

CONTENTS :

CONTENTS

CONTENTS

CONTENTS

CONTENTS

CONTENTS

CONTENTS

KEYNOTE ADDRESS

The Interface with Computing: In the Small or In the Large?
John W. Tukey, Princeton University

THE INTERFACE WITH COMPUTING: IN THE SMALL or IN THE LARGE?*

John W. Tukey, Princeton University, Princeton, NJ 08544

If "the interface" is to be a real interface, it needs to discuss problems that are real on both sides. If statisticians are to help the practice of analyzing data to even approach an economic balance between the decreasing costs of computation and the advancing salaries of statistical consultants (and of statistical theorists), they must ask their computer - - and soon their workstation - - for much larger and more important computing tasks.

Either of these points leave statisticians seeking to formulate problems very much larger than we have been accustomed to consider - - large enough to be a real strain for computing-science specialists to consider (and often too hard for statisticians alone.)

We must also strain the statisticians! Problems that will challenge the computer scientists will not be found by sticking to simplicity, where simplicity is inappropriate! I shall return, shortly, to this challenge to statisticians, but before I do, let me illustrate a few larger problems of computation.

There are a variety of major problem-classes, including factorial data, and regression, where minimizing the sum of absolute (values of) residuals, conveniently "minimizing the L^1-norm" is a useful intermediate step to what we really want to do. (Simplicity is already disappearing). When we are making a linear fit, the minimum-L^1 fit is rarely unique.

Rather there is a convex subset of parameter space, bounded by hyperplanes, throughout which the L^1-norm is constant at its minimum. In moving toward uniqueness, it is natural to first focus on the vertices (the corner points) of this convex set, of which there may be many. (Each vertex turns out to correspond to a fit with many exact-zero residuals.) It might be helpful to get a list of all these vertices; it might be even more helpful to get a list of those in a subset which also makes extreme some second criterion (maximizing the L^1-norm is often handy). Work by Peter Rousseeuw in the linear regression case, and by Eugene Johnson in the factorial analysis case has made important progress, but I suspect our computing science colleagues can help us further, if only about how to do the generalizations to non-linear fits.

Configural polysampling (Tukey 1986), in which different weighting schemes allow a single set of sample configurations to be honestly taken as representing each of a few - - not just one - - tastefully selected parent distributional situations, is today our only effective way to learn about the possibilities of robustness in real, finite samples. To make things work we need to evaluate a few more-dimensional integrals for each of a few parent situations, doing this for each of several hundred configurations. This can easily mean 10,000 multidimensional quadratures, of smooth but somewhat nasty integrands!

Location and scale together call for two dimensional integrals; fitting straight lines calls for three-dimensional ones; regressions with 3 or more coefficients call for numerical integration in even more dimensions. At the moment, by working hard, one can do a probably satisfactory job in two-dimensions, and dream of doing the same in three dimensions.

It is not satisfactory to "compute to death" such a problem by adopting integration procedures that evaluate the integrand at many, many points! In four dimensions, 100,000 evaluations, at 100 or so arithmetic operations per evaluation would only be 10 million or so arithmetic operations per integral, taking a minute on a reasonably available computer. For one integral this is fine; for 10,000 integrals its a week's work and one more dimension *will surely ruin us!* What to do?

If we only want a few millions of arithmetic operations, we can have them easily. One thing we all ought to want to do with them is to have our workstations busily chewing over the last set of data we put in, whenever it is not doing something we explicitly asked for, to see what it can find. What sort of program should we write to do this? What sort of "cognostics" should it look at? How should it schedule its attention to different aspects (of a single body of data)? What sorts of reports should it make? How should it assess urgency of reporting? These are sample questions. Looking at them we can see the need for combining both real data-analysis experience and sound complex-computing-system experience, if these questions are going to be answered effectively and well.

As a final example, think about the facilitation of preparing and modifying what statisticians are happy to call "expert systems". (Some computer scientists think our systems are too simple for such a title.)

Many of us are going to at least need aid in preparing and modifying such systems. We all hope the tasks will be simpler, nor harder. Within what framework should our systems be built and modified for this to be so? We can have ideas about this today, but it is almost certainly too soon to freeze one or more standard frameworks - - however it is not too soon to start thinking about what will be important when we start to freeze frameworks!

I turn now to the challenge more specifically aimed at statisticians: The

"back side of omnicompetence"

- - which some might vender, less delicately, as the

"back side of incompetence".

I have long complained of the heresy/fallacy of omnicompetence, of the claim that "just tell me exactly what arithmetic has been done to the data, then I will know what your results mean!" In other words: "If we look at a detailed description of some arithmetic that can be applied to data, it is easy to understand just what that arithmetic does!" As most of you know: no one of us, here or elsewhere, can correctly make such a claim. To understand just what a given set of arithmetic does - - even so simple arithmetic as taking the arithmetic mean - - is not easy (in that case, we were still learning a hundred and fifty years after Gauss).

I have been slow to realize that omnicompetence has two sides, the one just mentioned, and an even more threatening "back side" which can be recognized/identified by these two statements: "Don't bother me with procedures that are not simple and transparent!" (If they weren't, I might not be omnicompetent!). "I admit data analysis has to be inductive, but understanding how to select ways to do it should be - - nay, must be - - deductive"! (Else I might not be omnicompetent.) The back side is the more threatening one, since its acceptance would keep us from polishing our methods to make them work better, whenever that polishing would make either arithmetic or natural heuristics more complicated, and would confine us to methods whose advantages we could deduce, thus ruling out, in particular, heuristically-based suggestions whose performance has been validated by simulation. It would mean tying our hands, keeping us far from doing as well as we could - - in the interest of having a simple, relatively teachable account of how our methods got that way. *How they got that way, NOT how they perform!*

The implicit assumption is that we are trying to teach how the then-current classic methods came about and can be extended, rather than teaching how they behave and when to use them.

Fred Mosteller and I have worked a lot together. I admire his teaching skills; so I asked myself how one could approach such problems in a way that might satisfy both of us. One possibility seems to be a two-layer approach, combining a very simple, easy-to-follow procedure for understanding with a warning that that procedure was usually not good enough and you ought to do better, as by using the following canned program, which has thus-and-so properties. When I tried this on Fred, he said that he and some colleagues were already doing something close to this in a non-statistical field. So maybe it is a good idea.

There was time when it was felt to be the kiss of death for a new statistics book to call it "a cookbook". I am saying that this is no longer reasonable or wise. Who among you would like to throw out all cookbooks from your household, and replace them by a text entitled "Principles of Culinary Chemistry"? Do you think such a change would make your meals taste better? It's a pretty good idea to stick to "Fannie Farmer"!

Let us also, for a moment, compare ourselves with analytical chemists - - a respectable profession of comfortable antiquity. Analytical chemists live in a real world, they take the occurrence of "interferences" seriously. They do not expect to analyze for a single particular substance in the same single way, when that substance occurs together with very different sets of other substances. Analyses in in one "matrix" - - in urine, for example - - need not be like analyses in another matrix, either in blood or in distilled water. Many details of the analytical procedures are important, for most of them there are heuristics, for some there are none. The test of a good procedure is how well it works, not how well it is understood. No one believes that, even from a highly detailed knowledge of chemistry, it is today possible to deduce the operations by which we should analyze for a particular substance.

If we are to do as well as the analytical chemists, we will have to disavow simplicity and deducibility as absolute standards.

We can retain simplicity and deducibility as secondary goals - - as ways of choosing among otherwise equally qualified candidates; as reasons for slight modulations of very good procedures, modulations keeping nearly all - - but perhaps not all - - of the highest performance we can otherwise get. (I would always give up 1% of efficiency for simplicity - - and 0.4% for deducibility - - partly for the sake of the procedure itself, but mainly because I would hope to do better with analogs of the procedure in more general or more complicated problems.) But we dare not accept simplicity and deducibility as major goals.

As I look back over more than four decades of experience with both practice and theory, I have to conclude that almost every major step with which I have been concerned offers an example or two of the back side of omnicompetence. You may doubt this, so I shall talk briefly about four examples.

First, multiple comparisons. The first serious proposal about multiple comparisons used the studentized range, for interesting reasons that do not concern us here. Shortly thereafter, F-based methods became temporarily popular, perhaps because they seemed simpler, and more closely related to previous practice. It took a long time before it was generally recognized that they spent error rate where it either wasn't needed or couldn't be effective.

It took another decade or so to realize that there were situations where much more complicated multiple comparisons procedures (like those of Ramsey, or even those of Peritz) were needed, that there were several kinds and slices of multiple comparisons problems, for each of which different procedures, not necessarily simple one, were appropriate (cp. Braun and Tukey 1983c).

Spectrum analysis, of the kind once called modern, also started for unexpected reasons. The great thing that made its techniques fly was giving up the simple picture of energy at a *few isolated frequencies*, which was altogether too simple, and had led to procedures that were inadequate for - - and misleading when applied to - - many real-world problems.

Omnicompetence has returned to spectrum analysis recently, in the form of overbelief in maximum entropy estimates, accompanied in some minds by ideas equivalent to "if you've calculated some moments, any fit you take seriously must match these moments exactly!". For those of us who have thrown off the shackles of the arithmetic mean - - itself an example of moment matching - - this attitude hardly seems either reasonable or wise. (There are places where maximum entropy estimates will serve us well, not for esoteric reasons, but because they can be seen to perform well there.)

Next, let us turn to robust/resistant techniques. Here there was a history of many decades of simple ideas: ideas like rejection rules and trimmed means. The Princeton Robustness Study (Andrews, et al 1977e), and its follow-on waves, looked at several hundred individual estimates and several thousand kinds of linear combinations of these estimates. Out of this non-simplicity came the biweight (with a tuning constant of about 7), for which there is still no deductive approach, but for which experimental sampling - - both in its "swindled" form as Monte Carlo and in its even more sophisticated form as configural polysampling - - has shown the high quality of its behavior.

Recently, I have seen a further stage of the same other side. A paper showing how to fit a straight line better - - where "better" was demonstrated by experimental sampling - - in the face of very uncomfortably non-constant variance revolted referees and editor because the procedure was "too complicated". If we are to do well in complicated problems, we are almost sure to start with complicated solutions, which we may later learn to simplify - - but probably only to a limited extent.

Let me turn finally to clustering, where there are reputed to be review articles with thousands of references - - almost every one starting with some simple procedure, possibly deriving some conceivably interesting possibilities, and taking no thought at all for modification or fine tuning.

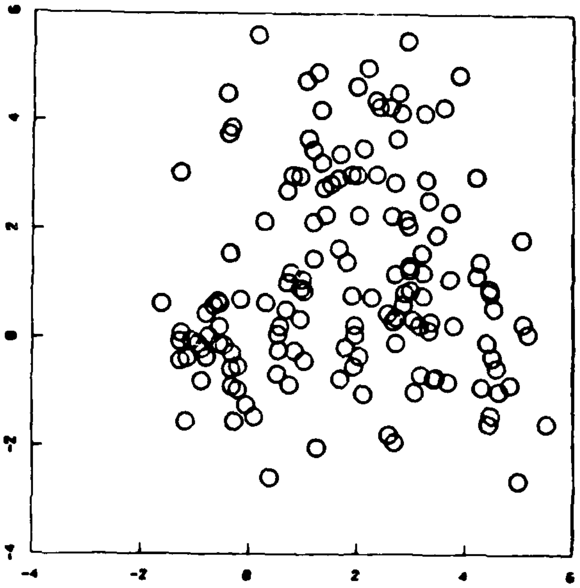
The implied attitude is it seems to me "we can always invent the wheel (even if ours is square, we don't have to modify it)"!

Recently Katherine Hanson and I have been trying the opposite approach: starting with a very simple test bed, and a somewhat more plausible initial approach, how can we modify and adjust and complicate the procedure until its performance at least comes close to that of the human eye.

Let me show you a couple of pictures, pictures that are real and typical for a special situation, but possibly somewhat misleading, because tuning for a wider variety of situations may force us to degrade performance in this special case.

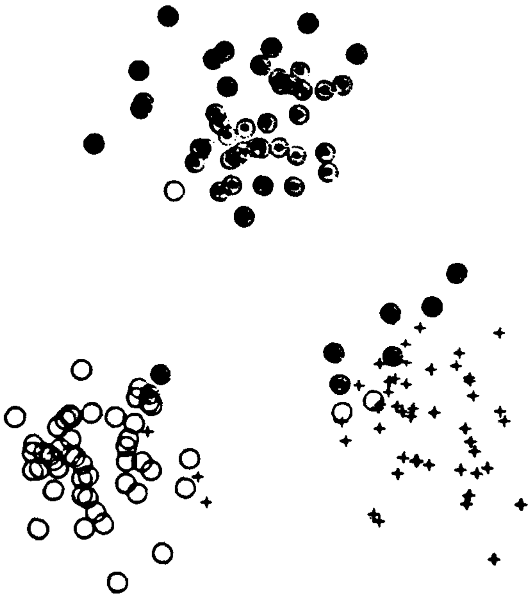
First a challenge, a picture of 150 undistinguished points. Some of you may be able to see 3 concentrations, some may guess that this is a mixture of samples, of nearly equal size, from 3 circular Gaussian populations, populations that clearly interpenetrate one another quite a little.

The challenge - - 150 undistinguished points



Now see what a carefully tuned, fully automatic procedure - - which knows it is looking for 3 clusters, but makes no explicit use of Gaussianity, or of isotropy, or of equal sample size, can do! Here we have distinguished the 3 samples by

**The response: segmentation (artificially separated)
based on 150 undistinguished points and "find 3"**



14 misclassifications, only a few more than for population based discriminants (8)

using different plotting characters - - and also separated the three "clusters" from one another - - so we can see both the overlap of the samples and the performance of the technique.

As this last example suggests, the eye-brain combination is not simple If we are to make AI - - here Automatic Insight - - competitive with human processes, we cannot expect to build effective AI on very simple rules and then trust to millions billions of CPU cycles to get the right answer.

Large parts of the AI profession are badly infested with their version of the same "back side" The combination of very simple elements and very massive computer processing is unlikely to provide either efficient or even effective support for human purposes We shall need to choose subtle basic elements and to combine them creatively - - only then can we make good use of all those CPU cycles.

If all my life I and a few others had taken the "back side of omnicompetence" seriously, there would have been long delays in appearance and use of effective multiple comparisons, effective spectrum analysis, effective straight-line fitting under very difficult circumstances, and, I hope, effective clustering.

I urge each of you to purge yourself of the "back side of omnicompetence" as thoroughly as you can!

REFERENCES**

Andrews, D. F., Bickel, P. J., Hampel, F. R., Huber, P. J., Rogers, W. H., and Tukey, J. W. (1972e). ***Robust Estimates of Location: Survey and Advances,*** Princeton University Press, Princeton, 373 pp.

Braun, H. I. and Tukey, J. W. (1983c). "Multiple comparisons through orderly partitions: the maximum subrange procedure," *Principals of Modern Psychological Measurement: A Festschrift for Frederic M. Lord,* Lawrence Erlbaum Associates, Inc., Hillsdale, New Jersey, 55-65.

Tukey, J. W. (1986). "Configural polysampling," to appear in *SIAM Review.*

*Prepared in part in connection with research at Princeton University sponsored by the Army Research Office (Durham).

**Letters match those in the bibliographies in J. W. Tukey's *Collected Works,* Wadsworth, Monterey, CA.

PARALLEL ALGORITHMS—TUTORIAL

Organizer: *G.W. Stewart, University of Maryland, College Park*

Communication in Parallel Algorithms: An Example
G.W. Stewart, University of Maryland, College Park

Matrix Computations, Signal Processing, and Systolic Arrays
Franklin T. Luk, Cornell University

COMMUNICATION IN PARALLEL ALGORITHMS: AN EXAMPLE

G. W. Stewart
University of Maryland at College Park[1]

1 Introduction

Although the problem of distributing computations over networks of processors has received a great deal of theoretical attention, only now, as commercial systems have become available, are the practical limitations of parallel algorithms becoming apparent. In particular, communication costs can render an otherwise attractive algorithm unsatisfactory. It is therefore necessary to have analyses of parallel algorithms that indicate under what conditions they will perform effectively.

The purpose of this note is to describe and analyze a parallel algorithm for computing the QR factorization of an $n \times p$ matrix X (for applications of this factorization see [1, Ch.9]). The algorithm is designed to run on a ring of r processors that communicate by message passing. In the next section, we will describe the sequential algorithm and its numerical properties. The parallel version of the algorithm will be described in §3, and its analysis given in §4. The analysis suggests some modifications of the algorithm, which are sketched in §5.

2 The Sequential Algorithm

Let X be an $n \times p$ matrix with $n \geq p$. Then X can be factored in the form

$$X = QR, \qquad (2.1)$$

where

$$Q^{\mathrm{T}}Q = I \qquad (2.2)$$

and R is upper triangular. From (2.1) and (2.2) it follows that

$$A \overset{\mathrm{df}}{=} X^{\mathrm{T}}X = R^{\mathrm{T}}R; \qquad (2.3)$$

that is, R is the Cholesky factor of the cross-product matrix A. This suggests the following three step algorithm for computing Q and R.

1. Compute A from (2.3).
2. Compute the Cholesky factor R of A.
3. Partitioning X and Q by rows in the form

$$X = \begin{pmatrix} x_1^{\mathrm{T}} \\ x_2^{\mathrm{T}} \\ \vdots \\ x_n^{\mathrm{T}} \end{pmatrix} \qquad Q = \begin{pmatrix} q_1^{\mathrm{T}} \\ q_2^{\mathrm{T}} \\ \vdots \\ q_n^{\mathrm{T}} \end{pmatrix},$$

solve the systems

$$q_i^{\mathrm{T}}R = x_i^{\mathrm{T}} \qquad (i = 1, 2, \ldots, n). \qquad (2.4)$$

for the rows of Q.

From a computational point of view, this algorithm is quite satisfactory. The formation of A, or rather its upper half, requires about $np^2/2$ floating-point additions and multiplications; the formation of R requires $p^3/6$; and the solution of the system (2.4) requires $np^2/2$. Thus for large n, the entire algorithm requires about np^2 floating-point additions and multiplications.

From a numerical point of view, the algorithm is less satisfactory. On the positive side, because the q_i's are generated as solutions of (2.4), the method has a backward rounding-error analysis. Specifically, if the computations are performed in t-digit decimal arithmetic, then there is a matrix E of order $10^{-t}\|X\|$ such that

$$QR = X + E. \qquad (2.5)$$

[1] Department of Computer Science and Institute for Physical Science and Technology. This work was supported in part by the Air Force Office of Sponsored Research under Grant AFOSR-82-0078.

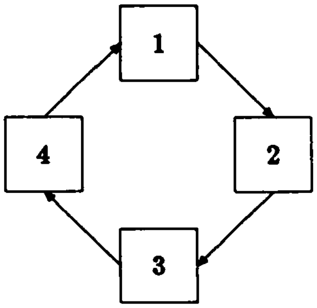


Figure 1: A Ring of Four Processors

Thus little information about X is lost in the passage to Q and R.

However, the columns of Q may fail to be orthonormal; that is, (2.2) may fail to hold—even approximately. If the columns of X are scaled so that they have norm one, then this phenomenon occurs precisely when R^{-1} is large; for in this case the system (2.4) will be ill conditioned, and Q will be inaccurately determined. This means that we can at least recognize the problem when it occurs by applying a condition estimator to R [1, Ch.1].

Moreover, the problem admits a fix. For if we apply the algorithm again to Q, we will obtain a matrices P and S, with S upper triangular such that

$$PS = Q + F \tag{2.6}$$

If (2.6) and (2.5) are combined, the result is

$$PSR = QR + FR = X + E + FR.$$

Thus the factorization $P(SR)$ also reproduces X. Usually P will have columns orthogonal to working accuracy. If not, the reorthogonalization can be repeated.

3 Parallel Implementation

The algorithm sketched above has a natural implementation on a ring consisting of r processors. Such a ring is shown for $r = 4$ in Figure 1. Here we shall suppose that communication is clockwise, as indicated by the arrows.

```
$A_i = X_i^T X_i$
$A = 0$
for $k$=1 to $r$ loop
    $A = A + A_i$
    send($A_i$)
    receive($A_i$)
end loop
compute $R$
solve $Q_i R = X_i$
```

Figure 2: Program for Processor i

The idea of the implementation is simple. The matrix is partitioned in the form

$$X = \begin{pmatrix} X_1 \\ X_2 \\ \vdots \\ X_r \end{pmatrix}, \tag{3.1}$$

where each block has roughly

$$m = \frac{n}{r}$$

rows, and block X_i is assigned to processor i. Each processor initially computes

$$A_i = X_i^T X_i.$$

The A_i are then circulated around the ring. As they pass, the processors add them together, so that at the end each processor has a copy of the cross product matrix $A = \sum_{i=1}^{n} A_i$. It is then a simple matter for each processor to compute R and solve the systems (2.4) to form $Q_i = X_i R^{-1}$, where Q_i is the ith block of Q in a partitioning conformal with (3.1).

Figure 2 contains a program implementing this algorithm. In it the array A_i is used both to hold the initial A_i computed by the processor and to hold the A_i's from other processors as they circulate around the ring. The function send sends a block of data (in this case whatever is currently in A_i) to the next processor. The function `receive` informs the system where to place incoming data.

An attractive feature of this algorithm is that each processor ends up with a copy of R, from which it can check whether a reorthogonalization step is necessary. Since each processor will reach the same conclusion, the reorthogonalization can start forthwith without any initial communication among the processors.

4 Analysis of the Algorithm

We now turn to the analysis of the parallel algorithm. The time required to complete the algorithm may be divided into two parts: the time devoted to computing and the time devoted to communication. Let us look at the computing time first.

We shall assume that it requires time α to perform an addition and a multiplication in the program of Figure 2. As is customary in this kind of analysis, α includes all the indexing and looping overhead in the algorithm, so that it will be considerable greater than the time for the bare arithmetic operations.

The costs of the various computations are summarized below.

$X_i^T X_i$	:	$\frac{1}{2}mp^2\alpha$
Summing	:	$\frac{1}{2}rp^2\alpha$
R	:	$\frac{1}{6}p^2\alpha$
Q_i	:	$\frac{1}{2}mp^2\alpha$

Since $X_i^T X_i$ is symmetric, we need only compute half of it, which accounts for the factor of 1/2 in the first item. Summing these items and making the substitution $m = n/r$, we obtain the total computing time

$$T_a = \left[\frac{np^2}{r} + \frac{1}{2}rp^2 + \frac{1}{6}p^2 + \frac{1}{2}mp^2\right]\alpha. \qquad (4.1)$$

For the communication time, we shall assume that the send-receive sequence in the program requires a fixed setup time σ which is independent of the length of the message. Thereafter, data is transmitted at a rate of τ^{-1} items per unit time. Since only half of the A_i's are being transmitted, the total communication time will be

$$T_c = r\sigma + \frac{1}{2}rp^2\tau. \qquad (4.2)$$

Looking at (4.1) and (4.2), we see that their contributions can be divided into two parts. The first is the term $np^2\alpha/r$, which decreases with the number of processors. Since $np^2\alpha$ is the time taken by the sequential algorithm, we could hardly expect greater speedup.

The second part consists of the other terms, which are linear in r and must ultimately dominate the first part. Although we may initially see a decrease in time as we add processors, we must ultimately come to a point where adding processors actually increases the total time. This point can be computed by adding (4.1) and (4.2), differentiating, setting the results to zero, and solving for r. The result is

$$r_{\min} = \left[\frac{2n}{1 + \bar{\tau} + \frac{2\bar{\sigma}}{p^2}}\right]^{\frac{1}{2}}, \qquad (4.3)$$

where we have introduced the relative parameters

$$\bar{\sigma} = \frac{\sigma}{\alpha}$$

and

$$\bar{\tau} = \frac{\sigma}{\tau}.$$

There are several things to observe about this expression. The number $\sqrt{2n}$ is an upper bound on the number of processors we can profitably use. It comes from the fact that we must sum the A_i's on each processor, and has nothing to do with communication time. However, poor communication can certainly make things worse. For if the time take to transmit numbers is greater than the time to perform arithmetic operations $\bar{\sigma}$ will be greater than one, and will dominate in (4.3). If p is not too small, the relative setup time $\bar{\sigma}$ will play a smaller part; however, on some existing computers with a large startup time even this term can dominate.

5 Revising the Algorithm

The importance of an analysis like the one in the last section is that it can suggest how to modify an algorithm to make it run better and it can indicate features that it is desirable to have in a parallel computer. The algorithm for the QR factorization is a case in point. The dominant terms come from passing the A_i around the ring and from summing

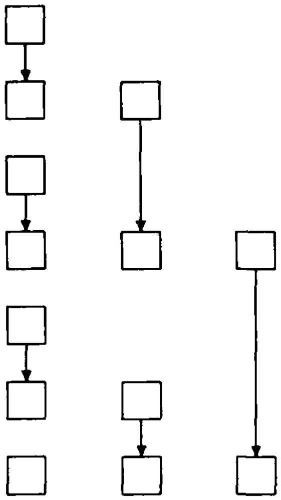


Figure 3: Pairwise Summation

them on the processors. Let us see what we can do about these two roadblocks, beginning with the latter.

Although we cannot calculate A entirely in parallel, we can reduce the amount of computation by a pairwise summation algorithm. Specifically, the odd numbered processors send their A_i's to the next even numbered processors which add them to their own A_i's. Then the even numbered processors pair off to do another pairwise summation—and so on until there is only on processor left, which will of course contain A. An example for $r = 7$ is given in Figure 3. From this it is seen that pairwise summation reduces the time from r to approximately $\log_2 r$.

However, we still have a communications problem; for at the last step we must pass a block of p^2 numbers a distance of about $m/2$ processors. We can do nothing about this unless we are willing to assume something further about the how the processors communicate. One possibility is that the processors can link up and pass numbers bucket brigade fashion from the source processor to the destination processor. If the number of processors in the brigade is d, then the operation will take time

$$(d + p^2 - 1)\beta, \qquad (5.1)$$

where beta is the time to transmit a single number in brigade mode. Although we have not gotten rid of the dependence on r we have reduced its influence by a factor of p^2.

We have been deliberately vague about the details of this algorithm in the hope that the reader will undertake the rewarding task of fleshing it out. The program will be considerably more complicated, as will be the analysis. However, at the end you will have formulas by which you can compare the two algorithms for yourself.

References

1. J. J. Dongarra, J. B. Bunch, C. B. Moler, and G. W. Stewart, *The LINPACK Users' Guide,* SIAM, Philadelphia, 1979.

MATRIX COMPUTATIONS, SIGNAL PROCESSING AND SYSTOLIC ARRAYS

Franklin T. Luk, Cornell University

Abstract

Parallel matrix computing has become an essential part of real-time signal processing. Systolic arrays and associated algorithms for computing the symmetric eigenvalue decomposition, the singular value decomposition and the generalized singular value decomposition are described in detail.

Introduction

Numerical linear algebra is an important tool for modern signal processing practitioners, who must solve systems of linear equations, compute eigenvalues, eigenvectors, singular values and singular vectors(cf. Bromley and Speiser[1]). The necessity that these matrix operations be completed in real time, together with the availability of VLSI/VHSIC technology, has led to the development of special purpose multiprocessor systolic arrays. In this paper we discuss systolic arrays and their associated parallel algorithms for computing the symmetric eigenvalue decomposition, the singular value decomposition and the generalized singular value decomposition.

A given symmetric $n \times n$ matrix A can be diagonalized via a similarity transformation:

$$A = V \Sigma V^T, \qquad (1)$$

where the matrix V is $n \times n$ orthogonal and Σ is $n \times n$ diagonal. The eigenvalue decomposition (EVD) is extensible to the diagonalization of an $m \times n$ ($m \geqslant n$) matrix A. Two different transformations are required to compute the singular value decomposition (SVD):

$$A = U \Sigma V^T , \qquad (2)$$

where the matrices U $(m \times m)$ and V $(n \times n)$ are orthogonal, and the matrix Σ $(m \times n)$ is nonnegative diagonal. For applications and computations of these decompositions see Golub and Van Loan[2] and Dongarra et al.[3]. The SVD can be extended to a simultaneous diagonalization of two real matrices A $(m \times n)$ and B $(p \times n)$ by two orthogonal matrices U $(m \times m)$ and V $(p \times p)$ and a nonsingular matrix X $(n \times n)$:

$$U^T AX = D_A \equiv \mathrm{diag}\left(\alpha_1 , \cdots , \alpha_n\right) \qquad (3)$$

and

$$V^T BX = D_B \equiv \mathrm{diag}\left(\beta_1 , \cdots , \beta_n\right). \qquad (4)$$

The factorization (3-4) is called a generalized singular value decomposition (GSVD). It is useful for solving various constrained and generalized least squares problems, e.g.,

$$\| Ax - b \|_2 = \min$$

subject to

$$\| Bx - d \|_2 \leqslant \gamma .$$

Many systolic arrays have been proposed for these matrix decompositions. Brent and Luk[4] presented a systolic array for computing the symmetric eigenvalue decomposition. SVD arrays are given in Brent and Luk[4], Brent, Luk and Van Loan[5], Finn, Luk and Pottle[6], Heller and Ipsen[7], Luk[8] and Schimmel and Luk[9], and GSVD arrays in Brent, Luk and Van Loan[10] and Luk[11].

The most effective parallel eigenvalue and singular value algorithms for full, dense matrices are of the Jacobi type. Jacobi techniques are easily implementable on mesh-connected processors. Indeed, they were used for finding eigenvalues and singular values on the ILLIAC IV, the first parallel computer (see Luk[12] and Sameh[13]). To compute an $n \times n$ SVD, a parallel Jacobi scheme[5,8] requires n^2 processors and $O(nS)$ time, where S denotes the number of sweeps for convergence. The parameter S is a slowly growing function of n and is conjectured to equal $O(\log n)$[4]. In comparison, the LINPACK[3] SVD procedure requires time $O(n^3)$. Unfortunately, Jacobi-SVD algorithms are applicable only to square matrices. For a rectangular matrix A, an obvious strategy is to first compute its QR decomposition (QRD) of A :

$$A = Q \begin{pmatrix} R \\ 0 \end{pmatrix}, \qquad (5)$$

where the matrix Q is $m \times m$ orthogonal and R $n \times n$ upper triangular, and to then apply the SVD procedure to the square matrix R. This approach is particularly suitable for the case where $m \gg n$ (cf. Chan[14]). However, we need to handle the interfacing of different arrays. To alleviate the problem, Luk[8] suggested one "triangular" processor array for computing both the QRD and the SVD. Subsequently, a new GSVD algorithm implementable on the same array was proposed by Luk[11].

Symmetric Eigenvalue Decomposition

The basic tool in a Jacobi method is the 2×2 plane rotation

$$J(\theta) = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix}, \qquad (6)$$

as the basic problem concerns the diagonalization of a 2×2 matrix by the rotation:

$$J(\theta)^T \begin{pmatrix} p & q \\ q & r \end{pmatrix} J(\theta) = \begin{pmatrix} d_1 & 0 \\ 0 & d_2 \end{pmatrix}. \qquad (7)$$

Suppose $q \neq 0$ (else choose either $\theta = 0$ or $\theta = \pi/2$). It is well known that $t \equiv \tan\theta$ satisfies the quadratic equation:

$$t^2 + 2\rho t - 1 = 0 \,, \tag{8}$$

where

$$\rho = \frac{r-p}{2q} \equiv \cot 2\theta \,. \tag{9}$$

The two solutions to (8) are

$$\begin{aligned} t &= \frac{\mathrm{sign}(\rho)}{|\rho| + \sqrt{1+\rho^2}} \,, \\ \cos\theta &= \frac{1}{\sqrt{1+t^2}} \,, \\ \sin\theta &= t \cos\theta \end{aligned} \tag{10}$$

and

$$\begin{aligned} t &= -\mathrm{sign}(\rho) [\ |\rho| + \sqrt{1+\rho^2}\ ] \,, \\ \cos\theta &= \frac{1}{\sqrt{1+t^2}} \,, \\ \sin\theta &= t \cos\theta \,. \end{aligned} \tag{11}$$

The angle θ associated with (10) is the smaller of the two possibilities; it satisfies $0 \leqslant |\theta| < \pi/4$, whereas the one associated with (11) satisfies $\pi/4 \leqslant |\theta| < \pi/2$. We refer to a rotation through the smaller angle as an "inner rotation" and one through the larger angle as an "outer rotation" (cf. Stewart[15]). The "inner rotation" is chosen in Brent et al.[4,5] and the "outer rotation" in Luk[8]. If the given matrix is diagonal ($q = 0$) then an "inner rotation" means $\theta = 0$ and an "outer rotation" implies $\theta = \pi/2$. In the former case the matrix stays unchanged, whereas in the latter case the eigenvalues are interchanged:

$$\begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} p & 0 \\ 0 & r \end{pmatrix} \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} = \begin{pmatrix} r & 0 \\ 0 & p \end{pmatrix} .$$

By solving an appropriate sequence of 2×2 EVD problems, we compute an EVD of a general $n \times n$ matrix A. The Jacobi transformation is

$$T_{ij} \ : \ A \leftarrow J_{ij}^T \, A \, J_{ij} \,, \tag{12}$$

where J_{ij} is a rotation in the (i,j) plane chosen to annihilate the (i,j) and (j,i) elements of A. The transformation T_{ij} will produce a matrix $\hat{A}$ satisfying

$$\mathit{off}(\hat{A}) = \mathit{off}(A) - 2a_{ij}^2 \,, \tag{13}$$

i.e., the matrix $\hat{A}$ is more "diagonal" than A. The value of (i,j) is determined according to some ordering, to be determined such that all the off-diagonal elements will be annihilated once in any group of $n(n-1)/2$ rotations (called a "sweep"). A well known example is the cyclic-by-rows ordering, illustrated here in the $n = 4$ case:

$$(i,j) = (1,2),(1,3),(1,4),(2,3),(2,4),(3,4).$$

Jacobi methods lend themselves to parallel computations. Brent and Luk[4] developed a square processor array and a "parallel" ordering that allows $\lfloor n/2 \rfloor$ simultaneous rotations. Their new ordering is amply illustrated by the $n = 8$ case:

$$\begin{aligned} (i,j) = \ & (1,2),(3,4),(5,6),(7,8), \\ & (1,4),(2,6),(3,8),(5,7), \\ & (1,6),(4,8),(2,7),(3,5), \\ & (1,8),(6,7),(4,5),(2,3), \\ & (1,7),(8,5),(6,3),(4,2), \\ & (1,5),(7,3),(8,2),(6,4), \\ & (1,3),(5,2),(7,4),(8,6). \end{aligned}$$

Rotation pairs associated with each "row" of the above ordering can be calculated concurrently. We present a parallel Jacobi algorithm for A.

Algorithm EVD.

```
do until convergence
    for each $(i,j)$ according to the "parallel" ordering
        $A \leftarrow J_{ij}^T \, A \, J_{ij}$ .  □
```

By convergence we mean that the parameter $\mathit{off}(A)$ has fallen below some pre-selected tolerance. However, it is difficult to monitor $\mathit{off}(A)$ in the settings of parallel computations. Since convergence is fast (ultimately quadratic) it is a usual practice to stop iterations after a sufficiently large number (say ten) of sweeps. Details on the processor array are given in Brent and Luk[4]. Important points worth emphasizing are that only nearest neighbor connections are required, that broadcasting can be avoided through a staggering of computations, and that one sweep of the algorithm is implementable in time $O(n)$.

Numerical experiments were performed on a VAX-11/780 at Cornell University. Double floating data types were used: each number is binary normalized, with an 8-bit signed exponent and a 57-bit signed fraction whose most significant bit is not represented. The accuracy is thus approximately 17 decimal digits. The results are presented in Table 1. We started with random $n \times n$ matrices whose elements came from a uniform distribution in the interval $(-1,1)$; we stopped when the parameter $\mathit{off}(A)$ had been reduced to 10^{-12} times its original value. The rate of convergence was quadratic, confirming theoretical predictions, and only eight or fewer sweeps were required for $n \leqslant 200$. Empirically we find that $S = O(\log n)$, and there are theoretical reasons for believing this, although it has not been proved rigorously. In practice S can be regarded as a constant (say 10) for all realistic values of n (say $n \leqslant 1000$).

Table 1. Average Number of Sweeps
Required by Algorithm EVD

n	trials	# sweeps
4	5000	2.64
6	5000	3.37
8	2000	3.79
10	2000	4.09
20	1000	4.94
30	1000	5.41
40	1000	5.74
50	1000	5.99
100	500	6.78

One may expect that software (or hardware) for the symmetric eigenvalue problem can be used to solve the SVD problem. For example, we may compute an eigenvalue decomposition of the matrix

$$A^T A \quad \text{or} \quad \begin{bmatrix} O & A \\ A^T & 0 \end{bmatrix}.$$

However, Brent et al.[5] gave detailed explanations why we should *not* approach the SVD problem as a symmetric eigenvalue problem.

Singular Value Decomposition

The basic problem here concerns the diagonalization of a 2×2 matrix by the two rotations $J(\theta)$ and $K(\phi)$:

$$K(\phi)^T \begin{bmatrix} w & x \\ y & z \end{bmatrix} J(\theta) = \begin{bmatrix} d_1 & 0 \\ 0 & d_2 \end{bmatrix}. \qquad (14)$$

A two-stage procedure is adopted. First, find a rotation $S(\psi)$ to symmetrize the matrix :

$$S(\psi)^T \begin{bmatrix} w & x \\ y & z \end{bmatrix} = \begin{bmatrix} p & q \\ q & r \end{bmatrix}. \qquad (15)$$

If $x = y$ we choose $\psi=0$, otherwise we compute

$$\begin{aligned} \rho &= \frac{w+z}{x-y} \equiv \cot\psi , \\ \sin\psi &= \frac{\text{sign}(\rho)}{\sqrt{1+\rho^2}} , \qquad (16) \\ \cos\psi &= \rho \sin\psi . \end{aligned}$$

Second, diagonalize the resulting symmetric matrix:

$$J(\theta)^T \begin{bmatrix} p & q \\ q & r \end{bmatrix} J(\theta) = \begin{bmatrix} d_1 & 0 \\ 0 & d_2 \end{bmatrix}. \qquad (17)$$

Finally, $K(\phi)$ is given by

$$K(\phi)^T = J(\theta)^T S(\psi)^T , \qquad (18)$$

i.e., $\phi = \theta + \psi$. Again, by solving an appropriate sequence of 2×2 SVD problems, we compute an SVD of a general $n \times n$ matrix A. The Jacobi transformation is

$$T_{ij} : A \leftarrow J_{ij}^T A K_{ij} , \qquad (19)$$

where J_{ij} and K_{ij} are rotations in the (i,j) plane chosen to annihilate the (i,j) and (j,i) elements of A. The transformation T_{ij} will produce a matrix $\hat{A}$ satisfying

$$\mathit{off}(\hat{A}) = \mathit{off}(A) - a_{ij}^2 - a_{ji}^2 , \qquad (20)$$

i.e., the matrix $\hat{A}$ is more "diagonal" than A.

Luk[8] proposed a triangular processor array that directly computes an SVD of a rectangular matrix. The associated SVD algorithm has two stages. First, a QR decomposition is computed of A as it is fed into the array. Second, a Jacobi-SVD algorithm is applied to the resultant triangular matrix. The pivot block is restricted to contiguous diagonal elements, so as to preserve the triangular structure of the matrix. This so called "odd-even" ordering is well illustrated by the $n = 8$ case:

$$(i,j) = (1,2), (3,4), (5,6), (7,8),$$
$$1, (2,3), (4,5), (6,7), 8.$$

"Outer rotations" are required to ensure that all off-diagonal elements will be annihilated. Details on the array are presented in Luk[8]. Again the important points concern the nearest neighbor connections, the avoidance of broadcast, and the completion of a sweep in $O(n)$ time. We present here the associated SVD algorithm for an $n \times n$ upper triangular matrix A:

Algorithm SVD.

```
do until convergence
    begin
    { "outer rotations" are required }
    for $i = 1, 3, \cdots (i \text{ odd }), 2, 4, \cdots (i \text{ even })$ do
        $A \leftarrow J_{i,i+1}^T \; A \; K_{i,i+1}$ ;
    end.    □
```

Simulation experiments similar to the ones described in the previous section were performed at Cornell. The only difference was that the initial matrices were upper triangular. The results are presented in Table 2, where we observe that $S = O(\log n)$.

Table 2. Average Number of Sweeps
Required by Algorithm SVD

n	trials	# sweeps
4	1000	2.97
6	1000	3.76
8	1000	4.21
10	1000	4.55
20	100	5.54
30	100	6.09
40	100	6.40
50	100	6.72
100	10	7.56

Generalized Singular Value Decomposition

We now present a parallel GSVD algorithm. Only the simple case where A and B are both square $(n \times n)$ and B is nonsingular is considered. The first direct GSVD procedure was given by Paige[16]. It implicitly applies a Jacobi-SVD algorithm to the matrix $C \equiv AB^{-1}$, and is numerically appealing in that only orthogonal transformations are applied to A and B and that the matrices B^{-1} and C are never explicitly formed. We assume that both matrices A and B are upper triangular (do two preparatory QR decompositions if necessary). Orthogonal transformations U, V and Q are to be determined so that the two resulting matrices $U^T AQ$ and $V^T BQ$ have parallel rows, i.e.,

$$U^T AQ = D \cdot V^T BQ , \qquad (21)$$

where D is some diagonal matrix. Defining the nonsingular matrix $X \equiv B^{-1}V$, we get the desired GSVD:

$$\begin{aligned} V^T BX &= I , \\ U^T AX &= U^T AQ \cdot Q^T X \\ &= D \cdot V^T BQ \cdot Q^T B^{-1} V \\ &= D. \end{aligned}$$

On the other hand, note that

$$U^T (AB^{-1}) V = D. \qquad (22)$$

So the transformations U and V can be obtained via an SVD procedure applied to C. The gist of Paige's method lies in its implicit computation of an SVD of AB^{-1} without explicitly forming the matrices B^{-1} and C.

Luk[11] modified Paige's algorithm for parallel computations by adopting the "odd-even" ordering. A big advantage is that the upper triangular structures of both A and B can be preserved. Now, if both A and B are upper triangular, then so are the matrices B^{-1} and C. As such, the two satisfy these special relations:

$$\begin{aligned} (B^{-1})^{i,i+1} &= (B^{i,i+1})^{-1}, \qquad (23) \\ C^{i,i+1} &= A^{i,i+1} (B^{-1})^{i,i+1}, \end{aligned}$$

where $M^{i,i+1}$ denotes the 2×2 submatrix of M formed by intersecting its ith and $(i+1)$st rows and columns. We have thus proved

$$C^{i,i+1} = A^{i,i+1} (B^{i,i+1})^{-1}, \qquad (24)$$

the key condition for an implicit application of Algorithm SVD to the upper triangular matrix C. We find rotations Y and Z for a 2×2 SVD:

$$Y^T C^{i,i+1} Z = S, \qquad (25)$$

where S is diagonal. Then

$$Y^T A^{i,i+1} = S \cdot Z^T B^{i,i+1}, \qquad (26)$$

i.e., the two rows of $Y^T A^{i,i+1}$ and $Z^T B^{i,i+1}$ are parallel. We can thus find one rotation W to triangularize both matrices (cf. Paige [16]).

How do these transformations affect the two $n \times n$ upper triangular matrices A and B? We have

$$\begin{aligned} A &\leftarrow U_{i,i+1}^T \, A \, Q_{i,i+1}, \qquad (27) \\ B &\leftarrow V_{i,i+1}^T \, B \, Q_{i,i+1}, \end{aligned}$$

where $U_{i,i+1}$, $V_{i,i+1}$ and $Q_{i,i+1}$ denote appropriate $n \times n$ rotations in the $(i, i+1)$-plane. Note that both matrices $U_{i,i+1}^T A$ and $V_{i,i+1}^T B$ have only one non-zero subdiagonal element each, in the $(i+1, i)$-position. These two extraneous elements are annihilated by the same rotation $Q_{i,i+1}$, that restores both A and B to triangular forms. Here is a parallel GSVD algorithm[11] for upper triangular A and B:

Algorithm GSVD.

```
do until convergence
    for $i = 1, 3, \cdots (i \text{ odd}), 2, 4, \cdots (i \text{ even})$ do
        begin
        { $U_{i,i+1}$ and $V_{i,i+1}$ are "outer rotations" }
        determine $U_{i,i+1}$ and $V_{i,i+1}$ to
        annihilate $c_{i,i+1}$ and $c_{i+1,i}$ ;
        $A \leftarrow U_{i,i+1}^T A$;   $B \leftarrow V_{i,i+1}^T B$;
        find $Q_{i,i+1}$ to zero out $a_{i+1,i}$ and $b_{i+1,i}$ ;
        $A \leftarrow AQ_{i,i+1}$;   $B \leftarrow BQ_{i,i+1}$
        end.    □
```

By convergence we mean that the rows of A and B become parallel according to some predetermined measure. Algorithm GSVD is easily implementable on the triangular QRD-SVD array of Luk[8]. We compute initial QR decompositions of both A and B as they are fed into the array. The SVD of $C^{i,i+1}$ and the triangularization of both $A^{i,i+1}$ and $B^{i,i+1}$ are performed in parallel on the processor array in a straightforward manner[8]. The significant fact is that a GSVD can be computed in time $O(nS)$.

Addenda

Real time signal processing is an exciting, new research area. We have described two different processor arrays for finding eigenvalues and singular values. There are plenty of open problems that await satisfactory solution. Two important examples are data partitioning (cf. Brent et al.[5] and Schreiber[17]) and fault tolerance (cf. Huang and Abraham[18], Jou and Abraham[19], Luk[20] and Luk and Park[21]).

Acknowledgements

This work was supported in part by the Office of Naval Research under contract N00014-85-K-0074.

References

1. K. Bromley and J.M. Speiser, "Signal Processing Algorithms, Architectures, and Applications," *Tutorial 31, SPIE 27th Annual Internat. Tech. Symp.*, San Diego, 1983.
2. G.H. Golub and C.F. Van Loan, *Matrix Computations*, The Johns Hopkins University Press, Baltimore, 1983.
3. J.J. Dongarra, C.B. Moler, J.R. Bunch and G.W. Stewart, *LINPACK Users' Guide*, SIAM, Philadelphia, 1979.
4. R.P. Brent and F.T. Luk, "The solution of singular-value and symmetric eigenvalue problems on multiprocessor arrays," *SIAM J. Sci. Statist. Comput.*, 6 (1985), pp. 69-84.
5. R.P. Brent, F.T. Luk and C.F. Van Loan, "Computation of the singular value decomposition using mesh-connected processors," *J. VLSI Computer Systems*, 1 (1985), pp. 242-270.
6. A.M. Finn, F.T. Luk, and C. Pottle, "Systolic array computation of the singular value decomposition," *Proc. SPIE Vol. 341, Real Time Signal Processing V* (1982), pp. 35-43.
7. D.E. Heller and I.C.F. Ipsen, "Systolic networks for orthogonal equivalence transformations and their applications," *Proc. 1982 Conf. on Advanced Research in VLSI*, MIT, Cambridge (1982), pp. 113-122.
8. F.T. Luk, "A triangular processor array for computing singular values," *J. Lin. Alg. Applics. (Special Issue on Parallel Comput.)*, to appear.
9. D.E. Schimmel and F.T. Luk, "A new systolic array for the singular value decomposition," *Proc. Fourth MIT Conf. on Advanced Research in VLSI*, (1986), pp. 205-217.
10. R.P. Brent, F.T. Luk, and C. Van Loan, "Computation of the generalized singular value decomposition using mesh-connected processors," *Proc. SPIE Vol. 431, Real Time Signal Processing VI* (1983), pp. 66-71.
11. F.T. Luk, "A parallel algorithm for computing the generalized singular value decomposition," *J. Parallel Distrib. Comput.*, 2 (1985), pp. 250-260.
12. F.T. Luk, "Computing the singular-value decomposition on the ILLIAC IV," *ACM Trans. Math. Softw.*, 6 (1980), pp. 524-539.
13. A.H. Sameh, "On Jacobi and Jacobi-like algorithms for a parallel computer," *Math. Comput.*, 25 (1971), pp. 579-590.
14. T.F. Chan, "An improved algorithm for computing the singular value decomposition," *ACM Trans. Math. Softw.*, 8 (1982), pp. 72-83.
15. G.W. Stewart, "A Jacobi-like algorithm for computing the Schur decomposition of a non-Hermitian matrix," *SIAM J. Sci. Statist. Comput.*, 6 (1985), pp. 853-864.
16. C.C. Paige, "Computing the generalized singular value decomposition," *SIAM J. Sci. Statist. Comput.*, 7 (1986), to appear.

17 R. Schreiber, "On the systolic arrays of Brent, Luk, and Van Loan," *Proc. SPIE Vol. 431, Real-Time Signal Processing VI* (1983), pp. 72-76.

18. K-H. Huang and J.A. Abraham, "Algorithm-based fault tolerance for matrix operations," *IEEE Trans. Comput.*, C-33 (1984), pp. 518-528.
19. J-Y. Jou and J.A. Abraham, "Fault-tolerant matrix operations on multiple processor systems using weighted checksums," *Proc. SPIE Vol. 495, Real-Time Signal Processing VII* (1984), pp. 94-101.
20. F.T. Luk, "Algorithm-based fault tolerance for parallel matrix equation solvers," *Proc. SPIE Vol. 564, Real-Time Signal Processing VIII* (1985), pp. 49-53.
21. F.T. Luk and H. Park, "Fault-tolerant matrix triangularizations on systolic arrays," Report EE-CEG-86-2, School of Electrical Engineering, Cornell University, 1986.

PARALLEL ARCHITECTURE IN STATISTICS

Organizers: *William Eddy, Carnegie–Mellon University;*
Donald E. McClure, Brown University

Parallel Architecture: A Tutorial for Statisticians
William F. Eddy, Carnegie–Mellon University

Discrete–Finite Inference on a Network of Vaxes
William F. Eddy, Mark J. Schervish, Carnegie–Mellon University

PARALLEL ARCHITECTURE

A TUTORIAL FOR STATISTICIANS

William F. Eddy, Carnegie-Mellon University

1. INTRODUCTION AND SUMMARY

For a number of years computer science research has studied the notions of parallel processing from both a hardware and a software point of view. Much of this research has had little practical impact up to now. However, this situation is rapidly changing and the next few years will make systems which incorporate the notions of parallel processing much more widely available to statisticians. The Department of Statistics at Carnegie-Mellon University has experience using vector processors, has an attached processor on one of the department VAXes, and has experience using a network of processors in a data-flow system. This experience has led us to believe that the time is ripe for statisticians to begin a major use of parallel computation.

The purpose of this article is to provide a brief introduction to the various notions of parallelism and their realization in various hardware architectures. Roughly speaking the article is divided into two broad parts: the first is an introduction to the terminology and programming notions needed in concurrent programming; the second is a short review of the particular classes of parallel architecture that we have found useful.

The sections on concurrent programming will introduce the two key notions that are necessary for an understanding of the parallel execution of programs:

- Interprocess Communication

- Process Synchronization.

The sections on hardware architectures will discuss the three broad classes of machines which we have used and which we believe can be generally useful for statisticians

- Vector Processors

- Attached Processors

- Networks of Processors.

In the references a number of papers are listed that are not explicitly referred to in the text; we have found these papers very helpful in the organization of our thinking about parallel computation.

2. SPECIFICATION OF PARALLELISM

In order to think clearly about the specification of parallelism it is necessary to remember the *fundamental concepts of sequential programming*. The single most important concept, for our purposes here, is the notion of a sequential process. A *sequential process* is the actual execution of a sequential program; a *sequential program* specifies the sequential execution of a list of statements.

A *concurrent program* specifies the execution of more than one sequential program that can be executed as *parallel* or *concurrent processes*. These sequential processes, executing in parallel, do not necessarily have to reside in separate memories nor do they have to be executed on distinct processors. Thus, parallel processes can be implemented on real machines in one of three ways.

- The processes can be *multiprogrammed*, so that they share the memory of a single processor. This is, of course, precisely what a time-sharing system does.

- The processes can share a single memory but be run on separate processors. This is usually referred to as *multiprocessing* and obviously requires specially-designed hardware.

- The processes can be *distributed* to a number of individual processors, each having its own memory, connected by a communications network.

As mentioned above the two basic problems that concurrent programs (or rather the programmers who create the programs) face are process synchronization and interprocess communication. Execution of a concurrent program can be represented by an acyclic directed graph where each node represents a process and each directed arc indicates that the process at its end node cannot execute *until the process at its source* node has completed; this graph determines the process synchronization and is called a *process flow graph*. Communication between concurrent processes can be represented by an acyclic directed graph where each node represents a read or write of data by a process and a directed arc indicates the transfer of data from the process at its source node to the process at its end node; this graph determines the interprocess communication and is called a *data flow graph*. The graph is made acyclic by making multiple copies of processes which send and receive data more than once.

2.1. Coroutines

One of the earliest notions for the specification of concurrent processes was the idea of *coroutines*. Broadly speaking a coroutine can be thought of as a process implemented as a subroutine; the distinction is that subroutines are initiated by a *call* and terminated by a *return* in a strictly hierarchical way. Coroutines are initiated and terminated by a *resume* statement in a non-hierarchical fashion; except, of course, when they are initiated and terminated in a hierarchical fashion (so that they are actually subroutines). The crucial points to notice are that:

- The *resume* command serves to synchronize the processes in the separate coroutines so that all the processes implemented as coroutines can execute on a single processor with no loss in execution time (except for overhead).

- Only one coroutine can be executing at a time so that the switching between processes that can occur in a multiprogramming environment is completely determined by the programmer.

The only process flow graphs for concurrent programs that can be implemented by coroutines are linear.

2.2. Fork and Join

Another early notion for the specification of concurrent processes was the *fork* statement. The statment specifies that the invoked routine should execute in parallel with the invoking routine. The invoking routine can execute a *join* statement to force synchronization with the completion of the invoked routine. The fork-join mechanism is a very powerful tool for implementing parallel processing and is a feature of the Unix operating system. its only major flaw is that definition of a process is directly connected to its synchronization. An arbitrary process flow graph can be implemented using fork and join.

2.3. Cobegin and Coend

The statements *cobegin* and *coend* delineate blocks of statements that can be executed concurrently. The essential notion is that all blocks within the scope of the cobegin-coend begin at the same time and the cobegin-coend terminates only when all blocks within its scope have *terminated.* One distinct advantage of the cobegin-coend structure is that there is only one path in and one path out of the construct; another advantage is the explicit specification of which processes are being executed concurrently. All process flow graphs that are series-parallel can be implemented by cobegin-coend; however, the implementation of arbitrary graphs by cobegin-coend requires the introduction of extra null processes.

3. SYNCHRONIZATION

While there is only one fundamental reason for being concerned with the synchronization of concurrent processes (to control the cooperation/interference of one process with another), there are two distinct types of synchronization which can be useful. The first of these types occurs when the values of some shared variables are not "correct;" that is, one process should not use the variables until some other process has changed their value during its (the second process's) execution. The second of these types occurs when some subset of statements in a process must be treated as an indivisible operation. The dividing line between these two types of synchronization is not always sharp, as can be seen by considering the following simple example. Suppose a variable X has the value 1. Consider a concurrent program which consists of the following two assignment statements to be executed in parallel

$2 \cdot X \rightarrow X$

$3 \cdot X \rightarrow X$

The two statements can be executed in either order. The end result of this concurrent program will be correct provided that one of the statements does *not use the shared variable X until the other* statement has set its value. equivalently. the result will be correct provided each statement is treated as an indivisible unit. These two types of synchronization can be implemented in several distinct ways which are discussed in the following sections.

3.1. Shared Variables

Perhaps the simplest way to synchronize two processes is to have one of the processes set a variable when a particular condition is satisfied and to have the other process test the variable until its value indicates that the condition is satisfied. One obvious drawback to this technique is that the process which is waiting will be "spinning its wheels" testing the shared variable; this is an obvious waste of CPU time. A less obvious drawback is that the programmer is compelled to understand the synchronization that is necessary and to explicitly program it. The most serious drawback is the possibility of a *deadlock.* A deadlock occurs when two (or more) processes are waiting for events which can never occur. As a simple example consider the following concurrent program.

These programs implement two concurrent processes as the subroutines named ONE and TWO. The synchronization is implemented through the two shared variables named START1 and START2; the variable STARTi has the value TRUE when process numbered i is in the part of its program which directly influences the process numbered 3-i. This part is generally referred to as the *critical section.* The subroutines WORK1 and WORK2 which are not given should contain the actual critical sections of the two processes. The important point to notice is that both processes could enter the loops at about the same time with the result that neither one of them could continue doing anything productive.

A fairly simple and straightforward modification of Program 3-1 alleviates most of the possiblities for a deadlock and is given in Program 3-2 below. If the two processes executing subroutines ONE and TWO run at exactly the same speed and start at the same time then this program will never actually execute the subroutines WORK1 and WORK2.

Peterson (1981) introduced a third shared variable into this protocol; the third variable guarantees that both processes will eventually get to execute the critical sections and is implemented in Program 3-3 below.

3.2. Semaphores

A better and more detailed notion for implementing the mutual exclusion above using shared variables is to use a *semaphore.* A semaphore is a non-negative integer: there are two operations defined on a semaphore s

- *Signal(s)* which executes the assignment $s + 1 \rightarrow s$ and

- *Wait(s)* which delays execution until s is positive and then sets $s - 1 \rightarrow s$.

Using a semaphore, the concurrent program with critical sections given in the previous section can be rewritten as in Program 3-4 below.

Program 3-1: Mutual Exclusion Protocol with Possible Deadlock

```
      block data
      common start1,start2
      logical start1,start2
      data start1/.false./,start2/.false./
      end

      subroutine one
      common start1,start2
      logical start1,start2
c work that does not need to be synchronized
c with the other process can be done here
      start1=.true.
      do while (start1)
              if .not. start2 then
                      call work1
                      start1=.false.
              endif
      enddo
c work that does not need to be synchronized
c with the other process can be done here
      end

      subroutine two
      common start1,start2
      logical start1,start2
c work that does not need to be synchronized
c with the other process can be done here
      start2=.true.
      do while (.start2)
              if .not. start1 then
                      call work2
                      start2=.false.
              endif
      enddo
c work that does not need to be synchronized
c with the other process can be done here
      end
```

Program 3-2: Modified Mutual Exclusion Protocol

```
      block data
      common start1,start2
      logical start1,start2
      data start1/.false./,start2/.false./
      end

      subroutine one
      logical start1,start2
      common start1,start2
      do while (.true.)
c work that does not need to be synchronized
c with the other process can be done here
      start1=.true.
              if .not. start2 then
                      call work1
              endif
      start1=.false.
c work that does not need to be synchronized
c with the other process can be done here
      enddo
      end

      subroutine two
      common start1,start2
      logical start1,start2
      do while (.true.)
c work that does not need to be synchronized
c with the other process can be done here
      start2=.true.
              if .not. start1 then
                      call work2
              endif
      start2=.false.
c work that does not need to be synchronized
c with the other process can be done here
      enddo
      end
```

Program 3-3: Peterson's Mutual Exclusion Protocol

```
      block data
      common start1,start2,process
      logical start1,start2
      integer process
      data start1/.false./,start2/.false./
      data process/1/
      end

      subroutine one
      common start1,start2,process
      logical start1,start2
      integer process
c work that does not need to be synchronized
c with the other process can be done here
      start1=.true.
      process=2
      do while (start2.and.process.eq.2)
      enddo
      call work1
      start1=.false.
c work that does not need to be synchronized
c with the other process can be done here
      enddo
      end

      subroutine two
      common start1,start2,process
      logical start1,start2
      integer process
c work that does not need to be synchronized
c with the other process can be done here
      start2=.true.
      process=1
      do while (start1.and.process.eq.1)
      enddo
      call work2
      start2=.false.
c work that does not need to be synchronized
c with the other process can be done here
      enddo
      end
```

Program 3-4: Mutual Exclusion with Semaphores

```
        block data
        common s
        integer s
        data s/1/
        end

        subroutine one
        common s
        integer s
        do while (.true.)
c work that does not need to be synchronized
c with the other process can be done here
                call wait(s)
                call work1
                call signal(s)
c work that does not need to be synchronized
c with the other process can be done here
        enddo
        end

        subroutine two
        common s
        integer s
        do while (.true.)
c work that does not need to be synchronized
c with the other process can be done here
                call wait(s)
                call work2
                call signal(s)
c work that does not need to be synchronized
c with the other process can be done here
        enddo
        end

        subroutine wait(s)
        integer s
        do while (s.le. 0)
        enddo
        s=s-1
        return
        end

        subroutine signal(s)
        integer s
        s=s+1
        return
        end
```

3.3. Data-flow Synchronization

A data-flow algorithm, in the sense of Treleaven, Brownbridge, and Hopkins (1982), is a collection of statements together with a directed graph which represents the flow of data among the statements. We find the notion of a data-flow algorithm, as a means for implementing a concurrent process, compelling, for a variety of reasons. First, it is possible to implement a data-flow system on any network of processors which supports some simple communication primitives; nearly every computer installation has a network of processors so this approach to parallel processing has very broad applicability. Special hardware is required only for those tasks which have a communication/computation ratio which cannot be supported by standard hardware. Second, programming a data-flow system is simple; it is not *necessary* to have any special-purpose languages nor any special understanding in order to program such a system. A description of a primitive data-flow system developed for a network of VAXes is given in Eddy and Schervish (1986). Third, a programmer need not worry about the complex synchronization issues that are usually attendant to parallel programs; this separates the task of scheduling computations from the task of programming them. The critical notion in any data-flow system is the *granularity* of the problems that are appropriate for it, that is, the size of the computational tasks which will be treated as indivisible units. For example, O'Leary and Stewart (1985) discuss a special-purpose data-flow system known as the ZMOB. One interesting and unusual feature of data-flow systems is that the processor speeds need not be identical, and they can in fact be stochastic. Additionally, the communications network used for interprocessor communication can have a bandwidth and latency which are stochastic also.

The essential feature of any problem that can be run efficiently on a data-flow system is that the interprocess communication cost must be a small fraction of the process computation cost. We believe that any problem which is amenable to parallel computation can be implemented in an efficient manner on a data-flow system; the only dilemma is what are the correct size grains (indivisible computational tasks) for the particular application.

One interesting research problem is the use of a data-flow system for the numerical computation of high-dimensional integrals. Briefly, the problem is to decide how to decompose the integral across the processors, for example:

1. Should iterated integrals be performed recursively with different levels of the recursion calculated on different processors?

2. Is it better to use Gaussian quadrature or adaptive Newton-Cotes techniques on a data-flow system?

3. How do Monte Carlo techniques compare with iterated Gaussian techniques or with composite Gaussian techniques?

A second interesting problem is the optimization of high-dimensional functions on a data-flow system. Some work on this problem is reported by Schnabel (1986). Again, briefly, the basic question is how to decompose the computation, for example:

1. In a global optimization problem, how often, relative to function evaluations, should the separate processors exchange information?

2. Should the separate processors only evaluate the function or should the decision making be decentralized also?

3. Should the computation of function values be localized so that individual processors "learn" about the local behavior of the objective function?

Another interesting class of research problems is the stochastic modeling of the data-flow process. A data-flow system can be thought of as a multiprocessor with stochastic service times. A typical problem might be to minimize the *makespan* (the expected value of the completion time) if preemptive scheduling is not allowed and there is a cost associated with assigning tasks to processors. Related problems have been attacked by Bruno, Downey, and Frederickson (1981) and others.

4. MESSAGE PASSING

4.1. Communication Channels

A communications channel, in an abstract sense, is the specification of a source and destination for messages. Such channels are usually implemented in by either *direct naming* where the process names serve as source and destination or by *mailboxes* where the mailbox name is global and can be used by several source (or destination) processes simultaneously. The use of communication channels affects the interaction of concurrent processes in another way. The processes may perform their communication so that the output of each process becomes the input of another process; such communication is usually called a *pipeline*. Pipelines are a feature of the Unix operating system.

One common alternative mode of process interaction is a *client/server* model. A *client* process requests some particular service from a *server* process; upon completion the server process send a completion message to the client process. The client/server communications relationship is usually implemented with a *mailbox*. A client process can send a message to any possible server process and a server process can receive a message from any possible client process.

4.2. Synchronous/Asynchronous Communication

A critical feature of the interprocess communication is whether the communications protocol is synchronized or not. An additional complication is that the synchrony can be different at each end of the communications channel. In particular, each time an actual read or write is issued by a process it can, in principle, be executed with an implied "wait until completion" or not. Consider as an example a dedicated server process. Presumably, the server process begins by issuing a read with a wait until completion since the process does nothing until it receives a request from a client. The client process on the other hand presumably issues the request for service as a write with no wait if the service is not time critical or as a write with a wait if the client is unable to continue until the service is completed. When the server process has finished its work it issues a write (back to the client) without a wait and then reissues the read with a wait. The client process issues a read from the server process with or without a wait depending on the exact nature of the service provided. The point here is that whether or not the communication is synchronous or not depends on the application.

In the case of asynchronous writes, one (often unanticipated) problem is that a large number of writes may be issued without a corresponding number of reads. As a consequence the receiving process must have available an essentially unlimited amount of buffer space to store the messages until the receiving process is ready to read them. A sort of intermediate protocol between totally synchronous and totally asynchronous communication is to have some fixed limited amount of buffer space; this allows the sending process to operate asynchronously unless it gets too far ahead of the receiving process. In this case the communications protocol should not let the asynchronous write be executed because of a lack of buffer space.

5. SEQUENTIAL PROCESSORS

The essential features of the "von Neumann" model of a stored program digital computer are

1. The data is represented by digits in some number system.

2. The program and data both reside in the same memory.

3. Instructions are executed one after another.

In the last three decades the speed of serial computers has improved enormously. While a large part of this improvement has stemmed from improved technology at least some of this improvement has come from the partial introduction of parallel processing. In particular:

1. Separate processors now handle the input and output; this was previously a major burden on the CPU because of the relative slowness of external I/O devices.

2. Execution of instructions is somewhat overlapped both because of the existence of special processors for certain instructions and also because the individual instructions are decomposed into (and processed in) separate parts. This decomposition is referred to as *pipelining* and is discussed in more detail in Section 6 below.

3. Memory design now allows essentially simultaneous access to consecutive storage elements without interference (interleaving).

Flynn (1966) introduced a nomenclature for models of computation which has become, although imprecise, the standard. His scheme classifies machines on the basis of two attributes:

1. whether the machine can process more than one instruction simultaneously;

2. whether the machine can process more than one data item simultaneously.

The resulting four types of machines are:

1. SISD - single instruction, single data;

2. MISD - multiple instruction, single data;

3. MIMD - multiple instruction, multiple data;

4. SIMD - single instruction, multiple data.

It will be helpful in the discussion that follows to describe various computer models by analogy to a fast food restaurant; the customers (or, equivalently, their orders) represent instructions, the items ordered represent the data being processed, the servers represent processors and the Assistant Manager, if present, represents the control unit.

The von Neumann model describes the SISD machines. In the fast food analogy there is a single queue of customers and a single server under direction of the Assistant Manager; each customer's order is filled before the next customer's order is taken.

It is generally agreed that MISD machines do not exist although some authors put pipeline machines in this category.

The MIMD machines are generally very specialized, even unique. Two early examples are the C.mmp and Cm* machines built at Carnegie-Mellon University. Cm* consisted of 50 processors, each with its own memory, in five clusters of ten processors each. For example, memory references could refer to memory attached to the same processor, memory attached to a different processor in the same cluster, or memory attached to a different processor in a different cluster. The times required for access to data items in these three case are in the ratio of 1:3:9. Obviously then, efficient use of Cm* requires programs to have a certain "locality" in addition to their parallel nature.

An early commercial MIMD machine was the Denelcor *H*eterogeneous *E*lement *P*rocessor (HEP). The HEP could have up to 16 processors attached to its memory bus; unfortunately only a very small number of these machines were built. See Kowalik (1985) for considerably more detail. The fast food analogy for an MIMD machine involves a chain of fast food restaurants. If one restaurant runs out of, for example, strawberry milkshakes, it may obtain more from another restaurant in the chain or it may send customers to another restaurant in the chain.

A different example of MIMD architecture which will have growing importance is discussed in Section 8 below.

Turning now to the most important of Flynn's categories, there are several major types of SIMD machines:

1. array processors

2. associative processors

3. data flow processors

4. pipeline processors

5. systolic processors

In the next sections some of these will be described briefly.

6. VECTOR PROCESSORS

Vector processors are having a major impact on the world of scientific computing because of their raw speed and their ability to perform as general-purpose machines. This means that larger problems can be solved more rapidly and users do not have to learn new concepts to use the systems. There are a number of such machines all having slightly varying hardware architecture such as the Cray 1S, the CDC Cyber 205, and the two Japanese machines, the Fujitsu VP-200 and the Hitachi S810. These machines achieve their phenomenal speeds through a number of unusual architectural features together with the technology used to implement the system. The single most important feature is that the basic machine cycle time is on the order of 10 nanoseconds; this is roughly 1000 times faster than a typical personal computer.

The design of these machines is optimized for the processing of arrays because most large-scale scientific calculations are based on linear algebraic operations. The critical architectural features include pipelining of instructions, parallel functional units and the use of vector operations. A functional unit is a specialized part of the arithmetic/logical unit of the CPU which implements some specific portion of the instruction set and operates totally independently of the other units. Although a functional unit may require more than one clock period to complete its calculation, new pairs of operands may enter each unit during each clock period. This is because data is moved into a new set of registers (within the unit) at the end of each clock period. This is the notion of pipelining.

The use of vector operations is implemented by way of a set of special functional units and a set of special instructions which are executed by those units.

7. ATTACHED PROCESSORS

An intermediate approach to gaining high performance, between brute-force processor speed-up and large numbers of processors, is to use special-purpose hardware which does not have the *full capabilities of a general* purpose computer. Generally, these special-purpose processors are attached to some general-purpose processor. The attached processor appears to be some sort of I/O device to the general machine; the attached processor performs its I/O through the standard facilities of the general machine.

There are a surprising variety of attached processors with quite diverse characteristics. A few that should be of particular interest to statisticians are the Star Technologies ST-100, the FPS 164 and 264, the CSPI Mini-Map 211, the Mercury Zip, the Analogic AP400 and AP500 and the Skye Warrior. These machines range in price from about $5000 to $500,000; their speeds and capabilities cover an equally broad range. Generally speaking attached processors are array processors in the sense that their architecture is optimized for array operations; they achieve speeds approaching that of the vector processors described in Section 6 above at a fraction of the cost.

8. NETWORKS OF PROCESSORS

From our point of view, networks of similar processors provide the most exciting prospect for parallel computation. The reason is simply that we believe that the data-flow approach described in Section 3.3 above is the easiest method of concurrent programming currently available and is simultaneously the only existing method which will scale up to a very large numbers of processors. By the words "scale up" here, we mean simply that whatever works now for four or forty processors will ultimately work for 400 or 40,000 processors. The crucial detail in any system based on a network of processors is what sort of hardware intercommunications network exists among the processors.

Obviously, the most desirable situation occurs when every one of the processors is able to communicate directly with every other processor. In this case, the interprocessor communications network is a *complete graph*. Obviously, and unfortunately, such a network will not scale up; simple geometrical considerations show that the number of hardware channels on a single processor cannot increase without bound as the number of processors does increase. So a system with n processors and n hardware channels per processor does not scale up.

The entire game then is to invent interprocessor communication graphs which have short paths (where each channel counts as length 1) between any two processors and a number of communications channels per processor which grows very slowly (if at all) as a function of the number of processors in the network. Currently the most popular interconnection scheme is based on an n dimensional hypercube, letting the corners be processors and the edges be communication channels. If the hypercube has $n=2^k$ corners then there are $k=\log_2 n$ edges per corner (channels per processor). At this time there are at least three commercial hypercube systems: the Intel IPSC which can have up to 64 nodes (i.e., a 2^6 hypercube), the Hypernet System 14 which can have up to 256 nodes (i.e., a 2^8 hypercube), and the NCube Ten which can have up to 1024 nodes (i.e., a 2^{10} hypercube).

The hypercube interconnection scheme does not scale up to very large numbers of machines and a variety of other schemes are likely to appear in the market place in the next few years.

9. ACKNOWLEDGEMENT

The computer facilities of the Department of Statistics at Carnegie-Mellon University used in the preparation of this report were acquired with financial support from the National Science Foundation (under Grant MCS 82-05126 and Grant DMS 84-04927), from the Office of Naval Research (under Contract N00014-84-G-0137), and from Digital Equipment Corporation. Preparation of this report was partially supported by the Office of Naval Research (under Contract N00014-84-K-0588).

10. REFERENCES

- Andrews, G.R., and F.B. Schneider (1983). Concepts and notations for concurrent programming. *Computing Surveys*, **15**, 3-43.

- Banerjee, U., S.C. Chen, D.J. Kuck, and R.A. Towle (1979). Time and parallel processor bounds for Fortran-like loops. *IEEE Trans. Comp.*, **c-28**, 660-670.

- Bruno, J., P. Downey, and G.N. Frederickson (1981). Sequencing tasks with exponential service times to minimize the expected flow time or makespan. *Journal of the ACM*, **28** 100-113.

- Eddy, W.F. and M.J. Schervish (1986). Discrete-finite inference on a network of VAXes. *Computer Science and Statistics: Proceedings of the 18th Symposium on the Interface* (T.J. Boardman, Ed.) (this volume).

- Flynn, M.J. (1966). Very high speed computing systems. *Proceedings of the IEEE*, **14**, 1901-1909.

- Kowalik, J.S. (Ed.) (1985). **Parallel MIMD Computation: HEP Superocmputer and Its Applications.** MIT Press, Cambridge.

- O'Leary, D.P. and G.W. Stewart (1985 Data-flow algorithms for parallel matrix computations. *Communications of the ACM*, **28**, 840-853.

- Olson, R. (1985). Parallel processing in a message-based operating system. *IEEE Software*, July, 39-49.

- Peterson, G.L. (1981). Myths about the mutual exclusion problem. *Information Processing Letters*, **12**, 115-116.

- Satyanarayanan, M. (1980). Multiprocessing: an annotated bibliography. *Computer*, **13**, 101-116.

- Schnabel, R.B. (1986) Parallel computation for optimization problems. *Computer Science and Statistics: proceedings of the 18th Symposium on the Interface* (T.J. Boardman, Ed.) (this volume).

- Treleaven, P.C., D.R. Brownbridge, and R.P. Hopkins (1982). Data-driven and demand-driven computer architecture. *Computing Surveys*, **14**, 93-143.

DISCRETE-FINITE INFERENCE ON A NETWORK OF VAXES

William F. Eddy and Mark J. Schervish, Carnegie-Mellon University

SUMMARY

Because all data are discrete and because all digital computation is discrete, we investigate the consequences of abandoning the implied approximations involved in the use of continuous-parameter models for continuous random variables. In particular, we study models for discrete random variables specified by probabilities which can only assume a finite number of values. If we allow all possible models, the amount of calculation required is really formidable; for one very simple example we estimate that one million years of cpu time are required to determine the predictive distribution for a single future observation. Consequently, effort is needed both to reduce the amount of calculation required and to speed up the calculation that must be done. Of particular interest in this regard is the use of multiple microcomputers as parallel processors. This breakthrough has two important advantages. Firstly, it dramatically reduces the "wall clock time" required to perform the discrete-finite calculations. Secondly, it provides a numerically more stable algorithm for computing the results.

1. INTRODUCTION

We choose to investigate models for discrete data which assume only finitely many values for several reasons. First, observed data are discrete; only finitely many different values are possible in any particular situation. Second, all computations are routinely performed in a digital computer; this restricts the possible values at any stage of calculation to be a discrete finite set.

Third, we believe that the use of such models forces researchers to focus their attention on the important issues in statistical modelling. We agree with Geisser (1971, 1980) that the primary purpose of probabilistic inference is predictive in nature. That is, one models data statistically because one is interested in making predictions about other, as yet unseen, data. The reason that discrete-finite models force the focus onto predictive inference is that the models are simply vectors of non-negative numbers which add up to 1. We deliberately avoid expressing the models in terms of parameters which might be mistaken for quantities of interest.

Fourth, the availability of substantial amounts of computer time diminshes one of the major drawbacks to discrete-finite models. Calculation of any function of the predictive distribution from a discrete-finite model involves a sum over the various possible model vectors. Typically, the number of possible models is very large; in a fairly trivial example described in Section 3 below, the number of model vectors is 6 x 10^{16}. We have developed some computational algorithms which, combined with very high speed computation, will make some previously infeasible discrete-finite models computationally tractable. In particular, we have made use of a local area network of various models of DEC VAXes in order to perform most of the computations in parallel, thereby reducing elapsed time. Such parallel computation also facilitates partitioning of the numerical results into sums which can be accumulated with greater numerical accuracy.

2. DISCRETE-FINITE MODELS

Consider inference about a single observable X, which can assume one of d possible values $x_1, \ldots, x_d$. These values can be numerical or nominal, vector or scalar; we will refer to the set of possible values as the *observation space.* The distribution of X consists of a vector $p = (p_1,\ldots,p_d)^T$, where

$$p_j = \Pr\{X = x_j\}, \qquad (1)$$

and

$$p^T 1 = 1 \qquad (2)$$

Since all calculations we perform are discrete and finite, we assume each p_j is constrained to equal one of say m possible values, $v_1, \ldots, v_m$. For simplicity here, we will only consider the case where the $\{v_k\}$ lie on a grid that is equally spaced in [0, 1]; that is, we suppose that

$$v_k = (k - 1)/(m - 1).$$

One need only consider that subset of the collection of m^d "possible" vectors p which satisfy Equation 2. When p is specified, all inference about X can be based on Equation 1. We will refer to p as the *model vector* and to the set of all possible p vectors as the *model space*.

Next, assume that one is interested in making inference about a subset of some sequence $X_1, X_2, \ldots$ of observables which are exchangeable, in the sense that their labels provide no information about their joint distribution. We assume that each X_i must equal one of the values $x_1, \ldots, x_d$. A theorem of de Finetti (1937) (also see Hewitt and Savage, 1955) shows that conditional on some vector p satisfying Equation 2, the X_i are independent with distribution given by Equation 1. Once again, the model space is a finite collection of vectors, say, $\{r_1,\ldots,r_t\}$, with $r_s = (r_{1s},\ldots,r_{ds})^T$.

For convenience, let the distribution of p be uniform. We use subscript i to index observations and j to index possible values. Since, for each i,

$$\Pr\{X_i = x_j | p\} = p_j, \qquad (3)$$

we can calculate the conditional distribution of p given any finite subset X^* of $X_1, X_2, \ldots$ as follows. For j = 1, ..., d, let n_j be the number of observed X's in X^* equal to x_j so that $(n_1, n_2, \ldots, n_d)$ has a multinomial distribution conditional on p. Using Equation 3 and the conditional independence of the X_i given *p*, we obtain

$$\Pr\{p = r_s | X^*\} = K^{-1} \prod_{j=1}^{d} (r_{js})^{n_j} \text{ for } s=1,\ldots,t \qquad (4)$$

where

$$K = \sum_{u=1}^{t} \prod_{j=1}^{d} (r_{ju})^{n_j}.$$

The joint distribution of any further set of X's is given by Equations 3 and 4 and conditional independence given p. For example, the predictive *distribution of a single future* X is

$$\Pr\{X = x_j | X^*\} = \sum_{s=1}^{t} r_{js} \Pr\{p = r_s | X^*\} . \quad (5)$$

3. FINITE SAMPLE CALCULATIONS

An essential part of our investigations has been implementation of *various discrete-finite procedures* as Fortran programs.

3.1. The Simplest Case

With n = 15 observations on a variable assuming d = 10 different values, the amount of time to calculate the predictive distribution of one future observation on a VAX 11/750 for various grid sizes m is given in Table 1. The estimated time referred to in the last line of Table 1 is approximately equal to one million years. This is not to suggest that we should prepare to do calculations which require this amount of time, rather it suggests that serious work is required to find ways to *make inference* more feasible in a reasonable amount of time. We consider one direction in which this work may proceed in Section 3.2 below. This approacn is through restrictions on the model vectors. Another approach is to use parallel processing. We discuss our efforts in this direction in Section 4 below.

Table 1: Times To Compute Predictive Distribution

m	# of p's	Time(Seconds)
10	48620	29.53
15	817190	431.5
20	6906900	3529
300	6×10^{16}	3×10^{13}(estimated)

3.2. Smoothness

Consider the case in which each observable must equal one of d equally spaced numbers in the interval $[x_1, x_d]$. Assume $x_1 < x_2 < ... < x_d$. For continuous probability models, it is common to expect that $\Pr\{X \text{ near } x_i\}$ is close to $\Pr\{X \text{ near } x_j\}$ if $|x_i - x_j|$ is small. This property is the smoothness of the distribution of X. The traditional method of guaranteeing smoothness is to require the distribution of X to be a member of a parametric family of smooth distributions.

The method we choose for anticipating smoothness is to reduce the set of model vectors $\{r_s\}$ by *eliminating* all those wit 1 adjacent coördinates which are too far apart. This option has the potential for reducing the computational burden dramatically compared to the first approach and is, thus, the option we will pursue in detail. The reason is that the amount of time required to calculate a predictive distribution is proportional to the number of model vectors in the calculation.

There are several ways to implement the elimination of "rough" model vectors. The simplest is simply to specify some value ϵ and allow only those vectors r_s with adjacent coordinates closer than ϵ. *We have written some Fortran programs to* implement this simple smoothness criterion. For example, consider the bound with $\epsilon = k/(m-1)$. That is, we eliminate all vectors with $|r_{is} - r_{i+1,s}| > \epsilon$. Table 2 below gives the times in seconds (on an 11/750) required to compute the predictive distribution of a single future observation for *various values of n, d, m, and k. In the calculations* tabulated here, we made the added restriction that $r_{is} > 0$ for all i and s.

The Carnegie-Mellon Statistics Department has an attached processor which is optimized for floating point arithmetic (a CSPI Mini-Map MM-211). This processor is attached to a VAX 11/750 and is able to perform floating point calculations at up to to twenty times the speed of the VAX without degrading the VAX performance. We have also implemented the program described above on this processor. The results were, at best, disappointing. Over a range of problem sizes we obtained a speed-up (for this program) of only 25 to 60 percent, compared to the VAX 11/750. This "bad" performance is explained by the fact that the attached processor is optimized for floating point calculation and our program has very few such calculations in it.

Table 2: Times in Seconds To Compute
Predictive Distribution
With Smoothness Restrictions

n	d	m	k	# of p's	Time
15	10	20	5	46584	53
15	10	25	5	617283	630
15	10	25	6	720577	698
15	10	30	3	638896	524
15	10	30	4	1856064	1502
15	10	30	5	3352218	2576
15	10	40	5	23237938	16196
1	20	30	3	6099480	9416
1	50	54	1	19602	138
1	50	55	1	230444	1234
1	50	56	1	2123274	9409
1	100	104	1	161702	2156
1	100	105	1	3921519	40732

3.3. Other Approaches

One alternative way to eliminate "rough" model vectors is to consider only those vectors r_s which are unimodal in the sense that $\{i: r_{is} > c\}$ is a set of consecutive integers for all c. We have also programmed this unimodality criterion for comparison. Table 3 below shows the times required to calculate the predictive distribution of a future observation based on a sample of size n = 15 from a distribution assuming d = 10 distinct values for various values of m. The restriction that every coordinate of p be greater than 0 was also imposed in these calculations.

One method for choosing between the various alternatives is to supply some small set of hypothetical data and produce the predictive distribution for a future observation by each alternative method. These distributions can be easily plotted on a terminal screen allowing a user to choose the method which produces the most plausible predictive distribution. The chosen method can then be applied to the real data set. Of course,

Table 3: Times in Seconds To Compute
Predictive Distribution
With Unimodality Restriction

m	# of p's	Time
20	806	1
30	23028	24
40	248912	231
50	1604102	1406

the amount of time required to perform the necessary computations will also be an important factor in choosing between alternatives.

As an illustration, we can compare the predictive distributions produced by the models corresponding to

1. the third row of Table 1,

2. the seventh row of Table 2, and

3. the fourth row of Table 3.

The results were all based on the same sample of $n = 15$ observations of a variable with $d = 10$ possible values. The fifteen observations were 2, 3, 6, 6, 6, 6, 7, 7, 7, 7, 8, 8, 8, 9, 9. The predictive distributions corresponding to the three tables, respectively are given in Table 4 below.

Table 4: Three Predictive Distributions for 10 Possible Values

1	2	3	4	5	6	7	8	9	10
.02	.10	.10	.02	.02	.21	.21	.17	.13	.02
.06	.08	.08	.06	.07	.15	.18	.15	.12	.06
.03	.04	.05	.06	.08	.20	.24	.18	.10	.04

It is worth noting that the predictive distribution derived from the unimodal restriction is itself unimodal, because most of the data values were consecutive. The distribution derived under the smoothness condition on adjacent values is substantially smoother than the unrestricted distribution, and it is flatter than the unimodal one.

4. PARALLEL COMPUTATION

An interesting feature of our initial programs is that essentially all of the calculation is contained in a single loop which is executed once for each *model vector*. The result of one iteration through the loop is independent of all other iterations through the loop. This means that these calculations are particularly amenable to parallel computation. In particular, suppose there are L processors available to perform the calculations. Let W_S be the time required by that part of the computation which cannot be performed in parallel and let W_P be the time required by that part which can be performed in parallel, so that the time required to calculate a predictive distribution with one processor is

$$T_1 = W_S + W_P,$$

where W_P is typically many orders of magnitude larger than W_S. Obviously L processors can complete the task in time

$$T_L = W_S + W_P/L + W_0,$$

where W_0 is the added time to handle the overhead of the parallel processing. Hence, considerable improvement in speed of execution can be expected if L is large and W_S and W_0 are small compared to W_P. Unfortunately, no matter how large L is the ratio

$$T_L/T_1 > (W_S + W_0)/(W_S + W_P) > 0.$$

4.1. A Network of VAXes

We have actually begun implementing some of our algorithms on a system with truly parallel architecture. The Department of Statistics at Carnegie-Mellon University has, in addition to its VAX 11/750, several Microvax I's and Microvax II's which communicate via DECnet over Ethernet cables. We have developed a set of FORTRAN subroutines which allow us to create, on each of these machines, processes which can communicate with each other and divide up the work in an efficient manner so as to dramatically reduce the elapsed time required to perform the calculations described in Section 3. In fact, the system is general enough to be able to handle any problem in which the task is decomposable into parts which can be perfomed independently of each other.

The system we are using works as follows. There are essentially two programs and there are two types of processes on the network. One process will be called the *parent*, while the others will be called *children*. One of the two programs is run by the parent, and the other program is run by all of the children simultaneously. Both programs are required to perform interprocess communication, which will be described in more detail below. The parent program divides the set of model vectors into groups which we call ***messages***, and sends the messages to the children via the communication network. Each message consists of sufficient information for the child to construct the model vectors assigned to it and to calculate summands which will be added together by the parent to obtain the predictive likelihood and the predictive distribution of a future observation. When a child finishes its work, it sends its calculations back to the parent which combines them in an appropriate fashion.

Making sure that the above scheme runs smoothly requires careful attention to details. Because our system consists of three different kinds of VAX processors, timing can be a serious problem. For example, a Microvax I is slower than a VAX 11/750, which is slower than a Microvax II. It makes sense to assign the slower machines less work so that all children finish at about the same time. Since our system is in constant use by other researchers in the Department of Statistics, different processors will be subject to different demands on resources at different times. In order to avoid having a calculation delayed by one or two slow processors, a flexible system of message distribution is required.

We have chosen to divide the work into a large number of messages and to send them to the children as they are needed. That is, when a child finishes a message and returns the results to the parent, the parent then sends the child the next available message. The goal of creating the messages is to make them approximately the same size while keeping the amount of effort required of the parent to create each message relatively small. The reason is that a child may have to wait while the parent creates its next message. There are advantages and disadvantages to having small message sizes. One advantage of small sizes is that a slow child (either a slow CPU or a busy machine) will receive only a few messages, leaving the bulk of the work to be performed by those children who have the time and resources to do it.

An added advantage to partitioning the computation into small parts is that the numbers being added together are more nearly the same size than with a serial algroithm. This increases numerical accuracy in a manner similar to the pairwise algorithm described by Chan, Golub, and LeVeque (1983). See also, Eddy and Jones (1985).

On the negative side, small message sizes can cause a communications bottleneck for the parent. In order to make optimal use of resources, we have chosen to make the processor, on which the parent program runs, also run the child program. That is, one computer communicates with itself over DECnet as if it were a remote node. In order for this child to get any work done, the parent will have to spend most of its time in a quiescent state. A definite advantage to small messages appears when one considers the possibility of one child "dying" prematurely. All work done by that child, since it last reported results to the parent, is lost and must be redone. Since the network is not as reliable as one would wish, and systems sometimes crash for unanticipated reasons, it pays to have small messages.

One drawback to our system occurs when the number of processors becomes very large. The bandwidth of the communications channel will only handle a certain number of processors before becoming overloaded. Also, the parent must have a connection to every other processor, which can tax the resources of the processor on which the parent runs. This problem severely limits the size of the system in theory, but few organizations own enough stand-alone systems to encounter difficulty due to the bandwidth (for calculations of this kind.)

4.2. Description of the System

The system we use is an example of a *data flow system* as described by O'Leary and Stewart (1985). The concept of a data flow algorithm is that, once the program has started, the flow of data takes care of the distribution of work and the control of the program. In our system, after each child begins work, program control is handled by the return of results from the children which initiates the subsequent sending of the next message. In the paragraphs below, we describe in detail the different features of the network of VAXes.

4.2.1. Network communication

Communication between parent and child is done over DECnet. There are two communication channels, one called the *mailbox* channel and the other called the *data* channel. The mailbox channel is used to keep track of the network status. For example, when each child is created, the parent sends no data until it receives a message in the mailbox saying that the child has come to life. When a child dies, the parent gets a mailbox message saying the child is dead, and the parent must reassign the task on which the child was working, if there was one. The data channel is what the parent uses to send its messages (data) to the child and what the child uses to return its results.

4.2.2. Asynchronous communication

The parent only deals with data arriving from a child when it arrives. That is, the parent does not wait for messages or data, but rather issues asynchronous read requests and then goes on with its work. For example, upon starting a child, the parent opens the data and mailbox channels and issues an asynchronous read for a mailbox message. Then it goes to the next child and does the same until it runs out of children or is interrupted by the arrival of a mailbox message. When it reads a mailbox message saying that the child is alive, it issues an asynchronous read for a mailbox message, sends data to the child, and then· issues an asynchronous read for data. When the parent reads data returning from a child, it accumulates the results, sends more data, and issues another asynchronous read for more data. The child on the other hand, operates synchronously by reading data from the parent, performing its computations, writing the results back to the parent, and waiting for the next set of data. The key to this system working is that (i) the parent goes back to whatever it was doing after it issues an asynchronous read request and (ii) when an asynchronous read request is answered, the parent is interrupted from what it was doing and deals with what it reads. (The one exception to this is that if the parent is already reading the answer to an asynchronous read when another one is also answered, the second and all later ones queue up and are dealt with in order of arrival).

4.2.3. The parent process

Because the answers to asynchronous reads interrupt the parent and begin execution of a seperate set of code, they behave like subprocesses. In fact, the program flow following one of these asynchronous answers is completely seperate from the basic parent program. The basic parent program consists solely of the following:

1. Initialize with input data.

2. Loop through the children one at a time.

   - Open a link, if it is not currently open.
   - Issue asynchronous read request for mailbox message.

3. Wait some fixed amount of time.

4. Return to step 2.

All of the data handling is done by the subprocesses described below. Each subprocess is initiated when a read is answered by one of the children. Hence, the subprocess is associated with that child for the duration of its existence.

4.2.4. The mailbox subprocess

The first thing the mailbox subprocess does is check to see if the message is a birth message or a death message. A birth message means that the child is alive. In this case, data is sent to the child and an asynchronous read on the data channel is issued for the results. A death message means that the child is dead, and must be removed from the list of living children. In addition, if the child had been working on a set of data, the data must be requeued for delivery to another child at a later time. If, on the other hand, the child is dead because the parent killed it (when there is no data left to be sent), we need only remove it from the list of living children. When the last living child dies and there is no data to be sent, results are summarized and execution ceases.

4.2.5. The data subprocess

The data subprocess *begins* when the asyncronous read for the first set of results issued by the mailbox subprocess is answered. At that time, the results are accumulated. If there is no more data, a special data set is sent to the child which causes it to cease execution and send a death message back to the parent on the mailbox channel. If there is still data to be sent, the child receives the next packet of data. The subprocess then issues an asynchronous read for the results and quits. Notice that this last part of the subprocess is identical to part of the mailbox subprocess. In fact these two subprocess use the same code (which is *reentrant* for this purpose).

4.3. Empirical Study of the System

Our initial investigations of the network of VAXes has dealt with the questions of how much improvement do we get with more processors, and how large should each message be to make the best use of system resources. The example we used for comparisons had n=14 observations on a variable assuming d=19 values (0 to 9 in steps of .5) with a model space having m=22 and a smoothness criterion k=2 as described in Section 3.2. The total number of model vectors is 38,226,040 and the calculation of the predictive distribution of one future observation took 11 hours and 8 minutes on a single Microvax II (40,100 seconds).

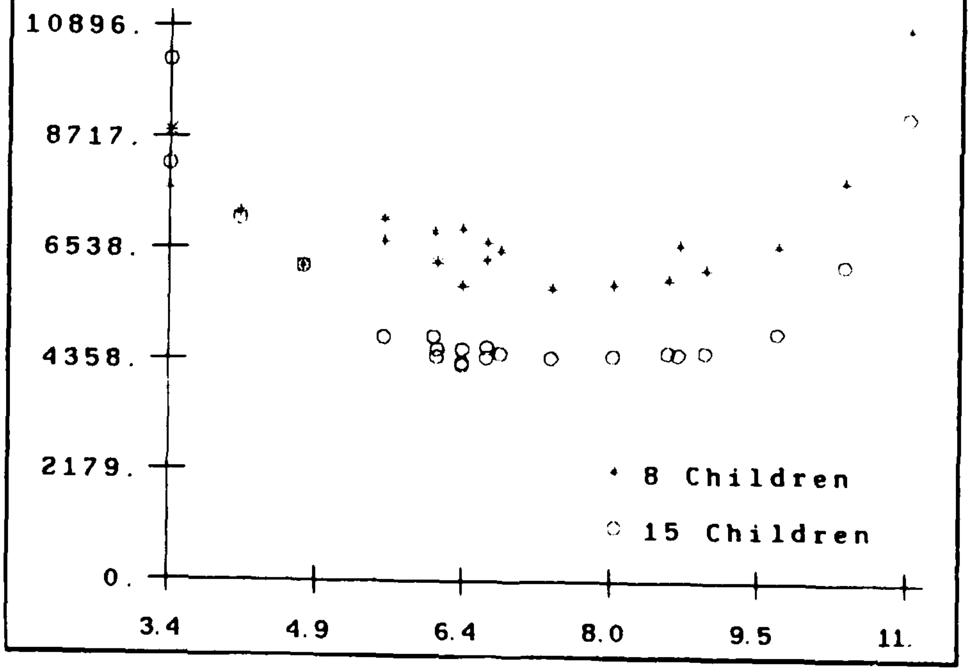
We ran the same case under several different conditions using the network of VAXes described above. Since the 11/750 runs about 80% as fast and the Microvax I's run about 20% as fast as the Microvax II's, we constructed two systems. The first system had 8 nodes: six Microvax II's the 11/750 and one Microvax I. This is roughly equivalent to seven Microvax II's. The second system had 15 nodes: eight Microvax II's, the 11/750 and six Microvax I's. This is roughly equivalent to ten Microvax II's.

The best timing obtained for the 15 node system was 4,303 seconds, and the best timing obtained by the 8 node system was 5,771 seconds. These numbers are 10.7% and 14.4% of the time taken by a single Microvax II. This is remarkably close to the best we could have expected, namely 10% and 14.3%. The time which the parent spent on each message varied from .13 to .15 seconds regardless of which system was used or how many messages were sent.

Figure 1 shows the total elapsed time for each system for several different sizes of message. The horizontal axis is Log (Number of messages) rather than message size. The pattern verifies our initial conclusion that when there are few messages (hence large ones), time will be wasted waiting for the last slow machine to finish its last message. And when there are too many messages (hence very small ones), time is wasted while the parent deals with all of the asynchronous read requests being answered almost immediately. It was generally true that the Microvax I's received about one-fifth as many messages as the Microvax II's when there were many messages, but not so many that each message was returned completed immediately.

Finally, we examined the numerical accuarcy of the computations. All of the calculations for the cases plotted in Figure 1 were done in double precision. We also ran the test case on the array processor attached to the 11/750, which only does single precision arithmetic. The computation took 35,036 seconds, and was incorrect by as much as 5% in some of the coordinates of the predictive distribution. This was verified by running the case serially in single precision on a Microvax II and obtaining an identical posterior distribution (in 36,642 seconds) despite the fact that VAXes and the attached array processor do not round identically

Figure 1: Seconds vs. Ln(# of Messages)



(c.f. Eddy and Jones, 1985). The parallel algorithm, however, was able to overcome the rounding error problem. We ran the same case in parallel with approximately the square root of the total number of model vectors per message (6237 model vectors per message in 6129 messages), but still in single precision. The predictive distribution was identical to the double precision result to five significant digits. This case took 37,151 seconds of total CPU time on Microvax II's. The message size was chosen to be optimal for numerical accuracy, not for speed.

4.4. Monitoring the System

In order to get the most out of the network of VAXes, it may be useful to monitor the network activity. We have arranged for a terminal screen to display the current status of each child as shown in Figure 2. One could display any information which seemed relevant in such a display. We have chosen to display the following:

- The name of the node on which the child is running and the time of the most recent message on the first line.
- Text describing the most recent message on the second line.
- Numbers of messages on the third line. The first number counts all incoming and outgoing messages to and from the parent. The second number is the number of data messages sent to that particular child. The third number is the total number of data messages sent to all children so far. The number -1 in either of these last two indicates that the communication link between parent and child is not open.

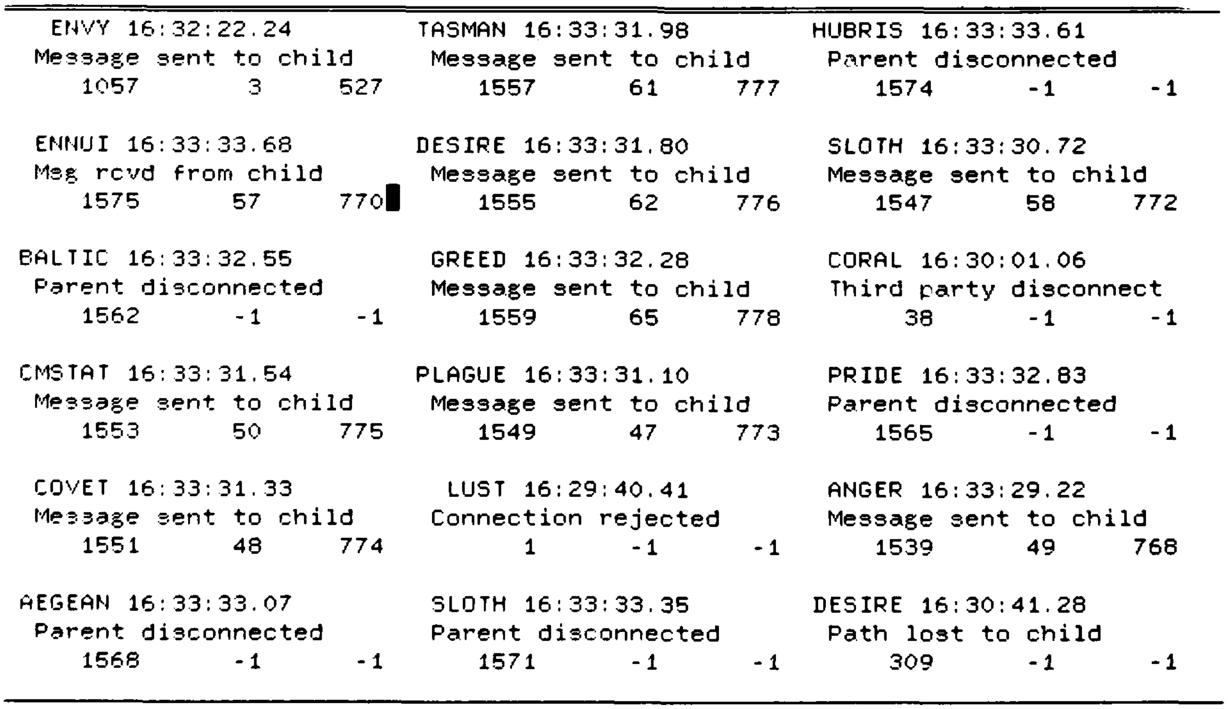
By use of such a monitoring device, one can see if attention needs to be paid to a particular node because the connection is not open. It also lets the user know if some of the children are not doing their fare share of the work. If the user knows how many messages will be sent, it also indicates how much of the problem has been completed.

5. FUTURE DIRECTIONS

The network of VAXes can actually be extended to include other kinds of processors, so long as a communications protocol is available. In addition, any problem which is decomposable into arbitrary sized pieces can be handled in parallel using the system. We explored discrete-finite inference in detail because it was the problem which first interested us in the system.

The most important thing still needed is a thorough analysis of system performance. It is straightforward to see that, if all processors perform their calculations at fixed (albeit different) rates, the elapsed time will be minimized by sending only one message to each processor with sizes being an increasing function of the rates. The reason for this, is that the overhead of sending messages is nearly independent of the size of the message. Of course, this might not be true for other applications. Our system, however, is part of a time sharing system, hence calculations are performed at different rates by the same processor at different times. Sending a single message to each processor can be disastrous in this case, if one of the processors has considerably more other work to do the others. Evidence of this appears at

Figure 2: Typical Terminal Screen While Monitoring Network Status



```
  ENVY 16:32:22.24          TASMAN 16:33:31.98        HUBRIS 16:33:33.61
 Message sent to child       Message sent to child     Parent disconnected
    1057        3      527      1557      61      777     1574       -1        -1

 ENNUI 16:33:33.68          DESIRE 16:33:31.80         SLOTH 16:33:30.72
 Msg rcvd from child         Message sent to child     Message sent to child
    1575       57      770      1555      62      776     1547       58       772

BALTIC 16:33:32.55          GREED 16:33:32.28          CORAL 16:30:01.06
 Parent disconnected         Message sent to child     Third party disconnect
    1562       -1       -1      1559      65      778       38       -1        -1

CMSTAT 16:33:31.54          PLAGUE 16:33:31.10         PRIDE 16:33:32.83
 Message sent to child       Message sent to child     Parent disconnected
    1553       50      775      1549      47      773     1565       -1        -1

 COVET 16:33:31.33            LUST 16:29:40.41         ANGER 16:33:29.22
 Message sent to child       Connection rejected       Message sent to child
    1551       48      774         1      -1       -1     1539       49       768

AEGEAN 16:33:33.07          SLOTH 16:33:33.35         DESIRE 16:30:41.28
 Parent disconnected         Parent disconnected       Path lost to child
    1568       -1       -1      1571      -1       -1      309       -1        -1
```

the left hand margin of Figure 1. We intend to perform a probabilistic analysis of the system performance, beginning with the case of discrete-finite inference, but hopefully extending to other types of problems as well.

6. ACKNOWLEDGEMENT

The computer equipment used during preparation of this report was acquired with financial support from the National Science Foundation (under Grant MCS 82-05126 and Grand DMS 84-04927), from the Office of Naval Research (under Contract N00014-84-G-0137), and from Digital Equipment Corporation. The work of Eddy was partially supported by Office of Naval Research (under Contract N00014-84-K-0588).

REFERENCES

Chan, T., G. Golub, and R. LeVeque (1983). Algorithms for computing the sample variance: analysis and recommendations. *American Statistician*, **37**, 242-247.

De Finetti, B. (1937). Foresight: Its logical laws, its subjective sources. *Ann. Inst. Henri Poincare*, **7**, 1-68.

Eddy, W. and A. Jones (1985). Array processors: A tradeoff between speed and accuracy?, in *Proceedings of the Statistical Computing Section, American Statistical Association*, 370-374.

Geisser, S. (1971). The inferential use of predictive distributions. In *Foundations of Statistical Inference*, V. Godambe and D. Sprott, eds., 456-459. Holt, Reinhardt and Winston, Toronto.

Geisser, S. (1980). A predictivist primer. In *Bayesian Analysis in Econometrics and Statistics*, A. Zellner, ed., 363-381. North Holland, Amsterdam.

Hewitt, E. and L.J. Savage (1955). Symmetric measures on cartesian products. *Transactions of the American Mathematical Society*, **80**, 470-501.

O'Leary, D. P. and G. W. Stewart (1985). Data-flow algorithms for parallel matrix computations. *Communications of the ACM*, **28**, 840-853.

MEDICAL DECISION MAKING

Organizer: *Richard H. Jones, University of Colorado School of Medicine*

A Brief Overview of Computer-Assisted Medical Decision Making
Richard H. Jones, University of Colorado

Teaching Medical Diagnosis Using Computer Feedback of Diagnostic Weighting
Robert S. Wigton, Kashinath D. Patil, Univ. of Nebraska College of Medicine; Vince L. Hoellerich, Johns Hopkins University School of Medicine; Roy Poses, UMDNJ-Rutgers; Marjeanne Collins, Randall D. Cebul, University of Pennsylvania School of Medicine

Toward a Science of Expert Systems
Eric Horvitz, Stanford University

Computer Applications of Bayesian Statistics in Medicine
Holly Jimison, Stanford University Medical Center

A BRIEF OVERVIEW OF COMPUTER ASSISTED MEDICAL DECISION MAKING

Richard H. Jones, University of Colorado

ABSTRACT

Decision trees are one of the most commonly used tools in medical decision making. This talk briefly explains the use of decision trees, how Bayesian statistical methods are used to convert prior probabilities before a test into posterior probabilities after a test, and how costs or utilities play an important role. As an example, a decision tree for a patient presenting with jaundice will be discussed. Three diagnostic tests for two diseases are considered.

1. Introduction

The purpose of this paper is to provide a brief introduction to decision trees, as used in medical decision making, for statisticians and computer scientists from the point of view of a statistician. The author is not an expert in medical decision making, but has been working with two surgeons at the University of Colorado, Ben Eiseman, M.D. and Brad Borlase, M.D., and a Biometrics graduate student, Maureen Haschke, to develop small, clinically useful, decision trees for use in surgical practice. Part of this effort is being supported by Rose Medical Center, Denver. Much of the terminology used in medical decision making is different from the usual statistical terminology. A second purpose of this paper is to explain the medical terminology in statistical terms.

Decision trees have been used for medical decision making for a number of years. A standard reference on the subject is Weinstein and Fineberg (1980). A recent book giving decision trees for use in surgical practice is Norton and Eiseman (1986). The Society for Medical Decision Making is a very active society that holds an annual meeting and publishes the journal Medical Decision Making.

2. An Example

Figure 1 shows an example of a decision tree for use in diagnosing a patient who presents at a surgical practice with jaundice. The two possible diseases considered in this example are gall stones (GS) and pancreatic cancer (PC). Two initial tests are ultrasound (US) and CT scan (CT). A more invasive test, ERCP, can be used at a later stage to confirm or rule out a diagnosis. Decision nodes are shown as filled in squares, chance nodes as open diamonds, and terminal nodes as filled in rectangles followed by an expected cost. Probabilities are associated with chance nodes.

After obtaining the patients history and doing a physical examination, the physician enters subjective probabilities (priors) for the various possible outcomes. The table below shows an example for the two disease case.

	D_1D_2	$D_1\bar{D}_2$	$\bar{D}_1D_2$	$\bar{D}_1\bar{D}_2$
Prob	π_1	π_2	π_3	π_4
Ex:	1%	85%	10%	4%

Table 1: Subjective prior probabilities for the two disease case. (The bar denotes absence of the disease.)

In this example the physician feels 85 percent certain that this particular patient has disease 1 and not disease 2. The probabilities at the first chance nodes depend on the prior probabilities and the sensitivities and specificities of the test for each disease. This is shown schematically in Figure 2 for the case a single test for a single disease. While this diagram and the terms used are very common in the fields of epidemiology and medicine, some translation may be necessary for statisticians and computer scientists who have not worked in these fields.

In epidemiology, the prior probability of a disease is referred to as the prevalence of the disease in the population being considered. This has the usual frequency interpretation of the proportion of people in the population with the disease. In the example presented here, the prior probabilities are the physician's subjective assessment of the chances that a certain patient has the disease. In most medical decision trees, the prevalence or frequency interpretation of prior probabilities are used. Defining a population as the people who come to a certain surgical practice with jaundice, the prevalence of gall stones is the proportion of these people with gall stones. This prevalence is baseline information for the physician. After taking the patient's history and doing a physical examination, the physician has much more information about this patient. This information is much more subjective since there is no longer a population, but a unique person.

The sensitivity of a test is the conditional probability that a person with the disease will test positive for the disease. The specificity of the test is the conditional probability that a person without the disease will test negative. A good test will have both the sensitivity and specificity near 1 being both sensitive and specific for the disease.

The probabilities of the outcomes of a test are

$$\begin{aligned} P(T+) &= P(D)P(T+|D)+P(\bar{D})P(T+|\bar{D}) \\ &= P(D)Se+P(\bar{D})(1-Sp) \\ P(T-) &= P(\bar{D})P(T-|\bar{D})+P(D)P(T-|D), \\ &= P(\bar{D})Sp+P(D)(1-Se) \end{aligned}$$

and these are assigned to a two outcome chance node. Se and Sp denote the sensitivity and specificity of the test. If the test is sensitive and specific for two different diseases, and the

Prevalence (Prior Probabilities)

$P(D)=\pi_1$ $P(\bar{D})=\pi_2$

	Disease	No Disease
Test +	True + (TP)	False + (FP)
Test -	False - (FN)	True - (TN)

Sensitivity = $\mathrm{Prob}(T+|D)=TP/(TP+FN)$

Specificity = $\mathrm{Prob}(T-|\bar{D})=TN/(TN+FP)$

Figure 2. Two by two table of test results and definitions of sensitivity and specificity.

outcomes of the test for the two diseases are statistically independent, a typical probability would be

$$P(+-) = P(D_1D_2)Se_1(1-Se_2)+P(D_1\bar{D}_2)Se_1Sp_2 + P(\bar{D}_1D_2)(1-Sp_1)(1-Se_2)+P(\bar{D}_1\bar{D}_2)(1-Sp_1)Sp_2.$$

Assumptions of statistical independence are often questionable. It is more likely that results of a single test for two different and unrelated diseases are statistically independent than that the results of two different tests for the same disease are statistically independent. The results of two different tests can be influenced by the stage of the disease. As the tree is traversed, both forms of statistical independence are assumed. The only way to avoid these assumptions is to obtain joint sensitivity and specificity data for multiple tests and multiple diseases.

The next step is called probability revision by medical decision makers. This is the application of Bayes' Rule to obtain posterior probabilities after a test is carried out. The usual medical or epidemiological terminology for posterior probabilities is predictive value of a test:

Positive Predictive Value

$$\mathrm{Prob}(D|T+)=\pi_1 TP/(\pi_1 TP+\pi_2 FP)$$

Negative Predictive Value

$$\mathrm{Prob}(\bar{D}|T-)=\pi_1 FN/(\pi_1 FN+\pi_2 TN)$$

Referring to the two by two table in Figure 2, predictive values are conditional probabilities obtained by normalizing across rows while the sensitivity and specificity are conditional probabilities by column. For one test and two diseases, a similar four by four table can be constructed.

If the tree is traversed forward, probabilities can be assigned to all chance nodes by recursive application of Bayes' Rule. Today's posterior becomes tomorrow's prior.

Costs are assigned to the terminal nodes which include: 1) costs of tests, 2) costs of hospitalization, 3) cost of incorrect diagnosis. The last is an expected cost since it involves the probabilities of various outcomes.

The tree is then folded back to obtain the expected cost at each node. At a decision node, the branch with the minimum expected cost is used.

The decision tree in Figure 1 was developed using the software package SMLTREE. It has been decided to program this type of tree in C. The goal is to develop a number of relatively small special purpose decision trees such as this for use in clinical practice. The actual tree will be hidden from the physician's view. A series of screens will display only the information relevant to the physician.

REFERENCES

Norton, L.W. & Eiseman, B. (1986) Surgical Decision Making, W.B. Saunders.

Weinstein, M.C. & Fineberg, H.V. (1980) Clinical Decision Analysis, W.B. Saunders.

SOFTWARE

DECISION MAKER, Borland International, Scotts Valley, California (TURBO Pascal).

SMLTREE, Jim Hollenberg, Pratt Medical Group, New England Medical Center.

TEACHING MEDICAL DIAGNOSIS USING COMPUTER FEEDBACK OF DIAGNOSTIC WEIGHTING

Robert S. Wigton, Kashinath D. Patil, University of Nebraska College of Medicine, Vince L. Hoellerich, Johns Hopkins University School of Medicine, Roy Poses, UMDNJ-Rutgers, Marjeanne Collins, Randall D. Cebul, University of Pennsylvania School of Medicine.

In learning medical diagnosis, medical students learn typical presentations of diseases, but there is little formal learning about what weight should be given diagnostic information in making a diagnosis or in distinguishing one disease from another. Furthermore, although they may learn the outcome of a case students are rarely given feedback on how well they have combined clinical information in making diagnostic or therapeutic judgements.

On the other hand, Hammond has shown that learners who were given feedback on how they appeared to have weighted information in making judgements learned much more effectively than those given only the correct outcome. (1)

To test whether this kind of feedback would have its predicted effect on the learning of medical diagnosis, we developed a microcomputer program to present simulated cases, obtain students' judgements for each case and then, after several cases, display the student's apparent weighting of information along with the correct weighting. Initial trials have indicated that his model is effective in teaching diagnostic relationships. In this paper, I will describe two applications of this program to medical decision making.

Design of Microcomputer Program

The microcomputer program operates by presenting simulated clinical cases and asking for a diagnostic or therapeutic judgement from the physicians or student in the form of an interval scaled measure. After each case, the student is given outcome feedback in the form of a score or probability calculated from a linear model. These models have either been derived empirically from analysis of large clinical populations or, where feasible, from published rules which have been validated on other populations.

The cases are constructed by loading descriptions of various levels of severity of each of the variables and using a fractional factorial design to determine the levels for each case.

After each series of simulated cases, the relationship between the clinical variables and the judgements made is calculated using dummy variable regression analysis and this result is presented to the student as a bar graph comparing the apparent weighting on the previous cases to that recommended by the model. Weights are expressed as percentage of total weight.

Both outcome feedback and feedback of weighting can be turned on or off under program control from any of 15 possible outcome measures. This allows investigation of diagnostic and treatment decisions as well as independent predictions of the likelihood of various diseases given the same cases (differential diagnosis).

Application to Medical Diagnosis

We first applied the method to examining the apparent weighting of clinical information by physicians in diagnosing pulmonary embolus.(2) (We refer to this as "apparent" weighting because it is not known whether a linear regression model is at all similar to how people use information in making such judgements.) The initial studies showed that physicians used these clinical factors in highly variable ways and showed great heterogeneity in apparent weighting. The average weights were also different from those derived from analysis of actual cases.

Diagnosing Urinary Tract Infection

The first application of the interactive features of the program tested whether medical students would learn the diagnosis of urinary tract infection more effectively if given feedback of weighting.(3)

The reference model was derived from analysis of records of ?28 patients suspected of urinary tract infection, seen in the Emergency Department of the University of Nebraska Hospital. This was later expanded to 750 patients. A five item rule was derived initially using discriminant analysis with the urine culture results as the outcome variable. This rule predicted the correct outcome in 80% of patients in a subsequent validation set.(4)

The variables were defined for each of three levels and the cases were displayed according to an underlying fractional factorial design with 18 iterations.(5)

After viewing each case, the student estimated the likelihood the urine culture would be positive and whether they would begin antibiotic therapy. After each case, they received outcome feedback in the form of the probability calculated from the rule derived from the actual cases.

After each set of 18 cases, the students were given a display of their weighting compared with that of the model in the form of bar graphs. After viewing the display and determining how his apparent weighting differed from the model, the student would proceed on to 18 new cases. Each student in this study completed 3 sets of 18 cases.

We compared learning for students who received this feedback with those who did not. Second year students were allocated to two groups. A control group received only the calculated probability after each case and the experimental group received the graphic display of weighting.

Both groups began the study with an average correlation between their probability estimates and those calculated from the model of .55. The experimental group learned more rapidly through the three sets, achieving a correlation of .80, while the control showed some learning but achieved a correlation of only .67 at the end. The improvement was accompanied in the experimental group by convergence of their average weighting on the weights used in the model.

Diagnosing Streptococcal Pharyngitis

In a second experiment, we looked at the effect of these types of feedback on the calibration of predictions as well as how well the model's weighting was learned. Poses and colleagues had studied how 11 student health physicians at the University of Pennsylvania used clinical information in diagnosing and treating streptococcal pharyngitis.(6) He found these experienced physicians greatly overestimated the probability of streptococcal pharyngitis in their cases. The apparent strategies of these physicians had been extensively studied. Logistic regression was used to model the relationship of clinical findings to the predicted and actual culture outcome. We decided to ask these same physicians to test the effect of computer feedback of weighting on their subsequent diagnostic performance.

A well validated decision rule for predicting streptococcal pharyngitis had been described by Centor (7) and was used as the model for the learning exercise. The rule gives equal weight to 4 items: fever, absence of cough, tonsillar exudate and enlarged anterior cervical lymph nodes. Three additional variables, important to the physicians, were added but given no weight in the rule.

Simulated cases were constructed from a fractional factorial design with 2 levels and 12 cases. Each case represented each of the variables as either present or absent.

Outcome feedback was again given as calculated probability. After each 12 cases, the weighting calculated from the answers was compared with that suggested by the rule.

The study consisted of 12 cases with no feedback as a baseline measure followed by a lecture explaining the rule and its rationale.

At one, two, and six months later, there were paired sessions of 12 cases each. Each of the three pairs of sessions consisted of 12 cases with outcome feedback after each case followed by lens model feedback. This was followed by 12 more cases also with feedback.

As in the previous study, the weights calculated from the judgements made by the physicians began to converge on the model weights as the study progressed. Correlation of the physicians' predictions with the likelihood calculated by the rule rose rapidly at first then continued a slow increase after two months. The mean probability estimates corrected rapidly. The decision rule had been corrected for the 5% prevalence of positive cultures in the student health population and the mean for the simulated cases was 6.5%. At first, there was considerable overestimation (24%) but the correct probability was reached after the first month.

A measure of calibration is the regression of the estimated probability on the actual probability; with perfect calibration falling on a diagonal line with a slope of 1 and an intercept of 0. Thus, if calibration improves, the slope approaches 1 and the intercept 0. In the early cases of this study, the intercept began at 2.7 and progressively declined to equal 1.05 at the last session. Similarly, the slope began at .17 and was .98 at the last session.

Both the lecture and lens model feedback produced changes in the appropriate direction and the changes persisted over 6 months. The program produced a rapid change in the mean probability estimates and calibration continued to improve. These changes occurred in simulated cases, but recent studies of these physicians after this intervention indicate they became more accurate and better calibrated in their real life predictions.

Thus, although these initial applications are quite limited in scope, the feedback of diagnostic weighting using simulated cases appears very promising in improving physicians' diagnostic and therapeutic predictions.

BIBLIOGRAPHY

1. Hammond KR: Computer graphics as an aid to learning. Science 172:903-908, 1971.
2. Wigton RS, Hoellerich VL, Patil KD: How physicians use clinical information in diagnosing pulmonary embolism: an application of conjoint analysis. Medical Decision Making 6:2-11, 1986.
3. Wigton RS, Patil KD, Hoellerich VL: Enhanced learning of clinical diagnosis through computer graphics feedback of diagnostic weighting. Proceedings of the 23rd Annual Conference on Research in Medical Education, 1984, pp. 111-116.
4. Wigton RS, Hoellerich VL, Ornato JP, Leu V, Mazzota LA, Cheng IH: Use of clinical findings in the diagnosis of urinary tract infection in women. Arch Intern Med 145:2222-2227, 1985.
5. Green PE, Carroll JD, Carmone FJ: *Some new types of fractional* factorial designs for marketing experiments. Research in Marketing 99-122, 1978.
6. Poses RM, Cebul RD, Collins M, Fager SS: The accuracy of experienced physicians' probability estimates for patients with sore throats. JAMA 254:925-929, 1985.
7. Centor RM, Witherspoon JM, Dalton HP, et al: The diagnosis of strep throat in adults in the emergency room. Med Decis Making 1981;1:239-246.

TOWARD A SCIENCE OF EXPERT SYSTEMS

Eric Horvitz, Stanford University

ABSTRACT

Over the last several years, teams working on expert systems have been exploring formal approaches for belief revision and information acquisition. The formalization of major components of expert systems operation is useful for understanding and characterizing system behavior and for predicting changes with modification. Formalization also facilitates the involvement of investigators in more well-developed disciplines such as statistics. While the use of formal methodologies for diagnostic problem solving is attractive because of the generality, power, and axiomatic basis of inference, the methodologies have been criticized for making inferences that are difficult to understand and explain. I shall focus on the problem of explaining formal reasoning methodologies. The PATHFINDER system for pathology diagnosis is presented as an example of current research on aspects of the use of formal methodologies in expert systems. I will demonstrate that a formal system is amenable to controlled degradation to enhance its explanation capability.

1. INTRODUCTION

It is fitting that there be a focus of discussion on expert systems in a session on computers and medical decision-making. Original ground-breaking research on expert systems was the result of attempts to build systems to reason about complex medical problems [4]. Expert systems research developed within the field of artificial intelligence over a decade ago and is now an established engineering sub-discipline of artificial intelligence. It is the intent of expert system research to develop methodologies for the representation and manipulation of the knowledge of experts in a variety of disciplines.

Artificial intelligence research is still in its youth. As in other new disciplines in which unifying theories have not been developed, much work has focused on non-axiomatic, descriptive models. In this paper, I would like to briefly introduce the descriptive and formal approaches to research in artificial intelligence in general. I will stress the usefulness of reasoning methodologies that follow from a set of well-characterized axioms. I will then introduce current problems with the use of formal systems. One frequent criticism of formal reasoning strategies is that they are difficult to understand and explain. I will focus on the problem of explanation in expert systems that use formal methods for reasoning under uncertainty. In this regard, I will present research on the PATHFINDER expert system for pathology diagnosis as an example of research on aspects of the use of formal methodologies in expert systems. In answer to some complaints about the rigidity and unnatural nature of formal systems, I shall describe how a formal system is amenable to controlled degradation so that it can perform more descriptively.

2. AXIOMATIC AND DESCRIPTIVE APPROACHES

Science has been marked by an ongoing attempt to explain observed patterns and relationships with models that provide reasonable explanations and predictability. Useful theories tend to simplify phenomena through explaining complexity with a relatively small number of empirically or intuitively justifiable properties or axioms.

Unfortunately, theories based on a set of justifiable axioms often do not exist; when a theory is enumerated, it is often not obviously optimal, unique, or desirable. Throughout the history of science, when useful axiomatic theories have not been available, scientists have resorted to *descriptive* models. Such models summarize complex behavior by *describing* phenomenology without resorting to fundamental axioms. They capture the behavior of systems, often through the postulation of relations that may be inconsistent with one another or with other accepted knowledge. As an example, before Newton constructed the theory of universal gravitation and Kepler developed equations describing the motion of objects orbiting in gravitational fields, astronomers often depended on epicycle machines. These machines could approximately *describe* the movement of heavenly bodies, as viewed from the earth, with a complex tangle of gears and chains. They did not explain the movement of heavenly bodies with a consistent theory of fundamental relationships.

2.1 Descriptive Expert Systems Research

Much of expert systems research can be characterized as either axiomatic or descriptive. The descriptive expert system approach centers on the design and empirical evaluation of algorithms that mimic aspects of human behavior. Descriptive expert systems research is not hindered by the lack of a formal axiomatic basis; it is the intent of the research to discover useful strategies for representing and manipulating expert knowledge regardless of the availability or acceptability of a set of self-consistent desiderata. Investigators in the descriptive school of research view exploration of the sufficiency of informal models of human problem solving as a more direct approach to difficult problems. That is, given poor understanding, many expert systems researchers attempt to capture expertise through building and experimenting with descriptive models in the spirit of the epicycle machines of long ago.

As an example of the descriptive approach to expert system design, the Present Illness Program (PIP) [23], developed ten years ago at M.I.T., was an attempt to simulate the cognition of a physician's reasoning about patients presenting with edema (swelling). A central aspect of the design of the system involved an analysis of the behavior of the clinician. Final versions of PIP had descriptive cognitive structures called the *supervisory program*, the *short-term memory*, and *long-term memory* were constructed.

A large category of descriptive systems is based on the *rule-based* methodology [4]. The rule-based expert system methodology is the result of attempts to adapt the use of an automated logical inference methodology, called *production systems* [32, 7], to capture aspects of human expertise. Production systems are comprised of sets of logically interacting inference rules of the form IF E THEN H, where H is a hypothesis and E is evidence having relevance to the hypothesis. In practice, rules of logical inference are used in automated deduction. For example, *modus ponens* and simple rules of *unification* can be applied to a set or *knowledge base* of rules to do proofs that consist of the forward or backward "chaining" of rules.

One of the most prolific early expert systems was MYCIN [31], a rule-based expert system for the diagnosis of bacterial infection. The MYCIN reasoning framework remains one of the most popular expert system methodologies. MYCIN's knowledge is stored as rules that capture the relationships among relevant medical evidence and hypotheses. For example, a rule in MYCIN might be: "*if an organism infecting a patient is gram-positive and grows in clumps then add support to the hypothesis that the organism is staphylococcus.*" It was recognized early on in the MYCIN research that straightforward application of the production rule methodology would be insufficient because of the uncertainty in the relationships between evidence and hypotheses in medicine.

In order to accommodate these non-deterministic relationships, MYCIN uses certainty factors [4]. To each rule, a certainty factor is attached which represents the *change* in belief about a hypothesis given some evidence. Certainty factors range between -1 and 1. Positive numbers correspond to an *increase* in belief in a hypothesis while negative quantities correspond to a *decrease* in belief. An *ad hoc* calculus for evidence combination was presented in the original research [30].

2.2 The Axiomatic Approach

In contrast to the descriptive approach, investigators pursuing the formal axiomatic approach are interested in exploring the adequacy of systems that satisfy desired properties. That is, they design expert systems that are necessarily *consistent* with desired properties. When such a set is deemed *optimal* for reasoning in the context of particular tasks it is termed a *normative* theory for reasoning.

Investigators interested in the formal approach attempt to design expert systems that behave consistently with established theories for *reasoning under uncertainty*. In exploring the automation of reasoning under uncertainty, investigators have focused on the use of theories for the consistent *revision of belief* in the context of previous belief and for controlling *information acquisition*. Examples of axiomatic theories that have been used in expert systems research for belief revision include probability [24], fuzzy logic [39], Dempster-Shafer theory [28], certainty factors [30], and multi-valued logics [13]. Theories used for controlling information acquisition include information theory [29] and decision theory [25, 26].

Alternative formalisms are often based on clear sets of properties. An expert system engineer can base an expert system on a set of properties that is viewed to be a particularly intuitive or desired. For example, a set of simple properties about continuous measures of belief can be shown to necessitate the use of probability theory to manage the consistent assignment of belief [6, 36, 20]. Agreement with the properties *necessitates* the use of probability theory. A small set of intuitive properties also lies at the foundation of decision theory [37]. Of course, there are differences of opinion among the formalists about the optimality or necessity of particular sets of axioms. For example, there has been ongoing debate in the artificial intelligence community regarding the alternative methodologies for the revision of belief [5, 20].

To date, there have been several attempts to base expert reasoning systems on well-defined formalisms. Three examples are the Acute Renal Failure [15] system, the MEDAS [1] system for emergency medicine, and the PATHFINDER [17] system for lymphoma diagnosis. These systems were designed to be consistent with well-understood formalisms for reasoning.

Both the descriptive and axiomatic approaches have led to the construction of systems that perform at levels rivaling experts in a variety of domains. Given the complexity of problems at hand and the youth of the field, both approaches have been useful in exploring techniques for automated reasoning. In general there has been a healthy interplay between the the descriptive and the axiomatic research; a dynamic research milieu is created by the co-existing approaches.

3. THE BENEFITS OF FORMALIZATION

A worthy fundamental goal of research should be the eventual development of useful *theories*. As in any science, the study of automated reasoning would benefit greatly from attempts to construct theories for representing and manipulating knowledge. Whether an investigator initially chooses to become involved with descriptive or formal research, a fundamental goal should be the construction of a formal science. A strong theoretical basis for components of expert reasoning systems would be extremely useful. While there have already been strides in the application of formal theories to expert systems, greater understanding could facilitate the design, control, and characterization of expert systems.

The subscription to axiomatic bases for components of expert reasoning can be useful in a number of ways. It can assure a system engineer that the behavior of his system wil! remain consistent with a set of desired properties. Basing a system on a formal theory also ensures that the system will be self-consistent. If an axiomatic theory is not used in building an expert system, it can be quite difficult to maintain self-consistency. The presence of inconsistencies in complex computer systems often leads to unpredictable behavior.

Recent research on the *ad hoc* certainty factor model used for combining evidence in the MYCIN system introduced above has found the original model to be self-inconsistent [16, 18]. Recent work has focused on removing inconsistencies in the model [16]. The consistent reformulation of certainty factors demonstrates that the belief revision theory is a *specialization* of probability in that assumptions of conditional independence are imposed by the methodology. For example, it can be shown that evidence must be conditionally independent given H and its negation [16]. The determination of inconsistency and the detection of constraints were facilitated by the formalization of MYCIN's reasoning strategies.

Formal models can also assist an engineer greatly when a system is modified. A formal system allows for the crisp prediction of changes in system behavior in response to system modifications. It can be quite difficult to predict the impact of modifications on systems for which no underlying theoretical structure is available. Having the ability to control the effect of system modifications is extremely important for the maintenance of systems, for the generalization of specific successes, and for the incremental refinement of techniques. Incremental refinement can be particularly significant in the continuing development of a theoretical framework for automated reasoning.

Most relevant for this conference, formalization can also be crucial for expert systems research to benefit from the participation of investigators in other highly-developed disciplines. Issues surrounding descriptive and axiomatic expert systems research are of special relevance in this regard. For example, expert systems research would benefit if it could attract statisticians to assist in solving difficult problems. Formal descriptions of systems and methodologies are important as they provide conceptual handles necessary for communication with researchers in other fields.

4. PROBLEMS WITH THE FORMAL APPROACH

Two central issues that arise in discussions of the axiomatic approach are problems regarding the pragmatics of engineering and computation, as well as explanation.

4.1 Tractability of Engineering and Computation

More so than for any other reason, researchers in artificial intelligence have looked beyond axiomatic-based techniques for complex domains because of the computational overhead of inference and the requirement for large amounts of knowledge. Formal methodologies are viewed as having an insatiable thirst for data and computer processing [8, 34].

4.2 Explanation

Another significant problem cited with respect to formal methodologies is that it is difficult to explain recommendations to users. The explanation of expert systems has been identified as an significant factor in the acceptance of expert systems [35]. In fact, the transparency of reasoning has been cited as a fundamental feature of expert systems, distinguishing them from numerical programs and other kinds of reasoning systems in artificial intelligence [3]. The important role of reasoning transparency in expert systems has made explanation an artificial intelligence research focus.

It has been said that formal methodologies like probability theory and decision analysis lead to unavoidable losses in comprehensibility to expert system users [8, 34]. The manipulation of the equations of conditional probability or decision trees may indeed be quite difficult to succinctly explain. Such difficulties have provoked some of the ongoing work on techniques for justifying the results of formal reasoning strategies [33, 27, 20]. We shall focus more closely on this problem below.

5. GRACEFUL DEGRADATION OF PERFORMANCE

The concerns about problems with explanation, knowledge acquisition and computational tractability of systems based on formalisms for reasoning under uncertainty are valid. Indeed the methodologies demand large amounts of data and computation. Complaints about the opacity of explanations of recommendations are also justified.

Formal methodologies for reasoning under uncertainty have been put forth as general theories. They have not been designed for use in complex reasoning systems that might be dominated by limitations in computational and engineering resources. An interesting and potentially fruitful area for investigation is the development of strategies for modifying formal methodologies to perform under specified constraints. The process of identifying pressing resource limitations followed by an attempt to reformulate theories (deemed optimal in a world with infinite resources) to perform in constrained environments could be more useful than the outright dismissal of the theories. Such techniques could allow and engineer to gracefully degrade a systems performance to reflect diminishing amounts of available engineering or computational resource.

Theories of belief revision and information acquisition have not traditionally been accompanied by tools that allow a well-defined relaxation of restrictions or requirements. It would be productive to develop such methodologies to generate well-characterized trade-offs such as between the accuracy of a recommendation and computation time. Useful approaches to graceful degradation of various aspects of reasoning behavior would make the disagreement with properties of general parent theories clear. The development of strategies for the controlled degradation of reasoning would allow artificial intelligence researchers to continue to build upon the theoretical achievements of more mature disciplines.

We will now turn to an example of the degradation of expert system performance to satisfy constraints on the *complexity of inference*. As we shall see, degrading an optimal reasoning methodology can serve to enhance the explanation capability in an expert system.

6. EXPLAINING COMPLEX REASONING

I would like to demonstrate an example of the decomposition of a complex reasoning methodology. I hope that it may serve as an example of a category of strategies that can help investigators successfully apply axiomatic models. First I will present an information-optimizing reasoning strategy that makes inferences that are difficult to explain. I will then describe how a less efficient but more explainable strategy could be generated.

6.1 The Complexity of Reasoning Under Uncertainty

We have proposed [19] that a central aspect of the difficulty that investigators have had in explaining expert system recommendations is based on the intrinsic complexity of formal reasoning under uncertainty. As often noted, a fundamental difference between simple deduction and more general reasoning under uncertainty is the inference complexity: within a deductive system, any particular path to a conclusion is considered to be a sufficient proof; in contrast, reasoning under uncertainty usually entails the consideration of all paths [5] Formal theories of belief revision and information acquisition generally involve the parallel consideration of a greater number of propositions than simple logical deduction problems. For example, probabilistic reasoning systems calculate the values of single conditional probabilities to summarize many steps of inference. This complex summarization process, so central in probabilistic inference, has been seen as a problem in expert system understandability [8].

What is the fundamental basis for problems with complexity? Cognitive psychology results can lend insight to this question. Problems associated with the comprehension of complex problems such as the operation of complex reasoning strategies have been a longtime research focus within cognitive psychology [2]. Classic research in this field has demonstrated severe limitations in the ability of humans to consider more than a handful of concepts in the short term [21]. In fact, studies [38] have discovered that humans cannot retain and reason about more than two concepts in an environment with distractions. Such results underscore the need for managing the complexity of expert systems inference.

For humans to successfully understand, plan, prove, and design in environments that are informationally complex, they must devise schemes for decomposing large unwieldy problems into smaller, interrelated sub-problems. I will present our work on the enhancement of explanation through the decomposition of complex formal reasoning. Before presenting the work, I must first describe the hypothetico-deductive architecture of PATHFINDER.

7. THE PATHFINDER PROJECT

PATHFINDER [17] is a hypothetico-deductive expert system for the diagnosis of lymph node pathology based upon the appearance of microscopic features in lymph node tissue. Disease manifestations in lymph node pathology are microscopic *features*. Features are each subdivided into a mutually exclusive and exhaustive list of *values*. Features are evaluated by the selection of a value that reflects the status of the feature in the case being reviewed. We say that the assignment of a value to a feature constitutes a *piece of evidence*. The PATHFINDER system reasons about 80 diseases, considering over 500 pieces of evidence.

7.1 The Hypothetico-Deductive Architecture

The PATHFINDER system is based on the hypothetico-deductive architecture. The hypothetico-deductive method (also referred to as the method of *sequential diagnosis* [14]) has been studied in several expert systems research projects including the Acute Renal Failure [15] system, the INTERNIST-1 [22] system for diagnosis within the field of internal medicine, and the MEDAS [1] system for emergency medicine.

Hypothetico-deductive systems are presented with an initial set of evidence. The initial evidence is used to assign a probabilistic or quasi-probabilistic score to each hypothesis and a list of plausible hypotheses is formulated from the scores. Then, questions are selected which can help decrease the number of hypotheses under consideration. After a user replies to requests for new information, a new set of hypotheses is formulated and the entire process is repeated until a single diagnosis is reached.

The question selection strategies are termed hypothesis-directed in that reasoning strategies operate on the current list of hypotheses under consideration to generate recommendations for additional evidence gathering. Investigators in the INTERNIST-1 and PATHFINDER research groups have explored the usefulness of tailoring different reasoning strategies to the current list of diseases under consideration or *differential diagnosis*. For example, the strategy selected to narrow the differential diagnosis may depend upon the number of diseases on the differential, the probability distribution over the differential, or both.

The advice generated by hypothesis-directed strategies is often difficult to explain because of the complexity of their operation. This is especially true if recommendations are the result of inferences based on a large hypothesis list. Hypothesis-directed strategies may consider the relevance of hundreds of hypotheses in a single inference step.

The scoring scheme employed by PATHFINDER is based upon the theory of subjective probability [9]. The subjective probabilities of experts are used to infer the probability that each disease is responsible for the evidence that has been entered into the system. Depending on the number and the distribution of probabilities among diseases on the differential diagnosis, PATHFINDER chooses one of several alternative diagnostic strategies for selecting questions. As in other hypothesis-directed systems, it is the goal of the question selection strategies to suggest the optimal test to be evaluated next in an effort to reduce the uncertainty in the differential diagnosis.

Several PATHFINDER strategies discriminate among large numbers of diseases and features in the generation of advice. I shall not describe all of the hypothesis-directed reasoning strategies used by PATHFINDER. Rather, we will look at issues surrounding the explanation of a particular PATHFINDER hypothesis-directed reasoning strategy termed *entropy-discriminate* and its descendant, *group-discriminate*.

7.2 A Strategy to Minimize Uncertainty

The PATHFINDER *entropy-discriminate* reasoning strategy was originally used to refine differential diagnosis disease lists ranging in size from two to eighty diseases. The strategy makes recommendations about information acquisition by searching for tests that maximize a measure of information contained in the differential diagnosis. Similar information-maximizing strategies have been examined in the MEDAS and Acute Renal Failure systems.

Entropy-discriminate makes use of a measure of information known as *relative-entropy*. In this context, relative entropy is a measure of the additional information provided by a piece of evidence E_i about a differential diagnosis DD. Formally,

$$H(DD,E_i) = \Sigma_j \; p(D_j|E_i) \; \log[p(D_j)/p(D_j|E_i)],$$

where $p(D_j)$ is the probability that disease D_j is present before evidence E_i is known, the *prior* probability of the disease, and $p(D_j|E_i)$ is the probability that disease D_j is present after evidence E_i is known, the *posterior* probability of the disease. For a justification of relative entropy as a measure of information gain, see [29].

As each feature consists of a set of mutually exclusive and exhaustive values, we can denote the possible evidence associated with a particular feature, F, as $E_1..E_n$, where n is the number of mutually exclusive values associated with the feature. Entropy-discriminate selects features which give the highest expected relative entropy

$$\langle H(DD,F_n)\rangle = \Sigma_i \; p(E_i) \; H(DD,E_i),$$

where the quantity is summed over feature values $E_1..E_n$, and $p(E_i)$ is calculated using the expansion rule

$$p(E_i) = \Sigma_j \; p(E_i|D_j) \; p(D_j).$$

In an information-theoretic sense, the questions selected by the entropy-discriminate strategy are *optimal* assuming that the goal of the pathologist is to reduce uncertainty in the differential as much as possible.

7.3 Problems With the Optimal Strategy

Soon after the implementation of entropy-discriminate mode, we discovered that several expert pathologists, including the expert that provided the system's knowledge, often found that selected questions were difficult to understand when the differential contained more than approximately ten diseases. The entropy-discriminate strategy of selecting questions that best discriminate among *all diseases on a differential diagnosis often* seemed to be too complex for experts. This is not surprising in light of the limitations of human short term memory discussed above.

We also had problems explaining the recommendations of entropy-discriminate whenever there were more than two diseases on the differential. Attempts were made to provide textual and graphical explanations for the powerful strategy's recommendations. One such graphical explanation justified questions by listing, for each disease, the feature value that would most favor the disease. Physicians found such complex summarizations to be difficult to understand.

7.4 The Graceful Decomposition of Diagnostic Problem Solving

The observed problems with the entropy-discriminate strategy stimulated our interest in strategies for simplifying and explaining hypothesis-directed reasoning. We discovered that pathologists often manage the complexity of the diagnostic problem-solving task by reasoning about a very small number of disease categories or groups at any one time. Questions that discriminate among natural groups tend to be proposed.

Specifically, the chief expert pathologist on the PATHFINDER team often imposes a simple two-group discrimination structure on the problem-solving task. As opposed to a strategy of discriminating among all the diseases on the differential, the pathologist's discrimination task at any point in reasoning about a case is constrained to only two groups of diseases. As categories of diseases are ruled out, the particular pairs of groups considered become increasingly specific. For example, if there are benign and malignant diseases on a differential diagnosis, the pathology expert often deems most appropriate those questions that best discriminate between the benign and malignant groups rather than questions that might best discriminate among all of the diseases. If all benign diseases have been ruled out, leaving only primary malignancies and metastatic diseases on the differential diagnosis, the pathologist will attempt to discriminate between the primary malignancy and the metastatic categories.

We found that the expert's diagnostic strategy can be described by the traversal of a hierarchy of disease categories. The problem-solving hierarchy (see Fig. 1) is a binary tree of disease groups. The hierarchy can be used to group the differential diagnosis at various levels of refinement.

It is interesting to note that several previous studies of medical reasoning have identified similar problem-solving hierarchies [10, 11, 12] for managing the complexity of a wide-variety of reasoning tasks.

The discovery of this expert reasoning strategy in lymph node pathology suggested the development of a new question-selection strategy that could discriminate among binary groups of diseases instead of individual diseases. It was hoped that design and application of such a strategy would make explanation clear, as the user would only have to consider the relevance of a recommendation to two groups.

Our attempt to naturally constrain the discriminatory focus of the entropy-discriminate strategy led to a new reasoning strategy we named *group-discriminate.* The group-discriminate strategy selects questions based on their ability to discriminate between the most specific pair of disease categories that account for all diseases on the differential.

For a given differential diagnosis, group-discriminate identifies the most specific grouping possible and then selects questions that best discriminate among groups of diseases. More formally, suppose the differential is split into two groups, G_1 and G_2, of n_1 and n_2 diseases respectively:

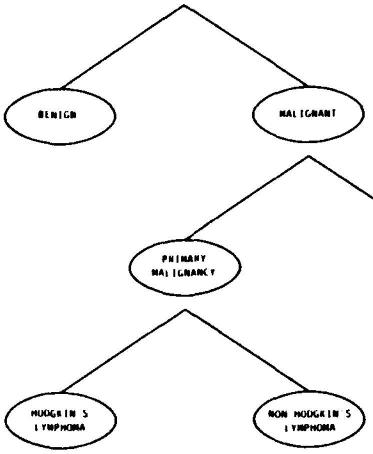


Figure 1: Heuristic problem-solving hierarchy

$$G_1 = \{D_{11}, D_{12}, \ldots D_{1n_1}\}$$

$$G_2 = \{D_{21}, D_{22}, \ldots D_{2n_2}\}.$$

As we assume that only one lymph node disease is present in PATHFINDER, we can consider the diseases to be mutually exclusive events. We are interested in the probability that the true diagnosis will be in each group. To calculate this probability we add the probabilities of all the diseases within each group. That is, the probability that a group contains the true diagnosis is

$$p(G_j) = \Sigma_k \; p(D_{j,k}), \quad j = 1, 2.$$

We can also calculate $p(G_j|E_i)$, the probability of the final diagnosis being contained in a group, considering a new piece of evidence E_i. This is

$$p(G_j|E_i) = \Sigma_k \; p(D_{jk}|E_i), \quad j = 1 \text{ or } 2.$$

Therefore, a relative entropy of the grouped differential can be defined. In particular,

$$H_G(DD, E_i) = \Sigma_j \; p(G_j|E_i) \log[p(G_j|E_i)/p(G_j)].$$

This quantity represents the additional information contained in E_i about the grouped differential diagnosis. Group-discriminate selects those features which give the highest expected relative entropy.

Notice that the group-discriminate strategy ignores information concerning the probabilities of diseases within each group. Only the probabilities that the true diagnosis lies within a group is considered in the calculations.

8. DISCUSSION

We integrated the group-discriminate strategy into the PATHFINDER system so that it continues to refine differential diagnosis lists until all diseases remaining on the differential diagnosis are in a category at one of the leaves of the binary problem-solving tree. At this point, other hypothesis-directed strategies are applied to continue pursuing a diagnosis. As the group-discriminate reasoning strategy has a simpler discriminatory focus and more closely follows the decision making protocol of the expert lymph node pathologist than entropy-discriminate, it is quite easy to explain.

Instead of having to present complex summaries explaining how each piece of evidence might impact on belief in the presence of a number of diseases, an explanation of questions generated by group-discriminate must simply demonstrate how possible responses affect the two groups under consideration.

The PATHFINDER system justifies the usefulness of questions selected by group-discriminate with a graphical display. Fig. 2 presents a small portion of a PATHFINDER consultation. At the top of the figure is the differential diagnosis, grouped into benign and malignant categories (at the current level of refinement). Below, several lymph node features recommended by group-discriminate are listed. The group-discriminate strategy has determined that these features can best discriminate between the benign and malignant diseases. In this case, the user requested explanation for the *follicles density* recommendation.

The positions of a set of asterisks in the justification graph at the bottom of the figure are used to indicate the degree to which each group of diseases is favored by each possible feature value. Specifically, the position of an asterisk is a function of the likelihood ratio $p(E_i|G_1)/p(E_i|G_2)$. In the example, the values *separated* and *far apart* strongly support diseases on the differential diagnosis that are in the benign group, while the values *back-to-back* and *closely packed* strongly support the malignant disease hypotheses.

A user can easily ascertain how a question discriminates among two groups of diseases; evidence is either supportive for one group or the other. Even in an environment filled with distractions, the behavior of the strategy is adequately explained by such simple graphs.

Unfortunately, the more explainable group reasoning strategy has some disadvantages. A predictable problem with the use of group-discriminate is that the differential diagnosis refinement process does not always proceed as quickly as it does with the application of the optimal entropy-discriminate. That is, group-discriminate is not as efficient as the more powerful entropy-discriminate; on average, a larger number of evidence-gathering requests will be made by group-discriminate to achieve a similarly refined differential diagnosis. This must be the case as

```
> ask

Discriminating:

 Malignant

    Small cleaved, follicular lymphoma
    Mixed, small cleaved and large cell,
        follicular lymphoma
    Large cell, follicular lymphoma
    Kaposis sarcoma
    Small noncleaved, follicular lymphoma

 Benign

    Florid reactive follicular hyperplasia
    Reactive hyperplasia
    AIDS


I recommend that the following
features be evaluated:

  Status of follicles
  Follicles density
  Subcapsular sinuses
  Medullary sinuses
  Comparison of cytology inside and
        outside the follicles


> justify

 Which feature do you want justified?

> follicles density

The following table elucidates the
discriminating power of this feature.
The position of the asterisk indicates
which of the two groups of diseases is
favored by each value.

 Malignant
         Benign
 |       |
 v       v
 *........ back-to-back
 *........ closely packed
 .......*. separated
 ........* far apart
```

Figure 2: PATHFINDER consultation

detailed information about the plausibility of individual diseases within each group is discarded in the grouping process.

In general, simplification of an optimal strategy will lead to a less-efficient strategy. Also, given the limits of human cognition identified by research in cognitive psychology, it is not unexpected that a reasoning strategy derived through the constraint or decomposition of a complex problem-solving task may be easier to understand and explain. It seems that for a wide variety of reasoning strategies, there will frequently be an inverse relationship between reasoning understandability and efficiency. In making decisions about alternative reasoning strategies and the clarity of explanation for expert systems, computer scientists may be able to make use of a well-characterized explainability/efficiency trade-off.

9. CONCLUSION

I discussed the usefulness of automated reasoning methodologies that follow from desired fundamental properties and presented an example of the application of a strategy that gracefully degrades complex reasoning of an expert system. The degradation was based in the decomposition of the diagnostic task. The degradation strategy enabled the system to generate transparent justifications for its requests for information, in exchange for a reduction in the optimality of its recommendations.

I believe that continuing research on the pragmatics of applying formal models in the face of severe limitations in data and computation, as well in the abilities of system users will be beneficial. The development and refinement of methodologies for the controlled degradation of reasoning will allow artificial intelligence researchers to build upon the elegant achievements of other disciplines.

Acknowledgements

I am indebted to David Heckerman for many productive conversations. Mr. Heckerman has been an insightful leader of the PATHFINDER Project. I thank Moshe Ben-Bassat, Lawrence Fagan, Ben Grosof, Ted Shortliffe and Peter Szolovits for interesting discussions. I am grateful to Bharat Nathwani and Costa Berard for sharing with me their thoughts on problem solving in pathology. This work was supported in part by the Josiah Macy, Jr. Foundation, the Henry J. Kaiser Family Foundation, the Ford Aerospace Corporation, and the SUMEX-AIM Resource under NIH Grant RR-00785.

References

[1] Ben-Bassat, M., et. al.
Pattern-based Interactive Diagnosis of Multiple Disorders: the MEDAS System.
IEEE Transactions on Pattern Analysis and Machine Intelligence 2:148-160, 1980.

[2] J.S. Bruner, J.J. Goodnow, G.A. Austin.
A study of thinking.
Wiley, 1956.

[3] Buchanan, B. G.
Research on Expert Systems.
In J. Hayes, D. Michie, Y. H. Pao (editors), *Machine Intelligence*, pages 269-299. Ellis Howard Ltd., Chichester, England, 1982.

[4] Buchanan, B. G., and Shortliffe, E. H., eds.
Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project.
Addison-Wesley, Reading, Mass., 1984.

[5] Cheeseman, P.
In defense of probability.
In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence.* IJCAI-85, 1985.

[6] Cox, R.
Probability, frequency and reasonable expectation.
American Journal of Physics 14(1):1-13, January-February, 1946.

[7] Davis, R., Buchanan, B., and Shortliffe, E.
Production Rules as a Representation for a Knowledge-Based Consultation Program.
Artificial Intelligence 8:15-45, 1977.

[8] Davis, R.
Consultation, Knowledge Acquisition, and Instruction.
In P. Szolovits (editor), *Artificial Intelligence In Medicine*, . Westview Press, 1982.

[9] de Finetti, B.
Theory of Probability.
Wiley, New York, 1970.

[10] Elstein, A. S., Loupe, M. J., and Erdman, J. G.
An experimental study of medical diagnostic thinking.
Journal of Structural Learning 2:45-53, 1971.

[11] Elstein, A. S.
Clinical Judgment: Psychological research and medical practice.
Science 194:696, November, 1976.

[12] Elstein, A. S., Shulman, L. S., and Sprafka, S. A.
Medical problem solving: An analysis of clinical reasoning.
Harvard University Press, Cambridge, Mass., 1978.

[13] Gaines, B.R.
Fuzzy and probability uncertainty logics.
Information and Control 38:154-169, 1978.

[14] Gorry, G. A., and Barnett, G. O.
Experience with a Model of Sequential Diagnosis.
Computers and Biomedical Research 1:490-507, 1968.

[15] Gorry, G. A., Kassirer, J. P., Essig, A., and Schwartz, W. B.
Decision Analysis as the Basis for Computer-Aided Management of Acute Renal Failure.
American Journal of Medicine 55:473-484, 1973.

[16] Heckerman, D.E.
Probabilistic Interpretations for MYCIN's Certainty Factors.
In *Uncertainty in Artificial Intelligence*, . North Holland, New York, 1986.

[17] Horvitz, E.J., Heckerman D.E., Nathwani, B.N., and Fagan, L.M.
Diagnostic Strategies in the Hypothesis-Directed PATHFINDER System.
In *Proceedings of the First Conference on Artificial Intelligence Applications*, pages 8. Denver, CO, December, 1984.

[18] Horvitz, E. J., and Heckerman, D. E.
The Inconsistent Use of Measures of Certainty in Artificial Intelligence Research.
In *Uncertainty in Artificial Intelligence*, . North Holland, New York, 1986.
Also available as Technical Report No. KSL-85-57, Knowledge Systems Laboratory, Stanford University.

[19] Horvitz, E.J., Heckerman, D.E., Nathwani, B.N., Fagan, L.M.
The use of a heuristic problem-solving hierarchy to facilitate the explanation of hypothesis-directed reasoning.
In *Proceedings of Medinfo.* Medinfo, October, 1986.
Knowledge Systems Lab Technical Report KSL-86-2, Stanford University, 1986.

[20] Horvitz, E. J., Heckerman, D. E., Langlotz, C. P.
A framework for comparing formalisms for plausible reasoning.
In *Proceedings of the AAAI.* AAAI, Morgan Kaufman, Philadelphia, August, 1986.
Knowledge Systems Lab Technical Report KSL-86-25, Stanford University.

[21] Miller, G.A.
The magical number seven, plus or minus two.
Psychological Review 63:81-97, 1956.

[22] Miller, R. A., Pople, H. E., and Myers, J. D.
INTERNIST-1, An Experimental Computer-Based Diagnostic Consultant for General Internal Medicine.
New England Journal of Medicine 307(8):468-476, 1982.

[23] Pauker, S. G., Gorry, G. A., Kassirer, J. P., Schwartz, W. B.
Toward The Simulation Of Clinical Cognition: Taking A Present Illness by Computer.
American Journal of Medicine 60:981-995, 1976.

[24] Pearl, J.
Fusion, propagation, and structuring in Bayesian networks.
1985.
Presented at the Symposium on Complexity of Approximately Solved Problems, Columbia University, 1985.

[25] Pratt, J. W., Raiffa, H., and Schlaifer, R.
Introduction to Statistical Decision Theory (Preliminary Edition).
McGraw-Hill, New York, 1965.

[26] Raiffa, H.
Decision Analysis: Introductory Lectures on Choice Under Uncertainty.
Addison-Wesley, Reading, Mass., 1968.

[27] Reggia, J.A., Perricone, B.T.
Answer Justification in Medical Decision Support Systems Based on Bayesian Classification.
Comp. Biol. Medicine 15(4):161-167, 1985.

[28] Shafer, G.
A Mathematical Theory of Evidence.
Princeton University Press, 1976.

[29] Shore, J.E.
Relative entropy, probabilisitic inference, and AI.
In *Uncertainty in Artificial Intelligence*, . North Holland, 1986.

[30] Shortliffe, E. H. and Buchanan, B. G.
A model of inexact reasoning in medicine.
Mathematical Biosciences 23:351-379, 1975.

[31] Shortliffe, E. H.
Computer-Based Medical Consultations: MYCIN.
Elsevier/North Holland, New York, 1976.

[32] Simon, H.A.
The Theory of Problem Solving.
Information Processing 71:261-277, 1972.

[33] Spiegelhalter, D.J., and Knill-Jones, R.P.
Statistical and knowledge-based approaches to clinical decision-support systems, with an application in gastroenterology.
J. R. Statist. Soc. A 147:35-77, 1984.

[34] Szolovits, P.
Artificial Intelligence in Medicine.
In P. Szolovits (editor), *Artificial Intelligence In Medicine*, . Westview Press, 1982.

[38] N.C. Waugh, D.A. Norman.
Primary Memory.
Psychological Review 72:89-104, 1965.

[39] Zadeh, L.A.
The role of fuzzy logic in the management of uncertainty in expert systems.
Fuzzy Sets and Systems (11):199-227, 1983.

Computer Applications of Bayesian Statistics in Medicine

Holly Jimison, Stanford University

ABSTRACT

The Bayesian statistical methodology is an especially important technique for computer-assisted medical decision making because there is often a shortage of data directly related to a given clinical decision or classification problem. Also, many clinical relationships are noisy and weakly correlated. Under these conditions a priori information has significant influence on the ultimate classification rule. Bayesian statistics incorporates a priori knowledge and conveniently handles variations in costs of misclassifications with a loss function. This paper shows how Bayesian analysis is appropriate for medical decision making, reviews problems seen with such systems, and provides suggestions and examples of how some systems have addressed these problems.

VALUE OF A BAYESIAN APPROACH

Medicine is a very difficult domain for decision making, and an especially challenging area in which to try and automate this process. Medical decisions are characterized as being important, in that the utilities of the possible outcomes can be dramatically different. Also, the decisions typically need to be make fairly quickly, but with incomplete and noisy data. A priori information becomes an important part of a model in such a situation, and Bayesian analysis provides for explicit representation of both a priori information and utilities. As one considers problems from areas on the continuum going from physics and engineering to biology, physiology, and clinical medicine, the appropriate models for systems in these areas become less deterministic and more ill structured. For most clinical situations, there are an intractable number of confounding factors that may affect a particular variable of interest, in ways that are highly situation-dependent. This leads to weak models with weak correlations. Accurate clinical models often need to make use of associational relationships more than causal or mechanistic relationships. This shift parallels the clinical reasoning used by physicians as they gain clinical experience. Medical students naturally have a tendency to rely on "textbook" knowledge, which is mainly causal and mechanistic. As clinical experience is gained, more use is made of associational knowledge, and much of the reasoning for diagnosis and treatment seems to be pattern-matching. However, causal knowledge is still relied on for explanation and for reasoning about new situations. A statistical model is very appropriate for the pattern-matching reasoning. A Bayesian statistical model is especially appropriate for many medical domains where the relationships between variables are weak and noisy, and when a priori information has more influence. However, causal knowledge is not explicity captured in strictly Bayesian systems, so that there is no natural mechanism for providing a causal explanation of the resulting classification. Some Bayesian systems that incorporate a form of explanation are described later.

Another characteristic of the medical domain is that there is almost always a shortage of statistical data relevant to the patient at hand. A small data set also makes the a priori information more influencial in the classification rule. Again, situations like this favor a model that explicitly incorporates a priori information. A further feature of a Bayesian approach to classification and decision making is that variations in costs of misclassifications are easily incorporated into the decision rule, as shown below. In medical applications it is often not appropriate to assume equal costs of misclassifications, as many other approaches do. Quite often in medicine false negatives are more serious than false positives. Trade-offs between the two types of errors depend upon the specific application.

CONCERNS WITH A BAYESIAN APPROACH

Although Bayesian approaches to medical decision making have certainly been popular for many applications, there are caveats that need to be addressed when designing such a system. What follows is a description of the major concerns with Bayesian systems as well as recommendations on how to rectify the problems.

1. Assumptions Required

The use of Bayes rule in its complete form for the assignment of probabilities to a field of many diseases requires an immense amount of data. The following formula shows that prior and conditional probabilities on combinations of features and diseases are necessary in order to determine posterior probabilities for each disease.

$$P(D_i|F) = \frac{P(D_i)P(F|D_i)}{\Sigma P(D_j)P(F|D_j)}$$

In this formula D_i represents a particular disease and F represents a particular feature vector.

The following assumptions are usually made, not for theoretical reasons, but for simplicity, ease of calculation, and due to lack of data.

a. Conditional independence of features given disease: Classification features are almost always assumed to be independent of one another. This greatly simplifies the calculation of the posterior probabilities and significantly reduces the amount of data collection necessary for such a system. If the individual features are conditionally independent of one another, one need only have data for each feature value given a disease instead of data on each combination of feature values. This provides an exponential reduction of the amount of data required. For a small feature vector **F**=$\{f_1,f_2,f_3\}$ the formula now becomes

$$P(D_i|f_1,f_2,f_3) = \frac{P(D_i)P(f_1|D_i)P(f_2|D_i)P(f_3|D_i)}{\Sigma P(D_j)P(f_1|D_j)P(f_2|D_j)P(f_3|D_j)}$$

Probably the most common criticism of Bayesian algorithms in medicine is that the features used are not conditionally independent, even though the assumption is made. Actually, good system design involves careful feature selection. The choice of features and their program-specific definitions can be optimized using an information metric, such as directed divergence. Indepence can also be tested for by observing the correlation between features. For features that are correlated, the system developer has the option of creating a new single feature that is an index based on some weighting of the correlated features. Thus, there are ways of dealing with the problem of assuming conditional independence, and it is also important to note that other algorithm models, such as rule-based systems, often require conditional independence, even though the assumption is not made explicitly.

b. Diseases are mutually exclusive: Although it's possible to consider all possible combinations of diseases as separate diseases in order to hold to the same Bayesian structure and yet handle combinations of diseases, this approach requires a significant amount of data that is usually not available. Unless diseases are correlated or have a high prior probability, there is not much of a chance of obtaining sufficient data on combinations of diseases. This in itself suggests a solution to the problem of combinations of diseases. That is, if the combination is prevelant enough for there to be data available, then perhaps it should be considered as a separate entity to be diagnosed. Otherwise, the diseases could be assumed to be independent of one another and a meta-diagnostic strategy could be used to say which diseases and disease combinations were actually present. A simple threshold on probability might be used, or one might choose a more complicated strategy like the partitioning algorithm used in the INTERNIST system from the University of Pittsburg.

c. Diseases are collectively exhaustive: This assumption simply means that the model covers the universe of possible events. All possible diseases should be represented, as well as the event "no disease at all." Of course, this is not feasible in a practical system. What is usually done is to work within the context of a smaller domain, covering the diseases or events of interest, and leaving a final class of "other" for any remaining diseases or events. The set then becomes collectively exhaustive.

2. Subjective vs Objective Probabilities

Naturally, objective probabilities, frequency data derived from observations, should be incorporated into a Bayesian system if sufficient relevant data are available. There are many reasons why sufficient relevant objective data may not be available, and subjective probabilities must be obtained from experts. Firstly, data on humans is usually very expensive and difficult to obtain. There may be quality data on rats or mice, but subjective probabilities would be required to modify it for inference about humans. Even a study providing good data on humans is not likely to perfectly match a given patient in question. A clinician will want to subjectively modify probabilities to account for patient-specific factors. More generally, causal knowledge about disease processes needs to be encoded subjectively if its effect has not been accounted for in observed data from a study. Another situation where subjective evaluation of probabilities for a Bayesian system is necessary comes when one tries to incorporate data from different studies. The results may conflict, definitions or study designs may be different, the population sampled may be quite different, etc. The synthesis of these results is necessarily quite subjective, or at least heuristic. There have been Bayesian systems of both types that have provided expert level performance. For example, de Dombal's system for acute abdominal pain used observed frequencies from a teaching data set for its system's probabilities (the performance degraded when experts' subjective probabilities were used), and Gorry's system for management of acute renal failure achieved expert performance using subjective probabilities from experts.

Often clinicians find it easier to provide quality information when asked to estimate prior odds and likelihood ratios. This is appropriate for the odds form of Bayes rule.

$$\frac{P(D|F)}{P(notD|F)} = \frac{P(D)}{P(notD)} * \frac{P(F|D)}{P(F|notD)}$$

3. Learning of Priors and Conditionals

A priori probabilities and conditional probabilities obtained for use in Bayesian systems must be verified and possibly modified for application in different patient populations (new locations, new clinics, etc.). Also, the probabilities may change in time with changes in lifestyles, environmental factors, treatments, and general disease characteristics. Ideally, a system would be able to learn and update these probabilities on its own. One fairly common mistake, that needs to be avoided, is using the system's own classification or diagnosis on each event as data for calculating new probabilities. This type of learning is decision-directed learning, and the problem with this is that mistakes propagate mistakes. In fact, it is possible to equilibrate with inaccurate probabilities that produce poor performance. One solution is to have the classifications checked by a human, trying to avoid the bias of knowing the machine's classification ahead of time. Another solution that provides an automatic update of the probabilities, is to have a totally independent classification algorithm just used for updating purposes. At first glance this seems like at least double the effort in designing a system, but often there are tests (or features) that are very sensitive but not specific. These would not be that useful for classifying each event, but very useful for updating the conditional probabilities. In other words, $P(F|D_i)$ does not have to updated every time D_i is diagnosed. The feature, or set of features, that were very specific but not sensitive could be used to pick out cases that were especially likely to be D_i and update $P(F|D_i)$ only on those cases. Forbes et. al. at Hewlett-Packard Laboratories did just that in a Bayesian computer algorithm to classify ambulatory electrocardiogram waveforms as being normal or abnormal. For this application, each heart beat has to be classified in real time. They use features of the waveform, such as polarity, amplitude, width, phase, etc., to classify each beat. The program is initialized with conditional probabilities based on physiological principles and general observations, but since the system allows for arbitrary lead placement, the shape of the normals must be learned very quickly. An independent algorithm comparing relative widths and time intervals between beats is used to pick out a subset of the beats that are especially likely to be normal. These are used to update P(F|normal). It is important for this application that the learning continues, since the waveform shapes of the beats can change with patient movement. They chose to keep the conditional probabilities for the abnormal beats at their default values because there are many types of possible abnormal beats. Also, since cost of misdiagnosis and prior probability for each class is inversely related and difficult to assess, that portion of the decision rule was also left static at values that optimized performance. This system has been shown to be very successful at smoothly and quickly adapting to changes in the shape of normal beats. The overall point on automating the learning of probabilities for Bayesian systems is that one should think of using independent classification algorithms that are optimized for updating.

4. Explanation

Perhaps the most important feature of a medical decision aid is its ability to provide quality explanation of the inference and reasoning used in coming to a decision or diagnosis. This is a major focus of expert systems research, but not a comfortable concept for designers of statistical tools. For complicated medical decisions, physicians are not generally content to be shown a list of regression weights as justification for some sort of action on their part. Even proven good performance does not always inspire confidence, since each new case is seen as having some unique factors. Physicians are most comfortable with explanations that simulate their own reasoning strategies. A couple of researchers have addressed this problem, and have provided explanation capabilities with their Bayesian systems. In the gastroenterology decision-support system of Spiegelhalter and Knill-Jones there is a display that provides a complete summarization of the decision rule in an easily understandable format. Evidence for a disease is listed on one side and evidence against the disease is listed on the other. Each feature has an accompanying log-likelihood ratio that serves as its score. If the score is positive, then the feature and score go under "evidence for" the disease. Otherwise, it is evidence against the disease. Since the log of the likelihood ratios are shown, the scores add, and it is easy to see the relative importance of each finding. The a priori probability of disease is also shown in log form and is added in to obtain a final score. This is converted into the resulting probability of disease. "Evidence for" and "evidence against" is a natural representation of information for physicians. The scoring may appear ad hoc to the casual user of the system, but to those that understand Bayesian analysis, a complete representation of the reasoning is presented in a useful way. Sometimes system users may be put off by the amount of detail provided by such a scoring scheme and may prefer to have it represented graphically in a histogram.

Reggia and Perricone came up with another form of explanation for their Bayesian system to classify strokes. After acquiring values for the features probabilities for the various types of strokes are shown. The user has the option to justify any of them. Justification includes an optional explanation of Bayes formula, and a list of the features, their values, and their scores. Also provided is a list of features with unknown values that might alter the results if their values were known.

In general, it seems possible to provide a good explanation and useful summarization of the analysis in a Bayesian system. However, the explanations are by necessity associational, and we need to keep in mind that humans usually reason causally for explanation, trying to find a mechanism or process that explains the observations.

SUMMARY

The advantage of using a Bayesian approach to medical decision making is that a priori information and costs of misdiagnoses can be represented explicitly and easily. The concerns usually posed regarding Bayesian systems do not seem insurmountable. Three assumptions are usually made for practical reasons: conditional independence of features, mutual exclusivity of diseases, and the assumption that the set of diseases are collectively exhaustive. It was shown that with careful design, the problems with these assumptions could be avoided. It was also shown that subjective probabilities could be used in a Bayesian system giving expert level performance. Finally, automated learning of probabilities and explanation features are new areas for Bayesian systems, and the systems reviewed suggest mechanisms for significantly improving the performance and acceptability of Bayesian systems for medical applications.

REFERENCES

Atkinson, P., Training for Certainty, *Soc. Sci. Med.*, Vol.19, No. 9, pp. 949-956, 1984.

Clancey, W.J., Shortliffe, E.H., *Readings in Medical Artificial Intelligence: The First Decade.* Menlo Park, CA: Addison-Wesley.

Forbes, A.D., et. al., A Dual Channel Bayesian Algorithm for Ambulatory Electrocardiogram Analysis, *Proceedings of Computers in Cardiology*, 1986.

Pople, H.: The formation of composite hypotheses in diagnostic problem-solving: An exercise in synthetic reasoning. In *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, pp. 1030-1037. Pittsburgh, PA: Carnegie-Mellon University, Department of Computer Science. 1977.

Reggia, J.A., Perricone, B.T., Answer Justification in Medical Decision Support Systems Based on Bayesian Classification, *Comput. Biol. Med.*, Vol. 15, No. 4, pp. 161-167, 1985.

Reiss, Eric: In Quest of Certainty. *Am J of Med*; 1984; 77:969-971.

Spiegelhalter, D.J., Knill-Jones, R.P.: Statistical and Knowledge-based Approaches to Clinical Decision-support Systems, With an Application in Gastroenterology. *J R Statist Soc A*; 1984; 147:35-77.

CONTROLLING GRAPHICS

Organizer: *Paul Velleman, Cornell University*

MacSpin: Graphical Data Analysis

Andrew W. Donoho, University of Texas, Austin; David L. Donoho, University of California, Berkeley; Miriam Gasko, University of Chicago

GRAP—A Language for Statistical Displays

Jon L. Bentley, Brian W. Kernighan, AT&T Bell Laboratories

MACSPIN: GRAPHICAL DATA ANALYSIS

Andrew W. Donoho, University of Texas, Austin
David L. Donoho, University of California, Berkeley
Miriam Gasko, University of Chicago, G.B.S.

Over the last decade, computer graphics researchers have developed systems for the dynamic display of data. These interesting and stimulating displays showed the potential usefulness of interactive graphics in the analysis of multivariate data. Videotapes of such displays showed instantaneous graphical responses to researchers' queries. However, these systems -- like PRIM-9 (SLAC) and PRIM-H (Harvard) and the Orion (Stanford) -- were one-of-a-kind installations accessible to only a very few people. Because of their expense, there was not much prospect of widespread use of such systems; as long as their primary purpose was seen as the probing of theoretical horizons and the production of videotapes to take to conferences, such important issues as ease-of-use and capability for working with real data on an everyday basis could be ignored.

A useful graphics system must have the following properties: first, it has to offer a graphical "toolbox" adequate for interactive exploratory data analysis. Second, its user interface has to be simple to operate, simple enough not to distract the user from his main task, analyzing the data at hand. Last, but not least, it must have input facilities that accomodate the researcher's existing files and output facilities that help him document his findings. We describe a dynamic graphics system, MacSpin, which meets these requirements and runs on an inexpensive desktop computer -- the Macintosh.

MacSpin has advanced graphics capabilities. It goes beyond two-dimensional x-y plots, and lets you view and interact with data in 3 dimensions -- and more. You can view x-y-z plots, rotating them in real time to get a true three-dimensional perception of the structure of your data. By means of animation, you can make movies of your data showing how the three-dimensional cloud varies as a function of a fourth variable. MacSpin is a useful tool for identifying trends, patterns, clusters and outliers in high-dimensional data. The capability for augmenting the display with text information allows one to identify datapoints (e.g. outliers) and correlate qualitative information with the patterns you observe in the display.

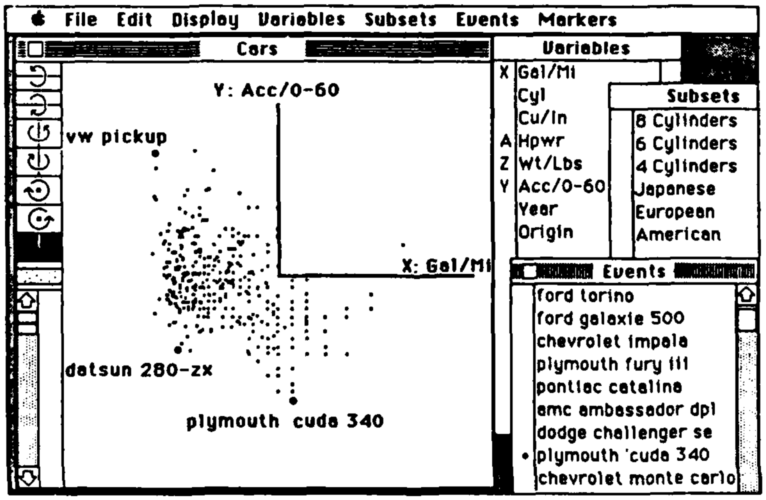
MacSpin offers easy modes for inputting data and for porting data from other programs and mainframes. It also has useful documentation capabilities, that allow one to produce hard copy of screen views, or to insert such images into other computer documents.

And all these operations can be carried out by MacSpin with the user simply pointing and clicking with the mouse. Here we illustrate the main features of the program by telling the story of its application to two datasets.

The Cars Story

The first dataset we consider consists of all the cars road-tested by Consumer Reports magazine between 1971 and 1983. The data will help us see how the auto industry changed over the last decade or so. The names of the 418 Cars are listed in the events window on the MacSpin display (lower right): the portion we see includes the Plymouth Barracuda and Plymouth Fury III, cars from the early 1970's. The variables window (partially obscured by the events window) shows the variables we have measured for each car; things involving performance (Gallons per Mile, Seconds to reach 60 MPH from a full stop), size (horsepower, weight, ...), and miscellaneous (model year, continent of origin).

X-Y Plots. The view in the plot window shows all the cars (American, European, and Japanese) in an x-y plot, with x=Gal/Mi (i.e. fuel usage per mile) versus y=slowness (Secs. 0-60). The points represent individual cars. By moving the cursor to a point and clicking, we can find out its identity. The point at the upper left (slow but economical) is a VW pickup; the point in the lower right (fast gas-guzzler) is a Plymouth Barracuda. By holding down the control and option keys as we



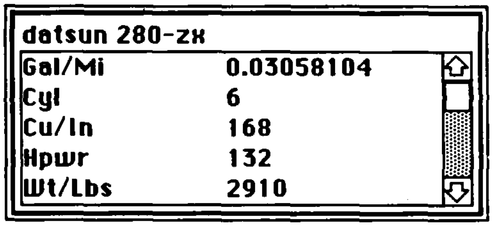
X Gal/Mi
Cyl
Cu/In
A Hpwr
Z Wt/Lbs
Y Acc/0-60
Year
Origin

8 Cylinders
6 Cylinders
4 Cylinders
Japanese
European
American

ford torino
ford galaxie 500
chevrolet impala
plymouth fury iii
pontiac catalina
amc ambassador dpl
dodge challenger se
• plymouth 'cuda 340
chevrolet monte carlo

Figure 1: Gasoline consumption and acceleration time for 418 cars reviewed by Consumer Reports between 1973 and 1981.

identify, the full data record pops up. This shows us that the fast, efficient car in the lower left is a 1981 Datsun 280 ZX.



datsun 280-zx

Gal/Mi	0.03058104
Cyl	6
Cu/In	168
Hpwr	132
Wt/Lbs	2910

Figure 2: Information pop-up for the Datsun 280zx.

X-Y-Z Plots. The x-y plot shows the general trend of the auto industry -- what combinations of speed and economy are available. By rotating the plot, we can get an extra dimension into the display. We point at a rotation icon on the far right, and hold down the mouse button. The 2-d plot becomes a rotating 3-d plot, the previously hidden z-variable coming into play. When we do this for the Cars data, and bring z=weight into the display, we see a cloud of points rotating smoothly in space. The cloud is shaped like a sausage and shows the combinations of economy, speed, and weight being built during the 1971-1983 period. As we rotate this, we notice a few interesting things. First, one point turns out to be an outlier. We stop the rotation and identify it; it is an International Harvester truck. Somehow a truck has slipped in to a database on Cars! When we scroll to the truck's name in the events window, we see that Consumer Reports road-tested a few other trucks, too. By pointing at their names on the list, we can highlight them in the plot window. They are also outliers. By choosing "Exclude" from the events window, we can (temporarily) remove them from the display. The rotation has helped us identify and remove outliers.

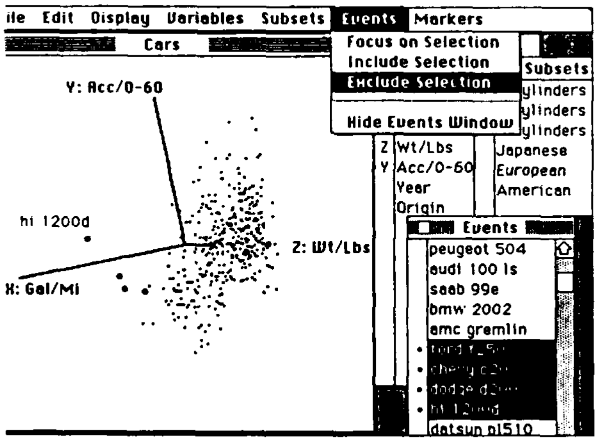


Figure 3: Highlighting and excluding trucks from the display.

Highlighting Subsets. Further rotation shows that the data consist of three clusters. Seeking for an explanation, we bring the Subsets window to the front. This shows some subsets of the data predefined (by us) as being interesting to look at. By pointing at the name of any subset, we can highlight its members on the display. When we do this, we see that the 3 clusters consist of 8, 6, and 4 cylinder cars, respectively. We could also highlight American, European, and Japanese subsets in turn, and find out where they are on the display.

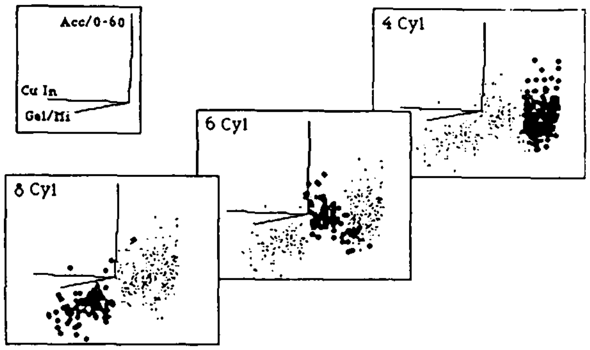


Figure 4: Highlighting subsets of 4, 6, and 8 cylinder cars.

Animation permits us to study the effect of a fourth variable on a display. Suppose we are interested in how the American auto industry has changed over time. We can highlight the American cars, and then select Focus from the events menu. MacSpin now temporarily excludes imported cars from the display. We then drag the Year variable to the scroll bar in the lower left. This will let us scroll through the data model-year by model-year. We begin at 1971. The cars made then are concentrated in the lower left of the display: fast, heavy, gas-guzzling cars. As we scroll smoothly forward, we see that the data drift systematically towards the upper left -- toward slower, lighter, more economical machines.

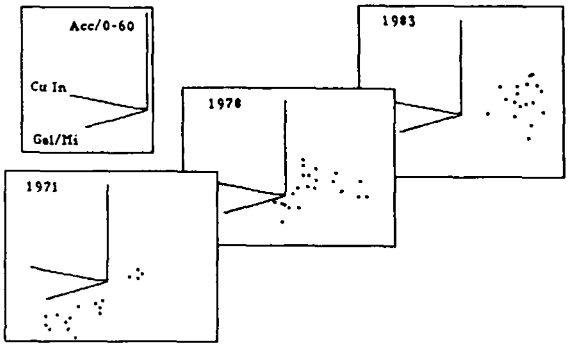
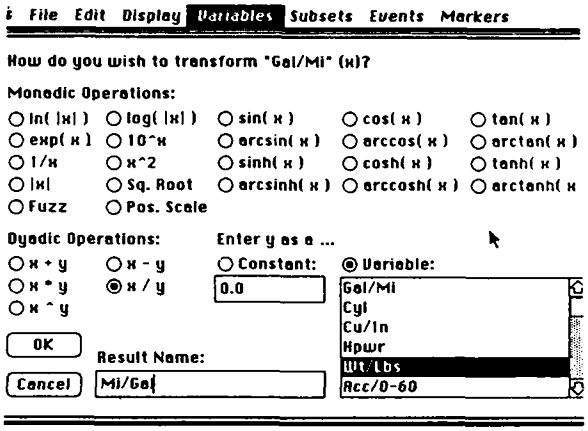


Figure 5: Animation showing changes in the performance of American cars over time: the years 1971, 1978, and 1983 are shown.

Transformations. The researcher can also transform existing variables to create new ones. Features like this make MacSpin useful not just for displaying data but also for manipulating it to get the right display. We just saw that cars became more economical over the period 1973-1981. Did they just become lighter and smaller, or was there an actual increase in mechanical efficiency? Dividing Gal/Mi by Weight gives us a standardized measure of fuel efficiency in which the effects of weight are taken out. Looking at plots with this new variable would give us insight into whether American cars got more efficient or whether they just got smaller over this period. Variable transformations are all included in a special transformations window, and executed by pointing and clicking with the mouse.



How do you wish to transform "Gal/Mi" (x)?

Monadic Operations:

○ ln(|x|) ○ log(|x|) ○ sin(x) ○ cos(x) ○ tan(x)
○ exp(x) ○ 10^x ○ arcsin(x) ○ arccos(x) ○ arctan(x)
○ 1/x ○ x^2 ○ sinh(x) ○ cosh(x) ○ tanh(x)
○ |x| ○ Sq. Root ○ arcsinh(x) ○ arccosh(x) ○ arctanh(x
○ Fuzz ○ Pos. Scale

Dyadic Operations:

○ x + y ○ x - y
○ x * y ● x / y
○ x ^ y

Enter y as a ...

○ Constant: 0.0

● Variable: Gal/Mi, Cyl, Cu/In, Hpwr, Wt/Lbs, Acc/0-60

Result Name: Mi/Gal

OK Cancel

Figure 6: Transformations menu window.

Markers. MacSpin also makes it easy to get hard copy of data displays. (Since you can mark subsets with special symbols, you can use these to convey some of what the dynamic exploration showed you.) "Screen dumps" are generated using the Command-Shift-3 sequence. The figure below is derived from a screen dump. 1971 model cars are marked with a box, and 1983 model cars with an asterisk. The resulting image was cropped, and shadows and captions were drawn in, using MacPaint.

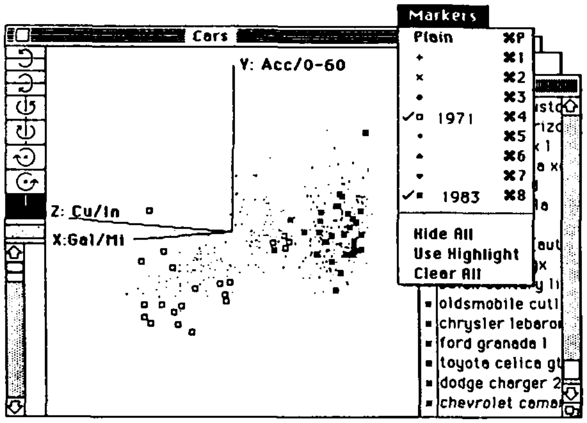


Figure 7: American cars, with special markers given to model years 1971 and 1983.

The Diabetes Story.

These data were provided by Reaven and Miller of Stanford University. The original graphic analysis of it was done on the PRIM-9, and reported in *Diabetologica* in 1979. The data describe 145 nonobese and nonketotic patients who agreed to participate in a medical experiment. The purpose was to assess the relationships among various measures of plasma glucose and insulin in order to illuminate the etiology of "Chemical" and "Overt" diabetes. Each patient underwent a glucose tolerance test, and the following quantities were measured: Age, Relative Weight, Fasting Plasma Glucose, Test Plasma Glucose (a measure of insulin intolerance), Steady State Plasma Glucose, and Plasma Insulin during the test. In addition we have the doctors' classification of these patients as overt diabetic, chemical diabetic, or normal. Age and Relative Weight turned out to be unimportant hence are excluded from our analysis (as they were by Reaven and Miller).

The opening view of our demonstration has the following variables assigned to the three axes: Fast Glucose on the X-axis, Test Glucose on the Y-axis and Test Insulin on the Z-axis. (Figure 8).

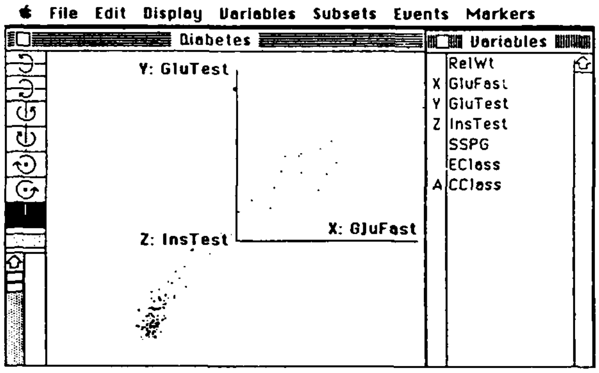


Figure 8: Glucose measurements of 145 patients who underwent a glucose tolerance test.

This view, showing the data distributed in a sausage-shaped cloud, supports the interpretation that there is but one direction in which abnormality develops, as we progress form normal patients to Chemical to Overt diabetics. However, as soon as we start to rotate the data around the X-axis, and tilt it a bit to better show the third dimension, Z (Figure 9), we can see the pointcloud has, in fact, the shape of a boomerang. We can no longer accept that there is just one direction of disease developoment.

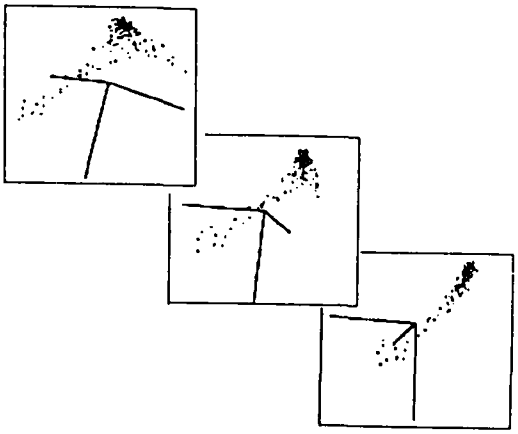


Figure 9: Rotation showing the boomerang aspect of the data.

The most natural question at this point is: What makes the two arms of the boomerang? As the doctors have classified the diabetic patients as either Chemical or Overt, we can highlight each subset separately. As Figure 10 shows, each arm corresponds to one of the groups. We can also mark the points corresponding to Overt diabetics with x's, and points of Chemical diabetics with diamonds (Figure 11).

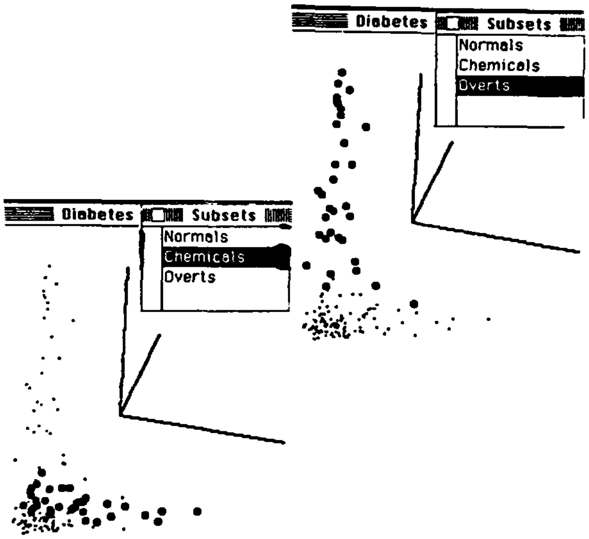


Figure 10: Chemical and overt diabetics shown occupying the two arms of the boomerang.

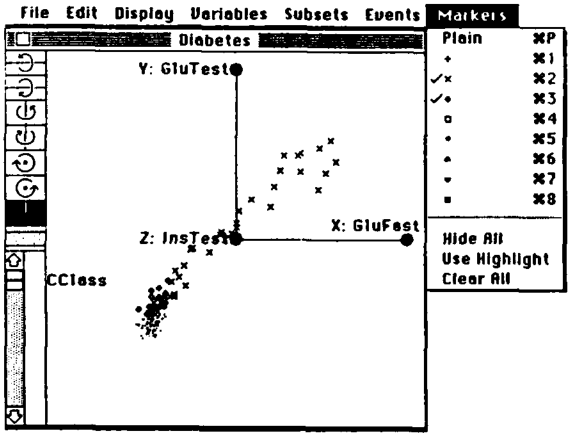


Figure 11: The glucose measurements with the "Chemical" and "Overt" subsets marked.

Whatever our notation, our conclusion is that Chemical and Overt diabetes are two different syndromes, not just one manifested at different levels of intensity.

The examples show MacSpin's usefulness in exploratory data analyis: how its dynamic graphics can reveal data structures and the answers to focused questions about data. Among the important features we illustrated were:

- rotation to show a third dimension
- identifying interesting points
- highlighting important subsets
- animation to look at a fourth variable
- transforming the data
- marking subsets

Our live demonstrations of these examples at the Interface conference testified to the system's ease of use. While MacSpin is not a replacement for standard statistical procedures (and, hence, has been designed to facilitate the porting of files between programs and mainframes), it is a valuable addition to the data analyst's "kit of tools".

REFERENCES

Becker, R.A. and Cleveland, W.S. (1984). Brushing a Scatterplot Matrix: High Interaction Graphical Methods for Analyzing Multidimensional Data. Manuscript.

Fisherkeller, M.A., Friedman, J.H., and Tukey, J.W. (1974). An Interactive Multidimensional Data Display and Analysis System. SLAC PUB 1408. Stanford Linear Accelerator Center, Stanford, CA.

Friedman, J.H. and Tukey, J.W. (1974). A Projection Pursuit Algorithm for Exploratory Data Analysis. *IEEE Transactions on Computers* IC-23.

MacDonald, John A. (1982). *Interactive Graphics for Data Analysis.* Thesis, Stanford University.

Reaven, G. and Miller, R. (1979). An Attempt to Define the Nature of Chemical Diabetes Using a Multidimensional Analysis. *Diabetologica* 16, 17-24.

GRAP — A Language for Statistical Displays

Jon L. Bentley
Brian W. Kernighan

AT&T Bell Laboratories
Murray Hill, New Jersey 07974

ABSTRACT

This paper describes Unix® tools for preparing publication-quality graphical displays. A general-purpose graphing language, GRAP, provides for automatic scaling and tick marks, input data transformation and processing, multiple independent coordinate systems, and multiple graphs in a single display. Specialized languages (implemented as GRAP preprocessors) deal with specialized graphs, such as dotcharts, box plots and scatter-plot matrices.

Although originally designed for document preparation, GRAP has been used for such diverse tasks as exploratory data analysis and prototyping new graphical displays.

1. Introduction

The Unix* operating system includes a family of tools for document preparation. The basic tool is a venerable text formatter called TROFF. That formatter does not deal directly with complicated material like mathematics and tables. Instead, specialized kinds of typesetting are handled by preprocessors that translate specialized languages into TROFF commands. For example, a language called EQN translates expressions like

```
X bar = 1 over n sum from i=1 to n f sub i x sub i
```

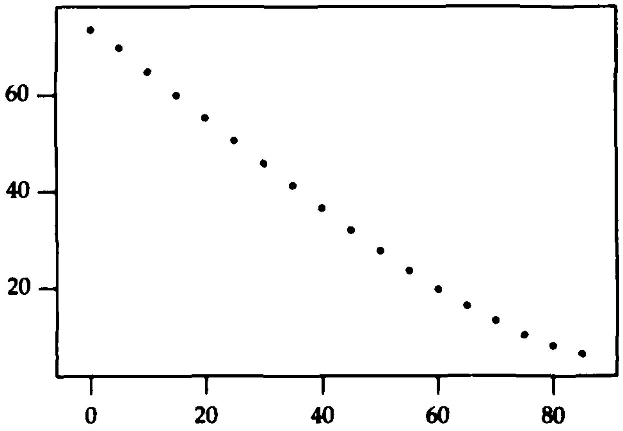
into $\overline{X} = \frac{1}{n} \sum_{i=1}^{n} f_i x_i$. Other languages include TBL for specifying tables, and PIC for drawing simple line diagrams. The Unix document-preparation tools are described in [1]; a survey of the field can be found in [2].

One area not served by the suite of programs mentioned above is the graphical display of data. In most document preparation systems, the only way to include a graph is by (mechanical or electronic) cutting and pasting of a separately prepared figure. The GRAP language [3, 4] was designed to make it easy to describe graphs and to include them in documents prepared with TROFF and related programs. This paper was typeset by those tools, without benefit of scissors or paste, physical or electronic.

This paper will describe the elements of the GRAP language, and illustrate its use as a vehicle for experimenting with new forms of statistical display and packaging them for convenient use.

2. The GRAP Language

In its simplest use, GRAP converts a set of x,y pairs into a scatter plot, generates ticks automatically, and puts the result in a standard frame. Given pairs showing remaining life expectancy as a function of age, GRAP produces this (simple) plot:



A graph is often part of a larger document. The parts of the document intended for GRAP are delimited by the commands `.G1` and `.G2`; other text is copied through untouched. The input for the graph above is just the data itself (the ellipsis marks omitted data items):

[*] Unix is a trademark of AT&T Bell Laboratories.

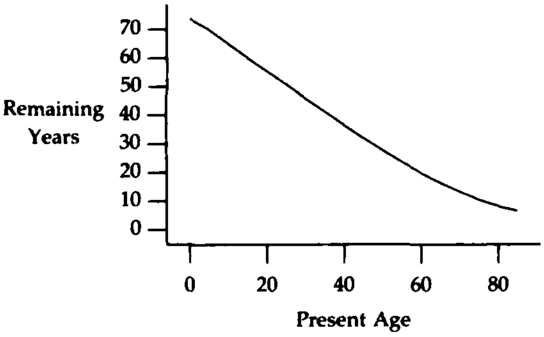
```
.G1
0   73.6
5   69.8
10  64.9
...
85  6.5
.G2
```

GRAP translates graph specifications into PIC commands. To format a document containing graphs, one would normally use a pipeline of commands such as

```
grap filenames | pic | troff
```

The default display may be refined by specifying more parameters. Labels may be added on any side, ticks may be defined by an explicit list or an iterator, data may be copied from a separate file, and the points may be connected by lines of various styles:

```
label bottom "Present Age"
label left "Remaining" "Years" left .1
ticks left from 0 to 70 by 10
frame ht 1.3 wid 2 top invis right invis
draw solid
copy "life.d"
```



The file `life.d` contains the age-expectancy data shown above. The clause `left .1` moves the text from its default position by that many inches. The `.G1` and `.G2` delimiters are not shown in this and subsequent inputs.

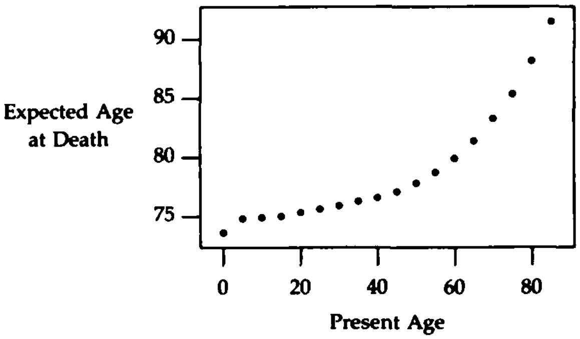
The core of GRAP includes commands for plotting arbitrary text at any point, drawing arbitrary lines and arrows, setting the range and optional logarithmic scaling of coordinate axes explicitly, and drawing grid lines.

GRAP does not provide a large variety of built-in graph types. Rather, it offers primitive operations out of which many different graphs can be built. One of .he most important of these primitive operations is a simple macro processor. The statement

```
define name { replacement text }
```

defines a macro. Subsequent occurrences of *name* will be replaced by the *replacement text*. Instances of `$1`, `$2`, etc., in the replacement text will be replaced by the corresponding arguments in a macro call like *name*(arg_1,arg_2,...).

To illustrate, consider plotting expected age at death rather than remaining years, for which the *y* coordinate is the sum of age and expectancy:



```
frame ht 1.3 wid 2
label bottom "Present Age"
label left "Expected Age" "at Death" left .1
define show { bullet at $1, $1+$2 }
copy "life.d" through show
```

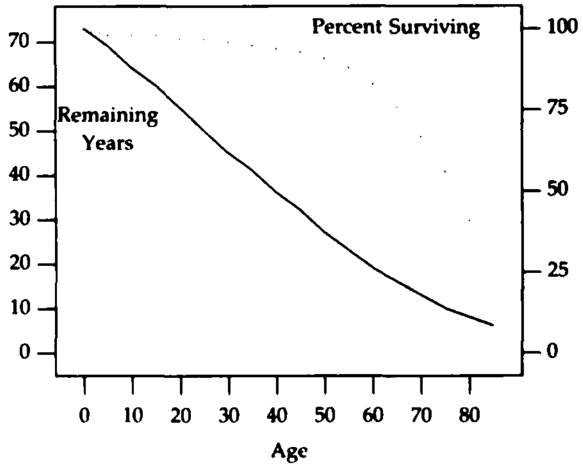
In a copy statement, each line of the source file is converted into a call of the specified macro, with each field becoming the corresponding argument. In fact, it is not necessary to define the macro separately:

```
copy "life.d" through { bullet at $1, $1+$2 }
```

is equivalent, and notationally more convenient.

As this example suggests, GRAP provides the ability to do arithmetic, both on input data and on variables. It also has an `if-else` statement and a `for` loop.

It is possible to show multiple curves on a single plot; each set of values is independently scaled and plotted. For example, this graph plots a second set of data that shows the fraction of an original 100 people still alive at the given age:

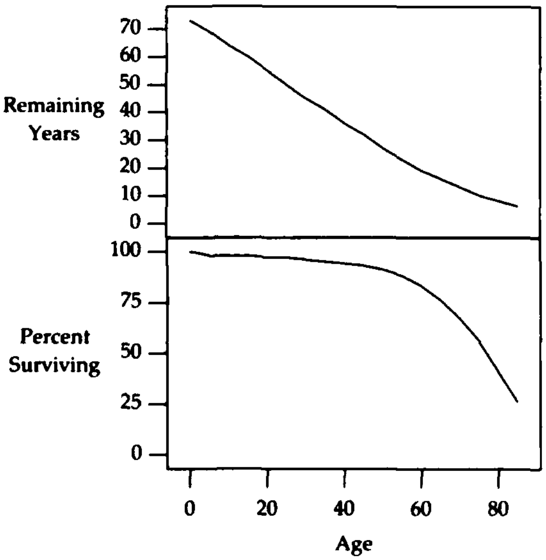


```
frame wid 2.5
label bot "Age"
ticks bot from survivors 0 to 80 by 10
ticks left from expectancy 0 to 70 by 10
ticks right from survivors 0 to 100 by 25
draw expectancy solid
draw survivors dotted
copy "life3.d" through {
    next expectancy at expectancy $1, $2
    next survivors at survivors $1, $3
}
"Percent Surviving" at survivors 65, 100
"Remaining" "Years" at expectancy 5, 50
```

Data or parameters may be plotted in a particular coordinate system by placing the name of that system before the scalar value or x,y pair.

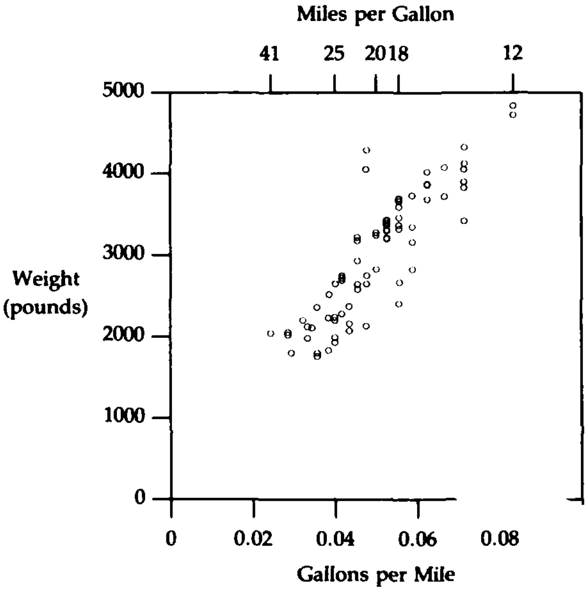
One of the most useful features of GRAP is the ability to combine several subgraphs into one overall graph. As a simple example, the life expectancy and survivor data above may be plotted as two separate graphs with a common x axis:

```
graph Exp
    frame ht 1.25 wid 2
    ticks left from 0 to 70 by 10
    tick bottom off
    label left "Remaining" "Years" left .1
    draw solid
    copy "life3.d" through { $1, $2 }
graph Frac with .Frame.north at Exp.Frame.south
    frame ht 1.25 wid 2
    ticks left from 0 to 100 by 25
    label left "Percent" "Surviving" left .1
    draw solid
    copy "life3.d" through { $1, $3 }
    label bottom "Age"
```



The **graph** statement defines a subgraph with its own coordinate systems, data, etc. Subgraphs may be positioned arbitrarily with respect to previous subgraphs using PIC positioning commands such as **with**.

The examples above show that GRAP gives the user a great deal of freedom in preparing x,y plots in standard forms. It has also proven to be a useful tool for experimenting with display formats. The file **cars.d** contains the mileage (miles per gallon) and the weight (pounds) for 74 models of automobiles sold in the United States in the 1979 model year. A simple scatter plot shows that as mileage increases, weight decreases nonlinearly. A more interesting graph shows that inverse mileage (gallons per mile) is proportional to weight.



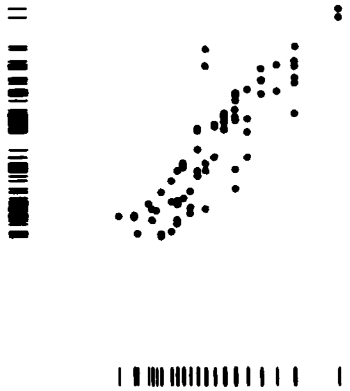
The top ticks denote the extremes, quartiles, and median. The graph was generated by

```
frame ht 2.2 wid 2.2
coord x 0, 0.1 y 0, 5000
label left "Weight" "(pounds)" left .2
label bot "Gallons per Mile"
ticks bot from 0 to 0.10 by 0.02
label top "Miles per Gallon"
ticks top at 1/12 "12", 1/18 "18", \
  1/20 "20", 1/25 "25", 1/41 "41"
copy "cars.d" thru { circle at 1/$1,$2 }
```

In [5], Tufte proposes the "dot-dash-plot" as a means for maximizing data ink (showing the two-dimensional distribution and the two one-dimensional marginal distributions) while minimizing what he calls "chart junk" — ink wasted on borders and non-data labels. His graph is easy to execute in GRAP:

```
frame invis ht 2 wid 2
coord x 0, 0.1 y 0, 5000
copy "cars.d" thru {
  tx = 1/$1; ty = $2
  bullet at tx,ty
  tick bot at tx ""
  tick left at ty ""
}
```

which produces:



Although visually attractive, we do not find the resulting graph as useful for interpreting the data as the first representation. Tufte's graph, however, does point out two facts not obvious in the previous format: there is a gap in car weights near 3000 pounds (exhibited by the hole in the *y*-axis ticks), and the gallons per mile axis is regularly structured (the ticks are the reciprocals of an almost dense sequence of integers).

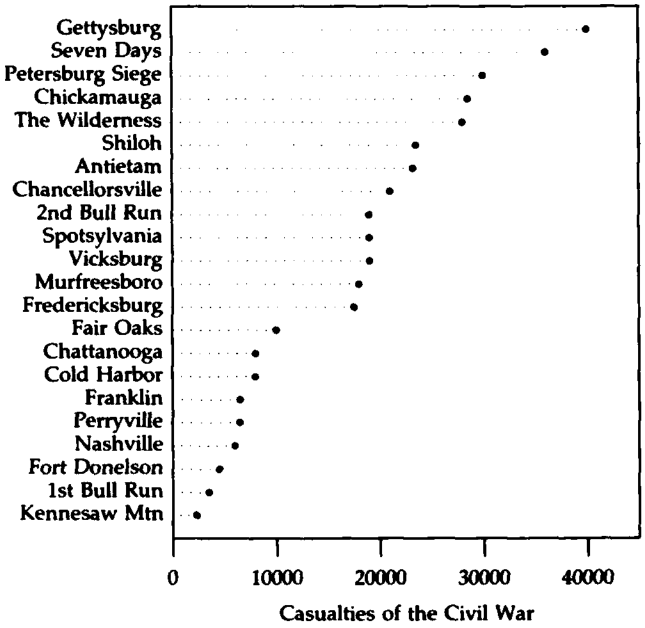
A word on implementation: GRAP is implemented as a preprocessor for PIC so as to take advantage of PIC's features for plotting and positioning text. GRAP itself handles collection of data, maintains the independent input coordinate systems, and scales the outputs in each.

The language is specified with a YACC grammar and the processor is written in C; it is about 3000 lines of code altogether. GRAP went from initial conception to use by people other than the authors in about a week and to books and published papers within several months. The total software development time is perhaps three or four person-months.

3. A Language for Dotcharts

Macros provide one way to encapsulate a complicated or lengthy sequence of GRAP commands. More interesting, however, is the notion of a "little language." If a particular class of graph is used frequently, one can design a small language for describing instances of the class, and implement a "compiler" that translates from that specialized language into GRAP statements. In that case, GRAP serves as an assembly language.

The simplest example is a language for describing "dotcharts" or "lolliplots" [6]. The following dotchart shows total Northern and Southern casualties (killed and wounded) in the major battles of the Civil War.



This straightforward GRAP program produces that dotchart:

```
label "Casualties of the Civil War"
coord x 0 to 45000
ticks left off
nr = 0
copy "civwar.d" through {
    nr = nr + 1
    yval = -nr
    line dotted from 0,yval to $1,yval
    bullet at $1,yval
    $2 rjust at -0.02,yval
    lastx = $1
}
"" at 0,0; "" at 0,-(nr+1)
frame ht (nr+1)*0.125 wid 2.5
```

The data items are counted as they are printed; the frame height is computed and the frame drawn after the data has been plotted.

If one is preparing only a few dotcharts, each can be built with a text editor. If there are a large number of similar graphs to be prepared, however, it is probably worth automating the job. We therefore designed a DOTCHART language in which one can specify a large class of dotcharts. The DOTCHART "compiler" reads a dotchart specification and generates GRAP commands (only slightly more complex than those above) to print the desired figure. For example, the dotchart above was specified as

```
file "civwar.d"
label "Casualties of the Civil War"
spread .125
coord x 0 to 45000
width 2.5
quoted
```

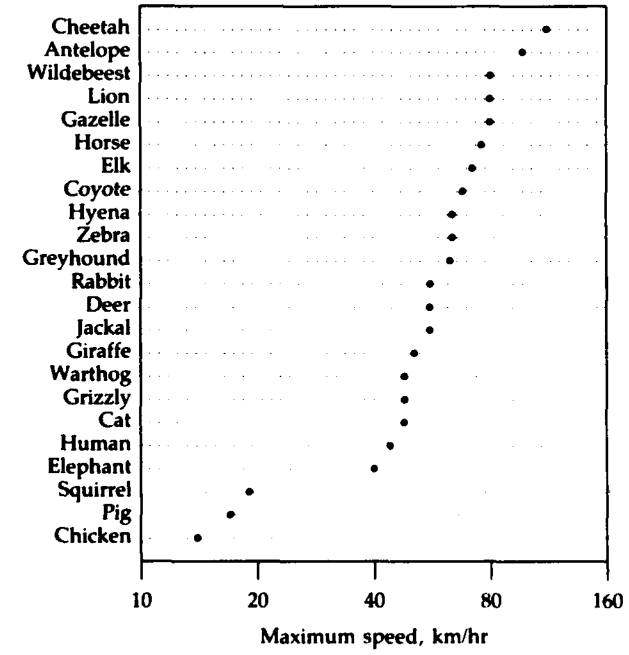
The file command specifies that input comes from a file civwar.d:

```
40000   "Gettysburg"
36000   "Seven Days"
30000   "Petersburg Siege"
28500   "Chickamauga"
28000   "The Wilderness"
...
2270    "Kennesaw Mtn"
```

Other commands set parameters of the dotchart as needed, and any remaining lines (such as label and coord) are assumed to be GRAP commands that make sense in context; they are copied through verbatim.

The implementation of DOTCHART is noteworthy mainly for its small size — the first version, adequate for dotcharts like the one above, is less than 25 lines long (see [3]). It is written in AWK, a general-purpose string processing language [7].

With the basic design of DOTCHART in hand, it is easy to add features that express variations on the theme. For example, Cleveland advocates dotted lines that go all the way across the chart when the baseline is at zero. Four lines of AWK code in the DOTCHART compiler and another parameter in the language implement the new style guide across:



```
file "animal.d"
label "Maximum speed, km/hr"
coord x 10 to 120 log x
ticks bot at 10, 20, 40, 80, 160
spread .125
width 2.5
guide across
```

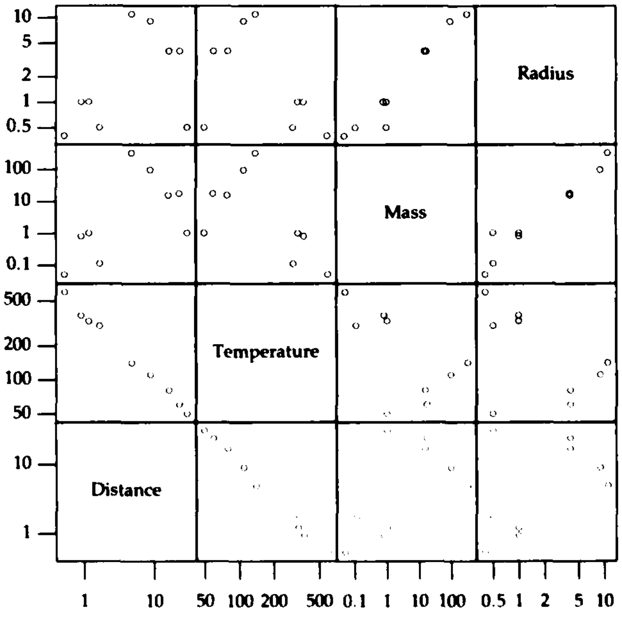
4. Other Specialized Languages

DOTCHART is not the only little language that prepares input for GRAP, although it was perhaps the easiest to implement. Another language, SCATMAT, is used for describing scatter-plot matrices [8].

Given a set of n observations of k attributes, a scatter-plot matrix is a $k \times k$ array of scatter plots. For example, this file contains data on the nine planets: distance from the sun, temperature, mass and radius:

```
0.4   600   .05   .4
0.75  370   .8    1
1     330   1     1
1.5   300   .11   .5
5     140   318   11
...
40    50    1     .5
```

(Temperature is degrees Kelvin; other attributes are normalized to earth's value.) This scatter-plot matrix shows the six pairwise relationships among the four variables:



Although it would be possible to specify a scatter-plot matrix "by hand" using GRAP's facility for defining subgraphs, it would require an inordinate amount of work, much more than for a dotchart. Thus we designed another language, again to be processed into GRAP by a small compiler, also written in AWK.

The input language for SCATMAT is similar to the DOTCHART language:

```
file "planets.d"
frames ht .75 wid .75
spread 0
alllog
name Distance
     field $1
name Temperature
     field $2
name Mass
     field $3
name Radius
     field $4
```

The first version of SCATMAT was about 35 lines of AWK; the current version is about 100 lines. It provides many more parameters, and can also deal with variations like using the other diagonal and printing only one triangle of the matrix.

It is useful to build languages for other particular graphs as well; we have done so for box plots, and have seen one for pie charts. Such languages are easy to build and can be easy to use because the common output language and (usually) common implementation encourage a similar style.

5. Conclusions

Our original goal was a language for preparing *publication-quality graphs. That goal has been* achieved: with the addition of GRAP to the Unix document preparation tools, we are now able to produce complicated graphical displays with little effort. The quality is acceptable for books. Examples may be found in Cleveland [6], Aho, Sethi and Ullman [9] and Bentley [10].

GRAP has also proven useful for exploratory data analysis, even though that was not our intent. This is certainly not because it runs fast (for most graphs it is much slower than, for example, the S system [11]), but apparently because its textually based interface fits well with other Unix tools. It is easy to prepare data with some program, massage it into the right format (either with a general tool like AWK or with the input processing of GRAP itself), then plot it to see what things look like.

GRAP has also turned out to be surprisingly useful for prototyping statistical displays. It has built-in facilities for both display and computation, and provides an easy escape to the Unix environment when the built-in mechanisms are not adequate.

As a more general observation, many tasks can be profitably approached by designing and implementing a "little language" specialized to that task. Users can thereby express their solutions in terms closely related to their view of the problem. Specialized languages for graphs, dotcharts, and scatter-plot matrices are merely examples from one domain. Indeed, the entire family of Unix document preparation programs consists of such little languages, some feeding TROFF directly, while others compile into intermediate languages.

In most of these languages, it appears necessary to provide some degree of programmability; otherwise, users are restricted to those things that the implementor thought of. For GRAP especially, the ability to program the processor to define a new style has proven invaluable.

Acknowledgements

We are grateful for the helpful comments of Rick Becker, Bill Cleveland, Doug McIlroy, Rob Pike, and Chris Van Wyk.

References

1. B. W. Kernighan and M. E. Lesk, "UNIX Document Preparation," pp. 1-20 in *Document Preparation Systems*, ed. J. Nievergelt, G. Coray, J.-D. Nicoud, A. C. Shaw, North-Holland Publishing Co. (1982), pp. 1-20.

2. R. Furuta, J. Scofield, and A. Shaw, "Document Formatting Systems: Survey, Concepts and Issues," *Computing Surveys* **14**(3), pp. 417-72 (September 1982).

3. J. L. Bentley and B. W. Kernighan, "GRAP — A Language for Typesetting Graphs," *CACM* **29**(7) (July 1986).

4. J. L. Bentley and B. W. Kernighan, *GRAP — A Language for Typesetting Graphs. Tutorial and User Manual*, AT&T Bell Laboratories CSTR 114 (December, 1984).

5. E. Tufte, *Visual Display of Quantitative Information*, Graphics Press (1982).

6. W. S. Cleveland, *The Elements of Graphing Data*, Wadsworth (1985).

7. A. V. Aho, B. W. Kernighan, and P. J. Weinberger, "AWK—A Pattern Scanning and Processing Language," *Software Practice and Experience* **9**, pp. 267-280 (April 1979).

8. J. M. Chambers, W. S. Cleveland, B. Kleiner, and P. A. Tukey, *Graphical Methods for Data Analysis*, Wadsworth (1983).

9. A. V. Aho, R. Sethi, and J. D. Ullman, *Compilers: Principles, Techniques and Tools*, Addison-Wesley (1985).

10. J. L. Bentley, *Programming Pearls*, Addison-Wesley (1986).

11. R. A. Becker and J. S. Chambers, *S: An Interactive Environment for Data Analysis and Graphics*, Wadsworth (1984).

REMOTE SENSING AND IMAGE PROCESSING

Organizer: *Paul Switzer, Stanford University*

Binary Image Classification

Carl N. Morris, University of Texas at Austin

Partial and Interaction Spline Models for the Semiparametric Estimation of Functions of Several Variables

Grace Wahba, University of Wisconsin, Madison

BINARY IMAGE CLASSIFICATION

Carl N. Morris, The University of Texas at Austin

1. An Empirical Bayes Model for Image Data.

Motivated by the LANDSAT problem of estimating the probability of crop or geological types based on multi-channel satellite imagery data, Morris and Kostal (1983), Hill, Hinkley, Kostal, and Morris (1984), and Morris, Hinkley, and Johnston (1985), henceforth labeled MK83, HHKM84 and MHJ85, developed an empirical Bayes approach to this problem. We return here to those developments, making certain improvements and extensions, but restricting attention to the binary case of only two attributes.

Label the pixels in a rectangular lattice as $i = (j, k)$, $j = 1, 2, \ldots, J$ and $k = 1, 2, \ldots, K$. Each of these $n = JK$ pixels has attribute θ_i taking values $\theta_i = 0$ or $\theta_i = 1$ to indicate *only two possible distinct types*. A one-time vector D_i of measurements is available for each pixel, usually involving several bandwidths for several time points. In this simple version a one-dimensional function y_i of D_i is all that will be considered (as noted in MHR85, if D_i is multidimensional, then y_i is the best one-dimensional summary if it is chosen as the logarithm of the likelihood ratio of D_i for $\theta_i = 1$ and $\theta_i = 0$).

An empirical Bayes model is defined as one that *provides two families of distributions*, one for the data, conditional on the parameters, and one for the parameters. The *descriptive* empirical Bayes model specifies distributions

(1) $p(y \mid \theta)$ for the data $\{y_i\}$, conditional on the unknown parameters $\{\theta_i\}$, *i.e.*, the likelihood functions, and

(2) a parametric family $p_\alpha(\theta)$ of distributions for the parameters, indexed by hyperparameters $\alpha \in A$.

The *inferential* empirical Bayes model is mathematically equivalent to the descriptive model, but respecifies the distributions as

(1^*) $p_\alpha(y)$, *the marginal distribution for the data, now* dependent on $\alpha \in A$, and

(2^*) $p_\alpha(\theta \mid y)$, the conditional distribution for the parameters $\{\theta_i\}$ given the data $\{y_i\}$ and α.

The distributional choices here are the same as for MHR 85. The key simplifying assumption is that the correlations between observed measurements enters entirely through the parameters, $\{\theta_i\}$ the observed data $\{y_i\}$ being conditionally independent, given $\{\theta_i\}$

DESCRIPTIVE MODEL:

(1) Given θ_i, $i = 1, \ldots, n$, $\theta_i = 0$ or 1, assume $y_i \mid \theta_i \sim N(\delta(\theta_i - .5), 1)$ independently, with δ a known constant.

(2) The $\{\theta_i\}$, in the wide sense, have a stationary, isotropic distribution on the lattice with $\pi = P(\theta_i = 1)$ all $i = (j, k)$, and for $-s \le t, u \le s$, $\rho_{t,u} = \mathrm{Corr}(\theta_{j,k}, \theta_{j+t,k+u})$, all j, k.

Some comments on this model are required.

The distribution (1), which says y_i has mean $\pm\delta/2$ and unit variance is equivalent via location and scale changes to any model for y_i giving y_i a normal distribution with means μ_0 or μ_1 when $\theta_i = 0$ or $\theta_i = 1$ and variances $\sigma_1^2 = \sigma_2^2 = \sigma^2$. Then $\delta = (\mu_1 - \mu_0)/\sigma$. The parameters (μ_0, μ_1, σ) and the form (normal) of the density of y_i are assumed known. In practice, these would be known based on vast experience with "training data" (where the values of θ_i could be observed along with y_i).

The stationary assumption (2) for the parameters justifies letting the parameters π and $\rho_{t,u}$ be independent of $i = (j, k)$. This assumption needs to hold only in the wide-sense because the inferential methods used do not involve more than these first and second moments. The isotropic assumption would serve further to simplify $\rho_{t,u}$, and we do identify $\rho_{t,u} = \rho_{u,t} = \rho_{|t|,|u|}$, but do not take full advantage of the rotational invariance. The hyperparameters α then are $r + 1 = (s+1)(s+2)/2$ dimensional. For example, if $s = 2$, then $r = 5$ and

$$\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_6) = (\pi, \rho_{10}, \rho_{11}, \rho_{20}, \rho_{21}, \rho_{22}) \, .$$

It is important to realize that the hyperparameters must be estimated from the observed data $\{y_i\}$ in the target site. Training data taken from other settings can be used to determine the conditional distribution of $\{y_i\}$ for known $\{\theta_i\}$, *i.e.*, the distribution (1), but different hyperparameters, say $\tilde{\alpha}$, would prevail at the training site, so training data could be used to estimate α only if the unexpected assumption $\tilde{\alpha} = \alpha$ held.

2. Results for the Inferential Model: The Discriminant Function Approximation and Identity.

Development of the inferential model proceeds as in MHR85, but with a more useful representation of the discriminant function. The model (1),(2) leads to a very complicated exact form for each marginal posterior probability, given α, $P(\theta_i = 1 \mid \underset{\sim}{y}, \alpha)$. However, a good approximation to this probability, with accuracy improving as $\delta \to 0$, is of logistic form. We go further to approximate the logistic function by the discriminant function, which effectively predicts $P(\theta_i = 1 \mid \text{data})$ from the "ring" averages, these being averages of those data values in specified locations (rings) relative to pixel i, as in Figure 1.

Figure 1: Ring locations, $R_0, R_1, \ldots, R_r$, centered with pixel i at ring R_0. The four nearest points, marked 1, are R_1, the next four R_2, and so on. Pairs (t,u) indicate similar correlation structure, with common values $\rho_{t,u}$ if $(t,u) = (\pm t, \pm u) = (\pm u, \pm t)$.

$$\begin{array}{ccccc}
\ddots & & \overset{3}{(0,2)} & & \cdot^{\cdot^{\cdot}} \\
 & \overset{2}{(-1,1)} & \overset{1}{(0,1)} & \overset{2}{(1,1)} & \\
\overset{3}{(-2,0)} & \overset{1}{(0,1)} & \overset{0}{(0,0)} & \overset{1}{(1,0)} & \overset{3}{(2,0)} \\
 & \overset{2}{(-1,-1)} & \overset{1}{(0,-1)} & \overset{2}{(1,-1)} & \\
\cdot^{\cdot^{\cdot}} & & \overset{3}{(0,-2)} & & \ddots
\end{array}$$

Define the $n \times (r+1)$ matrix X of "regressors" to have the i^{th} row element and t^{th} column element as the average of measurements at ring t, R_t, $t = 0, 1, \ldots$. Thus, with $i = (j,k)$, $x_{i0} = y_i$, is the regressor for ring 0,

$$(2.1) \qquad x_{i1} \equiv (y_{k,\ell+1} + y_{k-1,\ell} + y_{k,\ell-1}, y_{k+1,\ell})/4 \ ,$$

is the regressor for R_1, and so on, as in Fig.1. Then modify X so that all column totals add to zero, by subtracting the column average from each column.

If all the parameters $\{\theta_i\}$ were known, we would calculate the discriminant function as follows. Let $C(\theta) = X'\theta/n$, $\bar{\theta} = \sum \theta_i/n$. Denote

$$(2.2) \qquad R_\theta^2 = nC'(\theta)(X'X)^{-1}C(\theta)/\bar{\theta}(1-\bar{\theta})$$

as the multiple correlation coefficient between θ and the columns of X, and

$$(2.3) \qquad \mathrm{RSS}(\theta) = \bar{\theta}(1-\bar{\theta})(1-R_\theta^2)$$

as the residual sum of squares.

The discriminant analysis formula has the form

$$\begin{aligned}
\lambda_i &= \log\left(\frac{P(\theta_i = 1 \mid y)}{P(\theta_i = 0 \mid y)}\right) \\
(2.4) \qquad &= \log(\bar{\theta}/(1-\bar{\theta})) + \frac{n}{\mathrm{RSS}(\theta)}\, x_i'(X'X)^{-1}C(\theta) \\
&\quad - (\bar{\theta} - .5)\left(\frac{R_\theta^2}{1-R_\theta^2}\right) .
\end{aligned}$$

This form is equivalent to that of MHJ85, but is in more useful form because the quantities involved are directly related to standard linear regression output.

The sample autocovariances c_t, $t = 1, 2, \ldots, r$, being

$$c_t = \frac{1}{n} \sum y_i x_{it}$$

and the mean $\bar{y}$ provide estimates of the hyperparameters. Let $\hat{\pi} = .5 + \bar{y}/\delta$. Then

$$(2.5) \qquad E_\theta \bar{y} = \delta(\bar{\theta} - .5) \ , \quad E_\theta \hat{\pi} = \bar{\theta} \quad \text{and}$$

$$(2.6) \qquad E\left(\delta\hat{\pi}(1-\hat{\pi}) \ , \ \tfrac{1}{\delta}c_1, \ldots, \tfrac{1}{\delta}c_r\right)' \doteq EC(\theta) \ .$$

Thus $\bar{\theta}$ is estimated by $\hat{\pi}$ and $C(\theta)$ by

$$\hat{C} = \left(\delta\hat{\pi}(1-\hat{\pi}) \ , \ c_1/\delta, \ldots, c_r/\delta\right)' \ .$$

Therefore (2.4) may be estimated, even when θ is unknown.

We now can specify the

INFERENTIAL MODEL:

(1′) The marginal distribution of the data y satisfies (2.5)-(2.6); and,

(2′) the posterior probability $P(\theta_i = 1 \mid y, \alpha)$ follows approximately the form (2.4).

3. Empirical Bayes Estimation: Estimating Discriminant Function Parameters from Remotely Sensed Data.

Empirical Bayes modeling stops short of specifying a unique method for approximating the function (2′), but (2.5) and (2.6) provide obvious approaches. The simplest estimate of $\bar{\theta}$ is $\hat{\pi}$ defined by (2.5), since $\hat{\pi}$ is unbiased in the empirical Bayes sense, $E(\hat{\pi} - \bar{\theta}) = 0$ for all joint distributions considered in (1)-(2), that is, for all α. The approximately unbiased estimate $\hat{C}$ in (2.6) of $C(\theta)$ was used in MHR85 to replace $C(\theta)$ in (2.4), and in the R_θ^2 and RSS(θ) formulae (2.2),(2.3). This works well when $\hat{C}$ has small variance. The estimator of MHR85 results:

$$(3.1) \qquad \begin{aligned}
\hat{\lambda}_i = \log(\hat{\pi}/(1-\hat{\pi})) &+ \frac{n}{\mathrm{RSS}_0}\, x_i'(X'X)^{-1}\hat{C} \\
&- (\hat{\pi} - .5)\frac{\hat{R}^2}{1-\hat{R}^2}
\end{aligned}$$

with

$$(3.2) \qquad \begin{aligned}
\mathrm{RSS}_0 &\equiv \hat{\pi}(1-\hat{\pi})(1-\hat{R}^2) \ , \\
\hat{R}^2 &\equiv n\hat{C}'(X'X)^{-1}\hat{C}/\hat{\pi}(1-\hat{\pi}) \ .
\end{aligned}$$

The estimator (3.1) was shown to fare well in a variety of settings, compared with the "ideal" estimator λ_i (which is not available in practical problems).

Still, improvements to (3.1) may be necessary in certain cases: because λ_i depends on the data being used to estimate $C(\theta)$, so that one is not guaranteed that

$$(3.3) \qquad Ex_i'(X'X)^{-1}\hat{C} \doteq x_i'(X'X)^{-1}C(\theta) \ .$$

for instance; and because λ_i is non-linearly dependent on $C(\theta)$, so that variability of any nearly unbiased estimator $\hat{C}$ will cause bias in non-linear estimation of $C(\theta)$.

Recognizing that $\hat{p}_i = \exp(\hat{\lambda}_i)/(1+\exp(\hat{\lambda}_i))$ estimates the posterior probability $p_i = E(\theta_i \mid \text{data})$, since such calculations were used to justify the logistic form of p_i, the data-dependency objection can be handled by replacing θ_i by $\hat{p}_i$ in $C(\theta)$, still using $\hat{\pi}$ for $\bar{\theta}$ (experience shows that $\hat{\pi}$ is very close to $\sum \hat{p}_i/n$). This suggests an iterative rule:

(a) Calculate $\hat{p}_i$, $1 \le i \le k$ using (3.1);

(b) Replace $\bar{\theta}$ with $\hat{\pi}$ and $C(\theta)$ with $(\hat{p}_1, \ldots, \hat{p}_n)'$ in (2.4) to get $\hat{\hat{\lambda}}_i$;

(c) Compute $\hat{\hat{p}}_i = (1+\exp(-\hat{\hat{\lambda}}_i))^{-1}$; and,

(d) Return to (b), using $(\hat{\hat{p}}_i, \ldots, \hat{\hat{p}}_n)$.

Initial experience with this rule has resulted in more stable relative values of the regression coefficients $\beta \equiv (X'X)^{-1}X'\theta$, but their absolute values are too large after several iterations. This could be due to over estimation of the convex function $1/\text{RSS}(\theta)$.

The problem of quadratic dependence of R^2_θ on θ in (2.2) can be easily handled by methods conditional on the data, since, with expectation conditional on X, and $p \equiv E\theta \mid X$,

(3.4)

$$\begin{aligned} E\theta'(X'X)^{-1}\theta &= \text{tr}\left((X'X)^{-1}E\theta\theta'\right) \\ &= p'(X'X)^{-1}p + \text{tr}\left((X'X)^{-1}\Sigma_X\right) . \end{aligned}$$

To implement this, however, the currently unavailable posterior probabilities of pairwise ocurrences of θ_i and θ_j also are needed to compute Σ_X, the conditional covariance matrix of the θ vector.

4. Other Uses of the Spatial Logistic Estimator: Detection of Edges, Corners, and Shapes.

The technology of Section 2 also can be used for the purposes of determining the probabilities that edges of shapes, or even part of a particular shape, exist at a location. For edge and corner detection, it is convenient to shift the entire rectangular lattice up and sideways one-half pixel, so that points for this form of detection are relocated at the original pixel boundaries and corners rather than at pixel centers. Then the matrix X of (2.4) is specified not as in (2.1) and the surrounding discussion, but with other codings sensitive to boundaries. For example, a "signed horizontal edge detector", as in Fig.2, when placed at location i, adds the 12 y-values above and subtracts the 12 y-values below, the sum producing the value x_i at location $i = (j,k)$. Note that x_i has an expected value of 12δ at locations for which all 12 pixels above are of type 1 and all 12 pixels below are type 0, still assuming $y_i \mid \theta_i \sim N(\delta(\theta_i - .5), 1)$. Of course, -12δ is obtained if all 24 pixels are reversed, and values between -12δ and 12δ result in more scattered situations. $Ex_i = 0$ in the middle of a large homogenous shape. Large values of $|x_i|$ indicate the presence of a horizontal edge, but without suggesting whether $\theta_i = 1$ to the north or the south.

Figure 2: Assignments of numerical weights for a horizontal 4 x 6 edge detector placed at the center of 24 pixels.

1	1	1	1	1	1
1	1	1	1	1	1
-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1

Similarly, vertical edge detectors would correspond to turning Fig.2 on its side, and two possible corner detectors might be as in Fig.3 (these types would work best in a checkerboard setting). The first detector would be particularly sensitive to the meeting of two corners, the second one to the southwestern corner of a figure extending into a homogeneous area.

Figure 3: Corner detectors for a rectangular grid.

-1	-1	1	1
-1	-1	1	1
1	1	-1	-1
1	1	-1	-1

-1	1	.5	0
-1	1	1	.5
-1	1	1	1
-1	-1	-1	-1

Variables x'_i and x''_i might be coded as indicated by Fig.3 for each position $i = (j,k)$, with the data values $\{y_i\}$ assigned weights according to the values in the detectors (and zero outside the detector). Thus x'_i would be the sum of values of the eight nearest pixels to the northeast and southwest minus the sum of the other eight nearest values.

Detectors for other, more general shapes, could be set up in an analogous fashion, assigning positive values to data in locations where the shape would exist, and negative values elsewhere.

The true values θ_i again must be defined as binary values, if the methods of Section 2 are to apply. For example, we might have $\theta_i = 1$ if a horizontal or vertical boundary exists at location i, otherwise $\theta_i = 0$. Then the X matrix might have two variables, row i being $(1, |x_i|, |\tilde{x}_i|, \ldots)$, x_i from the edge detector of Fig.2 and $\tilde{x}_i$ as a signed vertical edge detector. Formula (2.4) again is available, providing an estimate of $P(\theta_i = 1 \mid \text{data})$ via the discriminant function. However, estimates of $\bar{\theta}$ and $C(\theta)$ are required, and will take a different form than given in Section 3. If such values are available, these methods can be used in conjunction with estimates of the probabilities of each classification as given in Section 3 to give more accurate estimates at the borders of those regions having significant sizes.

5. Conclusions.

Empirical Bayes methods, as reported in MK83, HHKM84 and MHR85, provide a helpful perspective from which one can view the remote sensing, image restoration, and other spatial problems. Lessons include:

(i) Spatial correlation can be modeled as occurring entirely in the ground truth process $\{\theta_i\}$, or jointly in ground truth and in the observations $\{y_i\}$. The conditional covariances of (y_i, y_j) given θ would be derived from training data.

(ii) Training data are inadequate for estimating the hyperparameters α. Their proper use is to determine the likelihood function (*e.g.*, "Badhwar numbers", "greenness", and "brightness" formulas derived for use in LANDSAT satellite data applications).

(iii) Remotely sensed data may be used to estimate the proper Bayes rules and thus are used for dual purposes of hyperparameters $\{\alpha\}$ estimation *and* ground truth $\{\theta_i\}$ estimation.

The celebrated work of D. Geman and S. Geman (1984) falls within the empirical Bayes paradigm in that the hyperparameters they specify are estimated from the marginal distribution of the data, rather than being arbitrarily chosen. While their results are more general than those here, we hope eventually to extend further the approach in this paper to include the polytomous case (several possible values) for θ_i, non-normal multivariate distributions for y_i (the likelihood ratio then plays a central role), and dependencies in the distribution of the observations, given the ground truth. This paper indicates some extensions of MHJ85 by suggesting improvements in estimation of hyperparameters, and in expanding the role of the technique to include detection of edges and shapes. Continuing with this approach not only will provide further insights, but will provide computationally quicker methods than the computer intensive techniques necessary to estimate posterior modes. This will allow more data and larger regions to be analyzed.

Acknowledgements

This investigation was supported jointly by the National Aeronautics and Space Administration for a project "Mathematical Pattern Recognition and Image Analysis" through Subcontract Agreement No. L20076 to the Texas A&M Research Foundation, College Station, Texas 77843 and by NSF Grant No. DMS 8407876, Department of Mathematics and Center for Statistical Sciences, The University of Texas, Austin, Texas 78712.

References

1. Geman, D. and Geman, S., (1984). "Stochastic relaxation, Gibbs distribution, and Bayesian restoration of images", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-6, No.6, 721–741 (November).

2. Hill, J., Hinkley, D., Kostal, H. and Morris, C.N., (1984). "Spatial Estimation from Remotely Sensed Data Via Empirical Bayes Models", *Proceedings of the NASA Symposium on Mathematical Pattern Recognition and Image Analysis* (L.F. Guseman, Jr., ed.), Houston, Texas, June 1984, 115–136.

3. Morris, C.N., Hinkley, D.V. and Johnston, W.,(1985). "Classification in a Spatially Correlated Environment", *Proceedings of the Third Annual Symposium on Mathematical Pattern Recognition and Image Analysis*, Texas A&M University, Dept. of Mathematics, College Station, Texas 77843, (L. Guseman, ed.); also, CSS Technical Report No. 20, Center for Statistical Sciences, The University of Texas, Austin, Texas *78712 (August 1985)*.

4. Morris, C.N. and Kostal, H., (1983). "An Empirical Bayes Approach to Spatial Analysis", *Proceedings of the NASA Symposium on Mathematical Pattern Recognition and Image Analysis* (L.F. Guseman, Jr., ed.), Houston, Texas, June 1-13, 1983, 143–165.

Partial and Interaction Spline Models for the Semiparametric Estimation of Functions of Several Variables.

Grace Wahba, University of Wisconsin–Madison

A partial spline model is a model for a response as a function of several variables, which is the sum of a "smooth" function of sevaral variables and a parametric function of the same plus possibly some other variables. Partial spline models in one and several variables, with direct and indirect data, with Gaussian errors and as an extension of GLIM to partially penalized GLIM models are described. Application to the modelling of change of regime in several variables is described. Interaction splines are introduced and described and their potential use for modelling nonlinear interactions between variables by semiparametric methods is noted. Reference is made to recent work in efficient computational methods.

1. Introduction

Partial spline models have proved to be interesting both from a practical and a theoretical point of view, partly because of their dual nature both as solutions to certain intuitively reasonable variational problems, and as Bayes estimates with certain parsimonious priors. In these proceedings we will attempt to give a quick rundown concerning some of their more interesting manifestations, and to report briefly on two new developments, first, the use of partial spline models to describe discontinuities or changes of regime, in two, three and higher dimensions, and, second, the idea of interaction splines for use in studying nonlinear interactions between variables semiparametrically.

2. Partial spline models - one splined variable

A response as a function of the variables $x, z_1, \ldots, z_k$ is modelled as

$$y_i = f(x(i)) + \sum_{j=1}^{p} \theta_j \Psi_j(x(i); z(i)) + \varepsilon_i \qquad (2.1a)$$

where

$$z(i) = (z_1(i), \ldots, z_k(i)) \qquad (2.1b)$$

the Ψ_j's are given parametric functions and the ε_i's are independent, zero mean Gaussian random variables with common (unknown) variance. The estimate $(f_\lambda, \theta_\lambda)$, where $\theta_\lambda = (\theta_{1\lambda}, \ldots, \theta_{p\lambda})$, is found as the minimizer, in an appropriate space, of

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - f(x(i)) - \sum_{j=1}^{p}\theta_j \Psi_j(x(i); z(i)))^2 +$$

$$\lambda J_m(f) \qquad (2.2a)$$

where

$$J_m(f) = \int_0^1 (f^{(m)}(x))^2 dx. \qquad (2.2b)$$

We have the following

Theorem: (Kimeldorf and Wahba (1971) - KW) Let $\Phi_1, \ldots, \Phi_m$ span the null space of J_m. If the design matrix for least squares regression on span $\Phi_1, \ldots, \Phi_m; \Psi_1, \ldots, \Psi_p$ is of full column rank, then there exists a unique minimizer $(f_\lambda, \theta_\lambda)$ for any $\lambda > 0$, and f_λ is a polynomial spline function.

The parameter λ as well as m can be choosen by generalized cross validation (GCV).

The appropriate function space here is the Sobolev space W_2^m, however, J_m (and W_2^m) can be replaced by any seminorm in a reproducing kernel (r. k.) Hilbert space of real valued functions on [0,1] provided that least squares regression onto the span of the null space of the seminorm is well defined - you get a Bayes estimate with the r. k. related to the prior covariance. Details may be found in KW and Wahba (1978) but we will not discuss the Bayesian aspect any further, other than to note that the prior behind J_m is the most parsimonious member of a large class of equivalent priors.

Partial spline models with one splined variable were introduced by several authors in different contexts, with some interesting applications, see Anderson and Senthilselvan (1982), Engle et al. (1983), Green, Jennison, and Seheult (1983), Shiller (1984).

3. Partial Spline Models - Several Splined Variables

Now, let the model be

$$y_i = f(x(i)) + \sum_{j=1}^{p} \theta_j \Psi_j(x(i); z(i)) + \varepsilon_i \qquad (3.1a)$$

where

$$x = (x_1, \ldots, x_d), \, x(i) = (x_1(i), \ldots, x_d(i)). \qquad (3.1b)$$

Again, we find f in an appropriate space to minimize

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - f(x(i)) - \sum_{j=1}^{p}\theta_j \Psi_j(x(i); z(i)))^2 +$$

$$\lambda J_m(f) \qquad (3.2)$$

where now, we can use the "thin plate spline" penalty functional. For $d=2$, $m=2$, it is

$$J_m(f) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f_{x_1x_1}^2 + 2f_{x_1x_2}^2 + f_{x_2x_2}^2, \qquad (3.3)$$

and for arbitrary d it is

$$J_m(f) = \sum_{\alpha_1+\ldots+\alpha_d=m} \frac{m!}{\alpha_1! \cdots \alpha_d!} \times$$

$$\int \cdots \int \left(\frac{\partial^m f}{\partial x_1^{\alpha_1} \cdots \partial x_d^{\alpha_d}} \right]^2 dx_1 \cdots dx_d. \qquad (3.4)$$

provided $2m>d$. The null space of J_m is the span of the $M = \begin{bmatrix} m+d-1 \\ d \end{bmatrix}$ monomials of total degree less than m, call them $\Phi_1, \ldots, \Phi_M$. Again, there will be a unique minimizer $(f_\lambda, \theta_\lambda)$ for every nonnegative λ if the design matrix for least squares regression on $\Phi_1, \ldots, \Phi_M; \Psi_1, \ldots, \Psi_p$ is of full column rank, and f_λ is a thin plate spline function.

Partial splines with several splined variables were introduced in Wahba (1984a), Wahba (1984b), Wahba (1985), and a discrete version has been proposed by Green, Jennison, and Seheult (1986). Transportable code (GCVPACK, Bates et al. (November 1985)) is available for fitting the partial spline models of (3.1)-(3.4) and computing the GCV estimate $\hat{\lambda}$ of λ. This code does well with up to around 400 data points on the VAX 11/750 in the Statistics Department at Madison. The work primarily depends on n, and not d, but, of course good estimates with large d will require large n. Diagnostics for splines (without the "partial" part) have been developed by Eubank (1986), it can be anticipated that this work will extend to partial spline models.

4. Indirect measurements

Let

$$g(x;z) = f(x) + \sum\theta_j\Psi_j(x;z), \qquad (4.1)$$

and now let

$$y_i = L_i g + \varepsilon_i \qquad (4.2)$$

where L_i is a bounded linear functional, for example:

$$L_i f = \int w_i(x;z) g(x;z) \pi dx \pi dz. \qquad (4.3)$$

This kind of data comes up in X-ray tomography, satellite tomography, stereology, and in other remote or indirect sensing problems in the physical and biological sciences. One finds f and θ to minimize:

$$\frac{1}{n}\sum(y_i - L_i f - \sum\theta_j L_i \Psi_j)^2 + \lambda J_m(f). \qquad (4.4)$$

The use of variants of (4.3), and (4.4) may also provide a good way to deal with heterogeneous aggregated economic data. For an application in stereology, see Nychka et al. (1984).

Data involving mildly nonlinear functionals can be accomodated - then

$$y_i = N_i g + \varepsilon_i \qquad (4.5a)$$

where

$$N_i g = \int\int w_i(x, z, g(x;z)) \pi dx \pi dz. \qquad (4.5b)$$

One finds f and θ to minimize

$$\frac{1}{n}\sum(y_i - N_i(f + \sum\theta_j\Psi_j))^2 + \lambda J_m(f). \qquad (4.6)$$

The minimization can be performed using basis functions and a Gauss-Newton iteration and λ chosen by GCV for nonlinear problems, see O'Sullivan and Wahba (1985).

5. Non Gaussian errors (semiparametric penalized GLIM models)

Here

$$g(x,z) = f(x) + \sum\theta_j\Psi_j(x;z) \qquad (5.1)$$

but

$$y_i \,\tilde{}\, F_g.$$

For example :

$$y_i \,\tilde{}\, Poisson\ with\ \Lambda_i = e^{g(x(i);z(i))},$$

$$y_i \,\tilde{}\, Binomial\ with\ p_i/(1-p_i) = e^{g(x(i));z(i))},$$

etc. Here, one finds $f_\lambda, \theta_\lambda$ to minimize

$$L(f,\theta) + \lambda J_m(f) \qquad (5.2)$$

where L is the log likelihood. O'Sullivan (1983) and O'Sullivan, Yandell, and Raynor (1986) proposed numerical methods and a GCV for penalized GLIM models. See also Green and Yandell (1985), Silverman (1982), Cox and O'Sullivan (October, 1985), Leonard (1982). Further work on numerical methods for penalized GLIM and nonlinear indirect sensing problems is reported in this proceedings by Yandell.

6. Use of partial splines to model functions which are smooth except for specified discontinuities

Let $d=1$ and let

$$g(x;z) = f(x) + \theta|x - x^*|$$

that is, $\Psi_1(x;z) = |x - x^*|$. Then the partial spline estimate of g will have a jump in the first derivative at x^* of size 2θ. In two dimensions we may use a partial spline to model a jump in the first derivative with respect to x_2 along a given curve $x_{2*}(x_1)$: Let

$$\gamma(x) = \gamma(x_1, x_2) = |x_2 - x_2^*(x_1)|,$$

$$g(x;z) = f(x) + \theta(x_1)\gamma(x)$$

where θ may depend on x_1. Then

$$\left.\frac{\partial g}{\partial x_2}\right]_{x_2 = x_2(x_1)^*_-} - \left.\frac{\partial g}{\partial x_2}\right]_{x_2 = x_2(x_1)^*_+} = 2\theta(x_1).$$

If, for example

$$\theta(x_1) = \sum_{j=1}^{p}\theta_j q_j(x_1)$$

where the q_j's are given, then

$$\Psi_j(x;z) = q_j(x_1)\gamma(x).$$

This fits right into the partial spline setup, and GCVPACK may be used to compute the estimate. A generalization to $d=3$ with a jump in the first derivative with respect to x_3

along a surface $x_3^*(x_1,x_2)$ is straigtforward. For details, and a description of an application to the three dimensional modelling of the tropopause in the atmosphere and the thermocline in the ocean, see Shiau, Wahba, and Johnson (Dec. 1985).

7. Linear inequality constraints

Expressions (2.2), (3.2), (4.4), etc. can be minimized subject to finite families of linear inequality constraints. See Villalobos and Wahba (March 1985).

8. Main effects and interaction splines

The thin plate spline is defined on Euclidean d space for any d with $2m-d>0$, provided there are enough data points for mth degree polynomial regression, but unless there are very large data sets, in many applications will be desireable to reduce the amount of structure involved. Several authors have suggested modelling f as a linear combination of functions of one variable, that is,

$$f(x)=f_0+\sum_{al=1}^{d} f_\alpha(x_\alpha),$$

where $x=(x_1,\dots,x_d)$, and $\int_0^1 f_\alpha(x_\alpha)dx_\alpha=0$. (Note the switch to the unit cube.) See Friedman, Grosse, and Stuetzle (1983), Stone (1985), Burman (June, 1985). We have been working on generalizations of this idea, whereby f is modelled successively as linear combinations of functions of one variable, functions of one and two variables, functions of one, two and three variables, etc. The resulting estimates may be called main effects splines, first order interaction splines, second order interaction splines, etc., by analogy with analysis of variance. We consider here two quite different but interesting penalty functionals which we will refer to as TEPR (for "tensor product"), and THPL (for "thin plate"). We will briefly sketch some early results of some work in progress, by describing the simplest examples.

The main ideas are most easily explained by first considering only spaces of periodic functions on the unit d-dimensional hypercube, that satisfy certain linear equality or boundary condtions, and then removing these conditions. Let $\phi_\nu(x_j)=\cos 2\pi\nu x_j$ or $\sin 2\pi\nu x_j$ (with some abuse of notation), and let $\theta_0=1,\theta_\nu=2\pi\nu,\nu>0$, and let H_{TEPR}^{per} and H_{THPL}^{per} be, respectively, the collections of all functions f of the form

$$f(x_1,\dots,x_d)=$$

$$\sum_{\nu_1,\dots,\nu_d=0}^{\infty} c_{\nu_1\cdots\nu_d}\phi_{\nu_1}(x_1)\cdots\phi_{\nu_d}(x_d) \qquad (8.1)$$

with

$$\sum_{\nu_1,\dots,\nu_d=0}^{\infty} [\theta_{\nu_1}\cdots\theta_{\nu_d}]^{2m} c_{\nu_1\cdots\nu_d}^2<\infty,\ \ H_{TEPR}^{per} \qquad (8.2)$$

or

$$\sum_{\nu_1,\dots,\nu_d=0}^{\infty} [\theta_{\nu_1}{}^2+\cdots+\theta_{\nu_d}{}^2]^m c_{\nu_1\cdots\nu_d}^2<\infty\ \ H_{THPL}^{per}(8.3)$$

It can be shown that H_{TEPR}^{per} will be a reproducing kernel hilbert space with (8.2) as squared norm for any $m>1/2$, and H_{THPL}^{per} will be a reproducing kernel space with the squared norm (8.3) for any $m>d/2$. These spaces are not equivalent, and reflect different ideas of what is "smooth". However, each can be written as the direct sum of 2^d orthogonal subspaces, namely, H_o, the $\begin{bmatrix} d\\ 1\end{bmatrix}$ "main effects" subspaces of the form

$$H_\alpha=span\ \{\phi_{\nu_\alpha}(x_\alpha),\ \nu_\alpha=1,2,\dots\}\ \alpha=1,\dots,d,$$

the $\begin{bmatrix} d\\ 2\end{bmatrix}$ first order interaction spaces of the form

$$H_{\alpha\beta}=span\ \{\phi_{\nu_\alpha}(x_\alpha)\phi_{\nu_\beta}(x_\beta),\ \nu_\alpha,\ \nu_\beta>0\},\ 1\le\alpha<\beta\le d,$$

and so on.

Letting

$$J_o(f)=[\int_0^1\cdots\int_0^1 f(x_1,\dots,x_d)\prod_\alpha dx_\alpha]^2, \qquad (8.4)$$

the squared norm (8.2) on H_{TEPR}^{per} can be shown to be equal (in H_{TEPR}^{per}) to

$$J_o(f)+J^{THPL}(f), \qquad (8.5)$$

where

$$J^{THPL}(f)=\sum_{\alpha_1+\dots+\alpha_d}\frac{m!}{\alpha_1!\cdots\alpha_d!}\times$$

$$\int_0^1\cdots\int_0^1\left[\frac{\partial^m f}{\partial x_1^{\alpha_1}\cdots\partial x_d^{\alpha_d}}\right]^2 dx_1\cdots dx_d \qquad (8.6)$$

is the thin plate penalty functional.

For lack of space we will not discuss the thin plate spaces further, but analyses similar to but slightly more complicated than those below can be carried out. In what follows, we will only consider the tensor product case and sub or superscripts TEPR are to be understood.

Let

$$J_\alpha(f)=\int_0^1 dx_\alpha[\int_0^1\cdots\int_0^1\frac{\partial^m f}{\partial x_\alpha^m}\prod_{\beta\ne\alpha}dx_\beta]^2 \qquad (8.7a)$$

$$J_{\alpha\beta}(f)=\int_0^1\int_0^1 dx_\alpha dx_\beta[\int_0^1\cdots\int_0^1\frac{\partial^{2m}f}{\partial x_\alpha^m\partial x_\beta^m}\prod_{\gamma\ne\alpha,\beta}dx_\gamma]^2\ (8.7b)$$

$$J_{1,\dots,d}(f)=\int_0^1\cdots\int_0^1\left[\frac{\partial^{2md}f}{\partial x_1^m\cdots\partial x_d^m}\right]^2 dx_1\cdots dx_d.(8.7c)$$

Then the squared norm (8.1) on H_{TEPR}^{per} can be shown to be equal to

$$J_o(f)+\sum_{\alpha=1}^{d} J_\alpha(f)+\sum_{\alpha<\beta} J_{\alpha\beta}(f)+\cdots+J_{1\ldots d}(f). \quad (8.8)$$

As an example, we will consider below $f \varepsilon H_{TEPR}^{per}$ which consists only of a mean, all d main effects and the first order interaction between $x_1$andx_2. Thus f is of the form

$$f(x_1,\ldots,x_d)=f_0+\sum_{\alpha=1}^{d} f_\alpha(x_\alpha)+f_{12}(x_1,x_2), \quad (8.9)$$

where f_0 is a constant, $f_\alpha \varepsilon H_\alpha$, and $f_{12} \varepsilon H_{12}$. We can now define the periodic interaction smoothing spline as that function f_λ of the form (8.9) which minimizes

$$\frac{1}{n}\sum_{i=1}^{n}(y_i-f(x(i))^2+\lambda[\sum_{\alpha=i}^{d} J_\alpha(f_\alpha)+J_{12}(f_{12})], \quad \mathbf{(8.10)}$$

where $x(i)=(x_1(i),\ldots,x_d(i))$.

Using Lemma 5.1 in KW it can be shown that there is a unique minimizer of (8.10) in $H_o \oplus \sum_\alpha H_\alpha \oplus H_{12}$. An explicit representation for it may be found using this lemma and the fact that the reproducing kernel $K(x,z)$ for $\sum_\alpha H_\alpha \oplus H_{12}$ is given by

$$K(x,z)=\sum_\alpha B_m(x_\alpha,z_\alpha)+B_m(x_1,z_1)B_m(x_2,z_2) \quad (8.11a)$$

where

$$B_m(s,t)=$$

$$\sum_{v=1}^{\infty}\theta_v^{-2m}[\cos 2\pi vs\cos 2\pi vt+\sin 2\pi vs\sin 2\pi vt] \quad (8.11b)$$

A closed form expression for B_m may be found in Craven and Wahba (1979). GCVPACK may be used to compute $f_{\hat{\lambda}}$. In principle, $\sum_\alpha J_\alpha(f_\lambda)$ can be replaced by $\sum_\alpha w_\alpha J_\alpha(f_\lambda)$, where the w_α are positive weights, but problems concerning their estimation from the data have not been studied to date.

We will now sketch how to remove the rather restrictive periodicity conditions from H_{TEPR}^{per}. For g a function of one variable, let

$$L_0 g=\int_0^1 g(u)du \quad (8.12a)$$

$$L_v g=\int_0^1 g^{(v)}(u)du=g^{(v-1)}(1)-g^{(v-1)}(0), \quad (8.12b)$$

and let $L_{v(x_\alpha)}f$ mean L_v applied to f as a function of x_α. Then $L_{v(x_\alpha)}f=0$ for $v=0,1,\ldots,m$, $\alpha=1,2,\ldots,d$, any f in H_{TEPR}^{per}. Now, it can be shown that H_α is that subspace of the Sobolev space

$$W_2^m[0,1]=\{g: g, g',\ldots,g^{(m-1)} abs.cont., g^{(m)} \varepsilon L_2\}$$

of co-dimension $m+1$ which satisfies the $m+1$ conditions $L_v g=0$, $v=0,1,\ldots,m$. Let $k_v=\frac{b_v}{v!}$, $v=0,1,\ldots,m$, where the b_v are the Bernoulli polynimials, we have $L_v k_\mu=0$, $\mu\neq v$, $L_v k_v=1$, $\mu,v=0,1,\ldots,m$, and thus k_v is not in H_α. Let $W^0=span\ \{k_0,\ldots,k_{m-1}\}$ and let W^1 be isomorphic to $H_\alpha \oplus \{k_m\}$. Then it can be shown that W_2^m endowed with the inner product

$$<g,h>_{W_2^m}=\sum_v L_v g L_v h+\int_0^1 g^{(m)}(u)h^{(m)}(u)du \quad (8.13)$$

satisfies

$$W_2^m=W^0 \oplus W^1. \quad (8.14)$$

Letting $g \varepsilon W_2^m$ with $g=g_0+g_1$, $g_0 \varepsilon W_0$, $g_1 \varepsilon W_1$ we can call g the polynomial part of g, and g_1 the "smooth" part. Now let

$$H_{THPL}=W_2^m \otimes \cdots \otimes W_2^m \quad d\ times \quad (8.15)$$

$$=(W^0 \oplus W^1)\otimes\cdots\otimes(W^0\oplus W^1).$$

$$=(\prod_{\alpha=1}^{d} W_\alpha^0)\oplus(\sum_{\alpha=1}^{d} W_\alpha^1 \otimes \prod_{\substack{\beta=1\\ \beta\neq\alpha}} W_\beta^0)\oplus$$

$$(\sum_{\alpha<\beta} W_\alpha^1 \otimes W_\beta^1 \otimes \prod_{\gamma\neq\alpha,\beta} W_\alpha^0)\oplus\cdots\oplus$$

$$(\prod_{\alpha=i}^{d} W_\alpha^1),$$

where the Greek subscripts make explicit which variables are involved. We can now identify the "polynomial" subspace

$$H_0=\prod_{\alpha=1}^{d} W_\alpha^0,$$

the main effects subspaces

$$H_\alpha=W_\alpha^1 \otimes \prod_{\substack{\beta=1\\ \beta\neq\alpha}}^{d} W_\beta^0,\ \alpha=1,\ldots,d,$$

the first order interaction spaces

$$H_{\alpha\beta}=W_\alpha^1 \otimes W_\beta^1 \otimes \prod_{\gamma\neq\alpha,\beta} W_\alpha^0,$$

etc.

The induced tensor product inner product in H_{TEPR} is a natural extension of the inner product of (8.7) and (8.8). Letting J_α be the induced norm on H_α, etc., we can now seek f_λ in the new, non periodic version of, for example $H_0 \oplus \sum_\alpha H_\alpha \oplus H_{12}$ to minimize

$$\frac{1}{n}\sum_{i=1}^{n}(y_i-f_\lambda x(i))^2+\lambda[\sum_\alpha J_\alpha(f_\alpha)+J_{12}(f_{12})]. \quad (8.16)$$

Existence and uniqueness for any $\lambda>0$ can be shown via Lemma 5.1 in KW provided the design points $x(i)$, $i=1,\ldots,n$ are such that least squares regression in

H_0 is unique. The reproducing kernels for the various subspaces then follow:- The r. k.'s R_0 and R_1 for W^0 and W^1 can be shown to be

$$R_0(u,v) = \sum_{\nu=0}^{m-1} k_\nu(u) k_\nu(v),$$

$$R_1(u,v) = k_m(u)k_m(v) + B_m(u,v),$$

and the r. k. for H_{TEPR} with the inner product induced by (8.13) is

$$\prod_{\alpha=1}^{d} (R_0(x_\alpha, z_\alpha) \oplus R_1(x_\alpha, z_\alpha)),$$

thus, for example the r. k. for $\sum_\alpha H_\alpha \oplus H_{12}$ is now

$$Q(x,z) = \sum_\alpha R_1(x_\alpha, z_\alpha) \prod_{\beta \neq \alpha} R_0(x_\beta, z_\beta) +$$

$$R_1(x_1, z_1) R_1(x_2, z_2) \prod_{\beta \neq 1,2} R_0(x_\beta, z_\beta).$$

Given the r. k. an explicit representation for f_λ can be given, and, again GCVPACK can be used to calculate $f_{\hat{\lambda}}$. For $m=1$, $R_0(x_\alpha, z_\alpha) = 1$, H_0 is one dimensional as before, and we only replace B_m in the discussion of periodic spaces by R_1 and the same expressions hold. For $m>1$, a typical element of H_α with, say $\alpha = 1$ is now of the form

$$f(x_1, \ldots, x_d) =$$

$$\sum_{\nu_1, \ldots, \nu_d=0}^{m-1} f_{\nu_2 \cdots \nu_d}(x_1) k_{\nu_2}(x_2) \cdots k_{\nu_d}(x_d). \qquad (8.18)$$

The $\nu_2 = \cdots = \nu_d = 0$ term depends only on x_1 but the other terms do depend on the other variables albeit in a parametric (i. e. polynomial) way. The case $m=2$ is probably of special interest, then x_β with $\beta \neq \alpha$ enters at most linearly in functions in H_α.

There are now many interesting questions. Some of the major ones are - the development of good methods for choosing which interactions to include (GCV?), numerical methods for vary large data sets, methods for interpreting the results, development of confidence intervals, and so on.

Acknowledgements

This research supported in part by NASA under Contract NAG5-316 and by NSF under Grant DMS 8404970.

References

Anderson, J. A., and Senthilselvan, A. (1982), "A Two-Step Regression Model for Hazard Functions," *Appl. Statist.*, 31, 44-51.

Bates, D. M., Lindstrom, M. J., Wahba, G., and Yandell, B. (November 1985) "GCVPACK - Routines for Generalized Cross Validation." Technical Report 775, Dept. of Statistics, University of Wisconsin-Madison.

Burman, P. (June, 1985), *Estimation of Generalized Additive Models*, Rutgers University. (manuscript)

Cox, D. D., and O'Sullivan, F. (October, 1985), *Analysis of Penalized Likelihood-Type Estimayors with Application to Generalized Smoothing in Sobolev Spaces*, manuscript.

Craven, P., and Wahba, G. (1979), "Smoothing Noisy Data with Spline Functions: Estimating the Correct Degree of Smoothing by the Method of Generalized Cross-Validation," *Numer. Math.*, 31, 377-403.

Engle, R., Granger, C., Rice, J., and Weiss, A. (1983) "Nonparametric Estimates of the Relation Between Weather and Electricity Demand.." Discussion paper 83-17, San Diego: Dept. of Economics, University of California,.

Eubank, R. L. (1986), "Diagnostics for Smoothing Splines," *J. Roy. Stat. Soc. Ser. B*, 47. (to appear)

Friedman, J. H., Grosse, E., and Stuetzle, W. (1983), "Multidimensional Additive Spline Approxmation," *SIAM J. Sci.Stat. Comput.*, 4, 291-301.

Green, P., Jennison, C., and Seheult, A. (1983), "Comments to Nearest Neighbour (NN) Analysis of Field Experiments by Wilkinson, Et. al.," *Journal of the Royal Statistical Society, Ser. B*, 45, 193-195.

Green, P. J., and Yandell, B. S. (1985) "Semi-Parametric Generalized Linear Models." Technical Report#2847, Math. Research Center, U. of Wisconsin.

Green, P., Jennison, C., and Seheult, A. (1986), "Analysis of Field Experiments by Least Squares Smoothing," *Journal of the Royal Statistical Society, Ser. B*, 47.

Kimeldorf, G., and Wahba, G. (1971), "Some Results on Tchebycheffian Spline Functions," *J. Math. Anal. Applic.*, 33, 82-95.

Leonard, T. (1982) "An Empirical Bayesian Approach to the Smooth Estimation of Unknown Functions.." Technical Report 2339, Mathematics Research Center, Univ. of Wisconsin.

Nychka, D., Wahba, G., Goldfarb, S., and Pugh, T. (1984), "Cross-Validated Spline Methods for the Estimation of Three Dimensional Tumor Size Distributions from Observations on Two Dimensional Cross Sections," *J. Am. Stat. Assoc.*, 79, 832-846.

O'Sullivan, F. (1983), *The Analysis of Some Penalized Likelihood Estimation Schemes*, University of Wisconsin-Madison, Statistics Dept.. (Technical Report #726)

O'Sullivan, F., and Wahba, G. (1985), "A Cross Validated Bayesian Retrieval Algorithm for Non-Linear Remote Sensing Experiments," *J. Comput. Physcs*, 59, 441-455.

O'Sullivan, F., Yandell, B., and Raynor, W. (1986), "Automatic Smoothing of Regression Functions in Generalized Linear Models," *J. Am. Stat. Soc.*, 81, 96-103.

Shiau, J., Wahba, G., and Johnson, D. R. (Dec. 1985) "Partial Spline Models for the Inclusion of Tropopause and Frontal Boundary Information in Otherwise Smooth Two and Three Dimensional Objective Analysis." Technical Report # 777 University of Wisconsin-Madison Statistics Dept.

Shiller, R. J. (1984), "Smoothness Priors and Nonlinear Regression," *J. Am. Stat. Soc.*, 79, 609-615.

Silverman, B. (1982), "On the Estimation of a Probability Density Function by the Maximum Penalized Likelihood Method," *Ann. Statist.*, 10, 795-810.

Stone, C. J. (1985), "Additive Regression and Other Nomparametric Models," *Ann. Statist.*, 13, 689-705.

Villalobos, M., and Wahba, G. (March 1985), *Inequality Constrained Multivariate Smoothing Splines with Application to the Estimation of Posterior Probabilities*, Dept. of Statistics, University of Wisconsin-Madison. (Technical Report No. 756)

Wahba, G. (1978), "Improper Priors, Spline Smoothing and the Problem of Guarding Against Model Errors in Regression," *J. R. Statist. Soc. B*, 40, 364-372.

Wahba, G. (1984a), "Cross Validated Spline Methods for the Estimation of Multivariate Functions from Data on Functionals," in *Statistics: An Appraisal, Proceedings 50th Anniversary Conference Iowa State Statistical Laboratory*, eds. H. A. David, and H. T. David Iowa State University Press.

Wahba, G. (1984b), "Partial Spline Models for the Semiparametric Estimation of Functions of Several Variables," in *Statistical Analyses of Time Series*, Tokyo: Institute of Statistical Mathematics, 319-329. (Proceedings of the Japan U. S. Joint Seminar)

Wahba, G. (1985), "Comments to Peter J. Huber, Projection Pursuit," *Ann. Statist.*, 13, 518-521.

THE LANGUAGE OF DATA

Organizer: *James L. Dolby (deceased), San Jose State University*

Tables and Graphs as Language

Nancy Clark, Editing and Designing of Textbooks, Language of Data Project

Implications of the Language of Data for Computing Systems

William H. Rogers, The Rand Corporation and the Language of Data Project

The Language of Data: A General Theory of Data

James L. Dolby (deceased), San Jose State University; Nancy Clark, Editing and Design of Textbooks; William H. Rogers, The Rand Corporation

Intelligent Data Management

Henson Graves, San Jose State University; Ruth Manor, San Jose State University and Tel–Aviv University

TABLES AND GRAPHS AS LANGUAGE

Nancy Clark, Editing and Design of Textbooks; Language of Data Project

The theoretical framework of the Language of Data is discussed in Dolby, Clark, and Rogers (1986). This paper describes the "language" half of the theory in further detail.

Although we are not accustomed to thinking of visual displays as having formal language properties, tables and graphs follow all the rules of written language, from the larger organizational structure of text down to the sentence grammar of ordinary English. In tables these properties are reflected directly in the tabular form. To identify their counterparts in graphs it is necessary to look more closely at the common structure of visual and verbal language.

The Medium of Communication for Data

Data differ from other forms of information in two respects. With data the usual activities of information gathering, organization, and synthesis are carried out independently by people who have little or no direct contact with each other. As a result, what is ordinarily a simple sequence of events is, for data, a chain of communication.

The second difference is the focus of these activities. With written information the entire process is directed toward synthesis of the contents into final form for dissemination to others. With data, however, synthesis does not lead to a single end product. Analysis is a multiple activity, and while the communication chain does not end with the analyst, the activities are directed primarily toward this stage.

Both of these characteristics imply that, for data, the issue of communication arises long before the presentation stage. The full communication chain runs from data collection, through editing and revision, and storage and retrieval, to analysis and presentation. At every interface information is transferred from one stage to the next through a visual intermediary; "data" is not a spoken language. Although tables and graphs are the primary conceptual structures for data, it is their visible form that does the communicating.

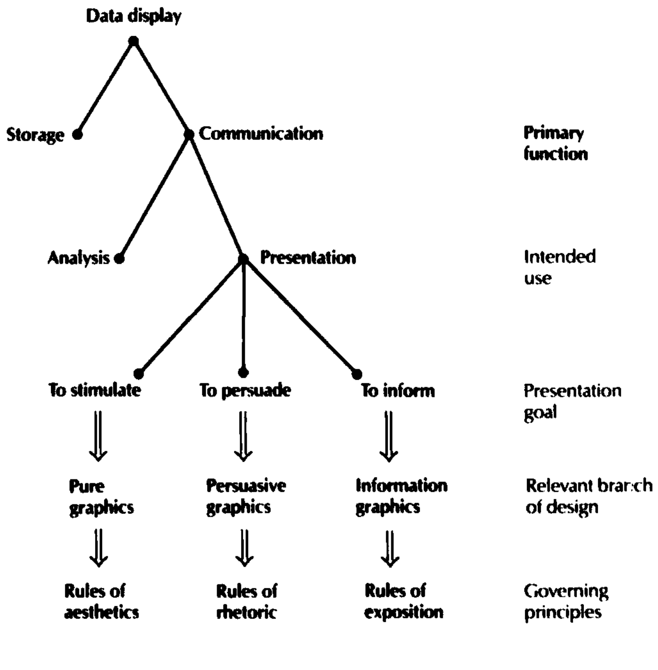
Because the communication function of data displays is usually treated as a presentation issue, the problems tend to accumulate at that level. However, some of the issues defined in Figure 1 arise as early as the data-collection stage. One is uncertainty about whether data displays are supposed to communicate at all--that is, whether tables and graphs should be viewed as a communication medium or as storage containers for data.

All tables have a capacity for communication which is inherent in the tabular form. For example, microdatabases and tables both have an information structure that corresponds directly to the organizational structure of text. A microdatabase, however, is specifically designed for information retrieval--the selection of individual variables, or subsets of variables, to be related in some context external to the database. Although any table can be used for this purpose, a table also has a communication structure; the limits of coherence require that the columns of a table have some logical relationship to each other. Thus for tables the design decision relates to primary and secondary functions, with the levels of access for each use organized accordingly. Archival tables are usually organized for access first at the retrieval level, and next at the communication level.

The difference in design objectives is more apparent in graphs. One type of graph designed explicitly for data storage is the standard heat-transfer diagram. However, the very qualities that make such graphs useful for their intended purpose make them correspondingly unuseful as communication devices. Although the most common problem is an effort to use a graph designed for one function for an entirely different function, the concept that information is stored in the data leads, at a deeper level, to a confusion of efficient storage with efficient communication. It also leads to the more important question of where the information does lie in a display.

The distinction between analysis and presentation, the second branch of the diagram, hinges on an equally fundamental issue. The most obvious difference is the difference in audience--self-communication versus communication to others. For the author of the communication, however, the two functions are related only by the fact that one

FIGURE 1 Basic functions of data displays
From Clark, Statistical Presentation--of What, to Whom, and for Which Purpose (1983)



Language of Data Project, Box R, Sausalito, CA 94966. Research supported by a grant from the System Development Foundation to San Jose State University.

often (but not always) follows the other. Analysis is the process that generates the content to be communicated; its transmission to others is essentially a writing problem.

Of course, analysis also takes place on the reader's side of the page, often so immediately that the transition from reader to analyst is too fleeting to grasp. As a result, the distinction between comprehension and analysis is similarly blurred, and the information appears to communicate itself without the aid of any vehicle.

Although the further distinctions at the bottom of Figure 1 relate to presentation, some of them might also apply in other branches of the diagram. (The three branches of graphic design are described in Clark, 1983). For example, the three purposes of presentation narrow the communication goal at all preceding stages to the communication of information, and this is in fact the goal to which the Language of Data is limited.

Some of the methodology of information graphics may also apply to the design of analytic tools. For example, perceptual persistence is utilized in textbook design to relegate successive levels of information to context as the ideas are absorbed, a process not unlike that in analysis, where discovery results from the accumulation of insight gained through successive views of the data.

Unfortunately the advantage of perceptual persistence during analysis backfires at the communication stage. We all have trouble seeing our own material from a state of innocence, but visual representations are particularly susceptible to the "Eureka problem." Once a particular meaning has been discovered, it seems to leap out of the page in almost any view of the data. As a result, the author of a visual display tends to read into it what he or she meant to show and assume that the table or graph actually communicates this information.

What the reader sees, of course, is the meaning (if any) to which he or she is visually directed by the table or graph. Since this last condition also holds for analytic tools, it is worth looking more closely at where the meaning lies in a data display.

The Relationship of Form and Content

The information value of a table or graph depends on the utility of the information it contains. Communication value, however, is the extent to which the form and content are a one-to-one match. This definition immediately implies the need for a formal separation of form and content, and as a next step, a definition of meaning at the content level.

Dolby's definition of the two components of information as the data and the operations on data involves a distinction so basic that it is rarely articulated (Dolby and Clark, 1982). However, it extends immediately to the corresponding components of language, the words and their syntactic relationships in a sentence.

The meaning of a sentence, however, lies in what the sentence says. Thus a third component of information is the intended meaning--the net result of a particular set of elements and their operational relationships.

These three components of information can be described in generic terms as the data elements, the set of operational relationships, and their resultant meanings, limited to the set of possible meanings in data displays:

Data elements + operational relationships = meaning

The data elements consist of all the elements of a complete datum, with the meaning of each descriptor cleanly defined by the classification scheme discussed in Dolby, Clark, and Rogers (1986). These elements, including the values, constitute the "words" in a data sentence. Since the relational statements in a table or graph are limited by the allowable operations on data, the set of meanings in the Language of Data is limited to statistical meanings.

Because the results of comparisons are often supplied by mental arithmetic, the derived data are commonly thought of as the information in a display. The information, however, lies in the entire relational statement, which may or may not include the derived datum. For example, in a statement of equality, $\underline{a} + \underline{b} = \underline{c}$, the term $\underline{c}$ has information value only with respect to its components--or as one of the components of a different relational statement at the next level of derivation.

The corresponding formulations for data displays refer to the visible counterparts of the content elements. In tables, as in verbal language, the syntactic relationships of the terms are implicit in the terms being related. For example, the form of a word usually specifies its functional role in a sentence. In a table these roles are denoted instead by spatial relationships. In tables the verbal elements carry the burden of communication:

Verbal elements + implicit relationships = implicit meaning

In visual language the situation is reversed. The most visible component in a graph is the set of relationships:

Visual elements + visible relationships = visible meaning

This formulation is a decomposition of a set of properties which are usually defined as single system. The properties themselves are discussed at length in Bertin's *Semiologie Graphique* (1983), a detailed application of Kandinsky's classic *Point and Line to Plane* (1926) to statistical data.

As Bertin points out, the properties of the graphic system are independent of content. In the communication of information, however, the objective is to relate them. One of the obstacles has been a mismatch in definitions. Whereas the entire graphic system is commonly thought of as visual syntax, the set of relationships at the content level is defined primarily in terms of the data component.

Although an exact match requires better articulation of the set of relationships in statistics, the three formulations above provide a framework for defining the properties of data displays directly in terms of the content elements they express.

The Language Properties of Tables

Although the descriptive information in tables is usually limited to the elements needed to identify the contents, a table is essentially a highly condensed form of text. In fact, when the descriptive elements are classified under Dolby's formalism for a complete datum, the resulting descriptor set is an even more condensed form of text (see Dolby, Clark, and Rogers, 1986).

Most of this information is given in the table title, which generally identifies the universe of discourse. In Figure 2, for example, the title covers four of the five dimensions of description. The space is U.S., the function discussed is consumption, and the matter consumed is energy. The observer is rarely specified in derived data, but in this case the observers (EIA's respondents) were the end users.

There is an important distinction in classification between the information specified in the description and that supplied from external knowledge about the data. The aspect descriptor, for example, does not require an inference; the BTU is formally defined as the unit of heat content. Heat content, of course, is merely the aspect discussed in this representation of the data, not the aspects (various) represented by the primary data.

The same words that describe the contents of the table, however, also have another function. At the data level they become the words in set of highly structured sentences all dealing with U.S. consumption of energy. In short, we are looking at an ordinary paragraph--a series of sentences all of which are expansions on a single topic.

The subject of the table is the variable arrayed in the stub--in Figure 2, the end-use sectors. The topic of the discussion is energy consumption, and the specific topic is the amounts of consumed, the set of values in the table field. The column heads comprise the third discourse variable, the "statement variable." As a set, they specify the nature of the statements made in the table, in this case the change in consumption over time. In other words, the stub describes what the table is about, and the column heads describe what the table says.

FIGURE 2 The descriptive elements of a table

U.S. CONSUMPTION OF ENERGY, BY END-USE SECTOR AND YEAR: 1977-1980

	Quadrillion BTU's		
End-use sector	1978	1979	1980
Residential & commercial	28.159	27.462	28.283
Industrial	29.373	31.551	30.284
Transportation	20.612	19.950	18.628

Source: James L. Dolby, Data Analysis: Tables In, Tables Out, Online '84, 1984. Data from Monthly Energy Review, June 1981, p. 18.

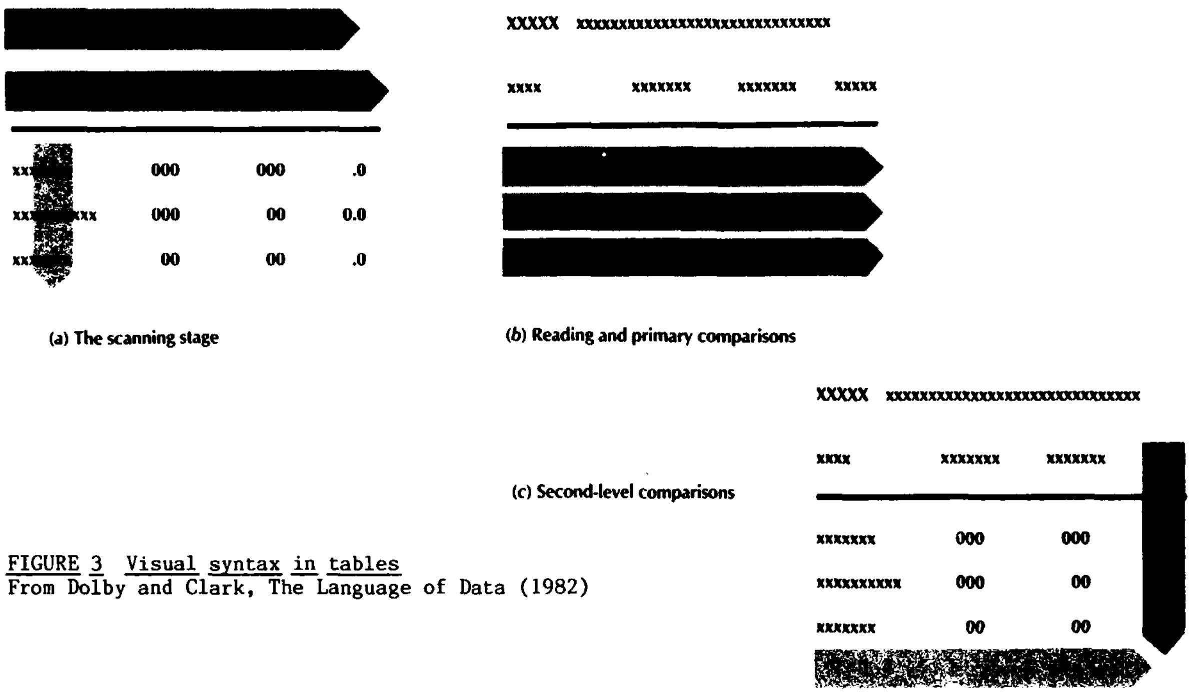
In a formal table all three discourse variables are named in the table title. The topic component of the title, which states the topic of the table, refers directly to the elements in the table field. The next two components usually appear in the partition rules. The subject component names the variable listed in the stub, and the statement component names the category in the column heads--generally in that order. Thus a properly constructed table title provides a useful structural description of the table for retrieval. Data are stored (and later, used) by classification category, they are retrieved by discourse category.

Although most dictionaries give only circular definitions of subject and topic, the difference between them shows up in their higher-level structures. The chief characteristic of the subject is unity, whereas coherence refers to the topical progression. Thus at the publication level the subject is a constant across the publication; topics, however, come in sets by definition. Once the subject has been partitioned off, the contents are organized into some logical topical sequence, with the discussion under each main topic partitioned into topical subsets, which also have a progressive relationship.

The same properties show up in tables as a single variable in the stub and a set of variables in the columns, often grouped into subsets by spanner heads. One of the requirements for the stub is that the subject elements be an aggregatable set. In contrast, the columns may be related by any signs of operation; the only requirement is that they all belong to some category that can be named. In some cases this higher-level node may be fairly high up in the topic structure. The topic structure itself is an aggregation tree, of course. However, the fundamental operations on data apply not only to nominal variables, but to the domain of visual variables in graphs.

The relationships between visual and verbal language is easiest to see in tables. A table, like any other visual representation, exists at the most fundamental level as a set of visual events which are perceived and organized by the eye in successive stages of resolution. The process is similar to the effect of decreasing distance. For example, if a page of text is held far enough away, the only discernible form is the page. As the distance is decreased a printed image becomes visible, first as a gray area and then as a pattern of uniform lines. Well beyond reading range, the lines become recognizable as strings of words, and eventually enough letterforms become discernible for the words to be read. For material at close range, the corresponding progression from peripheral vision to attention and perception is merely an instantaneous change of focus.

With tables, then, there is quite a lot that goes on before the reader gets to the data. The first perceptual task is bounding the set of events to be organized, and the next is locating the visual characteristics that identify the image as a table. With some tables this is a nontrivial problem, but even beyond reading range most tables have two identifying features: they contain a matrix area, with a discernible pattern of rows and columns, and they consist of material that is to be read.



(a) The scanning stage

(*b*) Reading and primary comparisons

(c) Second-level comparisons

FIGURE 3 Visual syntax in tables
From Dolby and Clark, The Language of Data (1982)

The second characteristic provides an important cognitive cue: recognition that material is to be read immediately directs the viewer to the starting point for reading. In Western cultures tbis is the upper left corner of the image--usually the beginning of the table title.

From this starting point the rest of the table is scanned to locate and identify its structural members. This search for visual structure takes place with all visual displays, but the scanning pattern for tables follows a specific sequence. With all written materials scanning tends to be vertical, in the sense that it starts at the top and progresses downward. Reading is horizontal, however, so that the scan usually consists of a horizontal sweep of the column heads, followed by a vertical sweep of the stub.

This sequence of directional responses narrows the scan down to the critical area--the values in the table field. Whereas the preceding responses are universal, the processes that go on in this part of the table vary widely with the reader's experience, interests, and ability to handle numbers.

Experienced data users will usually look at the field as a whole to determine the range of values, and then scan horizontally, vertically, and perhaps along the diagonal axis for any obvious patterns of increase or decrease. The general reader may scan the table field only for obvious exceptions to uniformity, or simply begin reading across each row as soon as the column heads and stub have been identified.

At both ends of the experience scale, however, before the data can be analyzed they have to be read--and even numbers are read from left to right. As a result, it is the directional preference for reading that determines the order in which the comparisons are seen in a table. For example, both tables in Figure 4 contain the same data. In the first table the most immediate observation is that the prison population increased considerably from 1976 to 1980; in fact, the female population almost doubled. In the second table the first observation is that the ratio of males to females is more than 20 to 1. Although both conclusions can be drawn from either table, the orientation of the table matrix determines the order in which this information becomes visible.

The reading stage in Figure 3b reduces each row of the table to a result, a new datum at the next level of derivation, so that the set of results can be compared--this time in a vertical column. Thus, if we stick to the primary comparisons in the table, there is another right-angle change which now brings the result column into focus. Although data are compared horizontally, numbers

FIGURE 4 Effect of orientation of the table matrix on meaning

California state prison population:

	1976	1980
Males	15,891	20,608
Females	590	1,039

	Males	Females
1976	15,891	590
1980	20,608	1,039

Data from California Prisoners, California Department of Corrections, 1979 and 1983.

are easier to compare in a column. Hence taking the primary comparisons across the table now puts the next level of comparison into the most advantageous position for the higher-level operations of statistics.

Because the last stage in Figure 3 is often the starting point for the analyst, it is easy to miss the language structure at the preceding stages. However, the visual sorting that goes on at the scanning stage merely establishes the discourse structure of the table, the framework for communication. Communication itself takes place at the reading level. This means that the elements being read have to have some syntactic relationship that results in an intelligible statement.

The most immediately available syntax in a table lies in the spatial relationship of its parts. Unlike prose, in tables each structural element partitions directly into its grammatical counterpart at the sentence level. As a result, all table sentences have the same basic form. The subject is the item listed in the stub, and each column entry is a statement of quantity about that subject. Thus, across the whole row, the sentence is a simple sentence with a compound predicate.

In the first table in Figure 4, for example, the first row might be expressed in telegram style as

(California prison population for)
Males: 15,891 in 1976 and 20,608 in 1980

Unlike prose, the subject of a table sentence corresponds directly to the subject of the discussion. The only verb in a table sentence is is or was (or for predictions will be); hence the values in the columns form the main predicate, and the column heads function as governing clauses. Although the device holds unpleasant memories for many, these relationships are easiest to see in sentence diagram:

	(in) 1976	(in) 1980
Males	(was) 15,891	(and) 20,608

The point of the statement, however, is the comparison of the two populations--about 16,000 in 1976 compared to about 20,500 in 1980. That substitution for and is the link between statistics and simple prose. It is also the source of meaning in a data display. All the statements in a table or graph are statements of comparison--statements of additivity, proportionality, and so on. The meaning in any particular instance depends on which of these relationships is specified.

For example, the transpose in Figure 4 changed the subject (but not the topic) of the table. In the first table the subject was gender, and in the second table it was the year. But it is the statements about the subject that are of interest, so a more important consideration is that the two tables have different predicates. As a result, they convey different information, despite the fact that they contain exactly the same data.

Notice that the comparisons in Fig. 4 were at the variable level, not the data level--a comparison of whole columns across the table. In the first table the primary comparison was a difference, and in the second table it was a ratio. The only place in the table these column relationships are expressed is the column heads--the statement variable. It is the relationships stated in the column heads of a table that are shown in the field of a graph.

Statements of Comparison in Graphs

Although the properties of visual language are usually defined as a system, they partition easily into to the two components of a relational statement, the set of terms and the set of relationships. In tables the focus is on the terms in the equation, and in graphs it is on the signs of operation.

The distinction between the discourse structure of text and communicative syntax, which are not usually thought of as connected in prose, provides us with a second partition. The visual structure of an image, as opposed to the syntactic relationships that convey meaning, is the direct counterpart of the discourse structure of text. In a table, for example, the process of visual sorting that takes place at the scanning stage identifies the subject, topic, and statement categories--which also corresponds (or should correspond) to the information structure of the contents.

Like a table, a graph also consists of a series of sentences all of which are an expansion on a single topic. In graphs, however, the starting point for reading is not the upper left-hand corner of the image; it is the most prominent visual element. The first sentence in the graph is the relationship of this element to the next most prominent element, where the prominence hinges on common aspect. Although a number of factors are involved, the levels of subordination in a graph are essentially a change in common aspect at each level.

The role of aspect as a structural link in data classification is discussed in Dolby, Clark, and Rogers (1986). It applies here in the same sense, as both the name of the variable and the specific topic. The three aspects of color--hue, chroma, and value (light/dark)--are well known. However, visual elements also have shape, size, orientation, and so on. The relational elements all have direction as well as extent, and a common direction may be the common aspect.

In a table, for example, the aspect descriptor links the topic elements in the table field to the common topic at that level, the topic term in the title. Where the variables differ in aspect, the common aspect is their relationship, for example, Boyle's law in the case of pressure, temperature, and volume. The higher-level structure of aspect is independent of the topic structure; however, it is often of particular interest in exploring data.

Aspect also serves as the pivot term between variables. In a table it functions as a hinge between the subject variable in the stub in the stub and the common topic--the difference between "temperatures in patients" and "temperature in malaria." As in verbal language, the change in antecedents represents a transfer of attention from one object to another. As a result, although the statements communicated by a graph are not limited to elementary comparisons, they can all be decomposed into a sequence of simple comparisons. All visual effects stem from a comparison with some visible or imaginary reference. At the "reading stage" in graphs this reference shifts, with the endpoint of one sentence becoming the starting point for next. The meaning, however, lies in the resultant. In general, graphs follow the concepts of vector algebra rather than linear algebra.

In graphs as well as tables, however, the basic reference level is the horizontal axis. For example, if the image in Figure 3b is visualized as a bar chart, the first information it shows is the lengths. If the page is turned 90 degrees, to make the image a column chart, it now shows comparisons of lengths--again read across the graph. As with the two tables in Figure 4, both pieces of information can be extracted from either graph, but the orientation of the graph determines the order in which this information becomes visible.

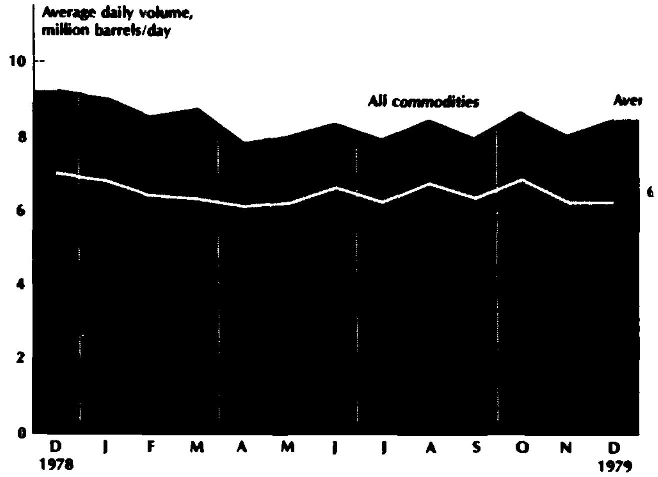
All graphs, in fact, show both levels of derivation. In a graph, however, the order can run in either direction--as a forward projection to the next level, with the focus on the relationships at first level, or as a backward projection to the components, with the focus on the derived level. As a result graphs are capable of much finer degrees of meaning than tables, but they also require a decision that takes care of itself in a table. In a graph the focus has to be on one level of derivation or the other, since the discussion cannot progress simultaneously in both directions.

Because the data elements are the carriers of meaning in tables, the column heads of a table are, in effect, an equation with elliptical signs of operation. These are the relationships stated in graphs; hence in a graph they have to be specified. For example, the graph in Figure 5a shows crude oil imports in relation to total imports. In operational terms the statement is a simple proportion. The primary comparisons are the change over time, so the proportion itself is a second-level comparison, a comparison of the structure of the two horizontal comparisons.*

It is easy enough to deduce from the two curves that crude oil imports accounted for most of the fluctuation in the total. But suppose the whole point of the discussion is the fluctuation of crude oil in relation to the stability of other commodities. Fluctuation and stability are two different aspects of the data, so the relationship in this case is a ratio, not a proportion. The graph in Figure 5b shows exactly the same data, but instead of a component relationship, the two separate curves enable the reader to see the ratio of fluctuation to stability.

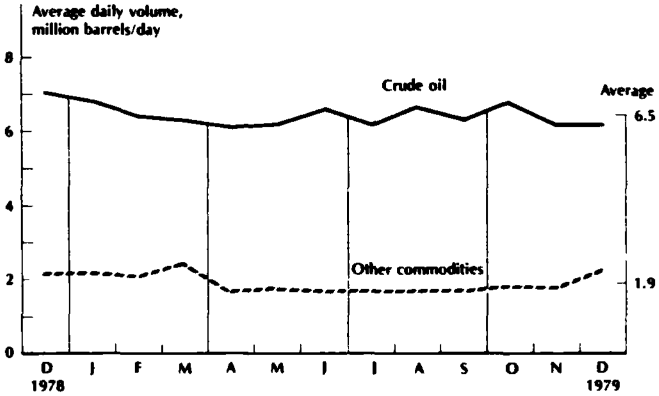
Again, the secondary information in the graph can be deduced; total imports is simply the sum of the two quantities. Notice, however, that this deduction involves mental arithmetic, not perception; the two fall in different domains. In fact, although the graph shows the ratio of two quantities, the primary visual comparison is not the quantitative ratio. It is the ratio of fluctuation to stability.

If the point were the amounts, the most appropriate form would be one that focuses on this



Includes imports of crude oil to the Strategic Petroleum Reserve. Data from the *Monthly Petroleum Statistics Report*, January 1980.

(a) A proportional relationship of two quantities



Includes imports of crude oil to the Strategic Petroleum Reserve. Data from the *Monthly Petroleum Statistics Report*, January 1980.

(b) A ratio of two properties: fluctuation and stability

FIGURE 5 Operational relationships in graphs: From Clark, Sample Pages and Specifications: Monthly Petroleum Statement (1981)

*At the time of his death Dolby was working on a statistical basis for the comparison of comparison structures.

Current source: Clark; 1981
Original source: EIA; 1979-1980

Observer: Petroleum importing companies
Matter: Crude oil
Function: Imports
Space: U.S.
Time: 12/78 to 12/79

Aspect: Volume, mbbl [→ Imports (F)]
Domain: 9.2, 9.0, . . . , 8.4

Aspect: Fluctuation [→ Imports (F)]
Domain: Pattern of variation
Domain: Amount of variation
Domain: Variation with respect to reference

FIGURE 6 Possible descriptor set for Figure 5

aspect of the data instead, a table. There are two factors involved. One is the focus on the particular aspect of the data most efficiently communicated by numerical symbols. The other involves screening out factors that distract from that focus, and in particular those elements that show something else.

The visual variable that makes fluctuation and *stability explicit in Figure 5b is shape; thus in* another view of the data this variable will be the chief source of noise. Removing it altogether implies a table--unless, of course, the focus is on some other aspect of the data, expressed by a different visual variable.

The descriptor sets for data, discussed in Dolby, Clark, and Rogers (1986), have especially important implications for graphs. Whereas the data component is the visible information in a table, in graphs the entire descriptive structure has to be supplied through another domain (as with the labels on graphs). In an interactive system this information would have to be available regardless of the mode of representation.

Descriptor sets play a more immediate role, as a mechanism for specifying the meaning to be communicated by a table or graph. The form developed for the classification of individual variables in a microdatabases is designed as independent mode representation which summarizes the information structure of the variable as well as its contents. The variables in a graph, however, require more precise specification of the intended meaning. In Figure 6, for example, the aspect descriptor links structures in more than one domain of representation.

Conclusion

Although the issue is usually posed as "tables versus graphs," for interactive use the answer may be both, depending on the analytic step. There are a number of situations in which the analyst may want to switch back and forth from one mode of representation to the other, either to focus on a particular aspect of the data or to move to a computational step. A table manipulator designed specifically for interactive use is discussed in Rogers (1986).

Once the applicability conditions for relating two variables have been defined, the descriptor sets will have a communication structure as well. Specification of the form of display will then be a matter of matching the aspect of the data the analyst wants to see, first to its domain at the content level, and then to the aspect of representation that makes this information visible. The development in this area has just begun, but there is reason to think the language of data will ultimately be trilingual.

REFERENCES

Clark, Nancy: Sample Pages and Specification: Monthly Petroleum Statement, Language of Data Project, 1981.

---: Statistical Presentation--of What, to Whom, and for Which Purpose?, presented at the Joint Statistical Meetings, Toronto, 1983.

Dolby, James L., and Nancy Clark: The Language of Data, Language of Data Project, 1982.

Dolby, James L., Nancy Clark, and William H. Rogers: The Language of Data: A General Theory of Data, presented at the 18th Interface Symposium, Fort Collins, CO, 1986

Rogers, William H.: Implications of the Language of Data for Computing Systems, presented at the 18th Interface Symposium, Fort Collins, CO, 1986.

IMPLICATIONS OF THE LANGUAGE OF DATA FOR COMPUTING SYSTEMS

William H. Rogers [1]
The Rand Corporation and the Language of Data Project

The Language of Data (LOD) is a program of basic research into the communication of quantitative information. The concepts developed under this umbrella range from the formal definition of a single datum through formalisms for microdatabases, tables, graphs, and the relationships between them. The theoretical aspects are covered in companion papers by Dolby, Clark, and Rogers [2], and Clark [3].

This paper discusses the kind of computer system envisioned under the Language of Data, including applications which have been implemented and elements not yet developed. As such, the system represents a design consideration for future software developers rather than a finished product. Moreover, the set of programs discussed here should not be confused with the theory itself. The understanding of the structure of the datum and of tables developed by this project could be applied to other programs as well.

The envisioned programs focus primarily on three themes: First, they focus on a principal application of LOD, the documentation of large databases. Second, they illustrate some of the formal ideas (computation with descriptors and applicability conditions) in ways familiar to statisticians. Third, they incorporate some of the insight into table structure to provide several natural and powerful tools for the data analyst.

The Overarching Computing Plan

The "grand plan" for a computing system is represented by the diagram in Figure 1. This diagram is especially relevant to a large survey study such as those Rand or many federal agencies would perform. Individuals are surveyed in several different ways and the data are collected into a microdatabase. The organizational structure of the survey itself results in a houselike structure for the microdatabase, as shown in Figure 2.

The individuals (cases) form the stub of the microdatabase and the variables are named in the column heads. Each variable comes with a set of descriptors that make up the second story of the house, and the attic consists of descriptor sets at the topic level that tie the columns of the microdatabase together. The field--the two-way matrix of numbers--is the portion of the data structure statisticians and statistical packages focus on. The overall structure might be viewed as a shorthand device for summarizing the descriptive content of each individual datum. However, this is a case in which the whole is greater than the sum of its parts.

LOD is not concerned with relational structures and other features of certain existing databases. Statisticians typically view these as expanded matrices and aggregate them to a more convenient unit of analysis, and then merge the results with other data having comparable units of analysis.

The first step is to find our way around by means of the descriptor sets. They themselves form a matrix in which the columns correspond to the descriptors and the cases are the variables of the microdatabase. This is the transpose of the way they appear in the house structure. The computerized tool for doing this is a descriptor manipulator called IDEA [4].

IDEA is an experimental program which operates on the transposed matrix of descriptors, using both familiar and novel tools of data analysis. It is especially good for moving around in the data to gain a view of the whole, and for visually selecting and arranging data (in terms of their verbal descriptors) using techniques for holding some of the data fixed and sorting others. There are also operators that uncover hierarchic structure through a highlighting technique sometimes called "slipping." More details about the program are given below.

This kind of analysis was especially popular when statisticians kept data on wall charts. It

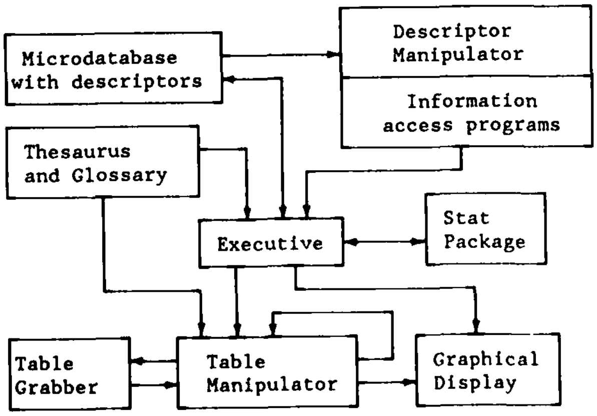


FIGURE 1

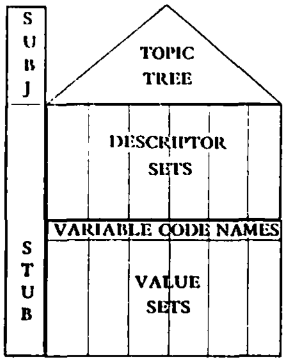


FIGURE 2

became unfashionable when computers made it more difficult to find one's way around a large data set than it was to do a regression analysis. In my experience as a consulting statistician, forcing the client to visually confront his or her data is more effective in revealing truth than the most powerful of statistical tests.

The upper levels of the microdatabase structure also provide indexing possibilities, for locating individual variables or subsets of variables at the lower levels. This is important in a large study which may involve many collaborators and 10,000+ variables. One can envision library-style searches of the descriptor sets, together with browsing capabilities among *similar* sets. These possibilities are currently being explored at Rand.

Moving down to the next level of Figure 1, we have the key undeveloped piece, the "Executive Intermanipulator" and its slave, the ubiquitous statistical package. The job of the Executive Intermanipulator is to convert data in microdatabase form to tabular form by aggregation, tabulation, summarization, or the more complex operations of statistical analysis. This activity is usually performed by the primary statistical analyst.

To aggregate successfully, the Executive Intermanipulator must know the rules of aggregation. It should check the glossary to determine whether the quantity being aggregated is additive and the thesaurus to know whether the units being aggregated oyer form an exhaustive partition.

The output of the Executive Intermanipulator might be a table or a graph. Adequate titles, *column heads, source information, units,* and labels would be generated from the microdatabase. These might be awkward in wording; they would be generated algorithmically and automatically.

The Executive Intermanipulator is also responsible for operations on microdatabase variables that create other microdatabase variables. For example, it would draw on the glossary to determine that population divided by land area gives population density and it would query the user if he or she attempts to subtract land area from mean January temperature, since that constitutes an invalid comparison (incommensurate units of measurement or descriptors that differ in more than one dimension).

If the result of the operation is a table, then the table should be in standard computer table interchange format. This computerized format breaks down the table into its structural components. (The structural components of a table are described in Clark [5].)

Tables may also be drawn from online sources. Gateway, an experimental program developed by the San Jose State University Mathematics Clinic [6] under the sponsorship of the Language of Data Project, can query the Lockheed Dialog information retrieval system for certain types of online tables. These tables are also generated in the standard interchange format. Additional information on the Gateway program is contained in the Mathematics Clinic report [6].

The key tool for secondary analysts is the Table Manipulator. This program, developed by Rogers, is written in a general-purpose language and has been implemented in experimental form on an IBM-PC. The Table Manipulator displays and operates on tables in the standard format. In addition to select and arrange operators characteristic of the descriptor manipulator, the Table Manipulator can combine tables and perform transformations and statistical analyses. Exploratory data analysis methods are available. The table structure and the contents of the table, including verbal information, are used (in conjunction with the glossary) to guide the computation and to check applicability conditions. For example, if the user requests aggregations, it checks that all the *appropriate partitions* are represented.

An important property of the Table Manipulator is that it operates on tables to produce objects which are themselves tables.

The Table Manipulator shares with spreadsheets the fact that changes in the data can be immediately reflected in the results. However, it differs from spreadsheets in several important respects. The Table Manipulator is aware of table structure and its implications, whereas spreadsheets require the user to specify what this structure is. The Table Manipulator can compute with the labels, the spreadsheet cannot. Taking advantage of the first two properties, the Table Manipulator can invoke statistical methodology without requiring specification at the individual datum level of what to do. The Table Manipulator has available a library of related information (via the Gateway program or other statistical analysis) to draw upon the analysis. Finally, it permits revision and editing, with full access to a step-by-step reanalysis of the data.

If the Table Manipulator is a tool of analysis, what constitutes an adequate audit trail for such work? Most interactive statistical packages create *a file of commands that have been executed. The* Table Manipulator also has this capability. The analyst can then go back to a previous step by rerunning the sequence of commands up to the given point. The analysis can also be repeated with new data, making it possible to do "yesterday's analysis on today's data."

The Table Manipulator (and the descriptor manipulator IDEA) go beyond this by giving the user full access to displays in previous steps, recorded as a state of the data and display rather than a fixed screen of characters. The analyst can go back to a previous state and continue from there simply by pushing a few keys. Moreover, if editing changes have been made, the refreshed displays can reflect the new information.

Elaborating on this technology leads us to a tool that enhances or clarifies the structure of the comparisons in tables. A comparison structure is a systematic way of making comparative inferences from the contents of the table. For example, each column or row might be compared with its neighboring column or row. The place to start is by elucidating comparison structures employed by naive and expert readers. More complex technologies move toward exploratory data analysis ideas pioneered by Tukey [7].

The Table Manipulator is a considerably different tool from a table-producing language such as TPL [8]. TPL is designed to produce attractive tabular displays from data sets but is not designed to operate on the data in the tables, *except though certain formatting* commands. That is, TPL does not simulate the structural elements of tables, but simply produces them line by line.

AD-A191 296 COMPUTER SCIENCE AND STATISTICS PROCEEDINGS OF THE 2/3
SYMPOSIUM ON THE INTER..(U) COLORADO STATE UNIV FORT
COLLINS T J BOARDMAN 26 AUG 87 AFOSR-TR-88-0153
UNCLASSIFIED AFOSR-86-0070 F/G 12/6 NL

1·0
2·8
3·15
3·5
4·0
4·5
2·2
2·0
1·1
1·8
1·25
1·4
1·6

The Table Manipulator is more like a spreadsheet in its feel, but differs from a spreadsheet in several respects. First, it takes advantage of table structure, including the inherent row and column structure and verbal information, such as "Total". Second, it has two-way table operators, such as analysis by means and exploratory data analysis operators, built in. Third, it has the ability to track what has changed and to go back and forth between displays. Fourth, there is a library of information available through sources such as Gateway that may be combined with the given table for use in making comparisons.

Some related work has also been done on the kind of mathematically rigorous language and table structure needed for the overarching system outlined here. Graves and Blaine [9], working in collaboration with the Language of Data Project, have described a computer language called ALGOS which facilitates the description of statistical methods in terms of algorithms or applicability conditions. ALGOS has extensible data types, so that the descriptive component of a datum can be carried and processed with its numerical component. Graves and Manor [10] have discussed the structure of a table in this framework.

Descriptors and the Descriptor Manipulator

A basis of the Language of Data theory is that the datum contains both a descriptive and a numerical part, where the descriptive part consists of a common, one-level classification of information. The terminology in the following examples is defined in Dolby, Clark, and Rogers [2].

Thus we might create two data, as shown in Figure 3a, which would be combined with the stub in a statistical matrix as shown in Figure 3b. The descriptors for the stub, generated at an earlier stage, contain information that would be crucial in a real data set.

statistical matrix as shown in Figure 3b. The de-

IDEA, the descriptor manipulator, processes a matrix of descriptive values consisting of the entries for Source,..., Domain. (Recall that this is the top section of Figure 1.) Like any matrix of numerical values, the descriptive values can be selected and arranged in useful formats. The following examples are from an LOD classification of the General Social Survey [11]:

```
       GSS83                       Version          0/ 0

              1          2          3   ...  7
        ID    Observer   Matter     Funct...  Domain
---------------------------------------------------------
 1    1    Adult mem  Adult mem  Type ...   Professio
 2    2    NORC       Adult mem  Occup...   0-9, 10-1
 3    3    Adult mem  Adult mem  Emplo...   Self-empl
 4    4    Adult mem  Adult mem  Type ...   Agricultu
 5    5    NORC       Adult mem  DOT o...   Relationa
 6    6    NORC       Adult mem  Relat...   Synthesiz
 7    7    NORC       Adult mem  Relat...   Mentoring
 8    8    NORC       Adult mem  Relat...   Setting-u
 .    .    .          .          .          .
 .    .    .          .          .          .
 .    .    .          .          .          .
20   20    NORC       Spouse     Occup...  Lowest le
---------------------------------------------------------
                                              On   0
                                          Row  1 Col 7
```

The display fills the IBM-PC display (25x80) and has function key commands which instantly scroll through the database. The function keys expand or contract fields.

One can arrange rows and columns in a specific order, as shown in the next display:

FIGURE 3 Structural Details of MICRODATASE

(a) Two data

Source:	Rand
Observer:	Rand Nurse
Matter:	Mary Jones
Function:	Systolic Blood Pressure
Space:	Rand Examination Center
Time:	8 am, January 6, 1986
Aspect:	Pressure, mm Hg (F) Systolic BP
Domain:	120

Source:	Rand
Observer:	Rand Nurse
Matter:	John Smith
Function:	Systolic Blood Pressure
Space:	Rand Examination Center
Time:	8 am, January 6, 1986
Aspect:	Pressure, mm Hg (F) Systolic BP
Domain:	135

(b) Data in microdatabase form

	Stub	Variable
Source:	Rand	Rand
Obsvr:	Survey Ctr.	Rand Nurse
Matter:	Patient	Patient
Functn:	Identity	Systolic Blood Pressure
Space:	Universal	Rand Examination Center
Time:	Universal	8 am, January 6, 1986
Aspect:	Name (F) Id.	Pressure, mm Hg (F) Systolic BP
Domain:	Alphabetic	120,...,155
Values:	Mary Jones	120
	John Smith	135
	...	...

```
        GSS83                   Version      0/ 0

            1         2         3         4   ...
        ID  Space     Observer  Matter    Funct...
--------------------------------------------------
 1  1   Continent Adult mem Adult mem Type ...
 2  2   Continent NORC      Adult mem Occup...
 3  3   Continent Adult mem Adult mem Emplo...
 4  4   Continent Adult mem Adult mem Type ...
 5  5   Continent NORC      Adult mem DOT o...
 6  6   Continent NORC      Adult mem Relat...
 7  7   Continent NORC      Adult mem Relat...
 8  8   Continent NORC      Adult mem Relat...
 .  .   .         .         .         .
 .  .   .         .         .         .
 .  .   .         .         .         .
20 20   Continent NORC      Spouse    Occup...
--------------------------------------------------
ARR 4 1 2 3                       On   0
                                  Row  1 Col 7
```

It is also possible to sort on any particular row or column.

The descriptor manipulator has two interesting operators used in conjunction with its sorting capabilities. The FIX operator highlights a set of rows and columns and keeps them in a fixed position on the screen, sorting or arranging around them wherever requested to rearrange data. This makes it possible to compare a set of fixed information with another set physically located in a different part of the display without having to create an artificial sort key or otherwise disturb the key information.

The SLIP operator uses **bold (*high intensity*)** to display hierarchic structure. If an item and all the items left of it are the same as in the previous row, then the item is not in bold. When a column entry changes from the one above it, that entry and all those to the right are shown in bold.

The descriptor manipulator also has the ability to transpose the data matrix, revealing expanded detail in the descriptor sets.

Operation of the Table Manipulator

The basic operating format of the Table Manipulator is a display approximating the desired form of the table:

```
1.          U.S. CONSUMPTION OF ENERGY,
     by END-USE SECTOR and by YEAR: 1977-1980

                      (BTU x 10**9)

                          YEAR

END-USE SECTOR       1977    1978    1979    1980
Residential & comm 27.569  28.159  27.462  27.283
Industrial         29.024  29.373  31.551  30.284
Transportation     19.735  20.612  19.950  18.628
  All end-use s[2] 76.332  78.150  78.968  76.201

SOURCE: Monthly Energy Review, June 1981, p. 18

NOTE 1: 07/05/84,JLD
NOTE 2: Totals may not equal sum of components
        due to independent rounding
--------------------------------------------------
Command?
```

The number in the upper left of each display is the display number generated by the program.

A command to total the columns would produce display 2:

```
2.          U.S. CONSUMPTION OF ENERGY,
     by END-USE SECTOR and by YEAR: 1977-1980

                      (BTU x 10**9)

                          YEAR

END-USE SECTOR       1977    1978    1979    1980
Residential & comm 27.569  28.159  27.462  27.283
Industrial         29.024  29.373  31.551  30.284
Transportation     19.735  20.612  19.950  18.628
  All end-use s[2] 76.332  78.150  78.968  76.201
  Total            76.328  78.144  78.963  76.195

SOURCE: Monthly Energy Review, June 1981, p. 18

NOTE 1: 07/05/84,JLD
NOTE 2: Totals may not equal sum of components
        due to independent rounding
--------------------------------------------------
Command? total
Command?
```

The rows or columns may be sorted with a command:

```
    Command? sort 1980
```

Suppose we now want to look at a table of sport parachuting deaths which is stored as 'dpara' on the disk in our table interchange format:

```
    Command? read dpara
```

We might then calculate both the row and column totals:

```
5.      DEATHS FROM SPORT PARACHUTING,
        by JUMP EXPERIENCE and by YEAR

                    YEAR
Number
of jumps       1973   1974   1975   Total
1-24             14     15     14     43
25-74             7      4      7     18
75-199            8      2     10     20
200+             15      9     10     34
Unreported        0      2      0      2
  Total          44     32     41    117

SOURCE: Metropolitan Life Insurance Company,
        Stat. Bull.:3 p 4 (1979)

NOTE 1: 03/13/86,NC
--------------------------------------------------
Command? read dpara
Command? totals
Command?
```

The Table Manipulator also produces transformations of the data. When a transformation is done, a question arises of what to do with totals. Ordinarily a total should be eliminated, but a mean should be recalculated; so the program is designed to do this.

The command to do a square-root transformation would produce display 6:

```
6. (TRANSFORMED) DEATHS FROM SPORT PARACHUTING,
          by JUMP EXPERIENCE and by YEAR

                        YEAR
Number
of jumps       1973   1974   1975
1-24            3.7    3.9    3.7
25-74           2.6    2.0    2.6
75-199          2.8    1.4    3.2
200+            3.9    3.0    3.2
Unreported      0.0    1.4    0.0

SOURCE: Metropolitan Life Insurance Company,
        Stat. Bull.:3 p 4 (1979)

NOTE 1: 03/13/86,NC
NOTE 2: values transformed by sqrt(*)
---------------------------------------------------
Command? totals
Command? sqrt(*)
Command?
```

We can also calculate percentages or other functions of the data, using either the entire value field or specific rows and columns. The program asks whether an analysis should be done by means or by medians. If the response were means, the result would be the table shown in display 7:

```
7. (TRANSFORMED) DEATHS FROM SPORT PARACHUTING,
          by JUMP EXPERIENCE and by YEAR

                        YEAR
Number
of jumps       1973   1974   1975   Effect
1-24           -0.2    0.2   -0.1    1.3
25-74           0.1   -0.3    0.2   -0.1
75-199          0.2   -1.0    0.7   -0.0
200+            0.4   -0.2   -0.2    0.9
Unreported     -0.6    1.0   -0.5   -2.0
  Effect        0.1   -0.1    0.0    2.5

SOURCE: Metropolitan Life Insurance Company,
        Stat. Bull.:3 p 4 (1979)

NOTE 1: 03/13/86,NC
NOTE 2: values transformed by sqrt(*)
NOTE 3: two-way analysis by means
---------------------------------------------------
Command? sqrt(*)
Command? analyze   Means or medians? means
Command?
```

After looking at the means, we might now wish to back up to the original counted data and request an analysis by medians, shown in display 8.

We might then spot a flaw in these data: we should not be comparing across number of jumps without some denominator. If we had information on the number of jumps by various categories in 1980, we could join these (hypothetical) data in the table as shown in display 9.

Note the need in joining data to understand differences in the stub values. In this case, the data joined were not classified in precisely the same way as the data in our existing tables, so some interpolation rule had to be employed. The rule employed in display 9 was one based on the exponential distribution.

```
8.     DEATHS FROM SPORT PARACHUTING,
       by JUMP EXPERIENCE and by YEAR

                       YEAR
Number
of jumps       1973   1974   1975   Effect
1-24             0      2      0       6
25-74            0     -2      0      -1
75-199           0     -5      2       0
200+             5      0      0       2
Unreported       0      3      0      -8
  Effect         0     -1      0       8

SOURCE: Metropolitan Life Insurance Company,
        Stat. Bull.:3 p 4 (1979)

NOTE 1: 03/13/86,NC
NOTE 2: two-way analysis by medians
---------------------------------------------------
Command? back
Command? analyze medians
Command?
```

```
9.     DEATHS FROM SPORT PARACHUTING,
       by JUMP EXPERIENCE and by YEAR

                      YEAR
Number                                       (Hypothet)
of jumps     1973    1974    1975   Effect   1980 jumps
1-24           0       2       0       6        336
25-74          0      -2       0      -1        525
75-199         0      -5       2       0        642
200+           5       0       0       2       2023
Unreported     0       3       0      -8
  Effect       0      -1       0       8

SOURCE: Metropolitan Life Insurance Company,
        Stat. Bull.:3 p 4 (1979); Imaginary
        Sport Parachuting Club (1982)

NOTE 1: 03/13/86,NC
NOTE 2: two-way analysis by medians
NOTE 3: 03/19/86,WHR
---------------------------------------------------
Command? join njumps
Match categories (linear interpolation)? exponent
Command?
```

The Table Manipulator has a cursor that can be moved from field to field. For example, we could move the cursor to row "Unreported" and column "1974", and then revise the original value 2 with the command sequence

```
Command? recall 4
Command? revise 1
```

The first command takes us back to display 4, and the second command changes the value to 1. This new value is carried through to all subsequent analyses. Thus we can now move forward in the displays to track the effect of that revision through the analysis. The effect on display 14, for example, would be as shown. Notice the automatically generated footnote in this display.

```
14. DEATHS FROM SPORT PARACHUTING.
    by JUMP EXPERIENCE and by YEAR

                      YEAR
Number                                     (Hypothet)
of jumps        1973   1974   1975  Effect  1980 jumps
1-24              0      2      0      6       336
25-74             0     -2      0     -1       525
75-199            0     -5      2      0       642
200+              5      0      0      2      2023
Unreported[2] 0        *2*      0     -8
  Effect          0     -1      0      8

SOURCE: Metropolitan Life Insurance Company,
        Stat. Bull.:3 p 4 (1979); Imaginary
        Sport Parachuting Club (1982)

NOTE 1: 03/13/86,NC
NOTE 2: original 1974 value was 2; 03/20/86,WHR
NOTE 3: two-way analysis by medians
NOTE 4: 03/19/86,WHR
-----------------------------------------------
Command? forward
Command? forward
Command?
```

Finally, an illustration of the automatic comparison capabilities. The Table Manipulator determines the amount of fuzz in the table and uses that to suggest a set of cut points, which are subject to acceptance by the analyst.

```
16. DEATHS FROM SPORT PARACHUTING,
    by JUMP EXPERIENCE and by YEAR

                      YEAR
Number
of jumps        1973   1974   1975
1-24             14     15     14
                 <<     <<     <<
25-74             7      4      7

75-199            8  >   2  << 10
                 <<     <<     <<
200+             15      9     10
                 >>     >>
Unreported        0      2      0

SOURCE: Metropolitan Life Insurance Company,
        Stat. Bull.:3 p 4 (1979)

NOTE 1: 03/13/86,NC
NOTE 2: comparisons cut at 3.1 and 6.2
-----------------------------------------------
Command? recall 4
Command? compare     3.1/6.2?
Command?
```

The Table Manipulator also has the ability to move rapidly around the display, save tables on the disk and print them out, calculate simple test statistics such as chi-square, reorder rows or columns on the basis of clustering algorithms or a determination of an optimal linear ordering (non-linear unidimensional scaling), compute subtotals, and do regressions of rows or columns.

To summarize, the Table Manipulator is an experimental tool for secondary data analysis using existing tabular material. It embodies design objectives dictated by the Language of Data and demonstrates some areas in which the design of existing interactive packages might be improved. It is one of many possible tools in a Language of Data computer system and should be viewed in this context.

NOTES:

[1] Address: The Rand Corporation, Santa Monica, CA 90406; Language of Data Project, Box R, Sausalito, CA 94966. This work supported by a grant from the System Development Foundation through San Jose State University.

[2] Dolby JL, Clark N, Rogers WH: A General Theory of Data, Presented at the 1986 Interface Meetings, Fort Collins, CO, March 1986.

[3] Clark N: Tables and Graphs as Language, Presented at the 1986 Interface Meetings, Fort Collins, CO, March 1986.

[4] Franzen A: The Descriptor Manipulator: A Preliminary Reference Manual, (program by W Houchin), LOD Technical Report, Language of Data Project, 1986.

[5] Clark N: Sample Pages and Specifications: Monthly Petroleum Statement, Language of Data Project, 1981, pp. 16-18.

[6] San Jose State University Mathematics Clinic: Pattern Recognition and Automatic Data Translation, Mathematics Clinic, San Jose State University, 1985.

[7] Tukey JW: Exploratory Data Analysis, Addison Wesley, 1977.

[8] US Bureau of Labor Statistics: Table Producing Language, Version 5: Language Guide and Print Control Language, US Department of Labor, Bureau of Labor Statistics, Office of Systems and Standards, Division of General Systems, 1979.

[9] Graves H, Blaine L: ALGOS: A Foundation for the Representation of Knowledge, Algos Project, Department of Mathematics and Computer Science, San Jose State University, 1984.

[10] Manor R, Graves H: ALGOS as a Knowledge Representation System: Representing Statistical Data and Tables, Language of Data Project, 1983.

[11] National Opinion Research Center: General Social Surveys, 1972-1984: Cumulative Codebook, Roper Public Opinion Research Center, University of Connecticut, Storrs, CT, 1984.

THE LANGUAGE OF DATA: A GENERAL THEORY OF DATA

James L. Dolby, San Jose State University; Language of Data Project
Nancy Clark, Editing and Design of Textbooks; Language of Data Project
William H. Rogers, The Rand Corporation; Language of Data Project

This paper, adapted from a recent talk by the late James L. Dolby, was prepared for presentation by Nancy Clark and William H. Rogers. It represents theoretical developments by Dolby and Clark, with a focus on Dolby's work. Because the paper was written in two halves, the first by Clark and the second by Rogers, it reflects a shift in perspective. We felt it appropriate to retain the two perspectives to give the reader some flavor of the cross-disciplinary nature of the Language of Data.

The Language of Data stems from some practical problems that affect more than one discipline. Most of these problems are old. Analysts have to work with data that are so poorly identified that there is often no clue to their ancestry, let alone to the phenomenon they represent. One problem is that this information is not documented--or at least not documented in a form in which it can be passed along.

Another problem is the lack of any mechanism for transmitting information, in unambiguous form, through an entire chain of communication. Although the communication of information is usually treated as a presentation issue, for data the problem is more fundamental. At every point in the communication chain information is transferred from one stage to the next through a visual intermediary, a table or a graph. Thus unambiguous communication depends on a precise definition of both the information to be communicated and the medium of communication. The properties that convey information in tables and graphs are discussed in Clark (1986). This paper focuses on the components of information, the content of the communication.

The general sequence of events for data is shown by the diagram in Figure 1. Each of these stages involves a different use of the data, and the activities at any stage may alter the content, generate new information, or lose information. The biggest loss of information is at the data-collection stage, where there is a wealth of descriptive information, but no criteria that cover the essential elements of description. There is, for example, no formal provision for including a description of the phenomenon that was measured as an integral part of the data. In fact, the reported values are commonly referred to as "the data," a definition that disconnects the measurements from what was measured at the first pass. The accompanying documentation may describe the measurement process in copious detail, but the essential ingredients have not made it into the record.

The Formal Structure of Data

The foundation of the Language of Data is a formalism for incorporating the essential elements of description, based on a faceted classification scheme originally developed in information science. Most classification structures are hierarchic. In biological classifications, for example, all the species are organized into a single hierarchy. In a faceted structure each object or event is identified in terms of a set of descriptors, each representing a different facet of that phenomenon.

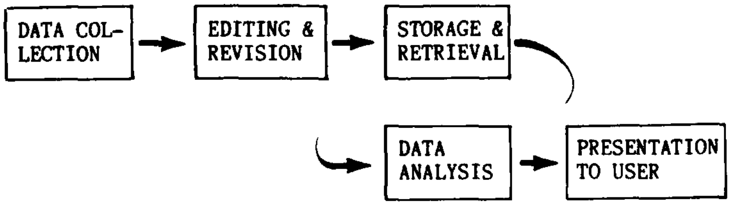
The Dolby model is based on a facet structure originally designed as a universal scheme for library classification, in which the contents of all documents were described in terms of five fundamental dimensions of description--

Observer, Matter, Function, Space, and Time

with the author name added for unique identification. In less formal terms, these categories are the standard Who, What, How, Where, and When of reporting. They are, in fact, the minimum requirements of description for any reported event. The application of this model to scientific data is discussed in Dolby (1983). For data, of course, a further level of specification is needed--the aspect observed and the values of the observation.

The result is a formal definition of a complete datum, shown in Figure 2. Dolby defined a datum as an ordered pair consisting of an observation component and a descriptive component--a finite set of terms that identify the phenomenon the data represent (Dolby and Clark, 1982). Notice that unique identification works in both directions in this model. The set of descriptors uniquely identifies the phenomenon being described. The part

FIGURE 1 The chain of communication for data



Language of Data Project, Box R, Sausalito, CA 94966. Research supported by a grant from the System Development Foundation to San Jose State University.

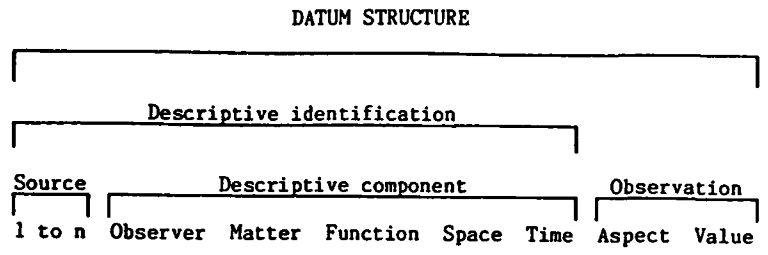


FIGURE 2 The formal structure of data

bracketed as "descriptive identification" refers to unique identification of the values. As a storage system, faceted classification provides a unique address for every number in the database.

Some of the structural details are easier to see if the descriptors are represented as an unorganized set, as shown in Figure 3a. The symbols O, M, F, S, and T refer to the descriptor categories--Observer, Matter, Function, Space, and Time--and X denotes the aspect and its domain of values. The classification scheme sorts the descriptive elements of data into five facets. Each descriptor belongs to some variable in one of these facets, which specifies its meaning in a particular context--the difference, for example, between a camel's-hair brush and a camel's hairbrush. The result is a set of descriptors, each defined in terms of its own underlying variable, but structurally independent of each other. This is an exceedingly useful arrangement for later manipulation of the terms during analysis, an application discussed in Rogers (1986).

From the standpoint of description, however, we still have only a set of descriptors--a collection of words, each with a cleanly defined meaning, but with no relationship to each other. Faceted classification specifies the semantic content of the description, but not the context. To connect the data to the phenomenon they represent, we need a basis of organization for a coherent description of the phenomenon. And to connect the observation to the phenomenon under study, we also need the formal link between aspect and the phenomenon.

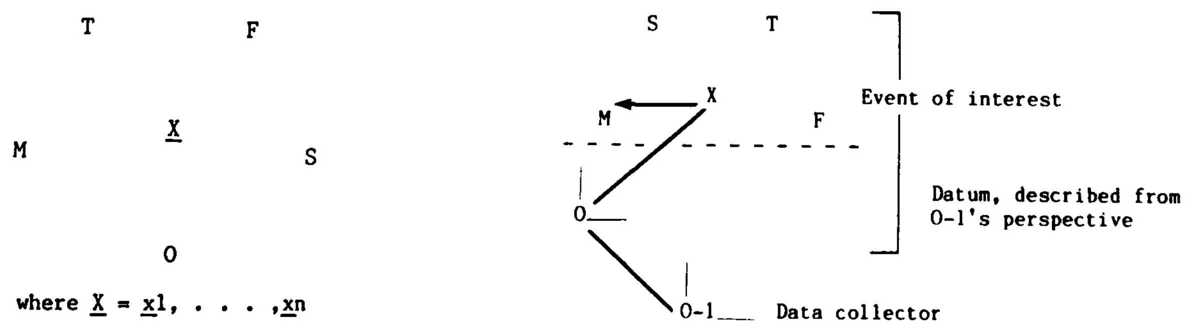
Neither of these structures lies in the classification scheme. However, the missing elements are provided by Clark's development of the role of aspect in the information structure of tables (Clark, 1986). The first link relates to the role of the observer in specifying the aspect observed.

In any description of reality we have to know the perspective from which the objects and events are being viewed in order to tell what we are looking at. In a direct observation the original observer determines the viewing point. Under these circumstances the focus of the observation might be any aspect of the phenomenon, depending on the observer's interest. However, data are collected about a phenomenon of interest to the data collector. In this case the original observer is responding to questions framed by the data collector, in connection with the event about which the data collector is gathering information. Thus the only vantage point represented even in raw data is that of the data collector, the person known as "source." The identity of the original observer, the witness who reported the values, is part of the formal record of the data collector's observation.

This is true even where the data collector and the observer are the same person; the researcher frames the questions in one role and records the observations in another. In subsequent publications of the data the identity of both observers may be available through the source chain. It is the author of the current source document, however, who specifies the aspect of the data being discussed.

Notice that as soon as we get into the structure of description the role of aspect begins to emerge. From a grammatical standpoint, aspect is a "relative term"--one that requires an antecedent. The distinction between "aspect measured" and "aspect discussed," for example, is in effect a transfer of attention from the object that was measured to the object of study, the topic term in

FIGURE 3 Descriptive structure of a datum



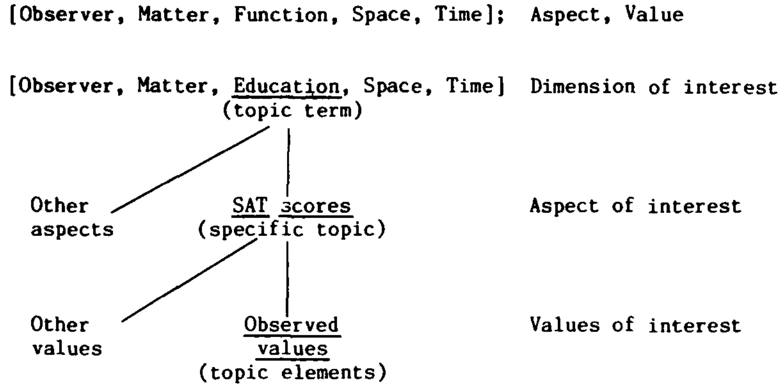


FIGURE 4 The role of aspect in classification

the descriptor set. However, aspect also implies a part/whole relationship--the aspect of the event that is the focus of the observation, and in particular, the one about to be discussed in further detail. Thus, from the standpoint of faceted classification, aspect is merely further specification in the dimension of interest.

The data structure in Figure 2 is shown at the top of Figure 4, represented as an ordered pair. When we start with the phenomenon instead of the values, the observation component shows up below the descriptive component as a direct expansion from the topic term in the descriptor set. In this case the topic of the discussion is education, and the specific topic is the aspect of education described by the data, SAT scores. The values of interest, at the next level, are the recorded values, compared to other possible sets of values. From the bottom up, of course, aspect is also the name of the variable. Thus the values of the variable are part of a fully connected structure, describing the particular aspect of education that was observed.

Aspect also serves as an important structural link between tables and graphs. However, of more immediate interest, its role as part of the topic structure gives us a basis for using descriptor sets to summarize the information structure of the data as well as the content.

The Classification Process

The purpose of an initial classification is to provide the analyst with an accurate representation of the author's meaning, as a starting point for analysis. This means a consistent picture of the author's descriptive structure, not the classifier's, and not the analyst's; that comes later.

Figure 5 shows the information requirements for faceted classification. Although the source chain is technically a part of the observer facet, the two source entries are sorted out here to keep the event structure straight. The source entries refer to the events of publication and data collection, and the kernel set--the next five descriptors, plus aspect and its domain--refer to the event described by the data. The last entry is also a trace back to a prior event.

In most cases the secondary analyst is working with variables that have passed through unknown hands and may be in various degrees of removal from the original source. However, the only descriptive structure we can classify is the one we can see, the current representation by the author of the current source document. The first source entry therefore identifies the author whose description is represented by the classification. The same author descriptor in both source categories identifies the data as primary data. Note,

FIGURE 5 Information requirements for classification
From Clark, Classification Procedures (1984)

Current source:	Author of current document; publication date
Original source:	Data collector; time of data collection
Observer:	Reporter/respondent, as described by author
Matter:	Entities involved in event described
Function:	Nature of event described
Space:	Location of event
Time:	Time of event
Aspect:	Aspect discussed by author [+ topic term]
Domain:	Nature of values
*Aspect:	Aspect described by data collector [+ topic term]

ATTITUDES ON SOCIAL CONTROL

Current source:	NORC; July 1984
Original source:	NORC; spring 1983

Observer:	Adult member of household
Matter:	Adult member of household
Function:	Social control
Space:	U.S./world
Time:	Current
Aspect:	Attitudes [→ Social control (F)]
Domain:	Civil liberties, sexual behavior, women's rights, criminal justice, violence, religion, suicide/euthanasia, economic controls (national spending)
*Aspect:	Same as current aspect

(a) Descriptor set for a main topic

(b) Descriptor set for a variable

CODE NAME: GRASS

Current source:	NORC; July 1984
Original source:	NORC; spring 1983

Observer:	Adult member of household
Matter:	Marijuana
Function:	Legalization
Space:	Continental U.S.
Time:	Current
Aspect:	Opinion [→ Legalization (F)]
Domain:	Should, should not, don't know, no answer, not applicable
*Aspect:	Same as current aspect

FIGURE 6 Descriptor sets for a main topic and a variable in the General Social Survey

however, that the time of publication is not the same thing as the time of data collection. The aspect entry at the bottom, once it is captured in the primary data, specifies the original context in which the data were collected.

Although the formalism is defined at the datum level, the classification scheme extends to the level of a whole database or any level in between. At the highest level the descriptor categories contain what are, in essence, the title elements for the whole database, with the purpose of data collection as the function descriptor and the set of main topics as the domain of values. The next breakout is an expansion on each of the main topics, and the domain at that level lists the subtopics at the level below.

For example, one of the main topics in the General Social Survey is attitudes on social control, shown in Figure 6a. The GSS is an annual survey consisting of about 300 questions, designed as a program of social-indicator research. The database consists entirely of primary data, and the data-collection agency, the National Opinion Research Center, is the author of the codebook. The respondents were adult members of households, the observer descriptor. They were being asked about themselves, so at this level of generality they are also the matter component, and the function descriptor is social control. The variables cover domestic and world events, as shown by the space descriptor, and the time was current, the respondents' attitudes at the time of interview.

The aspect entry consists of two terms--the aspect term and a pointer back to the topic term in the main descriptor set. These two descriptors, plus the arrow, specify the focus of the discussion, the underlying question the data were designed to answer. The topic term is a function, social control; therefore the subsets of this topic are also functions--civil liberties, sexual behavior, and so on.

Figure 6b shows a descriptor set for one of the variables in this subset. At the variable level the domain describes the set of values. At this level, if each variable is visualized as the set of answers to a question, it should be possible to tell from the descriptor set for the variable what the question was and who answered it.

The descriptor sets in Figure 6 contain a dense pack of information, including some that is not obvious. For example, not all the information is in the words. The visual grouping of the categories is part of the information structure, and in a highly condensed description this visual information is essential for rapid comprehension. Once the analyst is familiar with the data, the information at this level is no longer needed--except by the next user. At a later stage the descriptors are converted to a different display form for manipulation. This application is discussed in Rogers (1986).

Descriptor sets can also be used to summarize the contents of a table. The table in Figure 7 contains several function terms--deaths, sport, parachuting, jumps. However, the event being described is deaths from sport parachuting. The matter descriptor is the entities involved in this event, the parachutists. The space and time

FIGURE 7 Descriptive elements in a table

DEATHS FROM SPORT PARACHUTING

	Year		
Number of jumps	1973	1974	1975
1-24	14	15	14
25-74	7	4	7
75-199	8	2	10
200 or more	0	2	0

Source: Velleman and Hoaglin, ABCs of EDA, p. 224. Data from Metropolitan Life Insurance Company, Statist. Bull. 60, no. 3 (1979); reprinted by permission.

descriptors identify the place and time of occurrence. In this case the space might be either U.S. or world; however, unless this information is stated elsewhere in the source document, the appropriate descriptor is "unspecified." The time descriptor is the time period covered by the data. The source note gives both the current source and the original source, but who reported the values? In secondary data the original observer is often a missing element. The values are counts, so the aspect is simply number--number of deaths.

In this data set the anonymous observer does not affect most uses of the data. In medical data the identity of the observer would be critical; it is important to know whether the responses were supplied by the doctor or the patient. The ambiguity in the space facet, however, might easily lead to misinterpretation of the data. In working with existing data, descriptor sets are a useful tool for pinpointing what is and is not known about the data.

Descriptor sets are also a useful tool for determining whether two data sets--or two variables--can in fact be compared. Two data are "simply comparable" if they differ in only one descriptor and the differing descriptors are in the same facet (Dolby and Clark, 1982). In the first row of Figure 7, for example, the only source of variation is time; jump experience is common to both data, and all the other descriptors are constant over the set. By the same token, in the 1973 column, all the data have identical desccriptors except for jump experience, in the function facet. If one were interested in a comparison of deaths from sport parachuting in 1973 with U.S. motorcycle deaths in that year, pairing off the descriptors in each category would identify a possible mismatch in the space facet, and hence a confounded comparison.

Data Relationships

The operations on data are discussed below, but in brief, for every algebra that operates on the numbers, there is a corresponding algebra that operates on the descriptors. The total column in Figure 8, for example, is an aggregation over the time period as well as the number of deaths. The total in the stub raises some questions about table construction. The stubhead is number of jumps, but the subject of the table is "jump experience," and the aggregation is over categories of jump experience, not number of jumps.

Whereas the classification scheme provides the semantic content of the Language of Data, the relationships in a data set constitute its syntactic structure. Both components together produce meaning, the information derived from data. The result in tables is a set of "data sentences" that follow the grammatical rules of ordinary prose (Dolby and Clark, 1982). At the generic level, however, the components of information can be stated as

Data		operational		
elements	+	relationships	=	meaning

The set of possible statements in a table or graph is limited by the allowable operations on data. In the Language of Data, therefore, the meanings are limited to statistical meanings.

DEATHS FROM SPORT PARACHUTING
[by jump experience and year]

	Year			
Number of jumps[?]	1973	1974	1975	Total, 1973-1975
1-24	14	15	14	43
25-74	7	4	7	18
75-199	8	2	10	20
200 or more	0	2	0	34
All experience levels	44	32	41	117

Source: Velleman and Hoaglin, ABCs of EDA, p. 224. Data from Metropolitan Life Insurance Company, Statist. Bull. 60, no. 3 (1979); reprinted by permission.

FIGURE 8 Aggregation of descriptors

The medium of communication for data is a visual representation. The communication of meaning therefore depends on the visible counterparts of these content elements. In tables, as in verbal language, the syntactic relationships of the terms are implicit in the terms being related. Thus in a table the verbal elements carry the burden of communication:

Verbal		implicit		implicit
elements	+	relationships	=	meaning

In graphs this burden falls instead on the relational component:

Visual		visible		visible
elements	+	relationships	=	meaning

This part of the theoretical framework, a general theory of data representation, is discussed in more detail in Clark (1986).

First, however, there are some other pieces to cover. The picture so far has been a bird's-eye view of the communication diagram in Figure 1. The remainder of the discussion is a description of the Language of Data from a more familiar perspective, that of the data user.

The Communication Problem from the Analyst's Perspective

Each person in the chain of communication for data needs to understand what happened earlier in the chain. Analysis, a form of self-communication, is a key part of this sequence. However, each member of the chain has his or her own perspective. If Dolby writes a paper about energy data which you read, you need to know Dolby's frame of reference. You want to know that he started with data published by the Energy Information Administration, the original source, and you want to know that the observers who supplied the values were the energy companies. If you dig deeper, you may also want to know how Dolby got his numbers. How did he do the computations? What other decisions did he make about the data that affected the outcome of his analysis?

We will discuss the chain of communication from three perspectives: the careful reader (who could be anyone), the secondary analyst (in this case, an analyst who starts with the derived data and has weak access to the original source), and the primary analyst (who has primary data and strong access to the original source).

The careful reader will want to know where the numbers were obtained. This always includes "source" information, but a careful reader will also want to know how the numbers were calculated and what the units are. He or she may also require a detailed understanding of the row heads (the stub) and the column heads. Are all the possible row and column categories represented? What kinds of summary statistics were used? What are the omissions?

If the reader is also a data analyst, he will want an understanding of the link between the microdata and the aggregated data in the table and will be curious about what was not said, as well as what was said. What did the author of the data find along the way that influenced his choices? Even better, what hidden assumptions was the author operating under? The analyst may want to revise the methods, or to repeat yesterday's analysis with today's data. He will want to combine his own data with the table to shed more light on trends observed in the table, or he may want to manipulate the table further to reveal a more complex structure in the data.

If a secondary analyst has access to a microdatabase, he will repeatedly ask questions like "What is in this data set?" or "What does this particular variable mean?" We have all spent many long hours poring over someone else's undecipherable codebook.

Both the reader and secondary analyst live at the far end of the diagram in Figure 1 and depend on the primary analyst to supply the right information in a usable form.

The primary analyst has parallel but different concerns. He is often in charge of a very large microdatabase (in some Rand work there may be 10,000 raw variables and 1,000 derived variables). He must be able to track down and eliminate errors caused by programming mistakes or misunderstood instructions.

Part of the problem relates to storage and retrieval. Primary analysts usually work in a constantly changing setting, with data arriving continually and moving through various stages of processing. They must be able to find derived variables created by their colleagues in order avoid reinventing the wheel (or worse, a slightly different wheel). Communication is at various times communication to other audiences, self-communication, or communication with colleagues.

How does the Language of Data theory fit into these problems? We began with a definition of the datum and then we examined this structure in terms of microdatabases and tables. We could think of a microdatabase as a form of table. What are the differences between them? The data in tables are aggregated. The values are usually the downstream product of analysis, the results of statistical manipulations of microdatabases. Cross-tabulation, the operation that produces the contingency table, is a good example of analysis that starts with microdata and produces a table as the end result. Tables may also be manipulated into new tables, a subject discussed further in Rogers (1986).

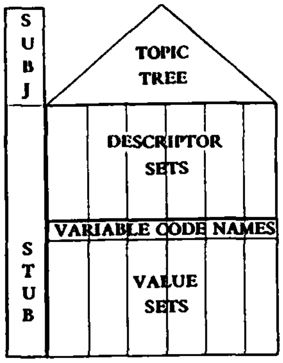


FIGURE 9 Structure of a Microdatabase

The Statistical Microdatabase

At this point, however, let us take a new look at a familiar object, the statistical microdatabase. The top-down classification outlined above for the General Social Survey results in the conceptual structure of a microdatabase shown in Figure 9. The variables are organized into columns, and each number is comparable to each of the other numbers in the same column. Ideally they are simply comparable--that is, the descriptors differ in only one dimension of description. The one descriptive dimension that differs between cases corresponds to the stub.

The following examples are two data that Rand might collect as part of its medical studies:

Source:	Rand
Observer:	Rand Nurse
Matter:	Mary Jones
Function:	Systolic Blood Pressure
Space:	Rand Examination Center
Time:	8 am, January 6, 1986
Aspect:	Pressure, mm Hg (F) Systolic BP
Domain:	120

Source:	Rand
Observer:	Rand Nurse
Matter:	John Smith
Function:	Systolic Blood Pressure
Space:	Rand Examination Center
Time:	8 am, January 6, 1986
Aspect:	Pressure, mm Hg (F) Systolic BP
Domain:	135

These are two simply comparable data about the systolic blood pressure of Mary Jones and John Smith (the matter facet). All the other descriptors are the same, and the two data share a common aspect and unit of measurement. The value of the measurement is placed in the domain.

To represent this, we create a header for each variable which consists of the descriptors common to that column. Any descriptor that corresponds to the stub is elliptical. The stub has a descriptor set which describes the sample. The result is a picture like that in Figure 10.

	STUB	VAR 1	VAR 2
Source:	Rand	Rand	Rand
Obsvr:	Survey Ctr.	Rand Nurse	Patient
Matter:	Patient	Patient	Patient
Functn:	Identity	Systolic Blood Pressure	Satisfaction w MD
Space:	Universal	Rand Examination Center	U.S.
Time:	Universal	8 am, January 6, 1986	Answered Quest.
Aspect:	Name (F) Id.	Pressure, mm Hg (F) Syst BP	Level of Satisf.
Domain:	Alphabetic	120,...,155	1=Very Satisfied,.. 5=Very Unsatisfied
Values:	Mary Jones	120	1
	John Smith	135	4
	...		

FIGURE 10 Structural details of a microdatabase

The individual patients (Mary Jones, John Smith) are listed in the stub and identified by category name as "patient" in the matter slot. A second variable is also shown. Notice that for this variable "patient" is also the observer. To analyze this variable with the first one we must assume that it does not matter that a different observer observed each patient, and that the observations were made at a different time on each patient. These data are not simply comparable. Otherwise, if we think time or observer affect the quality of the observation, we have to include time or observer information in our model, which complicates the analysis.

So far, we have a set of one-way tables with a common stub. A statistical database usually has an added piece of structure represented as the attic in Figure 10. This is the topic structure, which cements the database together. The topical organization captures the relationships between variables intended at the time of data collection. In a secondary analysis, we might use the data for a new purpose, leading to an entirely different organization of the variables, and hence a new topic structure. Curiously, we have found in experiments with the General Social Survey that statisticians tend to invent unique topic structures--unique from each other and different from those produced by the nonstatistical classifiers, who tend to agree. Perhaps this reflects analytic structuring even without numbers.

The topic structure is a hierarchy, and hierarchic structure also shows up in the descriptors themselves. Gender is divided into female and male. Races may be partitioned White, Black, Oriental, American Indian, and Other. Decades are divided into years, then months, then days. A good deal of statistics is devoted to the discovery and identification of classification categories. Problems arise in comparing data from two sources that use different partitions of the same concept: for example, fiscal years and calendar years. We need the equivalent of relational structures to handle this situation.

We also need concepts of balances and forbidden values. For example, the inventory at the beginning of the month, plus additions and subtractions, should give the inventory balance at the end of the month. The whole is the sum of its parts, so proportions should sum to 1. For most data a variable such as gender takes only two values, and if there are more than two something is wrong with the data. In one data set, for example, the reported values for gender were M, F, P, and Q. In the same data set gender also differed depending on the observer. Because existing systems do not embody concepts of domain and observer, this was a major problem to uncover and repair.

How do we formalize the relational structures? One way is through a thesaurus and glossary. The thesaurus holds the possible classification structures and the glossary contains the specialized definitions. It is very important in each application that the user be able to add his or her own thesaurus and glossary information.

Finally, we need a record of interactions with the data. The need for a time-stamped record of the editing and revision process is something that survey data processing centers and fiscal administrators have known all along, but the process seems to be foreign to analysts. Nevertheless, this is the only way we can gain access to assumptions that the analyst may not have been aware of.

Implications for a Computing System

Up to this point we have been talking about a theory of good documentation, ideally as part of the database, not external to it. The classification scheme provides the initial documentation. However, the initial classification is essentially static: it is a description of the given data in the context in which they were collected. This documentation has to reach the analyst, but each analyst is working in a different context, and the analyst's description of the data also has to be documented. Incorporating this documentation in the database as well increases the likelihood that it will be passed along and supplemented by others.

We also want to go deeper than documentation. A Language of Data computing system should be able to manipulate the descriptive component of the data as well as the numbers. To get to dynamics we need to understand what statistical manipulations are like. We have already discussed simply related data--data that have differing descriptors in only one facet. We can go on to the applicability conditions for aggregation:

1 The data must constitute a simply related data set (differ in only one descriptive dimension)

2 The set of items to be aggregated must form an exhaustive partition of the concept to which they are being aggregated.

3 The aspect must be a measure (in the formal sense).

A valid proportion is the ratio of a datum to a valid aggregate of which it is a part, and the domain is further restricted to be nonnegative with a meaningful zero (this collection of restrictions is sometimes called "ratio measure"). If we knew that a certain proportion had passed the applicability conditions, we could be sure it obeyed the value restrictions.

As a more complex example, monotonicity requires two simply related data sets. The descriptor that varies in each of the data sets must be the same descriptor, and the two data sets must match up on this descriptor. In other words, they must form a table with a common stub. For the numbers, we have the usual condition. The second data set (Y) is monotone increasing with respect to the first (X) if X2 > X1 implies Y2 > Y1 for all possible pairs (X1,X2).

What about more complex computations? First, we must recognize that statistical manipulations take place at several levels. At the microdata level we may manipulate one or more variables to produce derived data. The computation may be as simple as changing "No, I did not go to the doctor, if yes, how many visits?" to "zero visits." It may be a standard balance equation: net income equals gross income minus taxes. Population divided by area gives population density, but mean January temperature divided by land area gives nothing. We might be able to see this from the units, or better, from the glossary.

Derivation at the microdata level also may *involve comparing data from different sources:* A patient is adult-onset diabetic if the doctor says he has diabetes and the patient says the symptoms appeared after he was thirty. Or the derivation may involve complex estimation techniques, such as residuals, that depend on the whole data set.

In the transition from microdatabases to tables the manipulations are the standard statistical techniques--mostly confirmatory analysis. Classification structures for tables are derived from values in the microdata or created from the method of analysis (for example, the standard ANOVA table).

Finally, we have manipulations on tables that produce other tables. These manipulations are mostly exploratory (since tables usually contain aggregated data). Dolby has cataloged manipulations *on tables in three groups: select, arrange,* and transform. Here is where classification is most important: does a rearrangement of a table's rows violate the aggregation structure? Is a particular column total a reasonable aggregation? Is a subtraction a valid comparison? It depends on the data as well as the nature of the manipulation.

Summary

The general theory represented by the Language of Data has a number of implications for the commuication of information, a documentation system for databases, storage and retrieval systems, computational aids for analysis, and the construction of survey instruments. More important, it provides a general framework for a systematic approach to the communication of information through data.

REFERENCES

Dolby, James L.: Meaning from Data: Implications for Data Analysis and Database Management Systems, presented at the 149th National Meeting of the American Association for the Advancement of Science, Detroit, 1983.

Dolby, James L., and Nancy Clark: The Language of Data, Language of Data Project, 1982.

Clark, Nancy: Tables and Graphs as Language, presented at the 18th Interface Symposium, Fort Collins, CO, 1986.

Rogers, William H: Implications of the Language of Data for Computing Systems, presented at the *18th Interface Symposium, Fort Collins, CO, 1986.*

INTELLIGENT DATA MANAGEMENT

Henson Graves, San Jose State University
Ruth Manor, San Jose State University and Tel-Aviv University

Abstract

Intelligent computer support for statistical data analysis requires a system in which descriptive information is represented and used deductively to answer questions from data, definitions and assumptions. The knowledge representation requirements for supporting data analysis include flexibility in interacively introducing changes in the system and the capability of handling data revision and data discrepancy. We outline a formalism for representing descriptive information and auxilary assumptions for data analysis. This formalism is currently being developed and implemented in the Algos computational system.

1. Introduction

Data analysis involves a variety of activities whose results are communicated between individuals with very different perspectives. Much of the information that data analysts use will only be available on a computer. Computer systems are used to perform analytic operations on data, and serve as the medium for conveying the results of analysis.

Computer systems used in data analysis such as data base systems and statistical packages do not keep sufficient information to support analysis. The data analyst has to obtain (and remember) what definitions and conventions were used to produce the data. As the chains of new data derived from old data become longer, it becomes even more important for computer information systems to carry the descriptive information necessary for the determination of the meaning of data, and to use this information in drawing conclusions from data and in performing operations on data and tables.

This need for developing more intelligent software system to support data analysis is matched by the existence of Artificial Intelligence (AI) technology which can be used to build such systems. However, current efforts to build expert data analysis systems (Gale, Pregibon, 1983, 1985; Portier, Lai, 1983; Thisted, 1985a, 1985b), as well as discussions of future development of intelligent software in statistics (Hahn, 1985), focus on the expertise of data analysts rather than on representing the objects of analysis. Data analysis is a relatively well defined activity, which makes designing and building a representation system supporting it extremely useful in solving knowledge representation problems.

In this paper we describe the knowledge representation requirements needed to supprt data analysis, and how the system IDA (in development) satisfies them (Graves, Manor, 1985). IDA makes use of the theory of data description developed by the Language of Data Project (Dolby, Clark, 1982), and it is implemented in the knowledge representation system Algos (Graves, Blaine, 1984, 1985).

IDA is not intended as an "expert" system in the sense of knowing what data analysis activities to perform. Rather, its intelligence lies in its expressiveness and in the deductive inference capability used to check applicability conditions and to perform operations on the objects of analysis (data and tables) using, possibly, auxiliary information (e.g., assumptions and definitions).

2. Data Management Requirements

Computer support for data analysis require dealing with interesting knowledge representation problems. One basic problem concerns what are the primitive objects to represent, i.e., what is a datum Dolby, Clark, 1982). Although discussions of data bases and AI address some of the general issues of data modeling, i.e., how to represent, they hardly considert what primitive statistical entities should be represented. Data analysis must deal with the problems of data discrepancies and data revision. These are species of the more general problems of reasoning in the presence of inconsistencies and temporal and context dependency, discussed by logicians by the AI community.

2.1 Statistical Micro Databases and Tables

Too much data with too little information about what they represent is a rapidly growing affliction of the information processing world. Data travel through a chain of communication that proceeds from the first steps of data collection, through the processes of editing, revision and data analysis, to presentation and use -- with the concurrent need to store and retrieve at each interface (Dolby, 1984a). Data get seriously misinterpreted as a result of specific ambiguities regarding who collected the information, which event was measured, where and when the event occurred, what was measured and how it was measured. To correct this situation analysts must have systems that can process descriptive and numerical information together.

Typically, data is gathered by distribution of questionaires and collected into a micro-database. The data in the micro-database is analyzed and presented in a summary form (tables and graphs). Further analysis consists of transformations on these forms. At any step of this sequence, analysis may require back-tracking, revising an earlier step or restarting at an earlier step.

Figure 1: A Micro-database

Distribution of energy to user sectors in the U.S., 1977-80, as reported by the distributers
(Source: Department of Energy)

End-Use Sector	Distributer	Year	Amount in Quad. BTU
Industrial	PG&E	1977	2.14
Industrial	PG&E	1978	2.57
Industrial	PG&E	1979	3.02
Industrial	PG&E	1980	3.91
Industrial	So.Cal	1977	2.54
.			
.			
Res&Comm	PG&E	1977	1.96
.			
Transport.	...		
.			
.			

A micro-database (MDB) is a two-way matrix (e.g., Fig. 1.). Analysis of a MDB leads to a table, which is a typical vehicle for communicating and manipulating data. Statistical tables are basic information units. Their utility lies in their representation of collections of facts about related phenomena, arranged to make simple comparisons readily apparent, as well as that the applicability conditions are met. For example, the MDB presented in Fig. 1 might have served as a basis of the table T1, Fig. 2 (Dolby, 1984c).

Figure 2: Table T1

U.S. CONSUMPTION OF ENERGY
BY END-USE SECTOR and BY YEAR

(Quadrillion BTU's)

End-use sector	1977	1978	1979	1980
Industrial	29.024	29.373	31.551	30.284
Residential & commercial	27.569	28.159	27.462	27.283
Transportation	19.735	20.612	19.950	18.628

Source: Dolby, 1984c

A table can be obtained as an answer to a question, e.g., "What is the energy consumption in the U.S. in 1977-80, by end-use sector and by year?" and it is used to make comparisons among data and to study how comparisons change over time. It can be used to answer questions such as "What is the industral energy consumption in the U.S. in 1977?" by extracting from the table that the industrial consumption of energy in the U.S. in 1977 (in Quadrillion BTU's) is 29.024. The table may also be used to answer the question "What is the total end-use energy consumption in the U.S. in 1977?". However, to compute the answer to this question correctly, requires inference on the basis of additional assumptions.

We want a computer to be able to answer questions like these (formulated, of course, in the appropriate language) correctly. Hence, we need to represent in the system the numeric values (e.g., 29.024) together with their associated descriptive information (i.e., that the industrial consumption of energy in the U.S. in 1977 (in Quadrillion BTU's, as reported by Dolby 1984c) is 29.024. Moreover, for the system to represent the meaning of the data accurately, we need also to represent auxiliary information (definitions, classification, etc.), which may not be explicit in a table display. For instance, the meaning of the table T1 depends on whether the years are defined as calender years starting on January 1, or they are defined as fiscal years starting on April 1.

2.2 Expandability

What is needed to support the interactive nature of data analysis, is that the user be able to add or change assumptions and definitions (including definitions of data operations) whenever he wishes. In designing the system, we do not try to predict, all the possible definitions and assumptions a user may need. An expandable system offers the user the means to "engineer the changes in information".

2.3 Data Discrepancy and Data Revision

A significant activity of data analysts consists of resolving inconsistencies in the data they use. A data analyst is constantly searching for additional information from the data at hand, and discrepancies in the data often give him clues about where to look for information. A typical cause of discrepancy in data is the ambigous use of terms whose meanings chande with contexts. A resolution of the discrepancy consists of identifying the contexts in which the terms should be interpreted together with the interpretation. Whitmore (1984) describes the following example: in 1979 there were headlines about how the Department of Energy and the Bureau of the Census were reporting different amounts of oil imported into the country. The amount reported by the DoE was higher by close to 7% than that reported by Census. After tracking down the sources, Whitmore concluded that "It turns out - for completely legitimate reasons - these two government agencies were using different definitions of some elementary concepts. These were 'oil', 'the United States' and 'month'." Data statements and tables have sources. The source can be viewed as identifying the context in which statements should be interpreted.

Since agencies may occasionally revise the definitions they use, and often revise their data, it is important for the data analyst to be able to find out not just what is the most reliable information, but also the history of the revisions. Hence, an intelligent system should enable users to represent data revisions and trace its evolution.

3. Knowledge Representation in Algos

The basis for an intelligent systems is a computational system which can represent (user specified) theories about some domain. To represent the domain adequately, the theories must have a rich language and deductive capabilities. we employ the "logic" approach, in using the Algos computational system (Graves, Blaine, 1985) which implements a deduction system for a higher order function calculus.

Logical languages have been used as a paradigm for knowledge representation languages for a long time both in AI (McCarthy, Hayes, 1969) and in Database theory (Codd, 1970). The traditional formalization of the Relational Data Base Model represents facts as sentences in a first order language. Question answering involves retrieving the answers from a data base which is viewed as a model (in the logical sense) of the language.

Our problem was to find a logical system which is sufficiently expressive for data analysis. The language must represent entities, relationships, data structures (such as records, reports, and tables), as well as properties about these objects. For analysis, the language must represent algorithms or mathematical functions. For reasoning the language must represent assumptions used in reasoning about the domains (e.g., the assumption that a function with Boolean arguments is an additive measure). Such a language requires quantifying over functions, and is, therefore, higher order. We use a language of a higher order function calculus which has been specifically engineered to represent data structures and algorithms.

The language of Algos is based on topos theory (Goldblatt, 1984) which is expressively comparable to set theory and has been suggested as a foundation for mathemetics (Lawvere, 1976). The difference between these theories lies in their choice of primitives. Set theory is built on the single primitive membership relation. Topos theory is built on the primitive notions of function (map) and type.

3.1 Primitives and Definitions

The syntax of Algos uses elements of mathematical and programming language notation. Algos has a data language of terms which are used to represent the various kinds of data considered here: numeric and string data, descriptive data, algorithms, and assumptions. It has commands for introducing definitions and assumptions, and for making inquiries. A collection of definitions and assumptions constitutes a knowledge base (theory) of the system. The Algos system uses deductive inference to answer questions on the basis of the definitions and assumptions in the knowledge base. The model of computation used is term evaluation (simplification) which is a special case of deduction. Simplification uses deductive inference rules in the form of term reduction rules which correspond to the different kinds of term constructions (e.g., tuple, functional abstraction, etc.). For example, the command

```
      simplify 2+2;
```

evaluates to '4'. A formula is a term which evaluates to 'true' or 'false'. Thus, 'a > 0' is a formula. We can declare a map, p, to be a formula with the statement 'p : OMEGA'. If 'a' is undefined, the command

```
      simplify a > 0;
```

cannot be simplified and returns 'a > 0'. Users can add names and definitions. For example, we can enter definitions by

```
      density  =df  count/area
      months  =df  (jan feb march april may june
                 july aug sep oct nov dec)
```

The names used in these definitions, 'count', 'area', 'jan', 'feb', etc., need not have been previously defined for the definitions of 'density' and 'months' to be legitimate.

3.2 Some Map and Type Constructions

Algos has a product type construction and a corresponding tuple map construction. Products serve as record types and tuples serve as records. For example, an employee record type may be introduced with the declaration 'EMPLOYEE = product(NAME,AGE,SALARY)' and a map of record type with the declaration 'a : EMPLOYEE'. We use a tuple notation to specify a record

```
      a  =df  <John,34,3232.22>
```

A function is a term which can be "applied" to an argument. We use a "lambda" syntax to specify function definitions. For example, an algorithmic definition of the absolute value function can be expressed by:

```
     absolutevalue  =df (lam x)(if x >= 0 then x
                                  else -x)
```

This function can be applied to numeric arguments. The command to simplify 'absolutevalue[-3]' results in the system using the definition to return '3'. The type of the function is the expenential type, I|I, where I is the integer type.

In addition, Algos has lists, numbers and strings. We use the LISP notation for lists, e.g., the list of the first three natural numbers is represented as '(0 1 2)'.

Algos has the empty list, 'nil', and the list operations (as in LISP). The definition

```
   in  =df  (lam a,l)(if empty[l] then false
            else (if a = first[l] then true
            else in[a,rest[l]]))
```

tests if an element is a member of a list, so

```
      simplify in[3,(1 2 3)];
```

evaluates to 'true'.

Algos has a power type construction which is used to represent relations. We use 'POW(X)' to represent the power type of a type X. Relations on a type correspond to subtypes. Further, a relation corresponds to a formula. For example, we can introduce a function to represent 'is a male' with the statement 'male : HUMAN -> OMEGA'. The formula 'male' determines a relation '{male}' which has the value type POW{HUMAN}. This relation represents the data elements of HUMAN having the property of being males.

Many data operations involve aggregating the values of measurements of parts of some domain. We use relations to represent the "containment" and partition relationships. For each type there are a zero (empty) and unit relations, and the Boolean operations '+', '*' and '~' (union, intersection and complement, respectively). E.g., if 'a' is a relation, then the command to simplify 'a+0' returns 'a'. The definition

```
      sum  =df  (lam l)(if empty[l] then 0
                  else first[l]+sum[rest[l]])
```

provides a function to total lists of relations.

In order to represent validity conditions for totaling lists we need to represent partitions. Informally, a list (al ... an) partitions a if both 'a = sum[(al...an)]' and 'for any b,c in (al ... an) not(b = 0) and (if not b=c then b*c = 0)' evaluate to 'true'. Partition is defined as the test:

```
      partition  =df (lam e,l)(e = sum[l] and
         (any b)(if in[b,l] then not(b = 0)) and
     any b,c)(if in[a,l] and in[b,l] and not(b = 0)
                          then a*b = 0)
```

An additive measure is a function, whose arguments are additive over partitions, defined:

```
   measure  =df (lam m)(lam e,l)(partition[e,l]
                 implies  m[e] = sum[map[m,l]]).
```

3.3 Assumptions

We distinguish between a formula as a data object, and assuming it. Having the formula as an assumption means that we can use it in deduction. For example, entering the commands

```
      assume  a > 0;
      simplify absolutevalue[a];
```

returns 'a', because the evaluation of 'ablsolutevalue[a]' utilizes the assumption that a > 0. The user may want to see the assumptions which were used in the simplification. To obtain this the simplify command followed by ?' will result in displaying the assumptions used.

```
      simplify absolutevalue[a]?
```

returns 'a; depending on a > 0'.

In data analysis one often has only partial information about a function, such as knowing its values for specific arguments without having an algorithmic definition. We represent such information with commands of the form

```
      assume  f[a]  -  k;
```

The equality symbol '-' in this context represents the simplification relation. The command to simplify 'f[a]' results in 'k'.

The statement "the kinds of coal are anthracite, lignite, and bitumunous" is represented as an assumption. We represent 'kind' as a function which when applied to 'coal' yields a list. Namely,

```
      assume kind[coal] - (anth lign bitum);
```

Note that the terms in this formula such as 'kind' and 'coal', may not be defined. The question "what kinds of coal are there?" is represented as the simplification request to simplify 'kind[coal]' which simplifies as the list '(anth lign bitum)'.

For data analysis it will be important for us to total numerical lists on the basis of the assumption that the list represents the values of an additive measure over a partition. For example, on the basis of:

```
      assume  measure[m];
      assume  partition[a,(a1 a2 a3)];
      assume  m[(a1 a2 a3)]  - (1 4 2);
```

the command to simplify 'm[a]' returns '7', as a result of the inference:

```
      m[a]  -  sum[map[m,(a1 a2 a3)]]
            -  sum[1 4 2]
```

3.4 Context Dependency

Knowledge representation for data analysis is dynamic. Objects get added and their relationships change during the span of existence of the system. In order to represent reasoning about the dynamic aspect of knowledge acquisition, we need to represent the context dependency of information. Context dependency includes temporal and source relations. To require that all contextual dependencies be explicitly stated by the user, or that all assumptions used in a deduction be of the same sources, is too strong. A user may not be aware what these are, and he may sometimes want to use information from different sources anyway.

By representing context dependency we mean that the user may choose to specify contextual dependencies explicitly, or he may choose to omit it. In this case the system will record a contextual reference, which may be recovered at will. For instance, the system can record the time the definition or assumption was entered, the terminal used, etc. Similarly, in deductive inference, one may choose to ignore the fact that assumptions used have different contextual dependencies (e.g., we may want to ignore the fact that definitions were given by different sources, or entered in different times), or we may want to specify that only assumptions of a specified source should be used. A typical case of discrepancy arises when we ignore contextual dependencies, derive a contradictory conclusion and the we try to resolve it by restricting the context of the assumptions.

In Algos commands which modify the knowlege base index the new information by their "origin" which is the time and source of the statement. For example, consider the commands

```
      a  -df  5;
      a  -df  6;
```

and let t1 and t2 be the times in which these commands are executed. We view 'a' as one entity with two values at the two times. Thus,

```
      simplify a/t1;
```

evaluates to '5', while

```
      simplify a/t2;
```

evaluates to '6'. Similarly, we may specify the source of a definition. E.g., the commands

```
      a  -df 3/DOE;
      a  -df  4/Census;
```

specify values for a for each of the two sources, DOE and Census.

4. Satisfying the requirements in Algos

Statistical tables are viewed as linguistic entities. The language of data is, however, a very restricted linguistic context in that data statements, in general, represent the values of a measurements of some observed "chunk of reality". The enterprise attempted here is to represent these notions within a formal deduction system. We answer questions using information expressed in data, and in assumptions and definitions. However, as with any language, sentences may mean different things to different individuals. Successful communication depends on consensus of meaning regarding the descriptive terms of the data sentences. LOD approach to this problem is to choose classes of descriptors and index their meanings by context. This theory represents statistical data in terms of an explicit set of descriptors. This appears to be adequate for representing a large class of data.

4.1 Statistical Data

Following Dolby, the basic communicated entity is a datum, which is a statement concerning the value of a measurement. The measurement is associated with an observed "event", and the descriptive information in the sentence identifies the event, the object measured, how it was measured and the result of the measurement. Dolby has suggested that the descriptive part of statistical data statements has the following components: Aspect (of measurement), Object (of measurement), Unit (of measurement), Event observed, Observer, Matter, Activity, Space, Time, and Source.

Information lies in the comparison of related data, and not in singular data (Dolby, Clark, 1982). One of the problems we face, therefore, is to characterize ways in which data are related, in order to validate data comparisons. Analysis of examples yields that it is the descriptions and, specifically, the event descriptors, which serve to check data comparability, and we need to indicate how.

Consider for instance the descriptions: "the amount of total end-user consumption in the U.S. in 1980 in quadrillion BTU's (as reported by MER)", and "the amount of industrial consumption in the U.S. in 1980 in quadrillion BTU's (as reported by CT1)".We need to represent these descriptions in such a way that their comparability can be verified. The question of characterizing the "relatedness" of data can be reduced to questions regarding relations between the corresponding components of the descriptors. For example, table T1 (Fig. 2) displays data which share all their descriptive components except those in the observer and time slots. Similarly, the comparability of the data quoted above follows from the fact that they differ only in their users. However, as we have noted, different sources often use the same terms in different ways. Hence, even in this simple case, elementary parsing is insufficient and we need also to check that the terms used in these descriptors, e.g., 'consumption', do indeed carry the same meaning.

Following Dolby (1984b), we represent the datum as the assumption that the value of measuring the amount of energy associated with the observed event is 76.201. Amount (aspect of measurement) is represented as a function whose arguments are: energy (object measured), q-btu (unit of measure), and the event observed. We can represent the above datum as the following assumption which may be entered into the Algos system with

```
      assume  amount[energy,q-BTU,
         <all-end-users, energy, consumption,
                    U.S., 1980,MER>] = 76.201;
```

The representation of MDB exploits our basic representation of a statistical datum. Namely, the MDB presented in Fig. 1 is viewed as a collection of statistical data, and is represented as a function in the argument

```
      <energy,q-btu,
            <<distributer,sector>,
         energy,distribution,,U.S,year,DOE>>.
```

The MDB is represented as an assumption about the function 'amount'. Thus

```
      assume  amount[energy,q-btu,
        <<distributer[energy],sector[end-user]>,
         energy,distribution,
                    U.S.,year[1977-1980],DOE>]

             = (2.14, 2.57, ....);
```

Tables are viewed as displays of (comparable) data, and are represented in terms of lists of components. The information in the list is sufficient to support operation on data. Table T1 (Fig. 2) describing the "U.S. consumption of energy in 1977-80 by end-use sector and year" is represented as the list

```
   (<end-user,energy,consumption,US,1977-80,MER>
       <amount,energy,q-BTU>
         end-user
         (industrial res&com transportation)
         1977-80
         (1977 1978 1979 1980)
         (29.024 29.373 .......
         ...... ))
```

4.2 Dependency, Inconsistency and Revision

An Algos theory is a collection of definitions and assumptions from a knowledge base. Users are free to enter new assumptions into the theory. The Algos language is sufficiently rich to express the context dependencies of these assumptions. This has been demonstrated for single dependencies (Graves, Manor, 1985). Our future research will concentrate on inconsistencies and revision by distinguishing and representing multiple theories along the following lines.

Context dependency can be viewed as a means for identifying a (sub)theory, in that all entities and statements restricted to a context determine a theory. Of course, since there is practically no restriction on contexts, there are no restrictions on what sets of statements could make up a theory. In the Algos system inconsistency can only arise as the result of assumptions introduced by a user. When a contradiction is detected, the computation steps can be examined to locate what assumptions were used, and the contradiction may then be resolved by introducing further contextual distinctions in the contradictory hypotheses. Thus, we can consider the contradictory hypotheses as creating alternative theories. Moreover, we can consider a theory identified by some context (e.g., all assumptions associated with the DOE source) and trace its evolution through time.

A data analysis system will use a number of primary external data sources such as the DOE and Census. These sources use relatively stable definitions of their terms and stable assumptions about them. However, they may change over time. The terms occuring in a table whose origin is some beaureaucratic entity such as DOE are indexed by that source. Each source entity such as DOE has a collection of definitions and assumptions indexed by it.

The discrepancies discussed in Section 2 can be resolved by adding the following assumptions in which contextual dependency is explicit.

```
      assume  sum[   ] =  ..../CT1;
      assume  (state or district)[US]  =
                     (alabama .....) /census;
      assume  day[month]  =
              (15[month]...14[month+1])/census;
      assume  day[month]  =
              (first[month] ....last[month])/DOE;
```

The strategy for resolving inconsistencies the data is based on the principle that any non-identical entities should have different names. We attempt to resolve inconsistencies by disambiguating the descriptions. Data are usually aggregations over partitions. A first step is to look in the conflicting data, for the partition used, and to identify the terms used *to describe the aggregation and its components*. Typically, a discrepancy arises by using different partitions of the same aggregation. In these cases we distinguish them by reference to the context as above.

In Algos there are no restrictions on what could serve as a context dependency, although in data analysis the dependencies of interest are relative to time and source. Given appropriate temporal assumptions, we can consider a theory identified by some source, e.g., all assumptions associated with the DOE) and trace their evolution, compare conclusions based on different assumptions, and decide on their relative reliability, without having to delete any information from the knowledge base.

Bibliography

AAAI, Non-Monotonic Reasoning Workshop; 1984.

Bobrow, Daniel G., ed. Artificial Intelligence, 1980; 13(1,2).

Codd, E. F. A Relational Model for Large Shared Data Banks. Communications of the ACM; 1970; 13:377-387.

Dolby, James L.; Clark, Nancy. The Language of Data. The Language of Data Project; 1982.

Dolby, James L. Put the Information in the Database, Not the Program. Database Management Conference; 1984.

Dolby, James L. The Role of Description in the Evaluation of Data Quality. 9th International Codata Conference; 1984

Dolby, James L. A Theory of Data:Implications for Information Retrieval. Proceedings, Conference on Data Engineering;1984, 111-117.

Gale, William H.; Pregibon Daryl. Using Expert Systems for Developing Statistical Strategy. Annual Meetings of the ASA; Toronto; 1983

Gale, William H.; Pregibon Daryl. Artificial Intelligence Research in Statistics. The AI Magazine; 1985: 72-75

Gallaire, Herve; Minker, Jack. Logic and Data Bases. Plenum Press; 1978: 3-30.

Goldblatt, Robert. Topoi: The Categorical Analysis of Logic. North-Holland; 1984.

Graves, Henson; Blaine, Lee. Algos: A Foundation for Representing Knowledge. Algos 1984.

Graves, Henson; Blaine Lee. Algos: A Knowledge Representation System. Algos Project; 1985.

Graves, Henson; Manor Ruth. Intelligent Data Analysis. Algos Project; 1985 (In progress)

Hahn, Gerald, J. More Intelligent Statistical Software and Statistical Expert Systems: Future Directions", American Statistician; 1985 February; 39: 1-8, 15-16

Lawvere, F. William. Variable Quantities and Variable Structures in Topoi. In: Heller, A. Tierney, M., eds. Algebra, Topology, and Category Theory. Academic Press; 1976: 101-131.

Levesque, Hector J. The Interaction with Incomplete Knowledge Bases.IJCAI;1981 240-245.

Manor, Ruth; Graves, Henson. The Language of Data in Algos; 1984.

Portier, Kenneth, M., Lai, Pan-Yu. A Statistical Expert System for Analysis Determination. Statistical Computing, ASA 1983: 309-311

Reiter, Raymond. Deductive Question-Answering on Relational Data Bases. Gallaire, Minker, 1978:149-177.

Thisted, Ronald A. Knowledge Representation for Expert Data Analysis Systems. University of Chicago, 1985, Report No. 172.

Tukey, John W. Exploratory Data Analysis. Addison-Wesley; 1977.

Whitmore, Thomas. Problems of Data Gathering, Online '84 Conference; 1984.

KNOWLEDGE BASED SYSTEMS

Organizer: *Roy E. Welsch, Massachusetts Institute of Technology*

A Front-End for GLIM

John A. Nelder, D.E. Wolstenholme, Imperial College of Science and Technology, U.K.

Artificial Intelligence Techniques for Retrospective Help in Data Analysis

William H. Nugent, Harvard University

A Front-End for GLIM

J.A. Nelder and D. Wolstenholme, Imperial College.

1. INTRODUCTION

GLIM is a package built round an algorithm for fitting generalised linear models (GLMs) (McCullagh and Nelder, 1983). It is currently distributed to more than 900 sites in 50 countries. It has in addition facilities for data handling, tabulation, scatter plots and histograms, and an interpretive language with control structures for branching and looping. For a full description see Payne et al (1986).

Like most current statistical packages GLIM assumes that the user knows how to do an analysis, and provides him with easily-used tools for doing it. This paper describes a knowledge-based front end (KBFE) for GLIM, currently under construction, which embodies statistical expertise to aid the user in the choice of models for his analysis. The front-end will not be a black box, delivering the 'correct' analysis to the user, and requiring from him little more than a description of his d a. Our aims for the KBFE are three-fold:

(i) to give good advice to the user on the analysis

(ii) To do so in a way that encourages the user to do better next time

(iii)To satisfy the requirements of a range of user skills.

The front-end will be one with fixed rules, i.e. it will not be intelligent in the sense of Student (Gale, 1985), which learns from its experience of users and modifies its rules accordingly. It will be an expert system in the sense that it contains rules which encapsulate expertise.

2. TOOLS

The front-end is being written in Prolog, a declarative language for logic programming. GLIM is the algorithmic engine for the system, and the Prolog being used has its own front-end in the form of APES (see Section 5.1). Communication between the front-end and GLIM is controlled by Unix, which is the operating system for the development.

2.1 GLIM

We use GLIM 3.77, which is written in restricted Fortran 77. The interpretive language has statements in free format each beginning with a directive name of the form $letters, e.g. $FIT, $SORT etc. These are followed by none or more arguments, usually scalars or vectors; however $CALCULATE allows expressions with vectors and scalars as operands and the $FIT directive uses a model formula for the linear component of the model (Wilkinson and Rogers, 1973). A set of statements may be named as a macro, and the directives $LOOP and $SWITCH with macros as arguments allows looping and branching. Subfiles may contain any mixture of data, statements to be executed and macros. The program may be dumped at any stage, and its current state restored later.

2.2 Sigma-PROLOG

PROLOG is the most widely-used logic programming language (Hogger, 1984). Its basis is a subset of first-order predicate logic, with certain extensions. Using PROLOG, the programmer may describe the logical structure of a problem directly instead of being forced, as with conventional procedural languages, to describe in detail the steps the computer must take to solve the problem. This makes PROLOG a good tool for expressing knowledge, since the knowledge can stand alone, uncluttered by computer control instructions.

A typical logic, or PROLOG, program describing knowledge about why a car might fail to start is:

```
fails-to-start (_any car) if
  has-flat-battery (_any-car)

fails-to-start (_any-car) if
  not has-petrol (_any-car)

has-petrol (my-car)

has-flat-battery (my-car)
```

where an underscore symbol at the beginning of a word, as in _any-car, indicates that the word is a variable.

The PROLOG interpreter can use this program to solve certain problems, such as

```
fails-to-start (my-car)?
```

using the necessary inference rules. PROLOG may also be seen as a high-level procedural programming language, since it employs an inference mechanism known as 'resolution', whereby the first rule shown may be seen as having the procedural reading: to show that a car fails-to-start then show that the car has-flat-battery.

LPA sigma-PROLOG is a dialect of PROLOG, written in C, which is suitable for use on machines that support UNIX, such as the VAX 11/750 used in this project. It is a low-level implementation, with a LISP-like syntax, suitable for systems development. It may have front ends, such as APES, added to it, to provide additional features or alternative syntaxes. Thus the first rule given above, in standard sigma-PROLOG syntax is

```
((fails-to-start _any-car)
  (has-flat-battery _any-car))
```

Sigma-PROLOG's most useful features for the development of the GLIM KBFE are modules and the FORK primitive which permits the spawning of child processes in UNIX, e.g. FORTRAN programs such as GLIM. Communication with these processes is carried out simply using UNIX pipes as shown in Figure 1.

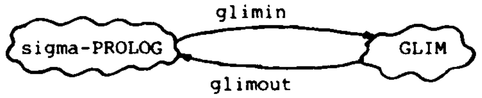


Figure 1

3. GENERAL PROPERTIES OF THE FRONT-END

There are two general characteristics of the front-end which deserve discussion

3.1 Transparency

An important property of Unix is its transparency, by which we mean that all the tools available to the operating system are also available to the user. Compare this with the older-style operating systems where, for instance, tools for creating and searching directories would certainly have been created by the originators but would not be accessible by the user for his own purposes. We think that expert systems should be similarly transparent, so that if the system uses certain procedures to obtain information on which to base advice, the user should have access to the same procedures for his own activities. We believe that transparency will aid the transfer of expertise and encourage the user to learn from that expertise; at the same time we recognize that the user who does not wish to think for himself, but rather hankers after a black box, may not like the system. We do not aim to cater for this class of user.

3.2 Libertarianism

An authoritarian system is one that controls the sequence of operations for a user, in the sense that if a certain state is reached, then other future, and hitherto possible, options may now be barred. Thus it might be that if in a simple linear regression a quadratic term is found to be significant then future actions involving the use of the linear fit only would be barred. An authoritarian system is one in which the system knows best - always. The alternative, which we favour, allows that the user may have background knowledge that the system does not know about and cannot easily discover. There is a pragmatic argument for libertarian systems in statistics, quite apart from more philosophical ones. Thi: is that the rules for such systems are themselves abstractions from whatever fields of application the originators know about, and that particular knowledge of those fields cannot be part of the rule system; thus the user will always bring background knowledge to his analysis which must be given full opportunity for expression.

4. LARGE-SCALE STRUCTURE

Prolog per se is too low-level a language to use in an unstructured way for constructing a KBFE. Thus higher-level structures must be developed for expressing the large-scale structure. In addition a general facility is needed for time-stamping information. In Prolog an assertion once made stays true; such a property is not suitable for a system involving trial-and-error learning. The method of time-stamping information is described in Section 4. Another feature of Prolog that needs attention is the assumption of negation as failure; this says that a fact that cannot be established to be true is taken as false, i.e.

not established to be true = established not to be true

This closed-world assumption must also be modified by use of a predicate defining 'established' if trial-and-error learning is to be correctly modelled.

4.1 Nodes

The analysis process has been split into a set of activities, each defining a node. These, with their two letter abbreviations and brief indication of scope, are:

SE set-up user environment

DI data input
- get data, with names of variables, for analysis.

DD data definition
- user provides information about variables (e.g. whether continuous or counts), the data structure (i.e. whether experimental or observational), etc., designed to establish whether GLIM is suitable and to give guidance about forms of analysis likely to be appropriate.

DV data validation
- detection of gross errors and inconsistencies.

DE data exploration
- mainly graphical techniques, to determine possible transformations and initial settings of link and variance functions.

MS model selection
- procedure for selecting one or more parsimonious models for the data.

MD model display
- display of statistics, e.g. fitted values and residuals, associated with models selected.

MC model checking
- checking the adequacy of models selected using various techniques.

MP model prediction
- summarising results of models found, including calculation of summary statistics and measures of uncertainty.

The nodes can be thought of as the nodes of a graph and the strategy of an analysis may be summarized by the particular path taken through the graph, together with variables defining the state at each node. The path will reflect the user's choice of methods, previous analyses and prior knowledge of the data set.

4.2 Tasks

A new command language is being developed to allow various tasks to be invoked by high-level commands. These tasks may be broadly divided into two categories:

(i) those providing general facilities such as
- access to the operating system,
- direct access to GLIM
- background information
- changing activity
- changing mode of use; e.g. from giving tasks directly to obtaining advice
- quitting the system;

(ii) those concerned with the details of statistical analysis using the front-end, such as
- inputting data
- creating analytical trees
- finding basic statistics
- plotting graphs.

The syntax for invoking both categories of task is the same. Each task is invoked by a keyword, e.g. 'find', 'create', etc., possibly followed by a sequence of keywords and variables. The syntax of each task is designed to facilitate both checking and prompting, since a given keyword uniquely determines the sequence of variables following it, together with the set of keywords from which the next keyword can come.

When the user specifies a task to be done, a four-level checking process is first undertaken.

(i) Pattern matching - the first syntax check - whereby the task specified is matched against the syntax of possible

tasks. Failure to match results in prompts which tell the user how much of the task specified, working from left to right, matches a possible task, and how this might be completed.
For example, the three tasks available for entering data from the keyboard take the forms:

```
   keydata  vector _name _values
   keydata  rows   _names _values
   keydata  columns _names _values.
```

If the user specified task

```
   keydata (A B C) (1 2 3 4 5 5 6 8 7)
```

i.e. with the second word missing, no pattern match would be found, and the following message would appear.

```
-----------------------------------------------
  the first part of the task given
                    keydata
  may be completed as follows
      vector  <name of vector>
              <bracketed list of values of
                 vector>
      rows    <bracketed list of vectors>
              <bracketed list of values of
                 given vectors>
      columns <bracketed list of vectors>
              <bracketed values of given
                 vectors>.
-----------------------------------------------
```

(ii) Basic type checking - the second syntax check. Following a successful pattern match, the type of each variable, if any, is checked.
For example, if the user specified the task

```
  keydata rows (A B C) (1 2 3 4 5 5 6 8 7)
```

the list (A B C) would be checked to ensure that each item on the list was a valid vector name, and the second list to ensure that each item was a number.

Failure of any type check results in failure of the task; the user could then explore the reason for failure, using the explanation facilities of APES, if (s)he wished.

(iii) Context-free checks - the first check on semantics. Checks to ensure that the task is feasible in some context.
For example, if the user specified the task

```
   keydata rows (A B C) (1 2 3 4 5 6 8 7)
```

checks would fail since the number of values is not a multiple of the number of vectors named. As with type-checking, failure would be open to explanation and exploration.

(iv) Context-sensitive checks - the final semantic check. Checks to ensure that the task is feasible or acceptable in the current state of the analysis. For example, if entering data, the names of vectors being defined should not already have been used. Full explanation of failure would again be available.

The actions to be carried out for a particular task may involve sending commands to GLIM, printing output to the screen, asserting facts to the database, accessing files, or some combination of these. Although the checks reduce the likelihood of error, these may occur(e.g. division by zero) when GLIM executes commands. Therefore, the GLIM output from any command that might cause error is checked for error messages. Since errors can leave the state of GLIM in an unknown, intermediate state, detection of an error leads to restoration of the previous state.

4.3 Time stamping

Both GLIM and the user are sources of information for the front-end. During an analysis, however, the state of the GLIM environment changes. Similarly, the user's mind may also be considered to change 'state' as knowledge is acquired and actions are carried out. Information found from either of these sources cannot therefore be assumed to remain valid throughout a session.

A simple solution is to assume that an answer holds true only when asked: this solution would, however, be unacceptable since the user might then be asked the same question many times over. Instead, all information found from the user or GLIM is time-stamped so that the interpreter can later 'decide' whether or not such information still holds true following certain actions. Only if its validity is doubtful will the user or GLIM be re-queried.

5. THE USER INTERFACE

The lowest level of the user interface is provided by APES, the Prolog front-end.

5.1 APES

APES, Augmented Prolog for Expert Systems, was developed by Hammond and Sergot (Hammond, 1982) to provide a logic-programming environment suited to the creation and development of knowledge-based systems and other logic-programming software. Many modifications have been made to it for this project.
The main features of APES useful for KBFE development are:

(i) Declarative dialogue handling

In a KBFE, some facts must of necessity be obtained from the user. Therefore, the system must query the user for the relevant data when needed. APES handles this interaction with the user declaratively (Sergot, 1982).
The main concept of query-the-user is that the program available to the interpreter can be seen as a combination of rules and facts within the computer and the extra information in the user's mind. If the query to be solved concerns a relationship not defined in the computer, it is assumed that the user can supply the necessary information. The interpreter obtains this information by printing a question and accepting an answer. For example, if the query to be solved is

```
            has-age  (Fred _years)
```

where "has-age" is not defined in the program, APES evaluation results in the interaction

```
  which (_years : has-age (Fred _years))?
  Answer is 17
```

where 17 is the user's response.
This approach should be contrasted with the more usual procedural one, in which "has-age" might be defined by a rule such as:

```
        has-age (_person _years) if
                print (How old is _person?) and
                read (_years)
```

Such an approach is rejected in APES because the rule has no acceptable logical reading, i.e. it is not true that a person's age is the logical conclusion of printing and reading.

(ii) Explanation facilities

the user may ask:

why a question is being asked;
how a solution or answer was reached;
why a solution was **not** reached.

(iii) Natural language templates

The explanations given by APES are in terms of the rules used. To improve their readability the programmer may specify natural-language equivalents of the rules and conditions used in the explanations.

e.g. instead of

fails-to-start (_any-car) if
has-flat-battery (_any-car)

the explanation might state

_any-car will fail to start if
it has a flat battery.

Similar natural-language templates may be used to improve the wording of questions.

(iv) Validity constraints

In order to ensure that the answers given by the user are sensible, the programmer may specify validity constraints on any answers given. Thus, if asking Fred's age, a simple check would ensure that the answer is a non-negative integer. Failure is open to explanation.

(v) Advice

The user may be asked either for hard, objective facts or for opinions.
In the latter case, it might help the user if the system could offer its own suggestions, which the user could then accept or reject. To provide such advice the programmer may specify secondary, "consultative" relations which should be used to provide such advice. This advice is only offered if requested, and even then can be rejected, so the user remains firmly in control.

(vi) Visual prompts

During a dialogue, the user may be asked a question of a graphical nature, e.g. "is Y linear against X?". To help him answer this question, the user may demand to see a "visual prompt", e.g. a plot of Y against X. These prompts are given only if asked for.

5.2 User questions at the statistical level

On top of APES' query-the-user we have allowed for higher-level queries by the user on statistical aspects of the front-end. These are of three types.

(i) Definition of terms

The user may not understand the meaning of a question because a word, e.g. 'aliasing', is unfamiliar. A system may need to supply such definitions (the lexicon of Gale and Pregibon, (1982)) to help the user, particularly if (s)he is fairly inexpert in statistics. Clearly a lexicon, if provided in great detail, would amount to an on-line statistical text-book; we do not plan to provide such detail, but to restrict ourselves to possibly less familiar words.

(ii) Explanation of questions

More generally the user may want to reply to a question with 'why are you asking me this question?', i.e. (s)he may want some background or information on why a particular line is being pursued. As with definition, this could lead to the writing of very large amounts of text if pursued in great detail. Again we plan limited amounts of information, which will provide a background to the strategies embodied in the rules.

(iii) Advice on strategy

The user's question is here 'what would you do?', and in the answers are embodied the expertise of the system. The advice is given in terms of the primitive tasks available at the current node, including of course the general tasks of moving to another node etc. In the final section we outline the advice available at the important model-selection (MS) node of the front end.

6. AN EXAMPLE OF STRATEGY

At the model-selection (MS) node the user can get the following background information on the organization of the node

Background on MS organization

Activity MS helps develop some parsimonious models for the data. It first establishes basic information about the GLM:

the response variate
the set of possible explanatory variables
the link function (1)
the variance function
prior weights, if any
offset, if any.

MS then searches for suitable sets of terms to include in a linear predictor. During the search a tree of possible models is constructed and for each node of the tree the current set of possible terms in the linear predictor is divided into three categories

(i) the kernel - terms currently thought of as necessary
(ii) free terms - terms whose status is currently doubtful
(iii) rejected terms - terms removed from further consideration.

Each node has a number and is associated with two basic nodes holding information given in (1) above.

There is a set of tasks available, which are useful steps in model selection procedures and are used by the system's own strategy. They are also available to the user for his/her own purposes.

The strategy used by the front-end is currently described as follows:

Background on MS strategy

The strategy has two main stages. Stage 1 looks for sets of primary terms that give parsimonious models. A primary term is a factor or a variate chosen from the initial set of possible explanatory variables. Stage 2 looks for additional compound terms that may improve the fit. These include squared (x***2) and cross-terms (x1 x2) of variates, interaction terms of factors (A.B), or mixed terms like A.x, where the slope for x varies with the level of a factor A. In more detail:

Stage 1

A series of nodes is created, each representing a tentative model as a kernel K of currently accepted terms, and a set of free terms F whose status is currently doubtful. By implication there is also a set R of terms originally free but now rejected. At each cycle the current set of free terms F is split into 3 subsets

FK : those transferred from F to K
FF : those remaining in F
FR : those transferred from F to R.

At cycle 0 the kernel K contains necessary terms, specified by the user as being essential, and F contains the rest of the initial set of explanatory variables. The three subsets FK,FF,FR are obtained as follows: each free term is tested by forming two F-values; the forward F-value is obtained by adding it singly to the kernel and the backward F-value by removing it from the maximal model K+F which includes all the free terms. The denominator of the F-statistic is either a prior value of the baseline mean deviance or is obtained from the fit of the maximal model. Two critical values for forward and backward F-values are defined, and an F-value exceeding its critical value is called positive, else negative. Any term yields one of four possible results and these are allocated as follows:

forward F-value	backward F-value	allocation
+	+	FK
+	-	FF
-	+	FF
-	-	FR

Default settings for the critical values are both 2. This cycling process continues until either (i) the set F becomes empty or (ii) the set F is non-empty but unchanging. If (i) occurs the stage-1 model selected is unique; if (ii) occurs each remaining free term is transferred to the kernel and the cycling repeated. The result is a tree of possible stage-1 models.

Stage 2

For a selected stage-1 model with initial kernel K1 and free set F1 and final kernel K2 and free set F2 (which may be null), second-order terms (cross-terms) are generated as follows:

Let K'=K1+F1, i.e. all primary terms originally considered. Then generate all compound terms of the form K'x(K2+F2), and assign (subject to marginality constraints) these terms to a set FC. Using K' as a working kernel we find forward F-values for elements of FC. Often there will be too many terms to obtain backward F-values, so we get a working set of free terms by selecting the positive terms from successive forward selections. This working set is then used with a stage-1 procedure to make a selection of compound terms. Finally all simple terms not occurring in any of the accepted compound terms are re-checked for inclusion by a stage-1 procedure.

The advice given by the system is in terms of the tasks defined for the node, and background information about these is also available to the user.

This strategy will undergo further development and refinement as the project proceeds. Data sets suitable for modelling by a wide variety of GLMs are being accumulated, and will be used to test the strategy both here and at other nodes.

REFERENCES

Gale, W.A. (ed.) (1985). Artificial Intelligence and Statistics. Addison-Wesley, Reading (Mass).

Gale, W.A. and Pregibon, D. (1982). An expert system for regression analysis. Computer Science and Statistics: Proceedings of the 14th Symposium on the Interface. Springer-Verlag, New York.

Hammond, P. (1982). APES: A user manual. Research Report DOC 82/9, Department of Computing, Imperial College, London.

Hogger, C.J. (1984). Introduction to Logic Programming, Academic Press, London.

McCullagh, P. and Nelder, J. (1983). Generalized Linear Models. Chapman and Hall, London.

Payne, C.D. (ed.) The GLIM System, Release 3.77 (Manual). Numerical Algorithms Group, Oxford.

Sergot, M.J. (1982). A Query-the-User facility for Logic Programming. Integrated Interactive Computer Systems, (eds. Degano, P. and Sandwell, E.), North-Holland, New York.

Wilkinson, G.W. and Roger, C.E. (1973). Symbolic description of factorial models for analysis of variance. Appl. Statist. 22, 392-349.

ARTIFICIAL INTELLIGENCE TECHNIQUES FOR RETROSPECTIVE HELP IN DATA ANALYSIS

William H. Nugent, Harvard University

With the advent of personal computers and workstations with the computing power and storage capacities of the main frames of 15 years ago, it is a simple task to run interactively multiple analyses on a single dataset. We can now do in hours which used to take days or weeks in the batch environment of 15 years ago. But the ability to so easily explore a data set in a relatively short period of time, begins to strain our capacity to keep the data analysis organized. The record of commands, or script, becomes so lengthy and complicated by the various branchings and dead ends in the process of exploratory data analysis, that it becomes difficult to determine the origins and interdependencies of the objects and data structures in the computer workspace. To help the analyst understand the evolution of his data analysis, statistical software must provide the tools for this meta-analysis problem. This paper presents a tool which has been developed to address the meta problem of script analysis: the determining of the definitions and interdependencies of commands and variables. This is a natural area to automate for three reasons:

1) Searching through a script to find a variable or command reference is a tedious process.
2) An analyst makes mistakes when searching manually through the script.
3) We have proven A.I. technology which can be applied to this problem.

This problem can be partially solved with the use of a search command in a text editor. But this is not a very efficient solution. For example, in searching for the definition of a variable, all occurrences must be examined, even if only a definition is sought. The analyst is forced to perform the following subtask before the variable definition can be found (Fig. 1).

```
SEARCH FOR THE FIRST OCCURRENCE OF THE VARIABLE
WHILE (NOT A DEFINITION)
    SEARCH FOR NEXT OCCURRENCE OF THE VARIABLE
```

Figure 1.

The user is doing the filtering; why not off-load this task to the computer?

To automate this subtask requires a program which has a syntactical knowledge of the computer language being searched by the analyst. That is, the program must be able to determine the difference between commands, functions, variables, and user defined command procedures. Further, such a program must be able to distinguish between the different types of uses of a variable and procedures, their definitions, references, modifications, and deletions. I have developed a program which provides precisely this kind of automation: the SAT program. SAT is a set of Script Analysis Tools.

SAT is a general purpose program which can be easily modified so that almost any computer language can be analyzed. SAT is currently working with ISP (1), but by adapted to a different language by changing the parser. SAT is similar in purpose to the Programmer's Apprentice developed at MIT's AI Lab, for providing a means to analyze a program. SAT allows the user to examine his/her data analysis from a more abstract level. SAT has been developed on an IBM PC using Gold Hill Computers' Golden Common LISP.

SAT's parser generates a database which is then referenced by simple functions that look for definitions, references, or all occurrences of variables and commands. The relationship between different variables can also be examined, resulting in forward and reverse dependency chains. The ability to generate dependency chains between commands is a very powerful tool for examining what the analyst has done in the data analysis, by showing how one command uses the results of an earlier command. What I have described so far is an interactive cross-reference program which can highlight items of interest. It is more efficient at finding specific occurrences in script than either a manual search or a search with a text editor.

Because of the limitations on memory space, SAT is currently only familiar with the syntax of the statistical package ISP. SAT's ability to understand a script currently includes only a general description of ISP commands and functions, however even with this restriction it can perform useful meta-analyses.

ISP commands and functions have order dependent input and output arguments. For example, the linear regression command REGRESS can have two input variables. The first is the independent variable, the second the dependent. SAT does not distinguish between the two inputs. The inputs are treated the same; as inputs. This is also true of results returned by commands, SAT only notes they are outputs. SAT's knowledge of command parameters also suffer from similar shortcomings. Global parameters are recorded, but SAT does not know which parameters are used by the various commands. To understand why these shortcomings are only superficial and not a major design flaw, a deeper understanding of the internal structure of SAT is required.

When SAT builds the database from the user script, cross-reference lists of all occurrences of commands, functions, variables, and procedures are generated. When a variable is being entered into the database, it is noted how the variable is being used: as a definition, reference, modification, or deletion. This task is made easier by the general uniformity of most ISP commands.

```
isp-command operands > results/parameters(=value)
            let result = operand
```

Figure 2.

In the first form in Figure 2, the first word is the command name, the next group of words up to the greater than sign are the inputs, and any words after the greater than sign are outputs. In the second form, to the left of the equal sign is the output and everything after the equal sign is the input. Parameters can occur anywhere on the line and are preceded by a slash. But over all, it is very easy to tag the different occurrences of variables because of the position dependencies of ISP commands. At this time, SAT does not incorporate knowledge about the positional dependencies of ISP operands and results. This would require a special handler for each command to label further each variable. Currently, only the commands which are the exceptions to the generalized ISP command syntax have their own handler; about a dozen of the seventy plus commands diverge from these two general command layouts.

WHY A NATURAL LANGUAGE INTERFACE?

During February 1986, a natural language interface was added to the interactive cross-reference program. A natural language interface was chosen instead of a menu driven system or command language for the following reasons:

1) Both menu driven systems, and command languages interfaces are well known. Both systems have their own advantages and disadvantages. The majority of menu systems are easy for the new user, but become burdensome for the advanced user. The tree structure of most menus causes the advanced user much trouble when having to repeatedly transverse up and down the branches. One area which should be explored for menu driven systems, which seems to be overlooked, is the ability to easily jump between branches; similar to how INFO, the EMACS help facility, works.
2) The traditional command language interface is that the commands will become too complicated to use, possibly even for experienced users.
3) The natural language interface was chosen because it seemed possible to design a natural language interface that would be easy to code, and yet still have the ability to understand complex commands. Another important advantage, is that the system would be able to give a higher degree of feedback; if a command is not understood, the system can ask the user a question. When a command is understood, a generic statement could be echoed to the user telling what the system is doing. With feedback of this nature, the user could learn what the limitations of the system are, and how to get around them. More experienced users could even teach the system new phrases and words.

The natural language interface developed for SAT was inspired by the computer program ELIZA written by Joseph Weizenbaum of MIT (2). This type of natural language interface, or engine, has an outstanding feature, its simplicity. This kind of engine only has a superficial knowledge of the english language. In brief, the way ELIZA works is to process each line of input from the user by searching it for the presence of a keyword. Associated with each keyword is a list of transformation rules. The appropriate transformation rule is applied to the input, and the program would answer back with a question based on the input. SAT, rather than asking a question, performs an action. The input is transformed into a command which is then executed by the interactive cross-reference subsystem.

For example, if a user types in "WHERE WAS VARIABLE PRICES DEFINED LAST", SAT would first print out a generic statement of what it understood the user request to be. Then SAT would search backwards from the current position looking for the last definition of PRICES. If the definition is found, it is displayed on the screen in reverse video along with any comments on the same line.

```
SAT> where was variable prices defined last
 Find the previous definition of variable prices.
```

The keyword in the above example is the word "variable". The associated action rule requires the words "where" "defined" and "last" to appear in the approximate locations shown above. The action rule allows for synonyms. The word "where" could be replaced with "find", "locate", or "show". The same is true for the other two words, "defined" and "last". Symbolically, the transformation rule is shown in Figure 3.

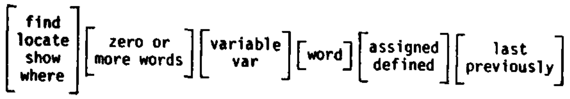


Figure 3.

The action rule in reality is more complicated than what is shown. It is able to represent many types of sentences in a single rule by allowing some words to be translated to symbols which are passed to the underlying LISP function. The above example has been simplified for the sake of understanding and brevity.

If not enough information is typed in by the user, SAT will respond with a question. For example:

```
SAT> where was variable xmax ( user input )
  WHAT ABOUT VARIABLE XMAX?  (SAT response)
```

At this time, the help system is not as good as one would like. It should help the user by listing some possiole keywords that would make the user's input into a valid action statement.

The types of questions SAT interprets are about the location of commands, functions, and variables. SAT is also able to build both forward and reverse dependency chains on either variables or command lines. A dependency chain can best be thought of as a question such as "WHAT DEPENDS ON VARIABLE PRICES?" or "WHAT DOES VARIABLE PRICES DEPEND ON?". In searching for the occurrence of a variable, SAT is able to distinguish between definitions, references, and deletions. The user is able to request searches in either the forward or reverse direction with only the next or all occurrences being sought.

At the present, SAT is not able to interpret complex user commands. A user input of "SHOW ALL REGRESSION COMMANDS USING VARIABLE X AS THE DEPENDENT VARIABLE" is too complicated and, as was mentioned earlier, SAT does not currently have a parser sophisticated enough to differentiate between input and output arguments.

FUTURE DIRECTIONS FOR SAT:

The power of SAT could be greatly enhanced by integrating it with a full parser for the underlying computer language, such as S, C, or some other computer or data analysis language. With a full parser, SAT would be able to differentiate between the modification of an array cell and the definition of the array. SAT will also be able to differentiate between order dependent arguments to commands and functions.

The next area in which SAT can be enhanced is its supporting complex and compound natural language commands. The user needs to be able to specify a compound natural language command, such as

"SHOW THE GSCAT FUNCTION WHICH USES
VARIABLES DATES AND PRICES"

The ability to process 'and', and 'or' clauses is necessary for further flexibility and functionality.

SAT also needs to be able to understand user macros or command procedures. When a macro is read in, SAT should treat that segment of code in a special manner. When the macro is later invoked in the script, some guesses can be made as to which variables will be modified as a result of the macro execution. Since a macro may have commands which are conditionally executed, some variables may not be modified when a particular macro is invoked. It is possible that there is information outside the scope of the static code image that SAT analyzes. Therefore, SAT has to make a guess about how these code segments affect variables in the code segments. SAT needs to be able to flag these variables as being in a gray area of SAT's ability to determine, and what are known facts concerning macro execution.

Another area which SAT could help the user is by providing special graphics for building a graph of the analysis tree as performed by DINDE (3). The ability to group sequences of commands together and displaying them as a node in a graph helps the analyst to abstract his/her work and to think about it at a higher level. By providing a graph with the results of analysis, reviewers can easily see what was done without having to read the particular statistical language.

Finally, SAT needs to be included into the design of a statistical computer language. By incorporating SAT into the design, SAT would gain clear knowledge about what occurs inside macros, what the different array dimensions are and how these affect commands. Further, by integrating SAT into a statistics package along with an editor, SAT could provide feedback on script changes as the user makes them.

SAT is a first step in providing a comprehensive set of meta-tools for a statistical computer language to help the analyst document and understand what has been done. When fully integrated, the power of the tools will increase because the information about a workspace environment can be directly accessed by SAT, rather than guessed at. The full extension of SAT would share many ideas with DINDE, but with a major difference; SAT would have an underlying command language.

Some of the options such an environment would provide are:

-- A script cleaning tool similar to Lint to provide the user with diagnostics about his/her script as to which commands are only informative, unnecessary, and the like, would help the analyst to streamline the code, and to help find potential trouble spots.
-- A Macro Learner, a tool that searches through multiple scripts looking for common commands sequences which could be generalized as a new macro. A prototype has already been written by Russell Almond, a graduate student at the Department of Statistics at Harvard University.
-- A Storage Manager which stores on disk the various scripts, session records, workspaces, and graphs so they can be retrieved and modified with full context.
-- A Perspective Help Daemon, which monitors the data analysis session progress and suggests potentially useful macros. The Perspective Help Daemon would work by comparing the user input to the macro library, and if a close match was found, it would suggest use of the macro.
-- Creation of the SAT program marks the beginning of a data analysis environment where many tedious housekeeping chores are assumed by the computer. Also, the computer can compare what the analyst is doing to a known database of previous sessions and libraries of macros to suggest alternative methods.

Right now, we are enjoying the hardware and software of small powerful machines. Analyzing large databases and more importantly doing multiple analysis quickly is something we can all do. Dreams of 15 years ago are our reality. Along with this reality, we have discovered the meta problem of maintaining coherency in our analysis paths. We can do so much so quickly we now must pay attention to organizing and knowing our voluminous output.

Developing the software, such as SAT, is going to be an exciting area of research both because of tough technical problems and to provide easy access for the unsophisticated user to these new high power tools.

ACKNOWLEDGMENTS:

I would like to thank Roy Welsch and Tom Boardman for inviting me to present this paper. I would also like to thank Russel Almond and Lenn Roseman for helping shape this paper and for useful suggestions on SAT. The Office of Naval Research for support with my Graduate Fellowship, and Peter Huber for his support in the preliminary research of this project with his NSF and ONR grants, and for suggesting the idea of a natural language interface.

REFERENCES:

1. Donoho, D.L. (1981), ISP 1.5 Command Descriptions, Harvard University, Department of Statistics.
Donoho, D.L. (1982), User's Guide for ISP on VMS, Harvard University, Department of Statistics.

2. Weizenbaum, J. (1966), ELIZA -- A Computer Program for the Study of Natural Language Communication between Man and Machine, Comm. ACM, 9:36-45.

3. Oldford, R.W. and Peters, S.C. (1985), DINDE: Towards more Statistically Sophisticated Software, Tech. Report No. 55, CCREMF MIT.

COMPUTER SUPPORT FOR SURVEY SAMPLING

Organizer: *William J. Kennedy, Iowa State University*

P.C. CARP: Variance Estimation for Complex Surveys
Dan Schnell, Gary Sullivan, William J. Kennedy, Wayne A. Fuller, Iowa State University

The Status of Computer-Assisted Telephone Interviewing
William L. Nicholls II, Bureau of the Census; Robert M. Groves, University of Michigan

Inference from Coarse Data Using Multiple Imputation
Daniel F. Heitjan, University of California, Los Angeles; Donald B. Rubin, Harvard University

P.C. CARP: VARIANCE ESTIMATION FOR COMPLEX SURVEYS

Dan Schnell, Gary Sullivan, William J. Kennedy and Wayne A. Fuller
Iowa State University

Taylor approximations for the variances of the approximate distributions of statistics computed from complex surveys are outlined. A program implementing variance estimation on the IBM-PC for use with large scale surveys is described. The program will compute estimators and estimated variances for totals, ratio subpopulation means and regression coefficients.

I. Introduction

Most large scale surveys of human population are of relatively complex design. Typically the population is subdivided into subgroups, called strata, and independent samples selected from each stratum. Sampling rates are often different in different strata. Also it is common to select individuals in clusters. Examples of such clusters include all persons living in a geographic area such as village and all persons in a particular housing unit. Stratification and clustering do not exhaust the complexities present in most surveys, but they are sufficient to explain why most samples cannot be treated as simple random samples of individuals.

The description of stratified cluster samples also establishes the three main components that determine the way an observation is treated in an analysis of survey data. These are the (1) stratum to which the individual belongs, (2) the primary sampling unit (cluster) to which the individual belongs, and (3) the weight (equal to the inverse of the selection probability) for the individual. The data record for an individual used in a statistical analysis must contain these three components.

The second dimension of survey analysis that requires special consideration is the volume of estimates produced. The basic output of most surveys is a large number of tables, most of which are two-way tables. Given the typical survey design, each entry in the table is a rather complex function of the observations. Consider an estimate of the fraction of females 26 through 30 years of age that are employed. For a stratified cluster sample, this estimate is defined by

$$\hat{\theta} = \left(\sum_{i=1}^{L} \sum_{j=1}^{n_i} \sum_{k=1}^{m_{ij}} w_{ijk} X_{ijk} \right)^{-1} \times \left(\sum_{i=1}^{L} \sum_{j=1}^{n_i} \sum_{k=1}^{m_{ij}} w_{ijk} Y_{ijk} \right) = \left(\sum_{i=1}^{L} \sum_{j=1}^{n_i} \hat{X}_{ij.} \right)^{-1} \sum_{i=1}^{L} \sum_{j=1}^{n_i} \hat{Y}_{ij.} ,$$

where

$$\hat{X}_{ij.} = \sum_{k=1}^{m_{ij}} w_{ijk} X_{ijk} ,$$

$$X_{ijk} = 1 \quad \text{if the individual is an employed female 26 through 30 years of age}$$
$$= 0 \quad \text{otherwise,}$$

$$Y_{ijk} = 1 \quad \text{if the individual is a female 26 through 30 years of age}$$
$$= 0 \quad \text{otherwise,}$$

w(ijk) is the weight for the k-th individual in the j-th cluster of the i-th stratum, m(ij) is the number of individuals in the j-th cluster of the i-th stratum, n(i) is the number of clusters in the i-th stratum, and L is the number of strata. The clusters are the primary sampling units and the estimator of θ is the ratio of sample means of cluster totals. As such, it is a nonlinear function of the cluster means. It follows that a method appropriate for nonlinear functions must be used to estimate the variance of the approximate distribution of the estimator. See Wolter (1985) for a discussion of variance estimation for complex surveys. The Taylor method (method of statistical differentials) is used in PC CARP. For the ratio estimator, the variance is estimated by

$$\hat{V}\{\hat{\theta}\} = \sum_{i=1}^{L} h(n_i, N_i) \sum_{j=1}^{n_i} (d_{ij.} - \bar{d}_{i..})^2 ,$$

where

$$h(n_i, N_i) = (n_i - 1)^{-1} n_i (1 - N_i^{-1} n_i)$$

$$d_{ij.} = \hat{X}_{...}^{-1} \sum_{k=1}^{m_{ij}} w_{ijk} (Y_{ijk} - \hat{\theta} X_{ijk}) ,$$

$$\hat{X}_{...} = \sum_{i=1}^{L} \sum_{j=1}^{n_i} \sum_{k=1}^{m_{ij}} w_{ijk} X_{ijk}$$

$$\bar{d}_{i..} = n_i^{-1} \sum_{j=1}^{n_i} d_{ij.} ,$$

and N(i) is the population number of clusters in the i-th stratum. The variance of the ratio estimator is given in such standard tests as that of Cochran (1977).

The project to develop statistical software for complex surveys is a joint undertaking between Iowa State University and the International Statistical Programs Center of the U.S. Census Bureau. The objective is to provide developing countries with software that can be used locally to process survey data collected locally. The Iowa State University project on variance estimation is a part of a larger project that includes the development of software for survey management, data editing and tabulation.

Beginning in the early 1970's, based on the work of Hidiroglou (1974) and Fuller (1975), a program was developed at Iowa State University for the computation of regression coefficients and the estimated covariance matrix of the coefficients for survey data. The program, called SUPER CARP, was later expanded to include total estimation, ratio estimation, subpopulation statistics, two-way tables and two stage samples. The last revision of SUPER CARP took place in 1980. That program furnished the starting point for the development of PC CARP.

The IBM Personal Computer XT was chosen by the Census Bureau as the equipment for which the software was to be designed. The personal computer seems an ideal machine for developing countries for several reasons. First, it is relatively tolerant of its environment, both physical and personal. When compared to mainframe computers, the personal computer can accept greater variation in temperature, humidity, and of electric current. The personal computer also has much lower requirements for trained operators and maintenance personnel. See Diskin (1985) for a description of the problems developing countries face in maintaining trained staff. The personal computer is under the direct control of the user and if a personal computer is placed in a survey unit, access to the computer becomes relatively easy. (Nothing is ever guaranteed in a bureaucracy.) Finally, the personal computer is inexpensive. A superior configuration for the program under the development costs about $6,000.

The IBM Personal Computer XT is equipped with a hard disk drive, one floppy disk drive and a monitor. In addition, PC Carp requires that the machine have 256K bytes of memory and a math coprocessor. The program is written almost entirely in FORTRAN. The FORTRAN language was chosen because it is the most widely known scientific programming language; hence, if necessary, the program can be easily modified to suit particular needs of the user. The IBM Professional FORTRAN compiler was selected for the project. A small portion of the code - some sections of the user interface - is written in IBM Assembly language. The program runs under DOS operating system, Version 3.0.

II. *Program Capability*

PC CARP is capable of handling both large and small data sets with equal ease and efficiency. It is most desirable to store large data sets on the hard disk because of its large capacity and the speed at which data can be transferred. If the hard disk is not available, large data sets may be stored on a series of floppy diskettes. A single floppy diskette is usually sufficient for a small data set. The program is also capable of accepting data entered from the keyboard during execution. The program sets no limit on the number of strata or clusters that can appear in a data set and a data set may have up to 50 input variables. The program accepts disk data files in either fixed or internal (binary) format. Along with the data, the user has the option of providing stratum sampling rates (f(i)) . These rates are kept on a diskfile separate from the data and are used in variance computations.

The program can be used to compute variances for one or two stage samples. An example of variance estimation for the ratio estimator in a one stage sample is given above. The relevant variance is within strata-between cluster variance component. In a two-stage sample, a second component, the within cluster variance component, also enters the variance expression. The program computes within cluster sampling rates (f(ij)) from the stratum sampling rates and the individual record weights. The within cluster sampling rate is f(ij) = 1/(w(ijk)f(i)) . These are sequentially written into a diskfile. The second component is added to the variance estimators if the user selects the two-stage option.

For purposes of variance computation, the user may instruct the program to eliminate one cluster strata by selecting the collapse option. If this option is chosen, a one cluster stratum is grouped with the following stratum in the data set by changing the stratum and cluster identifications on the involved records. To illustrate stratum collapse consider a simple data set composed of three strata, one of which contains a single cluster.

Input Data:	Str	Clus	Data
	1	1	Record 1
	1	2	Record 2
	2	1	Record 3
	3	1	Record 4
	3	2	Record 5

The algorithm combines the second stratum, represented by only one cluster, with the third stratum. The number of data records and unique clusters is unchanged. The collapsed data set is:

Collapsed Data:	Str	Clus	Data
	1	1	Record 1
	1	2	Record 2
	2	1	Record 3
	2	2	Record 4
	2	3	Record 5

The renumbered records are written to a new "collapsed" data file which is retained after the user exits the program. If stratum sampling rates are present, new rates are defined by

$$f_i^* = (n_i/f_i + n_{i+1}/f_{i+1})^{-1}(n_i + n_{i+1}) ,$$

where stratum i , with n(i) = 1 cluster, has been combined with stratum i+1 . These new rates are saved in an auxiliary rate file. One can see from the example that different orderings of the strata may produce different collapsed data sets and different collapsed stratum rates. A preliminary pass through the data is necessary when either the collapse or the two-stage options is selected.

III. Available Analyses

Table 1 contains a description of the types of statistics available to the user and of the nature of the computations required to obtain the estimates. A "Y" in the column headed "Cov" means that the covariance matrix of a vector of estimates of the type listed on the left can be obtained. The design effect, denoted by DEFF, is available as an option for many of the statistics. See Kish (1965) for a description of the design effect.

The population (Total and Ratio) analyses and stratum analyses are performed in a straightforward manner. Some details pertaining to Subpopulation Analyses, the Two-Way Table and the Regression Analysis are given below.

Table 1. Analysis capability of PC CARP

Analysis	Cov	DEFF	Comments
Total Estimation	Y	Y	1 pass; 40 variables
Ratio Estimation	Y	Y	1 pass; 50 variables without covs. 15 variables with covs.
Stratum Totals	Y	Y	1 pass; 40 variables
Stratum Means	N	Y	1 pass; 50 variables
Stratum Proportions	N	Y	2 passes; 50 variables
Subpopulation Analyses:			
Totals	N	Y	Crossed classifications; Multiple dependent variables; Multiple passes.
Means	N	Y	
Proportions	N	Y	
Two-Way Table	N	N	50 cells; 1 pass/dependent; Tests of independence
Regression (WLS)	Y	N	2 passes; 35 variables; Multiple degrees of freedom hypotheses tests; Residuals and predicted values

NOTE: Coefficients of Variation are computed for all estimators.

The subpopulation analyses give the user the option of crossing classification variables. This allows the user to create new classification structures from two or more input variables. For example, suppose the input data includes the classification variables age, sex and education with six, two and five levels, respectively. Then, by crossing age with sex with education, a new classification structure with 60 levels is produced. The user may then obtain estimates for any number of dependent variables under this classification structure.

The Two-Way Table analysis is defined by two classification variables and at least one dependent variable. Four tables are then computed for each dependent variable:

- Cell totals along with marginal row and column totals
- Conditional row proportions for each cell
- Conditional column proportions for each cell
- Cell proportions along with the marginal row and column proportions.

Standard errors are computed for all of the above estimators. Also, a test statistic for the hypothesis of independence is output.

The weighted least squares regression analysis computes coefficient estimates and an estimated variance-covariance matrix, which takes into account the sample design. These calculations are given in Fuller (1975) and outlined in Hidiroglou et al (1980). Multiple degrees of freedom F-tests for sets of coefficients and the usual t-statistics are available. The user also has the option of obtaining residuals and estimated true values.

IV. Program Details and User Interface

In PC CARP, certain tasks must be performed repeatedly, regardless of the analysis. These include data management, error handling and program output.

The program relies on a single data management subroutine which performs the following functions:

- Reads data from the diskfile and passes it onto any of the subroutines performing data organization or analysis.

- Retrieves and sends rates (stratum or two-stage) associated with each data record.

- Manages the set of files in which the data and rates are stored.

Isolating these functions in one routine allows an analysis routine to be readable and uncluttered with data management code. Also, this allows all analysis routines to be structured in a similar way.

In constructing the error handling system, the most important consideration was to avoid program termination caused by user misspecifications that could be easily corrected. These include checks for omitted responses, improper file names and invalid analysis variable specifications. If such an error is detected, PC CARP allows the user to re-enter information or exit the program unscathed. The program routinely performs checks to avoid computational errors such as division by zero.

The user has the option of sending program output to any combination of diskfile, screen or printer. Within the program, output is formed a line at a time. First, the output line is written to a "buffer", which is actually a character array. The character array is then sent to a subroutine which, in turn, routes it to the proper output device(s). As with data management, this approach prevents the unnecessary repetition of output statements.

Two primary concerns at the program development stage were to have a friendly user interface and to minimize the number of passes through the data. The interface was made user friendly by implementing an interactive, screen oriented response system, while a single pass algorithm for variance estimation helped minimize the amount of reading from data files.

When information is needed by PC CARP, the user receives a full screen of short response questions along with detailed instructions. The first set of screens displayed to the user ask for information pertaining primarily to data organization and location. One such screen is pictured below.

```
                        PC CARP - Problem Specification

INSTRUCTIONS: Key the problem identification and press the ENTER KEY. Next
use the "arrow" keys on the numeric keypad to position the cursor. Key a
response to each and every requested item. Responses replace slashes shown on
the screen. When you have finished keying responses, press the "END" key on
the numeric keypad (lower right side of keyboard).
                  ***                  ***                  ***

1.  Type a problem name(id) on the next line, replacing the slashes
   / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / /

2.  Give the total number of variables input (Replace //, Example 09) . . . /

3.  Do you wish to have an intercept variable generated? (Respond Y or N)  . /

4.  Do you wish to enter the variable names using the keyboard? . . . . . . /

5.  Specify (Y or N) in each case whether the following items will be input
    with the observations.  (Unit weights are the default)
                                a.  STRATUM ID . . . . . . . . . . . . . . . /
                                b.  CLUSTER ID . . . . . . . . . . . . . . . /
                                c.  WEIGHT . . . . . . . . . . . . . . . . . /

6.  Do you wish to enter the data using the keyboard? (Respond Y or N) . . . /

7.  Will stratum sampling rates be provided? (Respond Y or N)  Note that the
    response to 5a. above must be "Y" if sampling rates are provided . . . . /

8.  Is this a two stage sample?  (Y or N, if "Y" then response to 7 is "Y") /

9.  Output from analysis is routed to (Y or N)
                                a.  PRINTER  . . . . . . . . . . . . . . . . /
                                b.  SCREEN . . . . . . . . . . . . . . . . . /
                                c.  DISK . . . . . . . . . . . . . . . . . . /

10.  Do you wish to collapse strata (to avoid single unit in a stratum)  . . /
```

It is representative of the screens in PC CARP in that the user moves the cursor about the screen replacing slashes (///) with Y or N, numeric or short answer responses. Other screens will appear for data file information and variable name specification. After information pertaining to the data is input, the remaining screens allow the user to choose their analysis, available options and analysis variables. The particular screen which appears for variable selection depends upon the choice of analysis. For example, if a subpopulation analysis is specified, the user must key in classification variables, upper bounds for the number of levels, crossing indicators and dependent variables. If ratio estimation is specified, the numerator and denominator variables are entered in pairs for each ratio. The analysis specification and variable selection screens will appear for each analysis the user wishes to perform. PC CARP produces the screens using FORTRAN formatted write statements while cursor movement and response retrieval are supported by assembly language routines.

Up to three different variance quantities can be accumulated concurrently for any given estimator. These are the first stage variance component, the optional second stage variance component and the simple random sampling variance used in the computation of the design effect. Each variance is accumulated using a single pass algorithm for weighted means and weighted sums of squares and cross products matrices. The algorithm is described in Herraman (1968). By computing all variance quantities in a single pass through the data, a large amount of array space is needed. However, the elimination of entire passes through the data outweighs the use of additional array space.

Acknowledgement

This work was partly supported by the United States Census Bureau under agreement 50-YABC-6-66014. We thank Heon Jin Park for his assistance in testing and debugging the programs.

References

Cochran, W. G. (1977), Sampling Techniques. Wiley, New York.

Diskin, B. A. (1985), Microcomputers in developing country statistical offices: current use and a look to the future. Proceedings of the 45th Session of the International Statistical Instititute, Amsterdam.

Fuller, W. A. (1975), Regression analysis for sample survey. Sankhya C 37, 117-132.

Herraman, C. (1968), "Sums of squares and products matrix." Applied Statistics, 17, 289-292

Hidiroglou, M. A. (1974), Estimation of regression parameters for finite populations. Unpublished Ph.D. thesis, Iowa State University.

Hidiroglou, M. A., Fuller, W. A., and Hickman, R. D. (1980), SUPER CARP Dept. of Statistics, Iowa State University, Ames, Iowa.

Kish, L. (1965), Survey Sampling. Wiley, New York.

Wolter, K. M. (1985), Introduction to Variance Estimation. Springer-Verlag, New York.

THE STATUS OF COMPUTER-ASSISTED TELEPHONE INTERVIEWING

William L. Nicholls II, Bureau of the Census and Robert M. Groves, University of Michigan

Computer-assisted telephone interviewing, or CATI, lies on the interface between computer science and statistical data collection. It employs interactive computing systems to assist interviewers and their supervisors in performing the basic data collection tasks of telephone interviewing. This paper: (1) presents a definition of CATI; (2) reviews its growth and current status as a new data collection technology; and (3) summarizes available evidence on its consequences for survey interviewing costs and data quality.

1. Definitions

Computer-assisted telephone interviewing is part of a broader family of technologies called "computer-assisted data collection." In addition to CATI, this family includes: (1) computer-assisted personal interviewing (CAPI) which employs portable microcomputers for interviews in respondents' homes or offices; and (2) computerized self-administered questionnaires (CSAQ) in which similar equipment is operated directly by respondents. All three technologies may employ similar hardware and software; but their data collection characteristics probably vary with their usage and with the settings in which they are employed.

At least in principle, CATI and CAPI provide interviewers with the same types of online interviewing assistance. In state-of-the-art CATI systems:

a. The system displays instructions, survey questions, and response categories on the interviewers' screens.

b. The screen may contain "fills" or alterations of the display text based on prior answers or batch input.

c. Answers to closed questions may be entered by numeric or alphabetic codes. These and other numeric entries may be edited by sets of permissible values, by ranges, or by logical or arithmetic operations.

d. Edit failures result in: (1) an unaccepted entry and error message requiring another attempt; or (2) in display of additional probes or questions to be asked.

e. Extended text answers may be entered to open-ended questions.

f. Branching or skipping to the next item is automatic and may be based on logical or arithmetic tests on any prior entries or input data.

g. Interviewers may interrupt and resume interviews in mid-course; review, backup to, and (when permitted by the survey design) change prior entries; and enter interviewer notes at appropriate points.

This paper focuses on computer-assisted telephone interviewing and even more specifically on multi-station CATI systems. One-station CATI systems exist (Philipp and Cicciarella, 1983) in which each interviewer independently operates a stand-alone microcomputer to complete online interviews with assigned batches of cases. In multi-station systems, the interviewing stations are linked or networked to a common host. This permits many additional case management features including: system assignment of cases to interviewers; shared workloads and system scheduling of telephone calls and callbacks; online visual and audio monitoring of interviewers from supervisory stations, and common record keeping. While some of CATI's consequences for survey costs and data quality derive from online interviewing features shared with CAPI and one-station CATI systems, others result from the case management and supervisory capabilities of multi-station systems.

A CATI system may provide the equivalent of a blank questionnaire on which the interviewer enters a case number and other input data before placing a call. More commonly the system contains a file of sample cases identified by case numbers. The interviewer may access a case in one of two ways: (1) by case call-up, where the interviewer enters the identification number of a selected case; or (2) by online case assignment and call scheduling where the interviewer indicates readiness for an interview and the system supplies a case appropriately called at that time. This reduces the need for interviewer maintenance and review of paper or displayed listings in choosing cases to call.

Current CATI systems differ greatly in their use of online call scheduling. Some lack this capability, some limit its use to previously uncalled cases, and some employ it only up to the point where the sample household or office is reached. Others use it for virtually all calls except problem callbacks, such as recalls to initial refusals and missed appointments. The range of circumstances to which online call scheduling is applied should affect both interviewer productivity levels and measures of data quality dependent on the frequency and efficiency of calling.

2. History and Status

Market research agencies in the private sector created the first computer-assisted telephone interviewing systems and established initial expectations of CATI's data collection characteristics. Based on experiences in the first CATI survey, conducted by Chilton Research for AT&T in 1971, Nelson, Peyton, and Bortner (1972) described "three distinct advantages" for cathode ray tube interviewing (as it was then called) in comparison with conventional data collection methods. These were: "accuracy, speed, and reduced costs."

Throughout the 1970's and early 1980's, the number, size, and sophistication of commercial market research CATI systems increased rapidly (Dutka and Frankel, 1980; Fink, 1983; Smith and Smith, 1980). Single installations of 100 or more stations now exist, as do networks of geographically dispersed field sites. At least 100 CATI installations are in operation by commerical market research agencies. The majority are in the United States; but market research firms using CATI are also found in Australia, Canada, Italy, Great Britain, the Netherlands, New Zealand, Sweden, Switzerland, and West Germany.

University research centers began their largely independent development of CATI five years after its introduction in market research. The UCLA Center for Computer-Based Behavior Sciences led the way; and the Center's Director, Gerald Shure, coined the CATI name and acronym (Shure and Meeker, 1978). Development work at the Berkeley, Michigan, and UCLA Survey Research Centers and the Wisconsin Survey Research Laboratory followed shortly thereafter (Groves, 1983; Palit and Sharp, 1983; Shanks et al., 1980). Academic survey research centers greatly expanded the range of CATI capabilities, especially for probability (rather than quota control) sampling, call scheduling and callback routines necessary for high response rates, and greater freedom of interviewer movement. Today at least 18 university and private research organizations, such as RAND, RTI, Westat, and Mathematica, employ CATI.

U.S. governmental agencies demonstrated an early interest in CATI, but steps to acquire their own CATI capabilities did not begin until 1980 when the the U.S. Department of Agriculture Statistical Reporting Service and U.S. Census Bureau both established internal staffs for this purpose (House, 1984; Nicholls, 1983). Both completed their first tests of CATI in 1982 and have continued CATI testing and production data collection since that time. In the Netherlands, the Central Bureau of Statistics began CATI data collection for continuing surveys in 1984.

A major expansion of governmental CATI installations began in 1985. The USDA Statistical Reporting Service started placement of production CATI facilities in state offices with the goal of completing this task by January 1989. The Centers for Disease Control began installing CATI in 25 state offices (most of these two-station sites) for use in a continuing survey. The Bureau of Labor Statistics began testing CATI production on three surveys. The U.S. Census Bureau opened a 40-station CATI facility for production data collection and expanded programs of testing on demographic surveys while its Business Division began automating selected economic surveys with a system including CATI functions. The Census Bureau is currently developing plans for full CATI and CAPI implementation on the Current Population Survey and the National Crime Survey. If further evaluative tests prove successful and the survey's sponsoring agencies approve, this major change in data collection methods should be completed in the 1990's. Outside the U.S., Statistics Sweden began procurement in 1985 for an integrated prototype CATI/CAPI system for its major surveys; and this year Statistics Canada will begin testing CATI in its Labor Force Survey.

Applications to governmental data collection have placed new demands on the design of CATI surveys. A common government application is in telephone follow-up to mail nonresponse. This has required systematization of procedures for telephone tracing of difficult to reach respondents for inclusion in CATI interviewing and automatic call scheduling systems (Ferrari, 1986 and Nicholls, 1983). A related application with special promise for longitudinal demographic surveys is use of CATI for second- and later-visit interviews after an initial interview in person. In such designs, personal visits (eventually by CAPI) may continue for households unreachable by telephone. Other potential uses by governmental agencies include failed-edit follow-up and reconciliation reinterviewing.

3. CATI's Data Collection Characteristics

Survey organizations considering adoption of CATI frequently ask the following questions: (1) How much will CATI cost to install and operate?; (2) How will CATI change the time required to design and complete a survey?; and (3) What effects will CATI have on data quality compared with conventional methods?

In a previous paper, the authors (Nicholls and Groves, 1985) attempted to summarize the existing literature on CATI to determine how fully each of these questions could be answered. Our general conclusion was that the past literature provided few firm answers about CATI's effects on survey costs, timeliness, or data quality. This paper will take a somewhat more encouraging stance. Published and unpublished research released during the last year is contributing to a better (although still far from complete) understanding of at least some of CATI's data collection characteristics. These include key factors in interviewing costs and selected consequences for data quality. The remainder of this paper will focus on these areas.

By most standards, this evidence is still relatively weak. Four studies have been published which are (or closely approximate) controlled experiments in which probability subsamples of the same survey are interviewed by CATI and paper methods at the same time by the same staff under controlled conditions. These are the SRC-Michigan RDD Health Survey Test (Groves and Mathiowetz, 1984); the USDA California Dual Frame Cattle Inventory Survey (House, 1984 and Tortora, 1985); the USDA Nebraska Hog Survey (Coulter, 1985); and the Westat Florida Colo-Rectal Cancer Kin Survey (Harlow, et al., 1985). Most of these experimental studies employed relatively small samples, ranging from about 130 to 1,200 CATI cases and as few as four or five CATI stations. All also represent relatively early use of CATI by their organizations. Comparable information from organizations with at least three years CATI production experience is generally available only in the form of summary impressions rather than quantitative data (Palit and Sharp, 1984).

Comparisons of CATI and paper-and-pencil data collection are also available from four comparative studies which do not meet the requirements of fully controlled experiments. The SRC-UCLA Earthquake Survey (Fielder, 1985) and the SRC-Berkeley Malignant Melanoma Survey (Coleman, 1985) approximate before-and-after designs. Earlier waves of these surveys were conducted by paper methods and later waves by CATI. The comparisons are limited to first-visit interviews in repeated cross-sections or control samples. The U.S. Census Bureau also completed tests of CATI for telephone follow-up to mail nonresponse in the National Survey of Scientists and Engineers (Ferrari, 1984) and the 1982 Census of Agriculture (Ferrari, 1986). Each test assigned probability subsamples of 7,000 or more cases to each treatment, but the CATI and non-CATI staffs worked at widely separated sites which followed different hiring, supervisory, and management procedures. These uncontrolled factors and difficulties encountered in recovering all field work records from the the non-CATI site requires caution in interpreting their results.

While each of these studies is limited in sample size, design, or experience with CATI, they have the collective strength of representing largely independent efforts by seven different investigators in six different organizations utilizing five different CATI systems. Where consistent results are found, they suggest generalizations which may apply across varying organizational settings.

4. Data Collection Costs

The total costs of CATI data collection will depend on many factors, including the costs of: (1) hardware and software acquisition, installation, and maintenance; (2) CATI survey design, setup, and debugging; (3) interviewing costs; and (4) data preparation costs. Summary impressions have been published which suggest that total costs of a CATI survey will be less than those of a comparable survey conducted by paper-and-pencil methods, but these impressions have not been accompanied by supporting detailed evidence (Nelson et al., 1972; Palit and Sharp, 1983).

Quantitative evidence is available bearing on only one cost component, interviewing costs. Telephone interviewing may be divided into three main tasks: (1) placing calls to reach designated respondents; (2) interviewing respondents when reached; and (3) post-interview clerical tasks, such as editing completed forms for consistency, maintaining records of calls, and transcribing data between forms. Research on CATI has most frequently focussed on interview length once respondents have been reached.

The results of three experimental and two comparative studies are summarized in Table 1. House (1984) found the mean length of CATI and non-CATI interviews to be equal but reported problems of obtaining comparable timings across modes. The remaining four studies found CATI interviews to be longer. Groves and Mathiowetz (1984) and Harlow et al. (1951) reported CATI interviews about 13-14 percent longer in surveys where the telephone was the primary mode of data collection. Coleman (1985) found CATI interviews 22 percent longer but part of the difference was attributable to added questions in the CATI questionnaire. Ferrari (1984) reports CATI interviews 50 percent longer in one use of telephone interviews for follow-up to mail nonresponse, but the CATI timings included additional activities and also were accompanied by substantially lower rates of item nonresponse. While some uncertainties of measurement are present in all five studies, collectively they suggest that CATI interviews tend to be at least somewhat longer than comparable paper-and-pencil interviews.

TABLE 1

MEAN LENGTH OF COMPLETED INTERVIEW
IN MINUTES BY MODE IN FIVE SURVEYS

Survey	CATI	Non-CATI
USDA Cattle Multiple Frame Survey (House, 1984)..........	8.2	8.2
SRC-Michigan National Health Survey Test (Groves and Mathiowetz, 1984).............	52	46
Westat Colo-Rectal Cancer Survey (Harlow et al., 1985).........	28.5	25.1
Malignant Melanoma Study (Coleman, 1985)...............	10.9	8.9
U.S. Census Bureau Survey of Scientists and Engineers Telephone Follow-Up (Ferrari, 1984)	20.8	13.7

Although CATI relieves the interviewers of the task of turning pages and of finding the next question to ask, several hypotheses, none confirmed, have been advanced to explain the apparently longer length of CATI interviews. First, experienced paper-and-pencil interviewers often begin asking the next question while recording the last. With CATI, this is more difficult because the next question often is not displayed until the answer to the prior question is entered. Second, entering responses to open-ended questions may take longer in CATI because most interviewers write somewhat faster than they type (Groves and Mathiowetz, 1984). Third, to the extent that CATI ensures completion of items, probes, or other interviewing tasks occasionally missed with paper-and-pencil methods, this also will lengthen CATI interviews. Further increases in length will occur when the CATI questionnaire includes edit checks requiring added probes or other interviewer actions to reconcile apparent inconsistencies (Morton and House, 1983).

Although CATI interviews tend to be longer, CATI interviewers may spend less time between interviews. With efficient online call scheduling and case assignment, CATI systems should reduce interviewer time selecting cases to call and maintaining call records. Automatic branching between items, online editing, and the recording of entries directly on computer files should reduce the need for post-interview clerical review of completed forms and transcribing data between forms. The net effect may be an increase in interviewer productivity. Nelson et al., (1972) reported a 10 percent increase in interviewer productivity in the first CATI survey conducted in 1971. In 1983, Palit and Sharp (1983) reported a 20 percent increase in interviewer productivity (measured in sample points contacted per production hour) compared with paper methods in random digit dialing telephone surveys. Neither paper describes the methods by which the comparisons were made nor presents supporting data.

Estimates from two recent studies, shown in Table 2, suggest that the productivity of CATI interviewers may depend on the use of online call scheduling and case assignment. Using a CATI system without these features, Coulter (1985) reported CATI interviewers 12 percent *less* productive than paper-and-pencil interviewers when productivity was measured by the combination of completed and refused interviews per hour. By contrast, in the Census of Agriculture telephone follow-up which made extensive use of online call scheduling and case assignment, Ferrari (1986) found CATI interviewers 45 percent *more* productive than paper-and-pencil interviewers when measured by completed interviews per paid interviewer hour. Productivity appears to have been increased by reducing the time spent between interviews. The CATI interviewers placed 23 percent more calls per hour and spent 31 percent more time on the phone than the paper-and-pencil interviewers.

TABLE 2

ESTIMATES OF INTERVIEWER PRODUCTIVITY BY MODE IN TWO SURVEYS

Survey and Measure	CATI	Non-CATI
USDA Nebraska Hog Survey (Coulter, 1985)		
Completions and refusals per interviewer hour....	5.3	6.0
Sample size in cases......	(550)	(575)
Census of Agriculture Telephone Follow-Up (Ferrari, 1986)		
Completed interviews per paid interviewing hour..	1.06	0.73
Telephone calls per paid interviewing hour.......	10.42	8.45
Minutes phoning per paid interviewing hour.......	49.39	37.77
Sample size in cases......	(7,688)	(5,427)

The increased productivity reported for CATI by Ferrari is inflated by CATI's higher response rate and perhaps by: (1) failures of the paper-and-pencil interviewers to accurately record the number, timings, and outcomes of calls; (2) by analysis methods necessary to circumvent the loss of many paper call records; and (3) by previously mentioned uncontrolled factors in this comparative study. Nevertheless, the direction of the results is consistent with the summary impressions of agencies which conduct CATI surveys.

Interviewer productivity is only one component of total CATI interviewing costs. The initial training of CATI interviewers is often believed to take longer since they must learn to operate a computer terminal or microcomputer. CATI's consequences for interviewing supervision are less clear and may depend on the tasks that supervisors are assigned. Initial supervisory training may require more time for CATI, but a system with online call scheduling, case assignment, and automatic record keeping should free the supervisors of most clerical and report preparation tasks and eliminate the need for clerical support. This may provide more time for direct supervision and monitoring of interviewers.

To date, only one study has attempted to include these elements in cost comparisons of CATI and non-CATI data collection. Table 3 presents cost projections Ferrari (1986) based on comparative data from the Census of Agriculture but adjusted for the differing pay rates and supervisory practices at the CATI and non-CATI sites. Data entry salaries also are included since CATI data entry occurs simultaneously with interviewing. Searching telephone directories for respondents' numbers when unknown and agricultural analyst review of completed interviews for content consistency and completeness were assumed to require the same cost per case in both methods.

TABLE 3

PROJECTED INTERVIEWING AND KEYING SALARY COSTS PER CASE: CENSUS OF AGRICULTURE*

Activity	CATI	Non-CATI
Interviewer training...	$1.09	$ .39
Interviewing...........	2.48	2.07
Interviewer supervision	.77	.44
Clerical support.......	--	.18
Tel. number research...	.11	.11
Data keying............	--	1.26
Analyst review.........	.08	.08
	-----	-----
Total per case....	$4.53	$4.53
Total per complete	$8.83	$9.79

*Ferrari, 1986.

In Ferrari's analysis, CATI has higher costs per assigned case in three areas: interviewer training, interviewing, and interviewer supervision. The higher supervisory costs include both added training for the supervisors and a higher supervisor-to-employee ratio than used previously in this survey. At the same time, CATI achieves savings by eliminating clerical staff and data keying. When summed, total salary costs for CATI and paper methods are equal per *assigned* case. However, since CATI obtained a higher response rate in this study, CATI's *total* salary costs per *completed* interview were less, by 11 percent.

These projections are based on the use of CATI for telephone follow-up to mail nonresponse and may not apply to other applications or organizational settings. However, the analysis suggests the types of data required to begin assessments of CATI's cost-effectiveness as a data collection method. Future analyses should include the salaries of professional and technical staff in survey design and in processing as well as nonsalary costs, such as amortization of CATI and key entry hardware and duplication of paper forms.

5. Data Quality

One of the most common speculations of the small literature on CATI is that CATI will improve the quality of data collected in telephone surveys (Groves, 1983; House, 1984; Nelson et al., 1972; Nicholls, 1978; Rustemeyer et al., 1978; Shanks, 1983). Others have suggested ways in which CATI may lower data quality (Harlow et al., 1985; Presser, 1983). We will look at available evidence in four areas: (1) unit nonresponse, (2) item nonresponse, (3) data consistency, and (4) the recording of textual material.

5.1 Unit Nonresponse

CATI can affect survey nonresponse through the interviewers' and respondents' reactions to this new medium or through its special features, such as online call scheduling and case assignment. The small but consistent literature on interviewer and respondent reactions suggests: (1) that interviewers either prefer CATI to paper-and-pencil methods or are about evenly divided in their preferences between these two modes; and (2) that respondents accept CATI as well as other forms of telephone interviewing or are unaware of the interviewing mode employed. (Coulter, 1985; Groves and Mathiowetz, 1984; Morton and House, 1984; Nicholls, 1978.) While these reactions may change with time or special circumstances, both interviewers and respondents appear to regard CATI as an acceptable method of telephone data collection.

There is little reason to anticipate, therefore, that interviewers' or respondents' reactions will affect survey response rates. Coleman (1985) reports identical response rates for CATI and non-CATI treatments in the Malignant Melanoma Survey. Similarly, Groves and Mathiowetz (1984) found nearly identical reponse rates in two of three replicates of the RDD Health Survey Test. A statistically significant difference occurred only in the first replicate, during a period when the CATI system could not maintain acceptable response times between questions. Under these circumstances, interviewer or respondent reactions to CATI may have lowered the response rate.

Other investigators have focused specifically on refusal rates, that is the percent of contacted households who refuse an interview. Of the five experimental and comparative studies summarized in Table 4, four found no difference between CATI and non-CATI refusal rates. In the Census of Agriculture, where telephoning was used for follow-up to mail nonresponse, the CATI staff obtained a significantly lower refusal rate. This may be the result of the additional calls and time on the telephone made possible for the CATI staff by automatic call scheduling and case assignment. The CATI staff placed 76 percent more calls and averaged twice as much time on the phone with cases finally classified as refusals. The difference also may be attributable to the previously described uncontrolled factors in this comparative study.

TABLE 4

REFUSAL RATES BY MODE IN FIVE SURVEYS

Survey	CATI	Non-CATI
Colo-Rectal Cancer Survey (Harlow et al., 1985).....	8%	8%
Nebraska Hog Survey (Coulter, 1985)...........	9%	8%
Cattle Inventory Survey (House, 1984)............	8%	8%
Survey of Scientists and Engineers (Ferrari, 1984).	21.7%	21.2%
Census of Agriculture (Ferrari, 1986)...........	5.2%*	12.5%

*Statistically significant difference between CATI and non-CATI at the .05 level.

A second major component of unit nonresponse is failure to reach sampled respondents because their telephone numbers cannot be found or their numbers are not answered when called. Table 5 compares CATI and non-CATI contact rates, the percent of assigned cases whose households were reached, whether that contact resulted in an interview, a refusal, other noninterview or a determination of ineligibility. In telephone follow-up to mail nonresponse in the Survey of Scientists and Engineers and in the Census of Agriculture, current telephone numbers and

TABLE 5

CONTACT RATES BY MODE IN THREE SURVEYS

Survey	CATI	Non-CATI
Survey of Scientists and Engineers (Ferrari, 1984)...	50.2%*	44.4%
Census of Agriculture (Ferrari, 1986)............	84.3%*	79.0%
Cattle Dual Frame Survey (House, 1984)..............	72%*	57%

*Statistically significant difference between CATI and non-CATI at the .05 level.

addresses frequently were not available; and the interviewing assignments included tracing such respondents through directory assistance and other sources. The CATI system included these tracing steps in its call scheduling and case assignment procedures. The paper-and-pencil interviewers were given guidelines with the same procedures but independently selected cases to call within batched assignments. In both surveys, the CATI telephone follow-up staff obtained a significantly higher contact rate than the non-CATI staff. While online tracing and call scheduling appears to have produced this result, uncontrolled factors in these comparative studies also may have contributed.

House (1984) reports a similar result for a CATI system without online call scheduling. In this survey, however, CATI stations were in short supply, and to make maximum use of the available equipment, a supervisor stood behind the four CATI interviewers and chose cases for them to call. The non-CATI interviewers worked independently. Online call scheduling may be viewed as automation of such manual supervisory support.

5.2 Item Nonresponse

Item missing data arise both from interviewer failure to ask questions or enter responses and from respondent failure to provide substantive answers. One of the most frequently cited advantages of CATI is rigid control over question flow and recording of responses, forcing the interviewer through each question appropriate to the respondent and requiring an entry at each question displayed. In principle, this feature can eliminate errors from interviewers inadvertently or intentionally skipping items. While it is possible to prevent interviewers from entering "don't know" responses by limiting acceptable entries, in practice interviewers generally are permitted to enter "refused" or "don't know" to any question, just as in paper-and-pencil interviewing. Forced entry at each question does not ensure recording of a substantively meaningful value.

Three studies comparing item nonresponse by mode are summarized in Table 6. Groves and Mathiowetz (1984) found the same levels of item nonresponse for CATI and paper interviews on six demographic and income items. Fielder (1985) reports smaller levels of item nonresponse for CATI, although none of the differences reach common levels of statistical significance. In these two studies, which employed the telephone as the primary data collection mode, CATI seems to have little or no effect on item nonresponse.

The consequences of CATI for item nonresponse, however, may depend on the content of the items, the design of the questionnaire in each mode, and the training and supervision of the interviewers. In cases where the number of items asked is partly left to the interviewers' judgment, CATI may produce large reductions in item nonresponse. This can occur in telephone follow-up to mail nonresponse where the interviewer must reconcile the conflicting goals of obtaining as much information as possible and not antagonizing possibly reluctant respondents. The first test of the U.S. Census Bureau's CATI system for the Survey of Scientists and Engineers was this type of application. As shown in Table 6, the CATI staff obtained substantially lower rates of item nonresponse than the comparison paper-and-pencil staff. The greater difficulty of omitting applicable questions in CATI apparently contributed to this difference, but since the results are not based on a fully controlled experimental design they must remain only suggestive.

TABLE 6

MEAN PERCENT ITEM NONRESPONSE BY TOPIC AND MODE IN ONE EXPERIMENTAL AND TWO COMPARATIVE STUDIES

Survey and Topic	CATI	Non-CATI
Health Survey Test (Groves and Mathiowetz, 1984)		
Sex, age, education.....	0.7%	0.9%
Race, income, marital...	14.6%	15.0%
Sample size............	(942)	(1,137)
SRC-UCLA Earthquake Survey (Fielder, 1985)		
15 Demographic items....	0.3%	0.5%
11 Opinion items........	2.3%	5.3%
Sample size............	(536)	(516)
Survey of Scientists and Engineers (Ferrari, 1984)		
14 Items asked of all respondents**........	7.1%*	24.6%
12 Items asked of most respondents**.......	7.5%*	26.6%
Sample size............	(3,056)	(16,159)

*Statistically significant difference between CATI and non-CATI at the .05 level.

**Excludes imputed items and those constructed during post-interview computer edits.

5.3 Data Consistency

The data for a case may be described as "consistent" if the entered values do not contradict one another. Consistency may be limited to the responses to one interview or extended across successive interviews and to prior information from records and other sources. Paper-and-pencil methods strive for data consistency by asking the interviewer to probe obviously inconsistent responses, by supervisory or clerical review of completed forms, by batch computer editing after the interview is keyed, and by reinterviewing cases which fail these edits. Data consistency ensures neither validity nor reliability but is generally regarded as a useful measure of data quality.

Computer-assisted telephone interviewing may contribute to data consistency in two primary ways. The first is through automatic branching between items to ensure that all applicable questions are asked, or at least displayed for the interviewer. (Inapplicable questions are omitted or appropriately marked if previously asked and later found to be inapplicable.) The second is through online editing in which apparently inconsistent responses require additional actions by the interviewer, such as: backing to correct prior entries; or making an additional entry to explain the reason for the inconsistency.

Two studies with controlled experimental designs have provided first evidence of CATI's effects on data consistency. Groves and Mathiowetz (1984) analyzed a sequence of 28 questions with complex skip patterns and found that 8.8 percent of the entries by the paper-and-pencil interviewers had consistency errors compared with 1.8 percent of the entries by CATI interviewers. Contributions of CATI to data consistency through online editing have been reported by Tortora (1985) in the Cattle Inventory Survey. When data sets from CATI and non-CATI interviews were submitted to the same batch computer edits, the CATI data were found to contain 75 percent fewer critical errors than the paper-and-pencil data. Critical errors were defined as those requiring another contact with the sample case.

Placing a survey on CATI does not necessarily improve data consistency across all items. The effect will occur only where automatic branching and online editing enhance consistency. Improved consistency also may be difficult to detect if comparisons are made only after the data are clerically edited and imputations made for missing data and out-of-range entries. Ferrari (1986) reports only a trivially lower (although statistically significant) overall edit failure rate for CATI than for non-CATI telephone follow-up data in the Census of Agriculture when submitted to the same batch computer edits after the close of field work. Moreover, the CATI edit failure rate was significantly higher for some key items not included in the CATI online edits. For maximum gains in data consistency, online editing must be extensively employed and parallel the key requirements of the batch editing programs.

5.4 Recording of Textual Material

Computer-assisted telephone interviewing is generally viewed as most effective in obtaining numeric and precoded responses. Concerns are more frequently expressed about the quality of textual materials obtained, such as entries to open-ended questions and interviewer notes qualifying or explaining respondents' answers. Most CATI systems permit entry of extended answers and qualifying notes to any question, but their entry may be more awkward in CATI than with paper methods. CATI interviewers often are required to have minimal typing skills, at least 20 words per minute, and trained to inform respondents that they are using a keyboard when unable to keep up with the respondent's answers. Nevertheless, the slower rate of textual entry in CATI may reduce the completeness of answers to open-ended questions and discourage entry of qualifying notes. Separate actions required to access the notes function also may discourage its use.

Morton and House (1983) summarize field staff impressions on two CATI surveys which suggest that the recording of textual material is not a problem. They say: "... we found that the lack of typing speed did not seem to be an irritant to the respondent, and the speed did improve as interviewers felt more comfortable with the keyboard." No quantitative data have yet been presented to compare the quality of CATI and non-CATI responses to open-ended questions; but Harlow, et al., (1985) found that CATI interviewers entered 25 percent fewer comments than non-CATI interviewers; and this contributed to a lower rate of unresolved "don't know" responses after comments were employed in clerical editing. These differences were not statistically significant in the small samples examined, but they are supported by similar observations by Tortora (1985). Further research on these topics is clearly needed.

6. Summary

The remarkable growth of CATI in commercial market research, in university research centers, and in the planning of government agencies has proceeded largely without firm research results on its consequences for survey costs and data quality. Detailed evidence about CATI's data collection characteristics have begun to appear only in recent years.

When compared with paper-and-pencil methods, CATI typically entails higher costs in computing hardware and software and perhaps in survey design. Offsetting savings are most likely to be realized in interviewer productivity and post-interview processing. Due to the typically longer length of CATI interviews, interviewer productivity may not be increased by CATI systems without online call scheduling and case assignment. But with these capabilities, major increases in interviewer productivity seem possible.

The effects of CATI on data quality generally appear to be small or negligible except when specific data quality enhancement features are employed. CATI typically has no effect on the response rates and refusal rates of telephone interviews, but may increase contact rates in some applications with efficient online call scheduling and tracing routines. Similarly, CATI typically has little or no effect on item nonresponse except in applications where automatic branching encourages interviewers to ask questions they might otherwise omit. CATI does increase data consistency but only where its automatic branching and online editing features are used. At the same time, CATI may result in less complete entries to open-ended questions and less frequent interviewer comments.

CATI remains a promising technology for survey data collection, but like other data collection methods can be expected to have its own strengths and weaknesses. Further research is required to identify these more fully, both to guide appropriate choices of method for specific surveys and to stimulate corrective measures in areas where weaknesses are found.

References

Coleman, Charlotte M. "The Mole Survey: An Accidental Almost Methodological Study." Paper presented at the National Field Directors Conference, 1985.

Coulter, Richard. "A Comparison of CATI and NonCATI on a Nebraska Hog Survey." Staff Report No. 85. Statistical Research Division, Statistical Reporting Service, U.S. Department of Agriculture, 1985.

Dutka, Solomon and Lester Frankel. "Sequential Survey Design Through the Use of Computer-Assisted Telephone Interviewing." Proceedings of the American Statistical Association, Section on Business and Economic Statistics, 1980, 73-76.

Ferrari, Pamela W. "Preliminary Results from the Evaluation of the CATI Test for the 1982 National Survey of Natural Scientists and Engineers." Unpublished research report. U.S. Census Bureau, 1984.

Ferrari, Pamela W. "An Evaluation of Computer-Assisted Telephone Interviewing Used During the 1982 *Census of Agriculture*." Unpublished evaluation report. Agriculture Division, U.S. Census Bureau, 1986.

Fielder, Eve. Personal communication with the authors, 1985.

Fink, James C. "CATI's First Decade: The Chilton Experience." Sociological Methods and Research, Vol. 12, No. 2, 1983, 153-68.

Groves, Robert M. "Implications of CATI: Costs, Errors, and Organization of Telephone Survey Research." Sociological Methods and Research, Vol. 12, No. 2, 1983, 199-215.

Groves, Robert M. and Nancy A. Mathiowetz. "Computer-Assisted Telephone Interviewing: Effects on Interviewers and Respondents." Public Opinion Quarterly, Vol. 48, 1984, 356-69.

Harlow, Bernard L., Jeanne F. Rosenthal, and Regina G. Ziegler. "A Comparison of Computer-Assisted and Hard Copy Telephone Interviewing." American Journal of Epidemiology, Vol. 122, No. 2, 1985, 335-40.

House, Carol C. "Computer-Assisted Telephone Interviewing on the Cattle Multiple Frame Survey." Staff Report No. 82, Statistical Research Division, Statistical Reporting Service, U.S. Department of Agriculture, 1984.

Morton, Betsy and Carol C. House. "Training Interviewers for Computer-Assisted Telephone Interviewing," Proceedings of the American Statistical Association, Section on Survey Research Methods, 1983, 129-33.

Nelson, Raymond O., Boyd L. Peyton and Bruce Z. Bortner. "Use of an On-Line Interactive System: Its Effects on the Speed, Accuracy and Cost of Survey Results." Paper given at the 18th ARF Conference, New York City, 1972.

Nicholls, William L., II. "Experiences with CATI in a Large-Scale Survey." Proceedings of the American Statistical Association, Section on Survey Research Methods, 1978, 9-17.

Nicholls, William L., II. "Development of CATI at the U.S. Census Bureau." Proceedings of the American Statistical Association, Section on Survey Research Methods, 1983, 642-647.

Nicholls, William L., II and Robert M. Groves. "The Status of CATI." Proceedings of 45th Session, International Statistical Institute, 1985, Book III, Topic 18, 1-15.

Palit, Charles D. and Harry Sharp. "Micro-computer-Assisted Telephone Interviewing." Sociological Methods and Research, Vol. 12, No. 2, 1983, 169-89.

Philipp, Steven F. and Charles F. Cicciarella. "An Apple II Package for Computer-Assisted Telephone Interviewing." Behavior Research Methods and Instrumentation, Vol. 15, No. 4, 1983, 456-58.

Presser, Stanley. "Discussion" of Session II Contributed Papers. Proceedings of the American Statistical Association, Section on Survey Research Methods, 1983, 142-43.

Rustemeyer, Anitra, Gerald H. Shure, Miles S. Rogers, and Robert J. Meeker. "Computer-Assisted Telephone Interviewing: Design Considerations." Proceedings of the American Statistical Association, Section on Survey Research Methods, 1978, 1-8.

Shanks, J. Merrill. "The Current Status of Computer-Assisted Telephone Interviewing: Recent Progress and Future Prospects." Sociological Methods and Research, Vol. 12, No. 2, November 1983, 119-42.

Shanks, J. Merrill, George A. Lavender and William L. Nicholls II. "Continuity and Change in Computer-Assisted Surveys - Development of Berkeley SRC CATI." Proceedings of the American Statistical Association, Section on Survey Research Methods, 1980, 507-12.

Shure, Gerald H. and Robert J. Meeker. A Mini-computer System for Multiperson Computer-Assisted Telephone Interviewing." Behavior Research Methods and Instrumentation, Vol. 1, No. 2, 1978, 196-202.

Smith, Ronald and Richard Smith. "Evaluation and Enhancements of Computer Controlled Telephone Interviewing," Proceedings of the American Statistical Association, Section on Survey Research Methods, 1980, 513-15.

Tortora, Robert D. "CATI in an Agricultural Statistical Agency." Journal of Official Statistics. Vol. 1, No. 2, 1985, 301-14.

INFERENCE FROM COARSE DATA USING MULTIPLE IMPUTATION

Daniel F. Heitjan, U.C.L.A.
Donald B. Rubin, Harvard University

Inference from Coarse Data Using Multiple Imputation

Multiple imputation is a procedure for handling inadequate data by filling in several plausible values for each inadequately reported value. The basic ideas underlying multiple imputation are reviewed and then applied to a data set with coarsely reported ages of children. Sensitivity analyses and diagnostic displays are included.

1. Multiple Imputation

Pervasiveness of inadequate data

Essentially all data collected in surveys are inadequate in some aspects. Commonly for example, survey data suffer from nonresponse: some sampled units do not provide answers to some questions. Usually such nonresponse leads to missing values, but a less extreme form, perhaps called "partial nonresponse" leads to data much coarser than desired. For instance, in many surveys income questions suffer from either nonresponse because some individuals refuse to divulge their incomes, or partial nonresponse because only coarse information, such as above or below $20,000 is reported.

Imputation

Imputation is the process of filling in each missing value with a specific value, such as the respondents' mean for that variable or a value predicted from variables that are observed for that unit. For example, if income is missing for a group of individuals, it might be imputed using predictions based on a regression of log income on fully observed background characteristics using those individuals who reported income. With partial nonresponse, such as coarsely reported income, a specific value for income consistent with the coarsely reported value would be imputed.

Advantages and disadvantages of imputation

The practice of imputing for missing values is very common because it has the obvious practical advantage of allowing standard complete-data methods of analysis to be used. This advantage is extremely important not only when forming inferences but also when conducting diagnostic analyses. Imputation also has an advantage in many contexts in which the data collector (e.g., the Census Bureau) and the data analyst (e.g., a university social scientist) are different individuals, because the data collector may have access to more and better information about nonrespondents than the data analyst. For example, in some cases, information protected by confidentiality constraints such as zip codes of dwelling units, may be available to help impute missing values such as annual incomes. The obvious disadvantage of single imputation is that imputing a single value treats that value as known. Consequently, without special adjustments, single imputation cannot reflect sampling variability about the correct value to impute even supposing the reasons for nonresponse are known, nor can single imputation represent the additional uncertainty that arises when the reasons for nonresponse are not known.

Multiple imputation

Multiple imputation, in contrast to single imputation, replaces each missing value not with a single value but with a vector of $M \geq 2$ imputed values. The M values are ordered in the sense that M completed-data sets are created from the vectors of imputations: replacing each missing value by the first component in its vector of imputations creates the first completed data set, and so on. Standard complete-data methods are used to analyze each data set. That is, standard complete-data methods of inference and diagnosis are used on each completed data set. When the M sets of imputations are repeated random draws under one model for nonresponse, with each set corresponding to an independent drawing of the parameters and missing values from their posterior predictive distribution, the M complete-data inferences can be combined to form one inference that properly reflects uncertainty due to nonresponse under that model. When the imputations are drawn from two or more models for nonresponse, the combined inferences under the models can be contrasted across models to display sensitivity of inference to models for nonresponse, a particularly critical activity when the precise reasons for nonresponse are unknown. Thus multiple imputation has the advantages of single imputation but rectifies both disadvantages. The only disadvantage of multiple imputation over single imputation is that it takes more work to create the imputations and analyze the results. The extra work in analyzing the data, however, is really quite modest in today's computing environments since it basically involves performing the same task M times instead of once.

Multiple imputation was first proposed in Rubin (1978). A comprehensive treatment is given in Rubin (1986a); other easily accessible references include Rubin (1986b), Herzog and Rubin (1983), and Rubin and Schenker (1986).

Forming summary inferences from a multiply imputed data set

Forming summary inferences from a multiply imputed data set is quite direct. First, each data set completed by imputation is analyzed using the same complete-data method that would be used in the absence of nonresponse. Let $\hat{\theta}_\ell$, U_ℓ, $\ell = 1, \ldots, M$ be M complete-data estimates and their associated variances for an estimated θ, calculated from M repeated imputations under one model. For instance, when estimating a proportion θ from a simple random sample of size n, $\hat{\theta}_\ell$ is given by p_ℓ, the proportion of successes calculated using ℓth set of imputed values for the missing values, and U_ℓ is given by $p_\ell(1 - p_\ell)/n$, at least for modestly large n and p_ℓ not too near 0 or 1. The combined estimate is

$$\bar{\theta}_M = \sum_{\ell=1}^{M} \hat{\theta}_\ell / M.$$

The variability associated with this estimate has two components: the average within-imputation variance,

$$\bar{U}_M = \sum_{\ell=1}^{M} \hat{U}_\ell/M,$$

and the between-imputation component,

$$B_M = \Sigma(\hat{\theta}_\ell - \bar{\theta}_M)^2/(M-1)$$

(with row vector θ, $(\cdot)^2$ is replaced by $(\cdot)^T(\cdot)$.) The total variability associated with $(\theta_M - \theta)$ is

$$T_M = \bar{U}_M + \frac{M+1}{M} B_M,$$

where $\frac{M+1}{M}$ is an adjustment for small M. With scalar θ and small M, the reference distribution for interval estimates and significance tests is a t distribution,

$$(\theta - \bar{\theta}_M)\, T_M^{-\frac{1}{2}} \sim t_\nu,$$

where the degrees of freedom,

$$\nu = (M-1)\left[1 + \frac{M}{M+1}\,(\bar{U}_M/B_M)\right]^2,$$

is based on a Satterthwaite approximation (Rubin and Schenker, 1986). When M is large, the inference for θ is based on the normal approximation

$$(\theta - \bar{\theta}_M)\, T_M^{-\frac{1}{2}} \sim N(0, 1).$$

For θ with r components, significance levels for null values of θ can be obtained from M repeated complete-data estimates, $\hat{\theta}_\ell$, and variance-covariance matrices, $\bar{U}_\ell$, using *multivariate* analogues of the above expressions. Less precise p-values can be obtained directly from M repeated complete-data significance levels. Details may be found in Rubin (1986a).

Although multiple imputation is most directly motivated from the Bayesian perspective, the resultant inference can be shown to possess good sampling properties. For example, Rubin and Schenker (1986) show that in many cases interval estimates created using only two imputations provide randomization-based coverages close to their nominal levels.

Missing information

The ratio $\bar{U}_M/B_M$ estimates the quantity $(1-\gamma)/\gamma$ where γ is the fraction of information about θ missing due to nonresponse. This fraction is important in two ways. First, it affects the adequacy of the distributional approximations proposed above. Second, γ governs the efficiency of θ_M as an estimator of θ; specifically, the variance of $\bar{\theta}_M$ is larger than the variance of $\bar{\theta}_\infty$ by the factor

$$\left(1 + \frac{\gamma}{M}\right).$$

Beyond one summary inference from a multiply-imputed data set

Although the ability to form one summary inference when the multiple imputations are repeated draws from the posterior predictive distributions of the missing values is important, equally important is the fact that the creation of complete data sets allows (i) the use of standard diagnostic techniques to help criticize posited models and (ii) the assessment of sensitivity of inference to various models. These points will be illustrated here using a particular survey of nutritional status of children in Tanzania that suffers from coarse reporting of ages. Far more comprehensive presentations of the data and these analyses are given in Heitjan (1985) and Heitjan and Rubin (1986).

2. The Tanzania Nutrition Data

The data base

The data set that we use to illustrate multiple imputation for coarse data consists of anthropometric measurements on children under six years of age from eight poor rural areas in Tanzania taken by nutrition researchers interested in estimating the extent of malnutrition in the various regions (Kimati, 1985). Approximately five thousand children comprise the full data base; we focus on the 270 children from the Dodoma region. In addition to sex of child and age, as provided by the mother, weight, height, mid-arm circumference and head circumference were recorded by the researchers.

Objective of complete-data analysis

A simple way to measure the extent of malnutrition among such a group of children is to calculate the percentage of them that are classified as stunted or not and wasted or not. Stunted refers to being short for age and wasted refers to being light for height, where the definitions are established for boys and girls from data collected on groups of known normal healthy children from the U.S. (Hamill, et al., 1979). Children who are classified as stunted but not wasted may simply be short, and children who are classified as wasted but not stunted may simply be thin, but simultaneous stuntedness and wastedness in a single child are regarded as clear evidence of malnourishment. Assuming accurate measurements of sex, age, height and weight in the 270 children, inference for the extent of malnourishment would be based on p, the proportion of stunted and wasted children among the 270, and its standard error, SE = $[p(1-p)/n]^{\frac{1}{2}}$ where n = 270. The data as reported give p = 5.9% and SE = 1.4%.

Age-heaping

The problem with this simple answer is that, even though height and weight are accurate measurements, age as reported by mothers is quite coarse. For ages over a year, most ages in months are reported as divisible by 6 -- this phenomenon is common and is known as age heaping (e.g., see Ewbank, 1981). The problem is possibly more serious in Tanzania than the United States in the sense that precise date of birth is not very important in Tanzania, and so mothers may not even know their children's ages to the nearest month. Also, some evidence suggests that, as opposed to the situation in the United States where reported ages are typically truncated, reported ages may often be rounded to the nearest year or six months. Figure 1 displays the reported ages for the 270 children.

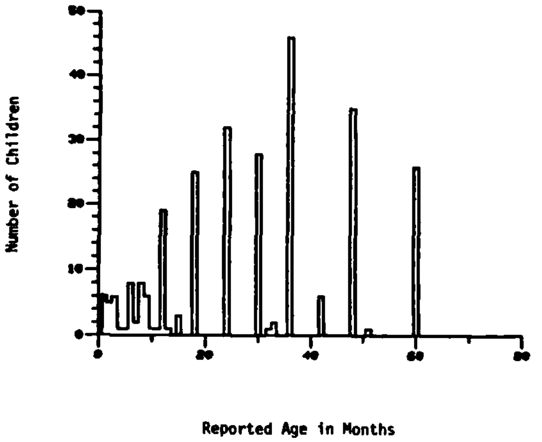


FIGURE 1: *Reported Age Histogram -- Dodoma*

Restrictions on true age imposed by reported age

We consider two versions of the possible intervals for true age given reported age, where for both versions, all ages reported as under 12 months except six month reporters, are correct to the nearest month. With "medium" intervals, all ages that are reported to be a full year (i.e., reported age in months = 0 mod 12) are considered to arise from possible true ages within ±6 months of the reported age, and all ages that are reported to be a mid-year (i.e., age in months = 6 mod 12) are considered to arise from possible true ages within ±3 months of the reported age. With "wide" intervals, the bounds are twice as large, that is, ±12 months for full-year reporters, and ±6 months for mid-year reporters. We label the width factor $W = 0$ (medium) and 1 (wide). The status of the six month reporters is determined by another factor $R = 0$ (rounded) and 1 (exact). If $R = 1$, a reported age of six months is treated as correct to the nearest month, whereas if $R = 0$, a reported age of six is treated as rounded, within ±3 months if $W = 0$, and within ±6 months if $W = 1$. It is important to realize that at this point, absolutely no assumption regarding the distribution of age within these intervals is being made.

Extreme tables imputation

Of course, it is perfectly possible with these definitions of possible true ages that the reported age data are accurate enough for purposes of inference about the proportion of malnourished children, even if no further assumptions are made. Specifically, since height, weight and sex are reported accurately, the wasted classification is accurate and the height component of the stunted classification is accurate. Perhaps no matter what the childrens' true ages within these rounding intervals, inferences for the proportion malnourished will be stable.

If all children are considered as young as possible given their reported ages, the resulting proportion of the 270 that will be classified as malnourished, p_{opt}, will be as small as possible -- the subscript opt is for optimistic. Similarly, if all children are considered as old as possible given their reported ages, the resulting proportion of the 270 that will be classified as malnourished, p_{pes}, will be as large as possible -- pes is for pesimistic. In other words, if we impute true ages that are as young as possible, we obtain the estimate p_{opt} with associated standard error $SE_{opt} = [p_{opt}(1 - p_{opt})/n]^{\frac{1}{2}}$. Table 1 gives the results, and indicates substantial sensitivity of answers, especially considering that each 1% represents many hundreds of children.

TABLE 1. *Extreme values for proportion malnourished*

	estimate (standard error)	
	Interval widths	
	medium	wide
pessimistic ages (overstate malnutrition)	6.7% (1.5)	8.2% (1.7)
optimistic ages (understate malnutrition)	3.3% (1.1)	1.5% (0.7)

Plan of attack

Since the extreme tables analysis suggests that the coarseness of the reported ages does have an important practical effect on inferences about the proportion malnourished, π, we proceed to perform more sophisticated statistical analyses specifically designed to take the coarseness into account. In particular, our plan is to use multiple imputation to create a sequence of data sets with various values for true ages from which the standard complete-data inference for π can be calculated. These imputations will be created using a variety of Bayesian models that relate true age Y to reported age X and the other reported characteristics Z = sex, height, weight, mid-arm circumference and head circumference. The analyses of the imputed data sets within each model are combined to form a valid inference under that model, and then these inferences are contrasted across models to display sensitivity of inference to modelling assumptions. Furthermore, the data sets completed by imputation are used to help diagnose the adequacy of the underlying model.

3. Models for True Age and Resulting Inferences and Diagnostics

A naive and obviously incorrect model

A naive first pass model corresponds to assuming that true ages Y are uniformly distributed within the allowable intervals defined by the value of reported age X and the levels of the factors W and R. Imputed true ages are created by drawing from these uniform distributions. This model is obviously incorrect given the existence of background variables Z because it implies, for example, that two children with the same sex and reported age, of say 36 months, have the same distribution of true ages despite the fact that one is taller, heavier, and has larger head and mid-arm circumferences than the other. Clearly, the bigger child is probably older than the smaller child, and the imputed true ages should reflect this fact.

Notation

Let g be the generally unobserved indicator for the degree of rounding; g takes on three values indicating: reports to the nearest month (0), reports to the nearest mid-year (1), reports to the nearest year (2). This indicator is observed as 0 when reported age is neither a mid-year or full-year ($X \neq 0 \bmod 6$), but is either 0 or 1 when age is reported as a mid-year ($X = 6 \bmod 12$) since a child with reported mid-year age might be that age to the nearest month and would have reported to the nearest month no matter what his true age. Similarly, g is either 0, 1 or 2 when age is reported as a full year ($X = 0 \bmod 12$). The statistical problem is to model the joint conditional distribution of g and Y given Z = (sex, weight, height, mid-arm circumference, head circumference). Reported age, X, is a fixed function of Y and g, and so its conditional distribution given (g, Y, Z) is fully specified a priori.

The regression of Y on Z

The joint conditional distribution of (g, Y) given Z can be defined by first specifying the conditional distribution of Y given Z and then the conditional distribution of g given (Y, Z). We assume a standard normal linear regression for Y^s given Z where the exponent or scale s is either 0 or ½. Thus the factor S has two levels, 0 = raw scale, 1 = square root scale.

The final specification for g

One class of models that we fit as a baseline assumes g is fixed and known from the value of reported age X. In particular, if reported age is a mid-year, then g is fixed at 1, and if reported age is a full year, then g is fixed at 2, otherwise, g is 0. Thus, all children with mid-year reported ages are regarded as always being mid-year reporters, and all children with full-year reported ages are regarded as always being full-year reporters. This model is intuitively not very satisfying and in fact does not fit in well with our stated objective to specify the conditional distribution of g given (Y, Z), but it is relatively easy to fit. The second class of models treats g as a random variable and posits a proper joint conditional distribution of g given (Y, Z), basically of a probit form -- g is created by trichotomizing an unobserved normal. Heitjan and Rubin (1986) provide details. The factor indicating whether the intervals are fixed or random is I with levels 0 for "g is fixed from X" and 1 for "g is a random variable."

Creation of imputations

Under each of the 2^4 models being considered, posterior distributions were estimated for the regression parameters under a noninformative prior distribution. An EM algorithm (Dempster, Laird and Rubin, 1977) was used in conjunction with Newton's method to find the mode, and then the posterior distribution was approximated as normal with mean equal to the mode and variance provided by the second derivative at the model. A regression parameter was then drawn from this posterior distribution, say ϕ_*, and conditional on $\phi = \phi_*$, values of true ages were independently imputed by drawing from the confined distribution of true age given X and Z for each of the 270 children. This process was effectively repeated independently hundreds of times for each of the 2^4 models to create hundreds of data sets with known nutritional status. Each data set was summarized by the standard complete-data statistics p and $p(1-p)/n$ and these were combined within models according to the methods of Section 1 to create 2^4 summary inferences about π. Results are summarized in Table 2 which gives the estimate p, its standard error SE, and the fraction of information missing about π due to coarse rather than precise age data.

TABLE 2: *Sensitivity of Inference for π across 16 models*

Model				Estimate of π		Standard Error of (π - p)		Fraction of information lost for π	
Factors									
I	S	W	R	$\hat{p}$	Effect	$T_M^{1/2}$	Effect	γ	Effect
0	0	0	0	5.64%	5.54%	1.49%	1.50%	10.8%	14.3
0	0	0	1	5.64	.01	1.49	.00	10.9	- 0.1
0	0	1	0	5.39	- .15	1.54	.02	19.5	4.4
0	0	1	1	5.39	.00	1.54	.00	19.4	- 0.1
0	1	0	0	5.46	- .15	1.47	- .01	11.5	1.2
0	1	0	1	5.45	.00	1.47	.00	12.2	0.0
0	1	1	0	4.76	.09	1.50	- .01	25.4	0.7
0	1	1	1	4.75	.00	1.49	.00	25.5	0.0
1	0	0	0	5.07	.23	1.49	.00	8.2	- 2.6
1	0	0	1	5.08	.01	1.49	.00	7.8	- 0.2
1	0	1	0	5.03	.08	1.54	.00	14.5	- 1.1
1	0	1	1	5.07	.00	1.54	.00	13.9	0.0
1	1	0	0	5.00	.06	1.49	.00	9.0	- 0.6
1	1	0	1	5.02	.00	1.49	.00	8.6	- 0.1
1	1	1	0	5.52	.03	1.5	.00	16.3	- 0.5
1	1	1	1	5.50	.08	1.52	.00	15.5	0.0

Discussion of Table 2 - sensitivity analyses

Table 2 also gives the Yates' 2^4 ANOVA decomposition into effects from the four factors in the design (Daniel, 1976). With respect to the point estimate p, factors I, S and W have fairly large main effects, but none is bigger than one-third of the associated standard errors, which are basically unaffected by the various models. The fractions of missing information are quite variable: all main effects except R are nonnegligible, and the IW interaction is also present. It is rather obvious that wider intervals should lead to larger information loss but it is interesting to see that the variable intervals models also lead to less information loss. Actually, after some thought, this is not surprising since for each child the possible interval under the random intervals model can be narrower and is never wider than under the corresponding fixed intervals model. For all models, however, the information losses are such that far fewer than 100 imputations under each model (say ten) would have provided essentially the same inference as an infinite number of imputations.

Diagnostic checks

One of the benefits of multiple imputation is the ability to draw valid inferences, such as summarized in Table 2, under a variety of models using standard complete-data statistics. Another advantage accruing from the creation of complete-data sets using multiple imputation is the ability to use standard complete-data diagnostic techniques. For instance, residual plots using imputed data are displayed in Figure 2 for models {I = fixed, S = raw, W = medium, R = rounded} and {I = fixed, S = square root, W = wide, R = rounded}. Such displays consistently support the conclusion that the square root scale is superior to the raw scale. Also, average (across imputations within a model) histograms of five imputed true ages were produced in Figure 3 for models {I = fixed, S = raw, W = medium, R = rounded} and {I = variable, S = square root, W = medium, R = rounded}. Such displays consistently support the random interval - medium width models since they did not have the objectionable underheaping at full-years and overheaping at mid-years present in the fixed and wide interval models.

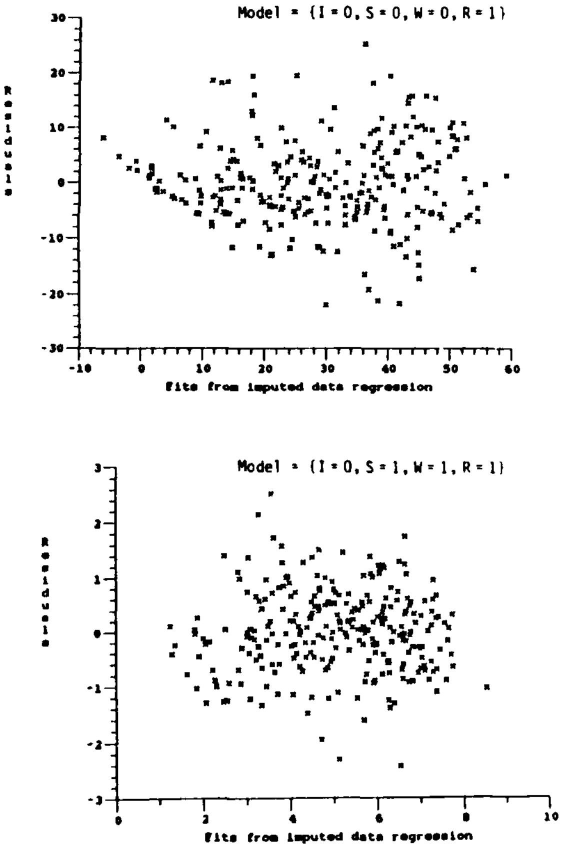


FIGURE 2: *Residual Plots*

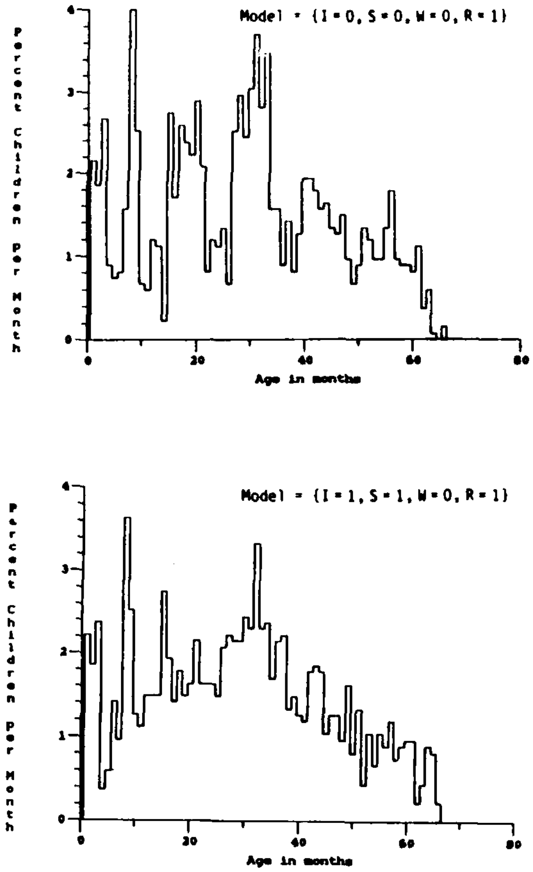


FIGURE 3: *Average Imputed Histograms, M = 5*

Summary

In summary, the procedure of producing multiple imputations under a variety of models generated the following conclusions.

1. The square-root scale model with medium, variable intervals was preferred on the basis of diagnostic displays.
2. The variable intervals models led to less information loss.
3. The point estimate of fraction malnourished was relatively insensitive to reasonable model specifications when considered as a fraction of its standard error.

For further research purposes, a multiply imputed data set is available with five repeated draws of true ages from the preferred model {I = variable, S = square root, W = medium, R = rounded}. From Table 2, the information loss for π with this model is approximately 8.6%, which means that estimates based on the five imputed values have 1.7% more variance than those based on an infinite number of imputed values, and the number of degrees of freedom in reference distributions is unaffected by the finiteness of the number of imputations.

References

DANIEL, C. (1976). *Applications of Statistics to Industrial Experimentation*, New York: Wiley.

DEMPSTER, A. P., LAIRD, N. and RUBIN, D. B. (1977). "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society, Series B* 39, 1-38.

EWBANK, D. C. (1981). *Age Misreporting and Age-Selective Underenumeration: Sources, Patterns, and Consequences for Demographic Analysis*, Washington, D.C.: National Academy Press.

HAMILL, P. V., et al. (1979). "Physical growth: National Center for Health statistics percentiles," *The American Journal of Clinical Nutrition* 32, 607-629.

HASSELBLAD, V., STEAD, A. G. and GALKE, W. (1980). "Analysis of coarsely grouped data from the lognormal distribution," *Journal of the American Statistical Association* 75, 771-778.

HEITJAN, D. F. (1985). *Analysis of a Set of Coarsely Grouped Data*, Ph.D. dissertation, The University of Chicago.

HEITJAN, D. F. and RUBIN, D. B. (1986). "Inference from coarse data via multiple imputation: age heaping in a third world nutrition study," submitted for publication.

HERZOG, T. N. and RUBIN, D. B. (1983). "Using multiple imputations to handle nonresponse in sample surveys," in *Incomplete Data in Sample Surveys, Volume 2*, edited by W. G. Madow, H. Nisselson, I. Olkin and D. Rubin, New York: Academic Press.

KIMATI, V. P. (1985). "The Nutritional Status of Tanzanian School Children -- a cross-sectional anthropometric community survey report," *East African Medical Journal* 62, 105-117.

RUBIN, D. B. (1978). "Multiple imputations in sample surveys -- a phenomenological Bayesian approach to nonresponse," section on Survey Research Methods, *American Statistical Association.*

RUBIN, D. B. (1986a). *Multiple Imputation for Nonresponse in Sample Surveys and Censuses*, New York: John Wiley and Sons, to appear.

RUBIN, D. B. (1986b). "Statistical matching using file concatenation with adjusted weights and multiple imputations," *Journal of Business and Economic Statistics* 4, 87-94.

RUBIN, D. B. and SCHENKER, N. 1986). "Multiple imputation for interval estimation from simple random samples with ignorable nonresponse," to appear in *Journal of the American Statistical Association.*

MANAGING THE DATA ANALYSIS ENVIRONMENT

Organizer: *Paula J. Cowley, Pacific Northwest Laboratory*

Experiences with a Data Analysis Management Prototype
Paula J. Cowley, Daniel B. Carr, Wesley L. Nicholson, Pacific Northwest Laboratory

Tools for Data Analysis Management
Ronald A. Thisted, University of Chicago

Statistically Sophisticated Software and DINDE
R.W. Oldford, S.C. Peters, Massachusetts Institute of Technology

EXPERIENCES WITH A DATA ANALYSIS MANAGEMENT PROTOTYPE

Paula J. Cowley, Daniel B. Carr, and Wesley L. Nicholson
Pacific Northwest Laboratory *

1. ABSTRACT

We have been evaluating our prototype data analysis management system, which was designed to aid the analyst in keeping track of the course of a data analysis. This paper describes some of our experiences using the prototype and summarizes our evaluation. Evaluated features include capabilities to graphically depict the course of the analysis, the ability to return to previous milestones in the analysis, the ability to use segments of the log that describe the course of the analysis, and the ability to associate both written and spoken documentation with milestones of the analysis.

2. INTRODUCTION

The Analysis of Large Data Sets (ALDS) Project of the Pacific Northwest Laboratory, operated by Battelle Memorial Institute, has implemented a prototype data analysis management system named ADAM [3-6]. ADAM is currently running on a DEC VAX 11/780 using AT&T Bell Laboratories' S statistical analysis system [1]. ADAM is implemented as a function within S, which means that the analyst can invoke ADAM from S at any time during the course of the analysis.

ADAM was designed and implemented to evaluate how software could be used to help the analyst track the course of the analysis. A team of statisticians and computer scientists considered the data analysis process and the way data analysts interact with the available tools. While the quantity and quality of tools to support data analysis are improving, there has been very little to aid the analyst in keeping track of what was really going on during the analysis. ADAM was designed to address the shortcomings of existing software in the area of data analysis management.

3. THE ADAM ENVIRONMENT

ADAM runs in a relatively unique environment on our DEC VAX 11/780. The S package is normally run using the UNIX operating system, but since we had such a large investment in software developed under the native VAX operating system, VMS, we did not convert to UNIX but are running EUNICE, a UNIX derivative that allows UNIX software to run on a VAX using VMS.

As will be described in more detail below, ADAM is graphics-oriented. We wanted to take advantage of an extensive graphics library developed in-house. Rather than converting the library to run in the EUNICE environment, our computer scientists developed a strategy that allows an S process running under EUNICE to create a VMS subprocess that handles the graphics portions of ADAM. Borrowing a phrase from science fiction, the computer scientists refer to this technique as going through a "wormhole," a wormhole being a way of moving between parallel universes.

A standard feature of S is the diary function. While using S, the analyst may turn on the diary so that all commands given to S will be recorded in a file. This file forms a temporal record of the course of the analysis. We have modified the S diary so that it records additional information for ADAM's use.

Use of ADAM is optional. When the user invokes S, start-up procedures will ask whether data analysis management is to be used. If the user indicates that it will be used, the modified diary is opened and date and time stamps are inserted. When the user wants to access ADAM to perform data analysis management functions, ADAM is invoked by entering

```
?adam
```

4. DEPICTING THE ANALYSIS

When we examined the data analysis process, we noted that analysts could identify significant milestones in the analysis. These milestone could be points at which some significant discovery was made, points marking the completion of a phase of the analysis, a dead end from which no more analysis would be performed, or points at which there were several alternative paths to be investigated. We also noted that these milestones could be logically linked together to form a tree. Since analysts often wish to pursue alternative paths from a given point in the analysis, we wanted to provide a facility to allow the analyst to recreate the state of the analysis at that point. We named these points "save-states". ADAM depicts the course of the analysis as a tree of these save-states (see Figure 1). The save-states are depicted as labelled boxes. The lines joining the boxes represent the analysis steps that occurred in moving from one save-state to another.

A number of functions can be performed on save-states. The analyst may use the CREATE function whenever the analyst decides that a significant point has been reached in the analysis. Once an analyst

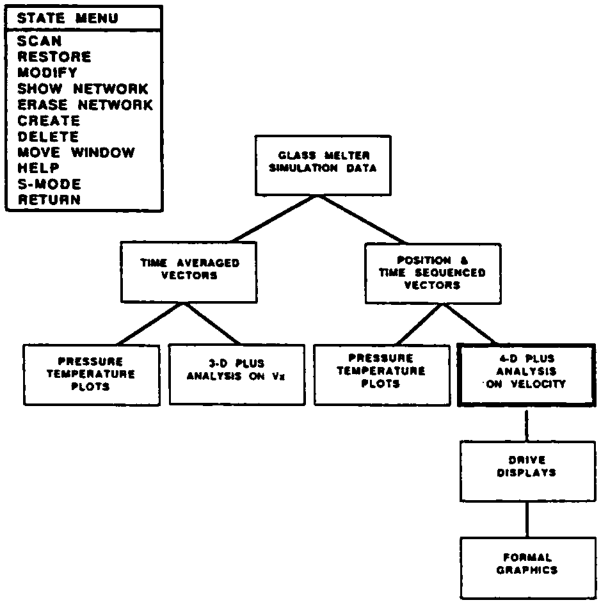
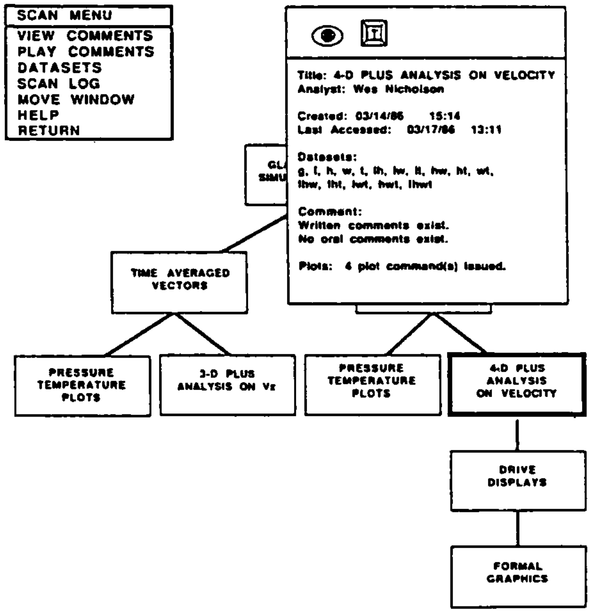


FIGURE 1: ADAM SAVE-STATES



Title: 4-D PLUS ANALYSIS ON VELOCITY
Analyst: Wes Nicholson

Created: 03/14/86 15:14
Last Accessed: 03/17/86 13:11

Datasets:
g, l, h, w, t, lh, lw, lt, hw, ht, wt,
lhw, lht, lwt, hwt, lhwt

Comment:
Written comments exist.
No oral comments exist.

Plots: 4 plot command(s) issued.

FIGURE 2: SCANNING A SAVE-STATE

has created a save-state, the analysis can continue from that point, or the analyst can chose to RESTORE a previously created save-state. In this way, an analyst can change the course of an analysis. When an analyst restores a previous save-state, the environment will be the same as when the save-state was originally created. Only the data sets that were active when that save-state was created will be available to the analyst when the save-state is restored.

In addition to depicting the relationships between the save-states, ADAM can also display more detail about a selected save-state through the SCAN command (see Figure 2). The analyst selects the save-state to be scanned and a window opens with an overview of the save-state. On the actual ADAM display, the selected save-state is a different color from the other save-states. In Figure 2, the selected save-state's box is heavily outlined. The analyst can open another window to see previously edited comments about the analysis or can listen to comments recorded on a cassette tape deck that is operated under computer control. The analyst can also scan the log (called a diary in S) to see the commands that led to the creation of the save-state being scanned.

Certain information about a restored save-state can be modified through the MODIFY command. This information includes the save-state's name; its author; icons that indicate the existence of plots (the eye icon), written comments (the keyboard icon), and spoken comments (the ear icon); an icon to indicate that special insight was gained in this portion of the analysis (the lightbulb icon); the save-state's data sets; and documentation about the save-state.

While the course of an analysis is primarily a tree, the process is not completely tree-like. At any point in the analysis, it may be useful to use a data set that was not derived as a part of the process that created the most recent save-state. To address this issue, we allow save-states to inherit data from other save-state. To reduce the clutter on the screen, these "data from" paths are not normally displayed, but the analyst can choose the SHOW NETWORK option in order to see what data has been associated with a save-state from non-ancestor save-states. The ERASE NETWORK is used to remove the "data from" arrows when the analyst is finished viewing them.

The menu window of Figure 1 shows four other menu options. The MOVE WINDOW option allows the user to reposition the various windows that ADAM uses. Every menu has a HELP option that explains the various menu options. Many menus have an S-MODE option that allows the user to leave ADAM and return to the S statistical package for further analysis. When the S-MODE option is used, the display of save-states is erased. The RETURN option is used to erase an existing menu and return to a higher-level menu. For example, the RETURN option on the SCAN menu of Figure 2 erases the large SCAN window and paints the state menu of Figure 1 on the screen.

5. USING THE LOG

We have modified the standard S diary function and refer to it as the ADAM log.

In addition to recording the analysis steps taken in S, ADAM also records data analysis management steps. Time-stamped information is recorded in the log every time a new S session is started and when a save-state is created or restored. S treats the ADAM entries as comments so the analyst can use segments of the log containing ADAM entries with no impact on S.

The ADAM log depicts the temporal sequence of the analysis while the ADAM display depicts its logical sequence. ADAM entries divide the log into segments that describe the development of a save-state from its parent save-state.

There are a number of functions the analyst can perform on the log segments. The contents of the log segments can be displayed. In addition, because of the importance of graphics in analysis, the analyst can choose to display only the plot commands. The analyst may edit log segments to remove superfluous entries or errors. The analyst may also edit the log to create procedures (called "macros" in S) that can be invoked at a later time.

6. DOCUMENTATION

One of the strongest points of ADAM is its documentation capabilities. There are three ways that an analyst using ADAM can document the course of the analysis. (1) The analyst can insert comments into the log while S is being used. This is a normal S function and can be used to provide a running commentary on the course of the analysis. (2) The analyst can edit an optional comment file associated with a save-state. (3) The analyst can record spoken comments associated with save-states. The tape deck on which the comments are recorded is under ADAM's control so that ADAM can track which selections on the tape are associated with which save-state. This option is useful for analysts who feel comfortable dictating.

Analysts can use all three documentation modes as desired. Inserting comments into the log while using S is valuable for recording the sequence of events during the analysis. The comments associated with the save-state are intended to capture information surrounding the creation of the save-state as well as discoveries and insights leading to its creation. The spoken comments can be used in the same way and could, in addition, be used to record discussions between analysts.

The documentation is useful for a number of purposes. It can provide a historical perspective on the course of an analysis. This is of particular significance if an analysis is examined later when it becomes difficult to remember exactly what was done. The documentation can be used as a basis for reconstructing the analysis. If an analysis has to be performed again or the same analysis is performed on a different data set, having the commands readily available can save much time. The documentation is useful for quality assurance purposes. For example, dead ends are useful in demonstrating that alternatives were examined. The documentation records insights, purposes, and relevance of the various activities in the analysis. In addition, the documentation can be used as sample analyses for training others.

7. ADVANTAGES AND DISADVANTAGES

Our evaluation of ADAM has pointed up both advantages and disadvantages to the current implementation. Many of the disadvantages of ADAM can be rectified through the use of tools that were not available to us when the design of ADAM began. ADAM makes extensive use of windowing but its windowing is too slow. Many workstations have built-in windowing software that would not only provide greater speed but would also provide a built-in mechanism so that experienced users could skip some levels of menus. It would also be advantageous to have multiple windows active at the same time. With the current implementation, the analyst is either in the ADAM mode doing data analysis management or is in S doing analysis. It is time-consuming to move from one mode to the other. In a workstation environment, a data analysis management system could run in one window while a data analysis system is run in another.

ADAM does not actively participate in the data analysis. Although the log is recording during the analysis, there is no intervention of ADAM in the process. As a result, the analyst can do things that will cause data analysis management to fail. For example, the ability to restore a save-state is based on the assumption that the data sets still exists. In this prototype, the analyst can delete a data set that ADAM needs for restoring a particular state. If ADAM actively participated in the analysis, ADAM could inform the analyst of the impact that deleting a particular data set would have. If the analyst wanted to delete the data set anyway, the affected save-state could be marked as non-restorable.

Another way in which a data analysis management system should actively participate in the analysis concerns the cleaning up of the log. In ADAM, the analyst can use the standard text editor to make changes to the log. It is possible that an analyst might remove necessary information from the log in addition to removing superfluous entries. Rather than having the analyst clean up the log, the data analysis management system could do the cleanup in consultation with the analyst. This would prevent the analyst from inadvertently removing necessary information. Work on the auditing of data analyses [2] demonstrates how logs can be processed to determine the evolution of data sets and serves as one approach to cleaning up data analysis logs.

ADAM allows analysts to create save-states as desired but provides no mechanism for imposing a superstructure on the save-states. The analyst may wish to group a set of save-states into a higher-level structure and label it. We would like to provide a capability so that this grouping would normally be seen on the tree. The analyst could choose to zoom in on that structure and see the save-states that make it up. Such a facility is available in DINDE [7], a prototype statistical system running on a Xerox 1108 personal workstation. DINDE graphically depicts the analysis but has no concept of save-state.

Our current implementation works only in conjunction with S. Often data preparation is done outside S. It would be useful to have ADAM track activities done outside of S as well as activities done in other data analysis packages. However, we used S for the prototype because S was designed to be extendable. We had access to the S source code. S has its own interface language that allows users to add functions. It would have been very difficult to implement ADAM using a statistical package other than S.

There are many advantages to the current implementation of ADAM. Strong points of ADAM include the ability to depict the logical course of the analysis in addition to the temporal course of the analysis represented by the log and the ability to restore previously-defined save-states.

The current implementation is non-obstrusive. It does not interfere with the course of the analysis, but can be easily invoked as needed. It makes no attempt to guide, assist, or consult with the analyst.

ADAM provides a familiar, comfortable environment in which to work. Because it is menu-driven, there are no extra commands to remember. It uses the standard text editor. The current implementation runs on the Tektronix 4100 series of terminals so it can easily be run from the analyst's office. Convenience of access is important to our analysts. Not as much use as anticipated has been made of the tape deck because it is not portable and is currently in an inconvenient location. Although the figures in this paper do not include color, ADAM does make use of color. On-line help is available for every menu.

8. CONCLUSIONS

Much has been learned in designing and implementing the prototype data analysis management system. It has validated many of the concepts we saw as being basic to understanding the data analysis process. We believe that the concepts of graphically depicting the course of the analysis through save-states and the ability to restore a save-state for further analysis are powerful and useful. Further work will continue to incorporate these concepts. Our next data analysis management system will be based on a workstation. This will be done to take advantage of the software available on the workstation and to take advantage of the speed at which graphics can be generated on the workstation. However, we will continue to use the VAX as the machine on which the analysis is performed since statistical analysis and graphics tools already exist there.

9. REFERENCES

[1] Becker, R. A., and J. M. Chambers. _"S"--A Language and System for Data Analysis._ Bell Laboratories, Murray Hill, New Jersey, 1981.

[2] Becker, R. A., and J. M. Chambers. "Auditing of Data Analyses," _Proceedings of the Eighteenth Symposium on the Interface of Computer Science and Statistics,_ March, 1986.

[3] Carr, D. B., P. J. Cowley, M. A. Whiting, and W. L. Nicholson. "Organizational Tools for Data Analysis Environments," _American Statistical Association 1984 Proceedings of the Statistical Computing Section,_ American Statistical Association, 1984.

[4] Cowley, P. J., and M. A. Whiting. "Managing Data Analysis Through Save-States," _Proceedings of the Seventeenth Symposium on the Interface of Computer Science and Statistics,_ March, 1985.

[5] Nicholson, W. L., D. B. Carr, P. J. Cowley, and M. A. Whiting, "The Role of Environments in Managing Data Analysis," _American Statistical Association 1984 Proceedings of the Statistical Computing Section,_ American Statistical Association, 1984.

[6] Nicholson, W. L., P. J. Cowley, D. B. Carr, and M. A. Whiting,"Managing the Data Analysis Process," _Data Base Management: Proceeding of a Conference, November 1-2, 1984,_

Department of Statistics, Stanford University, 1984.

[7] Oldford, R. W., and S. C. Peters. DINDE: Towards More Statistically Sophisticated Software. Technical Report # TR-55, Massachusetts Institute of Technology, October, 1985.

----

* This work was supported by the Applied Mathematical Sciences Group, Scientific Computing Staff, U. S. Department of Energy, under contract DE-AC06-76RLO 1830.

Tools for Data Analysis Management

Ronald A. Thisted, The University of Chicago

ABSTRACT

Interactive statistical computer programs represent one class of tools which have made it easier for statisticians to carry out the computations associated with data analysis. We discuss additional tools, both software and hardware, which can be combined with interactive statistical packages to make it easier for the statistician to implement a personal strategy for analyzing data. An integrated collection of tools for data analysis is called a *computing environment*. We describe the *DAMSL* computing environment which is built around off-the-shelf hardware and software costing less than $4,000. This environment is designed to alleviate many of the managerial burdens which arise in analyzing data.

0. INTRODUCTION

Adventures in Tomorrowland. The fascination and the value of the Interface meetings for statisticians is that we discuss new methods of data analysis, new computing methods, new hardware. We learn about new programs that people are working on that employ new metaphors for data analysis that arise from, or make good use of, new computing technologies. We explore new ideas for using computer hardware and for building computer software that may in the future radically transform the practice of data analysis. It is the glimpse of the future that these meetings afford that makes them so attractive for many of us. Let me note some examples from this year's Interface.

John Tukey (1986) writes of every statistician having a background program running on his or her personal workstation which will study and diagnose interesting aspects of a data set. This program will do its work during the statistician's off hours: during lunch—at night, during faculty meetings—in short, whenever the statistician is not using it for other useful work. Paul Tukey (1986) reports on his work developing "cognostics," algorithms and heuristics for an otherwise unassisted computer program to use to select interesting views of a multivariate data set worth further scrutiny by the program's owner (perhaps when he returns from breakfast). Richard Becker and John Chambers (1986) discuss the notion of "meta data analysis," in which the steps taken by a statistician during the course of analyzing a data set themselves become the raw data for a higher-level analysis, and tools for collecting such data using the notion of an audit. Wayne Oldford and Steve Peters (1986) outline their approach to building statistically sophisticated software—programs which know something about the process of data analysis. Paula Cowley and her colleagues (1986) present their experience developing a system for managing and organizing the data-analysis process.

What these approaches have in common is that they are all exciting, they all have potential to change the way we think about and carry out data analysis, and they all exist only in prototype systems not generally available for public consumption. What they also have in common is that at least part of each can, and should, be put into use today by practicing statisticians. What can be done (and how) is the subject of the remainder of this paper.

Three concerns.

[1] Statisticians have acquired familiarity with certain software tools, such as SAS, S, Glim, Minitab, and many others. Most data analysts know one or two such packages intimately and likely have at least passing acquaintance with others. This constitutes a major investment of time and energy spent. What can we do that builds on this investment rather than making it obsolete? As Crecine (1986) points out, users of computer systems must be able to learn a *transferable technology* which will not become obsolete or unavailable tomorrow.

[2] Statisticians have adopted *strategies* for doing data analysis using computer packages. What tools can be provided which make it easier or more natural to implement those strategies, and which also make it possible to think about and to reflect upon the strategies themselves?

[3] What can mere mortals (defined to be those with limited pocketbooks, limited time to learn new systems, and limited access to prototype systems) do *today* to make data analysis more productive, to help manage the data-analysis process, and to start thinking about personal statistical strategies?

The *DAMSL* system described below addresses these concerns by integrating nonstatistical tools based on off-the-shelf technology costing less than $3,000 at today's prices. The system is designed to be a computing environment that is equally useful to someone using Glim as to someone using Minitab, as the tools it incorporates exist on top of existing software rather than built into a particular statistical program. As a consequence, what the statistician knows about a particular statistical package does not become obsolete—it becomes more useful! Moreover, the tools we introduce are generally transferable, so that exactly the same tools can be used if one switches from Minitab to do a rough plot to Glim to do a logistic regression in the middle of a data-analysis session. What we do presuppose is that every data analyst has, and will continue to have, access to standard interactive statistical programs such as those we have already mentioned.

1. STATISTICAL STRATEGY

When an experienced data analyst sits down at the keyboard to examine a data set, he or she employs general *strategies* for learning what the data have to say. These strategies include heuristics both for combining specific techniques and methods of data analysis and for using the chosen computing system. The strategies may be conscious or not. Loosely speaking, they constitute the data analyst's "style." Statistical strategy has not been systematically studied until very recently, yet is is extremely important, for several reasons. First, in teaching statistical methods, what we seek to impart is really a collection of fruitful approaches rather than a catalog of formulas. Second, if we can bring the strategies that we (individually) adopt to the conscious level, it becomes possible to examine them, to identify successful ones, and to refine them. Doing so makes it possible to recognize more readily situations in which particular approaches may not be fruitful. If statistical strategies can be verbalized, they can then be discussed, taught, and debated. More generally, improved understanding of statistical strategy leads to better and more productive data analyses. Third, if there is any hope to construct expert systems that can assist in data analysis, it is essential to come to a more complete understanding of the nature of the data analysis process, including statistical strategy. Many of the chapters in Gale (1986) are devoted to the problem of collecting and representing knowledge about the strategies adopted by data analysts.

Reflection about "what we did" in a data analysis is difficult to carry out, since after the fact, many spur-of-the moment decisions will have been forgotten, ideas lost, and inter-relationships among pieces of the analysis obscured. Reasons for following a given line of attack—particularly unfruitful ones—may have evaporated: "it seemed like a good idea at the time." Specially designed computing environments (Thisted, 1986) can help by looking over one's shoulder during the analysis, and by making it possible to record information about both the intentions which lead to each step being taken and the deductions resulting from that step (Thisted, 1985). These ideas lie behind the notions of auditing a data analysis à lá Becker and Chambers (1986). The *QPE* system, in which data auditing is being developed, is a prototype system. Some of the features of data auditing can be immediately realized using *DAMSL*.

2. Data Analysis Management

Except for the most trivial data analyses, the analysis process is a long and involved one, requiring the statistician over several days to keep track of a plethora of modified and transformed variables, subsets of the data, results of intermediate analyses, loose ends to follow up on later, tables and graphs, side computations that must be done outside the main statistical package, and the like. This has become particularly difficult to do using the standard interactive computing systems with which most statisticians are now familiar, for example, using Minitab on a timesharing computer via a 24-line video-display terminal. Much of the statistician's mental energy and organizational skills are diverted from the data *per se* and redirected to these essential, yet peripheral, matters.

Although courses in statistical methodology rarely address the issue, the practice of data analysis involves projects of moderate to high complexity, the management of which is nontrivial. Researchers at Batelle Northwest Laboratories have made considerable progress in developing computer tools whose function is to assist in the management aspects of data analysis (Carr, Cowley, and Whiting, 1984; Cowley and Whiting, 1985; Cowley, Carr, and Nicholson, 1986). The *ADAM* system which they have developed is integrated with the *S* statistical language. Although *ADAM* only exists in prototype, some of the features which make *ADAM* attractive enhance the productivity of data analysts and are realized in *DAMSL*.

3. Computing Tools to Facilitate Data Analysis Management and Strategy

We now turn to some ideas for creating a computing environment which realizes some of the more important features of prototype systems such as *QPE* and *ADAM*, and which can do so at low cost. The system we describe can be obtained immediately, for less than $3,000. (As an existence proof, we name names and list prices.) The emphasis will be on providing tools for data analysis management and for analysis and implementation of statistical strategy.

Computer tools for assisting data analysis can be divided into three groups. ***Intelligent*** software incorporates knowledge about the process of data analysis. *DINDE*, described in Oldford and Peters (1986) is an example of a program which knows a moderate amount about such things as collinearity and means for diagnosing its effects in different situations. *Smart* programs incorporate structural knowledge about particular statistical software, but know nothing directly about how the software could be used to perform a data analysis. Examples in this category include *QPE* and *ADAM*, each of which knows something of the structure of *S*, but nothing of the structure of data analysis. The last class consists of the *dumb* software, which knows nothing of data analysis or of statistical programs. *DAMSL* is dumb software. But it is available.

The advantages of the dumb approach are many. It is cheap. The product is portable. It is hardware-independent, in the sense that it does not depend upon the particular type of machine on which you do your interactive computing; it is as much at home on a UNIX machine as on a DECSystem-20. In addition, it is software-independent, in the sense that it can be used with *S*, Minitab, Glim, SAS, SCSS, or any other statistical software, in a fashion that is insensitive to changes that may be introduced into any of these programs.

The idea is to take a set of ordinary programs and machines which can be adapted to perform tasks involved in developing statistical strategy or in managing data analysis. This hardware and software is then ***integrated*** into a coherent computing environment within which ordinary data analysis can also be conducted. The components of *DAMSL* are listed below. We shall discuss the capabilities of the hardware components in section 4, the capabilities of the software components in section 5, and the integration of these components with each other and with statistical software in section 6.

The *DAMSL* Hardware and Software

• A 512K Macintosh	$1,495
• Imagewriter II printer	495
• External Apple disk drive (400K)	395
• Prometheus 1200 bps modem	250
• Switcher	20
• Versaterm	100
• Microsoft Word	120
TOTAL	$2,875

The prices quoted are unrestricted prices available from computer stores or by mail order in March 1986. These prices reflect a reasonable *upper bound* on what this system should cost.

4. The Set of Hardware Tools

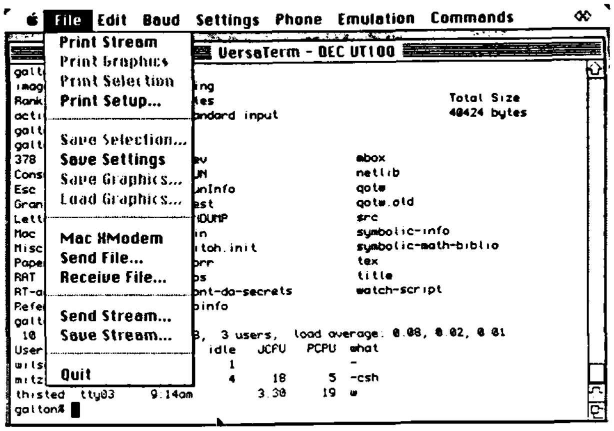
The hardware needs of a statistical computing environment could be met by any of a number of personal computers. The features which are important (which the Macintosh, for instance, includes) are: high resolution display capable of both text and graphics processing, adequate memory for multiple applications to run simultaneously, printer interface, and a telecommunications (modem) interface. These are included in the price of the Macintosh; systems based on other personal computers (such as the IBM PC) would generally have to take account of the additional expense these items entail. The Macintosh has the advantage that it is particularly easy to learn to use, thus minimizing the amount of time that must be invested to learn the system.

The printer selected is integrated with the computer. It is a modern dot-matrix printer that has good resolution and is capable of printing high-resolution graphics as well as text without additional hardware. A second disk drive is an essential component of the system. The 1200 baud modem is the means for communication with the remote timesharing computer on which the statistical software will be accessed. Such devices generally can operate at either 1200 or at 300 baud interchangeably.

5. The Set of Software Tools

As with the hardware, other software components could serve our purposes to similar effect; the particular combination discussed here works particularly well with the Macintosh and with each other.

Versaterm. This program is a terminal-emulation program which makes it possible to use the hardware of the previous section as if it were a very smart terminal. Versaterm can (simultaneously) emulate both a standard video display terminal, the DEC VT100, and a standard graphics terminal, the Tektronix 4010. Moreover, the terminal emulator can run at 9600 baud using dedicated communications lines. Thus, at a minimum, the *DAMSL* system can simply act as the terminal on which data analysis is done. But Versaterm has many features which are particularly useful in data analysis. Several of these are illustrated in Figure 1. They are selected by pointing the mouse to particular choices in a menu, which temporarily overlaps the main display.



```
galt
imag                  ing
Rank                  es                                         Total Size
acti                  ndard input                                40424 bytes
galt
galt
378                   ev                       mbox
Cons                  JN                       netlib
Esc                   unInfo                   qotw
Gran                  est                      qotw.old
Lett                  DUMP                     src
Mac                   in                       symbolic-info
Misc                  toh.init                 symbolic-math-biblio
Pape                  rr                       tex
RAT                   bs                       title
RT-a                  nt-do-secrets            watch-script
Refe                  binfo
galt
 10                   3,  3 users,  load average: 0.08, 0.02, 0.01
User                   idle   JCPU   PCPU  what
wils                     1                 -
mitz                     4     18      5   -csh
thisted  tty03    9:14am        3.30     19  w
galton% █
```

FIGURE 1

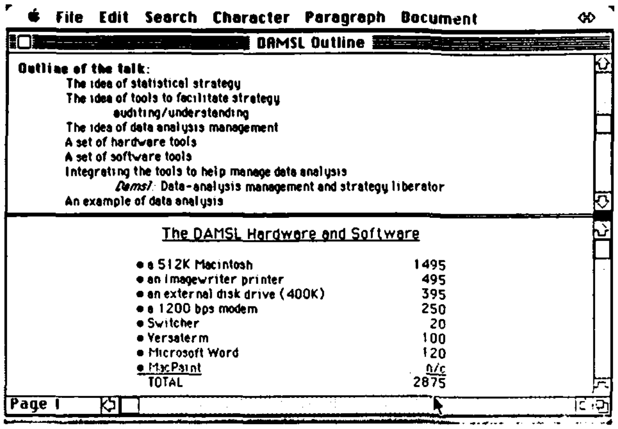
First, one can save the transcript of the terminal session to disk, either for later perusal or for archival purposes. This transcript includes everything that transpires on the screen, including both input and output. Thus, one has the option of retaining everything that one would otherwise have from a session conducted at a (much slower) hard-copy terminal. What is more, this stream-saving feature can be turned on or off at any time by moving the mouse to the "Save Stream" item on the menu, so that is can be used selectively.

A second related feature is that the stream (of both input and output) can also be echoed to the attached printer. Thus, if the data analyst knows that he or she will want particular results for permanent reference (or even frequent reference during the terminal session), that work can be printed automatically as it is generated. This feature, too, can be toggled at any time by selecting the item "Print Stream".

Versaterm (when configured as in *DAMSL*) also retains eighteen previous VT100 screens, as well as the current screen. The information on these screens really should be thought of as a single screen of some 450 lines, any 24 of which are visible at a single time. The particular 24 lines are determined by the position of the scroll bar on the right-hand side of the display. The white box can be thought of as an elevator car whose position in the elevator shaft corresponds to position in the 450-line terminal memory. The mouse can be used at any time to move the elevator box to any desired position. It is also possible to scroll forward or backward a line at a time. This feature makes it very easy to go back to a recently performed part of the analysis whose importance may not have been obvious at the time.

In conjunction with the scrolling feature, it may turn out to be useful to have a hard copy of a portion of the analysis, say a plot of the data, or a listing of potential outliers, or a table of summary statistics. Versaterm makes it easy to go back to that portion of the output, select any number of lines from it, and then to print the selection on the printer. Such a selection could also be saved on a disk file on the Macintosh as well.

In addition to the features listed above, Versaterm has a separate graphics screen for Tektronix emulation,

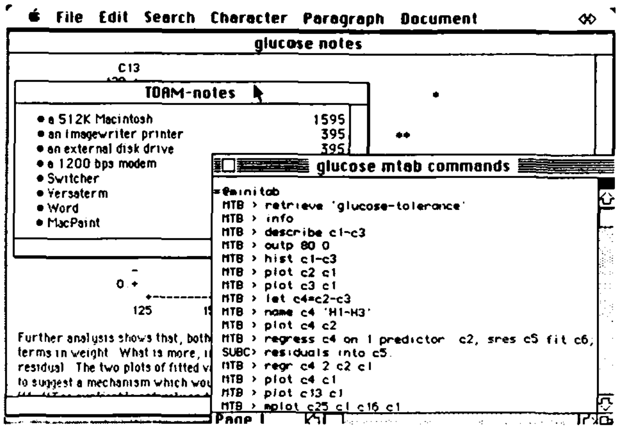


Outline of the talk:
The idea of statistical strategy
The idea of tools to facilitate strategy
auditing/understanding
The idea of data analysis management
A set of hardware tools
A set of software tools
Integrating the tools to help manage data analysis
Damsl: Data-analysis management and strategy liberator
An example of data analysis

The DAMSL Hardware and Software

● a 512K Macintosh	1495
● an Imagewriter printer	495
● an external disk drive (400K)	395
● a 1200 bps modem	250
● Switcher	20
● Versaterm	100
● Microsoft Word	120
● MacPaint	n/c
TOTAL	2875

FIGURE 2



● a 512K Macintosh	1595
● an Imagewriter printer	395
● an external disk drive	395
● a 1200 bps modem	
● Switcher	
● Versaterm	
● Word	
● MacPaint	

Further analysis shows that, both
terms in weight What is more, i
residual The two plots of fitted v
to suggest a mechanism which wou

```
=@minitab
 MTB > retrieve 'glucose-tolerance'
 MTB > info
 MTB > describe c1-c3
 MTB > outp 80 0
 MTB > hist c1-c3
 MTB > plot c2 c1
 MTB > plot c3 c1
 MTB > let c4=c2-c3
 MTB > name c4 'H1-H3'
 MTB > plot c4 c2
 MTB > regress c4 on 1 predictor  c2, sres c5 fit c6;
 SUBC> residuals into c5.
 MTB > regr c4 2 c2 c1
 MTB > plot c4 c1
 MTB > plot c13 c1
 MTB > mplot c25 c1 c16 c1
```

FIGURE 3

from which it is possible to produce quite adequate hard copies of graphics output. The Tektronix output can also be saved to disk, for later processing.

Finally, Versaterm has several protocols for transferring files between the Macintosh and other computers, including Xmodem and Kermit protocols. (The text for this paper, for instance, was written on the Macintosh and transferred using Versaterm to a computer with typesetting software.)

Microsoft Word. The second workhorse of the *DAMSL* system is a word processing program, some of whose features are illustrated in Figures 2 and 3. Word is one of several similar programs available for the Macintosh which allow use of multiple fonts, resizable windows, and a screen display which corresponds very closely to printed output. The features of Word which make it the choice for *DAMSL* are its ability to have up to four windows open simultaneously, in which four separate documents can be processed, and the fact that Word has keyboard equivalents for most options which can be selected by the Mouse.

As Figure 2 illustrates, a single document can be viewed in two different places at once, by splitting a window, so that comments can be typed in, say, the lower window, while observing the contents of the upper window. Figure 3 demonstrates that several different windows, with quite different contents, can be in use simultaneously. By clicking the mouse on any one of the windows, that window becomes the active window; by clicking twice on the top of the window next to the title, the window automatically increases in size to occupy the whole screen. Another double-click returns the window to its smaller size and position.

It is possible, with a few mouse movements, to move or copy any portion of any window into any position in any other window. Additions and insertions again require only a mouse click to initiate. Changes of font—both style and size—can be accomplished either with the mouse or with one or two keystrokes.

Switcher. Although most of the real work is done by Versaterm or by Word, the ingredient that makes the system work is the program *Switcher*. Switcher is a program which makes it possible to run different programs on the Macintosh simultaneously. In *DAMSL*, we run a word-processing program (Word) in one area of the Mac's memory, and a terminal emulator (Versaterm) in the other. Each program acts as if it had the Mac to itself when it is the active program, and each has its ow ı set of windows in which computation is done. Switcher makes it possible to move from one program to the other in less than a second, by simply clicking the mouse once, or by typing a single key (your choice). What is more, it is possible to move contents of one program's windows into those of the other programs. Thus, for instance, I can copy a scatterplot created by Minitab in Versaterm directly into the middle of a manuscript I am working on using Word. Alternatively, I can extract a Minitab command embedded in a Word window (such as a line from the "glucose mtab commands" window in Figure 3) and cause it to be executed by the mainframe program being run using Versaterm.

Switcher makes two things possible: to change focus almost instantaneously from a program where work is carried out (Versaterm) to a program where information about the work is recorded (Word) and vice versa, and to move information back and forth between these two areas of focus.

6. *DAMSL:* AN INTEGRATED SYSTEM

DAMSL is a somewhat strained acronym standing for **D**ata-**A**nalysis **M**anagement and **S**trategy **L**iberator. The *DAMSL* system consists not only of its individual hardware and software components, but the ways in which they are integrated with one another and with underlying statistical software such as Glim and Minitab running on a remote computer. The integration is accomplished through the environment-switching capabilities of Switcher, the multiple text-window capabilities of Word, and the graphics, marking, saving, and file-transfer capabilities of Versaterm. Viewed in the large, *DAMSL* can be thought of as providing up to six distinct windows onto the data analysis process, within each of which separate aspects of the overall task can be accomplished. They make it possible to focus more

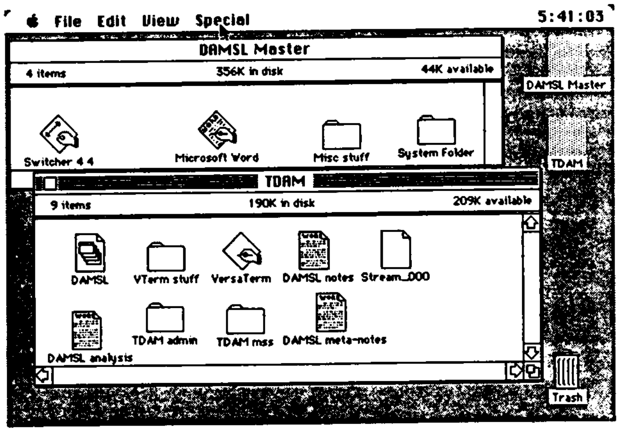
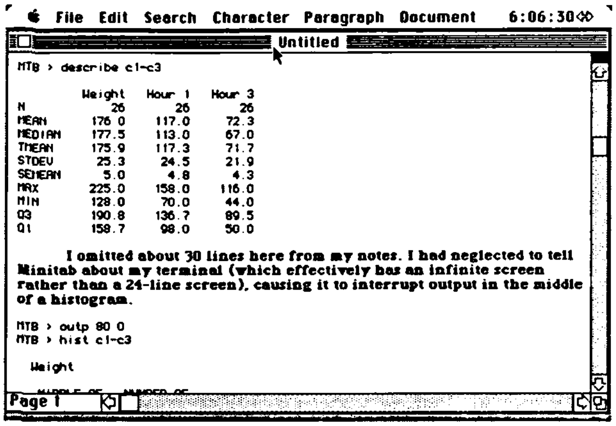


FIGURE 4



```
MTB > describe c1-c3

          Weight   Hour 1   Hour 3
N             26       26       26
MEAN       176 0    117.0     72.3
MEDIAN     177.5    113.0     67.0
TMEAN      175.9    117.3     71.7
STDEV       25.3     24.5     21.9
SEMEAN       5.0      4.8      4.3
MAX        225.0    158.0    116.0
MIN        128.0     70.0     44.0
Q3         190.8    136.7     89.5
Q1         158.7     98.0     50.0
```

I omitted about 30 lines here from my notes. I had neglected to tell Minitab about my terminal (which effectively has an infinite screen rather than a 24-line screen), causing it to interrupt output in the middle of a histogram.

```
MTB > outp 80 0
MTB > hist c1-c3

  Weight
```

FIGURE 5

easily on just one aspect of the data analysis at a time, yet moving from one aspect to another involves no more than a second to accomplish. When these tasks overlap or interact, information can be transferred readily from the context in which it is generated to another context in which it may prove useful.

For managing the data-analysis process, and for monitoring and evaluating statistical strategies, the key lies in the ways these windows and the other features of the constituent software are used. These are perhaps best illustrated with reference to an example.

The example involves the reanalysis of a published data set from Smith and Choi (1982) concerning glucose metabolism in 26 healthy male volunteers. Each was given a standard glucose challenge dosage, and the levels of plasma glucose (mg/dl) were recorded one hour and three hours after the challenge (X_2 and X_3). In addition, each subjects weight in pounds was also recorded (X_1). Smith and Choi used the data to illustrate a test comparing two dependent regression lines; they concluded that the regressions of X_2 on X_1 and of X_3 on X_1 were different. The question leading to the reanalysis was whether some simple model could be found that satisfactorily represented the relationship of glucose metabolism to weight. Although this is a relatively simple problem, it is sufficient to illustrate many of the ways in which *DAMSL* can assist in the data analysis process. Because Minitab 82.1 is widely familiar, I shall use it as the statistical package for my data analysis.

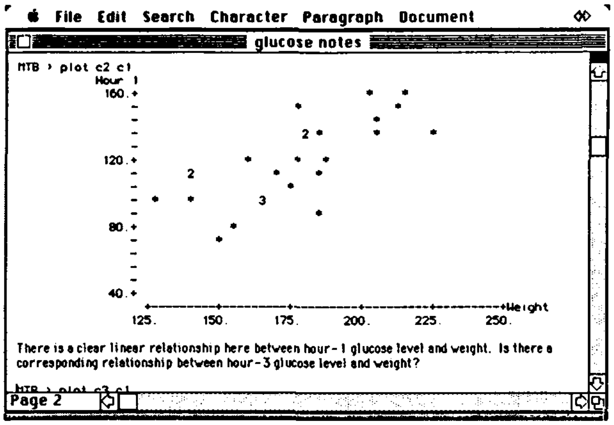
The first thing that we do is to insert two disks containing the *DAMSL* system into the computer; the Macintosh shows the contents of the two disks as in Figure 4. One disk contains Switcher and Word (DAMSL Master), the other contains Versaterm, a file called DAMSL, and other files and documents related to the current data analysis. Generally, the first disk is the same for all applications, while each different project will have its own incarnation of the second disk. To start *DAMSL*, one clicks twice on the file named DAMSL. This automatically starts Switcher, which in turn starts both Word and Versaterm in areas of memory of prespecified size (160K for Word and 256K for Versaterm). Once Switcher has been launched, at the upper left corner of the screen is a pair of arrows. By moving the mouse to either arrowhead and clicking once, Switcher moves in the next application program. Clicking the arrow now, for instance, immediately moves us to Word.

The next step, then, is to enter Word and to set up three windows. The first I call "glucose notes," the second "glucose variables," and the third "glucose mtab commands." These will contain, respectively, a running commentary on the most important aspects of the analysis in progress; a workspace in which I can keep track of the contents of the Minitab worksheet and its variables, matrices, and constants; and an area in which I can keep track of the sequence of Minitab commands that I have used in the analysis. This uses three of Word's four available windows. I reserve the fourth for possible temporary use later in the analysis. The Word side of *DAMSL* is now ready to go.

We then switch to Versaterm, and immediately choose the option to "Save Stream" under the File menu. This means that our entire session with the remote computer will be automatically recorded on the Macintosh disk. We can then examine it later, or print it out, or discard it, or even edit it. Next, we dial the remote computer (by selecting an item from the Phone menu), and we login as usual and launch Minitab.

The preliminaries come first: we load the data set (already saved, for convenience, in a Minitab worksheet), and ask for the brief information stored with the worksheet. This information I select using the mouse and then I copy it into the Word window "glucose variables", where it can serve as a constant reminder, both of the contents and the source of the data set. I then return to Versterm, this entire process having taken just a few seconds.

As a matter of course, I always obtain simple descriptive statistics for the variables in the data set, and histograms as well. This is starting-point information, and may well be referred to at many later stages of the analysis. I select this information, and copy it into the Word window "glucose notes", and the result is shown in Figure 5. Actually, as the figure makes clear, about

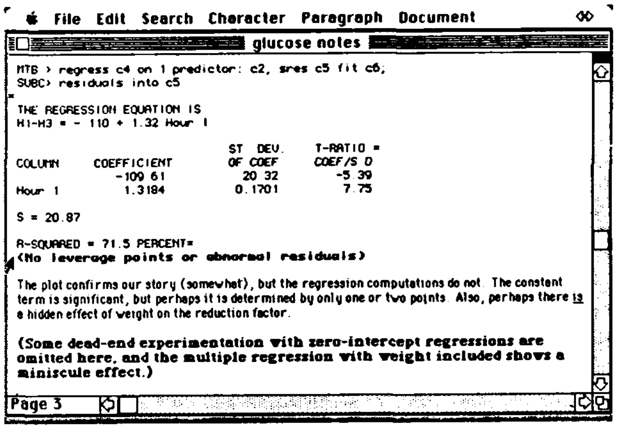


```
MTB > plot c2 c1
```

There is a clear linear relationship here between hour-1 glucose level and weight. Is there a corresponding relationship between hour-3 glucose level and weight?

```
MTB > plot c3 c1
```

FIGURE 6



```
MTB > regress c4 on 1 predictor: c2, sres c5 fit c6;
SUBC> residuals into c5

THE REGRESSION EQUATION IS
H1-H3 = - 110 + 1.32 Hour 1

                                ST  DEV.     T-RATIO =
COLUMN      COEFFICIENT         OF COEF      COEF/S D
               -109 61           20 32         -5 39
Hour 1          1.3184          0.1701          7.75

S = 20.87

R-SQUARED = 71.5 PERCENT=
```

(No leverage points or abnormal residuals)

The plot confirms our story (somewhat), but the regression computations do not. The constant term is significant, but perhaps it is determined by only one or two points. Also, perhaps there <u>is</u> a hidden effect of weight on the reduction factor.

(Some dead-end experimentation with zero-intercept regressions are omitted here, and the multiple regression with weight included shows a miniscule effect.)

FIGURE 7

thirty lines of the terminal session were *not* transferred to "glucose notes". I had made a trivial mistake in Minitab which I had no reason to perpetuate. The window "glucose notes" is to contain only those parts of the analysis which are important or useful in my chain of reasoning about the data; I edit out that which is useless, and I do so "on the fly." When I review the analysis (after having taken a lunch break, for instance) I examine "glucose notes" rather than leafing through the last 400 lines of computer output.

Already, I have begun to organize and to assist my own thinking about the problem. In the sequel, I do so in a moderately systematic way. Note that Figure 5 contains material in two different fonts—a monospace font (Monaco) in which Minitab input and output is displayed and a boldface font (New York) in which I have inserted commentary about the computing process. I use an additional lightface font (Geneva, seen in figure 6) for my plans and for my own analysis. These fonts visually represent different aspects of the computations I undertake, and I use them systematically.

Before going to Minitab to do a sequence of computations, I jot down a few notes in "glucose notes" concerning what I am about to do, and why, using the Geneva font. I then follow this idea in Minitab until I have done what I set out to do, or until it seems as if my plan is changing somewhat. At that point, I pause, and I copy relevant portions of the Minitab output into "glucose notes", using the Monaco font. I then record in a few words my interpretation of what I have seen, and how that has changed my view of what should be examined next (using Geneva again). It is also a trivial matter to annotate the Minitab output that has been copied to "glucose notes", and it is often useful to do so. If a regression coefficient in one analysis is close to its theoretical value, I can note that fact "on the output," as it were, and I can do so in real time. I then write down a few words about what I am about to do next, and then I repeat the cycle. This is illustrated in Figure 6, in which an interpretation of a plot is coupled with its implications for what to do next.

In copying the Minitab output to "glucose notes", I copy only those portions which are important in shaping my understanding of the problem or of the course of the analysis. Other aspects are merely summarized; for such summaries, I use boldface type. This is illustrated in Figure 7. Note that the boldface entries record something I looked at, but which did not contribute to the unfolding story. Using this system dead ends don't simply disappear, they actually contribute to the overall understanding of the problem. At the same time, the landscape is not cluttered with fragments of computation that must be waded through in order to reconstruct an analysis. When used in the manner outlined here, "glucose notes" represents an implementation of the "three-ring binder" ideas of Thisted (1986).

If at any time it becomes clear that there are *several* lines of attack to examine, I write them all down in a list as part of "glucose notes", and then I copy the list to a new (fourth) window, in which I keep temporary notes—in this case, a list of things "to do." As I do each one, I can check it off in this fourth window.

Occasionally, something passed over as insignificant (and hence not copied to "glucose notes") will later turn out to be relevant. The terminal memory is usually sufficient to cope with the problem; anything done fewer than 450 lines ago can be retrieved in a few seconds and pasted into its proper place in "glucose notes", along with the corresponding notes about its importance and how it came to be realized. If the 450-line boundary has passed, however, there is still the complete transcript of the terminal session that has been silently recorded throughout which can be used to retrieve *anything* of interest that transpired during the session.

One of the three windows that we originally set up has not yet been mentioned. It is sometimes useful to record the sequence of Minitab commands used in an analysis. These can easily be extracted by copying, say, several hundred lines at a time from Versaterm memory into "glucose mtab command", and then in a single pass, removing the Minitab output. Remarkably, Word makes it possible to do this with little effort, even though it is done "manually." What is more generally useful is to copy the contents of "glucose notes" (which contains Minitab input as well as output and comments) into "glucose mtab command", and then to strip out the non-commands. This was done, for instance, in the front-most window of Figure 3. If a sequence of commands, once done, appears to be particularly useful, it can be copied into "glucose mtab command". There, the Minitab prompts can be removed, the commands edited, and then the whole sequence can be copied into a Minitab EXEC file using the Minitab STORE command on the Versaterm side.

We have mentioned in passing "glucose variables". In this window, we keep track of transformed variables, subsets of the data, and the contents of Minitab worksheets that we create and save during the course of the analysis. It is here that we can emulate some of the features of "save states" discussed in Cowley and Whiting (1985), by recording the contents of various saved worksheets (in effect, Minitab snapshots) and their relationship to one another.

7. DISCUSSION

The value of *DAMSL* is that it makes it possible to isolate different aspects of the process of data analysis in separate "windows." As a result, the data analyst need not keep so many things in view all at once. Moreover, it automates (or at least simplifies) many of the bookkeeping and other managerial tasks associated with data analysis. Because it is not fully automatic—it is a dumb assistant—using it effectively depends upon the data analyst adopting a certain (mild) disciplined approach to data analysis. As with the discipline introduced when using structured programming techniques to write computer programs, this discipline serves to organize the analysis both mentally and computationally.

Some of the benefit of data analysis auditing (see Becker and Chambers, 1986) can be achieved using the "glucose mtab command" window. At the very least, this makes it possible to obtain a diary for any interactive statistical package, which could then be used as meta-data for further study of the data analysis process.

Many statistical packages have diary or journal features which, coupled with the ability to insert comment-commands could serve some of the purposes to which *DAMSL* has been put. What distinguishes *DAMSL* from what is available in S, say, is twofold. First, the process of annotation is carried out in a different "place" than is the analysis proper. This makes it easier mentally to focus on the separate activities of analysis, planning, and interpretation on the one hand, and the actual computation on the other. When one is typing comments into *S*, one is always aware that comments are being inserted into *S*—omit a single # mark and try to recover your train of thought! Second, in *DAMSL* both the input and the output are available for annotation and editing. Thus, one can "mark" on the output, or eliminate output that has been looked at but found not to be of further interest.

The contents of the three-ring binder ("glucose notes" in our example), particularly when compared against the entire session transcript, can be a powerful tool for reflecting on the strategies adopted in the analysis and on their effectiveness.

ACKNOWLEDGEMENTS

This work was supported by National Science Foundation Grant DMS-8412233.

REFERENCES

Becker, Richard A., and Chambers, John M. (1986). "Auditing of data analyses." *Computer Science and Statistics: Eighteenth Symposium on the Interface, Proceedings*, Thomas J. Boardman, editor. This volume.

Becker, Richard A., Chambers, John M., and Wilks, Allan R. (1985). "Workstations for statistical computing: Present and future developments." AT&T Bell Laboratories Statistical Research Report Number 10.

Carr, D. B., Cowley, P J., and Whiting, M. A. (1984). "Organizational tools for data analysis environments," *Proceedings of the Statistical Computing Section*, American Statistical Association.

Chambers, John M. (1985). "S as a programming environment for data analysis and graphics." AT&T Bell Laboratories Statistical Research Report Number 13.

Chambers, John M. (1986). "Computing environment for quantitative applications." AT&T Bell Laboratories Statistical Research Report Number 17.

Cowley, P.J., Carr, D. B., and Nicholson, W. N. (1986). "Experiences with a data analysis management prototype," *Computer Science and Statistics: Eighteenth Symposium on the Interface, Proceedings*, Thomas J. Boardman, editor. This volume.

Cowley, P. J., and Whiting, M. A. (1985). "Managing data analysis through save-states," in *Computer Science and Statistics: Seventeenth Symposium on the Interface, Proceedings*, D. F. Allen, editor. North-Holland: Amsterdam. In press., in press.

Crecine, John P. (1986). "The next generation of personal computer," *Science*, **231**, 935–943.

Gale, William A., editor (1986). *Artificial Intelligence and Statistics*. Addison-Wesley: Reading.

Oldford, R. W., and Peters, S. C. (1986), "DINDE: Towards more statistically sophisticated software," *Computer Science and Statistics: Eighteenth Symposium on the Interface, Proceedings*, Thomas J. Boardman, editor. This volume.

Smith, Philip J., and Choi, Sung C. (1982). "Simple tests to compare two dependent regression lines," *Technometrics*, **24**, 123-126.

Thisted, Ronald A. (1985). "Knowledge representation for expert data analysis sustems." in *Computer Science and Statistics: Seventeenth Symposium on the Interface, Proceedings*, D. F. Allen, editor. North-Holland: Amsterdam. In press.

Thisted, Ronald A. (1986). "Computing environments for data analysis" (with discussion), *Statistical Science*, **1**, in press.

Tukey, John W. (1986) "The interface with computing: In the small or in the large?" Keynote address, *Computer Science and Statistics: Eighteenth Symposium on the Interface, Proceedings*, Thomas J. Boardman, editor. This volume.

Tukey, Paul A. (1986). "Cognostics for looking at higher-dimensional data," *Computer Science and Statistics: Eighteenth Symposium on the Interface, Proceedings*, Thomas J. Boardman, editor. This volume.

STATISTICALLY SOPHISTICATED SOFTWARE AND DINDE

R.W. Oldford and S.C. Peters, Massachusetts Institute of Technology

Abstract

We describe a prototype system, which we call DINDE, and the directed network model of statistical analysis on which it is currently based. DINDE is a highly interactive display oriented system where the user carries out the analysis by building and maintaining a network representation of it. An example analysis is used to describe this interaction and the analysis management tools required.

1.0 Introduction

By statistically sophisticated software, we do not mean software that implements a sophisticated statistical method, but rather, software that contains information on how and when that method is most frequently used in practice.

As shown in Figure 1, there are at least three reasons why one might attempt implementation of such software: first, to have software which guides the user to a better statistical analysis (e.g. Gale and Pregibon[1982], Oldford and Peters[1984]); second, to use the software as a medium for studying statistical strategy (e.g. Pregibon[1985] or Oldford and Peters[1984,1985ab]); and, third, to have software which helps the user manage the analysis (e.g. Carr et al [1984], Becker and Chambers[1985], Oldford and Peters[1985b]).

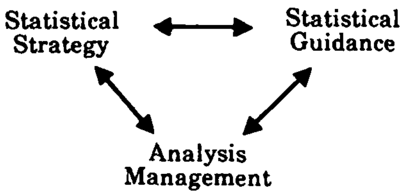


Figure 1: Three interrelated objectives

These objectives are interrelated and software for one often leads naturally to software for another. For example, to successfully guide the analysis one needs to understand and implement the supporting statistical strategies. To study the strategies of good statistical data analysis with software, one needs to be able to manage the analysis. In developing DINDE, we have repeatedly found ourselves concentrating on each objective in turn; advancement toward one objective has often produced insight on one of the others.

Whether the chief interest is to *guide*, *study*, or *manage* the analysis, some model of an analysis is required. The next two sections address this question rather generally. The remaining sections describe the current implementation of one such model in DINDE. Like any model, the current model of statistical data analysis used in DINDE is temporary and will improve with experience. The last section indicates some of the modifications we already foresee.

2.0 What's an Analysis?

The simplest, and least satisfying, view of a statistical analysis is as a *specified sequence* of steps. An example would be regression modelling by a forward selection procedure, where variables are added to the current regression model one at a time according to some criterion.

This model of statistical analysis seems to underlie the batch-oriented statistical packages of the late 1960s and early 1970s, or, at least, to underlie their common usage. The sequence of steps to be taken in the analysis is defined in advance and the corresponding set of packaged routines is run. In light of the results, this sequence may be modified and the resulting new set of procedures run. Each run is regarded as a different analysis. The refinement of the analysis continues until the analyst is satisfied that the final one is "correct" for the problem. In this paradigm, novices typically would not substantially modify their original analysis.

A more accurate view of statistical analysis, based largely upon a scientific modelling paradigm as expressed, for example, by Box[1976], is that it is an iterative procedure whereby statistical models are alternatively fitted and criticised.

Unrolling this iterative loop produces a different kind of sequential process, one whose steps are not predetermined. Instead, the analyst decides what to do next based upon the results of the preceding step. Rather than a specified sequence that is continually refined, this model of an analysis is a *dynamic sequence*, one that grows as more is learned about the problem and the data. The analysis is represented by the entire sequence, not just by a final revision. Here, novices would typically produce shorter sequences than would experts.

As such, this model fits well with modern interactive statistical systems like S (Becker and Chambers [1984]), where each step of the analysis corresponds to a command issued to the system. After examining the results of each command, the analysis is grown by issuing another command. A macro facility is usually available to allow the user to compress many small sequential steps in the analysis into a single larger one that is easier to comprehend, and use, as a unit. In this way, new analysis steps are defined by the analyst. Diaries reinforce this model of analysis by recording the entire sequence of steps.

The dynamic-sequence model suggests that results and information from actions taken early in the analysis influence later actions only through the chain of steps given by the time ordering. But this is not an accurate description of an analysis. At any step, a number of different actions can be, and often are, taken. A model of analysis based only on the time order of actions hides this logical relationship between actions, and, hence, is seriously incomplete.

This is an important shift in focus: from the time ordering of statistical and arithmetic procedures to the conceptual steps of the analysis and the logical connections between the steps. Now, the simplest model of the analysis is a *tree*, where branches indicate a logical connection between one step and a number of others. For example, at different times, different actions or decisions may be taken from one step, resulting in many branches from it. With this model, novice analysts would likely produce short, sparse trees and expert analysts long, bushy trees.

A little reflection, however, shows that the tree model also falls short. Suppose that two branches of the tree represent two different sub-analyses that are pursued in parallel. It may happen that a new tack is taken in the analysis that is based on the combined results of both independent sub-analyses. Where should this new sub-analysis be attached? The obvious answer is to attach it to both of the previous ones, forcing the whole analysis to become a *directed network* rather than a tree.

This directed network model of statistical data analysis is currently the basis for DINDE. While it provides a better description than any of the previous models, it too has shortcomings. We discuss some of these in the last section, and indicate our planned modifications to the network model.

3.0 What are the steps again?

Implicit in the models considered above is the assumption that every analysis has identifiable steps: decision points where some action is taken. Further, it is assumed that many of these steps are generic enough to be usefully recorded. What, then, are the steps?

With current statistical systems, the steps are equivalent to the commands that are issued to the system. The first thing to notice is that the steps have varying granularity. The smallest grains include those steps where the actions are simple, arithmetic ones taken on scalars, vectors, matrices, and the like. Good statistical systems will always allow actions to be taken at this low level of analysis. Larger grains include strictly statistical actions, like regressing y on x, where the lower level steps needed to accomplish the task are suppressed from consideration. The regression step is really an abstraction of many lower-level analysis steps, an abstraction that becomes a powerful tool for the analyst.

More abstract steps are typically more powerful (i.e. do more for the analyst), but also have more restricted ranges of application (e.g. regression is a powerful tool but has smaller range of application than the matrix operations used to construct it). In designing useful steps for statistical analysis, there is always a potential tension between the range and the power of a newly proposed step.

In DINDE, our working philosophy has been to begin with steps that are reasonably generic and to increase their power in two ways that do not restrict their range of application.

First, we specialize some steps to become more context specific. The specialized steps are to be used in place of more general steps when the context warrants it. Therefore, the necessarily smaller range of applicability of the specialized steps does not inhibit the analysis. Instead, given the right context, they become powerful tools.

Second, in each step, information is incorporated as to which steps are often taken next. This simple addition makes the step more powerful at no loss to its range of application.

In DINDE, commands sometimes produce steps, but, steps are never commands. The steps in DINDE are quite different; they are collection points for possible actions (commands). Instead of specifying a sequence of actions to be taken, the abstraction in DINDE is to collect together a set of possible actions and the information that may help the analyst decide among them. A sequence of actions identified and captured in DINDE would simply be a more abstract action, not a step.

At present, the steps that have been developed in DINDE are either analysis goals or analysis artifacts.

For example, an analysis goal might be a reasonable description of the regression of y on x (i.e. the conditional expectation of y given x). This goal is represented in DINDE by the step *BivariateRegression* (bivariate since two variables are involved). A variety of information is immediately available at this step: which vectors x and y refer to, what set of actions are generally reasonable to take next (various plots and fitting routines), and which of these actions it is considered wise to take first (visually inspect the data via a scatterplot and histograms).

Choosing to do the scatterplot of y versus x would produce a *Scatterplot* as an artifact. Again, since it is a new step, particular actions would be made available for this kind of artifact (like fitting a straight line or a smooth curve to the points in the plot).

Clearly, a number of steps are necessary for an analysis with even the relatively simple goal of describing the regression of one variable on another. The challenge, then, is to make the steps generic enough to be useful in a variety of analyses.

4.0 DINDE

The challenge of DINDE is to produce and implement a model for statistical analysis that is both reasonably accurate and natural to use. Sections 2 and 3 indicate the underlying model; here, and in the sections which follow, we discuss its implementation and use.

DINDE is an enrichment of an extensive interactive programming environment: Interlisp-D with LOOPS which runs on the Xerox 1108 personal workstation (see Teitelman and Masinter [1981], Stefik et al [1983]). The combination of high interaction, extensive graphics, powerful dedicated computing, and powerful programming tools available in this environment has proved to be enormous leverage for designing and building DINDE.

The hardware is described in more detail in Oldford and Peters [1985b]. We note only two features here: first, an illusion of an infinitely large memory is maintained for the user (1.5 - 3.5 MBytes of real memory, 32 MBytes virtual), and second, the display is a high resolution bit-mapped display (about 1000 by 1000 individually addressable pixels) that can be interacted with using a "mouse" pointing device.

The software environment is at least as important as the hardware. Interactive programming environments are the most appropriate and productive locales for doing the sort of experimental programming that is involved in building a system like DINDE (and, we would argue, in carrying out statistical data analysis). This point is forcefully argued by Sheil [1983]. We have found the object-oriented programming paradigm, as available in LOOPS, to be especially useful: it is the backbone of DINDE (see Goldberg and Robson [1983], Bobrow and Stefik [1983], or Stefik and Bobrow [1985] for details on this programming paradigm).

In the object-oriented paradigm, there are classes, which contain generic properties and behaviours for a large "class" of individual objects, and objects, which are individual "instances" of a particular class. For example, the class *Car* would represent the common properties and behaviours of all *Cars*, while the object *MyCar* would be a particular instance of the generic class *Car*, as would *EdsCar*, *KarensCar*, and so on. In DINDE, each analysis step (goal or artifact) is represented as a class; the steps actually taken in a particular analysis are objects, instances of their corresponding classes.

The idea in DINDE, then, is to select the kind of step (class) which one wants to take next and to incorporate an instance of it (representing the step actually taken) at an appropriate place in the analysis (directed network).

The set of possible steps (i.e. classes) that can be taken in an analysis are displayed in an interactive window called the *toolbox*. Using the mouse, the analyst selects a step from the toolbox, as necessary, and attaches it (now an object) to the appropriate place in the analysis.

Recall that our model of an analysis is a directed network whose nodes are the steps actually taken. The analyst works with this model within another window called an *analysis map*. Here, the network representing the analysis is actually displayed and the analysis progresses by interacting with the network via the mouse.

This mouse interaction, called "mousing" in what follows, is possible in both the toolbox and the analysis map. In both windows the mouse behaviours follow some general principles. Figure 2 shows a generic window in DINDE and the basic mousing that can be done with them. There are two mouse sensitive areas in these windows: the title bar and the body. Mousing in the title bar allows interaction with the displayed contents as a whole; mousing on individual objects displayed in the body allows interaction with the mouse-selected object.

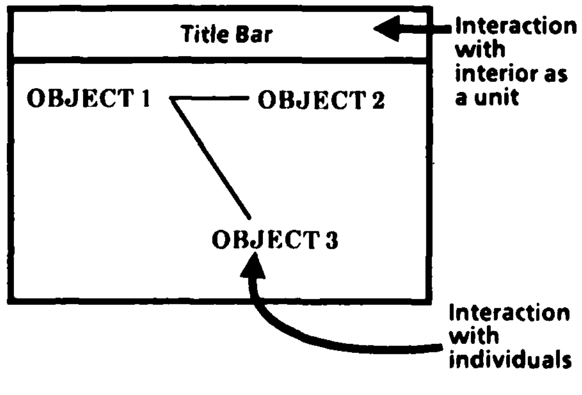

Title Bar
Interaction with interior as a unit
OBJECT 1
OBJECT 2
OBJECT 3
Interaction with individuals

Figure 2: Mouse sensitive areas of DINDE windows

The mouse that is used on the Xerox 1108 has two buttons, yielding three different combinations of depressed buttons: left, right, and middle (both buttons) depressed. Depressing any one of these typically causes a menu to pop up (at the mouse position) from which one of a number of items can be selected. Selecting an item causes some action to be taken.

We try to follow the principle that a button combination should yield a similar kind of menu regardless of what is being moused. So, in the toolbox and the analysis map, left-buttoning always produces menus whose items have something to do with either accessing or storing information on the thing being moused. If the title bar is moused, then the information pertains to the toolbox or analysis map itself. If an individual object in the body is moused, then the information pertains only to that particular object. Right-buttoning always brings up a menu containing items that allow the user to manipulate the window (move it, reshape it, shrink it, etc.). Middle-buttoning causes menus to appear whose items indicate the messages that the thing selected can respond to. Typically, these will be action items such as "fit a smooth curve to your points" if the thing selected is a *Scatterplot* object.

The toolbox and the analysis map are discussed in turn in the next two sections.

5.0 The Toolbox

All of the classes of objects that can be used in a statistical analysis in DINDE are arranged in the toolbox and displayed to the user. Figure 3 shows the contents of the toolbox as it currently exists in DINDE.

We have tentatively established a coarse partition of the possible classes into five basic element types: (i) *Data* (currently represented as *Arrays* or *TreeStructures*), (ii) *Graphics*, (iii) *Situations*, (iv) *Models* (e.g. probability models), and (v) *Tables*. To date, only the first three of these exist in DINDE. Only these were necessary to build the prototype regression analysis, but we anticipate that eventually *Model* and *Table* representations will also be required.

In the toolbox of Figure 3, the classes are displayed as nodes on several trees. Traversing the trees from left to right is equivalent to moving from generic steps to more specialized ones. For example, *BooleanArrays*, *StringArrays*, and *FloatArrays*, are all specialized *Arrays* (specialized to have array elements whose values must be booleans, strings, and floating point numbers, respectively). Once more tools are available in DINDE, other arrangements may be of interest (different indexing depending on interest). Indeed, it will likely become desirable to group tools together into smaller toolboxes, or toolkits, as the number of tools grows. At present, however, the classes are displayed only

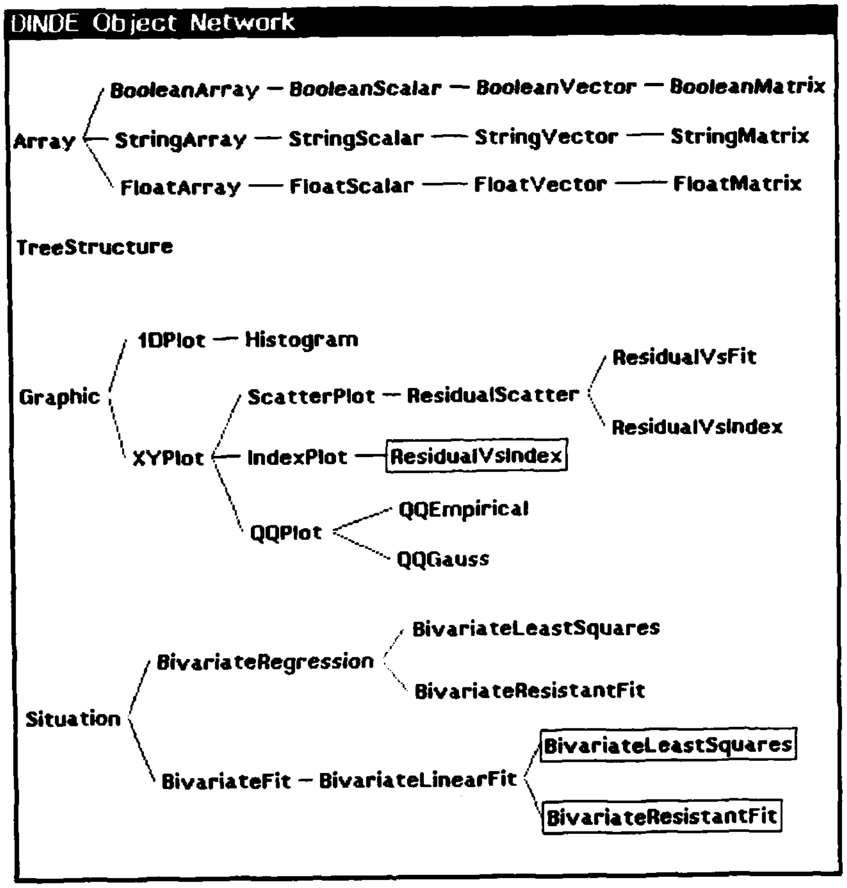

DINDE Object Network
Array
BooleanArray — BooleanScalar — BooleanVector — BooleanMatrix
StringArray — StringScalar — StringVector — StringMatrix
FloatArray — FloatScalar — FloatVector — FloatMatrix
TreeStructure
Graphic
1DPlot — Histogram
XYPlot
ScatterPlot — ResidualScatter
ResidualVsFit
ResidualVsIndex
IndexPlot — ResidualVsIndex
QQPlot
QQEmpirical
QQGauss
Situation
BivariateRegression
BivariateLeastSquares
BivariateResistantFit
BivariateFit — BivariateLinearFit
BivariateLeastSquares
BivariateResistantFit

Figure 3: The DINDE Toolbox

according to their specialization.

In general, a specialization has access to all the information and actions available to a more general step, and more. Thus, what seems to be a counterintuitive relationship between a *FloatMatrix* and a *FloatVector* makes sense, because a *FloatMatrix* has access to actions, such as taking the singular value decomposition of itself, which would make little sense for a *FloatVector*.

Those specializations which have been undertaken to increase the power of steps are more intuitive. For example, *ResidualScatter* is a specialization of *Scatterplot* which always has residuals plotted along the vertical axis. Given this context, *ResidualScatter* has access to actions which would not make sense for an arbitrary *Scatterplot*, like the ability to smooth the positive and negative values of the vertical coordinates (although smoothing all the vertical coordinates will be available to both kinds of plots). Similarly, the ResidualVsFit plot has access to actions which are helpful in determining whether the residual error is heteroscedastic.

Notice that these relations between the classes do not really conform to trees. LOOPS allows one to create classes that are specializations of more than one class (e.g. *ResidualVsIndex*). Such classes are identified by having a box drawn around them whenever they are repeated in the display. They have access to all information and actions that are available to any of their "parent" classes.

This ability to mix together classes encourages the abstraction of common aspects of different kinds of analysis steps. The abstractions are then represented as generic classes that can be usefully "mixed into" more than one step. This is perhaps one of the most challenging aspects of creating a sophisticated system like DINDE. It requires an identification and grouping of the elements that are practically important in a statistical analysis.

The nature of this research can be seen by considering those analysis steps that are classified as *Situations*. There are five different classes, of which only one can really be regarded as a goal (*BivariateRegression*), the others are better described as artifacts.

The analysis step *BivariateRegression* represents the goal of regressing y on x. It contains the necessary information on which vector is y and which is x, has access to various plotting methods (*Scatterplot* and *Histograms* of y and x), various fitting procedures (a straight line via least-squares or an outlier resistant procedure and a running linear least-squares smooth). Selecting any of these actions will produce an artifact (a plot or a fit), and hence a new analysis step. There are four kinds of fit artifacts. In order of increasing specialization they are as follows: *BivariateFit*, which is used to represent arbitrary fits (such as those from smoothers), contains information like y, x, the residuals, and the fitted values, and has its own set of actions, including the production of various residual plots; *BivariateLinearFit* which, in addition to the information and actions available to it from *BivariateFit*, also contains a slope and intercept for the fitted line; *BivariateLeastSquares* and *BivariateResistantFit*, each of which has access to more specialized information that is relevant to its particular fitting procedures (e.g. R^2 or t-statistics for *BivariateLeast- Squares*). Other factorizations are certainly possible.

In the toolbox, information is available on any of these steps. Selecting the class in question with the left mouse button depressed produces the menu shown below in Figure 4.

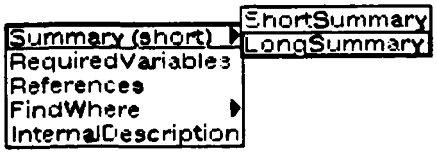


Figure 4: Left-button Menu for a Class

By selecting the appropriate item with the mouse, the user can get a short, or long, summary on that class, find out what variables it requires, get relevant references to the statistical literature, find out the classes from which it gains access to its information and actions, and a description of its internal software structure. (Note that a right arrow on a menu item indicates that a more detailed menu can be had by sliding the mouse to the right, along that item.)

Left-buttoning in the title bar of the toolbox produces a menu offering the user information on the toolbox itself, its contents, and the associated mouse behaviours. Other mouse buttons manipulate the display.

Rarely, in the course of an analysis, should the user need to retrieve tools directly from the toolbox. If the classes are defined well, then the tools made available through the actions at any given step should suffice.

6.0 Analysis Maps

Figure 5 shows an analysis of the relationship between the average brain and body weights of 62 mammals (taken from Becker and Chambers [1984]). Each node in the network represents a step in the analysis. The label displayed at each node is supplied by the object it represents, and consists of the name of the object (if there is one), its class, and, occasionally, capsule information on its contents (e.g. *BodyWts* is the unique name of a particular *FloatVector* object having 62 elements).

Mousing on a node permits interaction with the object it represents. For example, selecting a node with the left button down causes the menu of Figure 6 to appear. This menu allows information to be either added to, or, retrieved from, the selected object.

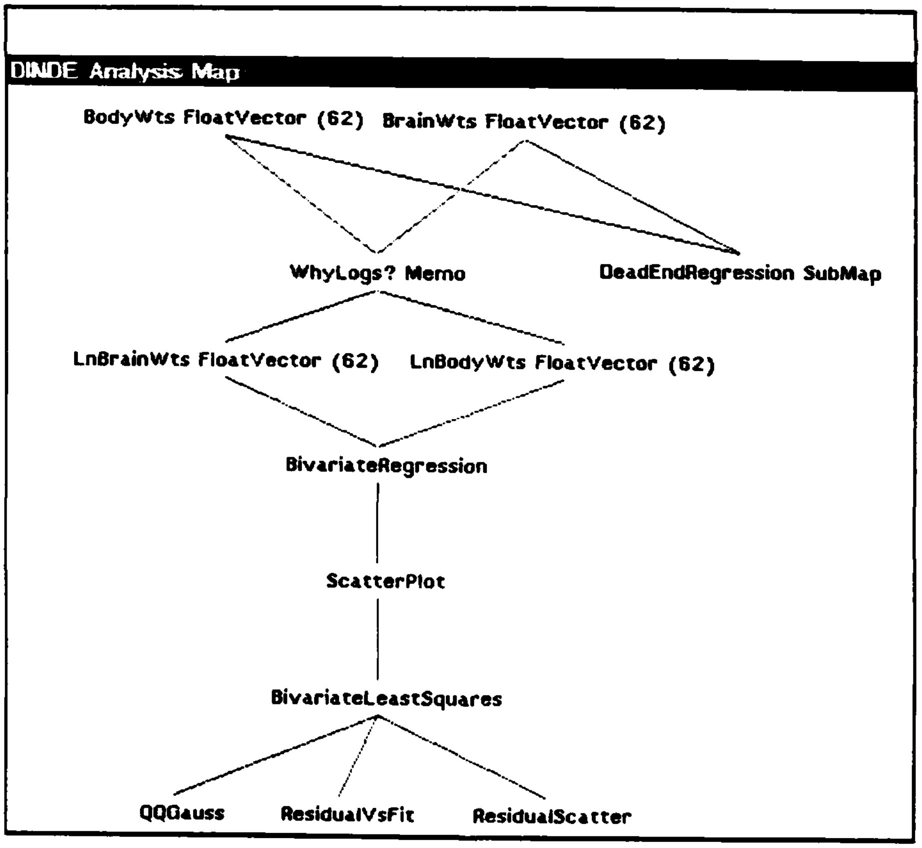
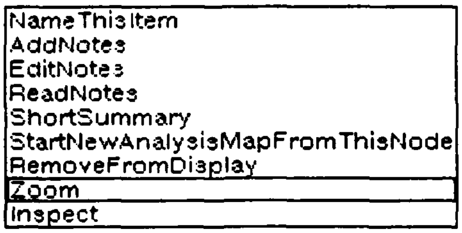


Figure 5: An Analysis Map



NameThisItem
AddNotes
EditNotes
ReadNotes
ShortSummary
StartNewAnalysisMapFromThisNode
RemoveFromDisplay
Zoom
Inspect

Figure 6: Left-button Menu for an Object

Information is added either by giving the object a meaningful and unique name, or, by adding notes to the object (the word processing capability of Interlisp-D used to construct this paper is made available). Both of these can be used to make the analysis easier to understand: the name at the node can make the display easier to follow, and the notes can record the analyst's observations on some facet of the analysis.

We consider the ability to make notes to be important enough that we include *Memo* objects as possible steps in the analysis. In Figure 5, a *Memo*, called *WhyLogs?*, was inserted between the two original data vectors, *BodyWts* and *BrainWts*, and the two derived *FloatVectors*, *LnBodyWts* and *LnBrainWts*. The latter two are the natural logarithms of the raw data, so the *Memo* is used to record the reasons for making the transformation.

Information is accessed in a variety of ways: by reading the user-recorded notes (ReadNotes), by printing a short summary on the class of the object (ShortSummary), by inspecting the internal program structure of the selected object (Inspect), and, by examining the detail contained in that node (Zoom).

The last of these is uniformly used in DINDE to access further detail on any node in the analysis. "Zooming" on a *Graphic* (e.g. *Scatterplot* or *QQGauss*) will cause a window containing the plot to appear. Zooming on other objects produces a window containing the "Zoomed" object and the data it can access. Figure 7 shows the effect of Zoom on the *BivariateLeastSquares* node of the map in Figure 5. (As in other DINDE windows, the mouse provides convenient way interaction with the displayed objects.)

The most important use of the Zoom facility is to retrieve the details of some sub-analysis. Sub-analyses can be represented in DINDE as objects called *SubMaps*, an example being the one named *DeadEndRegression* in Figure 5. Zooming in on this *SubMap* produces the analysis map of Figure 8. Except for its contents, this map is identical in every respect to that of Figure 5; both are instances of the same class.

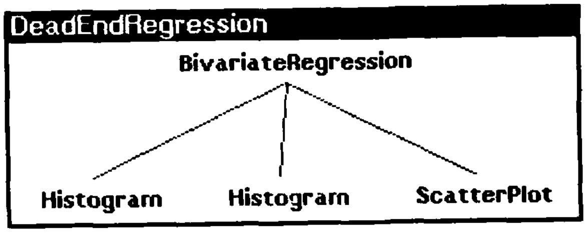
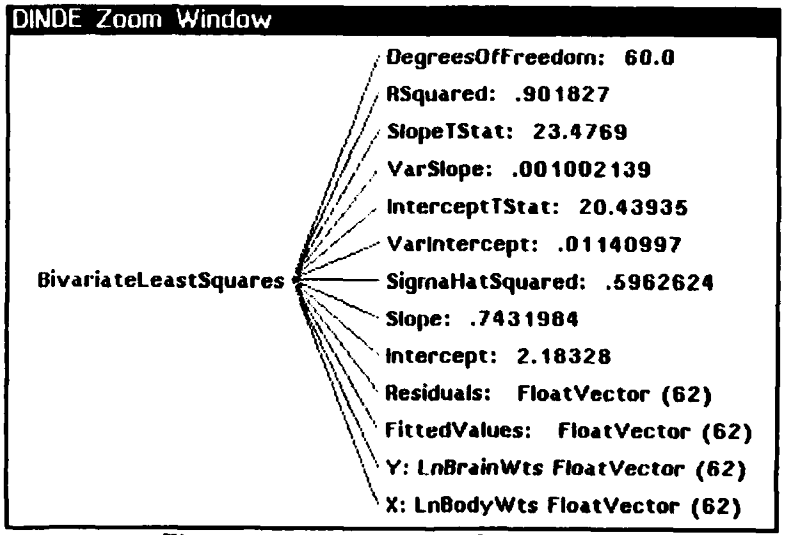


Figure 8: Zoom on a *SubMap*

The analyst can focus attention on this map, continuing the analysis there, without affecting the contents of any other map. Indeed, this map might also contain *SubMaps*, each representing a yet finer sub-analysis. These, in turn, may contain others, and so on, so that the whole analysis, in DINDE, is actually a directed network whose nodes might also be networks, each one representing a new level of detail.

The inverse of Zooming, in DINDE, is Compression. Middle-buttoning on the title bar of any analysis map produces a menu whose items correspond to operations on the displayed network. One of these items is Compress. Once selected, the user is required to identify nodes in the analysis (usually by mouse-selecting them) to be compressed into a single *SubMap*. All of the relationships between these nodes are maintained, so that Zooming on the new *SubMap* will reproduce the necessary detail.



DegreesOfFreedom: 60.0
RSquared: .901827
SlopeTStat: 23.4769
VarSlope: .001002139
InterceptTStat: 20.43935
VarIntercept: .01140997
SigmaHatSquared: .5962624
Slope: .7431984
Intercept: 2.18328
Residuals: FloatVector (62)
FittedValues: FloatVector (62)
Y: LnBrainWts FloatVector (62)
X: LnBodyWts FloatVector (62)

Figure 7: Zoom on *BivariateLeastSquares*

The other middle-button menu items from the title bar include the following: AddAnalysisNode which allows the user to select a new step from the toolbox and attach it anywhere in the network, MakeLink and BreakLink wich allow the user to make and break links in the network in order to make the analysis easier to understand, and, InterposeMemo, which allows the user to insert a memo between two nodes in the network. Together with Compress, these network tools should enable the analyst to construct an analysis of arbitrary complexity, whose display can be understood without much difficulty.

It is not necessary to select each new step in the analysis from the toolbox. Middle-buttoning on a step, already in the analysis map, will produce a series of menus that contain the actions that step can take. The consequence of many of these actions is a new step that is attached to the selected one.

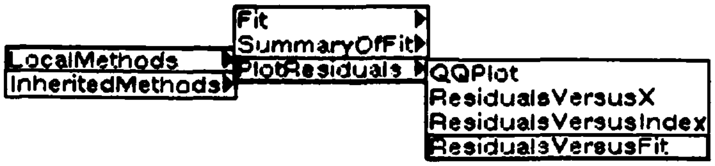


Figure 9: Some Menus from *BivariateLeastSquares*

Figure 9 shows some of the menus of actions that are accessible from a *BivariateLeastSquares* step (as in Figure 5). Here, the mouse has been moved to the right over the item LocalMethods, in the first menu, and over the item PlotResiduals, of the second menu, to display a menu of the possible residual plots that can be performed. The three steps *QQGauss*, *ResidualVsFit*, and *ResidualScatter*, of Figure 5, were produced by selecting from this menu the items QQPlot, ResidualsVersusFit, and ResidualsVersusX, respectively.

All steps in the analysis have access to menus that make sense for that step, and, typically, the analysis is constructed by selecting items from these menus. The step *BivariateLeastSquares* was produced by selecting the item AddALeastSquaresLine from a similar set of menus on the Scatterplot in Figure 5.

The system of menus available at each step makes the analysis easier to carry out; it does not restrict it. At any time, the analyst has access to a wide range of possibilities. For example, the analysis can proceed from any step in the network, including *SubMaps* and their contents, not just from the most recent one. Alternatively, a new step can be selected from the toolbox and added anywhere in the network. Finally, the menus can be ignored entirely and the analysis performed in the Interlisp/LOOPS environment using DINDE objects as they seem useful. DINDE never restricts access to the underlying programming environment, which means that the powerful tools we have available to us, as system builders, are also available to the user, as an analyst.

7.0 Concluding Remarks

We began by suggesting that there are three interrelated objectives that one might have for statistically sophisticated software: Guidance, Management, and Strategy. We close by pointing out how DINDE pays some attention to each of these.

The guidance in DINDE is minimal and quite local in nature. This is in keeping with our view as to what sort of guidance it is possible to competently give in a statistical analysis (see Oldford and Peters [1985a] for further discussion).

The guidance consists of the identification of the useful steps in an analysis, made available in the toolbox, and, of the actions and suggestions made available via menus at each analysis step. At present, the guidance is data-independent, in the sense that it does not depend on the data in hand. This does not rule out the possibility of data-dependent guidance, provided it too is of a quite local nature (e.g. noticing collinearity in regression).

Management of the data analysis is made easier in DINDE by having the analyst work with identifiable steps within a network metaphor for a statistical analysis.

With identifiable analysis steps, menus can be used to make those actions, which are often taken next, immediately available to the analyst. Further, notes can be added at each step, or inserted between steps, which will help the analyst recall the logic of the analysis. The computing environment also allows the notes, plots, and even snapshots of parts of the analysis, or individual analysis steps, to be inserted directly into a report.

The network paradigm is used to organize the steps. Much of this organization is done automatically at each step. When it is not, tools are available to organize the steps as the analyst sees fit. These tools include the ability to Zoom in on analysis steps, to yield detail, and, to Compress many analysis steps into a single *SubMap*, to suppress detail. Arbitrarily many levels of detail are thus available to the analyst. Finally, by actively using the network paradigm during the analysis, as opposed to after the analysis, the analyst is encouraged to organize the analysis as it proceeds.

As regards strategy, DINDE is based on our present view of what constitutes statistical strategy, and, on how that strategy can be fruitfully studied with software. We begin with a simple, yet general, model of statistical analysis: the directed network.

How a statistical strategy is implemented within this framework depends on the level, in the overall analysis, at which it is expected to operate. For example, one low-level strategy might be a heuristic used to determine the outlying points in a plot, while a high-level strategy might address the organization of a multiple regression analysis (see Oldford and Peters [1985ab] for further discussion).

High-level statistical strategy involves specifying the basic elements of the network model: the nodes, or analysis steps, and, the links which join them. In DINDE, the steps are represented as objects, and each object contains a specified set of actions that can be taken at that step. Thus, the analyst is strategically encouraged to link the present step to steps which result from taking the offered actions. Even stronger encouragement is provided through Suggestions, an action which produces canned text on what it is often considered wise to do first at this step. A low-level strategy is more likely to be implemented as an available action at some step.

Of the three interrelated objectives, that of using software to study the strategies of practical statistical analysis has been our primary focus. It is hoped that software, like DINDE, might provide a useful tableau on which statistical strategies can be recorded, and hence studied. To this end, we see that the model used in DINDE can be improved in two ways.

First, the fundamental data types in statistical practice are not vectors or matrices; these are artifacts of the mathematical analysis. More statistically meaningful are records on individuals, batches of numbers associated with a variate, and the dataset formed by combining many records or many batches. We are currently working on implementing such statistical objects as the fundamental *Data* objects in DINDE. (This in no way precludes the use of matrices and vectors to carry out the arithmetic.)

Second, the directed network model of analysis is not quite rich enough to suit our purposes. The difficulty is that not all links have the same meaning. For example, consider again the analysis map of Figure 5, and compare the links between the *FloatVectors* and the *Memo*, with those between the *BivariateLeastSquares* and the residual plots. Should the first set of links be considered to be as strong as the second? The first set were made by the user inserting a *Memo*, while the second set were a direct consequence of actions taken at the *BivariateLeastSquares* step. Suppose one would like to repeat a path in the analysis, with different values for the data (a sensitivity analysis perhaps), then it becomes important to distinguish between those links which are necessary to the analysis and those which are merely convenient. With the simple network, where all links are the same, this is not possible.

In closing, we feel that the three objectives in Figure 1 can be fruitfully pursued together. We also feel that real headway can be made on any of these by basing the software on a model of statistical analysis that is reasonably accurate, and, natural to use. DINDE is one such attempt that is currently based on a simple network paradigm for statistical analysis. As we improve the underlying model, DINDE will come closer to meeting these objectives.

8.0 References

Becker, R.A. and J.M. Chambers [1984] *S An Interactive Environment for Data Analysis and Graphics*, Wadsworth, Inc., Belmont CA.

Becker, R.A. and J.M. Chambers [1986] "Auditing Data Analyses", *Statistical Research Report No. 25*, AT&T Bell Laboratories, Murray Hill , NJ.

Bobrow, D.G. and M. Stefik [1983] *The LOOPS Manual*, Xerox PARC: Palo Alto, CA.

Box, G.E.P. [1976] "Science and Statistics", *JASA*, *71*, pp. 791-799.

Carr, D.B., P.J. Cowley, M.A. Whiting, and W.L. Nicholson [1984] "Organizational Tools for Data Analysis Environments", *Proc. A.S.A. Stat. Comp. Section*, pp. 214-218.

Gale, W.A. and D. Pregibon [1982] "An Expert System for Regression Analysis", *Proc. 14th Symposium on the Interface*, pp. 110-117.

Goldberg, A. and D. Robson [1983] *Smalltalk-80: The Language and its Implementation*, Addison-Wesley: Reading, MA.

Oldford, R.W. and S.C. Peters [1984] "Building a Statistical Knowledge-based System with Mini-MYCIN", *Proc. A.S.A. Stat. Comp. Section*, pp. 85-90.

Oldford, R.W. and S.C. Peters [1985a] "Implementation and Study of Statistical Strategy", to appear in *Artificial Intelligence in Statistics - Statistics in Artificial Intelligence* (W.A. Gale, editor), Addison-Wesley, Reading MA.

Oldford, R.W. and S.C. Peters [1985b] "DINDE: Towards more statistically sophisticated software", *CCREMS Technical Report # 55*, MIT, Cambridge, MA.

Pregibon, D. [1985] "A DIY Guide to Statistical Strategy", to appear in *Artificial Intelligence in Statistics - Statistics in Artificial Intelligence* (W.A. Gale, editor), Addison-Wesley, Reading MA.

Sheil, B. [1983] "Power Tools for Programmers", *Datamation*, February issue, pp.131-143.

Stefik, M. and D.G. Bobrow [1985] "Object-Oriented Programming: Themes and Variations", *The AI Magazine*, *5*, pp. 40-62.

Stefik, M., D.G. Bobrow, S. Mittal, and L. Conway [1983] "Knowledge Programming in LOOPS: Report on an Experimental Course", *The AI Magazine*, *3*, pp. 3-13.

Teitelman, W. and L.M. Masinter[1981] "The Interlisp programming environment", *IEEE Computer*, *14*, pp. 25-34.

9.0 Acknowledgements

This research was partially funded by the National Science Foundation grant IST-8420614 to the M.I.T. Center for Computational Research in Economics and Management Science. Computer facilities were provided by the M.I.T. Statistics Center.

RECENT DEVELOPMENTS IN MULTIVARIATE DATA ANALYSIS GRAPHICS

Organizer: *Daniel B. Carr, Pacific Northwest Laboratory*

A Data Viewer for Multivariate Data

Andreas Buja, University of Washington, Seattle, and Bell Communications Research; Catherine Hurley, University of Washington, Seattle; John Alan McDonald, University of Washington, Seattle

Interactive Color Graphic Display of Data by 3-D Biplots

K. Ruben Gabriel, Anup Basu, Charles L. Odoroff, Terry M. Therneau, University of Rochester

A Data Viewer for Multivariate Data

Andreas Buja
University of Washington, Seattle, and Bell Communications Research, N.J.

Catherine Hurley (speaker)
University of Washington, Seattle.

John Alan McDonald
University of Washington, Seattle.

Abstract

The Data Viewer is a system for the exploratory analysis of large, high-dimensional datasets, being developed on a Lisp Machine. Suppose we have a multivariate dataset consisting of up to 1000 observations on an (arbitrarily large) number of quantitative variables, how can we examine it? The data viewer tackles this problem using Grand Tour techniques: by moving projection planes it displays a scatterplot "movie". Design issues are crucial in the development of this system, in particular with regard to questions of user interface. The Lisp Machine supports object-oriented programming and the use of constraints, and these features are influential in our implementation.

Moving Scatterplots

Any two dimensional orthogonal projection of the data can be displayed as a scatterplot. A moving scatterplot results when the projection plane is modified every fraction of a second. If this is done as often as 10 times per second then the scatterplot appears to be in continual motion. Clearly, this means that a user may see more data views in a shorter time period; whether this is helpful or not depends on how the updating is done. If successive plots differ substantially then the advantage of dynamic plotting is lost: a user has difficulty assimilating even one new plot a second. However, if the scatterplot appears to move in a continuous manner then we gain by seeing more data, and by seeing it move. A static plot is confined to displaying two dimensions, whereas motion presents an additional two dimensions of information, as given by the speed vectors. This allows a perception of depth and shape of the pointcloud.

Creating Moving Scatterplots

A moving scatterplot is constructed by generating a family of 2-planes called views $\{v_t, t \geq 0\}$, where t represents time. At each time, the corresponding scatterplot is the projection of the data onto v_t. As t is increased in small increments, the view v is updated, and the new point coordinates recomputed and replotted. In order that the scatterplot appear to move smoothly the family of views need to fullfill certain requirements, as discussed in Asimov. The scheme which we have successfully implemented operates by constructing a sequence of views $\{v_i, i = 1, 2 ...\}$, by random sampling, for example. Then interpolation is done between successive views to obtain $\{v_t, t \geq 0\}$. Methods have been derived for interpolation between successive views such that the illusion of continuous motion is preserved. In order that moving plots be a technique adaptable to a wide range of data analytic situations, the data viewer provides a number of schemes for generating this sequence of views, and allows consideration to be reduced to more specialized sets of views.

Specialized Views

The most general kind of view is an arbitrary 2-plane, and the corresponding plot is the linear projection of the p-dimensional variable space on to this plane. This means that no particular orientation of the plot is specified. If a subset of the variables is of particular interest to the user, he can indicate this by classifying these variables as "active" and the remainder "inactive". Only active variables can have non-zero projection coefficients, so this achieves a reduction in dimensionality of the space being explored.

Specialized views arise when the data to be displayed consists of two disjoint groups of variables, of sizes p and q respectively, described by the matrices $X_{n \times p}$ and $Y_{n \times q}$. An example of this is the canonical correlation situation. Then the views of interest consist of linear combinations of the X variables plotted versus linear combinations of the Y variables, so that consideration can be reduced to particular kinds of 2-planes. Notice here also that an orientation of the plot is specified. Views of this type are obtained when variables are classified as either "X-variables", "Y-variables" or inactive, so that a variable cannot have more than one X or Y non-zero projection coefficient.

This setup has a number of important special cases. Variable scatterplots arise when $p = q = 1$. Regression also fits into this framework, where there is one response ($q = 1$) and p predictor variables. The data-viewer user controls the type of view generated by attaching classification labels to each variable, "X", "Y" and "A" for X, Y and active variables; otherwise the variable is rendered inactive.

Schemes for Generating Views

1. Scanning

A brute force approach to examining a high dimensional pointcloud is to look at all possible projections onto planes. Schemes for scanning high dimensional space are termed Grand Tour methods (Asimov, Buja). One such method approximates all possible data views by sampling planes uniformly and interpolating between them. So, with some time and patience a user may come arbitrarily close to every 2-d data projection. We have used this scheme successfully with up to 1000 data points, and there is no restriction on the number of variables. In practice, human vision and understanding limits our perception to 3-d, so discerning higher dimensional structure can be difficult.

2. Optimization

We use projection pursuit techniques to augment the scanning procedure. The aim is to avoid views that look like featureless "blobs", concentrating instead on projections showing structure. Commonly, context knowledge and information gained from the analysis to date dictate how the analysis should proceed, so that exploration is done with particular questions in mind. For example, are there relationships between groups of variables, are outliers present etc. It follows that he wishes to see views that help provide answers to these questions. The data viewer can cater to this situation by displaying a movie of informative views. The user selects (or defines) a measure F of how interesting a view is, then the sequence of views is chosen such that $F(v_1) \leq F(v_2) \leq .....$ Gradient methods are used to obtain successive views. Suggestions for suitable choices of F are given in Friedman & Tukey, and Huber. This scheme corresponds to an interactive, real-time optimization, interactive because both the starting plane and stepsize are under direct user control. This has the advantage that typical sources of difficulty with optimization methods, for example, the presence of function "flat spots",are no longer a problem. Also, since our emphasis is on exploration, finding views which are local maxima suffices.

3. Other Schemes

Another method for generating a view sequence permits exploration in a neighborhood of the current view. For example , this enables us to discover how sensitive the projected plot is to local changes in the optimizing view. Provision is also made for user specified views, and variable scatterplots are particularly easy to obtain.

To summarize, the essential capability of the data viewer is the moving scatterplot. The user has a measure of control over the motion via a choice of algorithms for generating views, and specification of the type of view to be generated. Other data viewing tools can be considered in one of three categories .

(i) Tools particular to the data viewer :
For example, a backtracking capability, which provides the ability to rewind or to play back the scatterplot movie, is useful. We would also like the ability to save away views for retrieval and re-examination at a later stage.

(ii) General graphical capabilities :
This includes multiple plots for easy comparisons, interactive painting (or brushing) and point identification utilities.

(iii) Data related tools :
This includes facilities such as variable scaling and transformation, and subset selection.

Design of a Data Viewer

As outlined above, the data viewer supports a high degree of functionality. It should be extendable to include other statistical methods, for example new view generating schemes. We aim towards a system with a unified graphical interface to simplify user communication. These requirements necessitate a careful system design. The data viewer is being developed on a Lisp Machine, which has the computational power necessary for the demanding task of scatterplot motion, and provides an environment and language features that encourage experimentation (McDonald & Pedersen).

Our approach to the design of the data viewer is to factor the system into various components and sub-components. Each component has its own specific task with well-defined connections or interfaces between them. This design can aid both system programmer and user. From the programmer's point of view it is highly modular, so that different components can be implemented independently except for the specified connections. It also means that a certain amount of flexibility is inherent in the design: components can be modified internally, for example a new view generating scheme added, without affecting the rest of the system. This design helps to provide a coherent user model, reducing the initial time overhead necessary to gain familiarity with the system. Also, a knowledge of the various components and their interrelationships means that the capabilities of the system are better understood; its power and flexibility should be transparent, not obscure to the user.

Object Oriented Programming and the Data Viewer

The Symbolics Lisp Machine provides the "Flavors" extension to Lisp for a style of programming called "object oriented " (Weinreb and Moon). An object consists of data and procedures bundled together. (In Flavors terminology these are termed respectively the "instance variables" and "methods" of the object.) One way to think of it is as a record in Pascal or structure in S, with its own internal procedures. The object oriented programming style is by nature modular, because it allows a data structure and the procedures that operate on it to be considered a single entity. The Flavors system gives us a natural way to factor the data viewer into components: an object is created corresponding to each component.

We can regard the data viewer as an object for displaying data views. Its components or instance variables are:

a dataset

a "view-maker"; an object whose task it is to supply views or projection planes.

a plot; a window object which shows the moving point-cloud.

The data viewer has a "draw-plot" method, which carries out its primary task. Draw-plot applies the projection from view-maker to the dataset and shows the result in the plot window. This is a one sentence explanation of the operation of data viewer. A further level of detail is got by considering the data viewer components and their respective tasks.

The "view-maker"

The view-maker object generates new views, so it has a "new-view" method for this task. In order to obtain a new view the following information is needed:

the current view

an algorithm (user specified) for generating views

the (user specified) classification of each variable i.e "X" "Y" "A" or " " (inactive)

speed; the amount by which the view is incremented.

These items are represented as instance variables of the view-maker. The new-view method updates the current view by the amount "speed", according to the specified algorithm and variable classifications.

The plot window

The plot has two major components, which are:

a point-cloud; the projection of dataset onto the current view

variable-boxes; for each variable in dataset the following information is displayed:

- the variable name
- the coefficient applied to this variable in the current view
- the variable classification i.e. "X" "Y" "A" or " ".

We can think of the plot window as being a mirror of the dataset and view-maker objects. This we refer to as the "display constraint". The point-cloud reflects the dataset and the view supplied by view-maker. The variable-boxes display the variable names from the dataset, and the variable classifications and projection coefficients from view-maker. This mirror analogy for the plot has some powerful consequences: the plot always shows the current state of the underlying objects. So, changes to the dataset or view-maker are immediately reflected on the screen. Motion results as a trivial consequence of the display constraint; changes in the view supplied by view-maker imply a corresponding update of the displayed point-cloud and projection coefficients. This means that except for the display constraint, the plot window and the pair of objects view-maker and dataset operate independently. Once this part of the user model is grasped, the user becomes aware of some interesting data-analytic applications.

Some implications of this design.

A typical data analysis involves the comparison of two or more datasets, for example males and females. One way of doing this is to put a number of plots each in its own window together on the screen, and examine, say, height plotted against weight for each set. If a data viewer is created corresponding to each dataset, and these viewers have a common view-maker object, then both displays will simultaneously reflect the same view-maker object. At a particular time t, one display shows the males dataset projected onto the corresponding view v_t, and the second display shows the projection of the female dataset, also onto v_t. When the view is modified by view-maker, both displays update themselves to reflect this change. This dynamic dataset comparison makes it possible to explore questions such as: are interesting views common to datasets? We describe these data viewers as being linked or connected by a shared view-maker object.

Similarly, two or more data viewers can be linked via a common dataset object, each with its own view-maker. Suppose the dataset object contains a component which defines a color (or any drawing symbol) for each case. The display reflects this information by drawing each point with the appropriate color. Then a shared dataset object means that two data viewers show different views of the same data, but with points linked by color.

For further discussion on the applications of automatic updating see Stuetzle.

A scheme for user interaction.

How can user interaction be explained by this model? Suppose a user wishes to identify the outlier in a scatterplot. An obvious solution is to make use of the pointing device (a mouse) to indicate the particular point, and have the case identification label appear in response. We interpret this action by realizing that we are communicating with underlying dataset "case" via its plotted representation, a point on the screen. So, the role of the display is extended from being simply a dumb reflection, to provide us with communication routes to the constituent objects.

Communication with data viewer components can also include modification. For example, the plot window has a dial or gauge which shows the value of the view-maker speed component. This gauge is what is termed "mouse sensitive", so that it can respond to mouse clicks by changing its value. In fact, what changes is the view-maker speed value, and the display modifies itself to remain up to date. A similar scheme can be used for changing variable classifications: variable boxes are mouse sensitive and provide hooks into the variable classification quantities which are part of the view-maker. Direct manipulation is the term given to this style of interaction, because the illusion is that the plotted representation or icon is actually the underlying data item, not just a picture of it. One gets the feeling that the plot window is somehow alive, and has knowledge, so communication can be more expressive.

Conclusion

A moving scatterplot is both an efficient and informative method for displaying high-dimensional pointclouds. The data viewer system enables this technique to be used for purposes of exploration. A variety of view generating schemes, and the capability to specialize views mean the user can adapt the system to the data analytic task at hand. In designing the data viewer we aim towards a coherent user model; this can aid system development and extension, and more significantly, its utility. The notions of object oriented programming provide the basis for our design. The user communicates with the system via directly manipulating plotted items. These features combine to make the data viewer good for a large range of approaches to exploration.

References

Asimov, D. (1985) "The Grand Tour: A tool for viewing multidimensional data", *SIAM J. Scientific and Statistical Computing 6(1), 128.*

Buja, A. (1985) "Grand Tour Methods: An Outline" (joint with D. Asimov). *Proceedings of the 17th Symposium of the Interface of Computer Science and Statistics*, Springer Verlag.

Huber, P.J. (1985) "Projection Pursuit", *Annals of Statistics,* vol 13, no. 2.

Friedman, J., Tukey, J. (1974) " A projection pursuit algotithm for exploratory data analysis" *IEEE Trans. Comp.*, C-23, 881-890.

McDonald, J.A., Pedersen, J., (1985) "Computing Environments for Data Analysis, Parts 1&2", *SIAM J. Scientific and Statistical Computing*. 6(4), 1004-1021.

McDonald, J.A., Pedersen, J., (1986) "Computing Environments for Data Analysis, Part 3", to appear in *SIAM J. Scientific and Statistical Computing*.

Stuetzle W. (1986) "Plot Windows", preprint, Statistics Department, University of Washington, Seattle.

Weinreb D., Moon D., (1985) Lisp Machine Manual.

INTERACTIVE COLOR GRAPHIC DISPLAY OF DATA BY 3-D BIPLOTS

K. Ruben Gabriel, Anup Basu, Charles L. Odoroff and Terry M. Therneau +
University of Rochester (+ Now at Mayo Clinic)

I. BIPLOTS. This paper discusses our experience in displaying data as 3-D biplots by means of two alternative graphics devices, (1) standard high resolution graphics terminals with attendant printer plots or pen plotters, (2) the Raster Technologies Model ONE/20 frame buffer color device with attendant black-and-white laser printer.

Biplots (Gabriel, 1981a) are particularly effective for exploring multivariate data matrices and for diagnosing models that fit these data or subsets of them. A 3-D biplot displays a rank three approximation to the data matrix (which is usually centered) by means of row markers a[i], i = 1,...,n (= the number of rows of the matrix) and column markers b[j], j = 1,...,m (= the number of columns of the matrix). The lower rank approximation may be obtained by least squares or by resistant methods (Gabriel and Odoroff, 1984a). The markers are obtained by factorizing the approximation as AB' and using the rows of A and B, respectively.

In view of this construction, the fundamental property of all biplots is inner product representation of the data, i.e., the inner product of any a[i] and any b[j] approximates y[i,j], the corresponding element of the data matrix (Gabriel, 1981b). Special types of biplots further approximate the variances and covariances of the columns by the configuration of of the column markers, and the Mahalanobis-type distances between the rows by the biplot distances between the row markers (Gabriel, 1971).

A particularly useful feature of biplots is that patterns of their markers may be used to diagnose the type of model that would fit the data, e.g., if the a[]'s were all on one line and the b[]'s on another line then the data would be fitted either by an additive model or by a one-degree-of-freedom-for-non-additivity model, depending on whether the two lines were perpendicular to each other or not. A limitation of biplots is that they are available only for data in matrix form. However, higher-way layouts may at times be usefully biplotted by collapsing them into a two-way table (Kester, 1979; Gabriel, 1981b; an example will be given below).

Biplots can display data only as well as the data can be approximated with lower rank. Planar (rank 2) biplots are often very useful, but 3-D biplots often do better (Gabriel, 1981b). We expect to do even better in higher dimensions and are studying the application of Banchoff's ideas (1986) for the displays of 4-D biplots, on which we hope to report at a later date.

Our 3-D biplots are produced by a number of techniques, each of which displays row markers and column markers, and allows these to be labelled to indicate the particular rows and columns represented.

II. DISPLAY DEVICES AND SYSTEMS. We have routinely produced these on standard graphics terminals by the BGRAPH system, due mostly to Mike Tsianco (1981). BGRAPH produces static displays of biplots by three dimensional perspective projections of a viewing cube or dodecahedron, and by using various depth cues. We have now implemented the ANIMATE system, due principally to Terry Therneau, which adds color and shape cues and is semi-animated in that it simulates the effect of depth by rapidly displaying successive views from slightly different angles (Odoroff et al, 1986). "Rocking" the picture back and forth through a series of about 6 such views may create an impression of three-dimensionality. (We distinguish this from a truly animated system, such as PRIM or MACSPIN, which is capable of calculating the coordinates of any number of views and displaying them in real time.)

TABLE 1: FEATURES OF THE BGRAPH AND OF ANIMATE SYSTEMS

	BGRAPH	ANIMATE
Markers	Points or vectors	Vectors, squares, circles or spheres with options of shading and color
Labels	Available	Available
Depth	Perspective cues on labels or Stereograms (needs stereo glasses) or Analglyphs (need color or polarized display and glasses)	Perspective cues on marker shapes Rocking through views of small angular separation Hiding of markers behind other markers
Rotations	Transformations to any viewpoint and angle (Not in real time)	Transformation to any angle (not in real time)
Zoom and Window	Available	Available
Mainframe	UNIX VAX 11/750	UNIX VAX 11/750
Display device	Graphics terminal (no local computing)	Raster Technologies ONE/20 24 frame buffer device which can store several pictures simultaneously and then display them in rapid succession
Hardcopy	Printer or pen plotter	Laser printer or slide photographs of screen

Attributes can be changed and displayed dynamically via keyboard or "mouse". ANIMATE makes use of certain features of the hardware, such as the ability to rapidly change the color table, and the ability to rapidly change the sector of the image memory displayed. However, it is constrained by certain features of the hardware such as the necessity to update displays from the remote host computer. We should like to ignore the constraints of hardware, but they will remain with us until the next generation of graphics workstations are available. The newer graphics workstations will not require some of the commands presently needed for ANIMATE. They will circumvent many of our present hardware limitations.

A command language to manipulate the graphics display is implemented in yacc, lex, and RATFOR on a VAX 11/750 with the UNIX operating system. The displays are built up from graphics primitives provided with the Raster Technologies Model ONE/20.

The basic strategy is to compute the display features on the host computer using the principles outlined in Newman and Sproull (1980). A set of pictures is written into the frame buffer memory, one at a time (about ten seconds per picture). When the host has completed computing all the displays, control is returned to the frame buffer and the pictures are displayed in rapid succession to simulate motion.

The Raster Technologies Model ONE/20 is a frame buffer device which displays a picture of 512 by 512 pixels on a color video monitor. Each pixel is addressed by a 24 bit word. The 24-bit word can partitioned flexibly to trade more pictures for fewer colors. By using hardware pan and zoom functions additional pictures can be obtained by sacrificing picture resolution. By these artificial means, ANIMATE can store either 192 pictures in four colors or 3 pictures in 256 colors. Colors are chosen from a palette of more than 16 million.

The frame buffer device provides the ability to use a large palette of colors in displays and since it is a raster device, motion can be simulated by animation. The host computer computes the attributes of the display and communicates via a 9600 baud communication link to the frame buffer device. As noted, up to 192 pictures may be written into the frame buffer memory before control is handed to the display processor. The system is thus limited in the rapidity with which a display can be changed by the speed of the host computer and the communication line. Modifications of color, choice of the sector of frame buffer displayed are rapid; modifications requiring the host to compute a new picture are slow. When more local processing power is available this problem will be solved.

We have found that 6-24 pictures displayed at ten frames per second in the frame buffer are adequate to simulate motion. We also have available the use of perspective, eclipsing of points by their neighbors, and the representation of points as illuminated spheres to aid in the simulation of depth.

III. SOME EXAMPLES. We now discuss several sets of examples, some of real data and some of artificially generated matrices. Each example told us something about the graphics systems and the usefulness of their various features. At the Interface Symposium we accompanied our paper with a series of color slide photographs of displays on the ONE/20 screen. We tried to imitate the rocking of views on the device by rapidly flipping back and forth through a set of slides in a projector carousel -- we hope this gave a sufficiently clear impression of the capabilities of the ANIMATE system. We cannot hope to achieve the same impression in this black and white printed report, so we show only a small number of laser printer reproductions of some of the slide views. In view of these difficulties of reproducing the semi-animated color display, the discussion of the examples will be very brief and the conclusions which we state will have to be taken on trust until the reader has an opportunity to see a demonstration of the ANIMATE system.

IIIa. THE IRIS DATA. For the well-known Anderson (1935) *Iris* data, we have a 150-by-4 matrix, the four columns being allocated to the variables of petal length and width and sepal length and width, the 150 rows to the 150 *Iris* flowers, the first fifty rows to *I. setosa* flowers, the next to *I. versicolor*, the last fifty to *I. virginica*.

The biplot of these data, after variable means had been subtracted, is shown in Figure 1 with vectors for the four variables' markers and individual *Iris* markers being circles of different shadings for *I. setosa* (the leftmost scatter), *I. versicolor* (the middle scatter) and *I. virginica* (the rightmost scatter). The configuration of the variables shows petal width and length to be very highly correlated with each other and highly with sepal length, but sepal width is not well correlated with any of them; the scatter of the flowers is seen to consist of clearly distinct clusters for the three species; the separation of the species' clusters is along the direction of the first three variables, showing that *I. setosa* is the smallest species on these measures, *I. versicolor* is larger and *I. virginica* somewhat larger, though the latter two are not as well separated from each other as they are from *I. setosa*.

Rotation of the biplot to another view emphasizes the species separation and shows *I. setosa* to be less variable than the other species. It also suggests that two outliers existed and points to a surprising crescent-shaped distribution of *I. virginica*.

WHAT WE HAVE LEARNED FROM THIS EXAMPLE. In exploring unlabelled data, we looked for shapes of distributions, for clusters and their separation, and for outliers. Graphical exploration of this kind was helped by the following features:

COLOR was very useful, as was SHAPE; (PERSPECTIVE was not tried on this example). ROTATION was crucial, and it needed to be fast to help in exploratory analysis. ROCKING did not add much to feeling of depth, HIDING did a little.

It seemed more important to be able to move a viewing plane through space than to get a feeling of depth, of space. WHY? Are the tools inadequate? Or are our questions essentially one and two-dimensional?

Study of covariance configuration needed LABELS. The lack of clear DEPTH cues hampered it more than it had hampered the study of scatters. Does this indicate that higher dimensional space is more important in studying configurations than scatters? Are we able to imagine covariance configurations spatially, but distributions planarly? Or is the effectiveness of more dimensions merely a result of the relative sizes of the collection of units (150) and variables (4). Can it be that we can visualize a few objects in higher dimensional space than a larger set of objects?

IIIb. TWO ARTIFICIAL 20-BY-15 DATA MATRICES. Figures 3 and 4 are biplots of two 20-by-15 matrices of data, generated from certain models (Gabriel and Odoroff, 1986a,b,c). Each has 20 column markers (dark spheres), and 15 row markers (light spheres).

No pattern is evident on Figure 3, but Figure 4 reveals a clear pattern, the column markers appearing to be on a an oblique plane. Comparing the two figures it is clear that it is use of perspective in Figure 4 that makes its pattern apparent. The pattern that generated Figure 3 is not at all apparent because that biplot was displayed without perspective.

Suitable rotations would show both biplots to have the row markers on one plane and the column markers on another. In Figure 3 the two planes are perpendicular, in Figure 4 they are not. It has been shown (Chuang, Gabriel and Therneau, 1984) that this diagnoses models r[i] + v[j] + t[i]w[j] and R + s[i]v[j] + t[i]w[j], respectively.

WHAT WE HAVE LEARNED FROM THESE EXAMPLES. It is possible to MODEL data by discerning patterns of biplot markers. Display by means of SHADED SPHERES is effective when PERSPECTIVE cues are used. This was a major aid in discerning patterns. Easy and rapid ROTATION and ROCKING are also of help.

IIIc. AN EXPERIMENT ON SOLAR WATER HEATING SYSTEMS. The last example uses the results of a four factor experiment (Close, 1967, quoted in Box, Hunter and Hunter, 1978). The data appeared naturally in a four-way layout -- high-low levels of I (insolation), high-low levels of S (size of tank), of W (water flow) and of D (discontinuity, or intermittency of sunshine). To biplot it, we arranged it in matrix form with factors I and S cross-classified in the four rows, and factors W and D cross classified in the columns. Before biplotting, the data were centered on the overall mean.

The biplot of these data -- Figure 5 -- has four row markers (dark spheres), labelled is, iz, js, and jz, for all combinations of the upper and lower levels i and j of factor I and upper and lower levels s and z of factor S. Also, it had four column markers (light spheres), labelled wd, wt, vd, and vt, for all combinations of the upper and lower levels w and v of factor W and upper and lower levels d and t of factor D. Inspection of this biplot, and of suitable rotations, allowed diagnosis of all the main effects and of the interactions between I and S (no interaction) and between W and D (which did interact). (For a detailed discussion of the logic of these diagnostics see Kester, 1979 and Chuang, Gabriel and Therneau, 1984, as well as Gabriel and Odoroff, 1986a,b,c.)

To inspect the IW, ID, SW, SD interactions we required markers for each factor averaged over the levels of the other factor of the pair. Construction of these averages on the biplot was therefore an important diagnostic tool. The resulting biplot -- Figure 6 -- has the markers for the averages displayed. Thus, for example, the average for d, the upper level of factor D, is displayed by label *d which is located midway between markers wd and vd. Similarly, half way between vd and vt, one should find the v* label which marks the upper level of factor V, but the v* marker is partly obscured by the sphere marker for is.

We now look at the (*d,*t) and the (w*,v*) directions, relative to the (i*,j*) and (*s,*z) directions. The (*d,*t) direction is found to be perpendicular to the IS-plane, and this allows us to conclude that D does not interact with I or S. On the other hand, upon rotating the biplot about the X-axis we find the (w*,v*) direction to be oblique to the IS-plane and so conclude that W does interact with I and/or S.

WHAT WE HAVE LEARNED: We cannot only model, but also observe effects, additivity and interaction in higher way layouts. To do so we obviously need LABELS and ROTATION. But a sense of DEPTH has also been needed. We found it most strongly suggested by the PERSPECTIVE cues on the SHADED SPHERES; ROCKING was less useful because the labels tend to jump about and make identification of individual markers very difficult.

Another most important help was the use of AVERAGE MARKERS. This allowed us to disentangle the effects of single factors which had originally been cross-classified with other factors. More generally, we think an important diagnostic aid to a graphics system is the ability to FIT MODELS ON THE PLOT itself. For example, if a set of markers is thought to lie on a plane, or line, it is extremely useful to be able to fit and display such a plane, or line, on the plot itself. Our system cannot do that yet.

IV. SUMMARY. *We can MODEL by using biplot graphics, and we can also EXPLORE data.*

The requirements for effective graphics are first of all DISPLAYS THAT ATTRACT THE EYE !! This is furthered by the use of COLOR, SHADING, and SHAPES.

DEPTH is also important for inspection of biplots, and it is most effectively simulated by PERSPECTIVE rather than by MOTION. That, at any rate, is the impression we got from using the BGRAPH and ANIMATE systems.

ANCILLARIES that we consider essential for any graphics system that is to be used for data analysis are: LABELS !!!, ROTATION !! and CAPABILITY TO MODEL ON THE DISPLAY !

ACKNOWLEDGEMENTS. This work was supported in part by the Office of Naval Research under Contract N00014-80-C-0837 on Biplot Multivariate Graphics (K.R.Gabriel, Principal Investigator).

REFERENCES

Anderson, E. (1935). BULLETIN OF THE AMERICAN IRIS SOCIETY, 59, 2-5.

Banchoff, T. (1986). In STATISTICAL IMAGE PROCESSING (E.Wegman and D.DePriest, eds.). New York, Dekker.

Box, G.E., Hunter, W.G., and Hunter J.S. (1978). STATISTICS FOR EXPERIMENTERS. New York, Wiley.

Bradu, D. and Gabriel, K.R. (1978). TECHNOMETRICS, 20, 47-68.

Close, D.J. (1967). SOLAR ENERGY, 11, 112.

Chuang, J.C., Gabriel, K.R., and Therneau, T.M. (1984). Geometrical diagnosis of linear/bilinear models with the biplot. Paper presented at the October 1984 ASA-IASC-SIAM Conference on Frontiers in Computational Statistics at Boston. (Paper in preparation.)

Gabriel, K.R. (1971). BIOMETRIKA, 58, 453-467.

Gabriel, K.R. (1981a). In ENCYCLOPEDIA OF STATISTICAL SCIENCES, Vol.I (S.Kotz, N.L.Johnson and C.Read, eds.). New York, Wiley, 262-265.

Gabriel, K.R. (1981b). In INTERPRETING MULTIVARIATE DATA (V.Barnett, ed.). London, Wiley, 147-173.

Gabriel, K.R. and Odoroff, C.L. (1984a). In DATA ANALYSIS AND INFORMATICS, III (E.Diday, M.Jambu, L Lebart, J.Pages and R.Tomassone, eds.). Amsterdam, North-Holland, 23-30. (Also in PROCEEDINGS OF THE FIFTEENTH SYMPOSIUM ON THE INTERFACE, 1983.)

Gabriel, K.R. and Odoroff, C.L. (1984b). Biplots for exploration and diagnosis. Examples and software. University of Rochester, Statistics Technical Report 84/04.

Gabriel, K.R. and Odoroff, C.L. (1986a). In STATISTICAL IMAGE PROCESSING (E.Wegman and D.DePriest, eds.). New York, Dekker.

Gabriel, K.R. and Odoroff, C.L. (1986b). In CLASSIFICATION AS A TOOL OF RESEARCH (W.Gaul and M.Schader, eds.). Amsterdam, North-Holland.

Gabriel, K.R. and Odoroff, C..L. (1986c). In PROCEEDINGS OF THE MULTIDIMENSIONAL DATA ANALYSIS WORKSHOP (J.deLeeuw, W.Heiser, J.Meulman and F.Critchley, eds.). Leiden, DSWO-Press.

Gabriel, K.R. and Zamir, S. (1979). TECHNOMETRICS, 21, 489-498.

Kester, N.K. (1979). Diagnosing and fitting concurrent and related models for two-way and higher-way layouts. Ph.D. THESIS, University of Rochester, NY.

Newman, W.M. and Sproull, R.F. (1980). PRINCIPLES OF INTERACTIVE COMPUTER GRAPHICS (Second Edition). New York, McGraw-Hill.

Odoroff, C.L., Basu, A., Gabriel, K.R. and Therneau, T.M. (1986). National Computer Graphics Association Annual Convention, Anaheim, California, May, 1986.

Tsianco, M.C., Odoroff, C.L., Plumb-Scanlon, S., and Gabriel, K.R. (1981). BGRAPH--A Program for Biplot Multivariate Graphics. Technical Report 81/20. Department of Statistics, The University of Rochester.

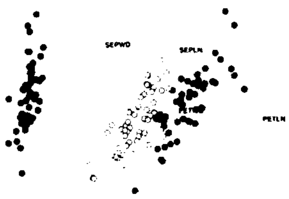


Figure 1. Iris Data -- initial biplot

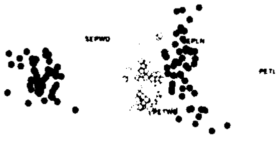


Figure 2. Iris Data - after rotation mostly about Y-axis

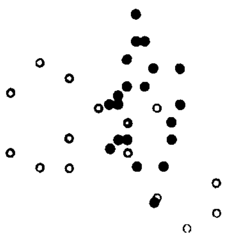


Figure 3. First 20 (dark) x 15 (light) Matrix -- no perspective cues

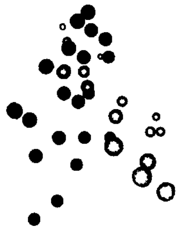


Figure 4. Second 20 (dark) x 15 (light) Matrix -- with perspective cues

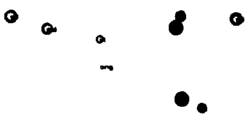


Figure 5. Solar Heater Data

Figure 6. Solar Heater Data with means

COMPUTATIONALLY INTENSIVE METHODS AND SUPERCOMPUTERS

Organizer: *Wesley L. Nicholson, Pacific Northwest Laboratory*

Algorithms for Bayesian Variable Selection in Regression
Toby J. Mitchell, John J. Beauchamp, Oak Ridge National Laboratory

Simulation of Radar and Surface Measurements of Rainfall
V. Chandrasekar, V.N. Bringi, Colorado State University

ALGORITHMS FOR BAYESIAN VARIABLE SELECTION IN REGRESSION

Toby J. Mitchell and John J. Beauchamp, Oak Ridge National Laboratory

ABSTRACT

We have developed a Bayesian approach to the problem of deciding which subset of a proposed set of p predictor variables to include in a linear regression model that is to be used for prediction. The direct implementation of this method requires the computation of the usual regression statistics for each of the 2^p possible submodels. We have developed a branch and bound method which yields the same results much more quickly by eliminating from consideration those submodels which are destined to have negligible posterior probability. Implementation of the algorithm on the Cray X-MP supercomputer is discussed.

1. Introduction

Our setting is that of standard linear regression. There are n observations on p predictors $x_1, x_2, \cdots, x_p$ and one dependent variable y. We shall assume the first order model:

$$y = \beta_0 + \sum_{j=1}^{p} \beta_j x_j + \epsilon \quad ,$$

where ϵ is normally distributed with mean 0 and variance σ^2, and all cases (runs) are independent.

At some point during the statistical analysis, one may be interested in the possibility of omitting some predictors from the model. The search for a "best" submodel (or set of submodels) is called *variable selection* or *subset selection*. It is undertaken for a number of possible reasons: (1) to express the relationship between y and the predictors as simply as possible, (2) to reduce future cost of prediction, (3) to identify "important" and "negligible" predictors, or (4) to increase the precision of statistical estimates and predictions.

We have developed a Bayesian approach to the problem of variable selection, the details of which will be submitted for publication elsewhere. In the next section, we summarize this approach.

2. The Bayesian Model

We assume that the predictors have been "suitably scaled," and shall avoid discussing this aspect any further here. In addition, we shall let the β_j's, $j=1,2, \cdots ,p$, be independently and identically distributed a priori, where the form of the common marginal distributions is similar to that of Box and Meyer [1986], i.e.,

$$\begin{aligned}
&P(\beta_j = 0) = h_2 \\
&P(\beta_j < b, \beta_j \neq 0) = h_1 \cdot (b + f), \quad -f < b < f \\
&P(|\beta_j| > f) = 0 \quad ,
\end{aligned}$$

where $h_1 > 0$, $h_2 > 0$, and $2h_1 f + h_2 = 1$. This is a "spike and slab" distribution, i.e., a mixture of a uniform distribution over the interval $(-f, f)$ and a distribution with all its probability mass at 0. We shall take f and γ as the parameters of this distribution, where

$$\gamma = h_2/h_1 \ ,$$

i.e., γ is the height of the spike divided by the height of the slab. We consider γ to be a measure of one's prior inclination to omit any predictor from the model. We shall treat it as an adjustable parameter of the Bayes model, that is, we shall not assign a distribution to it. However, we shall assign to σ the standard "noninformative" prior, i.e., $\ln \sigma$ is locally uniform, independent of the β's. We shall also take f to be arbitrarily large.

We are interested here in the posterior probabilities of the submodels, where each submodel is identified by the requirement that each member of a particular subset of the β's is 0.

It can be shown that the posterior probability of the m^{th} submodel is

$$P_m = \frac{w_m}{\sum_u w_u} \ .$$

where the logarithm of the "weight" w_m is given by

$$\ln(w_m) = k_m(-\ln(\gamma) + ½\ln(\pi)) + \ln\Gamma((n-k_m)/2)$$
$$- ½\ln|V_m| - ½(n-k_m)\ln S_m^2$$

where k_m is the number of terms in submodel m, V_m is σ^{-2} times the variance-covariance matrix of the least squares estimates of the β's omitted by submodel m, and S_m^2 is the residual sum of squares for submodel m.

From this posterior distribution, one can compute and plot various quantities of interest as functions of γ, e.g., the posterior probability that each β_j is 0. A useful way of assessing γ is to plot the posterior probability of goodness of fit as a function of γ; this is the sum of the posterior probabilities at all submodels that pass a standard F-test for goodness of fit at a specified level of significance.

3. Computations

In principle, this kind of analysis requires the computation of S_m^2 and $|V_m|$ for all 2^p submodels. Although there are efficient methods for doing all possible regressions [Furnival and Wilson, 1974], we really need only those submodels that have non-negligible posterior probability.

We define a "negligible" posterior probability as follows. Let m^* refer to the best submodel, i.e., the one with maximum posterior probability. If $P_m / P_{m^*} < 10^{-4}$, say, then P_m is negligible. Even though one can conceive of situations in which assigning posterior probability of 0 to all submodels classified as negligible by this definition will not result in a good approximation to the true posterior distribution, this is what we shall do. Our rationale is that such submodels would not have practical interest, since a much more acceptable alternative submodel exists.

We have developed a branch and bound algorithm that finds the weights for all non-negligible submodels. First, a forward selection routine is used to find a reasonably good submodel of $\hat{m}$ that we can use as a standard. (We would use m^* if we could, but we don't know it until the algorithm is finished.) We then define a cutoff value c for $\ln(w_m)$, where

$$c = \ln(w_{\hat{m}}) - \ln(10^4) \ .$$

The branch and bound algorithm finds all m and w_m such that $\ln(w_m)$ exceeds c. This catches all the non-negligible submodels (by the definition above) plus a few others, which can then be weeded out.

The algorithm is based on a tree of nodes, where each node is a collection of submodels. The root node consists of all submodels. All other nodes are characterized by a set of predictors that are in all submodels in that node and another set of predictors that are excluded from all submodels in that node. By computing the properties S_m^2 and $|V_m|$ of the largest and smallest submodels in a node, one can find upper and lower bounds on the posterior probabilities of all submodels in the node, as functions of k_m. If a node is found to be "bad," in the sense that none of its members have $\ln(w_m)$ that can exceed c, the node is not considered further. Otherwise, the node is "split" into two daughter nodes, where assignment of a submodel to one or the other of the two daughters is made on the basis of the presence or absence of a specified predictor. A heuristic choice of predictor is made for this purpose, the idea being to choose an apparently important one.

We have implemented this algorithm in a Fortran program that runs on our Digital Equipment Corporation VAX 11/780 with Berkeley UNIX 4.2 and on the Cray X-MP at the National Magnetic Fusion Energy Computing Center at Lawrence Livermore Laboratory. Our motivation for running it on the Cray is that we would like to consider models with many (> 50, say) predictors. At present, the largest cases we have tried have 25 predictors; this takes roughly one second of CPU time on the Cray. Future timing studies will need to take account of the fact that the computing time depends on the value of γ, since for some values of γ the number of non-negligible submodels is considerable.

We have also begun to modify our Fortran program to explicitly utilize the multitasking capability of the Cray X-MP, which has two processors.

REFERENCES

Box, G. E. P. and Meyer, R. D. (1986), "An Analysis for Unreplicated Fractional Factorials," Technometrics, 28, 11-18.

Furnival, G. M. and Wilson, R. W., Jr. (1974), "Regression by Leaps and Bounds," Technometrics, 16, 299-511.

SIMULATION OF RADAR AND SURFACE MEASUREMENTS OF RAINFALL

V. Chandrasekar and V.N. Bringi

Introduction

The radar remote measurement of rain intensity is a problem of continuing interest to radar meteorologists. For example, the remote monitoring of flash-flood producing storms is an important applications area. Even though considerable progress has been made in the development of new radar measurement techniques, e.g., using dual-polarized, dual-frequency or differential phase shift, the problems associated with the error structure of remotely sensed precipitation estimates appear to have received little attention so far in precipitation research, AGU (1984).

A pulsed, meteorological radar illuminates a radar resolution volume which depends on range, antenna pattern and pulse width; typically the volume is approximately 0.1 km^3 for a range of 50 km, 1° beam width and 1 μsec pulse, Doviak and Zrnic (1984). Within this volume, hydrometeors are assumed to be randomly positioned, and constitute a random medium from which radar measurements are obtained. Furthermore, the fractional volume concentration of the scatterers is generally very small (<< 1%) so that the independent backscatter approximation is valid. Hence, the average *backscattered power (or reflectivity) is* proportional to the (incoherent) sum of powers backscattered by each particle within the resolution volume. Statistical fluctuations of the received power are related to the Doppler velocity spectrum of the particles within the resolution volume. Conventional Doppler radars measure the mean power (or reflectivity, Z) as well as the first moment of the Doppler spectrum, i.e., the mean velocity. Rainfall rate, or the vertical flux of raindrops contained within the resolution volume, is conventionally related to Z by power law equations of the form $Z = aR^b$, where a and b depend on the unknown raindrop size distribution (RSD), Ulbrich (1983). The radar measured mean power ($\overline{P}$), an electromagnetic signal, is related to Z (a quantity of meteorological significance) by $\overline{P} = CZ/r^2$ where C is the radar constant and r is the range to the resolution volume. The parameters a and b of the Z-R relation are generally estimated by comparing radar measurements of reflectivity (Z) with surface instruments such as raingages (which estimate rain intensity) or raindrop size measuring devices (or disdrometers) which estimate the RSD. These surface devices have extremely small sampling volumes (0.1 to 1 m^3) compared to the radar sampling volume. Thus, even under ideal conditions, it is difficult to separate statistical fluctuations from fluctuations caused by physical processes (e.g., the changing RSD) with respect to the relationship between radar Z and surface measured R.

In order to overcome the problems associated with acquiring simultaneous radar/surface rainfall data, radar meteorologists often measure the RSD (either at the surface, or in-situ, using instrumented aircraft) or approximate the RSD by some functional form, and calculate both reflectivity (Z_{RSD}) and rainfall rate from the size spectrum. We use the subscript RSD on Z to denote that it is calculated based on a measured RSD or on an assumed form for the RSD. Hence, it is possible to relate Z_{RSD} and rain intensity, and to translate these relationships to radar measured Z versus rain intensity. However, statistical correlations between Z_{RSD} and R must be understood before such deductions can be made. If the RSD is approximated by a gamma distribution, the three parameters of the gamma distribution namely, particle number density, shape parameter and scale parameter, are generally unknown but are often estimated using measured RSDs via moment methods or MLEs, Ulbrich (1983), Mielke (1976), Wong and Chidambaram (1985). Again, statistical correlations between the various moments of the RSD must be understood before deductions regarding the physical fluctuation of the gamma parameters can be made.

Our paper is organized as follows: Section 1 describes the RSD model and intercomparison between radar reflectivity and rainfall intensity. Section 2 considers the theoretical correlation between Z_{RSD} and rain intensity (as well as other RSD moments) for a gamma RSD. Section 3 considers the statistical fluctuations inherent in the radar measurements of Z. Simulation methods for radar Z, and gamma RSDs (including moments) are given in Section 4 along with a description of computational complexity and the use of the CSU/Cyber 205 supercomputer. Section 5 describes the results of our simulations and shows examples of a few possibly incorrect deductions about physical processes that have been made which can be accounted for by statistical fluctuations only.

1. Raindrop Size Distribution (RSD).

The space-time evolution of the raindrop size distribution (RSD) is typically due to a variety of physical processes, e.g., evaporation, collision-coalescence, collisional breakup, drop sorting, etc. Both cloud models and measurements of RSDs at the surface show that a gamma RSD can account for many of the natural variations in the RSD, Ulbrich (1983):

$$N(D) = N_0\, D^m \exp(-\Lambda D) \qquad (1)$$

where N(D) is the number of raindrops per unit volume per unit size interval (D to D+ ΔD). In terms of the conventional gamma pdf, N(D) can be written in the equivalent form,

$$N(D) = \frac{N_T}{\Gamma(\alpha)\beta^{\alpha}}\, D^{\alpha-1} \exp(-D/\beta) \qquad (2)$$

where $\alpha > 0$, $\beta > 0$, $D \geq 0$. We note that $N_0 = \frac{N_T}{\Gamma(\alpha)\beta^{\alpha}}$, $m = \alpha - 1$; $\Lambda = \frac{1}{\beta}$. A physically meaningful parameter known as the median volume diameter D_0 can be defined by,

$$\int_0^{D_0} D^3\, N(D)\, dD = \int_{D_0}^{\infty} D^3 N(D)\, dD \qquad (3)$$

where D_0 is such that all drops with diameter $\leq D_0$ contribute to one half the total liquid water content. Ulbrich (1983) has shown that $\Lambda D_0 \sim 3.67+m$. Reflectivity (Z_{RSD}) and rain intensity can be formulated as various moments of N(D):

$$Z_{RSD} = \int_0^\infty D^6 \; N(D) \; dD \quad mm^6 m^{-3} \qquad (4)$$

$$R = \frac{\pi}{6} \int_0^\infty D^3 \; v(D)N(D) \; dD \quad mm \; hr^{-1} \qquad (5)$$

where $v(D) = CD^{3.67}$, is the raindrop still air fall speed.

RSDs can be estimated by using surface instruments, e.g., disdrometers (drop size meters), or using probes mounted on instrumented aircraft. The Joss-Waldvogel (1967) disdrometer is a momentum device with a sensor area of 50 cm^2, and estimates N(D) for D in the range 0.5 mm-5 mm with typical integration times of 30 sec - 1 min. A number of authors have used N(D) data from Joss-Waldvogel disdrometers, Ulbrich (1983), Joss and Gori (1978), Bringi et al. (1982), Goddard et al. (1982).

To define the radar measured reflectivity factor *we assume that the radar resolution volume is filled* homogeneously with raindrops. In the Rayleigh scattering limit the radar reflectivity factor Z is then defined as,

$$Z = \frac{1}{\Delta V} \sum_i D_i^{\,6} \quad mm^6 m^{-3} \qquad (6)$$

where the summation applies to raindrops within a volume ΔV. Since the range of Z can be quite large we define dBZ = 10 $\log_{10}(Z)$. There is a general correlation between Z and rain intensity expressed by a power law of the form $Z = aR^b$.

Zawadzki (1984) has intercompared radar Z measurements with surface disdrometer measurements of rain intensity. He has analyzed a number of factors which can cause discrepancies between radar-derived rain rates and surface-measured rain rates. Sampling errors affect both the surface measurements (due to inadequate sample volume) as well as the radar measurements of mean power (or dBZ) due to the finite Doppler velocity spectrum. Systematic errors can be caused by radar calibration problems, as well as by the non-coincidence (in space-time) of the raindrops measured by the radar as opposed to the raindrops which actually impact on the surface disdrometer. Fig. 1a taken from Zawadzki (1984) shows a scatter plot of rain intensity versus Z_{RSD} (estimated from a Joss-Waldvogel disdrometer using Eq. (6)) while Fig. 1b shows rain intensity versus radar-measured Z. Zawadzki notes that the variability in R for a given Z_{RSD} is significantly less (~ factor of 3) than the variability in R for the same radar-measured Z. By assuming that the discrepancies between Figs. 1a, b were due to physical causes, Zawadzki noted that, ". . . we must conclude that the variability of the drop-size distribution is a relatively minor factor affecting the precision of radar estimates of rain rate " We have simulated a similar experiment corresponding to the data of Figs. 1a, b and show in Section 5 that the discrepancies between Figs. 1a, b can be accounted for by statistical fluctuations only.

As another example, we consider the relationship between N_0 and m (see Eq. (1)) derived by Ulbrich (1983) using RSDs measured by a Joss-Waldvogel disdrometer. *Ulbrich estimates N_0, m, and D_0 based* on higher moments of the RSD. This procedure places more weight on the larger drop sizes as compared to MLEs which place more emphasis on drop sizes having a higher frequency of occurrence, Wong and Chidambaram (1985). Fig. 1c shows a scatter plot of $\log_{10}N_0$ versus m taken from Ulbrich (1983). From this data, Ulbrich concludes that physical processes result in a N_0-m relation, and that, effectively, the three-parameter gamma RSD reduces to a two-parameter form. We have simulated this experiment and show in Section 5 that the relationship between N_0 and m is due to the statistical correlations between estimators of the higher order moments of the RSDs. The above two examples imply that statistical fluctuations must be separated from variations induced by physical causes. Simulations offer a powerful method of studying the statistics of radar and surface measurements where the "natural" fluctuations can be introduced separately. However, such simulations involve large scale computations on a supercomputer since the physical parameters must be varied over a wide range.

2. Surface Disdrometer Measurements.

Gertzman and Atlas (1977) have shown that, in raindrop sampling devices such as disdrometers, the measurement variability is due both to statistical sampling errors and to real fine-scale physical *variations which are not readily separable from the* statistical ones. Sasyo (1965) and Cornford (1967) have shown that, for a constant mean rain intensity, the total number of raindrops observed will be distributed about its mean according to the Poisson distribution. This property has been used by Joss and Waldvogel (1969), Gertzman and Atlas (1977) and Wong and Chidambaram (1985) to obtain the fractional standard deviations of higher order moment estimators (which correspond to radar measurements) of RSDs. In this work we use a somewhat different approach so that the correlation structure of higher order RSD moment estimators can be computed.

We re-write Eq. (2) in the form of a gamma pdf,

$$f(D) = \frac{\Lambda^{m+1}}{\Gamma(m+1)} \; D^m \exp(-\Lambda D) \qquad (7)$$

In the following development we assume that the sampling volume, V, is constant and does not vary with raindrop size. If V does vary with D, then this dependency can be introduced by multiplying f(D) by the sampling volume function V(D).

If n raindrops are observed with a fixed sample volume V with diameters $D_1, D_2 \ldots D_n$, then this RSD is a composite distribution of total number of raindrops (or equivalently, the concentration of drops within any interval D to D+ΔD) and the drop diameter, where the diameters are distributed according to the gamma pdf, and the total number of raindrops (n) are distributed according to the Poisson distribution.

Conventional estimators of higher order RSD moments are expressed as follows:

$$Z = \frac{1}{V} \sum_{i=1}^{n} D_i^{\,6} \quad mm^6 m^{-3} \qquad (8a)$$

$$R = \frac{7.12 \times 10^{-3}}{V} \sum_{i=1}^{n} D_i^{\,3.67} \quad mm \; hr^{-1} \qquad (8b)$$

$$LWC = \frac{\pi}{6} \sum_{i=1}^{\underset{\sim}{n}} D_i^3 \quad gm\ m^{-3} \tag{8c}$$

where LWC stands for liquid water content. The above estimators can be written in general as,

$$\hat{p}_\alpha = \frac{C_\alpha}{V} \sum_{i=1}^{\underset{\sim}{n}} D_i^\alpha \quad \text{where } p_\alpha = C_\alpha \int D^\alpha N(D)\ dD. \tag{8d}$$

We can now find the mean of $\hat{p}_\alpha$ as,

$$E(\hat{p}_\alpha) = \frac{C_\alpha}{V} E(\sum_{i=1}^{\underset{\sim}{n}} D_i^\alpha)$$

where E( ) stands for the expected value. The above expectation is that of a random sum. Assuming the D_is are iid we have,

$$E(\sum_{i=1}^{\underset{\sim}{n}} D_i^\alpha) = E(\ E\{\sum_{i=1}^{\underset{\sim}{n}} D_i^\alpha | \underset{\sim}{n}\}\ ) \tag{9}$$

$$= E_A(\underset{\sim}{n})\ E_B(D^\alpha)$$

where E_A( ) is the expectation with respect to the total number of raindrops and E_B( ) is the expectation with respect to the gamma pdf. Hence,

$$E_A(\underset{\sim}{n}) = VN_T = V\ N_0 \frac{\Gamma(m+1)}{\Lambda^{m+1}} \tag{10a}$$

$$E_B(D^\alpha) = \frac{\Gamma(m+\alpha+1)}{\Gamma(m+1)\ \Lambda^\alpha} \tag{10b}$$

$$\text{and,} \qquad E(\hat{p}_\alpha) = C_\alpha N_T \frac{\Gamma(m+\alpha+1)}{\Gamma(m+1)\ \Lambda^\alpha} \tag{10c}$$

It is easily verified that p_α is an unbiased estimator of p_α. Similarly,

$$var(\hat{p}_\alpha) = \frac{C_\alpha^2}{V^2} var(\sum_{i=1}^{\underset{\sim}{n}} D_i^\alpha) \tag{11a}$$

$$= \frac{C_\alpha^2 N_0}{V} \frac{\Gamma(m+2\alpha+1)}{\Lambda^{m+2\alpha+1}} \tag{11b}$$

We note that the variance decreases with increase in sample volume, as expected. The fractional standard deviation (FSD) of $\hat{p}_\alpha$ is

$$FSD(\hat{p}_\alpha) = \left[\frac{\Lambda^{m+1}}{N_0 V}\right]^{1/2} \frac{\Gamma^{1/2}(m+2\alpha+1)}{\Gamma(m+\alpha+1)} \tag{12}$$

Eq. (12) is identical to Eq. (29) of Gertzman and Atlas (1977).

We now derive the correlation between two estimators $\hat{p}_\alpha$ and $\hat{p}_\beta$ defined as,

$$\hat{p}_\alpha = \frac{C_\alpha}{V} \sum_{i=1}^{\underset{\sim}{n}} D_i^\alpha$$

$$\hat{p}_\beta = \frac{C_\beta}{V} \sum_{i=1}^{\underset{\sim}{n}} D_i^\beta$$

The covariance between $\hat{p}_\alpha$ and $\hat{p}_\beta$ is,

$$cov(\hat{p}_\alpha, \hat{p}_\beta) = \frac{C_\alpha C_\beta}{V^2} cov(\sum_{i=1}^{\underset{\sim}{n}} D_i^\alpha, \sum_{i=1}^{\underset{\sim}{n}} D_i^\beta) \tag{13}$$

By conditioning on $\underset{\sim}{n}$ the covariance can be written as,

$$cov(\sum_{i=1}^{\underset{\sim}{n}} D_i^\alpha, \sum_{i=1}^{\underset{\sim}{n}} D_i^\beta) = E\{cov(\sum_{i=1}^{\underset{\sim}{n}} D_i^\alpha, \sum_{i=1}^{\underset{\sim}{n}} D_i^\beta) | \underset{\sim}{n}\}$$
$$+ cov\ \{E(\sum_{i=1}^{\underset{\sim}{n}} D_i^\alpha | \underset{\sim}{n}),\ E(\sum_{i=1}^{\underset{\sim}{n}} D_i^\beta | \underset{\sim}{n})\} \tag{14}$$

Since the D_is are iid, $cov[D_i^\alpha, D_j^\beta] = 0$ when $i \neq j$. Therefore, the first term on the right hand side of Eq. (14) simplifies to $E(\underset{\sim}{n})\ cov(D^\alpha, D^\beta)$, while the second term simplifies to $E(D^\alpha)E(D^\beta)var(\underset{\sim}{n})$. Thus,

$$cov(\hat{p}_\alpha, \hat{p}_\beta) = \frac{C_\alpha C_\beta}{V^2} E(\underset{\sim}{n})\ E(D^{\alpha+\beta}) \tag{15a}$$

$$= \frac{C_\alpha C_\beta N_0}{V} \frac{\Gamma(m+\alpha+\beta+1)}{\Lambda^{m+\alpha+\beta+1}} \tag{15b}$$

Combining with Eq. (11b) we get the correlation coefficient $\rho_{\alpha,\beta}$ as

$$\rho_{\alpha,\beta} = \frac{\Gamma(m+\alpha+\beta+1)}{\{\Gamma(m+2\alpha+1)\ \Gamma(m+2\beta+1)\}^{1/2}} \tag{16}$$

It is interesting to note that the correlation coefficient between $\hat{p}_\alpha$ and $\hat{p}_\beta$ is independent of N_0 and D_0. In Fig. 2 we show plots of $\rho_{\alpha,\beta}$ versus m, where $\hat{p}_6 = Z_{RSD}$ (radar reflectivity) while $\hat{p}_\beta$ represents estimators with $\beta = 0$ (concentration), $\beta = 1$ (mean raindrop size), $\beta = 2$ (optical extinction), $\beta = 3$ (liquid water content), and $\beta = 3.67$ (rain intensity, R). Note that estimators of Z_{RSD} and R are highly correlated and nearly independent of the shape (m) of the gamma pdf. Hence, experimental data which show scatter plots *of Z_{RSD} versus R obtained from disdrometers must* be carefully interpreted, i.e., experimentally derived correlations will contain the effects of both physical correlations as well as statistical correlations.

In order to simulate $\hat{p}_\alpha$ we need to derive its distribution function. Since $\hat{p}_\alpha$ is a moment estimator with a random sum, analytic derivations are hopelessly complicated especially for values of $\alpha>0$. Hence, we resort to simulating $\hat{p}_\alpha$ (for $\alpha = 6$ and 3.67) by first simulating the RSD. Typical concentrations of raindrops vary from 10^2 to 10^5 per m^3, Gordon and Marwitz (1984). Considering a 100 litre (0.1 m^3) sample volume, the number of raindrops can vary between 10 to 10,000 and the number increases with sample volume. To simulate the variability in concentration we have to use Poisson deviates. Then for each of these numbers ($\underset{\sim}{n}$) we need to simulate $\underset{\sim}{n}$ gamma deviates to represent the RSD. Next the physical parameters (N_0, m and D_0) of the gamma RSD must be varied over the range of observed values. For example, consider $N_0 = 8000\ mm^{-1}m^{-3}$, $D_0 = 2$ mm and m = 0 (exponential RSD) and $V = 0.1\ m^3$. To obtain a scatter plot of Z_{RSD} versus R under these conditions it takes 125 seconds of execution time on the CSU/Cyber 180/830 computer. It is apparent that large computing power is needed to handle such intensive computational needs.

3. Radar Measurements.

Radar measurements of reflectivity Z involve estimation of Z from the measurement of mean (time averaged) backscattered power ($\overline{P}$) from a given radar resolution volume, Doviak and Zrnic (1984).

The sample backscattered power is the incoherent sum of powers backscattered by each raindrop within the resolution volume. The fluctuation of the backscattered power is due to relative motion between the various drops, and can be related to the width of the Doppler velocity spectrum. To obtain the mean power, $\overline{P}$, power samples must be averaged over a large number of pulses. The total number of pulses depends on the radar pulse repetition time (PRT), and the dwell time of the antenna beam on the resolution volume. Doviak and Zrnic (1984) show that the power samples have exponential marginal distributions. Zrnic (1975) has developed a procedure for simulating the time series of power samples assuming a Gaussian Doppler velocity spectrum. We use his method in our work and refer to Zrnic's paper for details. The principal assumptions we make are as follows (for a typical meteorological radar):

Doppler Spectrum	Gaussian, 2 = 6ms'
Radar Reflectivity	$N_0\ \Gamma(m+7)\{D_0/(3.67+m)\}$
Pulse Repetition Time	1 millisec
Number of Samples	128
Radar Wavelength	10 cm
Receiver	Power Law (no noise)

Our radar simulations, for a given mean power (or, reflectivity) involve length N(128) exponential deviates, and length N complex FFTs. As the reflectivity is varied (by changing N_0, D_0, or m) we see that the radar simulations are also computationally intensive.

4. Simulations.

As discussed in Sections 2 and 3, the surface disdrometer and radar simulations are computationally intensive requiring simulations of gamma, Poisson and exponential deviates, as well as complex FFTs. These simulations are repeated many times as the physical parameters, namely, N_0, D_0 and m of the RSD are varied over a considerable range of values commonly found in rainfall. Thus, these simulations are ideal for implementation on a vector computer like the CSU/Cyber 205.

4.1 Exponential Random Deviates.

The inverse CDF technique for generating exponentials is used here. We generate them from uniform (0,1) deviates and take the negative logarithm. This method is easily vectorizable since the CDF is in closed form and conditional checks for specific values can be avoided. Exponentials with means differing from unity are needed here. Some timing runs made on the CSU/Cyber 205 indicate that a length 100 string of deviates can be computed 2.2 times faster than the scalar method, whereas the speed-up factor increases to 4.8 and 6.4 for strings of length 500 and 2500, respectively. This speed-up factor is important in our work since a large number of these simulations are needed.

4.2 Gamma Random Deviates.

Kennedy and Gentle (1980) discuss a number of methods of simulating gamma random deviates. Among the various algorithms, the one proposed by Cheng (1977) appears to be most suitable for vector implementation. This method is an adaptation of the envelope rejection technique and is described in Appendix A.

4.3 Timing.

Simulation of 50 samples of radar and surface measurements with $N_0 = 8000\ mm^{-1}m^{-3}$, $D_0 = 2$ mm, m = 0 and V = 0.1 m^3 took 125 seconds on the CSU/Cyber 180/830 whereas the vector code took 2.9 seconds of execution time on the Cyber 205. A complete simulation with N_0, D_0 and m varying took 160 seconds on the Cyber 205 for a 1 m^3 sample volume.

5. Discussion and Results.

We now apply our simulation results to two types of problems, namely

a) radar-surface disdrometer intercomparisons, and

b) inferences on gamma RSD parameters using surface disdrometer measurements.

5.1 Radar/Disdrometer Intercomparisons.

In Section 1 we discussed Zawadzki's (1984) interpretation of radar/disdrometer intercomparisons as shown using his Figs. 1a, b. In Figs. 3-6 we show our simulation results where Figs. 3a-6a show scatter plots of surface rain intensity versus Z_{RSD}, while Figs. 3b-6b show scatter plots of rain intensity versus radar Z. The physical parameters N_0, D_0 and m are varied differently in each Fig. 3-6. For example, in Fig. 3a, b we represent one rainfall condition with $N_0 = 8000\ mm^{-1}m^{-3}$, $D_0 = 2$ mm and m = 0. Fig. 3a shows very good correlation between R and Z_{RSD} whereas in Fig. 3b, R appears uncorrelated with radar Z. In Fig. 3b note that the distribution of R is asymmetric. We now vary the physical parameter m from 0.5 to 5 in Figs. 4a, b with the same N_0, D_0 values as before. The correlation in Fig. 4a is significantly higher than in Fig. 4b. In Figs. 5a, b we keep $N_0 = 8000\ mm^{-1}m^{-3}$, m = 0.5 and vary D_0 from 0.8 to 2.8 mm. In Fig. 6a, b we vary N_0, D_0 and m simultaneously over a broad range of values that can physically occur. Again, the same feature is deduced, i.e., R versus Z_{RSD} is more tightly correlated than R versus radar Z over a wide variety of physical rainfall conditions. We also note that the magnitude of the discrepancy is of the same order shown by Zawadzki's (1984) Fig. 1a, b. Our results are obtained for an ideal, noise-free radar with the radar resolution volume completely filled by a homogeneous rain medium identical to that sampled by the disdrometer. This implies that even without considering the various physical factors enumerated by Zawadzki (1984), we can observe a similar variability in our simulations as was observed in the experimental data shown in Figs. 1a, b. Our variations were obtained solely by the statistical nature of the measured quantities without any physical changes or instrumentation problems. Thus, it is not possible to conclude that the discrepancy between Figs. 1a, b can be accounted for by physical causes as enumerated by Zawadzki (1984). Therefore, it is also not possible to agree with Zawadzki's conclusion that the variability of the RSD is a relatively minor factor affecting the precision of radar estimates of rain intensity.

6.2 Disdrometer Inferences.

In Section 1 we discussed Ulbrich's (1983) conclusion that the three-parameter gamma RSD in fact reduces to a two-parameter gamma RSD with N_0 and m being related by $N_0 = 6 \times 10^4 \exp(3.2\ m)$, see Fig. 1c. Ulbrich (1983) estimates the parameters N_0, D_0 and m from disdrometer measured RSDs using three moment estimators, viz.,

$$G' = \frac{\int_0^\infty D^6 N(D)\, dD}{D_m^3 \int_0^\infty D^3 N(D)\, dD} = \frac{(6+m)(5+m)}{(4+m)^2} \qquad (17a)$$

$$D_m = \frac{\int_0^\infty D^4 N(D)\, dD}{\int_0^\infty D^3 N(D)\, dD} \approx \left(\frac{4+m}{3.67+m}\right) D_0 \qquad (17b)$$

$$LWC = \frac{\pi}{6} \int_0^\infty D^3 N(D)\, dD \approx 0.52\Gamma(4+m) N_0 \left[\frac{D_0}{3.67+m}\right]^{m+4} \qquad (17c)$$

Using experimentally measured RSDs, G', D_m and LWC are calculated using discrete versions of the integral in Eqs. 17a, b, c, from which the gamma RSD parameters N_0, D_0 and m are inferred. The result of this "inversion" procedure is given as a scatter plot of $\log_{10}N_0$ versus m in Fig. 1c which is taken from Ulbrich (1983). If a linear relationship between $\log_{10}N_0$ and m exists for natural gamma-parameterized RSDs, then in essence the gamma RSD is reduced to a two-parameter form.

To study the statistical fluctuations in the moment estimators defined by Eqs. 17a, b, c we use the Ulbrich inversion procedure with simulated gamma RSDs. In Fig. 7 we assume gamma RSDs with N_0 = 8000 $mm^{-1}m^{-3}$, m = 0 and V = 1 m^3, while D_0 = 1 mm. Observe in this figure that estimate of $\log_{10}N_0$ varies linearly with estimate of m, and the slope is within the range theoretically derived by Ulbrich (1983). In Fig. 8 N_0, D_0 and m are varied over a wide range to encompass the full range of naturally occurring RSDs. Our scatter points lie within the two straight lines in Fig. 8 derived theoretically by Ulbrich (1983) using a large number of empirical $Z = aR^b$ relations. Comparison of Fig. 8 with Fig. 1c also shows that our simulated (N_0,m) pairs lie within the experimental scatter derived by Ulbrich (1983). This raises the obvious question of whether the N_0-m relationship derived by Ulbrich is due to physical causes or whether it is due to statistical correlations between the various moment estimators defined in Eq. 17. Note that our simulations do not preclude the existence of a physical N_0-m relation; the implication of Fig. 8 is that other methods may need to be used to determine if a physical N_0-m relation indeed exists. Finally, in Fig. 9 and 10 we show scatter plots of m versus D_0, and N_0 versus D_0 using the simulations. Since these scatter plots indicate the correlation between the estimates is quite low it implies that if significant correlations between these parameters are observed, then it is a real physical observation.

Conclusions.

We have considered a class of statistical simulations which are computationally intensive and amenable to implementation on a vector computer. We have simulated two totally different types of measurements, viz., radar, and surface disdrometer, measurements of rainfall. These simulations involve exponential, Poisson and gamma random deviates. The problem is a large scale one since the parameters describing the rainfall must be varied over a wide range. Thus, we have complete control over the physical and statistical variables.

We have applied our simulations to explain why the correlation is less in plots of radar measured reflectivity verses surface measured rain intensity as compared to plots when both quantities are obtained from surface instruments. Previous interpretations have ascribed this feature to physical causes. While physical factors are important when comparing radar measurements of rainfall to surface measurements of rain intensity, it is important to have a good measure of statistical variabilities before ascribing the features to physical causes alone, Zawadzki (1984).

We have also applied our simulations to show that N_0-m relations, as derived by Ulbrich (1983) using experimental disdrometer raindrop size distributions and certain higher order moment estimators, cannot be ascribed to physical causes. The simulation results indicate that the moment estimators are correlated resulting in a high degree of correlation between N_0 and m, even when the gamma distribution parameters were widely varied. Though our simulations do not preclude the existence of a physical N_0-m relation (which is important for radar measurements of rain intensity), it suggests that other methods may be needed to confirm this.

Acknowledgements

The authors are grateful to Dr. Hari Iyer of Colorado State University for many helpful discussions and for a critical reading of this paper. This work was supported by the U.S. Army Research Office via NCAR Subcontract #S3024. One of the authors (VC) is a Fellow of CSU's Institute of Computational Studies which provided Cyber 205 computer resources for this study.

APPENDIX A

The standard gamma has the following distribution function:

$$f(x) = \frac{1}{\Gamma(x)} x^{\alpha-1} e^{-x} ; \qquad (A1)$$

Using Cheng's notation, let

$$M = \max_x \{ \frac{f(x)}{g(x)} \} \qquad (A2)$$

be finite. Take a pair of independent u(0,1) variables U_1 and U_2 say. Let $x = G^{-1}(U_1)$. Then if $[f(x)]/[Mg(x)] \geq U_2$ accept x, otherwise reject it. Each accepted x has density f(x). Cheng suggests using f(x) same as (A1) and

$$g(x) = \frac{\lambda\mu x^{\lambda-1}}{(\mu + x^\lambda)^2} \qquad (A3)$$

where $\mu = \alpha^\lambda$ and $\lambda = \sqrt{2\alpha-1}$.

M, the expected number of trials varies between 1.47 and 1.13 as α varies from 1 to ∞. The advantage of this method is it gives a reasonably small rejection ratio which translates into starting vector length of uniform deviates not much greater than the length of string of gamma required. This method also has one decision taking spot which can be easily accommodated by using control bit vectors that would have bit value "1" for accepted elements of the vector that can be gathered later. The following steps show equivalent vector form for Cheng's algorithm. (We denote vectors with an arrow above the symbols.)

Step 1: *Generate a pair of uniform random vectors* $\vec{U}_1$ and $\vec{U}_2$

Step 2: Set $\vec{V} = a \log\{\vec{U}_1/(1-\vec{U}_1)\}$,
$\vec{X} = \alpha e^{\vec{V}}$

Step 3: If $b + c\vec{V} - \vec{X} \geq \log [\vec{U}_1 \cdot \vec{U}_1 \cdot \vec{V}_2]$
set bit vector elements to 1.

The constants are:
$a = (2\alpha-1)^{-1/2}$, $b = \alpha - \log 4$
and $c = \alpha + a^{-1}$

Note that in the above equations, arithmetic operations like multiplication, division and exponentiation imply element by element operation. The length of these vectors n', is Mn where n is the vector length of random deviates required. If the number of "1" bits fall short of n' then those few random deviates can be computed by scalar version of algorithm. Timing runs made on Cyber 205 indicate that length 100 deviates can be computed 2.95 times faster, whereas the speed-up increases to 6.1 for length 50 and 7.1 for length 2500.

REFERENCES

AGU (1984), A new interdisciplinary focus on precipitation research, committee on precipitation, EOS, 65, 377.

Bringi, V.N., T.A. Seliga and W.A. Cooper, 1984: Analysis of aircraft hydrometeor spectra and differential reflectivity (Z_{DR}) radar measurements during the Cooperative Convective Precipitation Experiment. Radio Science, 19, 157-167.

Bringi, V.N., T.A. Seliga, and E.A. Mueller, First comparisons of rainfall rates derived from radar differential reflectivity and disdrometer measurements. IEEE Trans. Geosci. Remote Sensing, Vol. GE-20, No. 2, pp. 201-204, 1982.

Cheng, R.C.H. (1978), Generating of gamma variables with non-integral shape parameter, Appl. Stat. 26, 71-75.

Cornford, S.G., Sampling errors in measurements of raindrop and cloud droplet concentrations, Meteorol. Mag., 96, 271-282, 1967.

Doviak, R.J. and D.S. Zrnic', 1984: Doppler Radar and Weather Observations. Academic Press, Inc., Orlando, Florida.

Gertzman, H.S. and Atlas, D., 1977: Sampling errors in the measurement of rain and hail parameters. J. Geophys. Res., Vol. 82, No. 31, 4955-4966.

Goddard, J.W.F., and S.M. Cherry, 1984: The ability of dual polarization radar (co-polar linear) to predict rainfall rate and microwave attenuation. Radio Science, 19, 201-208.

Goddard, J.W.F., Cherry, S.M. and V.N. Bringi, Comparison of dual-polarization measurements of ran with ground-based disdrometer measurements. J. Appl. Meteor., 21, 252-256, 1982.

Gordan, G.L. and Marwitz, J.D., 1984: An airborne comparison of three PMS probes. J. Atmos. Oceanic. Tech., 1, 22-27.

Jameson, A.R., 1985: Microphysical interpretation of multiparameter radar measurement in rain. Part III. Interpretation and measurement of propagation differential phase shift between orthogonal linear polarizations. J. Atmos. Sci., 42, 607-614.

Joss, J., and E.G. Gori, 1978: Shapes of raindrop size distributions. J. Appl. Meteor., 17, 1054-1061.

Joss, J., and Waldvogel, Raindrop size distribution and sampling size errors. J. Atmos. Sci., 26, 566-569, 1969.

Joss, V.J., and A. Waldvogel, 1967: Ain spektograph fur niederschlagstropher mit automatischer Auswortung. Pure Appl. Geophys., 68, 240-246.

Kennedy, Jr., W.J. and Gentle, J.E., 1980: Statistical computing, Vol. 33, Statistics: Textbooks and monographs, Marcel Dekker, Inc. NY and Basel.

Mielke, P.W., Jr., 1976: Simple iterative procedures for two parameter gamma distribution maximum likelihood estimates. J. Appl. Meteor., 15, 181-183.

Richards, W. and C. Crozier, 1983. Precipitation measurements with a C-band radar in Southern Ontario. Atmos-Ocean., 21, 125-137.

Sasyo, Y., On the probabalistic analysis of precipitation particles, In Proceedings of the International Conference on Cloud Physics, pp. 254-259. International Association of Meteorological and Atmospheric Physics, 1965.

Seliga, T.A. and V.N. Bringi, 1976: Potential use of radar differential reflectivity measurements at orthogonal polarizations for measuring precipitation. J. Appl. Meteor., 15, 69-76.

Ulbrich, C.W., 1983: Natural variations in the analytical form of raindrop size distribution. J. Climate Appl. Meteor., 22, 1764-1775.

Wong, R.K.W. and Chidambaram, N., 1985: Gamma size distribution and stochastic sampling errors. J. Chim. and Appl. Meteor., 24, 568-579.

Zawadzki, I., Factors affecting the precision of radar measurements of rain. Preprints, 22nd Conference on Radar Meteorology, AMS, Sept 10-13, 1984, Zurich, 1977: Switzerland.

Zrnic', D.S., Simulation of weatherlike Doppler spectra and signals. J. Appl. Meteor., 14, 619-620, 1975.

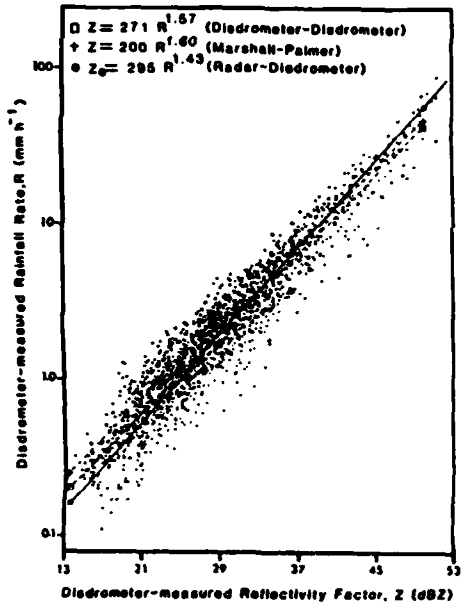


Fig. 1a. Scattergram of disdrometer derived values of Z and R for data in the Toronto, Ont. region taken during 1977/78/79 summer seasons (from Richards and Crozier, 1983). Each point represents a 7 min. sample.

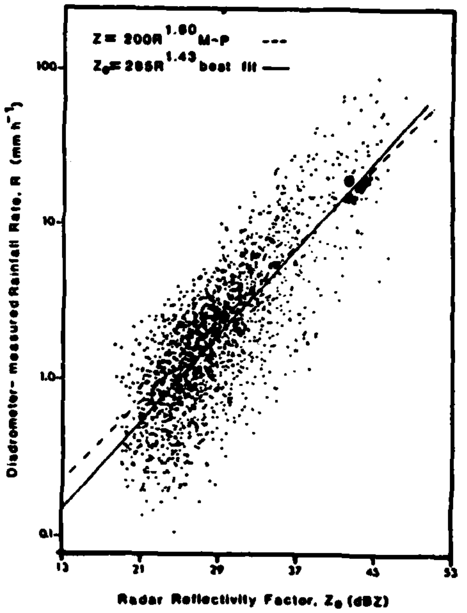


Fig. 1b. Scattergram of radar measured Z and disdrometer derived R for the same events as in Fig. 1a (from Richards and Crozier).

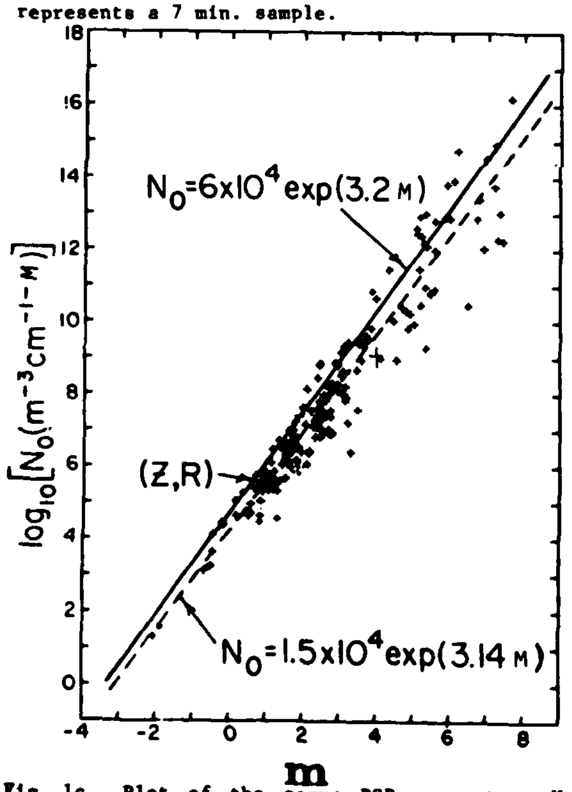


Fig. 1c. Plot of the gamma DSD parameters N_0 versus m as deduced from experimental dropsize spectra from moment to moment within a given rainfall type. The dashed line is the least squares fit to all the data. The solid line is the least squares N_0-m line from empirical Z-R relationships. Also shown as (Z,R) is the point deduced from empirical Z-R relation which applies to these data (from Ulbrich, 1983).

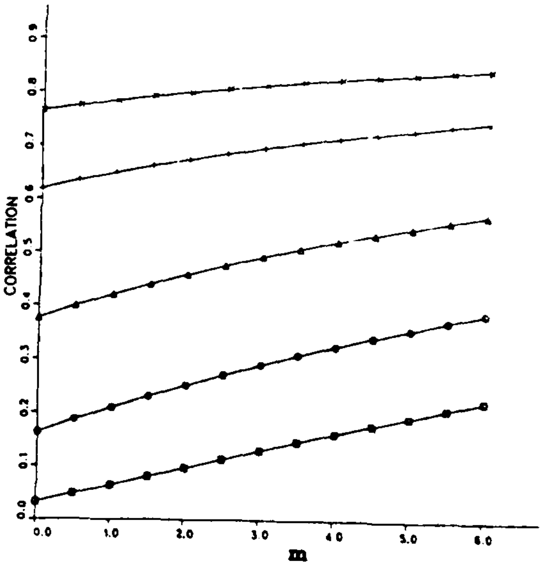


Fig. 2. Correlation of reflectivity, dBZ, with other meteorological quantities of interest plotted as a function of m. The various curves represent correlation of dBZ with (□) concentration, (.) mean particle size, (△) mean particle size squared, (+) liquid water content and (X) rainfall rate.

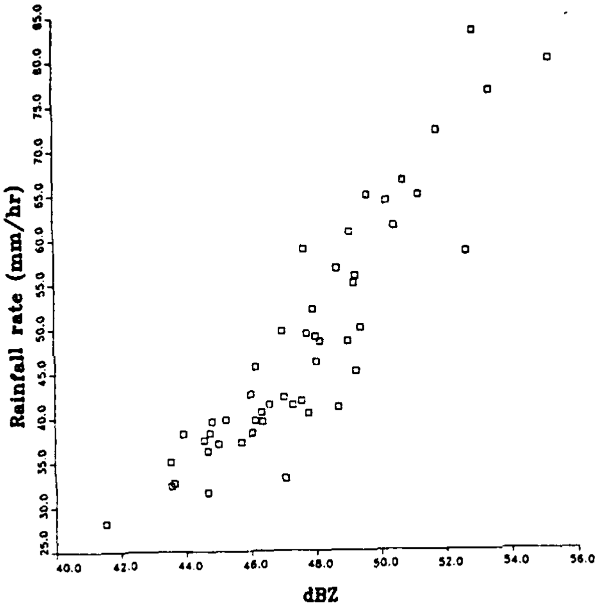


Fig. 3a. Scatterplot of rainfall rate versus dBZ where both are derived from the drop spectra observed by disdrometer. This figure shows 50 realizations under the condition N_0 = 8000 (m^{-3} mm^{-1}), D_0 = 2.0 mm and m = 0.

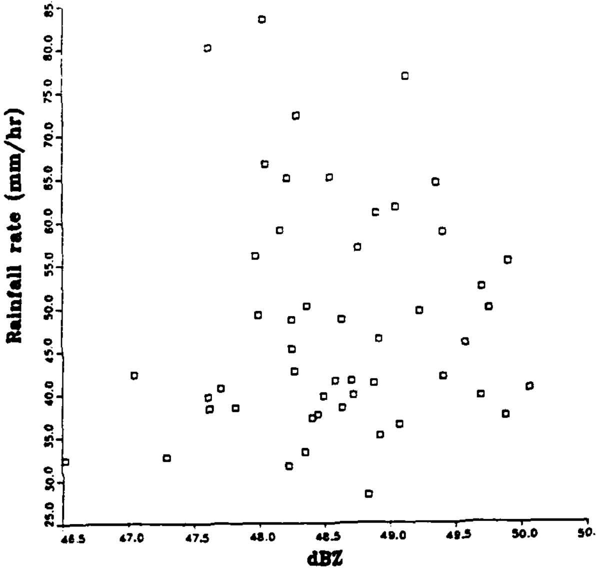


Fig. 3b. Same as Fig. 3a except dBZ is "measured" by radar.

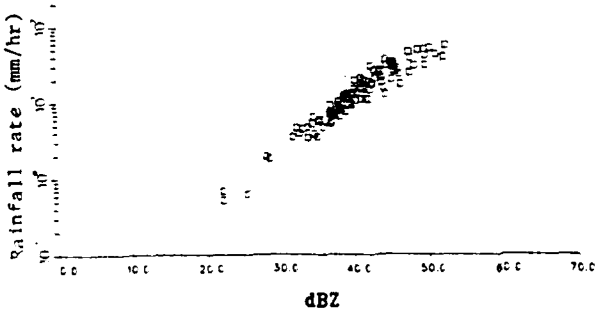


Fig. 4a. Scatterplot of rainfall rate versus dBZ, both observed on ground. This figure corresponds to N_0 = 8000, D_0 = 2.0 mm and m varying between 0.5 to 5.

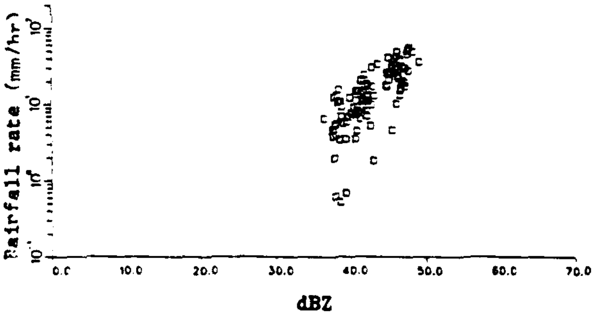


Fig. 4b. Same as Fig. 4a except dBZ is "measured" by radar.

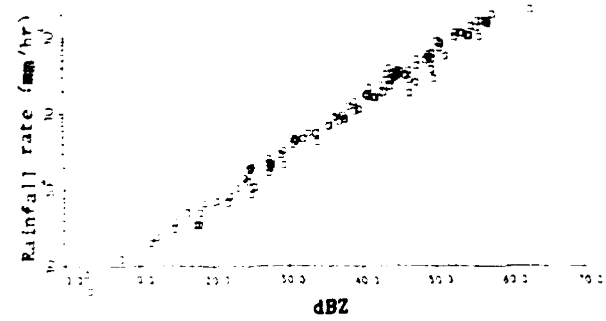


Fig. 5a. Scatterplot of rainfall rate versus dBZ with N_0 = 8000, m = 0.5 and D_0 varying between 0.8 to 2.8 mm.

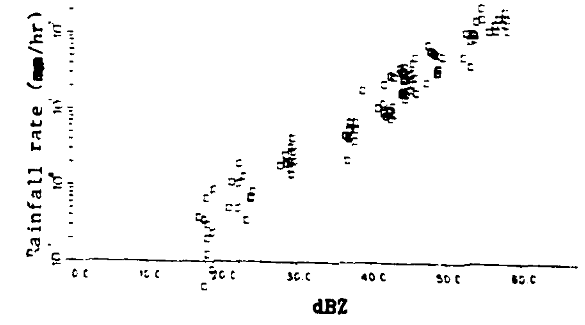


Fig. 5b. Same as Fig. 5a except dBZ is "measured" by radar.

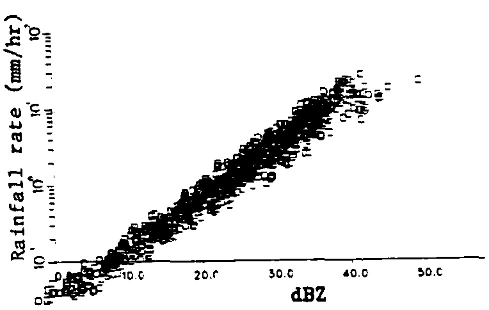


Fig. 6a. Comprehensive scatter plot of rainfall rate versus dBZ, both of which are derived from ground observations. Variation of N_0, m and D_0 are done to cover a wide range of suggested Z-R relationships, Ulbrich (1983).

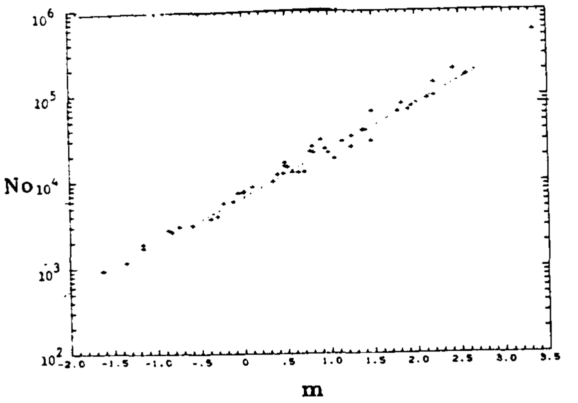


Fig. 7. Scatterplot of N_0 versus m estimates for average values of N_0 = 8000, m = 0 and D_0 = 2 mm. The estimates are tightly correlated with log N_0 and m being linearly related with slope = 0.45.

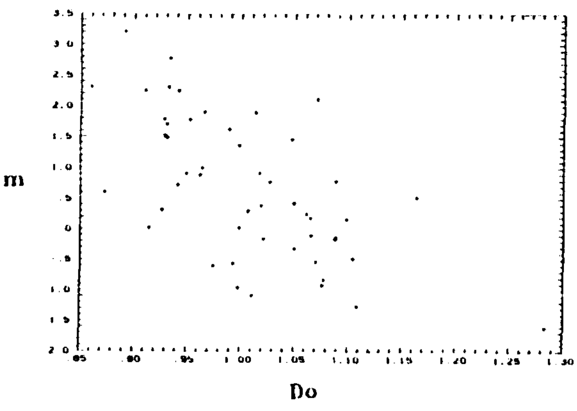


Fig. 9. Scatterplot of (m, D_0) estimate pairs for parameters as in Fig. 7. Note the very weak correlation between the estimates.

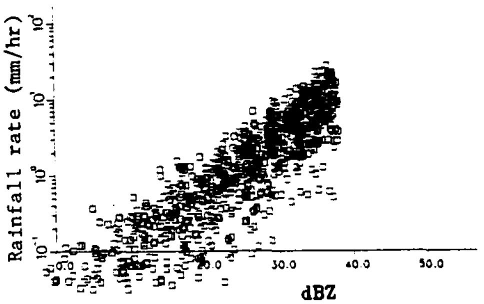


Fig. 6b. Same as Fig. 6a with dBZ "measured" by radar. The Doppler spectrum variance has been varied linearly between 1 and 6 m/s in proportion to the values of reflectivity.

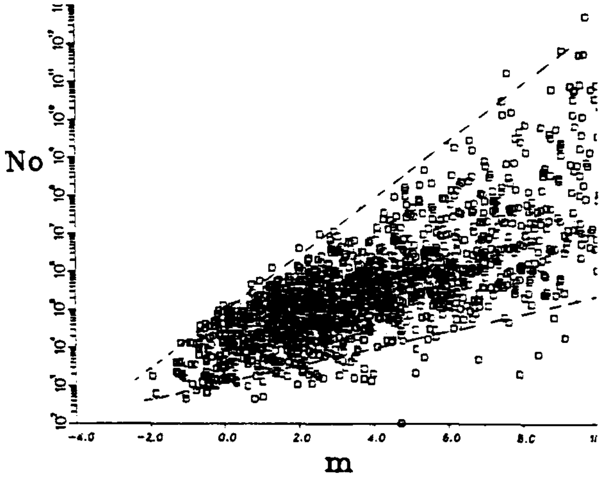


Fig. 8. Global scatterplot of N_0 versus m estimates where N_0 is varied between 200 to 2 x 10^4 (m^{-3} mm^{-1-m}), D_0 between 0.5 to 2.5 mm, and m between 0 to 5. Note that the scatterplot exhibits a correlation structure between log N_0 and m. The two dotted lines indicate the boundaries of (N_0,m) relationships derived theoretically by Ulbrich (1983).

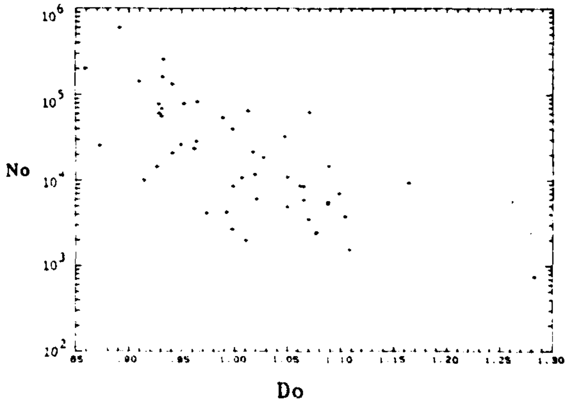


Fig. 10. Scatterplot of (N_0,D_0) estimate pairs for parameters as in Fig. 7. Note the very weak correlation between the estimates.

BENCHMARKING VENDOR PACKAGES

Organizer: *Robert F. Teitel, Teitel Data Systems*

Statistical Database Management on Microcomputers: The Benchmark Problems
Robert F. Teitel, Teitel Data Systems

Making the Push and Shove of Data Management Easier: Examples of Four File Problems
MaryAnn Hill, Laszlo Engelman, BMDP Statistical Software

PRODAS: Professional Database Analysis System
Henry Feldman, Conceptual Software, Inc.

Solving Complex Data Management Problems in P-STAT®
Shirrell Buhler, P-STAT, Inc.

Volume Testing with the PC/SAS® System
Katherine Ng, SAS Institute Inc.

SPSS/PC and Solutions for the Teitel Complex File Problems
Jon K. Peck, SPSS Inc.

The Case of the Missing Data
Leland Wilkinson, University of Chicago and SYSTAT, Inc.; Grant Blank, University of Chicago

Statistical Database Management on Microcomputers: The Benchmark Results
Robert F. Teitel, Teitel Data Systems

STATISTICAL DATABASE MANAGEMENT ON MICROCOMPUTERS: THE BENCHMARK PROBLEMS

Robert F. Teitel

TEITEL DATA SYSTEMS
Bethesda, MD 20814

The purpose of this paper is to present the data files and data manipulation problems posed for the participants of the session on "Benchmarking Data Management Capabilities of Microcomputer-based Statistical Systems" The data files described herein were distributed to 10 vendors of microcomputer-based statistical software, with the intent that they prepare solutions to the problems. These solutions would then be presented at the 18th Interface Symposium, and would be subjected to comparative performance benchmarking on a common machine by the author.

I. INTRODUCTION

The data management problems described herein have a long history. They were first used to provide a focus for my discussant role at the "Workshop on High Dimensional Files: Large or Complex", held as part of the 11th Interface Symposium in 1978. Though the problems circulated in the statistical computing community for some time thereafter, they were not published until Proceedings of the 13th Symposium in 1981 [Eddy 1981]. For that Symposium, data tapes were created and distributed, and six vendors of statistical software on main-frame computers presented solutions to the problems. The problems were published again in the Proceedings of the First LBL Workshop on Statistical Database Management [Wong 1982], together with the solutions of another set of six database and statistical system vendors.

The problems were published a third time as the appendix to a paper on "Statistical Database Management: A Benchmark Comparison Among Statistical and Database Systems" [Teitel 1982b] which concentrated on the different results obtained for one of the problems. Since the results of the required data manipulation were to be presented as simple cross-tabulations, it was quite surprising to discover solutions in which the cell counts differed.

For the present round, the problem descriptions and floppy disks containing the data files were sent to 10 vendors of micro-computer statistical software. (All has agreed in principal to participate in the exercise.) A departure from the previous rounds, in which only data manipulation *functionality* was stressed, this round included a *performance* component. Each of the vendors would send to me their completed solutions to the problems, and enough of their system to permit me to replicate the solutions on a common machine (or two). The results of those performance measures are presented in a subsequent paper.

The problems were designed to elicit responses to three fundamental questions a typical user would have when confronted with the analysis of non-rectangular data, as is becoming increasingly more common. The first question would be "can a potential system perform the necessary data management at all ? ". The second question would be "what do I, the user, have to do to get the system to perform the necessary data manipulation ? ". These two questions were the basis of the earlier functional benchmarking exercises. The third, and final, question a typical user would ask is "what resources does the system consume while performing the necessary data manipulation ? ". Hence our present interest in comparative performance analysis.

Though appearing at first glance to be an *ad hoc* set of data descriptions and tabulatory requests, the problems are based on an underlying concept or model of statistical database management articulated elsewhere [Teitel 1982a].

II. THE PROBLEM SET

The problem set consists of the description of two data collections, and two data manipulation exercises for each data collection. The data manipulation exercises are stated in terms of the desired result, that is, simple cross-tabulations. The format or structure of the tabulations is not of primary concern here; it is the data manipulation necessary to prepare the data for the tabulation step which is of interest here.

A. The TRIPS Data Collection

TRIPS is a large collection of data cosisting of four groups of variables (variously called segments, relations, tables, levels, or record types). The groups of variables are related to each other as shown in the following diagram, and further explained in the text.

```
          +-----------+
     +---! HOUSEHOLD !----+
     !   +-----------+    !
     !                    !
+------+             +----------+
! CARS !             ! PERSONS  !
+------+             +----------+
     !                    !
     !                    !
     !               +----------+
     +---------------!  TRIPS   !
                     +----------+
```

Each household contains a variable number of cars and a variable number of persons; each person contains a variable number of trips. We include zero in the definition of "variable number". In addition, each trip, if taken in a car owned by the household, will contain the identification of that car. There are no counts of persons or cars in the household record, nor is there a count of trips in the person record. The car record contains a model-year variable, the person record an age variable, and the trip record a duration variable in addition to the own-car variable already mentioned. All records contain appropriate identification or key variables, as described in the record layout section.

The first of the two cross-tabulations to be produced from the TRIPS data collection is a simple count of households by the number of cars owned and by the number of persons over the age of 16 in the household. The table should be something similar to the following. (The table definition should have made more explicit the possibility of 0 cars and/or 0 adults by including a 0 car row and a 0 adult column. As given, some vendors excluded those households with 0 cars and/or 0 adults.)

```
+-----------+-----------------------+
! household !    persons over 16    !
!   count   !  1    2    3    4     !
+-----------+-----------------------+
!           !                       !
! cars   1  !                       !
!           !                       !
!        2  !                       !
!           !                       !
!        .  !                       !
+-----------+-----------------------+
```

The second cross-tabulation to be produced from the TRIPS data is a simple count of trips of at least three days' duration taken in a car owned by the household by the age of the person taking the trip by the model year of the car, as shown at the top of the next column.

```
+-----------+-------------------------+
!   trip    !   model year of car     !
!   count   !..  1970  1971  1972  .. !
+-----------+-------------------------+
! age   ..  !                         !
!       16  !                         !
!       17  !                         !
!       18  !                         !
!       19  !                         !
!       ..  !                         !
+-----------+-------------------------+
```

The two tabulations, though superficially similar, require different complex data manipulation capabilities. Each table consists of a count of occurrences of one record type based on variable values in other record types. The first table requires "downward" access, from household to cars and persons; the second table "upward access", from trips to persons and cars.

B. The PEOPLE Data Collection

PEOPLE is a large data collection of various data elements on people. The data collection is relatively closed with respect to ancestry: for most people, data on their parents, grand parents, etc., as well as on their children are contained in the data collection. The variables available for each person include the year of birth, the level of education, sex, and the identification of the mother and father, if known. The records of (an apparently rectangular file) are in identification number order, but not necessarily in any relationship order.

The first of the two cross-tabulations to be produced from the PEOPLE data collection is a simple count of offspring by the educational level of each parent, as follows.

```
+-----------+-------------------------+
!offspring  !  father's education     !
!  count    !..  elem   hs   coll   ..!
+-----------+-------------------------+
!mother's   !                         !
!education  !                         !
!       ..  !                         !
!     elem  !                         !
!       hs  !                         !
!       ..  !                         !
+-----------+-------------------------+
```

The second cross-tabulation to be produced from the PEOPLE data collection is a count of the "last births" by the age of the mother at the birth of her last child by the sex of that last child. The count of "last births" could be interpreted as either the count of mothers at last birth or as a count of the youngest child of each mother.

```
+-----------+----------------+
!  last     !    child sex   !
!  birth    !  female  male  !
+-----------+----------------+
!mother's   !                !
!age     .. !                !
!        15 !                !
!        16 !                !
!        17 !                !
!        18 !                !
!        .. !                !
+-----------+----------------+
```

These tabulations also require different data manipulation capabilities, and different access paths to the necessary data. The first requires simple "upward" access to the education variables of the parents. The second tabulation, if viewed as a count of mothers, requires scanning the children to determine the youngest; if viewed as a count of the youngest child, it requires scanning the siblings to determine the youngest.

III. THE DISTRIBUTION FLOPPY

Programs have been written to generate the data for the TRIPS and the PEOPLE data collections. The programs permit the specification of the basic structure parameters, how many households, or how many trips per person, and the ranges of the values of each of the variables. The values of the variables necessary for the tabulations have reasonable values; the values of other filler variables are immaterial.

A. The TRIPS Data Collection is distributed as four separate files, named HHOLDS.DAT, CARS.DAT, PERSONS.DAT, and TRIPS.DAT, containing approximately 12k, 25k, 38k, and 115k bytes, respectively, in MS-DOS 2.00+ format. The record formats for these files are as follows.

```
Household record:
      1-4   : household identification.
       5    : record type = '1'.
      6-9   : ignore.
     10-60  : 51 one-digit variables.
     61-100 : 20 two-digit variables.
    101-120:  5 four-digit variables.

Car record:
      1-120: as above, except for
       5   : record type = '2'.
      6-7  : car within household id.
      The last four-digit variable is
      the model year of the car.

Person record:
      1-120: as above, except for
       5   : record type ='3'.
      6-7  : person within household id.
      The first two-digit variable is
      the age of the person.

Trip record:
      1-120: as above, except for
       5   : record type = '4'.
      6-7  : person within household id.
      8-9  : trip within person id.
      The first one-digit variable is
      the id of the car used for the
      trip (with a value of 0 meaning
      other conveyance).
      The last two-digit variable is the
      duration of the trip.
```

All variables of all record types should be retained in the system file or database.

B. The PEOPLE Data Collection is distributed as a single file named PEOPLE.DAT of approximately 95k bytes in MS-DOS 2.00+ format, and consists of only a single record type. The format of the record is as follows.

```
People record:
       1-5    : person id number.
       6-56   : 51 one-digit variables, the
                first of which is the sex
                variable ('1'=female,
                '2'=male).
      57-86   : 15 two-digit variables, the
                first of which is the
                education variable (01 .
                19 years of schooling)
      87-110: 6 four-digit variables, the
                first of which is the year
                of birth (1800,     .)
     111-115: the id of this person's mother
     116-120: the id of this person's father.
```

Again, all variables should be retained in the system file or database

IV. REFERENCES

Eddy, W.F. (1981), ed., Computer Science and Statistics: the 13th Annual Symposium on the Interface. Springer-Verlag, New York.

Wong, H.T.K. (1982), ed., The First LBL Workshop on Statistical Database Management, Lawrence Berkeley Laboratory, Berkeley, CA

Teitel, R.F. (1982a), "A Statistical User Interface to the Relational Model of Data", in Proceedings, Part II, of the Conference on Easier and More Productive use of Computer Systems, ACM SIGSOC BULLETIN, V13,02-3.

----- (1982b), "Statistical Database Management: A Benchmark Comparison among Statistical and Database Systems", in Proceedings of the Statistical Computing Section, ASA, Washington, DC.

Making the Push and Shove of Data Management Easier: Examples of Four File Problems

MaryAnn Hill and Laszlo Engelman

When data are collected they frequently fail to form the nice tidy rectangle required for analyses based on classical statistics. Often research projects suffer considerable time delays and extra costs by not having easy-to-use tools to ready the data for analysis. A new system developed by BMDP, the Data Manager, makes such tasks easier. We use the DM system here to solve four file problems. DM is a file manipulating tool designed to handle most file problems encountered in research projects, but it is simple enough that users comfortable with packaged statistical software can specify complex operations without help from a programmer. DM includes 20 instruction paragraphs (READ, SORT, AGGREGATE, etc) that can be assembled in a variety of ways, allowing the separation of a complex task into small manageable pieces that are assembled step by step in a logical manner. The resulting collection of paragraphs reads easily and is self documenting.

The four problems assigned to software developers at the 1986 Interface meetings illustrate several of the functions of DM. The first two problems use four separate files containing records, respectively, for cars, people, households, and trips. In Problem 1, information aggregated from the car and people files, is added to the household file. In Problem 2, the trips file is expanded by adding information from the car and people files. Values are replicated when cars and/or people take multiple trips.

The last two problems concern a file containing information about several generations of people. In Problem 3, records for each person's parents are found (if present) and information on each parent's education is added to each of their children's records. In Problem 4, the last child born to each mother is identified and its birth year used to compute the mother's age at the birth of that child.

We first present our strategy and instructions for the solution of these four file problems and then highlight additional supports the user may need to accomplish these tasks for a real study. It should be pointed out that the initial instructions are not a minimal set. We included additional instructions that a user with similar tasks should consider. For example, in the first problem about cars, people and households we could have ignored the household records and simply used the household id to link the aggregated results for cars and people. But we wanted to identify households with neither cars nor people, because in a real study, they may indicate errors. In the kid/mom/pop task we included instructions to sort the file, even though, by eye, we could see that the generated data were sorted properly for the merge operations.

Strategy and Solutions for the Four Problems

Problem 1. We begin with three separate files with information about cars, people, and households. Among the 79 variables stored on the car record there is a household id, a car id, and the model year of the car (it will be used in Problem 2). The people records also contain a household id plus the person's age and id. The household records contain no car or people identifiers.

The goal is to tabulate the number of adults over 16 years in each household by the number of cars owned by the household. The results obtained from the table program BMDP 4F are displayed in Figure 1.

# cars			# adults		
	no_one	no_adult	1 or 2	3 or more	TOTAL
0	3	2	5	11	21
1 or 2	6	4	15	17	42
3 or more	6	3	15	18	42
TOTAL	15	9	35	46	105

Figure 1 The number of cars and adults per household

These data are weird. There are twelve households with no people (6+6 in the first column); but the records indicate the household has one, two, or more cars. Possibly of greater concern are the seven (4+3) households that only report kids under 16 years, but report multiple cars.

In Figure 2, we display the DM instructions to create the input data file for this table. These instructions aggregate the car and people information by household and join it to the records in the household file. That is, we want the number of cars with each unique household id and the number of people over 16 years with each unique household id.

More specifically, the instruction paragraph labeled #1, in Figure 2, READS the CAR file with 210 records. Each record contains 79 variables stored in fixed locations. The model year of the car is variable 79. We use BMDP's convenient FORTRAN type format reader where the specification to read a four character field is written F4 instead of F4 0.

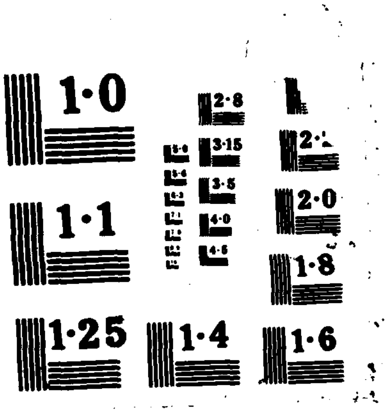
Note that the first dozen lines of Figure 2 document the instructions. The notation #i is used to reference instruction paragraphs below. On each line of instructions, the DM reader ignores text following a number (#) sign. For easier reading, we incented the instructions within each paragraph. This is not necessary, for DM instructions are written in free format.

In #2, the N function in the AGGREGATE paragraph is used to count the number of cars per household (home_id). The N function is one of more than two dozen functions available in the AGGREGATE paragraph for extracting summary information from values on different records. Other functions include minimum, maximum, mean, total, slope, etc. The output work file (c_per_h) contains one record per household with home_id and the number of cars.

A similar counting task is carried out in paragraphs #3 and #4. This time we READ the PERSONS file containing records for 315 people. AGE is the 55th variable for each subject. Note that the

AD-A191 296 COMPUTER SCIENCE AND STATISTICS PROCEEDINGS OF THE SYMPOSIUM ON THE INTER..(U) COLORADO STATE UNIV FORT COLLINS T J BOARDMAN 26 AUG 87 AFOSR-TR-88-0153 3/5

UNCLASSIFIED AFOSR-86-0070 F/G 12/6 NL


1·0
2·8
3·15
3·5
4·0
4·5
2·0
1·1
1·8
1·25
1·4
1·6

```
   #  1. Read the car file.
   #  2. Count the number of cars per household.
   #  3. Read the persons file.
   #  4. Count the number of people over 16 years in
   #     each household.
   #  5. Read the household file.
   #  6. Join the 3 files side-by-side linking records
   #     by home_id.
   #  7. When car and/or people info is missing, set
   #     result to 0.
   #  8. Store the file in a system file.
   #  9. Delete files that are no longer needed.
   #------------------------------------------------
READ
   SFILE  IS 'CARS.DAT'.
   VNAMES ARE home_id, (3)car_id, (79)model_yr.
   FORMAT IS 'f4, T1, f2, 2x, 51f1, 20f2, 5f4'.      / #1

AGGREGATE
   WITHIN  IS home_id.
   '# cars' = N ( home_id ).
   NEWFILE IS c_per_h.                               / #2

READ
   SFILE  IS 'PERSONS.DAT'.
   VARIABLES ARE 79.
   VNAMES ARE home_id, (3)personid, (55)age.
   FORMAT IS 'f4, T1, f2, 2x, 51f1, 20f2, 5f4'.      / #3

AGGREGATE
   WITHIN IS home_id.
   '# adults' = N ( home_id, age > 16).
   NEWFILE IS p_per_h.                               / #4

READ
   SFILE  IS 'HHOLDS.DAT'.
   VARIABLES ARE 78.
   VNAME  IS  home_id.
   FORMAT IS 'f4, T1, 4x, 51f1, 20f2, 5f4'.          / #5

JOIN
   FILES ARE HHOLDS, c_per_h, p_per_h.
   KEY IS home_id.
   KEEP  = home_id, '# cars', '# adults'.
   PRINT = 'HC.', 'H..'.
   NEWFILE IS  problem1.                             / #6

TRANSFORM
   IF ('# cars' EQ XMIS) THEN '# cars' = 0.
   IF ('# adults' EQ XMIS) THEN '# adults' = -1.     / #7

SAVE
   CODE  IS problem1.        NEW.
   SFILE IS 'problem1.sav'.                          / #8

DELETE
   FILES = HHOLDS, problem1, c_per_h, p_per_h.       / #9
```

Figure 2. DM instructions for PROBLEM 1. Aggregating car and people information by household.

N function (in #4) incorporates a condition about AGE -- age > 16. That is, within each household id, only the people over 16 years of age will be counted.

The READ paragraph labeled #5 reads 105 household records. In the JOIN paragraph (#6), home_id is used as a key to link the household records with the appropriate counts of cars and adults. We could have omitted the household records from this JOIN FILES list, but we wanted to check for households with no cars and people. A household with car(s) and no people would also be strange, so we insert a request to print the key (home_id) for each of these unusual occurrences. We specify

PRINT = 'HC.', 'H..'.

where a period (.) in each three character literal string indicates a missing record, a letter (H or C) indicates that the record from the Household or Car file is present. So at this point during our interactive run we identify the 12 households with one or more cars and no people and three households with no cars or people. We didn't have to wait to the table making step (Figure 1).

When there is no car record or people record available to JOIN with the household record, DM pads the positions of the values with the missing value flag XMIS. In the TRANSFORM paragraph (#7) we change the XMIS flag for '# cars' to zero. That is, if there is no c_per_h (cars per household) record then the household had zero cars. If the count of adults is missing then there are no people for that household. However note that if a household does have people, but no one is over age 16, the N function produces a zero. We decided to distinguish between no people and no adults, so we changed the XMIS flag (created when there is no p_per_h, people per household, record) to -1.

In the SAVE paragraph (#8), we save the counts of cars and people by household in a BMDP File. When the table program BMDP 4F reads this file, the names '# cars' and '# adults' will be stored with the data. If a household only has people age 16 or less, '# adults' is zero; if there is no one with that household id, the value is -1.

Problem 2. The structure of the file manipulation task in this problem is opposite to that in Problem 1. Instead of aggregating or accumulating across multiple car and people records per household we generate replicates of car and people records to link with trip records -- that is if a car or person makes more than one trip. The goal for this task is to tabulate information for each trip -- model year of the car used by age of the person taking the trip. The results from the table program BMDP 4F are displayed in Figure 3.

AGE	MODEL_YR												
	1971	1972	1973	1974	1975	1976	1977	1978	1979	1980	1981	1982	TOTAL
12 to 15	5	1	13	7	5	11	7	2	10	4	14	12	91
16 to 20	7	5	7	12	9	10	11	5	12	13	5	6	102
21 to 25	11	6	4	10	7	1	14	5	14	11	5	3	91
26 to 30	19	20	8	6	9	6	7	12	7	5	11	8	118
31 to 35	4	11	4	4	8	6	6	8	8	6	17	9	91
TOTAL	46	43	36	39	38	34	45	32	51	39	52	38	493

Figure 3. Model year of car and age of person taking the trip.

In Figure 4, we show the DM instructions to create the input file for this table. The numbers for the steps of the data manipulation tasks continue from Problem 1, because they were executed during the same interactive computer session. The SORT paragraphs (#10 and # 11) ensure that the PERSONS and CAR records are sorted within each Household key (home_id) by person_id and car_id, respectively. AGE is retained (KEEP) on each person's record.

```
 # 10. Sort the PERSONS file using home_id and
 #     personid.
 # 11. Sort the CARS file using home_id and car_id.
 # 12. Read the trips file.
 # 13. Sort the TRIPS file and join it with the
 #     PERSONS file, using home_id and personid.  The
 #     period in the HOTKEY instruction requests that
 #     the person record be replicated when s/he goes
 #     on more than one trip.
 # 14. Sort the 'trips+age' file and join it with the
 #     CARS file, using the home_id and car_id.  The
 #     HOTKEY instruction requests that the car info
 #     be replicated if it goes on more than one trip.
 # 15. Delete records with less than 3 'duration' or
 #     if the car_id code is 0.
 # 16. Store the file in a system file.
 #------------------------------------------------
SORT
   FILE = PERSONS.
   KEY  = home_id, personid.
   KEEP = home_id, personid, age.                   / #10

SORT
   FILE = CARS.
   KEY  = home_id, car_id.                          / #11

READ
   SFILE IS 'TRIPS.DAT'.
   VARIABLES ARE 80.
   VNAMES ARE home_id,(3)personid,trip_id,car_id,
             (75)duration.
   FORMAT  IS 'f4, f1, 2f2, 51f1, 20f2, 5f4'.
   KEEP = home_id, personid, trip_id, car_id, duration.
                                                    / #12
SORT
   KEY      = home_id, personid.                    / #13
JOIN
   FILES    = TRIPS, PERSONS.
   KEY      =  home_id, personid.
   HOTKEY   = 'T.'.
   PRINT    = '.p'.
   DROP     = '.p'.                                 / #13

SORT
   KEY      = home_id, car_id.                      / #14

JOIN
   FILES    = TRIPS, CARS.
   KEY      = home_id, car_id.
   HOTKEY   = 't.'.
   DROP     = '.c'.
   KEEP     = home_id, car_id, age, duration, model_yr.
   NEWFILE = problem2.                              / #14

TRANSFORM
   IF (duration lt 3 OR car_id eq 0) THEN USE = -1./ #15

SAVE
   CODE     = problem2.          NEW.
   SFILE    = 'problem2.sav'.                       / #16
```

Figure 4. DM instructions for PROBLEM 2. Replicating car and people records when each makes multiple trips and joining with trip information.

In instruction paragraph #12, we READ the 945 record TRIPS file. Each input record contains 80 variables and we KEEP the length of the trip (duration) and ids for the respective home, person, car, and trip. We SORT this file by person_id within each household (home_id).

In the JOIN paragraph (#13) we take each person's AGE and link it to their trip record using person_id and home_id as merge keys. The period in the HOTKEY instruction requests that the person record be replicated when s/he goes on more than one trip. We use the instruction

PRINT = '.p'.

to list id's for people who do not go on a trip. The DROP instruction deletes these records from the output work file that now contains trip information plus age. This file is then sorted by home_id and car_id.

In the JOIN paragraph (#14) the MODEL_YEAR of the car used (from the CAR file) is joined with the 'trips & age' records. The period in the HOTKEY instruction requests that the car information be replicated if the car goes on more than one trip.

The instructions for this problem requested that trips lasting less than three days be deleted from the report and also that values of car_id equal to zero not be used. Code 0 for car_id indicates that an airplane or vehicle other than a car was used for the trip. The instruction USE = -1 in the TRANSFORM paragraph (#15) deletes records with short trip duration and/or invalid car_ids. The SAVE paragraph (#16) saves the resulting file for use as input to program BMDP 4F.

Problem 3. In this problem our input data file contains information about people. Each record contains an id, the sex of the person, their level of education (in years), and their year of birth (birth_yr). In addition, the record contains the id of the person's mother (mom_id) and their father (pop_id). Thus, this one rectangular file contains records for kids, moms, pops, grandparents, and possibly great-grandparents. For Problem 3, our goal is to find each persons mother's record and father's record in the file, thus enabling us to link the parents' educational level to each record. We want to tabulate mother's education versus father's education as shown in Figure 5. Obviously the data are generated, note the 123 women who are college graduates who married dolts -- men with only an elementary school education. This table was obtained in program BMDP 4F.

MOM ed	POP ed				
	elemtary	high_sch	college	graduate	TOTAL
elemtary	198	111	0	0	309
high_sch	0	42	111	0	153
college	0	0	42	117	159
graduate	123	0	0	0	123
TOTAL	321	153	153	117	744

Figure 5. Education of mother versus education of father.

Our strategy is to make two copies of the original file. We will take mother records out of one copy, father records out of the other copy and children records from the original file. When we make the mom's file we rename id, education, and birth_yr to mom_id, mom_educ, and mom_b_yr, respectively. For the file copy that we will use as father's data, the id and education are renamed pop_id and pop_educ. Of course each copy of the file has the same records as the original file, but we will use the id's to select only the 'mom' records from the mom file and only the 'pop' records from the pop file. In Figure 6, for child 45 we link her mom's education (15 years) and birth year (1830) and her father's education (18 years) to the values on her record.

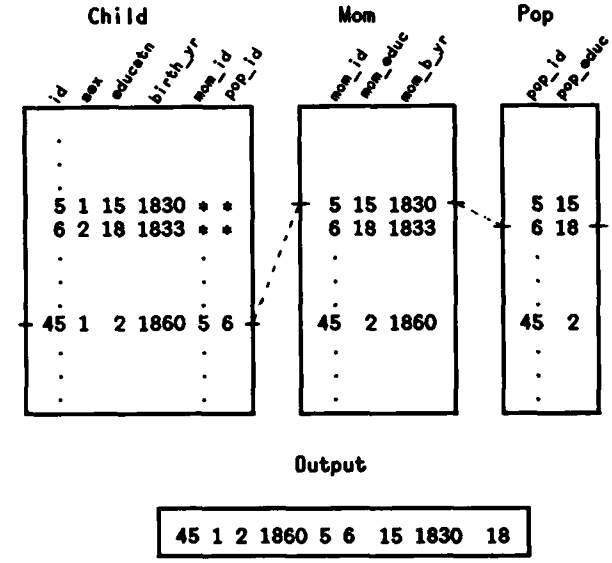


Figure 6. Linking mom 5 and pop 6 with child 45.

In Figure 7, we display the DM instructions to copy the file and link the respective mom and pop data with that of each of their children. In paragraph #1, we READ the PEOPLE file containing 784 records and 75 variables. We will find later that 744 of these 784 people have parents in the file. We will use the 'kids' records in this file and SORT it by mom_id (paragraph #2).

We use the EXTRACT paragraph (#3) to make the first copy of the file. We will use the 'moms' records in this file so we call it 'moms' and rename the variables from id to mom_id, educatn to mom_educ, birth_yr to mom_b_yr.

We next use mom_id as a key to JOIN (#3) the mom's record side-by-side by that of each of her children. The literal string in the HOTKEY instruction includes one position for each file being joined. The period (.) in 'k.' requests that the mother's records be replicated when she has more than one child. When a particular mom_id is present in the moms file but not in the kids file, no output will be made because of the instruction

DROP = '.m'.

```
# 1. Read the PEOPLE file.
# 2. Consider this to be the "kids" file and sort it
#    by mom_id.  Sex and birth_yr are needed in
#    Problem 4.
# 3. Copy (extract) the PEOPLE file and call it the
#    "moms" file (all people in the moms file are not
#    moms, however).  Rename variables, changing id
#    to mom_id, educatn to mom_educ, etc.  Join the
#    kids and moms file side-by-side using the mom_id
#    to link records.  The HOTKEY instruction re-
#    quests that the mother record be replicated when
#    she has more than one child.  The DROP instruc-
#    tion deletes records when a match for the mom_id
#    in the moms file is not found in the kids file.
# 4. Sort the new "kids + moms" file by pop_id.
# 5. Copy the PEOPLE file making it be "pops".
#    Rename id to pop_id, educatn to pop_educ.
#    Join the pops file to the "kids + moms".  The
#    HOTKEY and DROP instructions work as in #3 above
#    but with respect to replicating father's records
#    and eliminating pops without kids.
# 6. Store the file in a system file.
#-----------------------------------------------------
READ
   SFILE    = 'PEOPLE.DAT'.
   VNAMES   = id,sex,(53)educatn,(68)birth_yr,
              (74)mom_id,pop_id.
   FORMAT   = 'f5, 51f1, 15f2, 6f4, 2f5'.             / #1

SORT
   FILE     = PEOPLE.
   KEY      = mom_id.
   KEEP     = id, sex, birth_yr, mom_id, pop_id.
   NEWNAME  = kid_id.
   NEWFILE  = kids.                                    / #2

EXTRACT
   FILE     = PEOPLE.
   KEEP     = id, educatn, birth_yr.
   NEWNAME  = mom_id, mom_educ, mom_b_yr.
   NEWFILE  = moms.                                    / #3

JOIN
   FILES    = kids, moms.
   KEY      = mom_id.
   HOTKEY   = 'k.'.
   DROP     = '.m'.                                    / #3

SORT
   KEY      = pop_id.                                  / #4

EXTRACT
   FILE     = PEOPLE.
   KEEP     = id, educatn.
   NEWNAME  = pop_id, pop_educ.
   NEWFILE  = pops.                                    / #5

JOIN
   FILES    = kids, pops.
   KEY      = pop_id.
   HOTKEY   = 'k.'.
   DROP     = '.p'.                                    / #5

SAVE
   SFILE    = 'problem3.sav'.      NEW.
   CODE     = problem3.                                / #6
```

Figure 7. DM instructions for PROBLEM 3. Linking mother and father records to those of each of their children.

For example, in Figure 6, the code 6 is listed in the moms file as a mom_id, but 6 is not listed as a mom_id in the child's file (actually 6 is an id for a father). A report in the output tells us that 571 records were dropped because an id in the mom's file was not matched in the child's file. Therefore we figure that there were 213 mothers with children (784-571=213). The output work file contains records for children plus their mom's education and her birth year. We SORT this file by pop_id (#4).

We use the EXTRACT paragraph (#5) to make a second copy of the original file. We will use the pops records from this file so we change the name id to pop_id and educatn to pop_educ.

In JOIN (#5) we use pop_id to link the father's education with his child's record. The HOTKEY and DROP instruction are used in the same way as explained previously for the moms file. The output work file from JOIN contains the desired mom and pop values appended to their respective children's records. These records are saved in a BMDP File (#6) as preparation for input to program BMDP 4F.

Problem 4. For this task we are to identify the last child born for each mother and compute the mother's age at the birth of this child. The goal is to tabulate mother's age by sex of this last child. We display these results (obtained from program BMDP 4F) in Figure 8.

```
moms_age                last_sex
________                ________
              Female        Male          TOTAL
___________________________________________________
under_18          4            2     |        6
19 to 25         14           19     |       33
26 to 36         25           17     |       42
over_36          47           85     |      132
___________________________________________________
TOTAL            96          123     |      213
```

Figure 8. Sex of last child by mother's age at birth of last child.

The DM instructions to create the input data file for this table are displayed in Figure 9. As input for this task we use the data file created for PROBLEM 3 that contains records with the child's data plus mother's education, mother's birth year, and father's education. We SORT this file by mom_id (#7).

```
   # 7. Sort the "kids + moms + pops" file by mom_id.
   # 8. For each mom, find the sex of her last child
   #    and her age when the child was born.
   # 9. Store the file in a system file.
   #------------------------------------------------
SORT
   KEY      = mom_id.                             / #7

AGGREGATE
   WITHIN   = mom_id.
   RUSE     = birth_yr EQ MAX(birth_yr).
   last_sex= FVAL(sex).
   moms_age= FVAL(birth_yr - mom_b_yr).
   KEEP     = mom_id, last_sex, moms_age.         / #8

SAVE
   SFILE    = 'problem4.sav'.        NEW.
   CODE     = problem4.                           / #9

FINISH /
```

Figure 9. DM instructions for PROBLEM 4. Identifying the last child born to each mother and computing mother's age at last birth.

Because we just sorted the file by mom_id, the records for the children of each mother form a set. That is, the records for the children of the first mother form the first set, followed by the records for the children of the second mother, etc. The instructions in the AGGREGATE paragraph (#8) are executed for each set of records (all records with the same mom_id). The RUSE (or RECORD USE) instruction selects the record from each set that has the MAXIMUM value of birth year. That is, the latest date or the "last child born" to that mother. The FVAL factor (FIRST VALUE) picks the sex code from the record of this last born child. The age of the mother at birth of her last child is computed in the argument of the FVAL function as the difference between the child's birth year and the mother's birth year. This moms_age and sex code for the last child are output with the mom's id and stored in a BMDP File for input to program BMDP 4F (#9).

Additional Supports for the Data Manipulation Tasks

In estimating the time to solve these four problem, total time to do the task is an important factor. Instead of recording time to execute already debugged instructions, it would be more meaningful, if possible, to record the computer time to assemble and debug the correct set of instructions. Ideally, the shortest time should occur in an interactive setting where the user can access reports and features that identify both mistakes in the program instructions and errors in the data (e.g., incorrect keys), correct the mistakes, and immediately rerun the step in error.

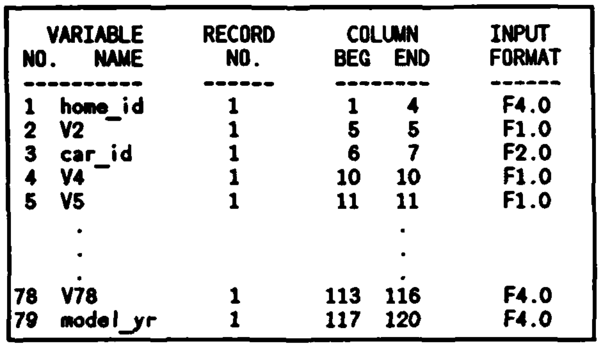
We now describe DM features that we utilized during the interactive development of the correct instructions. As we tackled the four problems we asked many questions. For example,

Did the File Merge Work? After complex file merging operations it is helpful to scan a data listing of the results. In the Problem 3 kid/mom/pop task, we inserted a PRINT paragraph after JOIN (#5 in Figure 7) and immediately identified two women married to dolts (see Figure 10). The mothers of subjects 181 and 186 each have over 18 years of education, but their husbands report only two or three years. We initially thought we had made a mistake but checked the input data and found these results to be correct. Before this point other checks are necessary.

```
 k     s    b      m      p      m     m     p
 i     e    i      o      o      o     o     o
 d     x    r      m      p      m     m     p
            t
 i          h      i      i      e     b     e
 d                 d      d      d           d
            y                    u     y     u
            r                    c     r     c
----   -   ----   ---    ---    ---   ----  ---
 181   1   1870    95     96     19   1852    3
 182   2   1879    97     98      6   1858    9
 183   1   1882    97     98      6   1858    9
 184   2   1885    97     98      6   1858    9
 185   2   1887    99    100     12   1868   15
 186   1   1876   101    102     18   1854    2
       .                              .
       .                              .
       .                              .
```

Figure 10. After a merge command, a data listing is used to check the kid/mom/pop output records in PROBLEM 3.

Did We Specify the Record Format Correctly? After reading the CARS file in PROBLEM 1, DM returns a report on the record format to the screen. We checked this to see if we read the car records correctly. See Figure 11.



```
  VARIABLE        RECORD       COLUMN       INPUT
NO.   NAME          NO.       BEG  END      FORMAT
----------        ------      --------      ------
1  home_id           1          1    4       F4.0
2  V2                1          5    5       F1.0
3  car_id            1          6    7       F2.0
4  V4                1         10   10       F1.0
5  V5                1         11   11       F1.0
    .                                .
    .                                .
    .                                .
78  V78              1        113  116       F4.0
79  model_yr         1        117  120       F4.0
```

Figure 11. The codebook for car records indicates that the model year is stored as the 79th variable in character positions 117 to 120.

In addition, as these records were being read during our interactive session, the DM system reported a record tally every 50 records and the total records in the file after reading was completed. The record tally is also reported for SORT, MERGE, JOIN, and AGGREGATE operations.

What Cases Have Problems? When linking information in PROBLEM 1 from the Household, Cars, and *People Files, we wondered if any households had* cars but no people. The PRINT paragraph (#6, Figure 2) requests a report of household keys for output records with the status 'HC.' where HC means that the Household and Cars records are present and the period (.) indicates that the People record is missing. We also requested a report on keys with the pattern 'H..' (both car and people records are missing for that household). In the resulting report we found 12 households reporting a car or cars and no people and three households with no people or cars. For example, for pattern HC. the keys are 7, 14, 21, 28, etc. report cars and no people; for pattern H.., keys 35, 70, etc.

In PROBLEM 2 (#13) we requested a report listing both the household key and the person id for people who did not take trips. From the resulting report we learned that person 1 in household 4 (the 22nd case in the file) did not take a trip, person 4 in household 5 (the 43rd case in the file) did not take a trip, etc.

Other Conveniences During Interactive Execution. What happens if you misspell an instruction? The Data Manager does not abort the job and drop you into the system. Instead the instructions just executed return to the screen with line numbers; without leaving the program, you can make corrections using the BMDP Line Editor and immediately execute them.

During an interactive session, you can also access system commands without exiting the program. If you forget, say, a file name you type

```
! DIR
```

and your system directory will scroll across the screen. Any system command may follow the exclamation (!) -- when execution of the system command is completed, control returns to DM.

If you forget the name of a DM command, you can request online help by typing, for example

```
HELP READ. /
```

The program then returns a brief definition of READ paragraph commands to the screen.

If you request a printout of your interactive *session your DM* instructions are easy to find and they are readable. A row of equal signs (=) precede and follow each paragraph of instructions, clearly setting the user's instructions apart from DM reports and responses. Scanning such a printout is useful for retracing your steps at a later time or for someone else to join in on the project.

Thus, the Data Manager is a convenient and comprehensive tool for preparing data for analysis; and, in addition, the DM instructions are the same for many systems ranging from the IBM PC to mainframe computers.

PRODAS : PROFESSIONAL DATABASE ANALYSIS SYSTEM

Henry Feldman, Conceptual Software, Inc.

PRODAS is an acronym for the Professional Database Analysis System. PRODAS combines powerful database management with a large array of statistical routines and graphics into an integrated system. It is command driven with syntax similar to SAS (SAS Institute Inc.). PRODAS has the most sophisticated database management and data entry capabilities found in a software system with statistical routines.

In 1985, Dr. Robert Teitel asked developers of the major IBM/PC data analysis packages to participate in a benchmarking problem set. Dr. Teitel wanted to evaluate how the different packages performed in solving data management problems. All vendors were informed that the packages would be run on one computer and timed. We were also told that the timings would be presented at the symposium.

Why test software packages for data management capabilities at a statistical conference? Most statisticians who process data know that an important part of their work is collecting accurate data and getting the data ready for analysis. The ability to quickly and easily restructure data, manipulate databases and produce reports is very important to a statistician.

How should we define quickly and easily? Does quickly refer to the amount of time the computer takes to run a problem, or should it refer to the amount of time the user must spend to solve the problem. Dr. Teitel told the vendors that he would compare the computer time for each package. Therefore, the vendors job was to write a program that had the shortest running time. Since programming time was not measured, we could spend hours, days, or weeks modifying the program for the minimum run time.

At the conference Dr. Teitel raised the question, "Is running time that important?" For example, to load the initial databases took PRODAS 5 minutes, and it took SAS 15 minutes. This means that PRODAS runs 3 times faster than SAS. Now, if it takes 60 minutes to write the program to read the files, then the total time to process the file is 65 minutes for PRODAS and 75 for SAS. As strange as it may seem, the authors of PRODAS, which had the best running times for every problem Dr. Teitel presented, feel that the important timing is the total amount of time (both human and computer) needed to solve the problem.

This is especially important as computers become a faster and cheaper resource. As new generations of computers are developed, users will be able to start up one program and then work on another program. It is better to spend 10 minutes programming and let the computer run for 60 minutes than to program for 60 minutes and let the computer run for 10 minutes. We feel that as long as the answer is arrived at in a timely fashion, it is better to minimize programming time.

Programming environments must make life easier for the programmer. We feel that PRODAS can make life easier; but since Dr. Teitel tested running time, we took advantage of the structure of the databases. I would like to describe how PRODAS can be used for intuitive programming, which is fast and logical to program but runs slower.

Dr. Teitel's second problem was to produce a table of driver's age versus year of car for trips of at least three days' duration. In addition to Dr. Teitel's problems 2 and 3 discussed here, see the Appendix for the solutions to Dr. Teitel's problems 1 and 4. Most of the packages solved Dr. Teitel's problem used the fact that the trip database and the person database were sorted by house and person. The two databases are merged together to produce a temporary database. The temporary database is resorted to match the sorted order of the car database. The temporary database is merged with the car database to find the year of the car. The driver's age and the year of the car are then tabled.

The following is the PRODAS program submitted to Dr. Teitel to solve the second problem:

```
program;
   /*
      Merge the trips database with the
      persons database to match the car
      id and age of driver for trips of
      3 or more days.
   */
   create temp;
   merge trips persons;
   by house person;
   if duration >= 3 and car <> '0' then
      output;
   keep house pers_age car;
   run;
prosort;
   /*
      Module sorts a database.  The
      default database is the last
      created database.
   */
   by house car;
   run;
program;
   /*
      Merge the car id and driver age
      with the car database
      to get the year of the car.
   */
   create temp;
   merge temp (in=intemp) cars;
   by house car;
   if intemp then
      output;
   keep pers_age car year;
   label pers_age = Age of Person;
   label car_year = Model Year of Car;
   run;
descrip;
   table pers_age car_year;
   title Number of Trips;
   run;
```

How would we have written the program if it was important to minimize programming time?

PRODAS has database features that are not found in any other software package with significant data analysis capabilities. PRODAS can randomly retrieve, edit, update and delete records from a database. PRODAS supports an unlimited number of keys per database and an unlimited number of variables per key. By using keyed databases, programming is greatly simplified because you do not have to become involved in the database structure.

Using keyed databases the PRODAS program to solve Dr. Teitel's second problem is greatly simplified. The intuitive solution is to read the trips database, if the trip is 3 days or longer, get the person who drove the car, and get the car for that trip. We will not have to sort any databases and it will not matter what the database order is.

The program module has several commands to randomly process databases. The open statement names the databases that will be processed randomly. The bread statement reads the database randomly. Bread stands for B-tree read (binary read). The 1 following the database name is the key number. Since PRODAS can manage any number of keys, it is necessary to specify the key number.

The new program is:

```
program;
   create temp;
   open persons cars;
   set trips;
   if duration >= 3 and car <> '0';
      /* The above if statement filters
         records based on the expression
      */
   bread persons 1;
   bread cars 1;
   keep pers_age car_year;
   label pers_age = Age of Person;
   label car_year = Model Year of Car;
   run;
descrip;
   table pers_age car_year;
   title Number of Trips;
   run;
```

The above program is simpler, faster to write, and is not dependent on any particular database ordering. It takes longer to run because random accessing is slower than merging two order databases. But the above program took very little time to write and is a more logical solution. Because in the future we will want to minimize programming time in preference to running time since computers are fast and cheap, the second solution is, in general, a better solution.

PRODAS is the only Professional Data Analysis System with capabilities that can solve Dr. Teitel's problem for either minimal computer time or minimal programmer time.

As a second example of how multikeyed databases can simplify a programming task, we will compare the minimum running time and the minimum programming time solution for Dr. Teitel's third problem.

The third problem required generating a table of each child's mother's education versus father's education.

We submitted to Dr. Teitel two versions of the PRODAS solution - general and specific. The general solution assumes the data is too large to store in memory. The specific solution loads the data into memory.

The following is the general solution:

```
program;
   /*
      This version of the education
      level table generation program is
      very general and can work on
      databases of any size.

      The person's education is written
      out as either a mother's or
      father's education for merging
      with descendant.

      The people's database is then
      sorted in mother order to merge
      with the mother database to get
      the mother's education level.

      The people's database is then
      sorted in father order to merge
      with the father database to get
      the father's education level.
   */
   create mother (keep person ed_years)
                 (rename person=mother
                       ed_years=mom_ed);
   create father (keep person ed_years)
                 (rename person=father
                       ed_years=dad_ed);
   set people;
   if sex = '1' then
      output mother;
   else
      output father;
   run;
prosort;
   in=people out=temp;
   by mother;
   run;
program;
   create temp;
   merge temp (in=inpeop)
         mother (in=inmom);
   by mother;
   if inpeop then
      output;
   label mom_ed = Mother's Education;
   run;
prosort;
   by father;
   run;
program;
   create temp;
   merge temp (in=inpeop)
         father (in=indad);
   by father;
   if inpeop then
      output;
   label dad_ed = Father's Education;
   run;
descrip;
   format educate. mom_ed dad_ed;
   table mom_ed dad_ed / missing;
   title Parent's Education Level;
   run;
```

If this same program was written using PRODAS's multikeyed databases, it would be very simple. The intuitive solution to this problem is to read each person, find the mother's education level, and then find the father's education level. Assuming the people database was keyed by the person's id, we can look up the mother and father randomly.

```
program;
   /*
      This version of the education
      level table generation program
      uses the multikeyed databases.

      If the person's parents are
      on the database, the program
      randomly reads the mother's record
      and saves the mother's education
      level. The program then randomly
      read the father's record and saves
      the father's education level.
   */
   create temp (keep = mom_ed dad_ed);
   open people;
   set people;
   if mother > 0 and father > 0;
   father1 = father;
   person = mother;
   bread people 1;
   mom_ed = ed_years;
   person = father1;
      /* The father1 variable is used
         because the current father
         variable has the mother's
         father. */
   bread people 1;
   dad_ed = ed_years;
   label mom_ed = Mother's Education;
   label dad_ed = Father's Education;
   run;
descrip;
   format educate. mom_ed dad_ed;
   table mom_ed dad_ed / missing;
   title Parent's Education Level;
   run;
```

As you can see, the multikeyed solution is much simpler to program and is much more intuitive than the "sort and merge" approach of the other solution. Since PRODAS supports both sequential merging and multikeyed random database accessing, you can decide if it is important to minimize running time or programming time.

Appendix

Solution for Dr. Teitel's second problem.

```
   /*
      This program produces a table of
      trips of at least three days'
      duration. The row and column axes
      are Age of driver and Year of car.
   */
program;
   /*
      Merge the trips database with the
      persons database to match
      the car id and age of driver for
      trips of 3 or more days.
   */
   create temp;
   merge trips persons;
   by house person;
   if duration >= 3 and car <> '0' then
      output;
   keep house pers_age car;
   run;
prosort;
   by house car;
   run;
program;
   /*
      Merge the car id and driver age
      with the car database to get the
      year of the car.
   */
   create temp;
   merge temp (in=intemp) cars;
   by house car;
   if intemp then
      output;
   keep pers_age car_year;
   label pers_age = Age of Person;
   label car_year = Model Year of Car;
   run;
descrip;
   table pers_age car_year;
   title Number of Trips;
   run;
```

Solution to Dr. Teitel's fourth problem:

```
program;
   /*
      This version of the last births
      table generation program
      solves the general case for a
      database of any size.

      For each person, a record is
      written with the mother and
      child's birth date.  The new
      database is sorted by the mother's
      id and child's birth year.  The
      youngest child is then the last
      record for the mother.
   */
   create temp (keep person c_birth
                      childsex);
   set people;
   if mother > 0 then do;
      person = mother;
      c_birth = birth;
      childsex = sex;
      output;
   end;
   label childsex = Sex of Last Child;
   run;
prosort;
   by person c_birth;
   run;
program;
   create temp;
   merge people temp (in=intemp);
   by person;
   if intemp and last.person;
   mom_age = c_birth - birth;
   label mom_age =
       Mother's Age At Lastest Birth;
   run;
descrip;
   format $sex. childsex;
   format momage. mom_age;
   table mom_age childsex;
   title Age of Mothers At Last Birth;
   run;
```

Since Dr. Teitel was timing programs for minimum running time, we submitted two specific solutions for problems three and four that assumed numeric "id" values.

Specific solution for problem 3:

```
program;
   /*
      This version of the education
      level table generation program
      assumes that each person's mother
      and father has preceded
      them in the input file.  Also,
      this program assumes that the
      number of people is sufficiently
      small so the education levels can
      be stored in memory.
   */
   conarray educate [1000];
   create temp;
   set people;
   educate[person] = ed_years;
   mom_ed = educate[mother];
   dad_ed = educate[father];
   label mom_ed = Mother's Education;
   label dad_ed = Father's Education;
   run;
descrip;
   format educate. mom_ed dad_ed;
   table mom_ed dad_ed / missing;
   title Parent's Education Level;
   run;
```

Specific solution for problem 4:

```
program;
   /*
      This version of the last births
      table generation program.
      This program assumes that the
      number of people is sufficiently
      small so the education levels can
      be stored in memory.
   */
   conarray m_birth [1000];
   conarray m_oldest[1000];
   conarray m_sex $1 [1000];
   create temp (keep mom_age childsex);
   set people (end=endpeop);
   if sex = '1' then
      m_birth[person] = birth;
   if m_oldest[mother] < birth then do;
      m_oldest[mother] = birth;
      m_sex[mother] = sex;
   end;
   if endpeop then
      for i = 1 to 1000 do
         if m_birth[i] and m_oldest[i]
                                 then do;
            mom_age = m_oldest[i] -
                               m_birth[i];
            childsex = m_sex[i];
            output;
         end;
   label mom_age =
         Mother's Age At Lastest Birth;
   label childsex = Sex of Last Child;
   run;
descrip;
   format $sex. childsex;
   format momage. mom_age;
   table mom_age childsex;
   title Age of Mothers At Last Birth;
   run;
```

SOLVING COMPLEX DATA MANAGEMENT PROBLEMS IN P-STAT®

Shirrell Buhler, P-STAT, Inc.

ABSTRACT

At the 13th Interface, six different packages submitted their solutions to four problems designed by Robert F. Teitel -- two each for two different data sets. At the 18th Interface, the same four problems were presented, this time to be run on a micro computer. Because P-STAT is functionally identical in all environments, the mainframe solutions presented at the 13th Interface could have been run in P-STAT on the PC with only those changes required by differences in the structure of the raw data files. However, P-STAT has had many enhancements in five years and the current solutions are even easier to program and to comprehend than the 1981 solutions. The solutions presented here are identical for all the machines on which P-STAT is supported. These solutions provide ample evidence that complex data managment problems can be solved on a micro computer such as a PC.

1. PROBLEMS FOR THE FIRST DATABASE

The first two problems involve four files which taken together describe a system with a hierarchical structure.

H A household record with
 C 0 to 9 car records, and
 P 0 or more person records:
 a person record may own many
  T trip records.

The household Id number is contained in all records. A car number provides a link between a car record and a trip record. A person number provides a link between a person record and a trip record. For the purposes of this exercise the only variables of interest aside from the linking variables and the record type are:

Cars file	model year of the car
Persons file	age of person
Trips file	duration of the trip

The record types are: 2=cars, 3=persons and 4=trips. The household file is needed only if a count of the empty households (no people or cars) is desired.

The first problem is to produce a table with counts of the number of people over the age of 16 by the number of cars in a household. The second problem is to produce a table of age of person by model year of car for trips of at least three days duration.

If empty households are ignored, the first problem only requires variables from the Cars file and the Persons file. One approach, if this were the only problem, would be: 1) aggregate the Cars file creating a file of summary records for each household; 2) aggregate the Persons file (selecting people over 16) creating a second file of summary records for each household; 3) join the two household summary files; and 4) tabulate the results of the join. These files are all P-STAT system files.

However, there are two problems to solve. A single sort can be used to arrange the Cars, Persons, and the Trips data (needed for the second problem), into a working file appropriate for both problems. Both tabulations can then be done in a single step although they are shown here as separate steps for the sake of clarity.

2. BUILDING THE FIRST DATABASE

The commands to create four P-STAT system files from the four raw data files are very similar. The command to create the Cars file is:

```
BUILD Cars,  FIXED,   FILE Cars.Dat,
     LENGTH 120;
VARS
 Household.Id 1-4      Record.Type 5
 Car.Number  6-7       C1 TO C51 10-60
 CC1 TO CC20 61-100    C41 TO C44 101-116
 Model.Year 117-120 (ALLOW 1900 TO 1986)$
```

The next step combines the records from the Cars, Persons and Trips files into a single file in the desired order. The Household file was omitted from this step so that houses with neither cars nor people would not be included in the computations.

P-STAT permits multiple files to be dynamically concatenated as they are input to any command. New variables can be created, existing variables can be recoded, and adjacent cases can be combined as the records from each file are processed. Here, the SORT command is used with three input files.

When files are dynamically concatenated using the plus (+) operator, all cases must ultimately have the same variables in the same order. As each case is processed, variables that are not found in the current file are created and set to missing. Finally a variable rearrangement (KEEP) is done so that all the cases in all the files have the same variables in the same order.

The resulting output file has all the records for a household together. The car records are first because Person.Id was set to 0 in the SORT step. Each Person record is followed by zero or more trip records. This file can now be used for either of the crosstabulations.

```
SORT Cars
    ( GENERATE Duration  = .M1.,
      GENERATE Person.Id = 0,
      GENERATE Age       = .M1. )
    ( KEEP Household.Id  Record.Type
           Car.Number    Model.Year
           Duration      Person.Id  Age )
 + Persons
    ( GENERATE Car.Number = .M1.,
      GENERATE Model.Year = .M1.,
      GENERATE Duration   = .M1. )
    ( KEEP Household.Id  Record.Type
           Car.Number    Model.Year
           Duration      Person.Id  Age )
 + Trips
    ( GENERATE Age = .M1.,
      GENERATE Model.Year = .M1. )
    ( KEEP Household.Id  Record.Type
           Car.Number    Model.Year
           Duration      Person.Id  Age),

 BY Household.Id  Person.Id  Record.Type,
 OUT Trips2 $
```

Because comparative timings were to be done, the single SORT using dynamic concatenation was used rather than the following two step procedure.

```
CONCAT Cars
    ( KEEP  Household.Id  Record.Type
            Car.Number    Model.Year )
    ( GENERATE Person.Id = 0 )

   Persons
    ( KEEP   Household.Id  Record.Type
             Person.Id      Age  )
   Trips
    ( KEEP   Household.Id  Record.Type
             Person.Id     Duration ),

 OUT  Trips2 $

 SORT Trips2,
      BY Household.Id  Person.Id
         Record.Type,
 OUT Trips2,  REPLACE $
```

The use of CONCAT followed by SORT requires two passes through the data file. The two step procedure is easier to program and the possibilty of an alignment error in the common variables is eliminated because CONCAT does the alignment automatically. When timings and disk space considerations are not important, the two step procedure is clearly preferable.

2.1 Problem 1

Obtaining the frequencies for the number of cars in a household by the number of persons over 16 is done in a single step using the TABLES command and on-the-fly aggregation in P-STAT's programming language.

```
TABLES Trips2

 ( IF FIRST ( Household.Id ),
   GENERATE #Cars = 0,
   GENERATE #Persons.Over.16 = 0 )

 ( IF Record.Type = 2, INCREASE #Cars )
 ( IF Record.Type = 3 AND Age > 16,
         INCREASE #Persons.Over.16 )

 ( IF LAST ( Household.Id ) CONTINUE )
 ( KEEP #Cars #Persons.Over.16 ) ;

 TABLE  'Household Count'

      Cars  BY  Persons.Over.16  $
```

In this example, the number of cars and number of persons over 16 are computed as the file is given to the TABLES command. FIRST and LAST permit a test for the start and end of a given household. Scratch variables (#Cars and #Persons.Over.16) are set to zero as the first record for each household is processed. #Cars is increased each time a car record, Record.Type = 2, is read. #Persons.Over 16 is increased when Record.Type equals 3 and Age is greater than 16.

Only a single record for each household is actually sent to the TABLES command. That record contains just the two variables, Cars and Persons.Over.16, that are needed for the tabulation.

```
              Household Count

              Persons.Over.16

                                    Row
Cars            0    1-2    3+    Totals
              -----------------
0            |  2 |   5 |  11 |     37
             |    |     |     |
              -----------------
1-2          | 10 |  15 |  17 |     42
             |    |     |     |
              -----------------
3+           |  9 |  15 |  18 |     42
             |    |     |     |
              -----------------

Total N        21    35    46      102
```

NOTE: There were two households with person records that had neither cars nor adults. This table was interactively post-processed and relabelled within the TABLES command so that it would fit within the constraints of this two column layout.

2.2 Problem 2

This second problem requires more from a package than simple aggregation. Both Cars and Persons are at the same level of the hierarchy. Trips are associated with Persons and only indirectly with Cars.

After the SORT, all the car records for a household precede any person records and a person record precedes each set of trip records. It is necessary to store both the model years for up to nine household cars and the age from the person record until a trip record for that person is read. Again, this can be done in the P-STAT programming language as the file is given to the TABLES command.

As each car record is read the model year is stored in the permanent (P) vector. The model year for car number 1 is stored in P(1), the model year for car number 2 is stored in P(2), etc.

As each person record is read, variable Age is stored in the scratch variable #Persons.Age. The P vector and scratch variables are used for aggregation and for moving values from one record to a subsequent record.

```
TABLES Trips2

 ( IF Record.Type = 2,
      SET P(Car.Number)  = Model.Year )

 ( IF Record.Type = 3,
      GENERATE #Persons.Age = Age )

 ( IF Record.Type = 4 AND
      Duration    >= 3 AND
      #Persons.Age AMONG ( 17 to 99 ) AND
      Car.Number > 0, CONTINUE  )

 ( GENERATE Year.of.Car = P(Car.Number))
 ( KEEP #Persons.Age, Year.of.Car  );

 TABLE  ' Trips in Household Cars '
        ' By Family Members Over 16 '
        ' Lasting 3 or More Days '

   Persons.Age  BY  Year.of.Car,
                 EDGES TL $
```

Each trip record is examined to see if: 1) duration is at least three days, and; 2) the current person age is over 16, and; 3) the trip was in fact made by car. This is done in the programming language by using an IF with a series of AND's. The records for which this IF test is true are the only records that the TABLES command receives. Any records that have missing values on any of the variables tested in the IF statement are automatically excluded.

Model year is moved into the selected trip record from the P vector. If the trip is in car number 3, P(Car.Number) references P(3) which is the location where the model year for the third car in the household is stored. The trip record that is given to TABLES contains two variables, Persons.Age, derived from the scratch variable #Persons.Age, and Year.of.Car.

```
          Trips in Household Cars
         By Family Members Over 16
          Lasting 3 or More Days

                 Year.of.Car

Persons     1971- 1975- 1979-    Row
Age         1974  1978  1982   Totals
            -----------------
17-23      |  39    46    47     132
           |
24-30      |  68    46    42     156
           |
31 +       |  23    28    40      91
           |

Total N      130   120   129     382
```

NOTE: this table was also interactively post-processed and relabelled within the P-STAT TABLES command to format the table so that it would fit within the 2-column layout.

3. PROBLEMS FOR THE SECOND DATABASE

The second data set contains records of a geneological nature. Variables in a person's record provide Id numbers for that person's father and mother. Other variables contain information such as sex. education and year of birth. Both of the problems for this data set require information from a given person record to be linked with information from the records of his parents. A zero in the Mother.Id or Father.Id fields indicates a mother or father record that is not present in the file.

ID	SEX	EDUCA TION	YEAR OF BIRTH	MOTHERS ID	FATHERS ID
1	1	16	1929	6	7
2	2	12	1935	6	7
3	1	12	1932	8	0
4	2	8	1905	15	17
5	2	10	1910	15	17
6	2	12	1900	33	44
7	1	12	1898	0	0
8	2	8	1910	0	56
etc.					

Here 6 is the mother of both 1 and 2 and 7 is their father, 8 is the mother of 3 who has no father record in the file.

The third problem is to produce a table of mother's education by father's education. This requires that information from three records in the file be simultaneously available. The fourth problem, which is to produce a table of mother's age by sex of child at last birth, requires that the mother information be combined with information from each of her children's records and that information for all of her children be scanned to locate the youngest child.

4. BUILDING THE SECOND DATABASE

Even though the second data set is a single file, it is tricky because information from several non-adjacent records needs to be simultaneously available. Three steps are required before the tabulations can be done.

```
 BUILD People,  FILE  People.dat,
       FIXED,   LENGTH 120 ;
 VARS
 Person.Id    1-5     Sex  6
 Education   57-58    Birth.Year  87-90
 Mother.Id 111-115    Father.Id  116-120 $
 $

 SEPARATE People  ( KEEP Person.Id
          Birth.Year  Education  Sex  ),
   OUT1 Mothers,   OUT2 Fathers,  EXTRA $

 LOOKUP  People
   ( KEEP   Person.Id   Mother.Id
      Father.Id   Birth.Year   Sex  )
       ( IF Mother.Id < 1 AND
            Father.Id < 1, EXCLUDE ) ,
 TABLE  Mothers

  ( RENAME  Person.Id   TO  Mother.Id,
    RENAME  Birth.Year TO  M.Birth.Year,
    RENAME  Education   TO  M.Education )

   Fathers ( DROP  Birth.Year )
   ( RENAME Person.Id   TO  Father.Id,
     RENAME Education   TO  F.Education ),

 OUT People.Parents $
```

The first step creates a P-STAT system file from the raw data. The second step creates a file of possible mother records and a file of possible father records. This is done in the SEPARATE command which, when provided with the EXTRA identifier, uses the rightmost variable (Sex) to determine the correct output file for each case. That extra variable is not included in the output file. The third step, uses the LOOKUP command to join mother and father information to each record in the People file.

In 1981 the LOOKUP command did not exist and the solution to these problems required several steps. Parent files were created; the child file was sorted by Father.Id and joined to the file of father data; that file was then sorted by Mother.Id and joined to the file of mother data. In 1986, four steps have been replaced by one. The result is fewer passes through the data file, and the solution is far easier to program and to program correctly the first time.

4.1 Problem 3

With the parent information available on all the records, it is only necessary to recode education into education groups and tabulate those groups. The recode is done in the P-STAT programming language as the file is passed to the TABLES command.

```
 TABLES People.Parents

   ( SET  M.Education = RECODE
   ( M.Education, 0 TO 12=1, 13 TO 16=2,
                  17 TO 24=3, X=4 ),

     SET  F.Education = RECODE
   ( F.Education, 0 TO 12=1, 13 TO 16=2,
                  17 TO 24=3, X=4 ));

 LABELS  M.Education  (1) No College
      (2) College        (4) Graduate Work /
          F.Education /,

 TABLE
 'Mothers Education by Fathers Education'
 'Offspring Count'

 M.Education BY F.Education   $
```

Because both variables require the same modification, a FOR loop can be used instead of two separate modifications. Since the modification is sequential, the NCOT function which supplies cutting points, can replace the RECODE.

```
 ( FOR ( J: M.Education  F.Education),
   SET V(J) = NCOT ( V(J),  12, 16 ));
```

This is very easy to program but is not as self-explanatory as the individual RECODEs.

```
   Mothers Education by Fathers Education
               Offspring Count

                 F.Education

                No           Grad
 M.            Coll   Coll   uate      Row
 Education     ege    ege    Work    Totals
              ------------------
 No College   |  351 |  111 |      |    462
              |      |      |      |
              |------------------|
 College      |      |   42 |  117 |    159
              |      |      |      |
              |------------------|
 Graduate     |  123 |      |      |    123
 Work         |      |      |      |
              -------------------

 Total N        474    153    117      744
```

4.2 Problem 4

For the final problem it is necessary to look at all the children for a given mother and find the case with the most recent birth date.

```
 TABLES  People.Parents

   ( COLLECT  20,   BY  Mother.Id,
        SORT  Birth.Year (D),
       CARRY  M.Birth.Year        )

   ( GENERATE Age.Last.Birth = NCOT
       ( Birth.Year.1 - M.Birth.Year,
             18, 25, 30, 35 ))
   ( GENERATE Sex.Last.Born = Sex.1 );

 LABELS
 Sex.Last.Born (1) male (2) female /
 Age.Last.Birth(1) 18 and under (2) 19-25
               (3) 26-30        (4) 31-35
               (5) 40 and over /,

 T    " Mother's Age at Last Childbirth "
      " By Sex of Child "

   Age.Last.Birth  BY Sex.Last.Born  $
```

The ability to COLLECT a number of adjacent cases and combine them into a single case provides the easiest solution to this problem. There may be up to 20 children per mother. The BY and CARRY variables are stored only once in the collected case because they are the same for all children of a given mother. Other variables are stored 20 times, once for each possible child, with a suffix added to the variable name so that each will be uniquely addressable. Thus Sex.1 contains the sex of the first child in the collected case.

SORT requests that a case be stored within the collected case in its sort order on one or more variables. By specifying a sort on Birth.Date and using (D) to indicate a downwards sort, the variables for the youngest child are placed first in the collected case. Sex.1 is, therefore, the sex of the last born. Mother's age is the difference between Birth.Date.1, the birth date of the last born, and the mother's birth date.

The COLLECT function did not exist in P-STAT in 1981. However, this problem can also be solved by using FIRST, LAST, and comparing the birth date of each child in turn to the value of a scratch variable which is reset each time a new youngest child is found.

With COLLECT the entire procedure takes three simple programming language statements. COLLECT and SPLIT makes it possible to solve problems which are otherwise very difficult, even with FIRST, LAST, and scratch variables.

```
             Mother's Age at Last Childbirth
                     By Sex of Child

                      Sex.Last.Born

 Age Last                                     Row
 Birth              male      female        Totals
                 -----------------
 18 and under   |    4   |     2   |          6
                |        |         |
                |-----------------|
 19-25          |   14   |    19   |         33
                |        |         |
                |-----------------|
 26-30          |   25   |    17   |         42
                |        |         |
                |-----------------|
 31-35          |   47   |    85   |        132
                |        |         |
                 -----------------

 Total N            90       123            213
```

5. SIZE ISSUES

The raw data files for both data sets contain many extra data values. They were included to see if the packages could handle large numbers of variables and to ascertain how much disk space the resulting files would need. P-STAT, like the other packages in the test, had no trouble handling the complete data set. However, because P-STAT uses a very aggressive algorithm for packing data values, the resulting P-STAT system files required less disk storage than the files produced by the other packages.

6. CONCLUSIONS

The session was titled "Benchmarking Vendor Packages". Given that all the vendor packages could solve the problems, there is some question about what was actually benchmarked and what the benchmarks mean. Three areas that can be compared are: 1) ease of use; 2) speed; and 3) use of resources.

Ease of use is difficult to evaluate. The best measure, short of a carefully designed experiment with novice users, is subjective: how easy is it to follow the command stream without reading the explanatory material. This will, to a large extent, depend on the reader's background and familiarity with packages. P-STAT in 1986 has a language that is easier to use than P-STAT in 1981. This is particularly evident in the COLLECT function and the LOOKUP command. This is a trend that will certainly continue.

In some of the problems illustrated here, the most readable sequence of commands is not the fastest. Ease of use and speed are two areas which should probably not be benchmarked simultaneously. P-STAT's timings were acceptable, as were those of most of the other packages.

The resources needed to run a program can be measured. P-STAT on a PC requires 640K, a math co-processor, and a hard disk. This puts P-STAT near the high end in terms of PC requirements. However, P-STAT is modularized so that not all of the program needs to be installed on the hard disk. In addition, P-STAT's system files require less disk storage than other packages because of the aggressive packing algorithms that are used.

The PC is a viable tool for complex data management problems. Data sets both larger and more complex than these problem data sets can be handled easily by packages such as P-STAT.

VOLUME TESTING WITH THE PC/SAS® SYSTEM

by Katherine Ng, SAS Institute Inc.

Abstract

This paper discusses the use of the PC/SAS system to solve the 4 problems posed by Dr. Robert Teitel. The problems are designed to test the complex data manipulation capabilities of the statistical/database systems currently available to the PC users. With the PC/SAS system, the tables requested in the 4 problems can easily be generated by a SAS tabulation procedure such as **FREQ** *or* **TABULATE** , *after the appropriate data bases have been created.*

Problem Descriptions

The first 2 problems use the TRIPS data collection. It has 4 components, which are henceforth referred to as the household, car, person, and trip records. Each household has a variable number of cars and a variable number of persons. Each person of a household took a variable number (0 to 99) of trips. Neither the number of persons nor the number of cars in a household, nor the number of trips taken by a person is explicitly coded. The person record contains the age of the person and his household identification. The car record has the model year of the car and identifies the household to which it belongs. The trip record identifies the person of a household who took the trip, the duration of the trip, and the car identification code if a family car was used. All records are in household identification order.

Two tabulations are requested. The first is a frequency distribution of households for the number of cars owned by the number of persons over the age of 16 in the household. The second is a frequency distribution of trips of at least 3 days' duration taken in a car owned by the household by the age of the person taking the trip and by the model year of the car.

The last 2 problems use the PEOPLE data. Each record in the PEOPLE data collection contains a person's identification code, sex, birth year, level of education, and the parents' identification codes if known. Records are in identification order.

Again, 2 tabulations are requested. The first is a frequency distribution of offspring by the education level of each parent, and the second, a frequency distribution of the "last births" by the age of the mother at the birth of her last offspring and by the sex of that last offspring.

The TRIPS Data

With the variability of the number of persons in a household, the number of cars belonging to a household, and the number of trips taken by a person of a household, it is best to keep the data in separate components. SAS system files for the different components are created for efficient retrieval in later analysis. To assess PC/SAS ability to handle data for a large number of variables, all given data are retained in the system files. Working SAS data sets with only the relevant variables and cases are created to gather information from the component files. These working data sets are then passed to a tabulation procedure to generate the desired tables.

Figure 1 has the SAS statements that build the 4 component files for the TRIPS data base and extract subsets of data into working data sets.

```
/*---------------------------------------*/
/*-------- Create SAS data sets ---------*/
/*----- HOUSES, CARS, PERSONS, TRIPS ----*/
/*---------------------------------------*/
DATA a.houses;
     infile "houses.dat"  lrecl=120;
     input house $1-4 type $5
           @10 (one1-one51)($char1.)
           (two1-two20)($char2.)
           (four1-four5)($char4.);
     run;

DATA a.cars;
     infile "cars.dat"  lrecl=120;
     input house $1-4 type $5 car 6-7
           @10 (one1-one51)($char1.)
           (two1-two20)($char2.)
           (four1-four4)($char4.)
           year 117-120;
     run;

DATA a.persons;
     infile "persons.dat"  lrecl=120;
     input house $1-4 type $5 person $6-7
           @10 (one1-one51)($char1.)
           age 61-62
           (two2-two20)($char2.)
           (four1-four5)($char4.);
     run;

DATA a.trips;
     infile "trips.dat"  lrecl=120;
     input house $1-4 type $5
           person $6-7 trip $8-9 car 10
           (one2-one51) ($char1.)
           (two1-two19)($char2.)
           days 99-100
           (four1-four5)($char4.);
     run;

/*------------------------------------------*/
/* Create temporary data sets :             */
/*    CARS, PERSONS, and TRIPS              */
/* Extract relevant variables and cases     */
/*------------------------------------------*/
DATA cars;
     set a.cars(keep=house car year);
     run;

DATA persons;
     set a.persons(keep=house person age);
     run;

DATA trips(drop=days);
     set a.trips(keep=house person car days);
     if car and days >= 3 then output;
     run;
```

fig. 1.

A Solution To Problem 1

Figure 2 shows how the different components of the TRIPS data base are merged to create a temporary SAS data file that contains only the information needed for table 1. The frequency distribution of the number of households by the number of cars owned and by the number of persons over the age of 16 in the household is easily generated by PROC FREQ. Figure 3 has the codes that generated the table shown in figure 4. Note that the data have been classified into a few groups to produce a more pleasing and compact table. In the SAS system, the grouping of data is easily done by defining a format variable with PROC FORMAT and using this format in the appropriate procedure for the appropriate variables.

```
/*---------------------------------------*/
/* Create temporary data set HHOLDS of    */
/* household records, with variables      */
/*   ncars (# of cars in the household)   */
/*   over16(# of persons over age 16)     */
/*---------------------------------------*/
DATA HHOLDS(keep = ncars over16);
     merge a.houses(in=h keep=house)
           persons
           cars;
           by house;
     length lperson $2;
     retain lcar lperson;
     ncars +(lcar ^= car);
     over16+((lperson ^= person)*(age>16));
     lcar=car; lperson = person;
     if last.house;
     if h then output;
     ncars = 0; over16 = 0;
     lcar = .;  lperson = "  ";
     run;
```

fig. 2.

```
/*---------------------------------------*/
/* Set up format for the count variables */
/*---------------------------------------*/
PROC FORMAT;
     value countfmt 0      = "0"
                    1-2    = "1 - 2"
                    3-high = "3+";
     run;

/*---------------------------------------*/
/* Generate the table for problem 1      */
/*---------------------------------------*/
TITLE3
  "Frequency Distribution of Households";
PROC FREQ;
     tables ncars*over16/
            nocol norow nopercent;
     format ncars over16 countfmt.;
     run;
```

fig. 3.

```
         VOLUME TESTING - TRIPS DATA
     Frequency Distribution of Households

          TABLE OF NCARS BY OVER16

NCARS     OVER16

Frequency|       0|   1 - 2|      3+|  Total
---------+--------+--------+--------+
       0 |      5 |      5 |     11 |     21
---------+--------+--------+--------+
   1 - 2 |     10 |     15 |     17 |     42
---------+--------+--------+--------+
      3+ |      9 |     15 |     18 |     42
---------+--------+--------+--------+
Total          24       35       46      105
```

fig. 4.

A Solution To Problem 2

Problem 2 also requires that information be gathered from across the different components of the data base. For each trip, we need the model year of the car used in the trip from the CARS file, and the age of the driver from the PERSONS file. With a series of sorting and merging steps, we can collate all that information into one data file, which we pass on to PROC FREQ to generate the desired table.

The PEOPLE Data Base

The last 2 problems need a data manipulation scheme quite different from the first 2 problems. The solutions to problems 1 and 2 require accessing information across the components of each record of the data base, whereas solutions to problems 3 and 4 require collating information from the different records of the same data file.

Again, a SAS system file is created for the PEOPLE data with all the variables kept. Retaining the data for all the 75 variables is not necessary for problems 3 and 4, but it has been done for completeness. A working SAS data set with only the relevant variables and cases is created to facilitate multipassing of the data. See figure 5.

```
/*----------------------------------------*/
/* Create a permanent SAS data set PEOPLE. */
/*----------------------------------------*/
DATA a.people; infile "people.dat" lrecl=120;
     input id $1-5  sex $6
     (one1-one50)($char1.) educ 57-58
     (two1-two14)($char2.) birthyr 87-90
     (four1-four5)($char4.)
     mom $111-115  dad $116-120;
     run;

/*----------------------------------------*/
/* Extract subset of relevant variables    */
/*----------------------------------------*/
DATA people
     set a.people
         (keep=id sex educ birthyr mom dad);
     run;
```

fig. 5.

A Solution to Problem 3

Problem 3 asks for a frequency distribution of the number of offspring by the education level of the parents. We note from the data description, the only information about the parents contained in each offspring's record is the parents' identification codes. Thus, the solution necessitates locating the parents records (if available) for each offspring, and collating their education information onto the offspring's record. Figure 6 shows the steps involved. With the information about the parents' education collated on the offspring's record, the table for problem 3 is easily produced by PROC FREQ.

```
/*---------------------------------------*/
/* Extract subsets consisting of male    */
/* records for the FATHER's file and     */
/* female records for MOTHER's file      */
/*---------------------------------------*/
DATA father(rename=(id=dad educ=dad_ed)
            keep=id educ)
     mother(rename=(id=mom educ=mom_ed
                    birthyr=momborn)
            keep=id educ birthyr);
     set people;
     if sex = "1" then output mother;
     if sex = "2" then output father;
     run;

/*------------------------------------------*/
/* Create the file FAMILY by collating the */
/* father's record and the mother's record */
/* into the offspring's record             */
/*------------------------------------------*/

/* sort PEOPLE by dad's id to match up     */
/* dad's id with his own record from the   */
/* FATHER's file                           */
PROC SORT data=people;    by dad; run;

/* father's record merged with child's     */
DATA family(drop=dad);
     merge people(in=child
                  keep=sex dad mom birthyr)
           father;  by dad;
     if child then output;
     run;

/* Sort FAMILY by mom's id to match up     */
/* mom's id with her own record  from the  */
/* MOTHER's file. Use child's birthyr as a */
/* second sort variable                    */
PROC SORT data=family; by mom birthyr; run;

/* mother's record merged with child's     */
DATA family;
     merge family(in=child) mother; by mom;
     if child then output;
     run;
```

fig. 6.

A Solution to Problem 4

The last table is the frequency distribution of the "last births" by the age of the mother at the birth of her last offspring and by the sex of that last offspring. The data set FAMILY created in the previous steps have most of the information needed for problem 4. It remains to find the record of the youngest child of each mother, and use the child's birth year and the mother's birth year to compute the mother's age (figure 7). PROC FREQ can be used to generate the desired table.

```
/*---------------------------------------*/
/* Create data set which contains records */
/* the records for the youngest children.  */
/* Compute mom's age.                      */
/*---------------------------------------*/
DATA youngest(keep=sex momage);
     set family; by mom;
     if last.mom then
        if mom ^= "    0" then do;
           momage = birthyr - momborn;
           output;
           end;
     run;
```

fig. 7.

Timing Estimates

For purposes of comparison and performance evaluation, some timing estimates for the execution of the different SAS program steps have been included (figures 8 and 9). All the steps have been done on an IBM PC/AT model 99 with 640k bytes of memory. The machine used has a numeric coprocessor. Note that there are a lot of variables retained in each data set, hence the DATA step took a proportionately large amount of time. However, after the SAS system files have been created, data retrieval is fairly fast. In fact, the step which gathered all the necessary information for the first tabulation took only 27 seconds, the actual tabulation took 15 seconds. We also note that sorting is very fast.

TRIPS Data Base		
Create		
A.HOUSES	78 vars, 105 obs	47 secs
A.CARS	79 vars, 210 obs	77 secs
A.PERSONS	79 vars, 315 obs	107 secs
A.TRIPS	80 vars, 945 obs	278 secs
Extract		
CARS	3 vars, 210 obs	18 secs
PERSONS	3 vars, 315 obs	20 secs
TRIPS	3 vars, 493 obs	29 secs
Solution to Problem 1		
Collate Data		
HOUSES+PERSONS+CARS	2 vars, 105 obs	27 secs
Tabulate data	3 x 3 table	15 secs
Solution to Problem 2		
Collate Data		
Sort PERSONS	2 keys, 315 obs	13 secs
Sort TRIPS	2 keys, 493 obs	14 secs
TRIPS+PERSONS	2 vars, 493 obs	27 secs
Sort TRIPAGE	2 keys, 493 obs	18 secs
Sort CARS	2 keys, 210 obs	12 secs
TRIPAGE+CARS	2 vars, 493 obs	27 secs
Tabulate data	3 x 3 table	22 secs

fig. 8.

PEOPLE Data Base		
Create		
A.PEOPLE	75 vars, 784 obs	215 secs
Extract		
PEOPLE	6 vars, 784 obs	28 secs
FATHER	2 vars, 392 obs	33 secs
MOTHER	2 vars, 392 obs	
Solution to Problem 3		
Collate Data		
Sort PEOPLE	1 key, 784 obs	17 secs
PEOPLE+FATHER	4 vars, 784 obs	32 secs
Sort FAMILY	2 keys, 784 obs	24 secs
FAMILY+MOTHER	6 vars, 784 obs	34 secs
Tabulate data	5 X 5 table	25 secs
Solution to Problem 4		
Extract Data		
YOUNGEST	2 vars, 213 obs	24 secs
Tabulate Data	4 x 2 table	20 secs

fig. 9.

Conclusion

The PC/SAS system is intended to be a complete implementation of the SAS system as available to the mainframe and mini computer users. It has the same set of data management tools, and can perform complex data analysis with as much ease. Inspite of the limitations of the personal computers and the complexity of the language, execution is reasonably fast. For small and medium size problems, we see it as an alternative to the mainframe and mini systems.

Reference

SAS Language Guide for Personal Computers, Version 6 Edition, SAS Institute, Cary, North Carolina.

SAS is a registered trademark of SAS Institute Inc, Cary, N.C.

SPSS/PC AND SOLUTIONS FOR THE TEITEL COMPLEX FILE PROBLEMS

Jon K. Peck, SPSS Inc.

1: Introduction

This paper shows the results for the Complex File Problems 1 and 2 that were posed to various microcomputer statistical packages. The approach shown here for SPSS/PC+ is the same as it would be for the mainframe SPSS-X package. An SPSS-X solution for the same problem was presented in an earlier paper from the benchmark session at an earlier Interface. A discussion of the solutions for the two problems is followed by a listing of the scripts used.[1] The output is not shown in order to conserve space. The scripts were run as batch jobs, but users could also run these solutions interactively.

2: Parameters of the Solution

The required tables are computed in SPSS/PC+ using multiple system files. A system file is a binary file that contains both the data and the variable and value definitions and label information. By default, these files embody a compression algorithm that often substantially reduces the file size over the uncompressed version at the cost of slightly longer processing time. It is common for the compressed file to be one-third the size of the uncompressed file. The user can, however, instruct the system to use uncompressed files instead. The user can also direct these files to a RAM disk in order to speed execution. SPSS/PC+ will allocate the maximum possible workspace area in memory unless instructed otherwise, but the amount of memory available has no effect on the problems discussed here as long as the system minimum requirements are met. Finally, the presence of the optional math coprocessor can significantly reduce the run time for problems in SPSS/PC+. The output can be formatted for various page widths and character sets according to the printing device to be used.

3: The Primary Commands Used in the Solution

The original ASCII data are read with the DATA LIST command. The other main commands that are used here are GET, SORT, JOIN (MATCH), AGGREGATE, and CROSSTABS. GET selects a system file as the active dataset and dictionary; SORT sorts cases on up to ten variables; JOIN (with aliases MATCH and ADD) performs an (outer) join on up to five files including a table-lookup facility or adds new cases to the file; AGGREGATE combines groups of cases with a choice of aggregation functions and missing value treatments, and CROSSTABS produces n-way tables. All of these facilities are part of the SPSS/PC+ base system. Lines starting with * are comments, and lines starting with DOS are DOS operating system commands. The DOS commands used here simply delete superfluous files at the end of the run, but SPSS/PC+ permits any reasonable DOS command or program to be executed during a run.

4: Complex File Problem 1: Cars

First, the four data sets, Households, Cars, Persons, and Trips, are defined and made into system files containing all of the variables in the original data using DATA LIST and SAVE. Since these system files contain many more variables than are required to compute the required tables, superfluous variables are dropped during subsequent joins and aggregations. Second, HOUSES, PERSONS, and CARS are joined on the household id, HOUSE. An outer join where one file has no matching records produces missing values for all variables taken from that file. Therefore, the car id variable, CAR, is recoded to zero if missing and one otherwise. Thus when the joined file is aggregated by household, the sum of the recoded car variable will give the number of cars owned by the household. An indicator variable is defined for persons with age over 16. This condition was interpreted as "age > 16" which means here age of 17 or more.

Next the joined file is aggregated by HOUSE producing a file in which the unit of analysis is the household and which has variables giving the number of cars, NUMCARS, (zero or more) and the number of persons over 16 (DRIVERS) using the SUM function of AGGREGATE. This function treats missing values as zero unless instructed otherwise, but when there are no records in the group with a nonmissing value, the resulting SUM has the value missing. RECODE is used to designate these missing cases as zero. Finally, the crosstabulation of NUMCARS by DRIVERS produces the first required table. In these data there are households having drivers but no cars and cars but no drivers. The number of households with each possible number of cars is the same.

For the second table, the age of each person taking a trip is added to the trip record by joining the TRIPS system file with the PERSONS system file on household and person ids treating PERSONS as a lookup table. After selecting according to the required conditions of trips of at least three days duration and in the household's own car, the joined file is sorted into house and car order. Next, the model year, variable CARYEAR from CARS, is added to the file as a table lookup based on household id and car number. From this, the table of AGE by CARYEAR is produced by CROSSTABS.

5: Complex File Problem 2: People

For this problem two tables are to be produced: first, the number of offspring classified by father's and mother's education, and second, the number of mothers classified by mother's age at birth of last offspring and sex of that offspring. This problem has only a single input file, which contains data on a set of people including both offspring and parents. This is the structure of the classic employee-manager database.

First, the dataset is defined and converted to the PEOPLE system file using DATA LIST and SAVE, although constructing the required tables does not actually require this system file. Second, the father's education and the mother's education and birth year and the parents' id numbers are added to the active file. This is accomplished by two table-lookup joins of the dataset against itself via SORT and MATCH. The join table is a system file named PERTAB that contains only the variables actually needed. Third, the table of offspring by parents' education is produced by CROSSTAB. The table is constructed with a separate row and column tallying the cases where the parent data are missing.

For the second table, the procedure is as follows. The cases in the active file from the previous table exercise are sorted by mother id and BIRTHYR. Next, for those cases where the mother is known (MOTHER > 0), the file is aggregated over MOTHER retaining the last occurrence of birth year, sex, and mother's birth year (MABIRYR). From this aggregated file, the mother's age at birth of last child, BIRTHAGE, is simply BIRTHYR - MABIRYR. Finally, CROSSTABS computes the required table.

The results of the second exercise reveal a remarkably effective way to determine the sex of ones final offspring.

6: Timing and File Size Data

The time required to complete each task and the various file sizes depend on the computer configuration and the SPSS/PC+ options selected. The system file sizes in K-bytes were as follows.

	Cars	
	Compressed	Uncompressed
HOUSES	16	69
PERSONS	43	202
CARS	30	136
TRIPS	126	609
	People	
	Compressed	Uncompressed
PEOPLE	119	487
PERTAB	28	44

Running times are reported for an IBM PC/AT with an 80287 math coprocessor and for an IBM PC/XT with an 8087 math coprocessor. For the AT, times are reported with the files other than the initial ASCII files stored on a RAM disk and stored on a hard disk. On the XT times are reported for compressed and uncompressed files stored on a hard disk. Times are reported in minutes. An * means that that case was not run.

Machine	PC/AT		PC/XT	
Data Location	RAM Disk	Hard Disk	Hard Disk	
Compression Option	On	On	On	Off
Cars	9.25	10.5	21.5	20.1
People	5.5	*	15.9	13.1

It should be emphasized that these times depend on many particulars of the machine and program settings, which will vary. They do suggest, however, that an AT runs these problems about twice as fast as an XT and that using a RAM disk on an AT for file storage has only a modest effect. For files such as these, compression dramatically reduces the system file sizes at a very modest decrease in execution speed. The XT to AT speed comparison is likely to be quite different for machines without a math coprocessor.

7: Command Script: Problem 1

```
SET  more off echo on compress on width wide
     length 60 listing 'cars.lis'.

TITLE 'Household File'.

DATA LIST  file='persons.dat'/house 1-4 hal to
           ha51 10-60 hbl to hb20 61-100 hdl
           to hd5 101-120.

SAVE out='houses.sys'.

TITLE 'Persons File'

DATA LIST  file='persons.dat'/house 1-4 person
           6-7 pal to pa51 10-60 age pb2 to pb20
           61-100 pcl to pc5 101-120.

SAVE out='persons.sys'.

TITLE 'Cars File'

DATA LIST  file='cars.dat'/house 1-4 car 6-7
           cal to ca51 10-60 cbl to cb20 61-100
           cdl to cd4 caryear 101-120.

SAVE out='cars.sys'.


TITLE 'Trips File'.

DATA LIST  file='trips.dat'/house 1-4 person
           6-7 trip 8-9 owncar ta2 to ta51
           10-60 tbl to tb19 days 61-100
           tdl to td5 101-120.

SAVE out='trips.sys'.

TITLE  'Crosstabulate number of cars by number of
        persons over 16'.

SUBTITLE  'Unit of analysis is households'.

*  count households by the number of cars owned
   and number of persons over 16.

MATCH  file='houses.sys'/keep house/
       file='persons.sys'/keep house age/
       file='cars.sys'/keep house car/by house.

RECODE car (sysmis=0) (else=1).

COMPUTE  age16=age > 16.

*AGE16 equals 1 if person over 16.

AGGREGATE  outfile=*/break=house/
           numcars 'Number of Cars Owned' =
           sum(car)
           drivers 'Number of Persons over
           16 years old' = sum(age16).

FORMATS  numcars drivers (F2.0).

RECODE  numcars drivers (sysmis=0).

SET  EJECT ON.

CROSSTABS  numcars by drivers.

SET  EJECT OFF.
```

```
TITLE 'Crosstabulate long trips by age of person
      and car'.

SUBTITLE ' Unit of analysis is Trips'.

MATCH file='trips.sys'/rename (owncar=car) keep
      car house person days/
      table='persons.sys'/keep age house person/
      by house person.
*LOOKUP PERSON'S AGE

SELECT IF (days ge 3 and car ne 0).
*SELECT TRIPS OF 3 DAYS PLUS.

SORT CASES by house car.

MATCH file=*/keep=age house car/
      table='cars.sys'/keep=caryear house car/
      by house car.

SET EJECT ON.
CROSSTABS age by caryear.
SET EJECT OFF
DOS erase house.sys.
DOS erase person.sys.
DOS erase cars.sys.
DOS erase trips.sys.
FINISH.
```

8: Command Script: Problem 2

```
SET listing 'people2.lis'
    more off echo on compress on width wide
    length 60 workdev d:.

TITLE 'People File'.

DATA LIST file='c:\teitel\people.dat'/person 1-5
          sex a2 to a51 6-56 educ b2 to b15
          57-86 birthyr d2 to d6 87-110 mother
          father 111-120.

SAVE out='d:people.sys'.

GET file='d:people.sys'/drop a2 to a51 b2 to b15
    d2 to d6.

SAVE file='d:pertab.sys'/drop sex father mother.

TITLE "Count persons by father's and mother's
      education".

SORT CASES by father.

MATCH table='d:pertab.sys'/rename (person=father)
      (educ=paeduc)/ keep=father paeduc/
      file=*/keep person father mother educ
      birthyr sex/ by father.

SORT CASES by mother

MATCH table='d:pertab.sys'/rename (person=mother)
      (educ=maeduc) (birthyr=mabiryr)/
      keep mother maeduc mabiryr/
      file=*/by mother.

RECODE paeduc maeduc (sysmis=99).

MISSING VALUES paeduc maeduc (99).

SET eject cn
CROSSTAB paeduc by maeduc/options 1.
SET eject off.

TITLE 'Mothers, children, and all that'.

SORT CASES mother birthyr.

PROCESS IF (mother > 0).

AGGREGATE out=*/presorted/break mother/
          birthyr = last(birthyr)/sex=last(sex)/
          mabiryr = last(mabiryr).

COMPUTE birthage = birthyr - mabiryr.

VARIABLE LABELS birthage "Mother's age at last
                birth"/
                sex 'Sex of last offspring'.

VALUE LABELS sex 1 'Male' 2 'Female'.

SET eject on.
CROSSTABS birthage by sex.
SET eject off.
*DOS erase d:people.sys.
*DOS erase d"pertab.sys.
FINISH
```

[1] The solutions were prepared by ViAnn Beadle and Jon Peck based on the earlier work of Jon Fry for SPSS-X.

THE CASE OF THE MISSING DATA

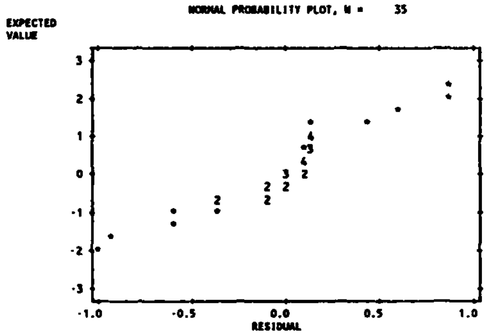
Leland Wilkinson
University of Illinois at Chicago and SYSTAT, Inc.

Grant Blank
University of Chicago

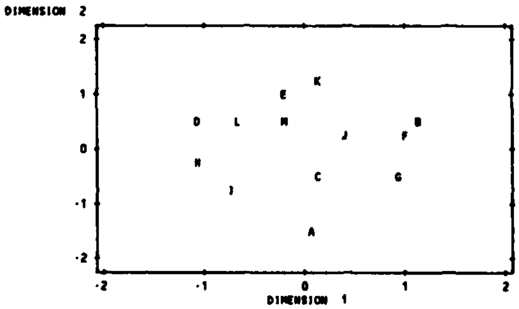
The real data in this exercise are missing. On examining closely the TRIPS and PERSONS data provided by Robert F. Teitel, we are forced to conclude that his data are not real. The following evidence is offered to support our conclusion.

Evidence

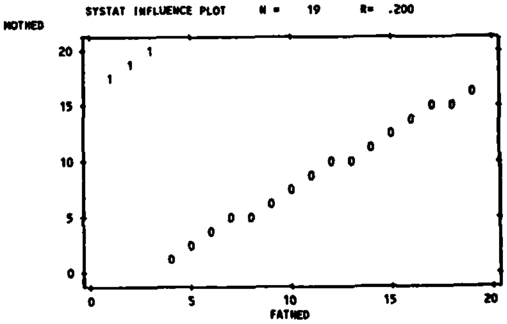
A. Teitel requests a tabulation of cars owned by number of persons over 16. We fit a log-linear model to this table and found a highly insignificant chi-square for an interaction hypothesis. As anyone knows, households with teenagers have significantly more cars (including those in the body shop). The following plot of standardized residuals to the additive model shows no conspicuous deviation from normality.



B. Teitel requests a table of age of trip-taking persons by the model year of their cars. It is common knowledge that model years comprise a simplex data structure because people buy cars as soon as they see a new one in their neighbors' driveway. To test this, we computed gamma coefficients on the <u>columns</u> of the table (model year) and did a multidimensional scaling on the matrix of resulting coefficients. The following plot shows the model years in alphabetical order. This plot resembles a random walk more than a one dimensional simplex.



C. Teitel requests a table of mother's versus father's education. We reduced this table to a single count per cell and performed an influence scatterplot. As can be seen plainly in the upper left hand corner, three of the cells reduce the correlation by at least .10. These consist of highly educated women married to dolts. This violates a law of nature.



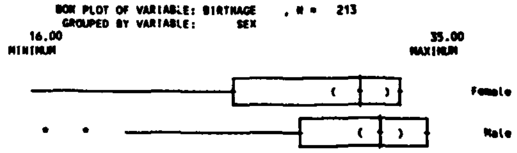
D. Finally, Teitel requests a table of mother's age by sex of last offspring. Here is the table.

TABLE OF BIRTHAGES (ROWS) BY SEXS (COLUMNS)

FREQUENCIES

	Female	Male	TOTAL
0-18	4	2	6
19-25	14	19	33
26-30	25	17	42
31-35	47	85	132
TOTAL	90	123	213

We did a grouped box plot to check the distributions of mothers' age at the birth of their last child. As anyone knows, last born females should be associated with higher maternal ages because families continue to reproduce until they have a woman who can grow up and become a statistician.



Conclusion

Several mainframe and microcomputer database packages offer the tools to solve the volume testing problem in this symposium. In fact, the database packages which include crosstabulation can probably solve them faster and with fewer commands than any of the statistics packages surveyed here.

A good statistics package should do more, however. It should be able to integrate database information with all of its statistical procedures without using complex commands. Tables are the beginning of statistical analysis, not the end.

Because of this simple distinction, we approached this database problem differently from the other vendors in this symposium. First, although SYSTAT can process character data, we treated the data as double precision numbers instead of one or two byte characters. As a consequence, our files are larger and required more time to convert from the raw data. Notice, however, that our statistical procedures read and computed tables on these numerical files as fast as the others computed tables from character files.

Second, to uncover the database fraud, we needed to produce more than tables. Because of the way people process multivariate information, tables are one of the worst methods for displaying complex multivariate relationships. We chose, instead, a few of SYSTAT's graphical displays which reveal at a glance the artificiality of the data. Some of these graphics resemble ones in the other programs, but the appearance is deceiving. Only SYSTAT offers multidimensional scaling, generalized gamma coefficients, influence plots and grouped box plots. Furthermore, since SYSTAT can save tables into files and treat them as ordinary data, accessing these other procedures requires only one command.

The following Figures 1 through 4 provide the SYSTAT code for producing the required tables. We could have speeded up processing by using programming tricks, but the code in these figures is more typical of the average SYSTAT programmer's approach to the problems. We believe the human processing time is as important as the computer processing time.

Figure 1
Read Input Datasets

```
DATA
NOTE 'Read HHOLDS records.'
SAVE HHOLDS / 'Household dataset' 'ID variable:  HHOLD'
GET HHOLDS
LRECL = 120
INPUT (HHOLD RECTYPE1 ONE1(1-51) TWO1(1-20) FOUR1(1-5)),
    (#4   #1       ^10 51*#1  20*#2      5*#4     )
RUN

NOTE 'Read CARS records.'
SAVE CARS / 'Cars Dataset' 'ID variable:  HHOLD CARS'
GET CARS
LRECL = 120
INPUT (HHOLD RECTYPE2 CAR ONE2(1-51) TWO2(1-20) FOUR2(1-4) MYEAR),
    (#4   #1      #2  ^10 51*#1  20*#2      4*#4      #4  )
RUN

NOTE 'Read PERSONS records.'
SAVE PERSONS / 'Persons Dataset' 'ID variable:  HHOLD PERSON'
GET PERSONS
LRECL = 120
INPUT (HHOLD RECTYPE3 PERSON ONE3(1-51) AGE TWO3(1-19) FOUR3(1-5)),
    (#4   #1       #2     ^10 51*#1  #2  19*#2      5*#4      )
RUN

NOTE 'Read TRIPS records.'
SAVE TRIPS / 'Trips Dataset' 'ID variable: HHOLD PERSON TRIP'
GET TRIPS
LRECL = 120
INPUT (HHOLD RECTYPE4 PERSON TRIP CAR ONE4(1-50) TWO4(1-19) NDAYS FOUR4(1-5)).
    (#4   #1       #2     #2  #1  50*#1       19*#2      #2    5*#4      )
RUN

NOTE 'Read PEOPLE records.'
SAVE PEOPLE / 'Sorted by PERSON'
GET PEOPLE
LRECL = 120
INPUT (PERSON SEX ONE(1-50) EDUC TWO(1-14) BIRTHYR FOUR(1-5) MOTHER FATHER).
    (#5    #1  50*#1     #2   14*#2     #4      5*#4      #5     #5   )
RUN
QUIT
```

Figure 2
The TRIPS Data Collection
Producing Table 1

```
DATA
SAVE TEMP
  USE HHOLDS (HHOLD) CARS (RECTYPE2 HHOLD) / HHOLD
  BY HHOLD
  HOLD

NOTE 'Count number of cars.'
NOTE 'Only increment cars count if household exists in CARS dataset.'
  IF BOG THEN LET NCARS = 0
  IF RECTYPE2 <> . THEN LET NCARS = NCARS + 1
  IF EOG = 0 THEN DELETE
RUN

NEW
SAVE PROB1P1
  USE TEMP PERSONS (RECTYPE3 AGE HHOLD) / HHOLD
  BY HHOLD
  HOLD

NOTE 'Count number of persons over age 16.'
  IF BOG THEN LET NPERSONS = 0
  IF RECTYPE3 <> . AND AGE > 16 THEN LET NPERSONS = NPERSONS + 1
  IF EOG = 0 THEN DELETE
RUN
QUIT

TABLES
USE PROB1P1
TABULATE NCARS * NPERSONS
QUIT
```

Figure 3
The TRIPS Data Collection
Producing Table 2

```
DATA
SAVE TEMP
  USE PERSONS (HHOLD PERSON AGE RECTYPE3) TRIPS (HHOLD PERSON CAR NDAYS) ,
        / HHOLD PERSON
  IF NDAYS < 3 OR RECTYPE3 = . OR CAR = 0 THEN DELETE
  SORT HHOLD CAR
RUN

SAVE PROB1P2
  USE TEMP CARS (HHOLD CAR RECTYPE2 MYEAR) / HHOLD CAR
  IF RECTYPE2 = . OR RECTYPE3 = . THEN DELETE
RUN
QUIT

TABLES
USE PROB1P2
TABULATE AGE * MYEAR
QUIT
```

Figure 4
The PERSONS Data Collection
Producing Table 3

```
DATA
NOTE 'Dataset of mothers sorted by MOTHER.'
NOTE 'It is crucial that FATHER be included in this dataset.'
SAVE MOTHER / 'Sorted by MOTHER'
  USE PEOPLE (MOTHER FATHER)
  NOTE 'Delete cases with no mother.'
    IF MOTHER = 0 THEN DELETE
  LET INCLUDE$ = 'YES'
  SORT MOTHER
RUN

NOTE 'Dataset with mothers'
SAVE PEOPLE2
  USE PEOPLE (PERSON EDUC)
  NOTE 'Rename PERSON to MOTHER and EDUC to MOTHED'
    LET MOTHER = PERSON
    LET MOTHED = EDUC
  DROP PERSON EDUC
RUN

NOTE 'Dataset of mothers education joined to offspring record.'
NOTE 'Sorted by FATHER.'
SAVE MOTHER2
  USE MOTHER PEOPLE2 / MOTHER
  IF INCLUDE$ <> 'YES' THEN DELETE
  SORT FATHER
RUN

NOTE 'Dataset with fathers.'
SAVE PEOPLE3
  USE PEOPLE (PERSON EDUC)
  NOTE 'Rename PERSON to FATHER and EDUC to FATHED'
    LET FATHER = PERSON
    LET FATHED = EDUC
  DROP PERSON EDUC
RUN

NOTE 'Add fathers education to dataset.'
NOTE 'Dataset contains both fathers and mothers education, sorted'
NOTE '   by FATHER.'
SAVE PROB2P1
  USE MOTHER2 PEOPLE3 / FATHER
  IF INCLUDE$ <> 'YES' THEN DELETE
  DROP INCLUDE$
RUN
QUIT

TABLES
USE PROB2P1
TABULATE MOTHED * FATHED
QUIT
```

Figure 5
The PERSONS Data Collection
Producing Table 4

```
DATA
SAVE MOTHER
  USE PEOPLE (BIRTHYR PERSON)
  LET MOTHER = PERSON
  LET MBIRTHYR = BIRTHYR
  DROP BIRTHYR PERSON
RUN

SAVE MOTHER2
  USE PEOPLE (MOTHER BIRTHYR SEX)
  SORT MOTHER
RUN

SAVE MOTHER3
  USE MOTHER2 MOTHER / MOTHER
  IF MBIRTHYR = . OR BIRTHYR = . THEN DELETE
  LET BIRTHAGE = BIRTHYR - MBIRTHYR
  SORT MOTHER BIRTHAGE
RUN

SAVE PROB2P2
  USE MOTHER3
  BY MOTHER
  IF EOG = 0 THEN DELETE
  LABEL SEX / 1 = 'Female' 2 = 'Male'
  IF BIRTHAGE<19 THEN LET BIRTHAGE$='0-18'
  IF BIRTHAGE>18 AND BIRTHAGE<26 THEN LET BIRTHAGE$='19-25'
  IF BIRTHAGE>25 AND BIRTHAGE<31 THEN LET BIRTHAGE$='26-30'
  IF BIRTHAGE>30 AND BIRTHAGE<36 THEN LET BIRTHAGE$='31-35'
RUN

TABLES
USE PROB2P2
TABULATE BIRTHAGE$ * SEX$
QUIT
```

STATISTICAL DATABASE MANAGEMENT ON MICROCOMPUTERS: THE BENCHMARK RESULTS

Robert F. Teitel

TEITEL DATA SYSTEMS
Bethesda, MD 20814

This paper presents the results of executing the solutions to a set of data manipulation problems supplied by vendors of microcomputer-based statistical systems on a common microcomputer. The description of the data files and the data manipulation problems are found in a companion paper, " ...: The Benchmark Problems", elsewhere in these proceedings. The description of each vendor's solutions to the benchmark problems are also found elsewhere in these proceedings.

PARTICIPATING VENDORS

BMDP	BMDP Software; 1964 Westwood Drive; Los Angeles, CA 90025
DASY	Statistical Software Resources; 20355 Seaboard Avenue; Malibu, CA 90265
PRODAS	Conceptual Software; P.O.Box 56627; Houston, TX 77265
PSTAT	PSTAT Inc.; 471 Wall Street; Research Park; Princeton, NJ 08540
SAS/PC	SAS Institute; P.O.Box 8000; Cary, NC 27511
SPSS/PC	SPSS Inc.; 444 North Michigan Avenue; Chicago, IL 60611
SYSTAT	SYSTAT Inc.; 2902 Central Street; Evanston, IL 60202

I. INTRODUCTION

The performance figures to be presented below -- for database creation time and database size, and execution of the benchmark data manipulation problems -- are based on the batch-oriented job streams, or "scripts", submitted by the vendors. Each script was executed just once, and performance was monitored with a stopwatch. The indicated total time should be correct to within 5 or 10 seconds. However, it has been our experience in other microcomputer timings that were these benchmarks to be rerun -- as we intend to do -- the numbers could vary by as much as 15 or 20%. Much of the variation will be due to the varying distribution of the data and program files involved in each test on the (usually) single large ("hard", "fixed") disk drive on most microcomputer systems.

The equipment used for the performance tests consisted of an IBM PC/XT with an 8087 Numeric Data CoProcessor (NDP) and a 30 MB disk of unknown brand with about 10MB of free space for our use.

After studying the various scripts, published elsewhere in these proceedings, and the timing figures below, I submit the following conclusion can be drawn: the variation among the vendors is negligible compared to the time it would take a typical user to prepare the job streams.

II. SCRIPT DISSIMILARITIES

There are a number of known dissimilarities among the scripts from each vendor which would impact performance. These include the following. The SYSTAT database sizes could have been reduced to half that shown, with apparently little affect on performance. The PSTAT scripts for the creation of the two databases assumed the input data files would come from a floppy disk; all other vendors assumed that the data files were on the large disk. The BMDP scripts do not save the created databases; thus no database sizes are presented.

The PSTAT timings include printing of the final tabulations because those scripts were the only ones to print just the final tabulations, all the other scripts required either that everything or nothing be printed. Since the amount of printing might have materially affected the execution timings, the non-PSTAT scripts were executed without printing anything.

The PRODAS scripts for the second (PEOPLE) set of problems assumed space existed for an in-core list of referenced record keys (as there was for the data distributed). The BMDP scripts for these problems explicitly sorted the referenced record keys, whereas "parents" (for the first tabulation) and "mothers" (for the second tabulation) always occurred before the applicable child record in the distributed data, a fact which could be used to eliminate the sorts.

The DASY scripts were not presented at the benchmarking session. Moreover, DASY doesn't have a concept of a "systems file" or a "database": it selects from input files only those data fields necessary for a particular task. PRODAS submitted a similar set of scripts, in addition to those which used a database constructed from the full set of distributed data. These two "speedy" results are presented together below, and represent likely similar results for the other systems executing in this fashion.

Finally, the vendors selected one of two basic script organizations. Since there are two tabulations requested from each database, some scripts perform some common procedures for the tabulations (even if only creating a subset database containing only the variables necessary for the two tabulations). Other scripts assume that the tabulations are totally independent, and proceed to produce them separately. Clearly, performance results are not comparable across such scripts, and are therefore presented separately below. (Note that the vendors were not necessarily consistent, using one script organization for the TRIPS data tabulations, the other for the PEOPLE data tabulations.)

Other dissimilarities will, no doubt, be discovered. In addition, expert users of these systems might find clever tricks to improve the performance. The scripts were executed as provided by the vendors, and represent (we hope) scripts which a typical users might have constructed.

III. RESULTS: MANIPULATING THE TRIPS DATA COLLECTION

The following table displays the number of records, raw ASCII size, database size by component, and time to load the database by vendor for the TRIPS data collection.

	HHOLDS	CARS	PERSONS	TRIPS	database load time
					mins:secs
records:	105	210	315	945	
raw size:	13k	26k	39k	115k	
BMDP	--	--	--	--	12:45
PRODAS	18k	31k	47k	124k	5:30
PSTAT	16k	26k	38k	105k	13:00 (floppy)
SAS/PC	21k	36k	49k	138k	15:30
SPSS/PC	16k	30k	44k	129k	15:00
SYSTAT	68k	136k	204k	616k	39:30

The following table displays the processing time required for each TRIPS data collection script by script organization by vendor.

```
============================================================================
!!                                                                        !!
!!    +----------------------------+       +----------------------------+ !!
!!    ! hholds/cars/persons/trips !        ! hholds/cars/persons/trips ! !!
!!    +----------------------------+       +----------------------------+ !!
!!                  !                            !                 !      !!
!!           +----------+                  +----------+      +----------+ !!
!!           ! process !                   ! process !       ! process !  !!
!!           +----------+                  +----------+      +----------+ !!
!!              /     \                          !                 !      !!
!!    +----------+  +---------+            +----------+      +----------+ !!
!!    ! xtab 1  !   ! xtab 2 !             ! xtab 1  !       ! xtab 2  !  !!
!!    +----------+  +---------+            +----------+      +----------+ !!
!!                                                                        !!
!!          Execution time                        Execution time          !!
!!            mins:secs            total     mins:secs         mins:secs  !!
!!          --------------         ---------------------------------------!!
!!    BMDP                         20:00+       6:15             13:45    !!
!!    PRODAS                        6:00+       1:30              4:30    !!
!!    PSTAT       13:30                                                   !!
!!    SAS/PC       8:45+                                                  !!
!!    SPSS/PC                      18:30+       5:45             12:45    !!
!!    SYSTAT                       10:30+       2:45              7:45    !!
!!                                                                        !!
!!    (Only PSTAT provided table only printing; The others were run with  !!
!!     display on screen only (that is, without any printing).)           !!
!!                                                                        !!
!!    PRODAS                        5:00+       1:30              3:30    !!
!!    DASY                          4:30+       1:15              3:15    !!
!!                                                                        !!
!!    (The above two results are for loading only the variables needed in!!
!!     order to perform the requested tabulations.)                       !!
============================================================================
```

IV. RESULTS: MANIPULATING THE PEOPLE DATA COLLECTION

The following table displays the number of records, raw ASCII size, database size, and time to load the database by vendor for the PEOPLE data collection.

	PEOPLE DATA COLLECTION	
Input records:	784	database load time
Approximate size:	96k	mins:secs
BMDP	--	7:00
PRODAS	114k	2:30
PSTAT	97k	5:00 (floppy)
SAS/PC	113k	6:45
SPSS/PC	122k	7:00
SYSTAT	479k	27:30

The following table displays the processing time required for each PEOPLE data collection script by script organisation by vendor.

```
==============================================================================
!!                                                                          !!
!!   +----------------------------+        +----------------------------+   !!
!!   :    people database         :        :       people database      !   !!
!!   +----------------------------+        +----------------------------+   !!
!!               :                              :                 !         !!
!!          +---------+                    +---------+       +---------+    !!
!!          : process :                    : process !       ! process !    !!
!!          +---------+                    +---------+       +---------+    !!
!!           /       \                          !                 !         !!
!!   +---------+   +---------+             +---------+       +---------+    !!
!!   ! xtab 3  :   ! xtab 4  !             : xtab 3  !       ! xtab 4  !    !!
!!   +---------+   +---------+             +---------+       +---------+    !!
!!                                                                          !!
!!         Execution time                         Execution time            !!
!!           mins:secs            total     mins:secs         mins:secs     !!
!!          ----------------      --------------------------------------    !!
!!   BMDP      15:30+                                                       !!
!!   PRODAS     3:45+                                                       !!
!!   PSTAT     10:30                                                        !!
!!   SAS/PC     8:15+                                                       !!
!!   SPSS/PC                      16:30+       6:30             10:00       !!
!!   SYSTAT                       16:00+       6:00             10:00       !!
!!                                                                          !!
!!   (Only PSTAT provided table only printing; The others were run with    !!
!!    display on screen only (that is, without any printing).)              !!
!!                                                                          !!
!!   PRODAS     3:15+                                                       !!
!!   DASY                          5:45+       3:15               2:30      !!
!!                                                                          !!
!!   (The above two results are for loading only the variables needed in   !!
!!    order to perform the requested tabulations.)                          !!
==============================================================================
```

V. SUMMARY

If one is given the data collection descriptions and the tabulation problem definitions, published elsewhere in these proceedings, and one studies the various scripts developed by the vendors of microcomputer statistical software to perform the necessary data manipulation for the tabulations, also published elsewhere in these proceedings, and the timing figures presented above, I submit the following conclusion can be drawn: the variation among the execution times of the scripts from the various vendors is negligible compared to the time it would take a typical user to prepare the job streams.

VI. ACKNOWLEDGEMENTS

Clearly this work could not have been done without the cooperation of the vendors involved, and their efforts are sincerely appreciated. In addition, the author wishes to express his deep gratitude to Thomas Boardman and and Jim zumBrunner and their staff at the Statistics Laboratory of Colorado State University for providing the equipment on which the benchmarks were executed, and on which drafts of the " ... PROBLEMS" and the "... RESULTS" paper were prepared for distribution at the Interface Symposium.

VII. REFERENCES

See the companion paper, ". . . : The Benchmark Problems," elsewhere in these proceedings.

See also the accompanying papers produced by the participating vendors on their solutions to the benchmark problems elsewhere in these proceedings.

FUTURE DIRECTIONS FOR STATISTICAL SOFTWARE

Organizer: *David M. Allen, University of Kentucky*

Statistical Language for Microcomputers
David M. Allen, University of Kentucky

Matrix Languages for Statistics
Kenneth N. Berk, Illinois State University

Publishing Statistical Software
John C. Nash, University of Ottawa

A STATISTICAL LANGUAGE FOR MICROCOMPUTERS

David M. Allen, University of Kentucky

1. INTRODUCTION

A programming language is a systematic notation by which one instructs the computer to do a task. A statistical language is a programming language which makes it easier to instruct a computer to do a statistical analysis. It would have predefined data types, default values for certain variables, and understood objects in certain expressions. It would have built-in procedures for application of the most frequently used statistical software.

There is presently a large amount of statistical software available. This goes by a variety of names including programs, packages, systems, and languages. Very few would satisfy a formal definition of a language, and all are well short of ideal. There are many considerations in the design of a statistical language, and many tradeoffs which must be examined. The language envisioned here can handle all aspects of data analysis that involve the use of the computer: data entry and management, editing, application of statistical techniques, and report generation.

Once the language design is specified it is necessary to write a translator or compiler that will change statements written in the language into object code (machine instructions). The efficiency of the data analysis depends on the language design, the quality of the translator, and the quality of object code produced by the translator. Design criteria for a good statistical language are very similar to the design criteria for a good general purpose computer language. Horowitz (1984) gives the following list of criteria for the design of general purpose computer languages: reliability, fast translation, extensibility, well defined syntatic and semantic description, efficient object code, orthogonality, machine independence, provability, generality, consistency with commonly used notations, subsets, and uniformity. Many of the items in this list are interrelated. The next three sections concentrate on the first three items in this list, but several others will be discussed in passing. A subsequent section will discuss relevant considerations in developing a language for a microcomputer as opposed to a large mainframe computer. The paper concludes with a summary of desired features in a statistical language. Implementation of the features would represent a considerable advancement from existing statistical software.

2. RELIABILITY

In both general purpose computer languages and statistical languages, a sequence of statements designed to instruct a computer to carry out a task is called a program. Programs written in a language should be reliable. A number of factors contribute to reliability. A statistical language should be similar to one's native language. Freedom to place comments within the code also enhances readability and reliability of programs.

Some programming languages require that variables be declared before use and others do not. Declaration of variables makes programs more reliable and efficient. The FORTRAN statements

```
      .
      .
      .
      HORSE = A+B
      .
      .
      .
      HOUSE = C+HORSE
      .
      .
      .
```

are perfectly legal. However, if HOUSE was intended to be HORSE, a programmer may have trouble finding why the program is not working as it should. The Pascal programmer is required to put a statement like

```
      VAR HORSE:INTEGER;
```

near the beginning of his program. Any occurance of HOUSE is immediately detected by the compiler.

The SAS statistical system tries to find out as much as it can about data and data structures to relieve the programmer of the burden of providing declarations. One convention is for a character variable to take its length attribute from its first use. Consider the SAS job

```
      DATA DUMMY;
      INPUT X;
      IF X>3 THEN Y='BIG';
      ELSE Y='SMALL';
      CARDS;
      5
      4
      3
      2
      1
      PROC PRINT;
```

The output will have "SMA" where "SMALL" was intended. Of course we can rearrange things to get what we want. However, it seems easier to always declare variables than to be always mindful of quirks in a language.

The following example illustrates how the combination of an additional data type and declaration of variables could improve reliability, ease of use, and execution speed. Consider the SAS job

```
      DATA STUDENT;
      INPUT CLASS $ Y;
      CARDS;
      FRESH 2.4
      FRESH 4.0
      .
      .
      .
      SOPH 3.2
      SOPH 2.3
      .
      .
      .
      GRAD 2.5
      CRAD 3.2
      PROC GLM;
      CLASSES CLASS;
      MODEL Y=CLASS;
      ESTIMATE CLASS 1 -1 0 0 0;
```

Suppose the number of data lines is large enough that the cost of running the job is not trivial and CRAD was intended to be GRAD. SAS would not detect the error and the results would not be usable. The ESTIMATE statement would compare a single graduate student to the average of the freshmen. SAS does not have a qualitative data type, but determines the information it needs for each application with the CLASSES statement. The CLASSES statement requires two passes through the data: one to determine what levels are present, and another to assign the integer codes.

Suppose we introduce a data type like Pascal's enumerated data type except that the string representation can be input and output directly. Near the beginning of the program would be the statements

```
     TYPE CLASSTYPE=(FRESH,SOPH,JUNIOR,SENIOR,GRAD);
     VAR CLASS:CLASSTYPE;
```

The character variable indicator and the CLASSES statement would be removed. On input, any value that did not match a value in CLASSTYPE would be detected as an error. The integer representations of the levels would be determined on input and made a permanent part of the data set. The data set would require less disk storage. The large time requirement of the CLASSES statement is eliminated. The coefficients in the ESTIMATE statement would now apply to the natural order as specified in the TYPE statement, rather than the obscure order associated with the character set's collating sequence.

The language should be such that typing errors are likely to be detected by the translator. Consider the FORTRAN statements

```
     DO 10 I = 1.5
     A(I) = X + B(I)
     10 CONTINUE
```

The 1.5 was intended to be 1,5. However, FORTRAN does not recognize blanks, and the statement intended to be a DO statement is actually the assignment statement DO10I = 1.5. Also, since FORTRAN does not require that variables be declared, the compiler did not detect an error. This example is from Horowitz (1984).

3. FAST TRANSLATION

Translation of a program to object code is part of every analysis. Generally, the speed of translation is very important. We will discuss two of a large number of factors that affect speed of translation.

One factor is the syntax and semantics of the language. Semantics deals with the meanings of sentences. Syntax refers to the rules by which words are put together to form phrases, clauses, and sentences. The syntax should be such that backing up is minimized. Consider the FORTRAN statement

```
     IF (5.EQ.MAX) GOTO 100
```

In FORTRAN, 5.E-2 is a legal number, hence the compiler reaches the "Q" before it knows the proper interpretation of the string of characters starting with the "5".

In FORTRAN, the expression DOG(I) could be either a function or an element of an array. In Pascal, DOG(I) is a function and DOG[I] is an element of an array. Pascal's way of doing things involves less table searching and compiles more quickly. In addition, programs are more reliable and easier for the human readers.

Some compilers make multiple passes over the program. This allows the programmer a certain amount of freedom from attention to detail, but requires more time to compile. Pascal programmers must place certain things in a specified order, but their programs can be compiled in a single pass with a one character look ahead.

Another factor affecting translation speed is the language in which the translator is written. Traditionally, compilers and translators were written in assembly language to make the best use of the higher speed portions of the computer and to reduce redundant instructions. Recently, the C programming language has become popular for systems work and compiler writing. The use of C gives a high degree of machine independence. Computer scientists are currently debating the relative efficiency of the object code generated by assembly language and C (Kernighan and Ritchie, 1978). The editor of DTACK GROUNDED (1985) cites examples of drastic performance reductions of recent versions of commercial software written in C. These observations regard the currently popular class of microcomputers. Some study and experimentation needs to be done to resolve this issue.

4. EXTENSIBILITY

Extensibility is the ability to define additional operators and objects in the language. The language developer can not include all known statistical techniques as operators in the language. The user should have the ability to add to the language the additional techniques he needs. When making extensibility part of the language, the developer must make some assumptions about the programming capabilities of the user. Generally speaking, a user wishing to make worthwhile extensions should be able to program in a higher level language like FORTRAN or Pascal. He should not, however, be expected to know a lower level language like assembly or C. This would severely limit the number of people able to write extensions.

The statistical systems SAS (SAS Institute, 1985) and S (Becker and Chambers, 1974) provide forms of extensibility. Both have a well defined grammar for an interface language. However, the technical knowledge required of the individual implementing the extensions is more than we feel necessary. The Forth programming language is extended by entering the extensions and typing FREEZE. Turbo Pascal and UCSD Pascal allow for automatic overlays. These software products demonstrate that extensibility need not be difficult for the user.

5. SPECIAL CONSIDERATIONS FOR MICROCOMPUTERS

Currently, the most popular class of microcomputers has an Intel 8088 processor, the MS-DOS or similar operating system, about 256 kilobytes of random access memory, approximately 720 kilobytes disk storage, and some graphics capability. The statistical language proposed here could certainly be implemented on this class of computer. These computers are slow, and thus a good batch facility is needed to provide for unattended operation. A procedure to check the syntax of programs and for the presence of requested data sets and devices would remove most of the causes of unsuccessful runs. An overlaying capability is needed to deal with the limited amount of random access memory. In situations where disk space is limited, the user should be able to remove procedures he never uses.

While the language could be designed to operate on the computers described above, it seems more reasonable to design for the popular computer of three years from now. Today's leading edge microcomputers like the IBM PC/AT or AT&T Unix PC would seem to be logical choices as target machines.

The capabilities of microcomputers and mainframe computers are both increasing rapidly with time. There will be a point when a microcomputer programmer will not have to design around speed and memory limitations. Operating systems will provide more support to make language implementation easier. However, it is likely that a large proportion of microcomputer users will continue to be isolated from the technical support received by many mainframe users. Hence, the language must be well documented and easy to use.

6. FEATURES OF THE ENVISIONED STATISTICAL LANGUAGE

In view of the previous discussion, a number of features are clearly desirable in a statistical language. These features are restated here in summary form.

The statistical language would, to the extent possible, satisfy the stated criteria for language design. Note, however, that a gain in machine independence is likely to result in a decrease in efficiency of object code. Other compromises may also be necessary.

The syntax of the language would be very similar to that of Pascal. Pascal compiles rapidly in one pass and is relatively free of quirks. Pascal is presently taught in high schools. Hundreds of thousands of copies of a Pascal compiler for microcomputers have been sold. Thus, many people already know Pascal and could learn the statistical language with little effort.

Data types of all variables must be declared. This will make programs more reliable and faster in translation. The data type of variables in data sets would be stored in the data set and hence repetitive declarations would not be required. A data type similar to Pascal's enumerated data type would be present. This would make analysis of data with qualitative or categorical variables more reliable and more efficient.

Some of the ways in which the proposed language would differ from Pascal follow. Input/output facilities for enumeration data types would be implemented. The file handling capabilities would be extended to allow indexed files. Techniques for parameter passing to procedures would be modified along the lines of Ada or S to allow for default values and more ease of use.

Extending the language would be as easy as writing a Pascal-type program and telling the language to attach the program to itself. Nearly all of the procedures and functions in the language would be available to the person writing the extension.

The language interacts with the user on at least two levels. The extension programming level would be similar to the Turbo Pascal environment where one can go quickly back and forth between the editor and compiler. The statistical analysis level would have to be command driven to be consistent with the batch capability. However, an interactive shell would be available to assist beginning users.

7. BIBLIOGRAPHY

Becker, R.A. and Chambers, J.M. (1984). S An Interactive Environment for Data Analysis and Graphics. Wadsworth, Inc.: Belmont CA.

DTACK grounded (1985). The Journal of Simple 68000/32081 Systems 42, 6-7.

Horowitz, E. (1984). Fundamentals of Programming Languages. Computer Sciences Press, Inc.: Rockville MD.

Jensen, K. and Wirth, N. (1974). PASCAL User Manual and Report, 2nd Edition. Springer-Verlag: New York.

Kernighan, B.W. and Ritchie, D.M. (1978). The C Programming Language. Prentice-Hall, Inc.: Englewood Cliffs NJ.

SAS User's Guide (1985). SAS Institute, Inc.: Raleigh NC.

Turbo Pascal Version 3.0 Reference Manual (1983). Borland International, Inc.: Scotts Valley CA.

Wirth, N. (1971). "The Design of a Pascal Compiler," Software-Practice and Experience, 1, No. 4, 309-33.

MATRIX LANGUAGES FOR STATISTICS

Kenneth N. Berk, Illinois State University

1. Introduction

Packaged statistical programs are useful, but there is much that they do not do. Languages such as FORTRAN, PASCAL, and C can be used as an alternative, but much effort is required to use them. It is easier to program in a language with facilities for manipulation of matrices and building blocks for statistics such as probability distributions, random number generation, plotting, ranking, and functions for linear model analysis.

This paper discusses seven matrix languages - APL combined with STATGRAPHICS, GAUSS, SAS IML and MATRIX, MATLAB, S, and SPEAKEASY - which are listed along with the vendor addresses in Figure 1. Tables 1 through 12 indicate the presence of various features in these languages. Section 2 explains these features and Sections 3 through 8 discuss the languages individually. Finally the last section gives a summary along with a discussion of some desirable features for future implementation.

Note that the author has not used APL, IML, or S. Their importance is such that it was thought best to include them, based on documentation and other sources. The author is indebted to W. Gerald Platt of San Francisco State for the APL benchmarks and Alan Eaton of SAS Institute for the IML benchmarks.

2. The Tables

Perhaps Table 1 is self-explanatory, except that "PROTECTED" refers to copy protection. In Table 2, it is indicated that all of the languages are interpreted. Here, "DIARY FILE" refers to the ability to store the results of an interactive session for later editing and, possibly, execution as a batch file. "PRINT TOGGLE" is the ability to turn on and off the automatic output of all assignment statements. This is a valuable aid in debugging.

In Table 3 "VECTOR" refers to the ability to reference one-dimensional structures by one subscript. Those without this feature require two subscripts.

Each of the languages have some sort of subroutine or macro structure, as indicated in the last column of Table 5. Here, "LOCAL" refers to the variables being local to the structure. "COMPILE" means that subroutines can be precompiled.

In Table 7 an "INDEX VECTOR" consists of the integers from 1 to N. The "SUBMATRIX" extracted consists of the first three rows.

In Table 8 the abbreviations stand for Choleski, eigenvalues and eigenvectors, singular value decomposition, generalized inverse, QR decomposition, Gram-Schmidt decomposition, and the fast Fourier transform.

The reason for having the "KRONECKER PRODUCT" in table 9 is that it can be used for creating interaction terms in the analysis of variance. The entries for STATGRAPHICS and SAS here in parentheses show alternatives which are easier to use for the same purpose.

The "VERSATILE" entry in Table 11 is intended to indicate powerful features in the given category. "CG" refers to the color graphics card.

The benchmarks in Table 12 were obtained with the 8088/8087 (IBM PC and clones), except that for IML they were obtained with the 80286/80287 (IBM AT).

3. APL / STATGRAPHICS

Created by Kenneth Iverson in the late 50's, APL is the oldest of the languages considered here. It has its own character set and to outsiders the programs may be difficult to understand. This is in contrast to the other matrix languages, all of which resemble ordinary matrix algebra. Nevertheless, APL has a strong following, and the APL conferences are well-attended. It seems to be the only matrix language available on the Macintosh (but MATLAB will be available soon) and there are a number of versions available for PC-DOS / MS-DOS machines, including four reviewed by Smith (1985). Unfortunately, APL by itself does not include eigenanalysis, tail probabilities, quantile plots, etc., so that it is advantageous to obtain such features additionally. One way to do this is with the STATGRAPHICS package. The tables in this paper refer to the combination of STATGRAPHICS and APL marketed by STSC, Inc.

Although STATGRAPHICS is a menu-driven statistical package, all of the APL source is available as building blocks for the APL programmer. On the other hand, it requires some effort to access the APL source code. Platt and Platt (1985) describe the combination of APL and STATGRAPHICS as being very powerful in its capabilities. Tables 1-12 verify that there are a wide range of features, especially in graphics and distributions. If one does not care for the APL character set, it is possible to use instead a word-oriented syntax.

MS-DOS commands can be issued from within the system, and outside programs can be run while preserving the current session.

There are two major criticisms of APL. First, note in Table 5 that there is very little provision for structured programming. Of course, the language was specified more than 25 years ago, when less emphasis was given to structuring code in blocks for easy readability and easy debugging. In APL a "do loop" must be built by setting and incrementing a counter and using the APL equivalent of the "go to" command. There are no "if" statements, so the conditional exit from the loop must involve a computation such as a logical function. It should be emphasized that there are many APL enthusiasts who do not consider the language deficient in programming structures.

The second criticism of APL is in the area of computational efficiency. As is indicated in Table 12, there are programs which may take as much as ten times as long in APL as compared to other languages. For some purposes this may not be a serious drawback, but for a Monte Carlo study it could make a significant difference in run time. A call to STSC did not produce any encouragement about speed fixes on the horizon. The suggestion from STSC is to find the bottleneck and replace it with assembler or C code, which can be linked with APL. This philosophy has been implemented in STATGRAPHICS, where the inversion operator has been rewritten in C.

4. GAUSS

GAUSS is only a few years old. It runs only on MS-DOS computers with the math coprocessor chip. At $250 it is the least expensive of the seven languages compared here. Also, its hardware demands are relatively simple in that it requires only 256K of memory and no hard disk.

GAUSS has a simple full screen editor with no copy or search facilities. Users may as an alternative use their own editors. Programs can be run interactively from the GAUSS editor. In the interactive mode one has access to what has just been run, and the commands can be edited rather than retyped. A screen full of commands from an interactive session can be saved, edited, and run as a batch job.

There is a looseleaf manual dated 1984 which describes the package as of that time. There are, however, a fair number of features which have been added since then. They are documented on the disks, but the quality of the explanations leaves room for improvement. There are also two disks of programs written in GAUSS, and they too are documented only on the disks. It would be nice to have written documentation of good quality for the whole system. Online help would also add to the ease of use. As it is, the looseleaf documentation is barely adequate, with much left for the user to figure out, and the disk documentation is of lower quality. The latest news from the authors is that new written and online documentation is on the way.

As in SAS MATRIX, all data are in the form of matrices with two subscripts. Character data are allowed and GAUSS permits mixing of character and numerical data in arrays, but the user must tell GAUSS which is which for printing. There is a weakness in the the area of input files. GAUSS has a program to convert ASCII files to its own format, but it is rather limited in that the data items must be separated by spaces in the ASCII file.

GAUSS interacts well with DOS. The program shares with others the ability to interrupt a session, execute DOS commands and run programs of any kind, and then return. A newly added feature includes links to the common FORTRAN compilers.

GAUSS does not have "DO" loops, but it does have a "DO...WHILE" structure which requires setting and incrementing a counter. There is an "IF...ELSEIF...ELSE...END" structure. The 1984 GAUSS had subroutines with nonlocal parameters in the style of BASIC. Recently, compilable subroutines with local parameters have been added, but they are not very well documented.

GAUSS shares with MATLAB the use of a "." before an operator to indicate an elementwise operation. For example, "A.*B" yields elementwise multiplication of A and B. A sequence of consecutive integers is easily specified in GAUSS, IML, MATRIX, and MATLAB using the ":" notation. For example, "2:5" specifies the integers from 2 to 5, inclusive. In GAUSS this is allowed only in subscripts, whereas the others allow assignment statements of the form "I=2:5".

GAUSS has available a good set of tail probabilities, including normal, t, chi-square, and F. There are no inverse probability functions on the main disk, but one of the two other disks has a GAUSS program that obtains the t inverse by the Newton-Raphson method.

Graphics was not a high priority in the 1984 edition of GAUSS. Only simple non-hires scatter plots with no axes are described in the manual. There are, however, some new graphics functions on the two supplementary disks, and more functions are on the way, using the color graphics adapter.

The advertising copy for GAUSS brags about its speed, and benchmarks verify that the program is fast. By the use of the Crout algorithm for inversion and the use of assembler to make sure that the math coprocessor accumulates dot products internally, the authors have achieved excellent times of 9.8 seconds for a 5Ø x 5Ø multiply and 14 seconds for a 5Ø x 5Ø inverse, as shown in Table 12.

Examples and additional benchmarks are available in a paper by Platt and Platt (1985).

5. SAS IML and MATRIX

SAS has been furnishing MATRIX with its base product since the late 197Ø's, but SAS is now phasing it out in favor of a separate prod' t called IML (Interactive Matrix Language). Version 5 IML runs on 32-bit minicomputers and IBM mainframes, and Version 6 IML runs on IBM microcomputers. The emphasis here is on the microcomputer version, although this version has not been released yet. Note that the marketing of IML for microcomputers is done on a site lease basis which is designed to appeal to large organizations. There is no provision for individual purchase. The hardware requirements are substantial - at least 512K of memory and more than 5 megabytes of hard disk space. It runs in PC-DOS and some versions of MS-DOS.

Those who have used the SPF editor in IBM's TSO may be pleased that a similar editor is included in the microcomputer implementation of SAS. One can accumulate commands in the editor and submit them whenever it is desired to process commands. Online help is available. MATRIX has a facility (PROC MATRIX PRINT) which causes output from each computation to be printed. IML improves on this by allowing the feature to be toggled on and off. The other languages have it, too, except for MATLAB. It is very helpful in debugging.

Another advantage that IML has over MATRIX is the ability to reference a one-dimensional array by only one subscript. In MATRIX, the elements of a vector are "V(1,I)" or "V(I,1)", whereas they can be called "V[I]" in IML Version 6.

MATRIX has only BASIC-style subroutines with global variables, but IML has also subroutines with local variables. IML shares with others the ability to interrupt a session and run DOS commands or any programs, including FORTRAN, etc.

One of the outstanding features of MATRIX which is carried over into IML is the subscript reduction operator, which is similar to the reduction operator of APL and others, but operates directly on the subscripts. There are a variety of useful operators, including the mean operator shown in Table 6. Moving the operator from the first to the second subscript switches from means of columns to means of rows.

Note that the only difference between MATRIX and IML in Table 7 is that IML uses square brackets for subscripting. The subscript syntax of Version 5 is a little messier than what is shown for Version 6 (microcomputers) in Table 7.

IML and MATRIX have a wide range of mathematical and statistical functions. Some of the more esoteric operators such as Gram-Schmidt orthogonalization and the fast Fourier transform are available in MATRIX and Version 5 of IML but not yet in Version 6.

For doing analysis of variance by regression a facility is needed for obtaining design matrices (creation of dummy variables), which both SAS languages do. Furthermore, the horizontal direct product is available ("HDIR" in IML, "@|" in MATRIX) for forming interaction terms. Similarly, STATGRAPHICS has a "CROSS" operator. Other languages can accomplish this with the Kronecker product, but it is comparatively awkward, especially in the unbalanced case. Note that the SAS languages also have an excellent orthogonal polynomial facility which works well in the unbalanced case.

SAS MATRIX has quite a complete set of tail probabilities, inverse probabilities, and random number generators. The inverse probabilities include only the normal and beta, but the t, chi-square, and F can be obtained from the beta. On the other hand, the only inverse available in IML is the normal, and random numbers are available only from the uniform and normal distributions.

MATRIX has to rely on communication with the rest of SAS to do graphics, but IML Version 5 has a substantial graphics component. This includes facilities for spline fits and character labels (of arbitrary length) for the points on a plot. There is as yet no such facility in the microcomputer version.

6. MATLAB

MATLAB has been available for about five years as a FORTRAN program on large computers. This review is based on a demo disk for PC-MATLAB, which is an enhanced version written in C for MS-DOS microcomputers. This will soon be available also for the Macintosh and VAX VMS. The emphasis is more on numerical analysis and engineering than on statistics. Not all documented functions are present on the demo disk. Notice in Table 1 that MATLAB is the only one of the microcomputer languages reviewed that is copy protected.

The program has an interactive mode in which commands are executed as they are given. There is also a batch mode, but the command file must be formed elsewhere because MATLAB has no batch editor. There is provision for a diary file in which interactive commands can be accumulated, but the file also contains the output, which would need to be edited out to get a batch command file. One does not need to retype an erroneous command in interactive mode because a line editor is available to edit the last command.

MATLAB has a useful online help facility. Typing "HELP" causes all of the commands to be listed on one screen. Choosing a command from this list, one can then get more detailed help.

There is not much ability to label output. Character data are allowed but they cannot be used to label the rows of a numerical matrix.

With others, MATLAB shares the ability to suspend a session and run a DOS command or any program. The MATLAB documentation suggests that a FORTRAN program can be used with MATLAB by first storing data, running the FORTRAN program on the data, and then reading back into MATLAB the FORTRAN results.

MATLAB allows subroutines with local variables. These subroutines can be precompiled.

Given that the program originated with the well-known numerical analyst Cleve Moler, it is not surprising that MATLAB contains a full complement of procedures for numerical analysis, as shown in Table 8. On the other hand, Table 10 shows that there are no tail probabilities available, and none of the corresponding inverses, although normal and uniform data can be generated.

For graphics, MATLAB supports the Hercules card, IBM color graphics, and IBM enhanced graphics, as does STATGRAPHICS. It is possible to overlay plots, and there is provision for showing strata such as male and female on one plot. Normal plots and exploratory data analysis commands are not included.

The benchmarks in Table 12 are quite good.

7. S

S was written to work within the UNIX system at Bell Labs, and until a recent implementation on VAX VMS, it ran only in the UNIX environment. Although there are AT&T microcomputers that support UNIX, the smallest machine for which AT&T recommends S is the 3B2. Note that S costs $8000, except that it costs only $400 for universities. This review is based mainly on the manual by Becker and Chambers (1984).

Both interactive and batch modes are supported by S. Batch files can be edited by a Unix editor.

The manual shows excellent facilities for character data and the labeling of output. Multidimensional arrays are allowed, as is true of APL but not the others, although facilities for character manipulation can be used in GAUSS and SPEAKEASY to index 2-dimensional arrays.

The program is integrated into UNIX - it allows UNIX commands and UNIX programs in C, FORTRAN, and PASCAL to be executed from within S. Extensions to S can be accomplished with macros.

There is a sweep operator in S, but it is for centering data, and not for doing Beaton (1964) sweeps.

The available commands include substantial facilities for regression analysis, even the Furnival-Wilson (1974) leaps and bounds all-subsets regression algorithm. On the other hand, there is not much available for analyzing designed experiments. In particular, there are no commands for forming dummy variables.

The facilities for probability distributions, inverse probability distributions, random number generators, and quantile plots are among the best available, and they are named consistently so as to be easily remembered. S also has one of the best plotting facilities available. Points can be labeled effectively, e.g., with two-letter state abbreviations if the points represent states. The values of a third variable can be indicated by using the values of the third variable to determine the size of circles centered at the points.

The S program has impressed many statisticians who have had a chance to use it. For a very favorable review of the manual and the program, see Larntz (1986).

8. SPEAKEASY

SPEAKEASY has been available on various IBM computers, and more recently the DEC VAX, since the 1960's. Written in FORTRAN, it now also runs on MS-DOS microcomputers. Although the microcomputer version does not have all of the mainframe features, it does have over 600 functions available. It uses Intel FORTRAN and requires 640K, just as BMDP has done. SPEAKEASY uses the full memory, with nothing else allowed in the way of coresident programs. It also takes about 5 megabytes of hard disk space.

The size of SPEAKEASY makes it harder to find the right command than it would be in a smaller system such as MATLAB, where the "HELP" command puts the complete command list on one screen. The "HELP" command in SPEAKEASY puts a list of command categories on the screen, and these can be examined with more detailed help commands. In general, there is room for improvement in the documentation. There is very little documentation specific to the microcomputer version, and it is not always clear how to use the commands.

Commands can be executed interactively or in batch mode, and a line editor is available to edit batch commands. The "JOURNAL" command is available to toggle on and off the automatic output of commands and/or results. By using it to record commands in interactive mode, one can form a file that can be edited and run in batch mode.

SPEAKEASY does a fine job with character data and the "TABULATE" command gives nicely labeled printouts of arrays. Ling (1985) has given examples of this feature.

Another useful feature stressed by Ling is the ability to interrupt a session and execute DOS commands and programs such as FORTRAN. A FORTRAN link is available, but only to Intel FORTRAN. A recently added feature in SPEAKEASY is the inclusion of subroutines with local parameters.

There are two kinds of two-dimensional structures, arrays and matrices. If A and B are arrays of the same size, then "A*B" yields the elementwise product, of that size. If A and B are matrices of appropriate size, then "A*B" is the matrix product. Thus commands are necessary to convert from matrix to array ("AFAM") and array to matrix ("MFAM"), to assure that the desired operation is performed.

The last two columns of Table 7 may require some explanation. If L and R each have N rows and L has M columns, then they can be joined by setting "L[1,M+1]=R". The procedure for vertical concatenation of arrays T and B with the same number of columns is to set "T[N+1,1] = B" if T has N rows. In general, U(I,J)=V allows you to insert V anywhere in U, regardless of their dimensions, and SPEAKEASY will adjust the dimensions of the result.

SPEAKEASY has a substantial set of graphics commands using both text characters and color graphics. Plots can be displayed interactively, with a new graph overlayed on one that is already on the screen. Labels can similarly be added. This is the only way, however, to display separate strata with separate symbols. Thus, the procedure would be awkward if there are several strata, because each stratum requires a separate plot command, although the commands could be placed in a loop.

Being written in FORTRAN, SPEAKEASY is not as fast as MATLAB and GAUSS, as shown in Table 12. Unfortunately, it does not seem possible with current FORTRANs to keep up with optimized assembler and C code for the math coprocessor.

9. Conclusions and Recommendations

The headings in Tables 1-12 indicate some of the features which are, in the author's opinion, important in a statistical matrix language. It is hoped that authors of such languages and reviewers of them will take into account the listed features.

One conclusion that can be drawn from the tables is the surprising degree of similarity among the various matrix languages. Especially in Tables 6 and 7 there are strong parallels among them. Their capabilities and notations are very similar for matrix operations, indices, subsetting, and concatenation.

There are, of course, significant differences among the languages. To some extent these gaps tend to narrow, perhaps because the authors are aware of what others are doing, perhaps because their users are aware of what others are doing and demand similar facilities. One might wonder if it is coincidental that SPEAKEASY, GAUSS, and MATLAB have all added recently subroutines with local variables, and IML also has this feature, although its predecessor MATRIX does not.

In the area of plots and probability distributions, it would be nice to see other languages copy from the example set by IML Version 5, S, and STATGRAPHICS. Especially for plots, which cannot be readily programmed by the user, it would be good to see some of the other languages add some features. Statistical detective work is enhanced by well-labeled plots, and every language should be able on scatter plots to distinguish different strata and to label each point with the values of an ID variable.

There are some features that are available in few if any of these languages. Many of them are not very hard to implement in a matrix language, but some are quite difficult. In the not so hard class are multivariate normal variate generation (available in STATGRAPHICS) and Wishart matrix generation, as described by Kennedy and Gentle (1980). More difficult are L1 estimation and all-subsets regression. The S package includes the Furnival-Wilson (1974) algorithm, which is remarkably efficient and and remarkably intricate. Many such difficult algorithms are obtainable free or at low cost in FORTRAN, so the language with FORTRAN linkages will allow the user great latitude in algorithms, beyond what is an integral part of the language.

It could be argued that time series operators should included among the hard-to-program features that should be included in these languages, but there are specialized languages such as RATS that include many time series functions along with a matrix language. For serious time series analysis, such specialized languages are better suited than any of the microcomputer languages considered here, except that STATGRAPHICS has strong time series features and is a contender in this area.

Sometimes textbook algorithms are wasteful of computer space and time. One example is the formation of a large square matrix from which only the diagonal is needed. The IML Version 6 manual, p.116, follows such an algorithm in forming a n x n projection (hat) matrix, which will be very large if the number of cases n is large, and then using only the diagonal (for leverages and confidence intervals). It might be asking too much to expect that an interpreter would recognize that only the diagonal is needed, and produce only the needed part, although this is a simpler form of optimization than is used in some FORTRAN compilers. At a minimum, the manual should show how to compute what is needed. That is, it should be explained that a more efficient equivalent of

```
                    VECDIAG(X*INV(X'*X)*X')
```

is

```
                    (X*(INV(X'*X))#X)( ,+).
```

The second method need not be recommended in all instances, but only when n is large.

There are two disparate goals for these matrix languages. On the one hand they should be interactive and easily debugged, and on the other hand they should be efficient. The first goal is more easily met by an interpreted language, and the second goal is more easily met by a compiled language. An ideal compromise would involve a dual-mode language which would allow debugging in interpreted mode and a switch to compiled mode for efficient operation. GAUSS and MATLAB already make this available, to some extent, by allowing some compilation.

It can be argued that the availability of powerful matrix languages on personal computers should change the way that statistics is done. The GAUSS documentation suggests that one should now be much more inclined to do maximum likelihood estimation and other computations that do not necessarily give closed-form answers. As stated in the GAUSS manual, page 9-2, "Remember, while you're having dinner, your PC and GAUSS can do computations that would cost hundreds of dollars on a mainframe!." The ideal situation is that the statistician knows exactly how to do the appropriate computations, given the powerful hardware and software on the desktop, but there will need to be a lot of changes to manuals and textbooks before this knowledge is widespread. Much of the software is here now, and the authors of these languages deserve thanks for making available excellent tools for doing the computationally intensive statistics of the future.

References

Beaton, A. (1964), "The Use of Special Matrix Operators in Statistical Calculus," Research Bulletin, Princeton: Educational Testing Service.

Becker, R. and Chambers, J. (1984), S: An Interactive Environment for Data Analysis and Graphics. Belmont, CA: Wadsworth.

Furnival, G. M. and Wilson R. W. (1974), "Regression by Leaps and Bounds." Technometrics 16, 499-511.

Kennedy, W. J. and Gentle, J. E. (1980), Statistical Computing. New York: Marcel Dekker.

Larntz, K. (1986), Review of S: An Interactive Environment for Data Analysis and Graphics, by R. Becker and J. Chambers, Journal of the American Statistical Association, 81, 251-252.

Ling, R. F. (1985), "Super-Interactive Statistical Software on Microcomputers," paper presented at the Annual Meeting, American Statistical Association.

Platt, W. G. and Platt, C. A. (1985), "Efficient Statistical Computing with Microcomputers", paper presented at the Annual Meeting, American Statistical Association.

Smith, B. (1985), "APL: Daring to Be Different," PC, December 10, 1985, 167-178.

Figure 1. Addresses

STSC APL, STATGRAPHICS
STSC, Inc
2115 E. Jefferson St.
Rockville, MD 20852

GAUSS
Applied Technical Systems
P. O. Box 6487
Kent, WA 98064

SAS IML, MATRIX
SAS Institute Inc.
Box 8000
Cary, NC 27511-8000

PC-MATLAB
The MathWorks, Inc.
158 Woodland St.
Sherborn, MA 01770

S
AT&T Technologies Software Sales
P.O. Box 25000
Greensboro, NC 27420

SPEAKEASY
Speakeasy Computing Corporation
222 West Adams St.
Chicago, IL 60606

Table 1. MATRIX LANGUAGES

	OPERATING SYSTEM	PRICE	PROTECTED?	NEED HARD DISK?	MEMORY
APL STATG	MS-DOS, etc.	595 +695 (295+350 ACAD)	NO	NO	512K
GAUSS	MS-DOS	250	NO	NO	256K
SAS, IML	PC-DOS, etc. VMS, TSO, CMS	(SITE LEASE)	NO	YES	512K
SAS,MATRIX	OS,CMS, VMS,etc.				
MATLAB	MS-DOS	695 (395 ACAD)	YES	NO	256K
S	UNIX VMS	8000 (400 ACAD)			
SPEAKEASY	MS-DOS, etc. VMS, CMS TSO	LEASE	NO	YES	640K

Table 2. MODES OF OPERATION (ALL INTERPRETED)

	BATCH	EDITOR	INTER-ACTIVE	DIARY FILE	HELP	PRINT TOGGLE
APL STATG	YES	YES	YES	YES	YES	YES
GAUSS	YES	YES	YES	YES	NO	YES
IML	YES	SPF-TYPE	YES	YES	YES	YES
MATRIX	YES	OPER SYS	OPER SYS	OPER SYS	NO	YES (NO TOGGLE)
MATLAB	YES	OPER SYS	YES	YES	YES	NO
S	YES	(UNIX)	YES	YES	YES	YES
SPEAKEZ	YES	LINE	YES	YES	YES	YES

Table 3. DATA STRUCTURES

	CHAR DATA	3-D ARRAYS	MISSING DATA	VECTOR
APL STATG	YES	YES	YES	YES
GAUSS	YES	SORT OF	YES	NO
IML	YES	NO	YES	YES
MATRIX	NAMES	NO	NO	NO
MATLAB	YES	NO	YES	YES
S	YES	YES	YES	YES
SPEAKEZ	YES	SORT OF	YES	YES

Table 4. ENVIRONMENT

	DOS COMMANDS	ASCII READ,WRITE	FORTRAN LINK
APL STATG	YES, AND PROGRAMS	YES	C,ASSEMBLER
GAUSS	YES, AND PROGRAMS	SPACES NEEDED	(NEW)
IML	YES, AND PROGRAMS	YES	SHARED DATA
MATRIX	-	-	AWKWARD
MATLAB	YES, AND PROGRAMS	SPACES NEEDED	SHARED DATA
S	UNIX	YES	YES (UNIX)
SPEAKEZ	YES, AND PROGRAMS	YES	INTEL

Table 5. PROGRAMMING STRUCTURES

	DO
	LOOPS
		IF
		ELSEIF
		ELSE
		END
			SUB
			ROUTINES

APL STATG	NO	NO	LOCAL
GAUSS	DO WHERE	YES	(NEW) LOCAL COMPILE

IML	DO
	END
		IF
		THEN
		ELSE
			LOCAL
			MACRO

MATRIX	DO
	END
		IF
		THEN
		ELSE
			LINK
			RETURN
			MACRO

MATLAB	FOR END	YES	LOCAL COMPILE

S	FOR( )
	{ }
		IF
		ELSE
			MACRO

SPEAKEZ	FOR NEXT	IF( ) ( )	LOCAL

Table 7. INDICES, SUBSETTING, CONCATENATION

	INDEX VECTOR	SUB MATRIX	HORIZ CONCAT	VERT CONCAT
APL	⍳N	A[⍳3;]	L,R	T,[1]B
GAUSS	1:N	A[1:3,.]	L~R	T|B
IML	1:N	A[1:3,]	L||R	T//B
MATRIX	1:N	A(1:3,)	L||R	T//B
MATLAB	1:N	A(1:3, :)	[L R]	[T;B]
S	1:N	A[1:3,]	CBIND (L,R)	RBIND (T,B)
SPEAKEZ	INTEGERS(N)	A(INTEG- ERS(3),)	Assign R to col (n+1)	Assign B to row (n+1)

Table 6. OPERATIONS

	ELEMENT PRODUCT	MATRIX PRODUCT	INVERSE	COL MEANS
APL	x	+.x	⌹A	(+⌿X)÷ 1↑⍴X
GAUSS	*	*	INV	MEANC(X)
IML	#	*	INV	X[:,]
MATRIX	#	*	INV	X(.,)
MATLAB	*	*	INV	[m,n]=size(X) sum(X)/m
S	*	%*	SOLVE	col(X,MEAN)
SPEAKEZ	* (array)	* (matrix)	INVERSE	SUMCOLS(X)/ NOROWS(X)

Table 8. NUMERICAL ANALYSIS

	CHOL	EIGEN	SVD	GINV	QR	GS	FFT
APL STATG	NO	EIGEN	NO	NO	NO	GS	FFT
GAUSS	CHOL	(NEW)	(NEW)	(NEW)	(NEW)	(NEW)	(NEW)
IML	HALF ROOT	EIGEN	SVD	GINV	NO	GSORTH (VERS ION5)	FFT (VERS ION5)
MATRIX	HALF	EIGEN	SVD	GINV	NO	GS	FFT
MATLAB	CHOL	EIG	SVD	PINV	QR	ORTH	FFT
S	CHOL	EIGEN	SVD	YES	YES	GS	NO
SPEAKEZ	CHOL- DECO	EIGEN- VALS EIGENVECTS	NO	NO	NO	NO	NO

Table 9. STATISTICAL OPERATORS

	BEATON SWEEP	DESIGN MATRIX	ORTHOG POLY	RANKS	KRONECKER PRODUCT
APL STATG	SWEEP	IND NEST	NO	YES	YES (CROSS)
GAUSS	NO	NO	(NEW)	(ADDED)	YES
IML	SWEEP	DESIGN	ORPOL	RANK	(HDIR)
MATRIX	SWEEP	DESIGN	ORPOL	RANK	@ (@1)
MATLAB	NO	NO	QR	SORT	KRON
S	NO	NO	GS	RANK	KRONECKER
SPEAKEZ	NO	NO	NO	RANKER	KRONECKER

Table 10. DISTRIBUTIONS

	TAIL PROBS	INVERSE PROBS	RANDOM NUMBERS	QUANTILE PLOTS
APL STATG	YES	YES	YES	YES
GAUSS	YES	(t) NORMAL	UNIF NORMAL	NO
IML	YES	NORMAL	UNIF NORMAL	VERSIONS
MATRIX	YES	NORMAL BETA	YES	SAS
MATLAB	NO	NO	UNIF NORMAL	NO
S	YES	YES	YES	YES
SPEAKEZ	YES	NORMAL	UNIF NORMAL	NO

Table 11. PLOTS

	HI-RES SCATTER	OVERLAY	LABELING	EDA,ETC.
APL STATG	VERSATILE	VERSATILE	VERSATILE	VERSATILE
GAUSS	(NEW CG) PRINTER	(NEW)	(NEW)	NOT MUCH
IML	VERSIONS	VERSIONS	VERSIONS	VERSIONS
MATRIX	SASGRAPH PLOT	SAS	SAS	SAS
MATLAB	VERSATILE	YES	YES	NOT MUCH
S	VERSATILE	VERSATILE	VERSATILE	VERSATILE
SPEAKEZ	CG, PRINTER	VERSATILE	YES	NOT MUCH

Table 12. BENCH MARKS

	50x50 MPY PC	50x50 INVERSE
APL	155 sec	139 sec
GAUSS	9.8 sec	14 sec
IML	40 sec	69 sec
MATLAB	10 sec	24 sec
S		
SPEAKEZ	95 sec	120 sec

PUBLISHING STATISTICAL SOFTWARE

John C. Nash, University of Ottawa

Abstract

Statistical computation is at the heart of a large part of all statistical research and analysis. The growing complexity and diversity of software for statistical computations implies that statisticians spend a growing proportion of their professional lives developing, learning and using such software.

This paper will review the mechanisms by which statistical software is published, that is, made available to statistical practitioners. In particular, emphasis will be placed on the issue of academic or commercial credit for the research and development work which good software demands. Potential approaches to inclusion of software in professional performance evaluation are discussed.

Traditional software publishing

Traditional approaches to publishing statistical software mirror the methods used to publish scientific ideas in general. That is, programs of a more academic nature have been disseminated in journals and related sources, while didactic or less weighty codes have appeared in trade or special interest magazines. Books, too, have had their role, either in discussing algorithms in a step-and-description fashion which readers can then program in a manner suited to their needs and interests, or in presenting listings (generally in FORTRAN) of the author's code.

Examples of journals which have published statistical software are the ACM Transactions on Mathematical Software, with the companion publication of the Collected Algorithms of the ACM (CALGO), which publishes the complete listings, and Applied Statistics, which has included numbered programs since the late 1960's. Magazines such as Byte and Interface Age have included codes from time to time for statistical applications. Sadly, many of these have been of inferior quality, which led to my own involvement in writing articles to attempt to expose the difficulties of preparing scientific software (e.g. Nash, 1981).

Books containing algorithms -- for example, Kennedy and Gentle (1980) or Nash (1979) -- are rarer than those presenting listings. In the latter category are Ratkowsky's (1983) presentation of nonlinear least squares FORTRAN programs and Maindonald's (1984) book on linear statistical computations with programs and examples mainly in BASIC.

Most statisticians are familiar with scientific software libraries, from which subroutines (or complete programs) may be called to carry out calculations. Well-known general purpose libraries are those of IMSL and NAG, while BMDP focuses on biostatistical routines. Several libraries suitable for microcomputer use have been advertised in the recent past. However, I am aware of only one -- C. Abaci's Scientific Desk -- of which the authors have a serious interest in and understanding of numerical reliability. This does not mean there are no other quality scientific software libraries, but that quality is difficult to ascertain. (Note that IMSL and NAG have released subsets of their mainframe libraries.)

Such libraries can be considered as a form of software publishing. This role becomes clearer when the activities of traditional publishing houses are noted: Wiley's efforts with the Peerless Engineering Service Scientific Subroutine Library and Wadsworth's investment in Statpro. Ignoring the important question of commercial viability of such ventures, it is necessary to decide whether the software is published or simply "made available", since source codes may not be released. For the present, we will take libraries as a form of publishing of software.

Similarly, packages may be thought of as a form of software publishing, though the creative core of the programmer/statistician's art is now almost certainly hidden in object code, and often behind the curtain of a (possibly inconvenient) user interface.

Finally, authors of technical reports and journal articles may offer to make their programs available privately. The notice of the availability of the programs constitutes their "publication".

Paper-based dissemination of software has the distinct advantage that humans can read and appreciate it. However, while it is relatively inexpensive to obtain listings, implementation and testing may be costly, if not in money, then certainly in effort and time. There is also the question of publication delay, which implies that the user gets a program which is likely to date from two years prior to publication, even if hot off the press. Computer magazines, traditionally up-to-date with news and information, now have publication delays approaching a year.

Updates to books and journal articles, except for errata, are likely to be similarly delayed. Worse, modest changes which dramatically improve the efficiency of the program even though they do not change the scientific content of ideas therein may not be considered "original" by journal editors. Updates and improvements are therefore unlikely to be easily obtained.

Private distribution by the author or his/her organization has been a popular mechanism for scientific software distribution. It works well as long as the chore does not become too tedious or expensive. Since people move, change jobs, take sabbatical leave, etc., the distribution link usually dies after a few years. With no external monitoring of the design and development of such software, quality may vary greatly.

Some paper publications also offer distribution of machine readable media. Originally this meant cards or magnetic tapes. More recently, diskettes have become popular, with some books even including a pocket in one cover for the diskette (e.g. Nash and Walker-Smith, 1986). Another approach involves printing bar-code style information in strips right on the paper which can be read with a special reader (the Cauzin Systems Softstrip mechanism).

Alternative publishing mechanisms

Other mechanisms exist by which software may be published.

"Shareware" or "freeware", also called "user supported software", is distributed by users giving copies to colleagues and associates. The software creator generally claims copyright on the product, but grants permission for the copying. A fee is demanded for the manual and for updates and corrections of the software or for technical support, which are usually supplied by the author. In some cases, for example the word processing program PC-WRITE, this concept has proven extremely successful. EPISTAT is a suite of statistical programs distributed as shareware.

Various computer bulletin boards and some commercial information vendors provide for the down-loading of public domain software. Mostly, this consists of games or utility programs, though some technical software may be found on the commercial systems, for example, the Byte Information Exchange (BIX). User charges are generally based on connect time plus telecommunications charges.

A further alternative is electronic mail. Dongarra and Grosse (1985, 1986) offer an impressive array of mathematical software, available to anyone with access directly or indirectly to the ARPAnet. This includes most of the academic electronic mail networks, such as BITNET (and NETNORTH), MAILNET, CS-NET, USENET, PHONENET, and others. To obtain a listing of what is available, a user sends the message

```
SEND INDEX
```

to the pseudo user NETLIB at the ANL-MCS node of the ARPAnet. (NETLIB@ANL-MCS.ARPA). A similar message, for example,

```
SEND CSVDC FROM LINPACK
```

will initiate the reply which consists of that FORTRAN subroutine from the LINPACK collection. Currently available are

EISPACK	--	matrix eigenvalue problems
LINPACK	--	linear algebraic calculations
CALGO	--	Collected Algorithms of the ACM
FUNPACK	--	special functions

and a number of other collections.

Comparison of alternative and traditional software publishing mechanisms

To the best of my knowledge, a comprehensive cost-benefit comparison of the different modes of software publishing has not been carried out. Statistical software, which is a relatively small market segment of the overall software marketplace, cannot be compared directly with such products as word-processing packages or communications packages.

Nevertheless, it is clear that shareware, online and eimail software delivery modes all offer easy updating of programs, providing the user is willing to pay the generally small charges for contacting the human or machine sources. It is in this placement of the responsibility for obtaining updates on the user that fairly substantial cost savings to the vendor / supplier arise. Instead of the software vendor mailing updates to many users, only those who request them are serviced. This may lead to many users working with obsolete or defective versions of the programs, but this is hardly different from the situation where a user has not seen or bothered to implement a published correction. Software & documentation are generally delivered in machine readable form by the alternative publishing mechanisms. Some users may prefer nicely printed manuals including tutorials, reference material, and installation guides. For electronically transmitted files, data compression may be advisable to cut telecommunication costs. Version/edition control may be a problem, especially with the temptation to add updates as soon as they are available.

The mode of use of software distributed by online or electronic mail mechanisms will be considerably different from paper-based distribution. Compared to conventional libraries/packages, the source of advice/support will likely become the database supplier in place of the computing center of a university / company / institute. The user may have a much bigger role to play in the evolution of software, for example, in the types of software supplied, the correction of faults, the enhancement of features, the development of applications and the preparation of documentation.

Economic and academic issues

All of the alternative distribution mechanisms have the advantage of initial low-cost of delivery to the user. They rely on word-of-mouth or traditional advertisements, however, to attract user attention, and may therefore not achieve a desired level of awareness. Shareware may impose costs of production for manuals, updates, etc. on the vendor of a very similar level to those experienced by traditional software publishers.

It is more difficult to estimate costs for online and electronic mail modes of distribution. For sake of discussion, a figure of the order of 25 cents/1000 characters is probably reasonable at the present time for the communications costs. (Elmail is probably cheaper than online distribution, but many of the costs are buried within the overall network management costs, frequently borne by universities as a service to their members.) Both of the electronic methods of distribution gain by the lack of human intervention in the distribution process. Furthermore, the user may choose to download only a small segment of software or documentation of particular interest.

To date, none of the software being distributed electronically is returning any revenue to its creators. Even commercial systems are charging mainly for telecommunications and database provision services, along with profit for the vendor. If software authors are to be expected to create programs for distribution, they must be rewarded, and pricing mechanisms which balance between author greed and user theft (i.e. unauthorized copying) are needed. The major unanswered question is whether a price exists which is high enough to cover the hardware and communications costs and a royalty to the author, but low enough so that it is not worthwhile to the user to have any but the latest authorized version.

The obstacles to the development of alternative methods for publishing statistical and other scientific software are primarily those relating to credit for the creation of intellectual property. In particular, the general academic fixation with paper-based journals implies that authors who support alternative publication vehicles may find that their work "doesn't count" for academic rewards in tenure or promotion. A related issue is that of crediting workers who actually support software in general use -- software upon which much research in practically all disciplines may depend. Some years ago, I derived formulas to extend the Gini Ratio -- a statistic used to assess the uniformity of income distributions -- to the situation where incomes may be negative. Furthermore, I documented the interpretation of the statistic in these cases, wrote a program to analyze the data, and ran a considerable portion of the calculations. However, when the report of this work was published, I was neither listed as author, nor mentioned as someone who had provided assistance. The point here is not that the authors were ill-intentioned, but that the role of "supporting cast" is often accorded a very limited status. In the case described, pointing out to the authors the extent to which the results of their work relied on my software resulted in satisfactory recognition.

A more general difficulty concerns the possibility that a program may be altered over time by contributions from a number of workers. Who should then get the credit? This is a continuing issue. It is compounded by the reality that a researcher gets more academic value from a completely different program, even if it does not work particularly well, than from a minor change to an existing program which doubles its performance.

What can be done to ensure good software is published?

This paper will not attempt to define "good software". However, whatever metric is used to judge software as good, I would suggest that the publication of such software should be such that:

1) it is widely available and easily obtained and installed;

2) the price should be reasonable, that is, comparable to the price of a statistics monograph.

It is possible that if such conditions apply, the only software available would be good by virtue of market competition. In order to be worthwhile for programmers and statisticians to create this software, the profession must recognize, as a body,

- that communications need not be on paper;
- that software is an important part of our work.

Notable movements in these directions are perceptible:

1) the Natural Sciences and Engineering Research Council of Canada asks researchers in the Computer Science category to list software packages produced.

2) statistics journals and newsletters, in particular American Statistician, have been including an increasing amount of software announcements and reviews.

3) the Dongarra / Grosse NETLIB project has established the technical possibility of electronic mail for software distribution.

While the development of electronic distribution depends on the growth of the required network infrastructure, there are a number of individual initiatives possible. Statisticians can strive to ensure that software support and assistance is properly credited. Personal software activity can be listed in annual reports. At the risk of giving offense to users, one can demand acknowledgement or even co-authorship. More subtly, the mindset of the profession can be influenced by asking questions about software generation and support in employment interviews or questionnaires. Use of existing electronic mail facilities for transmission of software, manuscript or bibliographic material is a useful step to learning how, and how well, systems work.

At present, it is unclear which of the different options for statistical software publishing will assume major roles in the coming decade. It seems likely that statistical software will be published mostly by electronic means at some point in the not very distant future. However, the precise mix of delivery methods remains to be revealed by the passage of time.

References

Dongarra J. & Grosse E. (1985) "Distribution of Mathematical Software by Electronic Mail", SIGNUM Newsletter, vol.20, no. 3, pp. 45-47 (July)

Dongarra J. & Grosse E. (1986) "Distribution of Mathematical Software by Electronic Mail", SIAM News, vol. 19, no. 1, pp. 15-16 (February)

Kennedy W.J. & Gentle J. (1980) Statistical computing, Marcel Dekker, New York.

Nash J.C. (1979) Compact numerical methods for computers:linear algebra and function minimisation, Adam Hilger, Bristol (Halsted Press, New York in U.S.A.)

Nash J.C. & Walker-Smith M. (1986) "Using compact and portable function minimization codes in forecasting applications", INFOR, in press.

Maindonald J. (1984) Statistical computation, Wiley, New York.

Ratkowski D.A. (1983) Nonlinear regression modelling, Marcel Dekker, New York.

FRONTIERS IN SIMULATION AND STATISTICS I

Organizer: *Dennis S. Friday, National Bureau of Standards*

Estimation of Response Time Characteristics in Priority Queueing Networks Via an Interpolation Methodology Based on Simulation and Heavy Traffic Limits
B. Simon, J. S. Willie, AT&T Information Systems Laboratories

Sequential Simulation Run Control Using Standardized Time Series
Lee Schruben, Cornell University

On Some Statistical Issues in Simulation Studies
C. L. Mallows, V. N. Nair, AT&T Bell Laboratories

ESTIMATION OF RESPONSE TIME CHARACTERISTICS IN PRIORITY QUEUEING NETWORKS VIA AN INTERPOLATION METHODOLOGY BASED ON SIMULATION AND HEAVY TRAFFIC LIMITS

B. Simon and J. S. Willie
AT&T Information Systems Laboratories, Denver, Colorado

Abstract

Consider a network of priority queues, and suppose one is interested in describing some characteristic, say $f(\lambda)$, of a particular response time distribution as a function of the arrival rate λ. Here, $f(\lambda)$ might be a moment or quantile of the response time distribution, or any of a number of other interesting functions of the arrival rate. In this paper, a technique for estimating $f(\lambda)$ as a function of λ over some region of interest is presented. The technique involves estimation of the $f(\lambda)$ at a few values of λ by discrete event simulation, normalization of the estimated $f(\lambda)$, regression of a low order polynomial on the normalized estimated $f(\lambda)$, and the heavy traffic value, and, finally, a renormalization of the fitted polynomial.

1. Introduction

In the study of complex networks of priority queues encountered in computer and communication system modeling, one is often interested in describing characteristics of some "steady state" response time distribution as a function of the rate at which customers arrive at the network. This paper presents a *simulation - heavy traffic* interpolation methodology that is useful for providing such descriptions. In Section 2, we describe a general class of queueing network models for which the interpolation methodology is applicable. Sections 3, 4, and 5 describe the three main ingredients of the interpolation method; namely, simulation, heavy traffic and the normalization. In Section 6 we will describe the interpolation technique and in Section 7 we illustrate the technique with an example.

2. A Class of Priority Queueing Network Models

The systems we will consider here are open networks of priority queues with $K < \infty$ customer types. A customer type, y, is specified by three vectors of length L_y (L_y is the number of steps in y's itinerary). The first vector, $(node_y(1), node_y(2), \ldots, node_y(L_y))$, gives the sequence of nodes that y visits. The second vector, $(prio_y(1), prio_y(2), \ldots, prio_y(L_y))$, gives the priority levels at each step, and the third vector, $(g_y(1), g_y(2), \ldots, g_y(L_y))$, are the service time distributions for each step.

Note that random routes are allowed as long as there is a bound on the potential length of the route (i.e. a finite number of possibilities). This finite restriction can be relaxed, but the notation becomes burdonsome. Complicated routing schemes, such as "nested Markov routing", are described in Simon [1985].

The queueing discipline at each node in the network is preemptive resume. Type y customers enter the system as a Poisson process with rate λ_y, and the arrival streams of the K customer types are mutually independent. Note that the number of nodes in the network and the number of priority levels at each node are arbitrary, and are given implicitly by the vectors $node_y$ and $prio_y$, $y = 1,2, \cdots, K$.

3. Discrete Event Simulation

In the study of complex networks of priority queues encountered in practice, discrete event simulation is a useful tool for providing reliable descriptions of response time characteristics of interest if care is taken in the design and implementation of the simulation experiment and analysis of the simulation output (see Fishman [1978] and Iglehart and Shedler [1980], for example).

For some system, arrival rate λ_j, $(0 < \lambda_j < c)$ and response time characteristic $f(\lambda_j)$ of interest, suppose we obtain a point estimate $\hat{f}(\lambda_j)$ of $f(\lambda_j)$ via a simulation experiment. Since $\hat{f}(\lambda_j)$ depends on the particular observation of the system from the simulation run, it is often important that some assessment of the accuracy of the point estimate $\hat{f}(\lambda_j)$ be obtained. There are several methods available to the experimenter for assessing the statistical precision of the estimates of response time characteristics based on simulation output data (see Welch [1983] and Heidelberger and

Lavenberg [1984]). For example, the regenerative simulation method (when applicable) provides a technique for approximating the distribution of $\hat{f}(\lambda_j)$. In the regenerative method, a cycle is said to begin when a state of the system is reached such that future behavior is independent of past behavior and identical (in distribution) to every other time a cycle begins. For a fixed number n of cycles, the regenerative method provides an estimate $\hat{f}(\lambda_j)$ and a statistic $\hat{se}[\hat{f}(\lambda)]$ such that if n is large,

$$\frac{\hat{f}(\lambda_j) - f(\lambda_j)}{\hat{se}[\hat{f}(\lambda)]} \;\dot{\sim}\; \mathrm{N}(0,1) \,. \qquad (3.1)$$

The $\dot{\sim}$ in (3.1) means that the random variable (or statistic) on the left is approximately distributed as the random variable on the right, and N(0,1) denotes a normal or Gaussian random variable with zero mean and unit variance. The denominator on the left in (3.1) is generally referred to as the estimated standard error of $\hat{f}(\lambda_j)$. Approximate confidence statements about the value $f(\lambda_j)$ can be based on (3.1).

In Section 6 below, an approximation like (3.1) will be an important aspect in the development of the interpolation methodology.

Note that a major drawback of pure simulation methodology is, of course, the computional costs associated with a detailed study of the system under investigation. This can be particularly true when a description of some response time characteristic for relatively high arrival rates is desired (see Blomqvist [1967]).

4. Heavy Traffic Theory

Many of the interesting performance measures of our queueing systems, such as moments and quantiles of response time and queue length distributions become unbounded as the arrival rate to the system approaches capacity, c . Roughly speaking, the heavy traffic theory of queues quantifies the rate at which these functions approach infinity, so that if the functions are properly normalized, one can obtain exact (finite) limits of the functions as the arrival rate approaches capacity. For example, if $W_n(\lambda)$ is the n^{th} moment of the response time distribution of a customer who requires service once, then

$$\lim_{\lambda \to c} (c - \lambda)^n W_n(\lambda) = \frac{n!}{\gamma^n} \qquad (4.1)$$

where γ is a quantity that can be calculated in terms of the system parameters for a large class of systems.

In many systems (even fairly complicated systems such as the example in Section 7), γ can be computed by hand. Systems with nested Markov routing can be solved exactly via systems of simultaeous equations (see Simon [1985]), and as a last resort, in the most general cases (i.e. nested semi-Markov routing), γ can be obtained from a light traffic simulation. Obtaining γ from a simulation experiment will be addressed in a forthcoming paper (see Simon and Willie [1986]).

Equation (4.1) is actually a consequence of a more general result. If $W(t,\lambda)$ is the probability that the response time of a customer (who requires service once) is greater than t when the arrival rate is λ, then

$$\lim_{\lambda \to c} W\left(\frac{t}{c-\lambda}, \lambda\right) = e^{-\gamma t} \,. \qquad (4.2)$$

Equation (4.2) allows us to compute quantiles: If $f(\lambda)$ denotes the p^{th} quantile of the response time distribution when the arrival rate is λ, (4.2) implies that

$$\lim_{\lambda \to c} (c-\lambda) f(\lambda) = \frac{1}{\gamma} \ln\left(\frac{1}{1-p}\right) . \qquad (4.3)$$

Both (4.1) and (4.2) can be generalized to customer types that require service more than once, or require service a random number of times (e.g. queues with feedback). The analog to (4.2) is a weighted sum of exponentials, and (4.1) becomes the n^{th} moment of that distribution. Although the analog of (4.3) cannot be written down in closed form in the general case, the heavy traffic limits of the quantile functions can be easily computed numerically.

It should be pointed out that there remain some unresolved issues associated with a rigorous derivation of (4.2). Equation (4.2) assumes that the stationary distribution of the limiting queueing process (reflected brownian motion) is the limiting stationary distribution of the queueing process. This interchange of limits has never been demomstrated rigorously, although empirical (as well as intuitive) evidence seems to imply it is true.

5. The Normalization

From equation (4.1) we see that if we normalize the function $W_n(\lambda)$ by $(c - \lambda)^n$, it will be finite for λ in the interval $[0 , c]$. The same normalization will keep $Q_n(\lambda)$, the n^{th} moment of the queue length distribution, finite.

A good normalizer will do more than just keep the function finite, though. Suppose we want to approximate a function, $f(\lambda)$, which has the form

$$f(\lambda) = \frac{A(\lambda)}{B(\lambda)} = \frac{A(\lambda)}{B_1(\lambda)\, B_2(\lambda) \cdots B_k(\lambda)}$$

where $A(\lambda)$ and $B_i(\lambda)$ $i = 1,2,\ldots,k$ are polynomials (i.e. $f(\lambda)$ is rational). Suppose we normalize $f(\lambda)$ by $B(\lambda)$ so that we are left to approximate $g(\lambda) = B(\lambda)\, f(\lambda) = A(\lambda)$. Since we approximate $g(\lambda)$ by a polynomial, (and since $g(\lambda)$ *is* a polynomial), if we have enough information about $g(\lambda)$ (i.e. a sufficient number of simulation points, along with the heavy traffic limit), the only error in the approximation will be due to uncertainty in the simulation data. For example, if $A(\lambda)$ is quadratic, and we have two simulation points and the heavy traffic limit (or one simulation point and the light and heavy traffic limits), then $A(\lambda)$ is uniquely determined. Thus, the ideal normalizer would be the "denominator" of $f(\lambda)$.

Many simple functions of queueing systems are known to be rational, and one may conjecture that $W_n(\lambda)$, $Q_n(\lambda)$ and other functions of interest are rational for very general systems. Unfortunately, even if we know that $f(\lambda)$ is rational, we may still have no idea what $B(\lambda)$ is. Our approach is to try to identify as many of the $B_i(\lambda)'s$ as possible, and *use their product as a normalizer*. Much of this work is heuristic, and it remains to be proven that the terms we identify actually appear in $B(\lambda)$.

First of all, heavy traffic theory shows conclusively that $B(\lambda)$ contains a term $(c - \lambda)^n$ if $f(\lambda)$ is $W_n(\lambda)$ or $Q_n(\lambda)$ (if $f(\lambda)$ is a quantile function then $B(\lambda)$ contains $(c - \lambda)$). We conjecture that there are two other classes of terms present in $B(\lambda)$. The first corresponds to "high priority traffic", the second is analogous to $(c - \lambda)$, but is due to non-bottleneck nodes (see Reiman and Simon [1985] for details). The optimal choice of a normalizer is an important and interesting research area.

The normalizer for response time quantiles will always have the form $(c - \lambda)n(\lambda)$, where $n(\lambda)$ is finite for $0 \le \lambda \le c$. Thus, we can rewrite (4.2) and (4.3) as

$$\lim_{\lambda \to c} W\left(\frac{t}{(c-\lambda)\, n(\lambda)}, \lambda\right) = e^{\frac{-\gamma t}{n(c)}},$$

and

$$\lim_{\lambda \to c} (c-\lambda)\, n(\lambda)\, f(\lambda) = \frac{n(c)}{\gamma} \ln\left(\frac{1}{1-p}\right),$$

respectively.

6. The Interpolation Methodology

In this section, we will discuss the estimation methodology associated with characterizing some "steady state" response time characteristic, say $f(\lambda)$ as a function of overall arrival rate.

Suppose we have estimates $\hat{f}(\lambda_j)$ of $f(\lambda_j)$, $j = 1, \cdots, J$ from a few (independent) simulation experiments. Suppose also, that we have statistics $\hat{se}[\hat{f}(\lambda_1)], \cdots, \hat{se}[\hat{f}(\lambda_J)]$ such that

$$\frac{\hat{f}(\lambda_j) - f(\lambda_j)}{\hat{se}[\hat{f}(\lambda)]} \mathrel{\dot\sim} \mathrm{N}(0,1), \qquad (6.1)$$

$j = 1, \cdots, J$ (see Section 3). Let λ_{min} denote the minimum of the arrival rates, $\lambda_1, \cdots, \lambda_J$ and let $n(\lambda)$ denote a normalizer of the type considered in Section 4. Assume that the heavy traffic limit

$$g(c) = \lim_{\lambda \to c} (c - \lambda)\, f(\lambda)\, n(\lambda)$$

is known (exactly).

Let us suppose that for $\lambda_{min} \le \lambda \le c$,

$$\begin{aligned} g(\lambda) &= f(\lambda_j)\,(c - \lambda)\, n(\lambda) \\ &\approx \sum_{k=0}^{d} b_k\, \lambda^k = g^{(d)}(\lambda), \qquad (6.2) \end{aligned}$$

for some d and coefficients, $b_0, \cdots, b_d$. In other words, we assume that *the normalized* $f(\lambda)$ *can be approximated by a some polynomial of order* d over the interval $[\lambda_{\min}, c]$. The problem of characterizing the normalized $f(\lambda)$ is now one of determining the order d and coefficients $b_0, \cdots, b_d$ of the approximating polynomial. For $j = 1, \cdots, J$, form

$$\hat{g}(\lambda_j) = \hat{f}(\lambda_j)\,(c - \lambda)\, n(\lambda),$$

and

$$\hat{se}[\hat{g}(\lambda_j)] = \hat{se}[\hat{f}(\lambda_j)]\,(c - \lambda)\, n(\lambda).$$

Pretenting that d is known, for the moment, a natural approach to determining the coefficients of the polynomial is to fit the right hand side of (6.2) to the statistics $\hat{g}(\lambda_1), \cdots, \hat{g}(\lambda_J)$ subject to the constraint that $g^{(d)}(c) = g(c)$. The approximation in (6.1) strongly suggests (see, for example, Lewis and Odell [1971]) that we fit the polynomial using a constrained, weighted least squares procedure: Let $\hat{\boldsymbol{g}}$ denote the vector of length J with elements $\hat{g}(\lambda_1), \cdots, \hat{g}(\lambda_J)$, and denote by $\boldsymbol{b}$, the vector of length $(d+1)$ with elements $b_0, \cdots, b_d$. Also, let V denote the $J \times J$ diagonal matrix with diagonal elements v_{jj} given by

$$v_{jj} = \left(\hat{se}[\hat{g}(\lambda_j)]\right)^2,$$

$j = 1, \cdots, J$, and

$$\Lambda = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ \lambda_1 & \lambda_2 & \cdots & \lambda_J \\ \lambda_1^2 & \lambda_2^2 & \cdots & \lambda_J^2 \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ \lambda_1^d & \lambda_2^d & \cdots & \lambda_J^d \end{pmatrix} .$$

Note that V is an estimate of the covariance matrix of $\hat{g}$. The constrained least squares estimate of the coefficients vector, $\boldsymbol{b}$, is the vector (call it $\hat{\boldsymbol{b}}$) which minimizes

$$(\hat{g} - \Lambda'\boldsymbol{b})'V^{-1}(\hat{g} - \Lambda'\boldsymbol{b}) \qquad (6.3)$$

subject to $\boldsymbol{c}'\boldsymbol{b} = g(c)$, where $\boldsymbol{c} = (\ 1\ c\ c^2 \cdots c^d\ )'$ and where $'$ denotes matrix transposition. Assuming Λ is of full-rank, $J \geq d+1$, and the v_{jj} are positive, the constrained minimum of (6.3) is at (see, for example, Section 6.3 of Lewis and Odell [1971])

$$\hat{\boldsymbol{b}} = \tilde{\boldsymbol{b}} + S^{-1}\boldsymbol{c}[\boldsymbol{c}'S^{-1}\boldsymbol{c}]^{-1}[g(c) - \boldsymbol{c}'\boldsymbol{b}]$$

where

$$\tilde{\boldsymbol{b}} = S^{-1}\Lambda V^{-1}\hat{g}$$

and

$$S = \Lambda V^{-1}\Lambda' .$$

Note that $\tilde{\boldsymbol{b}}$ is the unconstrained, weighted least squares estimate of $\boldsymbol{b}$. The variance-covariance matrix of $\hat{\boldsymbol{b}}$ is given by

$$\text{cov}(\hat{\boldsymbol{b}}) = S^{-1} - S^{-1}\boldsymbol{c}[\boldsymbol{c}'S\boldsymbol{c}]^{-1}\boldsymbol{c}'S^{-1}$$

Note that the first term on the right directly above is the variance-covariance matrix of $\tilde{\boldsymbol{b}}$, so that the added information contained in the constraint, namely $\boldsymbol{c}'\boldsymbol{b} = g(c)$, leads to a reduction in the variance of the estimate of $\boldsymbol{b}$ over the unconstrained estimate.

Now, for an arbitrary $\lambda, \lambda_{min} \leq \lambda \leq c$, we estimate $g^{(d)}(\lambda)$ by

$$\hat{g}^{(d)}(\lambda) = \sum_{k=0}^{d} \hat{b}_k \cdot \lambda^k = \boldsymbol{\lambda}'\hat{\boldsymbol{b}} .$$

where $\boldsymbol{\lambda} = (\ 1\ \lambda\ \lambda^2\ \cdots\ \lambda^d\ )'$. Renormalizing, we estimate

$$f^{(d)}(\lambda) = \frac{g^{(d)}(\lambda)}{[(c - \lambda)\, n(\lambda)]}$$

by

$$\hat{f}^{(d)}(\lambda) = \frac{\hat{g}^{(d)}(\lambda)}{[(c - \lambda)\, n(\lambda)]}$$

If the approximation given in (6.1) is reasonable, we expect the approximation

$$\frac{\hat{f}^{(d)}(\lambda) - f^{(d)}(\lambda)}{\hat{se}[\hat{f}^{(d)}(\lambda)]} \mathrel{\dot{\sim}} \text{N}(0,1) , \qquad (6.4)$$

should also be reasonable, with

$$\hat{se}[\hat{f}^{(d)}(\lambda)] = \frac{\sqrt{\boldsymbol{\lambda}'\,\text{cov}(\hat{\boldsymbol{b}})\,\boldsymbol{\lambda}}}{[(c - \lambda)\, n(\lambda)]} ,$$

(see, for example, Section 3b of Rao [1973]).

Above, we have been assuming that a value of d which provides a "good" approximation in (6.2) is known. Note that, except for a few very simple systems, this is generally not the case. However, the approximations given in (6.1) and (6.4) provide a means of partially checking the adequacy of the constrained fit of a polynomial of degree d to the normalized quantile estimates and the heavy traffic limit point. The problem of determining the order of the polynomial can therefore be approached empirically. For example, a sensible procedure is to choose the order of the approximation to be the smallest d, such that the fitted polynomial is "reasonablely" close (relative to sampling fluctuations) to the normalized quantile estimates.

As noted in Section 4, it is not always possible to compute $g(c)$ exactly. However, an estimate $\hat{g}(c)$ of $g(c)$ can generally be obtained from a light traffic simulation experiment. The extension of the above interpolation methodology to the case where $g(c)$ is estimated via simulation will be discussed in a forthcoming paper (see Simon and Willie [1986]).

7. An Example

The queueing network considered in this section has two servers, a CPU and a disk. Four different types of customers arrive at the system as independent Poisson processes with arrival rates in set ratios, and there are four levels of priorities at the CPU. Here, the overall arrival rate, λ, is the sum of the four arrival rates of the processes corresponding to the four customer types. Customers of type 1 have the highest priority at the CPU and require 4.0 milliseconds of CPU time before departing from the system. Type 2 customers require a random amount of service time at the CPU from a particular 10-point distribution: The service time is assigned the value 10.0, 20.0, 30.0, 40.0, 50.0, 60.0, 70.0, 80.0, 90.0, and 100.0 milliseconds with probability 0.23, 0.23, 0.23, 0.23, 0.03, 0.01, 0.01, 0.01, 0.01, and 0.01, respectively. Type 2 customers have second highest priority at the CPU and after receiving service at the CPU, they depart from the system. Customers of type 3 are similar to those of type 1 except that type 3 customers require only 1.0 milliseconds of CPU service and have only a relative priority level of three. A customer of type 4 first visits the CPU upon arrival to the system. After receiving 50.0 milliseconds of service at the CPU, the customer goes to the disk and requires a random amount of service time that is uniformly distributed over the interval (30.0,70.0]. The customer then returns to the CPU. This sequence of steps is replicated a total of 12 times and then the customer departs from the system. Type 4 customers have the lowest priority at the CPU.

The capacity of the system, c, is easily calculated as to be 0.0820 total arrivals per millisecond. The arrival rates of customers of type 1, 2, 3, and 4 are $0.47330 \cdot \lambda$, $0.20930 \cdot \lambda$, $0.31120 \cdot \lambda$, and $0.00620 \cdot \lambda$, respectively.

In this example, we are interested in describing the 0.99^{th} quantile of the "steady state" response time distribution associated with the first two steps of a type 4 customer (i.e., the first CPU to disk). In particular, we desire a characterization of this quantile as a function of overall arrival rate. Here, $f(\lambda)$ denotes this function.

For the above network, discrete event simulation experiments were performed for overall arrival rates of $\lambda_1 = 0.0410$, $\lambda_2 = 0.0492$, $\lambda_3 = 0.0573$, and $\lambda_4 = 0.0655$ total arrivals per millisecond. These arrival rates correspond to traffic intensities of 0.50, 0.60, 0.70 and 0.80, respectively. Based on the regenerative technique, estimates $\hat{f}(\lambda_j)$ and $\hat{se}[\hat{f}(\lambda_j)]$, $j = 1, \cdots, 4$, were constructed from the appropriate sequences of simulation output data. The methodology employed was a two-stage extension of the methodology described in Iglehart [1976]; see Willie [1986]. Results of another study (again see Willie [1986]) suggest that for the $\hat{f}(\lambda_j)$ and $\hat{se}[\hat{f}(\lambda_j)]$ in this example, the approximation (3.1) is quite reasonable. Alternative estimation methodologies are developed in Heidelberger and Lewis [1984].

Computing the heavy traffic limit for our system is a straightforward application of the material in Reiman [1985], or Simon [1985]. If W is the response time of the first CPU to disk for a type 4 customer, then

$$\lim_{\lambda \to c} P\left(W > \frac{t}{c-\lambda}\right) \approx e^{-0.239\,t} .$$

Thus, we have

$$\lim_{\lambda \to c} (c-\lambda) f(\lambda) \approx \frac{ln(0.01)}{0.239} \approx 19.3 .$$

Since there are customers at the CPU with priority higher than customers of type 4 and since a customer of type 4 goes to the disk, we normalize our system with two extra terms,

$$n(\lambda) = (c_H - \lambda)(c_2 - \lambda) ,$$

where $c_H \approx 0.118$ (the capacity at the CPU as seen by customers with higher priority than transaction 4), and $c_2 \approx 0.268$ (the capacity at the disk). Thus, we set

$$g(\lambda) = (c-\lambda) f(\lambda)\, n(\lambda).$$

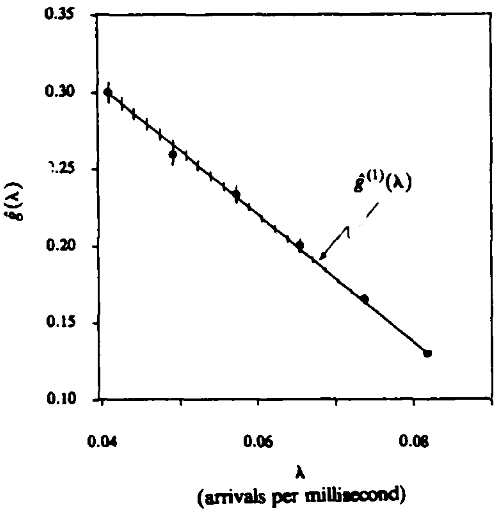
The heavy traffic limit of the fully normalized system is

$$\lim_{\lambda \to c} g(\lambda) \approx 0.129 .$$

In Figure 7.1, we present the approximation of the normalized 0.99^{th} quantile of the response time distribution of interest as a function of the overall rate of arrival of customers to the queueing network.

FIGURE 7.1

Approximation of the Normalized 0.99^{th} Quantile Function



The points

$$\left(\lambda_j, \hat{g}(\lambda_j)\right) , \; j = 1, \cdots 4, \text{ and } \left(c, \hat{g}(\lambda_j)\right)$$

are displayed with solid dots in Figure 7.1. The verticle bars eminating from the dots extend to $\hat{g}(\lambda_j) \pm \hat{se}[\hat{g}(\lambda_j)]$ The character of the points suggests approximating the normalized quantile function over $[\lambda_{min}, c]$ by a straight line: The line in the figure is the (linear) approximation $\hat{g}^{(1)}(\lambda)$, constructed in the manner described in Section 6.

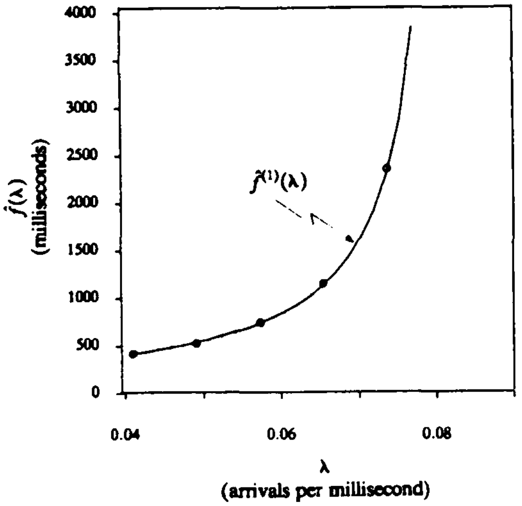
The verticle bars about the line extend to $\hat{g}^{(1)}(\lambda_{j'}) \pm \hat{se}[\hat{g}^{(1)}(\lambda_{j'})]$ for a selection of $\lambda_{j'}$ in the interval $[\lambda_{\min}, c]$. For a particular arrival rate $\lambda_{j'}$, say, the latter verticle bar is a measure of the typical error in $\hat{g}^{(1)}(\lambda_{j'})$ resulting from the sampling errors in the simulation points used in the interpolation. Note, this should not be confused with the error in $\hat{g}^{(1)}(\lambda_{j'})$ resulting from approximating the unknown $g(\lambda_{j'})$ by a linear function. The point in Figure 7.1 displayed with a circle is a normalized quantile estimate from an additional simulation experiment. This point was not used in the construction of the interpolation line.

The approximation $\hat{g}^{(1)}(\lambda)$ appears to be a very reasonable description of $g(\lambda)$ over the entire interval $[\lambda_{\min}, c]$.

Our approximation of the 0.99th quantile function, $\hat{f}^{(1)}(\lambda)$ is displayed in Figure 7.2. The points and bars in Figure 7.2 were obtained by renormalizing the corresponding points and bars in Figure 7.1.

FIGURE 7.2

Approximation of the 0.99th Quantile Function



8. References

Blomqvist, N. [1967]. The covariance function of the M/G/1 queueing system. *Skand. Akt. Tidskr.* **50**, 157-174.

Fishman, G. S. [1978]. *Principles of Discrete Event Simulation*. Wiley, New York.

Heidelberger, Philip and Lavenberg, Stephen S. [1984]. Computer performance evaluation methodology. *IEEE Transactions on Computers*, **C-33:12**, 1195-1219.

Heidelberger, P. and Lewis, P. A. W. [1984]. Quantile estimation in dependent sequences. *Operations Research* **32**, 185-209.

Iglehart, D. L. [1976]. Simulating stable stochastic systems: IV. Quantile estimation. *Journal of the Association for Computing Machinery*, **23:2**, 347-360.

Iglehart, D. L. and Shedler, G. S. [1980]. *Regenerative Simulation of Response Times in Networks of Queues*. Lecture Notes in Control and Information Sciences, Springer-Verlag, New York.

Lewis, T. O. and Odell, P. L. [1971]. *Estimation in Linear Models*. Prentice Hall, Inc., Englewood Cliffs, New Jersey.

Reiman, M.I., [1985]. A multi-class feedback queue in heavy traffic. Unpublished document.

Reiman M. I. and B. Simon [1985]. Open queueing systems in light traffic. Unpublished document.

Reiman M. I. and B. Simon [1985]. An interpolation approximation for queueing systems with Poisson input. To appear in *Operations Research*.

Rao, C. Radhakrishna [1973]. *Linear Statistical Inference and Its Applications*. John Wiley & Sons, New York.

Simon B. and J. S. Willie [1986]. Estimating heavy traffic limits from a light traffic simulation. In preparation.

Simon, B., [1985], Computing heavy traffic limits for networks of priority queues with nested markov routing. Unpublished document.

Welch, Peter D. [1983]. The statistical analysis of simulation results. In *Computer Performance Modeling Handbook*, ed. Stephen S. Lavenberg. Academic Press, New York.

Willie, J. S. [1986]. QSIM: A performance analysis and prediction tool for queueing network models. To be submitted to *AT&T Technical Journal*.

SEQUENTIAL SIMULATION RUN CONTROL USING STANDARDIZED TIME SERIES

Lee Schruben, Cornell University

In this paper the basic concepts of standardized time series analysis are presented. A detailed sequential confidence interval estimation procedure is presented based on standardized time series.

1. MOTIVATION AND BACKGROUND:

The motivational material presented in this section is expanded on in [Schruben, 1985]. Standardizing a time series is similar to the familiar procedure of standardizing or normalizing a scalar statistic. Standardizing a scalar statistic, such as a sample mean, involves centering the statistic to have a zero mean and scaling its magnitude to generic units of measurement called standard deviations. Limit theorems can be applied that give us the asymptotic (large sample) probabilistic behavior of correctly standardized statistics under certain hypotheses. This limiting model for scalar statistic is typically the Standard Normal probability distribution. This model can be used for statistical inference such as testing hypotheses or constructing confidence intervals. Here we extend this concept to the standardization of an entire time series.

The value of standardizing time series comes from the fact that the same mathematical analysis can be applied to series from a variety of sources. Thus the technique of standardization serves as a mathematical surrogate for experience with the data under study. No matter what the original time series looks like, the standardized time series will be familiar if certain hypotheses are correct. Unusual appearance of a standardized time series can be used to conclude that these hypotheses are not valid. The statistical significance of these conclusions can be computed in the same manner as with standardized scalar statistics.

1.1 Standardizing a Scalar Statistic:

As a guide to standardizing a time series, we review the procedure of standardizing a scalar statistic. We will use the familiar t-statistic as an example. The data will consist of n observations, Y_1, Y_2, ..., Y_n, that are independent and have identical distributions. We wish to make inferences about the unknown population mean μ. The sample average of the data, $\bar{\bar{Y}}_n$, will be the statistic used for these inferences. The population variance, σ^2, is an unknown nuisance parameter.

Standardization involves the following steps.

(STEP 1) CENTER THE STATISTIC: the population mean is subtracted from the sample mean giving the random variable. $\bar{\bar{Y}}_n - \mu$ which has an expected value of zero.

(STEP 2) SCALE THE STATISTIC MAGNITUDE: express the statistic in a common unit of measurement called a standard deviation. The magnitude of the statistic is scaled by dividing by $\sigma/\sqrt{n}$. Our statistic is now

$$Z_n = (\bar{\bar{Y}}_n - \mu)/(\sigma/\sqrt{n})$$

which is our standardized statistic. Standardized sample means will all have the same first two moments. The unknown scaling parameter, σ, can either be estimated or it can be cancelled out of a ratio statistic. The paper in the references by Glynn and Iglehart discuss these two alternatives from a theoretical viewpoint. The cancelling out of this parameter in a ratio statistic is the more common approach and this is followed here.

(STEP 3) CANCEL THE SCALE PARAMETER: the data is aggregated or batched into b exclusive adjacent groups of size m (assume b = [n/m]). The average of each batch is denoted as, $\bar{Y}_{i,m}$, i = 1,2,...,b. The usual unbiased estimator of the variance of the batched means is

$$S^2 = (n-1)^{-1} \sum_{i=1}^{b} (\bar{Y}_{i,m} - \bar{\bar{Y}}_n)^2.$$

Inferences about the parameter, μ, are based on the random ratio,

$$\begin{aligned} T_{b-1} &= ((\bar{\bar{Y}}_n - \mu)/(\sigma/\sqrt{n}))/\sqrt{(((b-1)S^2)/(\sigma^2(b-1)))} \\ &= (\bar{\bar{Y}}_n - \mu)/(S/\sqrt{n}). \end{aligned}$$

The parameter, σ, cancels out of this ratio.

(STEP 4) APPLY LIMIT THEOREMS: The limiting distribution of T_{b-1} is known. As $n \to \infty$ (making $m \to \infty$ since b is fixed) the distribution function of $(b-1)S^2/\sigma^2$ converges to that of a χ^2 random variable with b-1 degrees of freedom. Also as $n \to \infty$ $\bar{\bar{Y}}_n$ will converge to the constant μ and the distribution function of Z_n will converge to that of a standard Normal random variable. Thus the distribution function of T_{b-1} (being a continuous mapping) will converge to that of a t random variable with b-1 degrees of freedom.

(STEP 5) USE THE LIMITING PROBABILITY MODEL FOR INFERENCE: The limiting distribution of T_{b-1} can be used for statistical inference and estimation.

1.2. Standardizing a Time Series:

The concept of standardization can be applied to an entire time series. The original series of observations is transformed into s standardized series of observations. We will hypothesize (and test) that the series is stationary. We also assume that there is some minimal amount of randomness in the process; however, we do not assume that the data is independent. The mathematical assumptions needed are given in [Schruben, 1983] where it is argued that simulations on a computer will meet the imposed restrictions for applicability. Let $(Y_i,\ i = 1,\dots,n)$ denote the time series. We will standardize the sequence of cumulative means up to and including the k^{th} observations, given by,

$$\bar{Y}_{1,k} = (1/k) \sum_{i=1}^{k} Y_i$$

Similar steps to those in standardizing a scalar statistic are followed in standardizing a time series. These steps are as follows.

(STEP 1) CENTER THE SERIES: The sequence given by

$$S_m(k) = \bar{Y}_{1,m} - \bar{Y}_{1,k}$$

will have a mean of zero.

(STEP 2) SCALE THE SERIES MAGNITUDE: The scaling constant for dependent sequences that we use is defined as

$$\sigma^2 = \lim_{m\to\infty} m \operatorname{Var}(\bar{Y}_{1,m})$$

which is just the population variance in the special case of independent identically distributed data. Magnitude scaling is done by dividing $S_m(k)$ by $\sqrt{m}(\sigma)/k$. Again the scaling constant is unknown but will cancel out of our statistics as before.

Now there is one step required that was not necessary in the scalar standardization case. Different time series can be of different length so we must also scale the index of the series. Thus we have the additional step:

(STEP [illegible]) SCALE THE SERIES INDEX: We will define the continuous index, $t = k/m$. Our previous index is thus given by $k = [mt]$. We also add the starting point $S_0 = 0$ so that $0 \le t \le 1$. The result is that all standardized time series have indices on the unit interval.

We now have what we will call a standardized time series given by

$$T_m(t) = ([mt]) S_m([mt]) / (\sqrt{m}(\sigma)).$$

(STEP 3) CANCEL THE SCALE PARAMETER: There are several functions that might be considered for the denominator of of a ratio that cancels σ (see Schruben, 1983). We will consider here only one such function, the sum (or limiting area under the function $T_m(t)$)

$$A = \sum_{k=1}^{m} T(k/m).$$

(STEP 4) APPLY LIMIT THEOREMS: It is shown in [Schruben, 1983] that the standardized series, $T_m(t)$, will converge in probability distribution to that of a Brownian Bridge stochastic process. Thus the Brownian Bridge process plays the role in time series standardization that the normal random variable played in the scalar standardization. An the important feature of the standardized series, $T_m(t)$, is that it is constructed to be asymptotically independent of the sample mean, $\bar{Y}_{1,m}$.

There are several functions of $T_m(t)$ that will also be asymptotically χ^2 distributed. The area, A, will have a limiting normal distribution with zero mean and variance $V =$ [illegible]$(m^3 - m)$. Therefore, A^2/V will have a limiting χ^2 distribution with one degree of freedom.

Now consider where each of b independent replications (or b batches of data) are standardized in the manner above. We can then add the resulting χ^2 random variables, A^2/V, for each replication or batch to obtain a χ^2 random variable with b degrees of freedom. Also each of the replication or batch means can be treated as a set of scalar random variables and standardized giving another χ^2 random variable $(b-1)S^2/\sigma^2$ (given above). Due to the independence of $T_m(t)$ and the $\bar{Y}_{1,m}$'s, these two χ^2 random variables can be added giving a χ^2 random variable with 2b-1 degrees of freedom. This can be considered as a "pooled" estimator of σ^2 which we will denote as Q^2.

(STEP 5) USE THE LIMITING PROBABILITY MODEL FOR INFERENCE: Exactly like for the scalar case, the standardized (scalar) sample mean of all of the data can be divided by the square root of Q^2 over 2b-1 to form a ratio (independent of the scale parameter σ). For large values of m the distribution of this ratio can be accurately modeled as having a t distribution with 2b-1 degrees of freedom. The same types of inferences can be made for the dependent time simulation output series as were applicable in the independent data case. The resulting "t variate" is given by,

$$T_{b-1} = (\bar{\bar{Y}}_n - \mu)/(Q/\sqrt{n}).$$

Theoretical properties of confidence intervals formed using standardized time series are presented in [Goldsman and Schruben, 1984]. This paper compares the standardized time series approach to conventional methods.

2. APPLICATIONS OF STANDARDIZED TIME SERIES:

Standardized time series has been implemented in several simulation analysis packages. Most notably at IBM [Heidelberger and Welch, 1983], at Bell Labs [Nozari, 1985], and at G.E. [Schruben, 1986]. These packages typically control initialization bias (see also [Schruben, Singh and Tierney, 1983] and [Schruben, 1982]) and run duration as well as produce confidence intervals. Other applications of standardized time series have been to selection and ranking problems [Goldsman, 1983] and simulation model validation [Chen and Sargent, 1984].

3. AN ALGORITHM FOR SEQUENTIAL SIMULATION RUN CONTROL

The objective of the simulation run is to estimate the mean, μ, of the output series. The algorithm is based on standardized time series techniques and is implemented as a procedure or subroutine that can be used with any simulation model that generates a series of output observations. This procedure is called periodically during a simulation run to test if either of two run termination criteria are met. The run is stopped when a maximum run length is reached or an estimator relative precision criterion is satisfied. The procedure interrupts the run at various check points, truncates the output if a significant initialization bias is detected, computes a confidence interval estimate, and terminates the run if appropriate.

The test for significant initialization bias is the weighted-sum test for initialization bias [Schruben, Singh, Tierney, 1983]. Confidence interval estimates are computed using the combined classical-area confidence interval estimator [Schruben, 1983]. The output sequence is arbitrarily broken into 5 batches which gives 9 degrees-of-freedom for computing the confidence interval. This is felt to be a sufficient number of degrees of freedom. [Schmeiser, 1982]. The algorithm uses the sequential structure in Heidelberger and Welch (1983).

Run Control Procedure:

The experimenter selects the following input parameters for run control

α = confidence coefficient for the confidence intervals.
ε = acceptable estimator relative precision (i.e. H/C < ε), and
n_{max} = maximum number of observations in run

The user inputs α, ε, and n_{max} and initially sets the run length to $n = .1n_{max}$. The run control procedure is called after a total of n observations are generated by the simulation program. The procedure either terminates the run with an acceptably precise confidence interval estimate for μ or updates the run length, n. If the value of n returned exceeds n_{max}, the run is to be terminated with a message that the relative precision criterion was not met in the allocated maximum run duration.

The run control procedure is as follows:

```
STEP 1. Compute a truncation point, $n_0$.

STEP 2. If $n_0 > n/2$,
            then, set n = 1.5n and RETURN;
            otherwise, compute a α level
            confidence interval center point, C,
            and halfwidth, H, with the truncated
            output sequence (discard
            observations up to and including the
            $n_0$ th

STEP 3: If H/C < ε,
            then, report a confidence interval
                with H ± C and STOP;
            otherwise, set n = 1.5n and RETURN.
```

Truncation point selection (STEP 1):

Compute the "classical-sum" estimate of σ^2 using only the last half of the output and b=5 batches. Let $Y_{i,j}$ again be the jth observation and let $\bar{Y}_i$ be the sample mean in batch i. The batch size is m. That is, compute

$$A = \sum_{i=1}^{b} (12\, A_i^2/(m^3 - m) + m(\bar{Y}_i - \bar{\bar{Y}})^2$$

with

$$A_i = \sum_{k=1}^{m} \sum_{j=1}^{k} (\bar{Y}_i - Y_{i,j})$$

and

$\bar{\bar{Y}}$ = the average of all the retained data

Note that the expected value of $A/(2b-1)$ is σ^2. It is important to implement the computation of A so that there is no numerical overflow.

The truncation point is selected using a recursion that is equivalent to computing the weighted-sum initialization bias test statistic. This test statistic is computed starting at the end of the output sequence and moving toward the beginning of the run. The output is not batched. Let Y_j denote the output indexed in reverse order from that in which it was generated. The recursion starts with T(1) = 0 and is as follows:

$$T(j+1) = T(j)+((2j+1)/6)(jY_{j+1}-SUM_j), \quad j=1,\dots,n$$

with

$$SUM_j = \sum_{i=1}^{j} Y_i = \text{sum of the last j observations}$$

The truncation point n_0 is given by $n - \hat{j}$ where $\hat{j}$ is smallest index where for all $j > \hat{j}$

$$|(45/(j^5 - j))^{1/2}\, T(j)| > (A/(2b-1))^{1/2}\, t_{\alpha,2b-1}$$

That is, truncate all observations where T(j) goes outside it's α confidence limit and stays there.

Confidence interval estimation (STEP 2):

Using the truncated output sequence, compute A using the formula given above. The 1 - α confidence interval estimate is

$$C = b^{-1} \sum_{i=1}^{b} \bar{Y}_i$$

$$H = t_{\alpha,2b-1}\,(A/(2b-1)b)^{1/2}$$

CLOSING REMARKS

1) The sequential simulation run control procedure presented in section 3 has been

enriched and implemented for a microcomputer environment at the General Electric Corporate Research and Development Laboratories in Schenectady, N.Y. That procedure has the novel feature of controlling "run duration" by real user time. This is the critical resource (rather than number of observations or the CPU time) for a microcomputer simulation environment. A paper on this implementation is currently under preparation by Dr. Ralph Duersch of G.E. and this author.

2) The asymptotic arguments above require that the <u>batch size</u>, m, become large as n is increased. The common method to allow the batch size to grow as the sample size increases is to fix the number of batches, b. Fixing b at say 10 or 20 seems to be a reasonable thing to do in most applications as long as the sample size is large (see [Schmeiser, 1983]).

3) The limiting probability model of a t random variable has as its highest (and perhaps only) virtue the fact that it is widely tabled and has been studied extensively. There certainly exist other limiting models that might be used but none have been developed to the extent of the t model. The point is that there are alternative ways of performing each step in any standardization procedure. We wish to emphasize the importance of the concept of standardization and not the mechanics of the particular implementation presented here.

References:

<u>Chen, B., and R. G. Sargent, 1984</u>, "Confidence Interval Estimation for the Difference Between Two Stochastic Processes Using Standardized Time Series" Working paper 84-004, Dept. of IE and OR, Syracuse University.

<u>Glynn, P., and D. Iglehart, 1985</u>, "The Theory of Standardized Time Series" (forthcoming).

<u>Goldsman, D., 1983</u> "Ranking and Selection in Simulation", Proceedings of the 1983 Winter Simulation Conference, pp. 387-393.

<u>Goldsman, D., and L. Schruben, 1984</u>, "Asymptotic Properties of Some Confidence Interval Estimators for Simulation Output", <u>Management Science</u>, Vol. 30, pp. 1217-1225.

<u>Heidelberger, P., and P. Welch, 1983</u>, "Simulation Run Length Control in the Presence of an Initial Transient", <u>Operations Research</u>, Vol. 31, pp. 1109-1144.

<u>Nozari, A., 1985</u>, "Confidence Intervals Based on Steady-State Continuous-Time Statistics" <u>Operations Research Letters</u> Vol.4 No.5, pp. 213-220.

<u>Schmeiser, B. W., 1983</u>, "Batch Size Effects in the Analysis of Simulation Output", Operations Research Vol. 30, pp.565 568.

<u>Schruben, L., 1982</u>, "Detecting Initialization Bias in Simulation Output", Operations Research Vol. 30, pp.569-590.

<u>Schruben, L., 1983</u>, "Confidence Interval Estimation Using Standardized Time Series", <u>Operations Research</u>, Vol. 31, pp. 1090-1107.

<u>Schruben, L., W. Singh, and L. Tierney, 1983</u>, "Optimal Tests for Initialization Bias in Simulation Output", <u>Operations Research</u>, Vol. 31, pp. 1167-1177.

<u>Schruben, L., 1985</u>, "Standardized Time Series: An Overview" <u>Proceedings of the 1985 Winter Simulation Conference</u>, San Fransisco, CA

ON SOME STATISTICAL ISSUES IN SIMULATION STUDIES

C. L. Mallows and V. N. Nair, AT&T Bell Laboratories

1. *Introduction*

We point out some statistical techniques that have not been fully exploited in simulation studies, and show how they work in two very simple examples. The techniques we shall discuss are data-analysis, experimental design, and smoothing. Many workers in statistics have begun to move away from the use of rigid models for data towards more flexible, data-determined models. This goes along with a more algorithmic view of the process of data analysis. Often, in a simulation study, the purpose is to obtain qualitative understanding, and not (at least originally), precise estimates of model parameters. Thus these modern attitudes become relevant.

2. *A Birth-Death Process*

For our first example, consider a simple birth-death process, with state-dependent probabilities. Thus the state space is the non-negative integers, and for $k=1,2,...$

$$P\{X(t+1)=k+1\,|\,X(t)=k\}=p_k$$

$$=1-q_k=1-P\{X(t+1)=k-1\,|\,X(t)=k\}$$

while for $k=0$ the same is true except that q_0 is the probability that $X(t+1)=0$, given that $X(t)=0$. The problem is to estimate the stationary distribution $\{\pi_k\}$, or its moments or quantiles. We view this toy problem as a model of part of a larger system, so that we would not know the p's, though of course to simulate the system we have to choose values for these parameters; in fact we shall always choose to make them all equal to some value, p.

If we start a simulation run at $X(0)=x_0$ and run for N steps, we can compute the "histogram" estimate

$$\hat{\pi}_k^{HIST}=N_k/N$$

where N_k is the number of times the simulation visited the state k. However several other estimates are available. From the conservation equation

$$\pi_k p_k=\pi_{k+1}q_{k+1}$$

which simply says that in the long run, for every time the sample path goes up from k to $k+1$, there must be a compensating step from $k+1$ to k, we have that the stationary distribution is given by

$$\pi_k=\pi_k(p_0,p_1,\cdots)=c\,\frac{p_0p_1\cdots p_{k-1}}{q_1q_2\cdots q_k}$$

where the normalizing factor c is determined so that $\sum\pi_k=1$. Notice that this result will hold even if the transition probabilities are not constants, provided we interpret p_k as the average probability of the corresponding transition.

We define the "PHAT" estimate as

$$\hat{\pi}_k^{PHAT}=\pi_k(\hat{p}_0,\hat{p}_1,...)$$

where

$$\hat{p}_k=N_k^+/N_k$$

and N_k^+ is the number of times (out of N_k) that the sample path left k by passing to $k+1$ (and not to $k-1$). We have the simple

Lemma: If $X(N)=X(0)$, $\hat{\pi}^{HIST}=\hat{\pi}^{PHAT}$.

If $X(N)$ does not equal $X(0)$, the two estimates differ very slightly. Now we introduce some new estimates. If we know that p_k does not exceed ½, we can replace $\hat{p}_k$ by

$$\hat{p}^{TRUNC}=\min(½,\hat{p}_k)$$

and hence get a "truncated" estimate

$$\hat{\pi}_k^{TRUNC}=\pi_k(\hat{p}_0^{TRUNC},\hat{p}_1^{TRUNC},...)$$

Also we can smooth the raw $\hat{p}$'s, obtaining

$$\hat{\pi}_k^{SMOOTH}=\pi_k(\hat{p}_0^{SMOOTH},\hat{p}_1^{SMOOTH},...)$$

Finally, for calibration we can consider the maximum likelihood estimate

$$\hat{\pi}_k^{ML}=(1-\hat{\rho})\hat{\rho}^k$$

where

$$\hat{\rho}=\hat{p}/\hat{q}=N^+/N^-.$$

We ran 100 simulations, each of length $N=1000$, for several values of p. For each run, we computed five estimates of the mean position

$$\mu=p/(1-2p)=\sum k\pi_k,$$

namely $\hat{\mu}^{HIST}$, $\hat{\mu}^{TRUNC}$, $\hat{\mu}^{ML}$, and two versions of $\hat{\mu}^{SMOOTH}$, the first obtained by fitting a logistic regression to the raw $\hat{p}$'s and the second by fitting local logistic regressions with window width ½N. Thus for this last estimate, for each k we determined a window of values of k that included at least $N/4$ epochs on each side, unless fewer than that were available.

Table 1 gives the corresponding means and mean-square errors for two values of p, namely 0.35 and 0.45.

We see that for $p=.35$ the maximum likelihood estimate is by far the best, while all of the adjusted estimators do better than the crude HIST estimate. For $p=.45$ the maximum likelihood estimate has a large mean square error, due to a few cases where $\hat{\rho}$ is close to ½, while each of the other estimators has a much smaller mean square error, while being considerably biased towards small values of μ. We are searching for ways (hopefully of general utility) of reducing the bias of these adjusted estimators.

In general, we suggest that a similar strategy based on smoothing quantities a little beneath the surface of the raw simulation output may prove rewarding.

Once we think of smoothing $\hat{p}_k$ rather than the raw $\hat{\pi}_k$, we can bias the simulation to make it more efficient for estimating the p_k's. At an extreme, we can choose to make N separate one-step runs, starting at k exactly n_k times, where $\sum n_k=N$. The n_k's are at our choice, and we can choose them for efficient estimation of μ or any other interesting quantity. Notice that this is not simply importance sampling; here we can choose to put zero weight at some k's, and estimate the corresponding p_k's by smoothing. With the usual formulation of importance sampling, such a design would lead to an infinite variance. A simple calculation gives the following

Theorem: The optimal design for estimating $\mu(\pi)=\sum k\pi_k$, using the estimator $\hat{\mu}=\sum k\hat{\pi}_k^{PHAT}$, when truly $p_k=p$ for all k, is

$$\delta_k=\frac{(2k+1)\pi_k}{2\mu+1}\, k=0,1,2,...$$

The efficiency gain, relative to $\delta_k=\pi_k$, is

$$\frac{var(\hat{\mu}|\delta_k=\pi_k)}{var(\hat{\mu}|\delta_k=opt)}=2-\frac{1}{(1+2\mu)^2}.$$

In a real, complicated system, similar efficiency gains might be realized by sequential design. A simpler, more general strategy is the following. Every time the simulation gets into an "interesting" state (i.e. one that is influential with respect to the quantity of interest), we spawn several short daughter runs and hence improve the accuracy of estimation of the local parameters (the p_k's in our toy problem.) Notice that this is not simply the classical "splitting" strategy. We do not assign fractional weight to the daughter runs, but use each realized step with equal weight in estimating the local parameters.

3. *A Processor Sharing Queue*

For the second example, consider a *M/M/1 — PS* queue with Poisson arrivals at rate ρ and exponential service with mean one. Under this processor sharing queue discipline, if there are k jobs in the system, each job receives service at rate $1/k$. The problem is to obtain the equilibrium distribution of the sojourn time W. Notice from Little's law that the expected value of W under the processor sharing discipline equals that under the *FIFO* discipline, and is given by $1/(1-\rho)$. Coffman et al. (1970) derived the Laplace transform of the distribution conditioned on its required service time. From this, it is possible to obtain the variance of W. More recently, Morrison (1985) has obtained the following expression for the distribution of W.

$$P(W>t)=$$

$$2\int_0^{\pi}\frac{\exp\{-\theta[2\rho^{1/2}-(1+\rho)\cos\theta]/(1-\rho)\sin\theta\}}{(1-\rho)(1+\exp\{-\pi[2\rho^{1/2}-(1+\rho)\cos\theta]/(1-\rho)\sin\theta\})}$$

$$\times\exp\{-(1-\rho^2)t/(1+\rho-2\rho^{1/2}\cos\theta)\}\sin\theta d\theta.$$

We now describe an approach that combines data analysis with prior information about limiting cases to obtain a reasonable approximation to this distribution.

Let k be the queue length when a tagged job joins the queue. We shall examine the conditional distribution of W given k. We can obtain the unconditional distribution easily from this since the queue length distribution is geometric with parameter ρ (the same as under the *FIFO* queue discipline). Notice that as $\rho\rightarrow 0$, the conditional distribution of W_k tends to the mixture distribution with density

$$f_{W_k}(w)=[\exp(-w)+w\exp(-w)+\cdots+w^k\exp(-w)/k!]/(k+1).$$

This follows since for small ρ there are essentially no arrivals, and with probability $1/(k+1)$ the tagged job will be the jth one to receive service and so the sojourn time will be the sum of j exponential random variables, $j=1,...,k+1$. For $\rho\rightarrow 1$, it can be shown, and a heuristic argument can be used to con:.ince the reader, that W_k/k tends in distribution to an exponential random variable.

We simulated 100 values of W_k for values of k ranging from 0 to 20 and for ρ from .1 to .9. We simulated the process with $\rho=.9$ and thinned it to get the W's for the other ρ values. This was done to induce a high correlation among the simulation results for different ρ's. A plot of the means of W_k versus k and a least-squares analysis showed that the conditional expectation of W given k can be well approximated by a linear function in k. By plotting the least squares coefficients as a function of ρ and by using the results for the limiting cases, we arrived at

$$E(W_k)=(k+2)/(2-\rho),$$

which, in fact, is the correct answer (Coffman et al., 1970). A plot of the standard deviations of W_k versus k showed that the a linear approximation is reasonable except, perhaps, for k near 0. But this suggested that it may be better to consider the distribution of $\log(W_k)$ for which the variance will be independent of k. Analysis of the means and variances of $\log(W_k)$ showed that the mean is linear in $\log(k+2)$ and the variance is constant in k, except for k near 0.

Recall that for ρ near one and for k large, W_k is approximately distributed as kX where X is an exponential random variable. Similarly, for $\rho=0$ and $k=0$, W_0 has an exponential distribution. For $\rho=0$ and k large, the mixture distribution can be approximated by $(k+1)U$ where U is a uniform(0,1) random variable. These suggested the following approximation

$$\log(W_k)\approx\log(a)+\log(b)-X_{b:b}$$

where a and b depend on ρ and k and $X_{b:b}$ is the largest order statistic from b independent exponential random variables. Table 2 gives the values of a and b for the limiting cases. By considering these limiting values and equating a and b with the expectation of W_k using the above approximation, we obtained

$$a=k+1+\rho$$

and

$$b=\frac{k+2}{(k+\rho)(1-\rho)}.$$

Figure 1 shows the quantile-quantile (Q-Q) plots of 100 simulated W's against the quantiles from the approximating distributions. The plots for $k=0,2$ and 8 and $\rho=.1,.5$ and .9 are given in Figure 1. We see that all the plots are approximately linear with slope 1 and intercept 0. There is a slight nonlinearity in the lower tail, especially for $k=0$. However, the approximation seems reasonable overall, and more extensive plots for other values of ρ and k confirmed this finding. We can now use this approximation to the distribution of W_k to easily determine quantities of interest such as quantiles which would be much harder to obtain using the expression given by Morrison (1985).

4. *Concluding Remarks*

We have demonstrated the use of some common statistical techniques through two simple examples. We believe that similar approaches hold promise in more complex systems. For example, in the processor sharing case, we are trying to find a tractable approximation to the joint distribution of (W_k,l) where l is the number of jobs that are served before the tagged job. This would enable us to incorporate *PS* nodes in complex networks. Similarly, we view the simple birth-death model in Section 2 as a node embedded in a larger system. The use of smoothing techniques and designed simulations can improve the efficiency of the estimators in such cases.

References

Coffman, E. G., Muntz, R. R., and Trotter, H. (1970) Waiting time distributions for processor-sharing systems, *J. ACM*, 17, pp. 123-130.

Morrison, J.A. (1985) Response-time distribution for a processor-sharing system, *SIAM J. Appl. Math.*, 45, pp.152-167.

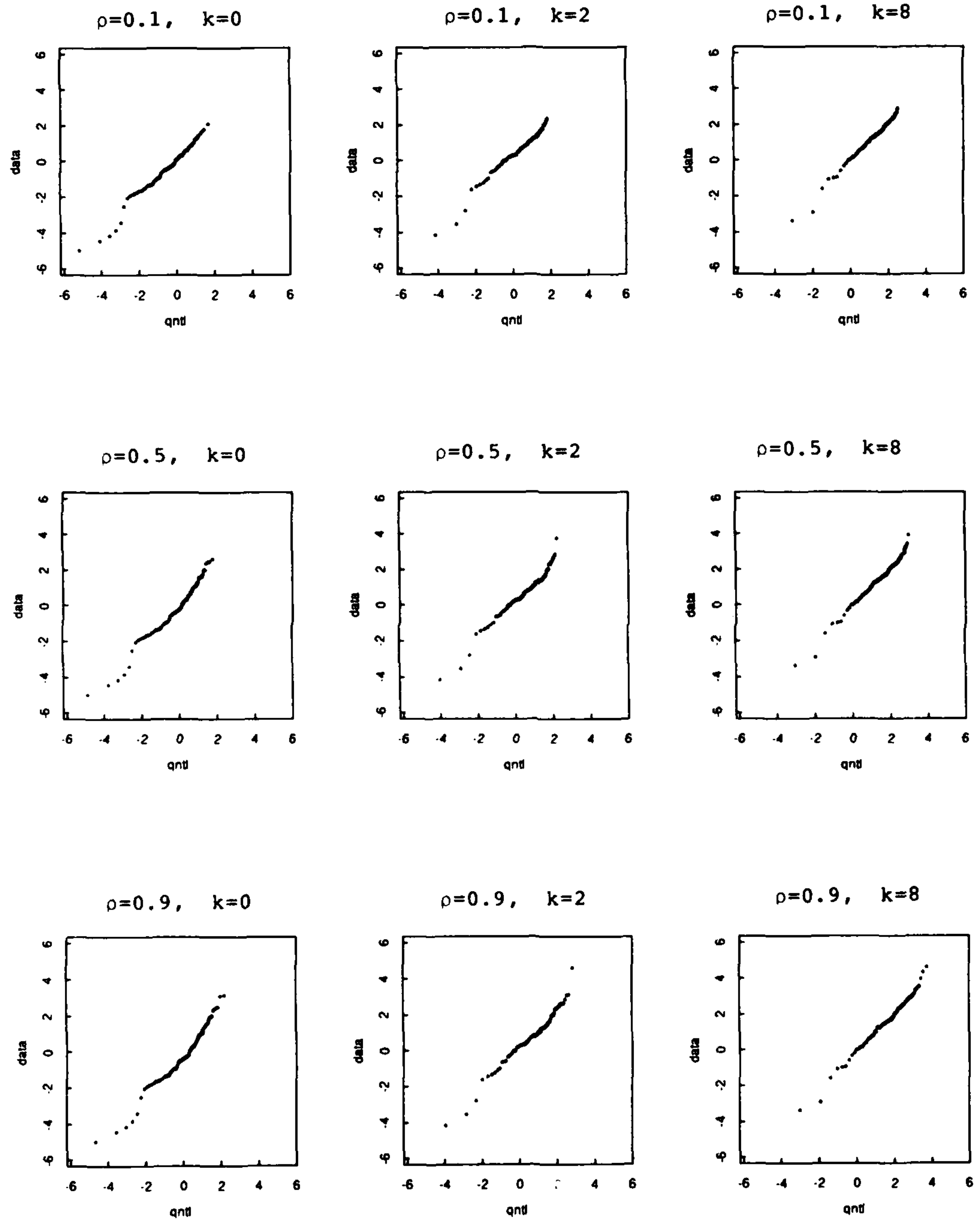
Table 1

		$\hat{\mu}^{HIST}$	$\hat{\mu}^{TRUNC}$	$\hat{\mu}^{SMOOTH}$	$\hat{\mu}^{SMOOTH,\frac{1}{2}}$	$\hat{\mu}^{ML}$
$p=.35$	mean	1.170	1.142	1.131	1.150	1.162
$\mu=1.167$	m.s.e. $\times 10^2$	5.15	3.24	3.51	4.19	1.72
$p=.45$	mean	4.07	3.36	3.83	4.07	4.86
$\mu=1.45$	m.s.e.	2.27	1.86	1.95	2.43	5.04

Table 2

	$\rho=0$	$\rho\to 1$
$k=0$	$a=1,\ b=\infty$	
k large	$a\simeq k+1,\ b=1$	$a\simeq k,\ b=\infty$

FIGURE 1: Q-Q PLOTS OF SOJOURN TIMES



ρ=0.1, k=0

ρ=0.1, k=2

ρ=0.1, k=8

ρ=0.5, k=0

ρ=0.5, k=2

ρ=0.5, k=8

ρ=0.9, k=0

ρ=0.9, k=2

ρ=0.9, k=8

FRONTIERS IN SIMULATION AND STATISTICS II

Organizer: *Dennis S. Friday, National Bureau of Standards*

Bootstrap Confidence Intervals

Robert Tibshirani, University of Toronto

BOOTSTRAP CONFIDENCE INTERVALS

Robert Tibshirani, University of Toronto

Abstract

We describe the various techniques that have been proposed for constructing non-parametric confidence intervals using the bootstrap. These include bootstrap pivotal intervals, percentile and bias-corrected percentile intervals, and non-parametric profile likelihood intervals. These methods are small sample improvements over the usual $\hat{\theta} \pm c\hat{\sigma}$ intervals. We discuss them in detail, outlining the underlying assumptions in each case. Finally, the various intervals are compared in a small simulation study.

1. Introduction.

Recently, a number of techniques have been proposed for constructing confidence intervals using the bootstrap (see Efron 1981, 1985, Schenker 1985, DiCiccio and Tibshirani 1985, 1986). These techniques are non-parametric in nature, and are designed to work well over a wide variety of situations. Because they are based on the bootstrap, they can be used in situations in which the "parameter" is an extremely complex functional of the distribution and an exact analysis would be impossible. In this paper, we describe and compare these bootstrap methods.

2. The Problem and Some Notation.

We observe $x_1, ...x_n$ assumed to be realizations of random variables $X_1, ...X_n \sim$ i.i.d F. The distribution F is unknown and the problem is to construct a confidence interval for the parameter $\theta = \theta(F)$. By a confidence interval, we mean lower and upper points $L = L(x_1, ...x_n)$ and $U = U(x_1, ...x_n)$ such that $P(L \leq \theta \leq U) = 1 - 2\alpha$, where $P(\cdot)$ denotes probability under the true distribution F. Since the intervals are to be non-parametric, we would ideally require that this hold for all F. We will confine our discussion to central intervals, i.e. intervals (L, U) such that $P(\theta \leq L) = P(\theta \geq U) = \alpha$. Non-central intervals can be obtained through obvious modification.

Given $X_1, X_2, ...X_n$, (X_i can be a scalar or vector random variable), we estimate θ by $\hat{\theta} = \theta(F_n^X)$ where F_n^X is the empirical distribution function of $X_1, ...X_n$. The observed value of $\hat{\theta}$ is $\hat{\theta}_{obs} = \theta(F_n)$ where F_n is the empirical distribution function of $x_1, ...x_n$.

We let $\boldsymbol{W}$ be a random vector with $W_i \geq 0$, $\sum_1^n W_i = 1$ and $\boldsymbol{w}$ be a realization of $\boldsymbol{W}$. Let $F_n(\boldsymbol{w})$ be the distribution putting mass w_i on x_i, $i = 1, 2, ...n$. Many of the techniques will utilize "bootstrap sampling"— that is, sampling from $x_1, x_2, ...x_n$ with replacement. This is equivalent to sampling $\boldsymbol{W}$ from the rescaled multinomial $Mult(n, \boldsymbol{w}^0)/n$, where $\boldsymbol{w}^0 = (1/n, 1, n, ...1/n)$. We'll use $\overset{*}{\sim}$ to indicate bootstrap sampling and a bootstrap value obtained in this way will be denoted by $\hat{\theta}^* = \hat{\theta}(F_n(\boldsymbol{w}))$. We'll refer to a bootstrap sample either by its weight vector $\boldsymbol{w}$, or by $X^* = (X_1^*, X_2^*, ...X_n^*)$. Finally, $\hat{B}$ will denote the empirical distribution function of $\hat{\theta}^*$ under $\overset{*}{\sim}$ ("the bootstrap distribution").

3. Overview.

Frequentist confidence intervals are usually based on a test function, say $t(X, \theta_1)$, appropriate for testing $H : \theta = \theta_1$. The interval is constructed as follows. For each trial value θ_1, we include θ_1 in our confidence interval if we would accept H in a $1 - 2\alpha$ size test based on $t(X, \theta_1)$. This procedure requires knowledge of the distribution of $t(X, \theta_1)$ for each θ_1. Usually, a simplifying assumption is made— that $t(X, \theta_1)$ is *pivotal*, that is, has a distribution not depending on θ_1. With this assumption, it is not necessary to consider each trial value θ_1 separately. We assume some parametric distribution for $t(X, \theta_1)$, then invert the pivotal to yield the confidence interval. A simple example is $X_1, X_2, ...X_n \sim \mathcal{N}(\theta, 1)$. Then a confidence interval for θ is found by inverting the pivotal $\bar{X} - \theta$, whose distribution is $\mathcal{N}(0, 1/n)$.

The *Bootstrap Pivotal, Percentile, Bias-Corrected Percentile, BC_a and BC_a^0* intervals (Sections 4 and 5) are non-parametric analogues of parametric pivotal intervals. The pivotal distribution is not assumed known; instead it is estimated non-parametrically using the bootstrap. In Sections 4 and 5 we provide the "recipes" for constructing these intervals and outline the underlying assumptions. In Section 6, we discuss the appropriateness of the various intervals in a few simple problems.

In Section 7 we describe a different approach to non-parametric confidence interval construction, through likeliihood methods.

In Section 8 we compare all the intervals in a numerical example.

4. Bootstrap Pivotal Intervals.

4.1. The Simple Pivotal

We assume that $\hat{\theta} - \theta$ is a pivotal quantity, that is

$$\hat{\theta} - \theta \sim H \qquad (A1)$$

where H is a distribution not involving θ, and also that approximately

$$\hat{\theta}^* - \hat{\theta}_{obs} \overset{.}{\sim} H \qquad (A2)$$

Assumption (A2) is based on the premise that if F_n is close to F, the bootstrap distribution of $\hat{\theta}^* - \hat{\theta}_{obs}$ will be close to that of $\hat{\theta} - \theta$, as long as $\theta(\cdot)$ is a reasonably smooth functional. Of course, if H is a continuous distribution, then (A2) is at best an approximation, since the bootstrap distribution is necessarily discrete. The intervals described in this section and the next section all use this kind of bootstrap approximation. To simplify the notation, *we will ignore the fact that it is only an approximation.*

Under (A1) and (A2), we have $1-2\alpha = P(H^{-1}(\alpha) < \hat{\theta} - \theta < H^{-1}(1-\alpha)) = P(\hat{\theta} - H^{-1}(\alpha) < \theta < \hat{\theta} - H^{-1}(1-\alpha))$.

Substituting $\hat{\theta}_{obs}$ for $\hat{\theta}$ and noting that $H^{-1}(\cdot) = \hat{B}^{-1}(\cdot) - \hat{\theta}_{obs}$, we obtain the *Bootstrap Pivotal* interval:

$$\theta \in (2\hat{\theta}_{obs} - \hat{B}^{-1}(1-\alpha), 2\hat{\theta}_{obs} - \hat{B}^{-1}(\alpha)) \qquad (1)$$

4.2. Other Pivotals

The bootstrap pivotal interval can be based on an arbitrary pivotal $t(X,\theta)$, as long as it is monotone in θ. We assume $t(X,\theta) \sim H$, $t(X^*,\hat{\theta}_{obs}) \overset{.}{\sim} H$, where $t(X,\theta)$ is monotone decreasing in θ. Inverting the pivot as above, we obtain

$$\theta \in (t_2^{-1}(H^{-1}(1-\alpha)), t_2^{-1}(H^{-1}(\alpha))) \qquad (2)$$

where $t_2^{-1}(\cdot)$=inverse of $t(\cdot,\cdot)$ with respect to the second argument.

The bootstrap pivotal interval is used by Efron (1981) in the form of a "bootstrap t" and by Schenker(1985), who calls it the "substitution method". We have introduced the obvious name "bootstrap pivotal interval" here.

4.3. The Role of Nuisance Parameters

We can think of an arbitrary distribution G as consisting of two parts, say $G = (\theta, \lambda)$, where $\theta = \theta(G)$ is the parameter of interest and $\lambda = \lambda(G)$ is a vector of nuisance parameters, possibly infinite dimensional. The true distribution can be written as $F = (\theta_{true}, \lambda_{true})$. With this decomposition, we can say more clearly the meaning of the statement "$t(X,\theta) \sim H$, H not involving θ". What we're really assuming is that F is a member of some family of distributions $\mathcal{F}$ existing in the space of possible distributions. The members of $\mathcal{F}$ correspond to different θ values and are characterized by the property $t(X,\theta) \sim H$. Because of this pivotal assumption, we don't have to know the structure of (or estimate) the entire family $\mathcal{F}$. Only a single member of $\mathcal{F}$ need be estimated. The empirical distribution function F_n estimates that member (i.e. $(F_n = (\hat{\theta}_{obs}, \hat{\lambda}_{obs}))$, and from this we obtain the distribution H. By construction, the interval will have correct coverage for $F \in \mathcal{F}$.

A family like $\mathcal{F}$ also underlies the percentile and bias-corrected percentile intervals (discussed next).

4.4. Some theory

The work of Singh (1981), Abramovitch and Singh (1985), Beran (1984) and Hartigan (1986) suggest that a bootstrap pivotal interval based on the pivot $(\hat{\theta}-\theta)/SD(\hat{\theta})$ will be accurate to $O_p(1/n)$ (under regularity conditions) for any θ. For $\theta = E(X)$, the obvious estimate for $SD(\hat{\theta}^*)$ is $\sum_1^n(x_i^* - \bar{x}^*)^2/n^2$ and Singh shows that this leads to an interval correct to $O_p(1/n)$. Unfortunately, for non-linear statistics calculation of $SD(\hat{\theta}^*)$ requires a bootstrap computation, and thus the entire procedure becomes a "double bootstrap". At the present time this procedure is too expensive computationally except for small problems.

5. Percentile Intervals.

5.1. Uncorrected Intervals

Here we assume A1 and A2, and further that

$$H \text{ is symmetric around } 0 \qquad (A3)$$

In this case, the pivotal interval (1) becomes:

$$\theta \in (\hat{B}^{-1}(\alpha), \hat{B}^{-1}(1-\alpha)) \qquad (3)$$

Efron calls this the *Percentile Interval* since it uses the percentiles of $\hat{\theta}^*$ as "percentiles" of θ.

5.2. Generalization of the Percentile Interval

If a symmetric pivotal exists on some other scale, i.e.

$$g(\hat{\theta}) - g(\theta) \sim H \qquad (A4)$$

and

$$g(\hat{\theta}^*) - g(\hat{\theta}_{obs}) \overset{.}{\sim} H \qquad (A5)$$

with H symmetric around 0 and $g(\cdot)$ is an unknown, monotone increasing function, then as in (3) we get as an interval for $g(\theta)$:

$$g(\theta) \in (\hat{G}^{-1}(\alpha), \hat{G}^{-1}(1-\alpha)) \tag{4}$$

where $\hat{G}$ is the distribution function of $g(\hat{\theta}^*)$. Transforming back to the θ scale gives

$$\theta \in (g^{-1}(\hat{G}^{-1}(\alpha), g^{-1}(\hat{G}^{-1}(1-\alpha)) \tag{5}$$

or

$$\theta \in (\hat{B}^{-1}(\alpha), \hat{B}^{-1}(1-\alpha)) \tag{6}$$

which is again the percentile interval. Thus the percentile interval has the correct coverage if a symmetric pivotal exists on any scale. Conveniently, we don't have to know $g(\cdot)$ because the resultant interval doesn't depend on $g(\cdot)$.

There is a simple connection between the bootstrap pivotal interval based on $\hat{\theta}-\theta$ and the percentile interval. Writing $(2\hat{\theta}_{obs} - \hat{B}^{-1}(1-\alpha), 2\hat{\theta}_{obs} - \hat{B}^{-1}(\alpha))$ as $(\hat{\theta}_{obs} - [\hat{B}^{-1}(1-\alpha)-\hat{\theta}_{obs}]), \hat{\theta}_{obs}+[\hat{\theta}_{obs} - \hat{B}^{-1}(\alpha)])$, we see that *the percentile interval is the bootstrap pivotal interval reflected about the point $\hat{\theta}_{obs}$.*

5.3. Bias-Corrected Percentile Intervals

5.3.1 *Normal Correction.*

If the distribution H in $A4$ and $A5$ is symmetric around $u \neq 0$, the percentile interval will be biassed and will not have the correct coverage. This would occur as a result of bias in the estimator $\hat{\theta}$. It turns out that if we are willing to assume a parametric form for H, then u can be estimated and a corrected interval can be derived. As was the case for the percentile interval, the corrected interval will not depend on the transformation $g(\cdot)$.

Since $P(g(\hat{\theta}^*) < g(\hat{\theta}_{obs})) = P(\hat{\theta}^* < \hat{\theta}_{obs})$, we can use the latter to estimate the bias. Using this correction, we then match the distributions of $g(\hat{\theta}) - g(\theta)$ and $g(\hat{\theta}^*) - g(\hat{\theta}_{obs})$ on the $g(\cdot)$ scale, then transform back to the θ scale.

As an example, suppose we choose $H = \mathcal{N}(u,1)$. Then

$$g(\theta) - g(\hat{\theta}) \sim \mathcal{N}(0,1) - u \tag{7}$$

and

$$g(\hat{\theta}^*) - g(\hat{\theta}_{obs}) \stackrel{\cdot}{\sim} \mathcal{N}(0,1) + u \tag{8}$$

We can solve for u by noting that $P(g(\hat{\theta}^*) \leq g(\hat{\theta}_{obs})) \doteq \Phi(-u) \doteq \hat{G}(g(\hat{\theta}_{obs})) \doteq \hat{B}(\hat{\theta}_{obs})$ so that $b \equiv u = -\Phi^{-1}(\hat{B}(\hat{\theta}_{obs}))$ ($\Phi(\cdot)$ denotes the cumulative distribution function of $\mathcal{N}(0,1)$) Now from (7)

$$P(g(\theta)-g(\hat{\theta}) < g(t)-g(\hat{\theta}_{obs})) = \Phi(g(t)-g(\hat{\theta}_{obs})+b) \tag{9}$$

and from (8) we obtain

$$\hat{B}(t) = P(\hat{\theta}^* \leq t) = P(g(\hat{\theta}^*) \leq g(t)) \doteq \Phi(g(t)-g(\hat{\theta}_{obs})-b) \tag{10}$$

Solving for $g(t) - g(\hat{\theta}_{obs})$ in (10) and substituting into (9) we have

$$P(g(\theta)-g(\hat{\theta}) < g(t)-g(\hat{\theta}_{obs})) = \Phi(\Phi^{-1}(\hat{B}(t))+2b) \tag{11}$$

Finally, to get a $1-2\alpha$ percent confidence interval, we set the right side of (11) equal to α and $1-\alpha$, and solve for t to obtain

$$\theta \in (\hat{B}^{-1}(\Phi(z_\alpha - 2b)), \hat{B}^{-1}(\Phi(z_{1-\alpha} - 2b))) \tag{12}$$

where z_p denotes the pth quantile of Φ. Interval (12) is called the *Bias-Corrected Percentile Interval.* The parametric assumption $\mathcal{N}(u,1)$ turns out to be not as restrictive as it appears. If we instead let $H = \mathcal{N}(u,\sigma^2)$, with σ^2 unknown, and repeat the above derivation, we get $b = \frac{u}{\sigma} = -\Phi^{-1}(\hat{B}(\hat{\theta}_{obs}))$ and we obtain the same interval (12) .

Note then when $b = 0$, the bias-corrected percentile interval reduces to the percentile interval. Hence we can think of the bias-corrected interval as a "fine-tuning" of the percentile interval.

5.3.2 *Other Symmetric Location Scale Families.*

In the bias-corrected interval above, we can just as well assume that H is some other symmetric, location scale family, say $H(x \mid u,\sigma) = H_0(\frac{x-u}{\sigma})$. This gives the bias-corrected interval

$$\theta \in (\hat{B}^{-1}(H_0(h_\alpha - 2b)), \hat{B}^{-1}(H_0(h_{1-\alpha} - 2b))) \tag{13}$$

where $b = -H_0^{-1}(\hat{B}(\hat{\theta}_{obs}))$ and h_p denotes the pth quantile of H_0.

A natural question to ask is: how much difference does the choice of H_0 make? Natural candidates to compare with the normal are symmetric, long tailed distributions. Benjamini (1983) provides an appealing definition of long-tailedness. Suppose F and G are both symmetric about the origin. Then G is said to *stretched* (or long tailed) compared to F if $G^{-1}(p)/F^{-1}(p)$ is an increasing function of p, for $1/2 < p < 1$. This definition reflects the intuitive meaning of long-tailedness, that the quantiles of G are "farther out" than those of F. Under this definition, distributions like the t, logistic and cauchy are stretched with respect to the normal, as we would expect.

Now suppose H_0 is stretched with respect to Φ. Assume $\hat{B}(\hat{\theta}_{obs}) = q > .5$, so that $\hat{\theta}_{obs}$ is biassed upward, and $b = -\Phi^{-1}(\hat{B}(\hat{\theta})) < 0$. *Then the bias correction under H_0 will be in the same direction as the bias-correction under Φ, but will be smaller.* The proof of this fact is easily derived from Benjamini's definition above. Denoting, as before, the pth quantiles of Φ and H_0 by z_p and h_p respectively, we note that $H_0(h_\alpha + 2h_q) > \alpha$. Hence

$$\frac{\Phi^{-1}(H_0(h_\alpha + 2h_q))}{H_0^{-1}(H_0(h_\alpha + 2h_q))} > \frac{\Phi^{-1}(\alpha)}{H_0^{-1}(\alpha)} = \frac{z_\alpha}{h_\alpha} \qquad (14)$$

This implies $\Phi^{-1}(H_0(h_\alpha + 2h_q)) < z_\alpha + 2z_\alpha(h_q/h_\alpha) < z_\alpha + 2z_q$. Thus $\Phi(z_\alpha + 2z_q) > H_0(h_\alpha + 2h_q) > \alpha$.

A similar argument shows that if $q < .5$, then $\Phi(z_\alpha + 2z_q) < H_0(h_\alpha + 2h_q) < \alpha$, and the corresponding results hold for the upper quantile. The above proof requires that $h_\alpha + 2h_q < 0$. This will be the case unless the bias in $\hat{\theta}_{obs}$ is so large that q is near $1 - \alpha$.

The numbers in Table 1 show the amount of bias correction (that is $(H_0(h_\alpha + 2h_q),\ H_0(h_{1-\alpha} + 2h_q))$ for the normal, logistic and the cauchy distributions, when $\alpha = .05$.

Table 1

q	Normal	Logistic	Cauchy
.40	(.015, .869)	(.023, .884)	(.045, .944)
.45	(.027, .916)	(.034, .927)	(.050, .950)
.55	(.084, .973)	(.073, .966)	(.050, .950)
.60	(.131, .985)	(.106, .977)	(.056, .955)

The choice of a *symmetric* pivotal distribution appears to make little difference. The effect of an *assymetric* pivotal distribution, however, can be large, as Example 1 will show.

5.3.3 *Another Justification for the Bias-Corrected Interval*

In place of $A4$ and $A5$, we could assume

$$h(\hat{\theta} - \theta) \sim H \qquad (A6)$$

and

$$h(\hat{\theta}^* - \hat{\theta}_{obs}) \mathrel{\dot{\sim}} H \qquad (A7)$$

with H symmetric, and h increasing and anti-symmetric $(h(-x) = -h(x))$. Letting H be a location-scale family, we again obtain the bias-corrected percentile interval (13) . When H is symmetric around 0, $\hat{\theta} - \theta$ is symmetric around 0 and the interval reduces to the percentile interval.

Finally, we could replace $h(\hat{\theta} - \theta)$ and $h(\hat{\theta}^* - \hat{\theta}_{obs})$ by $h(\hat{\theta}/\theta)$ and $h(\hat{\theta}^*/\hat{\theta}_{obs})$ respectively, with $h(1/x) = -h(x)$, and again the bias-corrected interval emerges.

5.4. The BC_a and BC_a^0 intervals

Efron (1985) proposed a further modification of the Percentile interval called the BC_a interval ("a" for acceleration). It assumes

$$g(\hat{\theta}) - g(\theta) \sim N(b(1 + g(\theta)), (1 + ag(\theta))^2) \qquad (15)$$

This generalizes the BC interval by introducing the acceleration constant "a" that allows the variance on the transformed scale to be non-constant. "a" is estimated by from a formula involving the jackknife values of $\hat{\theta}$. Efron proves that the one-parameter version of the BC_a interval is correct up to $O_p(1/n)$ under regularity conditions.

DiCiccio and Tibshirani (1985) studied the BC_a procedure and provided a method for constructing the transformation $g(.)$ in (15). The constructed $g(.)$ is a variance stabilizing transformation followed by a skewness reducing transformation. Using this $g(.)$, one can construct a confidence interval, called the BC_a^0 interval, without computing the bootstrap distribution of $\hat{\theta}^*$; through the use of an approximation for b due to Efron and T. Hesterberg, no bootstrap sampling is required, and just $n+2$ evaluations of the statistic $\hat{\theta}$ are needed.

6. Comparison Between the Bootstrap Pivotal and Percentile Intervals.

The bootstrap pivotal and percentile intervals differ in their assumptions. In constructing the bootstrap pivotal interval, we had to specify the exact form of the pivotal but we assumed nothing about its distribution. On the other hand, in building the percentile interval, knowledge of the exact form of the pivotal was not necessary but we did require that its distribution be symmetric around 0. For the bias-corrected percentile interval, we weakened that assumption to one of symmetry around any point, but we paid a price: it was necessary to specify a distribution for the pivotal.

Which of these intervals is better depends on the problem. It is helpful to look a few simple examples. In each case, the data are assumed to be Gaussian.

- *The Mean:* $\theta = E(X)$, variance known. The bootstrap pivotal interval based on $\hat{\theta} - \theta$ and the percentile interval will give very similar results, and both will have approximately the right coverage.

- *The Correlation Coefficient:* $X = (Y, Z)$ and $\theta = E(Y - E(Y))(Z - E(Z))/\{E(Y - E(Y))^2 E(Z - E(Z))^2\}^{1/2}$. The random variable $tanh^{-1}\hat{\theta} - tanh^{-1}\theta$ is approximately $\mathcal{N}(\theta/(2(n-3), 1/(n-3))$. Hence the bootstrap pivotal interval based on $t(\hat{\theta}, \theta) = tanh^{-1}\hat{\theta} - tanh^{-1}\theta$ and the bias-corrected percentile interval (using the normal family) both should work well. The uncorrected percentile interval will be biassed.

- *The Variance:* $\theta = E(X - E(X))^2$. The random variable $\hat{\theta}/\theta$ is χ^2_{n-1}, hence the bootstrap pivotal based on $t(\hat{\theta}, \theta) = \log\hat{\theta} - \log\theta$ will have approximately the right coverage. The distribution $\log\chi^2$ is not symmetric, however, so the percentile intervals may not work well (see Example 1). It is clear that a transformation to a symmetric pivotal doesn't exist here since such a transformation must also remove the dependence of the variance on θ. A simple delta method calculation shows that only $g(\hat{\theta}) = \log\hat{\theta}$ achieves this.

The above examples represent some of the problems that are well understood. In most situations, however, matters are much more difficult. To construct a bootstrap pivotal interval, we first need to specify a quantity $t(X, \theta)$ that is approximately pivotal. This alone is a difficult task unless we know something about the underlying distribution. Now suppose we are able to specify a pivotal $t(X, \theta)$. Then if $t(X, \theta) \sim H$ and $t(X^*, \hat{\theta}_{obs}) \sim {}^* H$, the resulting interval will have the correct coverage. In some problems, however, the bootstrap distribution of $t(X^*, \hat{\theta}_{obs})$ can be a poor approximation to H. One such example is the following. Consider the situation $X_1, X_2, ... X_{15} \sim e^{-1-x}$ for $x \geq -1$. The bootstrap pivotal interval for $\theta = E(X)$ based on $\bar{X} - \theta$ has poor coverage because the distribution of $\bar{X}^* - \bar{X}_{obs}$ is not a good approximation to the distribution of $\bar{X} - \theta$. This is because the high positive correlation between $\bar{X}$ and the sample standard deviation S causes underestimation of the scale when $\bar{x}$ is smaller than θ and overestimation of the scale when $\bar{x}$ is greater than θ. Basing the interval on $(\bar{X} - \theta)/S$ alleviates this problem and the resultant interval has good coverage.

7. Non-parametric profile likelihood intervals.

A different approach to constructing non-parametric confidence intervals can be developed through the use of an approximate profile likelihood. We will first review the profile likelihood then show how it can be used in this setting. Suppose the true distribution is a member of a parametric family of density functions f_η where η is an unknown k-vector of parameters lying in a subset $\Gamma \in R^k$. Our interest focusses on a real valued parameter $\theta = t(\eta)$. Let $l(\eta, y)$ be the log-likelihood of the data.

The *profile likelihood* for θ is constructed as follows. For each θ_0, let $\hat{\eta}(\theta_0)$ maximize $l(\eta, y)$ subject to $t(\eta) = \theta_0$, and let $\hat{\eta}$ be the global maximum likelihood estimator. Then the profile (log) likelihood is defined by

$$pl(\theta) = l(\hat{\eta}(\theta), y) - l(\hat{\eta}, y) \qquad (16)$$

We will assume that for each θ_0, there is a unique restricted maximum $\hat{\eta}(\theta_0)$ and hence $\hat{\eta}(\theta)$ forms a one-dimensional curve in Γ. We will call $\hat{\eta}(\theta)$ the *profile likelihood family*. Now let $R(\theta)$ be the signed square root of twice the profile log likelihood statistic:

$$R(\theta) = \pm[2(pl(\hat{\theta}) - pl(\theta))]^{1/2} \qquad (17)$$

the sign of $R(\theta)$ taken to be the sign of $\hat{\theta} - \theta$. Let $b(\eta) = E_\eta R(\theta)$. Then a second order correct confidence interval can be constructed by treating the pivotal quantity $R(\theta) - b(\hat{\eta})$ as $N(0, 1)$.

Consider now applying this to the non-parametric problem. In the spirit of the bootstrap, we consider the family of distributions to be multinomial with $\eta_i = log(Prob(X = x_i))$, the natural parameters. It turns out that $pl(\theta)$ is very difficult to compute, except for simple (linear) statistics. Hence we consider an approximation to $pl(\theta)$. A convenient approach is to construct a linear approximation $\tilde{\eta}(\theta)$ to $\eta(\theta)$ at $\hat{\eta}$, then form the approximate profile likelihood $\tilde{pl}(\theta) = l(\tilde{\eta}(\theta), y)$. The approximate family $\tilde{pl}(\theta)$ is actually Stein's "least favourable family" (Stein 1956). Given this approximate profile likelihood (which turns out to be easy to compute), we proceed as above, forming the pivot $R(\theta) - b(\hat{\eta})$ and inverting to find the confidence interval. One can show that this approximate interval still produces a second order correct interval. Note that computation of $b(\hat{\eta})$ in the multinomial requires bootstrap sampling, analogous to the calculation of b earlier. This method produces not only a confidence interval for θ but also an approximate non-parametric profile likelihood. For more details on this approach we refer the reader to DiCiccio and Tibshirani (1986).

8. An Example.

Table 2 illustrates the various confidence procedures for a familiar problem. The data $x_1, x_2, \ldots x_n$ are i.i.d N(0,1). The parameter of interest is $\hat{\theta} = Var(x_i)$. Level $1-2\alpha$ confidence intervals are to be based on the unbiased estimate $\hat{\theta} = \sum_1^n (x_i - \bar{x})^2/(n-1)$. The sample size n was taken to be 20 and $\alpha = .05$. The exact interval is based on inverting the pivotal $\hat{\theta}/\theta$ around its chi-squared $(n-1)$ distribution. The standard interval is of the form (16) with $\hat{\sigma} = \hat{\theta}(2/n)^{1/2}$ the estimated asymptotic standard error of $\hat{\theta}$. The bootstrap pivotal interval is based on the pivotal $\hat{\theta}/\theta$. The lower and upper values in Table 2 refer to averages over 300 monte carlo simulations of the intervals. The level column indicates the proportion of trials in which each interval didn't contain the true value $\theta = 1$.

Table 2.

Confidence intervals for the variance

Interval	Average left	Average right	Level(%)
Exact	.630	1.878	10.0
Standard	.466	1.531	11.0
Bootstrap Pivotal	.670	1.860	15.7
Percentile	.484	1.363	24.3
BC	.592	1.467	19.3
BC_a	.617	1.524	19.3
BC_a^0	.633	1.540	18.7
NP Prof Lik	.615	1.579	18.9

The standard interval overcovers on the left and undercovers on the right so that the overall coverage is about right. This illustrates why coverage alone is not a good way to assess confidence intervals. The bootstrap pivotal interval does fairly well, while the others display too low coverage. The percentile interval is especially poor. The BC_a, BC_a^0 and non-parametric profile likelihood intervals capture the asymmetry of the normal interval better than the percentile interval but still underestimate the right hand endpoint.

9. Closing Remarks.

We have discussed a number of bootstrap techniques for constructing confidence intervals. All are potentially useful as data-analytic tools because they are non-parametric and can be applied in complex situations. Further work is needed to evaluate and improve these methods. Our current research focusses the non-parametric profile likelihood interval.

Acknowledgements

The author wishes to thank Bradley Efron, Trevor Hastie, Tim Hesterberg, M. Vernon Johns and Werner Stuetzle for valuable discussions. This work was supported in part by the Natural Sciences and Engineering Research Council of Canada.

REFERENCES

Abramovitch, L. and Singh, K. (1985). Edgeworth corrected pivotal statistics and the bootstrap. Ann. Stat. 13, 116-132.

Benjamini, Y. (1983). Is the t-test really conservative when the parent distribution is long-tailed? J. Amer. Stat. Assoc. 78, pp 645-654

DiCiccio, T. and Tibshirani, R. (1985). Bootstrap confidence intervals and bootstrap approximations. Department of Statistics, University of Toronto Technical report 15.

DiCiccio, T. and Tibshirani, R. (1986). Approximating the profile likelihood through Stein's least favourable family. Department of Statistics, University of Toronto Technical report 2.

Efron, B. (1981). Non-parametric standard errors and confidence intervals (with discussion). Can. J. Stat. Vol 9., No 2. pp 139-172

Efron, B.(1982). The Jackknife, bootstrap, and other resampling plans. SIAM Monograph.

Efron, B.(1983). Tranformation theory: How normal is a family of distributions? Dept. of Statistics, Stanford Univ. Tech Rep. 173.

Efron, B. (1984). Bootstrap confidence intervals for a class of parametric problems. Biometrika 72, 45-58.

Efron, B. (1985). Better bootstrap confidence intervals. Tech. rep. 226, Dept. of Statistics, Stanford. Univ.

Hartigan, J. (1986). *Discussion of* "The Bootstrap method for assessing statistical accuracy" by B. Efron and R. Tibshirani, in Statistical Science, Vol. 1.

Schenker, N. (1985). Qualms about bootstrap confidence intervals. J.Amer. Stat. Assoc. 80, 360-361.

Singh, K. (1981). On the asymptotic acuracy of Efron's bootstrap. Ann. Statist. 9, 1187-1195.

Stein, C. (1956). Efficient non-parametric estimation and testing. Proc. 3rd Berkeley Symp. 187-196.

OPTIMIZATION ALGORITHMS FOR STATISTICAL PROBLEMS

Organizer: *Robert B. Schnabel, University of Colorado*

Efficient Algorithms for Maximum Likelihood Estimation of Probabilistic Choice Models

David S. Bunch, University of California at Davis
(Paper to appear in *SIAM Journal on Scientific and Statistical Computing,* January 1987, Vol. 8, No. 1)

Maximum Likelihood and Quasi–Likelihood for Nonlinear Exponential Family Models

David M. Gay, AT&T Bell Laboratories; Roy E. Welsch, Massachusetts Institute of Technology

An Efficient Algorithm for Orthogonal Distance Data Fitting

Paul T. Boggs, National Bureau of Standards; Richard H. Byrd, Robert B. Schnabel, University of Colorado

Maximum Likelihood and Quasi-Likelihood for Nonlinear Exponential Family Models

David M. Gay

AT&T Bell Laboratories
Murray Hill, New Jersey 07974

*Roy E. Welsch**

Sloan School of Management
Massachusetts Institute of Technology
Cambridge, MA 02139

ABSTRACT

Linear and nonlinear exponential family and quasi-likelihood regression models form a class of models exhibiting a common structure that invites using one algorithmic framework to compute parameter estimates and regression diagnostics for all members in the class. This framework extends our work on nonlinear least squares; it includes iteratively reweighted least squares, but also encompasses secant updates for a piece of the Hessian matrix of the likelihood or quasi-likelihood function along with adaptive decisions about when to use this information. The framework also provides much of the machinery needed to compute "leave one out"-style regression diagnostics. We describe the framework, discuss some implementation details, and present some numerical experience.

1. Introduction

Parametric regression models involve a vector of structural parameters, $\beta \in \mathbb{R}^p$, and a (possibly empty) vector of "nuisance" parameters, $\gamma \in \mathbb{R}^q$. Computing a parameter estimate often reduces to minimizing an objective function of the form

$$f(\beta,\gamma) = \sum_{i=1}^{n} \rho_i(\eta_i(\beta),\gamma). \qquad (1.1)$$

Often η_i is a linear form: $\eta_i(\beta) = x_i^T \beta$, but our approach allows η_i to be an arbitrary twice differentiable function. Similarly, ρ_i can be an arbitrary twice differentiable function; ρ_i or η_i can absorb any relevant response data y_i. (The choice of ρ_i and η_i in (1.1) is far from unique, but we do not have space here to consider the freedom (1.1) allows.)

2. Examples

Our framework includes both linear and nonlinear versions of the generalized linear models of Nelder and Wedderburn [NelW72], Wedderburn's [Wed74] quasi-likelihood models, and the extended quasi-likelihood models of Nelder and Pregibon [NelP86]. Some examples follow.

Least Squares: $q = 0$, $\rho_i(\eta_i(\beta),\gamma) = \rho_i(\eta_i(\beta)) = (y_i - \eta_i(\beta))^2$, where η_i is often a nonlinear function of β. (We could include the variance σ by taking $q = 1$, $\gamma = \sigma$, and $\rho_i(\eta_i(\beta),\gamma) = (y_i - \eta_i(\beta))^2/\sigma^2 + \log(\sigma^2)$, but it is slightly more efficient to estimate β and σ separately.)

Huber: $q = 1$,

$$\rho_i(\eta) = \begin{cases} \dfrac{(y_i - \eta)^2}{2\gamma} + \tau_2\gamma \text{ if } \dfrac{|y_i - \eta|}{\gamma} \le \tau_1 \\ \tau_1 \cdot (|y_i - \eta| - \dfrac{\tau_1\gamma}{2}) + \tau_2\gamma \text{ otherwise} \end{cases},$$

in which τ_1 and τ_2 are tolerances that must be properly chosen; γ is the scale parameter σ in §7.8 of [Hub81]. Many other robust regression problems (e.g. those described in [HolW77]) are also covered by our framework.

Poisson (with $\eta = \log(\mathit{mean})$): $q = 0$, $\rho_i(\eta) = c_i\exp(\eta) - y_i\eta$, where a total of y_i counts are observed in c_i replications of the ith set of experimental conditions.

Binomial: $q = 0$, $\rho_i(\eta) = -y_i\log(\eta) - (c_i - y_i)\log(1 - \eta)$, where

*Research supported in part by National Science Foundation Grant DCR-8116778 and Army Research Office Grant DAAG29-84-K-0207.

there are y_i successes in c_i tries under the ith set of conditions.

Quasi-likelihood with variance $V_\theta(\mu) = \mu^\theta$ (μ = *mean*) [NelP86]: $q = 2$, $\gamma = \begin{bmatrix}\phi \\ \theta\end{bmatrix}$,

$$\rho_i(\eta,\gamma) = \tfrac{1}{2}\log(2\pi\phi y_i^\theta) + \qquad (2.1)$$

$$+ \frac{(y_i^{1-\theta} - \eta^{1-\theta})y_i}{\phi(1-\theta)} + \frac{\eta^{2-\theta} - y_i^{2-\theta}}{\phi(2-\theta)}.$$

(L'Hôpital's rule gives special forms for ρ_i when θ = 1 or 2.)

3. Derivatives

The derivatives of (1.1) with respect to the structural parameters β have a form that often is worth exploiting. It seems essential to compute the gradient $\nabla f = \begin{bmatrix}\nabla_\beta f \\ \nabla_\gamma f\end{bmatrix}$ of f reasonably well. Its structural piece $\nabla_\beta f$ has the form

$$\nabla_\beta f(\beta,\gamma) = J^T \rho'_\eta(\eta,\gamma), \qquad (3.1a)$$

where $J \in \mathbf{R}^{n\times p}$ is the Jacobian matrix

$$J_{i,j} = \frac{\partial\eta_i}{\partial\beta_j}(\beta) \qquad (3.1b)$$

and ρ'_η is the vector

$$(\rho'_\eta)_i = (\rho'_\eta(\eta(\beta)))_i = \frac{\partial\rho_i}{\partial\eta}(\eta_i(\beta),\gamma)(3.1c)$$

The structural part $\nabla^2_{\beta\beta}$ of the Hessian matrix $\nabla^2 f(\beta,\gamma) = \begin{bmatrix}\nabla^2_{\beta\beta}f & \nabla^2_{\beta\gamma}f \\ \nabla^2_{\gamma\beta}f & \nabla^2_{\gamma\gamma}f\end{bmatrix}$ has the form

$$\nabla^2_{\beta\beta}f(\beta,\gamma) = J^T\langle\rho''(\eta,\gamma)\rangle J + \qquad (3.2a)$$

$$+ \sum_{i=1}^{n} (\rho'_\eta)_i \nabla^2\eta_i(\beta),$$

where $\langle\rho''\rangle$ is the diagonal matrix

$$\langle\rho''(\eta,\gamma)\rangle = \langle\rho''(\eta(\beta),\gamma)\rangle \qquad (3.2b)$$

$$= \operatorname{diag}(\frac{\partial^2\rho_1}{\partial\eta^2}, \ . \ . \ . \ , \frac{\partial^2\rho_n}{\partial\eta^2})$$

Just as in the nonlinear least-squares case [DenGW81], the information needed to compute the gradient furnishes an important component of the Hessian, i.e. the J in $J^T\langle\rho''\rangle J$.

Consider now the derivatives of f with respect to the nuisance parameters γ. The gradient components are

$$\nabla_\gamma f(\beta,\gamma) = \sum_{i=1}^{n} \frac{\partial\rho_i}{\partial\gamma}(\eta_i(\beta),\gamma). \qquad (3.3)$$

The Hessian components are

$$\nabla^2_{\beta\gamma}f(\beta,\gamma) = \sum_{i=1}^{n} \frac{\partial^2\rho_i}{\partial\gamma\partial\eta}(\eta_i(\beta),\gamma)\cdot\frac{\partial\eta_i}{\partial\beta}(\beta)$$

$$= J^T\rho''_{\eta\gamma}, \qquad (3.4a)$$

where $\rho''_{\eta\gamma}$ is the $n\times q$ matrix whose ith row is $\frac{\partial^2\rho_i}{\partial\eta\partial\gamma}(\eta_i(\beta),\gamma) = (\frac{\partial^2\rho_i}{\partial\eta\partial\gamma_1}, \ . \ . \ . \ , \frac{\partial^2\rho_i}{\partial\eta\partial\gamma_q});$ and

$$\nabla^2_{\gamma\gamma}f(\beta,\gamma) = \sum_{i=1}^{n} \frac{\partial^2\rho_i}{\partial\gamma^2}(\eta_i(\beta),\gamma) \in \mathbf{R}^{q\times q}. \ (3.4b)$$

The relevant partial derivatives ($\frac{\partial\rho_i}{\partial\gamma}$, $\frac{\partial^2\rho_i}{\partial\gamma\partial\eta}$, and $\frac{\partial^2\rho_i}{\partial\gamma^2}$) are often easy to compute, and we assume they are available, so we may compute $\nabla^2_{\beta\gamma}f$ and $\nabla^2_{\gamma\gamma}f$ directly.

4. Approximating the Mess

It seems relevant to ask how well the techniques that we found helpful in [DenGW81] for solving nonlinear least-squares problems carry over to the more general parameter estimation problems of concern here. One of the key ideas in [DenGW81] is use of a secant update to approximate the messy part of (3.2a), i.e. the sum of little Hessians, $\sum_{i=1}^{n}(\rho'_\eta)_i\nabla^2\eta_i(\beta)$. [In talks, Schnabel sometimes calls this "the mess matrix".] Of course, on some problems $\eta_i(\beta) = x_i^T\beta$ is linear, in which case the messy sum vanishes. But we wish to allow $\eta(\beta)$ to be nonlinear. Thus we are led to considering Hessian approximations $H = \begin{bmatrix}H_{\beta\beta} & H_{\beta\gamma} \\ H_{\gamma\beta} & H_{\gamma\gamma}\end{bmatrix} \approx \nabla^2 f$ in which $H_{\beta\beta}$ has the form

$$H_{\beta\beta} = H_{GN} + S. \qquad (4.1)$$

Here H_{GN} (the "Gauss-Newton" part of the

Hessian) is the part of $\nabla^2_{\beta\beta} f$ that we can easily compute, and $S \approx \nabla^2_{\beta\beta} f - H_{GN}$ is a matrix that we update after taking a step. A straightforward generalization of [DenGW81] considered in [Gay80] is to use

$$H_{GN} = J^{\mathrm{T}} \langle \rho'' \rangle J, \qquad (4.2)$$

but below we also consider an alternative based on the expected value of $J^{\mathrm{T}} \langle \rho'' \rangle J$.

In the process of stepping from the current iterate $\begin{bmatrix} \beta \\ \gamma \end{bmatrix}$ to the next iterate $\begin{bmatrix} \beta^+ \\ \gamma^+ \end{bmatrix} = \begin{bmatrix} \beta \\ \gamma \end{bmatrix} + \begin{bmatrix} \Delta\beta \\ \Delta\gamma \end{bmatrix}$, we learn (approximately) how S should look in the step direction $\begin{bmatrix} \Delta\beta \\ \Delta\gamma \end{bmatrix}$. Thus we determine a vector Ψ such that S^+, the new S matrix, should satisfy

$$S^+ \Delta\beta = \Psi. \qquad (4.3)$$

Many choices of Ψ are possible — we considered half a dozen choices, including those in [DenW78], in the work leading to NL2SOL [DenGW81] — but analogy with that work suggests the following choice of Ψ when $H_{GN} = J^{\mathrm{T}} \langle \rho'' \rangle J$. In this case we wish to have $S^+ \Delta\beta \approx \sum_{i=1}^{n} (\rho'_\eta(\beta^+, \gamma^+))_i \nabla^2 \eta_i(\beta^+) \Delta\beta$. But $\nabla^2 \eta_i(\beta^+) \Delta\beta \approx \nabla \eta_i(\beta^+) - \nabla \eta_i(\beta)$, and

$$\sum_{i=1}^{n} (\rho'_\eta(\beta^+, \gamma^+))_i \Big(\nabla \eta_i(\beta^+) - \nabla \eta_i(\beta) \Big)^{\mathrm{T}}$$
$$= (J^+ - J)^{\mathrm{T}} \rho'_\eta(\eta(\beta^+), \gamma^+),$$

so we are led to the choice

$$\Psi = (J^+ - J)^{\mathrm{T}} \rho'_\eta(\eta(\beta^+), \gamma^+) \qquad (4.4)$$

of Ψ in (4.3).

It seems reasonable to use some kind of least-change secant update [DenS79] to update S; the general idea is that in some sense we should make $S^+ - S$ as small as possible, subject to (4.3). The specific update suggested by Fletcher and Al-Baali [FleA85] is the best one we have seen for nonlinear least-squares problems, and its extension to the present context is the one used in the computing reported below.

5. Adaptive Modeling

Occasionally it is useful to initialize S by finite differences, but usually we just start with $S = \mathbf{0}$ (the $p \times p$ matrix of zeros); we always start with $S = \mathbf{0}$ in the computing reported below.

In the early iterations, S may contribute little to computing good steps $\begin{bmatrix} \Delta\beta \\ \Delta\gamma \end{bmatrix}$. Moreover, as noted above, choosing $S \equiv \mathbf{0}$ is appropriate on some problems. Thus it is useful to adaptively decide whether to include S in the optimization algorithm's model of its objective function. We do this as in [DenGW81]. (We also "size" S as in [DenGW81].)

6. IRLS choice of H_{GN}

Under appropriate assumptions, $\nabla^2_{\beta\beta}$ has expected value $J^{\mathrm{T}} \langle w^{IRLS} \rangle J$, where w^{IRLS} is the weighting vector in the iteratively reweighted least-squares algorithm suggested in [NelW72] and [Wed74] (see also §§1.4 and 2.5 of [McCN83]). Thus we are led to an alternate choice of H_{GN}, namely

$$H_{GN} = J^{\mathrm{T}} \langle w^{IRLS} \rangle J. \qquad (6.1)$$

Both choices of H_{GN} have the form $H_{GN} = J^{\mathrm{T}} \langle w \rangle J$, where w might be ρ'' or w^{IRLS}. Correspondingly, (4.4) generalizes to

$$\Psi = (J^+ - J)^{\mathrm{T}} \rho'_\eta(\eta(\beta^+), \gamma^+) +$$
$$+ J^+ \langle \rho'' - w \rangle J^+ \Delta\beta.$$

7. Trust-Region Steps

Some kind of step-size control is often needed to expand the region of convergence of a locally convergent iteration. In optimization algorithms, one often exercises step-size control by doing an approximate line search: looking at candidate next iterates on a (straight or curvilinear) search path until an acceptable one is found. We like using "trust-region" techniques for this purpose. The general idea is that we have an objective function $f(\alpha)$ whose behavior near the current iterate α we approximate by a model function $f^Q(\delta)$, so that $f(\alpha + \delta) \approx f^Q(\delta)$. (In the present context f^Q is a quadratic form,

$f^Q(\delta) = f(\alpha) + \delta^T \nabla f(\alpha) + \frac{1}{2}\delta^T H \delta$ with $H \approx \nabla^2 f(\alpha)$, but people sometimes use other local models, e.g. conic models [Dav80], [Sor80], [Gra84].) The approximation $f^Q(\delta) \approx f(\alpha + \delta)$ is generally good only for small $\|\delta\|$, so we maintain a bound ζ on the set of δ values $\{\delta\colon \|\delta\| \le \zeta\}$ for which we deem this approximation reliable. We choose a trial step δ^{trial} that approximately minimizes $f^Q(\delta)$ subject to the constraint $\|\delta\| \le \zeta$. If we use the norm $\|\delta\| := \|D\delta\|_2$, where D is a positive-definite diagonal matrix and $\|\cdot\|_2$ is the standard Euclidean norm ($\|x\| = \sqrt{x^T x}$), if f^Q is the quadratic model shown above, and if δ^{trial} exactly minimizes $f^Q(\delta)$ subject to $\|\delta\| \le \zeta$, then δ^{trial} satisfies

$$(H + \lambda D^2)\delta^{trial} = -\nabla f \qquad (7.1)$$

for some Lagrange multiplier $\lambda \ge 0$ that renders $H + \lambda D^2$ positive semidefinite. We end up with an iteration much akin to the Levenberg-Marquardt iteration, except that ζ controls λ rather than *vice-versa*. If the step thus computed fails to give good agreement between f^Q and f, then we reduce ζ and try again, thus effectively performing a curvilinear line search. Otherwise we may accept $\alpha + \delta^{trial}$ as the next iterate (or may increase ζ and try again); [DenGW81] explains the specific rules used in the computing described below. (For more details on matters related to (7.1), also see [Mor78], [Gay81], [Gay83], [MorS83] and the books [DenS83], [GilMW81].)

Sometimes an automatic choice of the scaling matrix D in (7.1) is useful, e.g. relating D_{ii} to some norm of the ith column of J. In the computing reported below, we considered both the fixed choice $D = I$ and an adaptive choice in which D_{ii}, $1 \le i \le p$, is based on $|\tilde{H}_{ii}|$, where $\tilde{H}_{ii} = |(H_{GN})_{ii}| + \max\{S_{ii}, 0\}$ for the structural parameters, i.e., $1 \le i \le p$, and $\tilde{H}_{ii} = (\nabla^2 f)_{ii}$ for the nuisance parameters, i.e. $p < i \le p+q$. The update rules for D are analogous to those in [DenGW81]:

$$D_{ii}^{+} := \begin{cases} \max\{0.6 \cdot D_{ii}, |\tilde{H}_{ii}|\} & \text{if } |\tilde{H}_{ii}| \ge 10^{-6} \\ \max\{0.6 \cdot D_{ii}, 1\} & \text{otherwise} \end{cases} .$$

None of the problems considered below has wild scaling, and the fixed choice $D = I$ usually worked better for them. See Table 3 below (§10) for full details.

8. Regression Diagnostic Hooks

Our current implementation of the algorithm sketched above provides hooks for "leave-one-out" regression diagnostics. The idea is to provide a quick indication of which observations wield the most influence on the parameter estimate. To do this, once we have found optimal parameter estimates (β^*, γ^*), we approximate the Hessian matrix $\nabla^2 f(\beta^*, \gamma^*)$ by finite differences. Then for each i, we let $f^{(i)}$ denote f with the ith observation deleted; we approximate $\nabla^2_{\beta\beta} f^{(i)}$ by $\nabla^2_{\beta\beta} f(\beta^*, \gamma^*) - \rho_i'(\beta^*, \gamma^*)\nabla\eta_i \nabla\eta_i^T$, estimate the parameters $(\beta^{*(i)}, \gamma^{*(i)})$ by

$$(\beta^{*(i)}, \gamma^{*(i)}) \approx$$

$$\approx (\beta^*, \gamma^*) - (\nabla^2 f^{(i)})^{-1} \nabla f^{(i)}(\beta^*, \gamma^*),$$

and point a finger at the i values for which $(\beta^{*(i)}, \gamma^{*(i)})$ and (β^*, γ^*) differ sufficiently. This can give diagnostics analogous to those in [BelKW80], [Pre79], [Pre81], and [Wel82].

9. Test Results

We have run tests with 16 problems that are summarized in Table 1 and described more fully in §10. For those problems where either η or $\log(\eta)$ is linear, we used the weighted least-squares calculation shown in [Fro81] to compute the initial guesses $(\beta^\circ, \gamma^\circ)$; otherwise we used the initial guesses shown in Table 2 (which, if possible, included the ones from the problem sources).

We used the stopping tests and tolerances described in [Gay83] (i.e., the stopping tests of [DenGW81] with tolerances appropriate to the double-precision VAX™ arithmetic we used).

For seven of the problems η is linear ($\eta_i(\beta) = \beta^T x_i$), so (4.2) with $S = \mathbf{0}$ makes $H = \nabla^2 f$, and we are doing Newton's method (with step-size control). These problems required between three and thirteen iterations

VAX is a trademark of Digital Equipment Corporation.

(between four and fourteen function and gradient evaluations).

As mentioned in §7 above, choosing $D = I$ (in (7.1)) usually worked better than updating D. In 30 pairs of runs, $D \equiv I$ was better 19 times, worse five times. The differences were often minor, but were dramatic for problem *mn202*, as Table 3 shows.

To see how useful using our secant approximation to the messy part of the Hessian is, we reran the test program with adaptive modeling turned off, i.e., only using $H := H_{GN}$ as the Hessian approximation. Since $S \equiv \mathbf{0}$ when η is linear and H_{GN} is given by (4.2), this could affect only 44 of the 60 pairs of runs summarized in Table 3. As Table 3 shows, adaptively using S helped on 13 of these runs and hurt on three of them; the degradations caused by using S were minor, but the improvements were sometimes substantial, e.g. on problems *e2.8* and *textile*.

On four of our test problems, the ρ'' and IRLS choices of H_{GN}, (4.2) and (6.1), respectively, are the same. On the remaining problems, (6.1) was better than (4.2) on eight, worse on seven of the runs using S summarized in Table 3.

10. Test Problem Details

Table 1 gives details of the test problems we used. In the following formulae, c_i, n_i, $x_{i,j}$ and y_i denote data (carried by η_i or ρ_i), and $x_i := (x_1, \cdots, x_p)^T$; c_i and n_i sometimes denote replication counts or batch sizes, as in [Fro84]. The choices of $\eta(\beta)$ include:

linear:

$$\eta_i(\beta) = x_i^T\beta; \qquad (10.1)$$

log linear:

$$\eta_i(\beta) = \exp(x_i^T\beta); \qquad (10.2)$$

logistic of linear:

$$\eta_i(\beta) = [\exp(-x_i^T\beta) + 1]^{-1}; \qquad (10.3)$$

special forms:

$$\eta_i(\beta) = \beta_1\{\beta_2 x_{i,1} + \qquad (10.4)$$
$$+ \frac{2\beta_3}{x_{i,2}}[1 - \beta_3(1 - \exp(-x_{i,2}/\beta_3))]x_{i,1}^2\};$$

$$\eta_i(\beta) = \{\exp(\beta_2 + \beta_3 x_{i,2}) + \qquad (10.5)$$
$$+ \exp(\beta_4)\}\exp(\beta_1 x_{i,1});$$

$$\eta_i(\beta) = \beta_1 x_{i,1}[1 - \exp(\beta_2 x_{i,2})]^{\beta_3}; \qquad (10.6)$$

$$\eta_i(\beta) = \beta_1 + \sum_{j=1}^{3} \frac{\beta_{2j}}{x_{i,j} + \beta_{2j+1}}; \qquad (10.7)$$

$$\eta_i(\beta) = \beta_1 + \beta_2\log(x_{i,1}) + \frac{\beta_3 x_{i,2}}{\beta_4 + x_{i,2}}; \qquad (10.8)$$

and

$$\eta_i(\beta) = \beta_1 + \beta_2\log(x_{i,1} - \beta_5) + \qquad (10.9)$$
$$+ \frac{\beta_3 x_{i,2}}{\beta_4 + x_{i,2}}.$$

Choices of $\rho_i(\eta,\gamma)$ include:

Poisson ($\eta = \mu$):

$$\rho_i(\eta,\gamma) = c_i\eta - y_i\log(\eta); \qquad (10.10)$$

Poisson ($\eta = \log(\mu)$):

$$\rho_i(\eta,\gamma) = c_i\exp(\eta) - y_i\eta; \qquad (10.11)$$

binomial ($\eta = \mu$):

$$\rho_i(\eta,\gamma) = -y_i\log(\eta) - (n_i - y_i)\log(1-\eta); \qquad (10.12)$$

binomial (*logistic*):

$$\rho_i(\eta,\gamma) = n_i\log(1 + e^{\eta}) - y_i\eta; \qquad (10.13)$$

binomial (*probit*):

$$\rho_i(\eta,\gamma) = -y_i\log[\Phi(\eta)] - \qquad (10.14)$$
$$- (n_i - y_i)\log[1 - \Phi(\eta)],$$

where Φ is the cumulative normal distribution function; and

gamma ($\eta = \mu^{-1}$):

$$\rho_i(\eta,\gamma) = y_i\eta - c_i\log(\eta). \qquad (10.15)$$

Problems *mn202* and *mn202.1* differ only in their starting guesses; the same goes for *mn205* and *mn205.1*. For problems with η_i given by (10.1) or (10.2), we computed initial guesses β° as for Poisson regression problems in [Fro81]:

$$\beta^\circ = (J^{\circ T}\langle c \rangle J^\circ)^{-1} J^{\circ T} y^\circ,$$

where $J^\circ = J(\beta^\circ)$ and $y^\circ{}_i$ is y_i for (10.1) and $c_i\log(\max\{\frac{y_i}{c_i}, \frac{1}{2c_i}\})$ for (10.2). [We do not explicitly form $J^{\circ T}\langle c \rangle J^{\circ}$.] Table 2 gives the initial we used for other choices of η_i.

(The starting guess for *e3.3* is the same, to six figures, as for *e3.1.)*

Table 3 summarizes our test runs; NF and NG stand for the number of function and gradient evaluations, respectively.

11. Acknowledgement

We thank Daryl Pregibon for helpful discussions and for giving us several machine-readable data sets.

12. References

[BelKW80] Belsley, D.A., Kuh, E., and Welsch, R.E., *Regression Diagnostics*, John Wiley & Sons, New York (etc.), 1980.

[BoxC64] Box, G.E.P., and Cox, D.R., "An Analysis of Transformations", *J. R. Statist. Soc. B* 26 (1964), 211−243.

[Dav80] Davidon, W.C., "Conic Approximations and Collinear Scalings for Optimizers", *SIAM J. Numer. Anal.* 17 (1980), 268−281.

[DenGW81] Dennis, J.E., Gay, D.M., and Welsch, R.E., "An adaptive nonlinear least-squares algorithm", *ACM Trans. Math. Software*. 7 (1981), 348−368.

[DenS79] Dennis, J.E., and Schnabel, R.B., "Least Change Secant Updates", *SIAM Rev.* 21 (1979), 443−459.

[DenS83] Dennis, J.E., and Schnabel, R.B., *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall, Englewood Cliffs, NJ, 1983.

[DenW78] Dennis, J.E., and Welsch, R.E., "Techniques for Nonlinear Least Squares and Robust Regression", *Comm. Statist.* B7 (1978), 345−359.

[DraH69] Draper, N.R., and Hunter, W.G., "Transformations: Some Examples Revisited", *Technometrics* 11 (1969), 23−40.

[FleA85] Fletcher, R. and Al-Baali, M., "Variational Methods for Nonlinear Least Squares", *J. Oper. Res. Soc.* 36 (1985), 405−421.

[Fro81] Frome, E.L., "PREG, A Computer Program for Poisson Regression Analysis", Biomedical Computing Technology Information Center Report R-1302, 1981.

[Fro84] Frome, E.L., "Regression Methods for Binomial and Poisson Distributed Data", manuscript presented at AAPM First Midyear Topical Symposium *Multiple Regression Analysis: Applications in the Health Sciences*, Mobile, AL, 1984.

[Gay80] Gay, D.M., "On Solving Robust and Generalized Linear Regression Problems", pp. 55−83 of *Ottimizzazione non lineare e applicazioni*, edited by S. Incerti and G. Treccani, Quaderni dell'Unione Matematica Italiana 17, Pitagora Editrice, Bologna, Italy, 1980; also (with fewer typos) Technical Summary Report No. 2000, Math. Research Center, Univ. of Wisconsin−Madison, 1979.

[Gay81] Gay, D.M., "Computing optimal locally constrained steps", *SIAM J. Sci. Statist. Comput.* 2 (1981), 186−197.

[Gay83] Gay, D.M., "ALGORITHM 611, Subroutines for Unconstrained Minimization Using a Model/Trust-Region Approach", *ACM Trans. Math. Software* 9 (1983), 503−524.

[GilMW81] Gill, P.E., Murray, W., and Wright, M., *Practical Optimization*, Academic Press, London (etc.), 1981.

[Gra84] Grandinetti, L., "Some Investigations in a New Algorithm for Nonlinear Optimization Based on Conic Models of the Objective Function", *J. Optim. Theory Applic.* 43 (1984), 1−21.

[HolW77] Holland, P.W., and Welsch, R.E., "Robust Regression Using Iteratively Reweighted Least-Squares", *Ann. Math. Statist.* 35 (1977), 73−101.

[Hub81] Huber, P.J., *Robust Statistics*, John Wiley & Sons, New York (etc.), 1981.

[McCN83] McCullagh, P., and Nelder, J.A., *Generalized Linear Models*, Chapman and Hall, London and New York, 1983.

[Mor78] Moré, J.J., "The Levenberg-Marquardt Algorithm: Implementation and Theory", in *Numerical Analysis, Dundee 1977, Lecture Notes in Mathematics 630*, edited by G.A. Watson, Springer-Verlag, Berlin, 1978.

[MorS83] Moré, J.J., and Sorensen, D.C., "Computing a Trust Region Step", *SIAM J. Sci. Statist. Comput.* 4 (1983), 553–572.

[NelP86] Nelder, J.A., and Pregibon, D., "An Extended Quasi-Likelihood Function", to appear in *Biometrika*.

[NelW72] Nelder, J.A., and Wedderburn, R.W.M., "Generalized Linear Models", *J. R. Statist. Soc.* A, 135 (1972), 370–383.

[Pre79] Pregibon, D., "Data Analytic Methods for Generalized Linear Models", *PhD Thesis*, University of Toronto.

[Pre81] Pregibon, D., "Logistic Regression Diagnostics", *Ann. Statist.* 9 (1981), 705–724.

[RobC65] Roberts, E.A., and Coote, G.G., "The Estimation of Concentration of Viruses and Bacteria from Dilution Counts", *Biometrics* 21 (1965), 600–615.

[Sor80] Sorensen, D.C., "The *Q*-Superlinear Convergence of a Collinear Scaling Algorithm for Unconstrained Optimization", *SIAM J. Numer. Anal.* 17 (1980), 84–114.

[TilM61] Till, J.E., and McCulloch, E.A., "A Direct Measurement of the Radiation Sensitivity of Normal Mouse Bone Marrow Cells", *Radiation Res.* 14 (1961), 213–222.

[Wed74] Wedderburn, R.W.M., "Quasi-Likelihood Functions, Generalized Linear Models, and the Gauss-Newton Method", *Biometrika* 61 (1974), 439–447.

[Wel82] Welsch, R.E., "Influence Functions and Regression Diagnostics", pp. 149–169 of *Modern Data Analysis*, edited by R.L. Launer and A.F. Siegel, Academic Press, New York (etc.), 1982.

Problem	n	p	q	η_i	ρ_i	Source
e1	20	2	0	(10.1)	(10.10)	[Fro84, pp. 8–10]
e2.2	27	3	0	(10.1)	(10.10)	[Fro84, pp. 10–13]
e2.6	27	3	0	(10.4)	(10.10)	[Fro84, pp. 16–19]
e2.8	30	4	0	(10.5)	(10.10)	[Fro84, pp. 19–20]
e3.1	5	2	0	(10.1)	(10.12)	[Fro84, pp. 20–24]
e3.3	5	2	0	(10.3)	(10.12)	[Fro84, pp. 22–24]
e3.5	72	9	0	(10.1)	(10.14)	[Fro84, p. 25]
ex1	11	2	0	(10.1)	(10.10)	[RobC65, p. 613]
ex2	20	9	0	(10.2)	(10.10)	[NewW72, pp. 378–379]
ex3	7	3	0	(10.6)	(10.11)	[TilM61]
ex8-10	4	2	0	(10.1)	(10.10)	[Fro84, pp. 8–10]
mn202	64	7	0	(10.7)	(10.15)	[McCN83, pp. 201–203]
mn202.1	64	7	0	(10.7)	(10.15)	[McCN83, pp. 201–203]
mn204	15	4	0	(10.8)	(10.13)	[McCN83, pp. 204–205]
mn205	15	5	0	(10.9)	(10.13)	[McCN83, pp. 205–206]
mn205.1	15	5	0	(10.9)	(10.13)	[McCN83, pp. 205–206]
speed	50	4	2	(10.1)	(2.1)	[DraH69, p. 33]
textile	27	6	2	(10.2)	(2.1)	[BoxC64, p. 223]

Table 1: Test Problem Summary

Problem	$\beta_1, \beta_2, \cdots, \beta_p$						
e2.6	8.0	1.0	3.1				
e2.8	3.0	2.0	1.0	3.0			
e3.3	0.0317714	0.00467588					
ex3	8.0	1.0	3.1				
mn202	1.0	1.0	40.0	2.0	22.0	3.0	32.0
mn202.1	1.0	2.0	3.0	4.0	5.0	6.0	7.0
mn204	1.0	1.0	1.0	1.0			
mn205	1.0	1.0	1.0	1.0	1.0		
mn205.1	−2.896	1.345	1.708	1.674	1.98		

Table 2: Starting Guesses

Problem	w in H_{GN}	Updated D				$D = I$			
	$= J^T(w)J$	Updated S		$S \equiv 0$		Updated S		$S \equiv 0$	
		NF	NG	NF	NG	NF	NG	NF	NG
e1	ρ''	5	5	5	5	5	5	5	5
	IRLS	5	5	5	5	5	5	5	5
e2.2	ρ''	6	5	6	5	4	4	4	4
	IRLS	6	5	6	5	5	5	5	5
e2.6	ρ''	12	10	12	10	10	9	12	12
	IRLS	12	10	12	10	9	9	9	9
e2.8	ρ''	29	26	29	23	33	29	(199	151)*
	IRLS	29	26	29	23	29	23	(157	151)*
e3.1	ρ''	4	4	4	4	5	5	5	5
	IRLS	4	4	4	4	5	5	5	5
e3.3	ρ''	11	9	14	12	8	8	11	11
	IRLS	11	9	14	12	7	7	7	7
e3.5	ρ''	12	9	12	9	9	9	9	9
	IRLS	12	9	12	9	9	9	10	10
ex1	ρ''	5	4	5	4	7	5	7	5
	IRLS	5	4	5	4	7	5	7	5
ex2	ρ''	8	8	9	9	7	7	9	9
	IRLS	8	8	9	9	5	5	5	5
ex3	ρ''	7	6	7	6	6	6	6	6
ex8-10	ρ''	5	5	5	5	5	5	5	5
	IRLS	5	5	5	5	5	5	5	5
mn202	ρ''	40	22	44	28	10	9	10	9
mn202.1	ρ''	24	16	24	16	17	15	17	15
mn204	ρ''	12	8	12	8	9	8	9	8
mn205	ρ''	19	11	19	11	11	10	9	8
mn205.1	ρ''	10	9	10	9	6	6	6	6
speed	ρ''	17	14	17	14	13	13	13	13
	IRLS	17	14	17	14	14	14	14	14
textile	ρ''	71	47	71	47	67	46	67	46
	IRLS	71	47	71	47	71	45	(175	151)*

Table 3

*Iteration limit reached.

AN EFFICIENT ALGORITHM FOR ORTHOGONAL DISTANCE DATA FITTING

Paul T. Boggs, National Bureau of Standards
Richard H. Byrd, University of Colorado
Robert B. Schnabel, University of Colorado

Abstract. One of the most widely used methodologies in scientific and engineering research is the fitting of equations to data by least squares. In cases where significant observation errors exist in all data (independent) variables, however, the ordinary least squares approach, where all errors are attributed to the observation (dependent) variable, is often inappropriate. An alternate approach, suggested by several researchers, involves minimizing the sum of squared orthogonal distances between each data point and the curve described by the model equation. We refer to this as orthogonal distance regression (ODR). This paper describes a method for solving the orthogonal distance regression problem that is a direct analog of the trust region Levenberg-Marquardt algorithm. The number of unknowns involved is the number of model parameters plus the number of data points, often a very large number. By exploiting sparsity, however, our algorithm has a computational effort per step which is of the same order as required for the Levenberg-Marquardt method for ordinary least squares. We summarize the theoretical properties of our algorithm, and provide the results of computational tests that illustrate some differences between the two approaches.

1. Introduction

The problem of fitting a model to data with errors in the observations has a rich history and a considerable literature. The problem where there are also errors in the independent variables at which these observations are made, however, has only relatively recently been given attention. In this paper, we consider a general form of this extended problem and provide an efficient and stable algorithm for its solution. Several names for this extended problem have been suggested; we prefer orthogonal distance regression (ODR).

Errors in independent variables virtually always occur, but are often ignored in order that classical or ordinary (linear or nonlinear) least squares (OLS) techniques can be applied (see, e.g., [LawH74], [Ste73], [Mor77], [DenGW81]). Also, if these errors are small with respect to those in the observed variables, then ignoring them does not usually seriously degrade the accuracy of the estimates. In some fields, however, measurement techniques are sufficiently accurate that errors in the independent variables are not insignificant compared to those in the observations. Examples at the National Bureau of Standards (NBS) include the calibration of electronic devices, flow-meters and calorimeters. Another class of examples comes from curve and surface fitting problems.

We first develop a formal statement of the ODR estimation problem and briefly discuss its application to statistical estimation and to curve fitting. The derivation and convergence analysis of a highly efficient algorithm for solving ODR problems is summarized in Sections 2 and 3. In Section 4, the results of some computations are shown which illustrate the performance of the algorithm and allow some comparisons with ordinary least squares. This material is presented in greater detail in Boggs, Byrd, and Schnabel [BogBS85].

Observations in applied science are often thought of as satisfying a mathematical model of the form

$$(1.1) \qquad y = f(x,\beta)$$

where y is taken to be the "observed" value, or independent variable; and $\beta \in R^p$ is the set of parameters to be estimated. The function f is not assumed to be linear, but is assumed to be smooth. The data are simply the pairs $(x_i, y_i), i = 1,\ldots,n$. Typically the number of data points, n, is far greater than the number of parameters, p.

In the classical case, only the observations y_i are assumed to be contaminated with errors. If these errors are additive and the mathematical model is exact then

$$(1.2) \qquad y_i = f(x_i,\hat{\beta}) + \hat{\epsilon}_i \qquad i = 1,\ldots,n$$

for some correct value of the parameters $\hat{\beta}$. If in addition the errors are normally distributed with mean 0 and variance $\sigma^2 I$, then maximum likelihood estimate of $\hat{\beta}$ is the solution to the least squares problem

$$(1.3) \qquad \min_{\beta} \sum_{i=1}^{n} [y_i - f(x_i,\beta)]^2 .$$

If f is a linear function of $\hat{\beta}$ then this is a classical linear least squares problem, otherwise it is a classical nonlinear least squares problem. Even when the above assumptions on the model or the errors are not satisfied, problem (1.3) is the most frequently used method for parameter estimation.

In the more general situation, the measurements of the independent variables x_i are also assumed to contain errors. If we assume that y_i has unknown additive error $\hat{\epsilon}_i$ and that x_i has unknown additive error $\hat{\delta}_i$, then (1.2) becomes

$$(1.4) \qquad y_i = f(x_i + \hat{\delta}_i;\beta) + \hat{\epsilon}_i .$$

An intuitively reasonable way to select the parameters in this case is to choose the β that causes the sum of the squares of the orthogonal distances from the data points (x_i, y_i) to the curve $f(x,\beta)$ to be minimized. If r_i is the orthogonal distance from (x_i, y_i) to the curve, then

$$r_i^2 = \epsilon_i^2 + \delta_i^2, \qquad i = 1,\ldots,n$$

where ϵ_i and δ_i solve

$$(1.5) \qquad \begin{array}{l} \min\limits_{\epsilon_i,\delta_i}(\epsilon_i^2 + \delta_i^2) \\ \text{subject to } f(x_i + \delta_i;\beta) + \epsilon_i = y_i. \end{array}$$

The constraint in (1.5) ensures that the distance r_i connects the point (x_i, y_i) to the curve. The minimization ensures that r_i is the radius of the smallest circle centered at (x_i, y_i) which is tangent to the curve $f(x_i;\beta)$. Therefore, the parameters β that cause the sum of the squares of the orthogonal distances from the data points to the curve to be minimized are found by solving

$$(1.6) \qquad \begin{array}{l} \min\limits_{\beta,\epsilon,\delta} \sum\limits_{i=1}^{n} r_i^2 = \min\limits_{\beta,\epsilon,\delta} \sum\limits_{i=1}^{n} (\epsilon_i^2 + \delta_i^2) \\ \text{subject to } y_i = f(x_i + \delta_i;\beta) + \epsilon_i, \\ \qquad i = 1,\ldots,n. \end{array}$$

Since the constraints in (1.6) are simple linear constraints in ϵ_i, we solve for ϵ_i and eliminate both these variables and all of the constraints thereby obtaining

$$\min_{\beta,\delta} \sum_{i=1}^{n} \left[(f(x_i + \delta_i;\beta) - y_i)^2 + \delta_i^2 \right]$$

which is now an unconstrained minimization problem.

Two slight extensions to this form constitute the ultimate problem to be considered. The first allows the possibility that $x_i \in R^m$ rather than R^1. Therefore, $\delta_i \in R^m$ and instead of δ_i^2 in (1.7) we have $\delta_i^T\delta_i = \sum_{j=1}^m \delta_{ij}^2$. (The superscript T denotes transpose.) The second extension merely admits a general weighting scheme on the problem. The form we have chosen results in the general nonlinear ODR problem

$$(ODR) \qquad \min_{\beta,\delta} \sum_{i=1}^{n} w_i^2 \left[(f(x_i + \delta_i;\beta) - y_i)^2 + \delta_i^T D_i^2 \delta_i \right]$$

where $w_i > 0, i = 1, ..., n$ and

$$D_i = \operatorname{diag}\{d_{ij} > 0,\ j = 1,\ldots,m\},\ i = 1,\ldots,n,$$

i.e., D_i is a diagonal matrix of order m. It follows that the vectors $y, w \in R^n$ and $x, \delta \in R^{nm}$ and that $\delta_i^T D_i \delta_i^2 = \sum_{j=1}^m \delta_{ij}^2 d_{ij}^2$.

While we have not assumed that f is linear, it is important to note that (ODR) is a nonlinear optimization problem even if f is the simple linear function

$$y = \beta_1 x + \beta_2$$

since we then have that

$$y_i = \beta_1(x_i + \delta_i) + \beta_2 + \epsilon_i.$$

Clearly the product of β_1 and δ_i is an unavoidable nonlinearity.

ODR problems have been considered by statisticians, usually under the rubric errors in variables. Most of this effort, however, has been devoted to linear models, i.e., when f is linear in β. (See e.g.,[Mor71], [KenSO83], [Bar74,p.67] and [Ful86].) As in the classical nonlinear least squares case, little theory on the statistical properties of the solution appears to exist. It is known that if both ϵ and δ are normally distributed with mean zero and variances $\sigma_\epsilon^2 I$ and $\sigma_\delta^2 I$ respectively, then the solution of (ORD) with $w_i = 1$ and $D_i = (\sigma_\epsilon/\sigma_\delta)I,\ i = 1,\ldots,n$ is a maximum likelihood estimate of the parameters. Unfortunately, as in the nonlinear classical case, no generally valid, computationally efficient, inferential statistical tests are known.

Independent of statistical considerations, ODR has potentially significant applications in curve and surface fitting. Consider, for example, the problem of finding the parabola which best fits the given set of points (We have seen this problem arise from a dental application.) Here it is clear that ordinary least squares will unduly weight the top data points, while fitting in the horizontal direction would undully weight the bottom data points. An orthogonal measure of distance alleviates these problems and provides a reasonable fit. A related case is the problem of fitting near an asymptote. Orthogonal distances here prevent the undue influence of points close to the asymptote. This problem is discussed further in Section 4.

The literature contains several algorithms for solving (ODR) and related problems. For example, Golub and Van Loan [GolV83] give a singular value decomposition procedure for the problem when f is linear. They refer to this problem as total least squares. Britt and Luecke [BriL73] consider the nonlinear case as well as the nonlinear implicit case and present an algorithm. Recently, Schwetlick and Tiller [SchT85] proposed an algorithm similar to the one here for the nonlinear problem. Our algorithm, however, does not make use of the singular value decomposition and it does incorporate a full trust region strategy.

2. The Algorithm

In order to solve the minimization problem (ODR),

$$(2.1) \qquad \min_{\beta,\delta} \sum_{i=1}^{n} w_i^2 \left[(f(x_i + \delta_i;\beta) - y_i)^2 + \delta_i^T D_i^2 \delta_i \right]$$

we first express it in a more convenient form and simplify the notation. Next, we give an overview of the iteration which is based on the trust region -Levenberg-Marquardt strategy popularized by Moré [Mor77]. (See also [Heb73], [DenS83].) We then show how to modify this technique to obtain an algorithm which requires the same order of work per iteration as these algorithms applied to the same problem without allowing changes to x_i. That is, if the $\delta_i's$ are held fixed at zero, ODR reduces to OLS and trust region methods require $O(np^2)$ operations per iteration. Our algorithm, by exploiting the structure of (ODR), still requires only $O(np^2) + O(nm)$ operations per iteration to solve the problem.

While we have designed and implemented the algorithm to handle the full generality of (2.1), the notation is considerably simplified by assuming $x_i \in R^1$. We temporarily make this assumption and rewrite (2.1) into the form of an OLS problem by the following device. Let

(2.2)
$$g_i(\beta,\delta) = \begin{cases} w_i\left(f(x_i+\delta_i;\beta) - y_i\right), & \text{i=1,\ldots,n} \\ w_{i-n}d_{i-n}\delta_{i-n}, & \text{i=n+1, \ldots, 2n.} \end{cases}$$

Also let $G : R^{p+n} \rightarrow R^{2n}$ have component functions $g_i(\eta)$ where $\eta = \binom{\beta}{\delta}$. Now (ODR) becomes

$$\text{(2.3)} \qquad \min_{\eta} \|G(\eta)\|^2 = \min_{\beta,\delta} \sum_{i=1}^{2n} \left(g_i(\beta,\delta)\right)^2$$

which is an OLS problem with $(p+n)$ parameters and $2n$ equations. (In all cases in this paper, $\|\cdot\|$ denotes the ℓ_2 vector or matrix norm.) Direct application of trust region methods to (2.3) would require $O(2n(n+p)^2)$ operations per iteration which rapidly becomes prohibitive if n is large. (Recall that n is usually far greater than p in practice.)

The basic idea of a trust region strategy is to choose as the step that vector which minimizes a linear approximation to G over a region in which the linearization is a "reasonable" approximation to G . Specifically, if $G'(\eta^c) \in R^{2n\times(n+p)}$ is the Jacobian matrix of G evaluated at the current iterate, η^c, then the step z is chosen by solving

$$\text{(2.4)} \qquad \begin{aligned} &\min_{z} \|G(\eta^c) + G'(\eta^c)z\|^2 \\ &\text{subject to}\|Zz\| \leq \tau \end{aligned}$$

where Z is a nonsingular (usually diagonal) scaling matrix and τ is the trust region radius. It is easy to show that the solution to (2.4) is given by the $z(\alpha)$ satisfying

$$\text{(2.5)} \quad \left(G'(\eta^c)^T G'(\eta^c) + \alpha Z^T Z\right) z(\alpha) = -G'(\eta^c)^T G(\eta^c)$$

where $\alpha > 0$ is the Lagrange multiplier for the inequality constraint. Note that if $\|z(0)\| \leq \tau$, $\alpha = 0$ and the constraint is inactive. Otherwise $\alpha > 0$ and the constraint is active. Equation (2.5) is the famous Levenberg-Marquardt formula, but this derivation has given rise to more stable and robust implementations. (See, e.g., [Mor77] and [DenS83]). Clearly (2.5) can be regarded as the "normal equations" for the extended least squares problem,

$$\text{(2.6)} \qquad \begin{bmatrix} G'(\eta^c) \\ \alpha^{1/2}Z \end{bmatrix} z =_2 - \begin{bmatrix} G \\ 0 \end{bmatrix}$$

where"$=_2$" means "equal in the least squares sense."

Our implementation is based on the careful exploitation of the structure of the extended Jacobian matrix in (2.6). From (2.2) we have that

$$G'(\eta^c) = \begin{pmatrix} J & V \\ 0 & D \end{pmatrix}$$

where

$$\begin{aligned} J \in R^{n\times p} : \quad J_{ij} &= \frac{\partial g_i(\beta,\delta)}{\partial \beta_j} = \frac{w_i \partial f(x_i+\delta_i;\beta)}{\partial \beta_j}, \\ &i = 1,\ldots,n, \qquad j = 1,\ldots,p; \\ V \in R^{n\times n} : \quad V_{ij} &= \frac{\partial g_i(\beta,\delta)}{\partial \delta_j} = \frac{w_i \partial f(x_i+\delta_i;\beta)}{\partial \delta_j}, \\ &i = 1,\ldots,n \qquad j = 1,\ldots,n; \\ D \in R^{n\times n} : \quad D &= \operatorname{diag}\{w_i d_i, i = 1,\ldots,n\}. \end{aligned}$$

Here, we have omitted the arguments of J and V for the sake of clarity. Observe that since g_i only depends on δ_i, $i = 1,\ldots,n$,

$$V = \operatorname{diag}\left\{\frac{\partial g_i(\beta,\delta)}{\partial \delta_i},\ i = 1,\ldots,n\right\}.$$

Commensurate with this partitioning of $G'(\eta^c)$, η^c is naturally partitioned into components $(\beta^c,\delta^c)^T$ and the step z into a step in β , say s , and a step in δ , say t. Furthermore, we allow for s to be scaled by a nonsingular diagonal scaling matrix S and t by a nonsingular diagonal matrix T. Thus (2.6) becomes

$$\text{(2.7)} \qquad \begin{bmatrix} J & V \\ 0 & D \\ \alpha^{1/2}S & 0 \\ 0 & \alpha^{1/2}T \end{bmatrix} \begin{bmatrix} s \\ t \end{bmatrix} =_2 - \begin{bmatrix} G_1 \\ G_2 \\ 0 \\ 0 \end{bmatrix}$$

where G_1 is the first n components of G and G_2 is the last n components.

Now, if $x_i \in R^m$, then (2.7) will have the same form except that $V \in R^{n\times nm}; T, D \in R^{nm\times nm}$ are still diagonal; and V , instead of being diagonal, has the"staircase" structure which is illustrated for $n = 4$ and $m = 3$ as follows:

$$V = \begin{bmatrix} xxx & & & \\ & xxx & & \\ & & xxx & \\ & & & xxx \end{bmatrix}.$$

The rest of the development now allows $x_i \in R^m$.

Boggs, Byrd, and Schnabel [BogBS85] derive in detail an efficient procedure for solving (2.7). Here we just give this procedure and summarize its derivation.

By forming the normal equations for (2.7), it is straightforward to show that the s that solves (2.7) is the solution to

$$\text{(2.8)} \qquad \begin{pmatrix} \bar{J} \\ \alpha^{1/2}S \end{pmatrix} s =_2 \begin{pmatrix} y \\ 0 \end{pmatrix}$$

where

$$\text{(2.9)} \qquad \bar{J} = (I - VP^{-1}V^T)^{1/2}J$$

(2.10)
$$y = (I \cdot\cdot VP\ {}^{1}V^T)^{-1/2}[-G_1 + VP^{-1}(V^TG_1 + DG_2)]$$

with P defined by $P = V^TV + D^2 + \alpha T^2$. The same derivation shows that given s, the t that solves (2.7) is given by

$$t = -P^{-1}(V^T G_1 + DG_2 + V^T Js). \tag{2.11}$$

Boggs, Byrd, and Schnabel then show that (2.9), (2.10). and (2.11) are equivalent to

$$\bar{J} = \text{diag}\left\{\left[\frac{1}{1+\omega_i}\right]^{1/2}, \; i = 1,\ldots,n\right\} J \tag{2.12}$$

(2.13)

$$y = -\text{diag}\left\{[1+\omega_i]^{1/2}, \; i = 1,\ldots,n\right\}(G_1 - VE^{-1}DG_2).$$

and

$$t = -E^{-1}\Big[V^T \text{diag}\left\{\frac{1}{1+\omega_i}, \; i = 1,\ldots,n\right\} (G_1 + Js - VE^{-1}DG_2) + DG_2\Big]. \tag{2.14}$$

where E is defined by $E = D^2 + \alpha T^2$ and

$$\omega_i = \sum_{j=1}^{m} \frac{V_{ij}^2}{E_{(i-1)m+j}}, \; i = 1,\ldots,n.$$

Equations (2.12)–(2.13) show that the system of equations (2.8) can be formed in $O(np + nm)$ operations. The solution of (2.8) then involves a QR decomposition of $\bar{J}$ (accomplished by Householder transformations with column pivoting) and then a sequence of plane rotations to eliminate $\alpha^{1/2}S$. The cost is for this phase is dominated by the $O(np^2)$ operations for the QR decomposition of $\bar{J}$. It is then easily verified that the cost of calculating t from (2.14) is dominated in cost by the $O(np)$ operations needed to form Js and several $O(nm)$ terms.

Thus the leading cost of calculating a step for ODR is the same $O(np^2)$ operations needed to do the factorization of an $n \times p$ matrix as in OLS. The only additional costs are a small number of calculations costing $O(nm)$ or $O(np)$ operations.

It may occur to the reader that an efficient QR factorization of the matrix in (2.7) might yield a procedure with the same order of work. By re-ordering the upper 2×2 blocks, one can, indeed, do the factorization of this part in $O(np^2)$ operations. The subsequent elimination of the αS and αT blocks, however, would require $O((nm+p)^2)$ operations for each α. It is for this reason, as well as others, that Schwetlick and Tiller [SchT85] do only a "partial" trust region strategy, i.e.. their trust region only applies to the step in the β variables. In some badly scaled problems, however, (e.g., Example 3 in Section 4) the ability to scale and constrain the step in δ is essential to solve the problem.

The above formulas for s and t are used for each α value in (2.5). Thus in order to complete the specification of the algorithm, we need to provide the procedure for computing the trust region parameter α to satisfy (2.4) and for adjusting the trust region radius r. These details are discussed in [BogBS85].

Since many users will want to compare the results of OLS with ODR, our code includes an option to do OLS. Enabling this option merely initializes the δ vector to zero and sets V to zero whenever it is computed. It is easily verified that, in this case, each step reduces to the OLS Levenberg-Marquardt step and yields $t = 0$ leaving $\delta = 0$. Using this procedure to do OLS, therefore, is equivalent to a standard OLS algorighm with a moderate extra algebraic overhead.

3. Local and Global Convergence Analysis

Trust-region-Levenberg-Marquardtmethods applied to the general nonlinear least squares problem have well known convergence properties (see e.g., [Pow75], [Mor77], [MorS81], [ShuSB85]). As long as the sequence of Jacobian matrices, $\{G'(\eta_k)\}$, is uniformly bounded, then

$$\lim_{k\to\infty} G'(\eta_k)^T G(\eta_k) = 0,$$

so that any cluster point satisfies the first order necessary conditions for a local minimizer. These results apply to our algorithm and nothing more needs to be said regarding global convergence.

The local convergence behavior of general trust-region-Levenberg-Marquardt methods for nonlinear least squares is discussed by Byrd and Schnabel [ByrS86] who show that, if there is a cluster point η_* where $G'(\eta_*)$ is nonsingular, then the iterates converge at least linearly to η_* independent of the size of $G(\eta_*)$. This theory also applies to our algorithms. If, in addition, the residual $G(\eta_*)$ is sufficiently small, Byrd and Schnabel show that asymptotically the trust region constraint becomes inactive, and that the Levenberg-Marquardt algorithm reduces to the Gauss-Newton iteration

$$\eta_{k+1} = \eta_k - \left[G'(\eta_k)^T G'(\eta_k)\right]^{-1} G'(\eta_k)^T G(\eta_k)$$

and is linearly convergent to η_*. The linear convergence analysis of the Gauss-Newton method is well known (see e.g., [OrtR70], [DenS83]). The constant of linear convergence depends upon the smallest singular value of $G'(\eta_*)$, the residual $G(\eta_*)$, and the nonlinearity of $G(\eta)$ near η_*.

The small residual analysis is particularly relevent to ODR because most applications of ODR will have small residuals. This is especially true when ODR is used to consider errors in independent variables in parameter estimation, because errors in the independent variables are most likely to be considered when the model and the dependent variable measurements are accurate, which implies that the residuals will be small.

It turns out that the application of the local Gauss-Newton analysis to ODR is nontrivial, although the expected results can be proven. To simplify the algebra here, we consider a version of the ODR problem (2.1) with the simplified weighting scheme $w_i = 1$ and $d_i = \sigma$ for all i, i.e.,

$$\min_{\beta,\,\delta} \sum_{i=1}^{n} \left[(f(x_i + \delta_i; \beta) - y_i)^2 + \sigma^2 \delta_i^T \delta_i\right] \tag{3.1}$$

where $\sigma > 0$. This weighting still allows the metric of distance from the curve $f(x;\beta)$ to the data points (x_i, y_i) to vary from vertical (as $\sigma \to \infty$) to orthogonal ($\sigma = 1$) to horizontal (as $\sigma \to 0$). (We explain this statement more carefully later in this section.) This is all the generality in the weighting that is usually used in practice, and precisely what we use in most of our computational results in Section 4.

To further simplify notation, we rewrite (3.1) as

$$\min_{\eta} R(\eta)^T R(\eta) + \sigma^2 \delta^T \delta \tag{3.2a}$$

or equivalently,

$$\min_{\eta} G(\eta)^T G(\eta) \tag{3.2b}$$

where $\delta = (\delta_1^T, \delta_2^T, \ldots, \delta_n^T)^T$, $\eta = (\beta^T, \delta^T)^T$, $R(\eta)_i = f(x_i + \delta_i; \beta) - y_i$, $i = 1, \ldots, n$, and $G(\eta) = (R(\eta)^T, \sigma\delta^T)^T$. Our analysis will not depend upon the special form of $R(\eta)$ in any way. Recall that

$$G'(\eta) = \begin{pmatrix} J(\eta) & V(\eta) \\ 0 & \sigma^2 I \end{pmatrix}$$

where $J(\eta)$ and $V(\eta)$ are, as in Section 2, the derivatives of $R(\eta)$ with respect to β and δ respectively.

The difficulty in applying standard Gauss-Newton analysis to (3.2) is that $G(\eta)$ and $G'(\eta)$ are functions of σ. In Theorem 3.1 we show that the convergence can be analyzed in terms of the properties of $J(\eta)$, $V(\eta)$, $R(\eta_*)$, and δ_* only, i.e., independent of σ except for its role in determining η_*. For the proof of Theorem 3.1, see [BogBS85].

In the statement of Theorem 3.1, we often omit the argument η; i.e., we denote $G(\eta_*)$ and $G(\eta_0)$ by G_* and G_0, respectively, and likewise for other symbols in place of G. Also for J having full column rank, J^+ denotes $(J^T J)^{-1} J^T$, and for V having full row rank, V^+ denotes $V^T(VV^T)^{-1}$. Note that $\|J^+\|^2 = \|(J^T J)^{-1}\|$.

Theorem 3.1. *Let $R(\eta) : R^{\iota} \to R^n$ be continuously differentiable in an open convex set $D \subset R^{\iota}$. Let $\eta^T = (\beta^T, \delta^T)$, $\beta \in R^p$, $\delta \in R^q$, let σ be a positive scalar, and let $G(\eta) = \begin{pmatrix} R(\eta) \\ \sigma\delta \end{pmatrix}$. Assume there exists $\eta^* \in D$ such that $G'(\eta)^T G(\eta^*) = 0$, and that there exists $\gamma \geq 0$ for which*

$$\|R'(\eta) - R'(\eta^*)\| \leq \gamma \|\eta - \eta^*\|$$

for all $\eta \in D$. Define

$$\begin{aligned} c_1 &= \gamma\Big[\|(J_*^T J_*)^{-1}\| \ \|R_*\| \\ &\qquad + \left(1 + \|(J_*^T J_*)^{-1}\| \ \|V_*\|^2\right) \|V_*^+\| \ \|\delta_*\|\Big] \\ c_2 &= (\gamma/2)\left[\|V_*^+\| + \|J_*^+\| \left(1 + \|V_*\| \ \|V_*^+\|\right)\right]. \end{aligned}$$

If $c_1 < 1$, then for any $c \in (1, 1/c_1)$, there exists $\epsilon > 0$ such that for all η_0 for which $\|\eta_0 - \eta_*\| \leq \epsilon$, the sequence generated by the Gauss-Newton method

$$\eta_{k+1} = \eta_k - \left(G'(\eta_k)^T G'(\eta_k)\right)^{-1} G'(\eta_k)^T G(\eta_k)$$

is well defined, converges to η_* and obeys

$$\|\eta_{k+1} - \eta_*\| \leq c\,(c_1 + c_2 \|\eta_k - \eta_*\|)\, \|\eta_k - \eta_*\|.$$

In practical ODR applications, the user may wish to solve (3.1) for various values of σ. Now we consider the behavior of the ODR problem (3.2) as the parameter σ is varied. For this purpose, let us denote the global minimizer to (3.2) by $\eta_*(\sigma)$. Then by standard analyses of barrier function methods, (see e.g., [FiaM68] or [Lue73]) we know that the limit of $\eta_*(\sigma)$ as $\sigma \to \infty$ is the solution to

$$\min_{\eta} \|R(\eta)\|^2 \quad \text{subject to } \delta = 0,$$

i.e., the standard OLS problem

$$\min_{\beta} \|R(\beta, 0)\|^2. \tag{3.3}$$

Similarly, the limit of $\eta_*(\sigma)$ as $\sigma \to 0$ is the solution to the implicit least squares (ILS) problem

$$\min_{\eta} \|\delta\|^2 \quad \text{subject to } R(\eta) = 0. \tag{3.4}$$

In the data fitting context where $R(\eta)_i = f(x_i + \delta_i; \beta) - y_i$, (3.3) is the standard problem where the independent variables x_i are assumed exact so that the metric of distance is in the y (vertical) direction only. In constast (3.4) is the case where the dependent variables y_i are assumed exact and the independent variables x_i inexact, so that the metric is entirely in the x (horizontal) direction.

The standard analysis of barrier function methods also shows that $\|R(\eta_*(\sigma))\|$ is a monotonically increasing function of σ, and that $\|\delta_*(\sigma)\|$ is a monotonically decreasing function of σ. This means that for all $\sigma \in (0, \infty)$, the values of $\|R(\eta_*(\sigma))\|$ and $\|\delta_*(\sigma)\|$ are bounded above by the optimal objective function values for problems (3.3) and (3.4), respectively. In data fitting terms, for any σ, the norm of the optimal vertical residuals in ODR is bounded above by the norm of the optimal residuals in OLS, and the norm of the optimal horizontal residuals in ODR is bounded above by the norm of the optimal residual for the ILS problem. The computational results of Section 4 demonstrate these relationships.

Combining the above facts with Theorem 3.1 shows that, if the optimal objective function values for problems (3.3) and (3.4) are sufficiently small, and if $J(\eta_*(\sigma))$ and $V(\eta_*(\sigma))$ are sufficiently well-conditioned for all $\sigma \in (0, \infty)$, then the Gauss-Newton algorithm applied to (3.2) is linearly convergent for any $\sigma \in (0, \infty)$.

Corollary 3.2. *Let η, β, δ, $R(\eta)$, $G(\eta)$, $J(\eta)$, and $V(\eta)$ be defined as in Theorem 3.1. For any $\sigma \in (0, \infty)$, let $\eta_*(\sigma) = (\beta_*(\sigma)^T, \delta_*(\sigma)^T)^T$ denote the global solution to*

$$\min_{\beta, \delta} \|R(\beta, \delta)\|^2 + \sigma^2 \|\delta\|^2.$$

Also let β_{OLS} denote the global solution to the ordinary least squares problem

$$\min_{\beta} \|R(\beta, 0)\|^2$$

and let $(\beta_{ILS}, \delta_{ILS})$ denote the global solutions to the implicit least squares problem

$$\min_{\beta,\ \delta} \|\delta\|^2 \text{ subject to } R(\beta,\ \delta) = 0.$$

Let $R_{OLS} = R(\beta_{OLS})$. Assume that there exist $\hat{\epsilon} > 0$, $\hat{\gamma} \geq 0$ such that for each $\sigma \in (0,\ \infty)$,

$$\|R'(\eta) - R'(\eta_*(\sigma))\| \leq \hat{\gamma}\, \|\eta - \eta_*(\sigma)\|$$

for all η for which $\|\eta - \eta_*(\sigma)\| \leq \hat{\epsilon}$. Assume also that for all $\sigma \in (0,\ \infty)$, $J(\eta_*(\sigma))$ and $V(\eta_*(\sigma))$ have full column and row rank, respectively, and let $\hat{J}$, $\hat{J}^+$, $\hat{V}$, and $\hat{V}^+$ be uniform bounds on the norms $\|J(\eta_*(\sigma))\|$, $\|J(\eta_*(\sigma))^+\|$, $\|V(\eta_*(\sigma))\|$, and $\|V(\eta_*(\sigma))^+\|$, respectively, over all $\sigma \in (0,\ \infty)$. Define

$$\hat{c}_1 = \hat{\gamma}\left[\left(\hat{J}^+\right)^2 R_{OLS} + \left(1 + \left(\hat{J}^+\right)^2 \hat{V}^2\right)\hat{V}\delta_{ILS}\right]$$

$$\hat{c}_2 = (\hat{\gamma}/2)\left[\hat{V}^+ + \hat{J}^+\left(1 + \hat{V}\hat{V}^+\right)\right].$$

If $\hat{c}_1 < 1$, then for any $c \in (1,\ 1/\hat{c}_1)$, there exists $\hat{\epsilon} > 0$ such that for any $\sigma \in (0,\ \infty)$, the sequence $\{\eta_k\}$ generated by the Gauss-Newton method applied to (3.2) starting from any η_0 for which $\|\eta_0 - \eta_*(\sigma)\| \leq \hat{\epsilon}$ is well-defined, converges to $\eta_*(\sigma)$, and obeys

$$\|\eta_{k+1} - \eta_*(\sigma)\| \leq c\,[\hat{c}_1 + \hat{c}_2\,\|\eta_k - \eta_*(\sigma)\|]\,\|\eta_k - \eta_*(\sigma)\|\,.$$

4. Computational Testing.

In this section we report the results of preliminary computational testing. These tests, consisting of two contrived problems and one real problem, were selected in order to illustrate the effectiveness of the implementation and to demonstrate the performance of the basic algorithm. They also allow us to contrast ODR and OLS, which can have rather dramatic differences, and to point out some of the inherent difficulties in ODR problems.

The contrast between OLS and ODR is best brought out in terms of the parameter σ and the function $\beta(\sigma)$ from Section 3. (Recall that $\beta(\infty)$ corresponds to the OLS solution.) Since, in practice, the correct value of σ may not be known exactly, it is of interest to compute $\beta(\sigma)$ for various values of σ.

The algorithm was coded in Fortran 77 and run in double precision on the Perkin-Elmer 3230 at the National Bureau of Standards (NBS). Graphs of the fitting functions for all three examples are given in [BogBS85].

Example 1. Consider

$$y = \frac{1}{x - 1}$$

and define $x_i = .01 + (i - 1) * .05$, $i = 1, \ldots, 40$. Next let

$$y_i = \frac{1}{x_i - 1}, \qquad i = 1, \ldots, 40.$$

Now we perturb the data points as follows:

$$x_i := x_i + rx$$
$$y_i := y_i + ry$$

where the rx are uniformly distributed on $(-.05, .05)$ and the ry are uniformly distributed on $(-.25, .25)$. The model for the data was taken to be

$$y = \frac{\beta_1}{x - \beta_2}$$

and the ODR program was run with several values of σ. The results are reported in two tables. Table 1 was generated by setting $\sigma = 1$ and taking $\beta^0 = (1,1)^T$. Subsequent solutions for higher values of σ used the previous solution for the initial approximation. In addition to the values of $\beta(\sigma)$, Table 1 contains the number of evaluations of the extended residual function G (cf (2.3)) and its Jacobian, and the optimal values of $\|R(\eta(\sigma))\|$ and $\|\delta(\sigma))\|$ for each value of σ. Since the value of δ was expected to be approximately the size of the variance of the errors, we set the weight $T = 10$. Table 2 is organized just as Table 1, but the results were generated by starting with the OLS solution using $\beta^0 = (1,1)^T$ and then decreasing σ.

Obviously, Tables 1 and 2 exhibit a nonuniqueness of the solutions. It appears that there are two local solutions for the OLS problem corresponding to the asymptote to $+\infty$ being on the left or right half of the curve, and that the trajectories emanating from these solutions come together around $\sigma = 600$ or that the trajectory represented in Table 2 fails to be continuous near $\sigma = 600$. A possible means of investigating this phenomenon is to write the differential equation describing the trajectory $\beta(\sigma)$ and to study possible bifurcation points. This is not pursued here.

Observe that β_2 determines the location of the asymptote and thus the data locate this parameter very well. The graph of the OLS fit, however, shows that the data point near the asymptote, corresponding to $(1.01, 100)^T$, completely dominates the fitting process for OLS in Table 2 and results in a value of -.3180 for β_1. The ODR fit is not nearly so influenced by this data point and, for a broad range of σ, does a very good job of fitting the data. This last point is important, namely that the parameter values do not vary much as a function of σ, which means that σ may not need to be known with much accuracy. The stability of $\beta(\sigma)$ has been noticed on all of our examples and on problems not reported here. This is not, of course, a proof that this phenomenon holds more generally.

A further difference between the OLS and the ODR fits is that the errors for both the the OLS fits do not appear to be random. The graph of the OLS fit shows that almost all of the errors to the left of the asymptote are negative while all to the right are positive. The ODR errors for reasonable values of σ appear to be much more random.

An examination of the computational results reveals that the only hard optimization problem in each set is the first. Subsequent solutions are found very quickly except, of course, for the problem corresponding to $\sigma = 500$ in Table 2 which appears to have jumped across a discontinuity in $\beta(\sigma)$. A detailed examination of the iteration process shows that the algorithm sometimes slows down (a very small value of τ is generated) but then recovers and final convergence is with full Gauss-Newton steps. For the case $\sigma = 500$, fairly large steps in δ were generated which led to apparent convergence with very poor β values (near $(0,1)$) and very large values of δ ($O(1)$). In this case, a very small value of τ was produced. When the procedure was restarted with a large value of τ, the algorithm immediately stepped over this bad region and converged quickly to the correct solution. Thus, it appears to be important to scale the step in δ correctly and to be on the lookout for unrealistic solutions.

Example 2. This example is a two dimensional version of Example 1. Here we take $x \in R^2$ and

$$y = \frac{1}{x_1 + x_2 - 1}.$$

This function has a line of singularities along $x_1 + x_2 = 1$. We take the data to be on the rectangular grid of width .1 in the x_1 direction and width .2 in the x_2 direction. The first point is $(.01, .01)^T$ and there are 10 points in the x_1 direction and 5 points in the x_2 direction. y is the evaluated at these points and the data are then perturbed according to the following:

$$\begin{aligned}(x_1)_i &:= (x_1)_i + rx\\ (x_2)_i &:= (x_2)_i + rx\\ y_i &:= y_i + ry\end{aligned}$$

where rx are normally distributed with mean 0 and standard deviation .01 and the ry are distributed normally with mean 0 and standard deviation .04.

The form of the model is

$$y = \frac{\beta_1}{\beta_2 x_1 + \beta_3 x_2 - 1}.$$

The results are given in Table 3 which is organized as Table 1. Again the values of $\beta(\sigma)$ do not vary quickly, the location of the asymptote is well-determined by the data, and only β_1 changes much as σ increases. Graphs of the fitting functions show that the fits depend more and more on the points near the asymptote as σ increases. Here the insistence on near vertical measures of the error forces β_1 to assume smaller values which has the effect of flattening the function as much as possible near the asymptote. This, of course, tends to minimize the vertical component of the error. As in Example 1, the errors for the OLS fit do not appear random while those for the ODR fits do.

Note that the first solution, corresponding to $\sigma = 1$, was computed with some difficulty. (This is the same situation as occured for $\sigma = 500$ from Table 2.) In these cases, the terrain in parameter space (β and δ) appears rather flat and fairly large values of δ were again obtained on intermediate iterations. The iteration stalled with an indication of convergence due to x-convergence and a very small value of the trust region radius. A restart (which resets the trust region radius to a larger value) then allows the iterates to step over this flat area and converge very quickly to the correct answer.

The non-uniqueness observed in Example 1 was again observed here. The details are not reported, but we found a second OLS solution which led to a trajectory of solutions that finally joined the above trajectory at $\sigma = 2$.

Example 3. The data here are actual measurements from a calibration run on an electronic device which was intended to give a flat response over a wide range of frequencies. In the (x, y)-data, the x-values are in units of frequency squared and the y-data are the gain. The x-values are scaled to the interval $(0, 1)$ with several measurements made in each decade from 10^{-8} to 1. More measurements were taken at the higher frequencies since most of the important information is obtained there.

The model for this data was obtained from theoretical considerations and has the form

$$y_i = \sum_{j=1}^{4} \frac{\alpha_j}{x_i + \gamma_j} + \mu, \qquad i = 1, \ldots, 44$$

where the parameters to be determined are

$$\beta = (\alpha_1, \ldots, \alpha_4, \mu, \gamma_1, \ldots, \gamma_4)^T.$$

Estimates of the pole locations—the negative γ-values—are likewise obtained from other analyses. The γ-values are approximately

$$\begin{aligned}\gamma_1 &= 1.38 \times 10^{-3}\\ \gamma_2 &= 5.96 \times 10^{-2}\\ \gamma_3 &= 6.71 \times 10^{1}\\ \gamma_4 &= 1.07 \times 10^{9}.\end{aligned}$$

Since all of the poles are negative and all of the data have positive x-values, there is no problem with being close to the asymptotes. The range of the x-values, however, implies the need to scale the trust region. We used, for the diagonal scaling matrices S and T, the following:

$$\begin{aligned}s_i &= \frac{1}{|(\beta^0)_i|}\\ t_i &= \frac{1}{|x_i|}.\end{aligned}$$

It turns out that the measurements are proportionately more accurate at the lower frequencies and we therefore took the d-weights to be the same as the t-weights.

While the data were measured quite accurately, there were simply no data at a sufficiently high frequency to warrant keeping the two terms corresponding to $j = 3$ and $j = 4$ in the model. This situation was evidenced by the fact that the Jacobian J had five almost identical columns.

With these terms removed, the resulting problem was easily solved as follows. Using a feature of the program which allows certain parameters to be held fixed at specified values, we fixed the pole values (the γ-values) and used an OLS estimate of the remaining linear parameters. We then freed all of the parameters and did an OLS fit and and ODR fits with several values of σ. In doing the ODR fits, we first specified a σ-value of .01 since the gain measurements in this data set were 100 times more accurate than the frequency measurements. Other values of σ were subsequently used for comparison. The results are in Table 4. Virtually no difference appears between the two fits at the lower frequencies, but some differences occur at the higher frequencies. In the enlargements of the fitting functions, one can easily see that the contribution of the error in the x-values causes ODR to get a significantly better fit than OLS. While the β-values are not reported here, there were, again, very slow changes in $\beta(\sigma)$.

In this section we have shown that our algorithm is effective on highly nonlinear problems, but that these problems themselves often have multiple solutions and other difficulties which imply that potential solutions need to be studied carefully. In subsequent papers, we will provide a more complete description of our implementation and further results on its performance.

Acknowledgements. The authors wish to thank E. Bromberg and T. Griffin of the Center for Applied Mathematics, National Bureau of Standards, for their help in the graphics aspects of these experiments and T. Griffin for his help in the writing of the data handling and driver routines.

Table 1

			Evals of		Final Values	
σ	$\beta_1(\sigma)$	$\beta_2(\sigma)$	G	G'	$\|R(\eta(\sigma))\|$	$\|\delta(\sigma)\|$
1	1.023	1.006	70	25	0.223	0.355
2	1.021	1.005	6	5	0.454	0.223
5	1.015	1.004	6	4	0.771	0.128
25	0.9847	1.002	6	5	1.280	0.080
100	0.9247	0.9972	9	8	3.204	0.061
300	0.9881	0.9928	13	12	10.408	0.035
500	0.9487	0.9953	12	11	15.524	0.018
1000	0.8248	0.9937	10	9	18.881	0.007
∞_1	0.6867	0.9909	7	6	21.774	0.

Table 2

			Evals of		Final Values	
σ	$\beta_1(\sigma)$	$\beta_2(\sigma)$	G	G'	$\|R(\eta(\sigma))\|$	$\|\delta(\sigma)\|$
∞_2	-0.3170	1.010	40	22	104.709	0.
1000	-0.3355	1.095	20	15	104.223	0.007
700	-0.3845	1.093	27	21	103.660	0.015
500*	0.9487	0.9953	103	43	15.524	0.018

Table 3

				Evals of		Final Values	
σ	$\beta_1(\sigma)$	$\beta_2(\sigma)$	$\beta_3(\sigma)$	G	G'	$\|R(\eta(\sigma))\|$	$\|\delta(\sigma)\|$
1*	0.8988	0.9482	1.015	147	60	0.184	0.670
2	0.9223	0.9478	1.019	7	6	0.428	0.618
4	0.9345	0.9506	1.027	8	7	0.989	0.540
10	0.9049	0.9510	1.047	9	8	2.379	0.429
40	0.7148	0.9568	1.044	10	9	6.411	0.315
100	0.3645	0.9343	0.9894	22	16	19.934	0.174
500	0.0914	0.8830	0.9675	25	17	30.424	0.039
∞	0.1192	0.8883	0.9338	27	12	77.440	0.

Table 4

	Evals of		Final Values	
σ	G	G'	$\|R(\eta(\sigma))\|$	$\|\delta(\sigma)\|$
∞	19	9	1.8702	0.
.01	18	7	0.0005	0.03993
.1	12	7	0.0016	0.00482
1.	5	4	0.0018	0.00006

REFERENCES

[Bar74] Bard, Y. (1974), *Nonlinear Parameter Estimation*, Academic Press, New York.

[BogBS85] Boggs, P. T., R. H. Byrd, and R. B. Schnabel (1985), "A stable and efficient algorithm for nonlinear orthogonal distance regression," University of Colorado Technical Report CU-CS-317-85.

[BriL73] Britt, H. I., and R. H. Luecke (1973), "The estimation of parameters in nonlinear, implicit models," *Technometrics* **15**, 233-247.

[ByrS86] Byrd, R. H., and R. B. Schnabel (1985), "A unified local and global convergence analysis of Levenberg-Marquardt methods," (in preparation).

[DenGW81] Dennis, J. E., Jr., D. M. Gay, and R. E. Welsch (1981), "An adaptive nonlinear least-squares algorithm," *TOMS* **7**, 348-368.

[DenS83] Dennis, J. E., Jr., and R. B. Schnabel (1983), *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice Hall, Englewood Cliffs, N.J.

[FiaM68] Fiacco, A. V., and G. P. McCormick (1968), *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*, John Wiley, New York.

[Ful86] Fuller, W. A. (1986) *Measurement Error Models*, John Wiley, New York.

[GolV83] Golub, G. H., and C. Van Loan (1983), *Matrix Computations*, The Johns- Hopkins University Press.

[Heb73] Hebden, M. D. (1973), "An algorithm for minimization using exact second derivatives," Rept. TP515m A.E.R.E., Harwell, England.

[KenSO83] Kendall, M. G., A. Stewart, and J. K. Ord (1983), *The Advanced Theory of Statistics, Fourth Edition*, MacMillan Publishing, New York.

[LawH74] Lawson, C. L., and R. J. Hanson (1974), *Solving Least Squares Problems*, Prentice Hall, Englewood Cliffs, N.J.

[Lue73] Luenberger, D. G. (1973), *Introduction to Linear and Nonlinear Programming*, Addison Wesley, Reading, Massachusetts.

[Mor71] Moran, T. A. P. (1971), "Estimating Structural and functional relationships," *J. Mult. Anal.*, **1**, 232-255.

[Mor77] Moré, J. J. (1977), "The Levenberg-Marquardt algorithm: implementation and theory," in *Numerical Analysis*, G. A. Watson, ed., Lecture Notes in Math. 630, Springer Verlag, Berlin, 105-116.

[MorS81] Moré, J. J., and D. C. Sorenson (1981),"Computing a trust region step," *SIAM J. Sci. Stat. Comput.* **4**, 553-572.

[OrtR70] Ortega, J. M., and W. C. Rheinboldt (1970), *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, New York.

[Pow75] Powell, M. J. D. (1975),"Convergence properties of a class of minimization algorithms," in *Nonlinear Programming* **2**, O. Mangasarian, R. Meyer, and S. Robinson, eds., Academic Press, New york, 1-27.

[SchSB85] Schultz, G. A., R. B. Schnabel, and R. H. Byrd (1985),"A family of trust- region algorithms for unconstrained minimization with strong global convergence properties," *SIAM J. Numer. Anal.* **22**, 47-67.

[SchT85] Schwetlick, H., and V. Tiller (1985),"Numerical methods for estimating parameters in nonlinear models with errors in the variables," *Technometrics* **27**, 17-24.

[Ste73] Stewart, G. W., III (1973) *Introduction to Matrix Computations*, Academic Press, New York.

CONTRIBUTED PAPERS

These contributed papers were presented in six sessions and are arranged alphabetically by first author's last name.

THE APPLICATION OF CONVEX HULLS IN MULTIPLE DIMENSIONS

Max Benson, University of Minnesota, Duluth
Kinley Larntz, University of Minnesota, Twin Cities
Michael Lavine, University of Minnesota, Twin Cities
Ronald Regal, University of Minnesota, Duluth

Convex hulls and minimum covering ellipsoids are two possible methods for defining the region of space enclosed by given points. Algorithms for computing these regions in multidimensional space were implemented and investigated. The minimum covering ellipsoid implementation takes much less time and storage than the convex hull. The expected probability content for the convex hulls in higher dimensions is small.

KEY WORDS: Convex Hull; Minimum Covering Ellipsoid

1. INTRODUCTION

The computation of a convex region containing a set of points has been extensively studied and applied in two dimensions. Both convex hulls and minimum covering ellipsoids have been used in statistical applications. The convex hull of a set of points is the smallest convex region containing all of the points. The minimum covering ellipsoid is the ellipsoid of smallest content that contains the points. Figure 1 shows the convex hull and the minimum covering ellipsoid for 20 points in two dimensions. Although convex hulls and minimum content ellipsoids have applications in more than two dimensions, programs for computing these regions in multidimensional cases are not readily available. In this study we report the implementation of programs to compute convex hulls and minimum covering ellipsoids in multiple dimensions. The time and space requirements of the implementations are investigated empirically for random points from normal and uniform distributions.

2. CONVEX HULLS

2.1 Uses

Convex hulls have been used in a number of statistical contexts. Kendall (1966) suggested that in a discrimination context a new observation could be assigned to be in the same class as a set of points if the new point was in the convex hull of the given class of points. Kendall also suggested finding the extent to which groups of points formed distinct classes by finding how many of the points in other groups were in the convex hull of a given group. Convex hulls have also been used in various versions of peeling or trimming multivariate data. Barnett (1976) discusses using convex hulls for ordering multivariate data. According to Huber (1972) the idea was originally suggested by Tukey. The idea is to take all points on the convex hull as the analog of the extreme order statistics. By peeling off successive outer layers of points, one can define robust estimators of location (Seheult, Diggle and Evans, 1976) or robust estimators of the correlation coefficient (Bebbington, 1978). Convex hulls have also been used in constructing non-parametric estimators of densities and modes (Eddy and Hartigan, 1977).

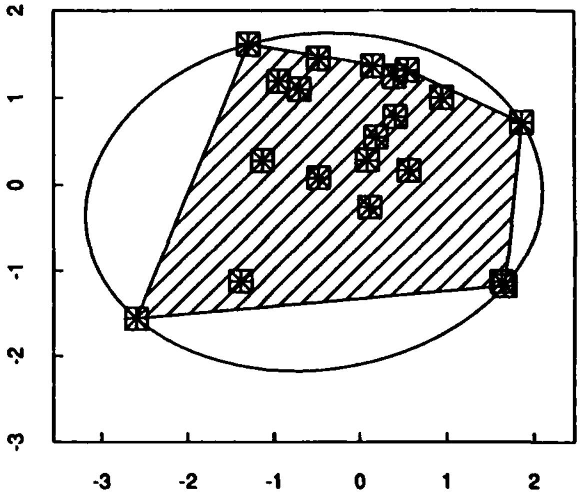


Figure 1. *Convex Hull and Minimum Covering Ellipse for n=20 Points in d=2 Dimensions. The convex hull is the shaded region.*

2.2 Computation

The computation of convex hulls in two dimensions is well studied, and algorithms are readily available. For example, the S software (Becker and Chambers, 1984) includes a function to find a two dimensional hull. Preparata and Shamos (1985) describe several methods for finding planar convex hulls. Examples include the quicksort analog first published by Eddy (1977). In two dimensions such algorithms provide easily implemented, reasonably efficient algorithms. However the complexity of the problem escalates tremendously when one steps up to the general multidimensional case. Preparata and Shamos (1985) describe two alternatives for the general case, the gift-wrapping method of Chand and

Kapur (1970) and the beneath-beyond method of Kallay (1981).

The algorithm implemented here is the gift-wrapping algorithm. The gift-wrapping analogy comes from thinking of the way one rotates a package from face to face by pivoting on edges of the package. The reason for our choice of this algorithm is historical. The interest in this project started when the Environmental Protection Agency Research Lab in Duluth, Minnesota, wanted a convex hull program to use in discrimination much in the way suggested by Kendall (1966). The EPA lab had tracked down a FORTRAN program written by Chand and Kapur at Lockheed-Georgia. Unfortunately the program was written in FORTRAN 66, and the code was not easy to follow or generalize. Rather than wallowing in FORTRAN 66 code, Benson implemented the algorithm from scratch in the programming language C. Benson's implementation is pointer based and has no upper limits (in theory) to the number of dimensions. The programs contain over 1800 lines of code and have been run on SUN and VAX computers using UNIX operating systems.

The primary drawback to the computations is that convex hulls in higher dimensions become very complex. Even if all faces are simplicial, the worst case time complexity of the gift-wrapping technique for n points in p dimensions is $O(n^{\lfloor p/2\rfloor+1}) + O(n^{\lfloor p/2\rfloor}\log n)$ and the worst-case number of faces is $O(n^{\lfloor p/2\rfloor})$ (Preparata and Shamos, 1985, page 130). Hence both the time required to compute the hull and the space required to store the results grow quickly as the dimensionality increases. Swart (1985) has suggested some possible refinements of the Chand and Kapur algorithm. Swart's suggestions involve storing some intermediate results rather than recalculating them. Benson is working on other modifications, but none of these modifications were used here.

3. MINIMUM COVERING ELLIPSES

3.1 Uses

The minimum covering ellipsoid (MCE) is the smallest content ellipsoid covering a set of points. Titterington (1975) introduced the MCE and described its relationship to optimal design. The idea of peeling or trimming multivariate data can be carried out with the MCE in place of the convex hull. Titterington (1978) discusses robust estimation of the correlation coefficient using elliptical trimming. Green (1981) discusses both convex hull and elliptical peeling. Cook and Weisberg (1978) describe using the MCE to define the region of applicability (interpolation region) of the independent variables in multiple regression. The MCE can be substituted for the convex hull in Kendall's (1966) suggestion for discrimination.

3.2 Computation

An iterative algorithm is described by Titterington (1978), and modifications are given in Silvey, Titterington and Silvey (1978). A noniterative algorithm for two dimensional ellipsoids is given by Silverman and Titterington (1980). The ellipsoid is described by two parameters, the center μ and the quadratic form Q. Points on the boundary of the ellipsoid satisfy the equation

$$(\mathbf{x}-\mu)^T \mathbf{Q}(\mathbf{x}-\mu) = p$$

The parameters are found by an iterative reweighting scheme. Let $\mathbf{x}_1, \ldots \mathbf{x}_n$ be n points in $\mathbf{R}^p$. The algorithm initially gives weight 1/n to each point and then computes

$$\mu = \sum w_i \mathbf{x}_i \tag{3.1}$$

$$Q = \left[\sum w_i(\mathbf{x}_i-\mu)^T(\mathbf{x}_i-\mu)\right]^{-1} \tag{3.2}$$

Next the ellipsoid is checked for each point by defining

$$d_i = (\mathbf{x}_i-\mu)^T \mathbf{Q}(\mathbf{x}_i-\mu)$$

and $d_{max} = max(d_1, d_2, \ldots d_n)$. If $d_{max} < p+\epsilon$, the search stops. Otherwise new weights are computed according to $w_i = (d_i/p)w_i$ and these are used to update μ and Q according to (3.1) and (3.2). In our simulation study described below, a value of 0.01 was used for ϵ.

4. SIMULATION STUDY

4.1 Design

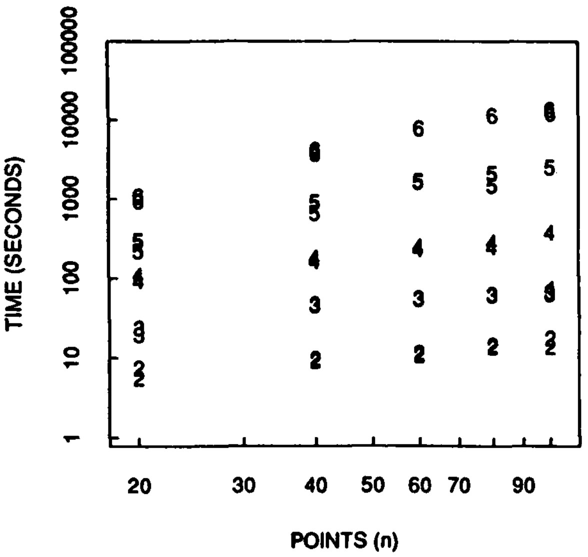
Both the convex hull and the minimum covering ellipsoid were computed under conditions defined by three factors:

(1) Distribution - Normal (0, I)
Uniform (on the unit cube)
(2) Points (n) - 20, 40, 60, 80, 100
(3) Dimension (p) - 2, 3, 4, 5, 6.

The uniform variates were generated in a cube of dimension p. The normal samples are transformed versions of the uniform variates. All computations were done on a VAX11/750.

For each combination of these factors we observed the time to compute the convex hull, the number of faces and vertices of the convex hull, the time to compute the MCE and the number of points on the surface of the MCE.

Figure 2. *Computation Times for Convex Hulls of Random Uniform Points on the Unit Cube. The text plotted for each result is the dimension, p.*



4.3 Results

Figure 2 shows the times required to compute convex hulls for random uniform variates in the unit cube. In some cases only one replication is shown because three of the runs ran out of space. The runs for 100 points in 6 dimensions took over 3 hours of cpu time. Figure 3 shows the corresponding times for the normal data. The time to compute hulls of random normal data is less, but the runs for 100 points in 6 dimensions still take nearly 3 hours. In contrast, Figure 4 shows the times for the MCE for normal data. None of the times is more than 2 minutes. The convex hull is time consuming to compute with the gift-wrapping algorithm because the hulls become very complex with many faces to find. In order to determine if a new point is in the convex hull, one needs to retain information for each face. Figure 5 shows the numbers of faces for the hulls of normal data. Even for 40 points in 6 dimensions, the convex hulls have over 1000 faces. For 100 points in 6 dimensions, the hulls have over 3000 faces. In dimensions 2 and 3 the expected number of faces can be found using results from Efron (1965). The expected values are given by stars on Figure 5. The simulated results are in good agreement with the expected values.

Figure 6 shows the results for the number of vertices on the hull, except that the number of vertices is converted to percent of n, the number of points. For 40 points in 6 dimensions, for example, over 90% of the points were vertex points of the convex hull. Even for 100 points in 6 d'mensions, about 70% of the points are vertex points. In contrast Figure 7 shows that the percentage of points on the surface of the MCE is much smaller. For example only about 10% of the 100 points in p=4 dimensions are on the surface of the MCE.

Figure 3. *Computation Times for Convex Hulls of Random Normal Points. The text plotted for each result is the dimension, p.*

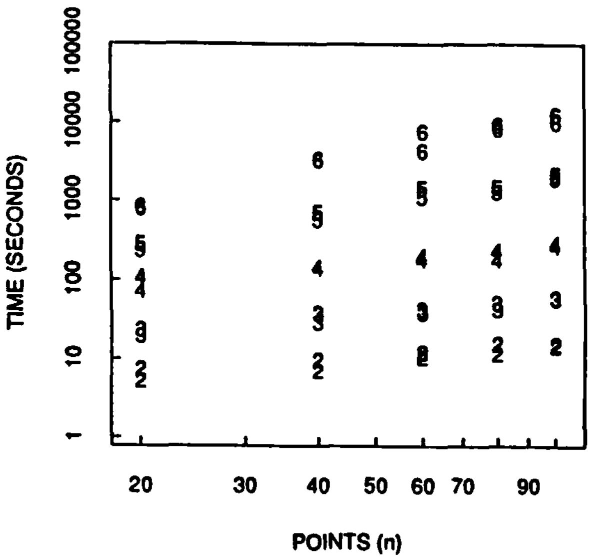


Figure 4. *Computation Times for Minimum Covering Ellipsoids for Random Normal Points. The text plotted for each result is the dimension, p.*

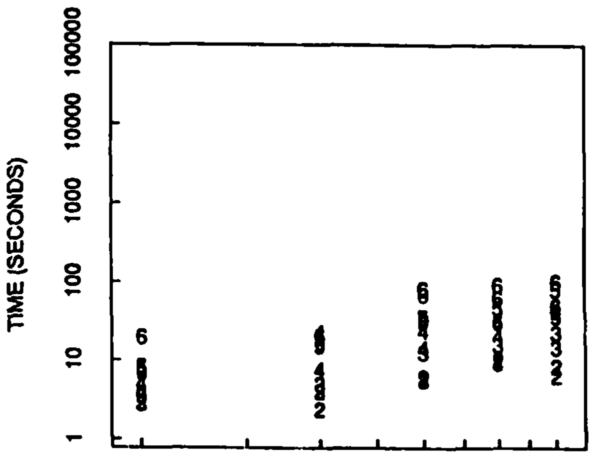
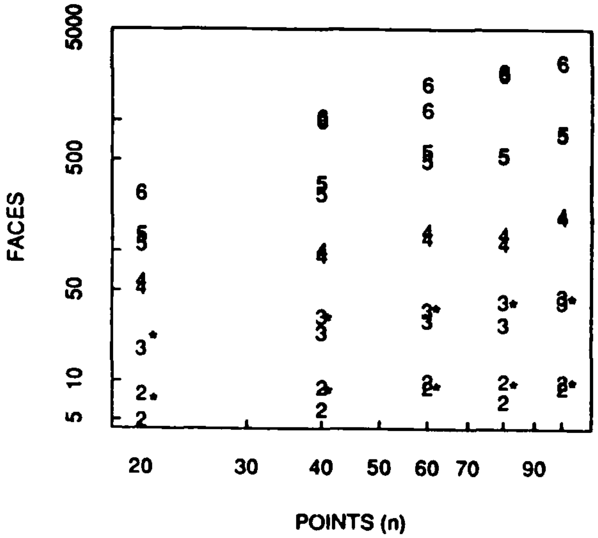


Figure 5. *Number of Faces for Convex Hulls of Random Normal Points. The text plotted for each result is the dimension, p. A * indicates the expected value for 2 and 3 dimensions.*



5. DISCUSSION

Convex hulls in higher dimensions are complex with many faces. The time to find the hull and space required to save the results could be enormous. The minimum covering ellipsoids were found in much less time, and the results require saving only the center and quadratic form of the ellipsoid. In higher dimensions the convex hull can be mostly vertex points. This has important implications for the potential use of convex hulls in peeling or discrimination. Clearly, peeling is not useful if most of the points are in the outside layer. In a discrimination context, if most of the points are vertex points, the expected probability content of the hull would be estimated to be small. Such a hull would have a small chance of containing a new point drawn from the same distribution. More explicitly, the expected probability content of a convex hull of n-1 points is $E(V)/n$ where E(V) is the expected number of vertices of a hull of n point. This is related to the observation that a point is in the convex hull of the other n-1 points if and only if it is not a vertex point for the hull of the original n points. From the results here one sees that a very large number of points is going to be needed in high dimensions in order for the convex hull to contain much of the probability space. By contrast the minimum covering ellipsoid has many fewer points on the surface and therefore has a much higher estimated probability content.

Figure 6. *Percent of Points on Surface of Convex Hulls of Random Normal Points.*

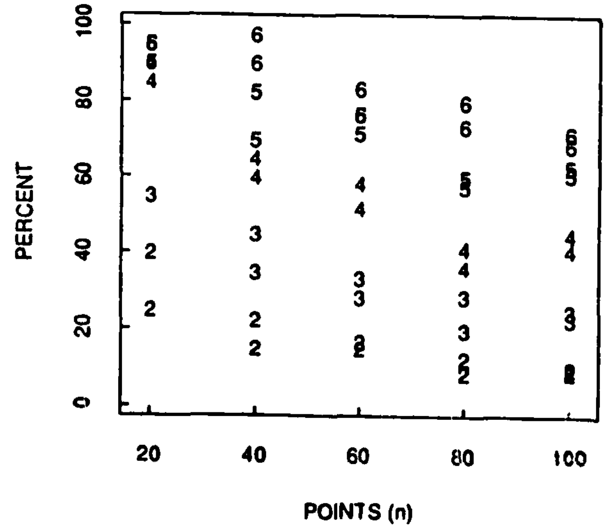
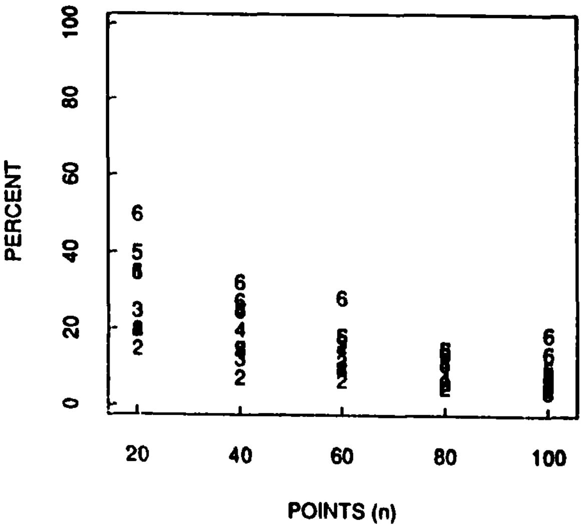


Figure 7. *Percent of Points on the Surface of Minimum Covering Ellipsoids of Random Normal Points.*



ACKNOWLEDGEMENTS

This research was supported by cooperative agreement CR-811981 between the U. S. Environmental Protection Agency and the University of Minnesota.

REFERENCES

Barnett, V. (1976), "The Ordering of Multivariate Data," *J. R. Statist. Soc. A*, 139, 318-344.

Bebbington, A.C. (1978), "A Method of Bivariate Trimming for Robust Estimation of the Correlation Coefficient," *Appl. Statist.*, 27, 221-226.

Becker, R.A. and Chambers, J.M. (1985), *S: An Interactive Environment for Data Analysis and Graphics*, Belmont, CA: Wadsworth.

Chand, D.R. and Kapur, S.S. (1970), "An Algorithm for Convex Polytopes," *JACM*, 17, 78-86.

Cook, R.D. and Weisberg, S. (1978), "Characterizing Extrapolations in Multiple Regression," Technical Report 330, School of Statistics, St. Paul.

Eddy, W.F. (1977), "A New Convex Hull Algorithm for Planar Sets," *ACM Trans. Math. Software*, 3, 398-403.

Eddy, W.F. and Hartigan, J.A. (1977), "Uniform Convergence of the Empirical Distribution Function over Convex Sets," *Ann. Statist.*, 5, 370-374.

Efron, B. (1965), "The Convex Hull of a Random Set of Points," *Biometrika*, 52, 331-343.

Green, P.J. (1981), "Peeling bivariate data," in *Interpreting Multivariate Data*, Vic Barnett (Ed), New York: Wiley.

Huber, P.J. (1972), "Robust Statistics: A Review," *Ann. Math. Statist.*, 43, 1041-1067.

Kallay, M. (1981), "Convex Hull Algorithms in Higher Dimensions," unpublished manuscript, Dept. of Mathematics, Univ. Of Oklahoma.

Kendall, M.G. (1966), "Discrimination and Classification," in *Multivariate Analysis*, P.P. Krishnaiah (Ed), New York: Academic Press.

Preparata, F.P. and Shamos, M.I. (1985), *Computational Geometry*, New York: Springer-Verlag.

Seheult, A., Diggle, P., and Evans, D. (1976), Discussion of "The Ordering of Multivariate Data", *J. R. Statist. Soc. A*, 139, 351-352.

Silverman, B.W. and Titterington, D.M. (1979), "Minimum Covering Ellipses," *Siam J. Sci. Stat. Comput.*, 1, 401-409.

Silvey, S.D., Titterington, D.M. and Torsney, B. (1978), "An Algorithm for Optimal Designs on a Finite Design Space," *Commun. Statist. A*,7, 1379-1389.

Swart, G. (1985), "Finding the Convex Hull Facet by Facet," *Journal of Algorithms*, 6, 17-48.

Titterington, D.M. (1975), "Optimal Design: Some Geometrical Aspects of D-Optimality," *Biometrika*, 62, 313-320.

Titterington, D.M. (1978), "Estimation of Correlation Coefficients by Ellipsoidal Trimming," *Applied Statistics*, 27, 227-234.

A JOHNSON CURVE APPROACH TO WARMING UP TIME SERIES SIMULATIONS

David A. Burn, IMSL, Inc.

The simulation of a time series from a specified autoregressive moving average (ARMA) model requires knowledge of the initial values of the time series and/or innovations process. Given independent, identically distributed zero mean innovations, the start values of the time series are often derived from a moving average approximation of the series. This approximation introduces bias of a transient nature into the system, and requires the simulation to be run for a period of time in order to diminish the influence of the initial values of the series. To avoid the necessity of warming up the simulation, we consider a Johnson curve approximation of the distribution of the initial values of the time series.

KEY WORDS: Autoregressive, Moving average, Skewness, Kurtosis, Transient.

1. INTRODUCTION

1.1 General ARMA Model

Define the general form of the ARMA(p, q) model by

$$\phi_p(B)W_t = \theta_0 + \theta_q(B)A_t \tag{1}$$

where

$$\begin{aligned} \phi_p(B) &= 1 - \phi_1 B - \phi_2 B^2 - \cdots - \phi_p B^p, \qquad p \geq 0 \quad (2) \\ \theta_q(B) &= 1 - \theta_1 B - \theta_2 B^2 - \cdots - \theta_q B^q, \qquad q \geq 0 \quad (3) \end{aligned}$$

and B is the backward shift operator defined by

$$B^k W_t = W_{t-k}, \qquad \text{for all } k.$$

This definition includes the following assumptions:

1. The innovations A_t are independent and identically distributed random variables with mean zero and variance σ_A^2.

2. The autoregressive operator $\phi_p(B)$ is stationary. Equivalently, the roots of the equation $\phi_p(B) = 0$ lie outside the unit circle.

3. The moving average operator $\theta_q(B)$ is invertible. Equivalently, the roots of the equation $\theta_q(B) = 0$ lie outside the unit circle.

The model is general in that the constant term θ_0 is included to allow for a nonzero series mean μ. Refer to Box and Jenkins (1976, pp. 91–93) for further discussion.

1.2 Equivalent Representations

The random shock form of the general ARMA(p, q) is given by

$$W_t = \mu + \psi_p(B)A_t \tag{4}$$

where

$$\psi_p(B) = \phi_p^{-1}(B)\theta_q(B) = 1 + \psi_1 B + \psi_2 B^2 + \cdots . \tag{5}$$

The ψ weights of the infinite order moving average may be determined by equating coefficients of B in

$$\phi_p(B)\psi_p(B) = \theta_q(B)$$

(see Box and Jenkins, 1976, pp. 95–96). The random shock model is particularly useful since the moments of the time series W_t can be derived for a specified distribution of the innovations A_t. Note that the general ARMA(p, q) model and its random shock form may be equivalently expressed as

$$\phi_p(B)\tilde{W}_t = \theta_q(B)A_t \tag{6}$$

and

$$\tilde{W}_t = \psi_p(B)A_t \tag{7}$$

respectively, where $\tilde{W}_t = W_t - \mu$ corresponds to a time series with zero mean.

1.3 The Simulation Problem

Suppose we wish to simulate a time series W_t of length n according to a specified ARMA(p, q) model. The *induction period* is the length of time required to minimize the transient bias induced by starting the run (Anderson, 1975). Let the total number of generated observations of the time series be $m + n$ where the simulation is warmed up with m discardable observations of the time series. Clearly, we desire as short an induction period m as possible.

The method of generating the initial values of the time series and innovations process required to start the simulation run directly affects the induction period. Since the innovations are assumed to be independent, the moving average part of the model may be easily initialized with $q+1$ pseudo-random numbers from the specified distribution of A_t. However, the W_t are not independent, so that production of the p series start values is a major problem of the simulation experiment. We identify two general approaches to this problem:

Approach A Generate the initial series values from an *approximation* of the *model* of the time series.

Approach B Generate the initial series values from an *approximation* of the *joint distribution* of the time series.

Other methods have been proposed; for example, see Piccolo and Wilson (1984).

A prototypical algorithm to simulate a time series consists of the following steps:

```
ALGORITHM ARMA

1. Generate $W_{1-p}, W_{2-p}, \ldots, W_0$.
2. Generate $A_{1-q}, A_{2-q}, \ldots, A_0$.
3. Set t=1.
4. Generate $A_t$.
5. Compute $W_t$ using ARMA($p$, $q$) model.
```

6. Set $t = t + 1$.

7. Repeat Step 4 through Step 6 until $t = m + n$.

The desired time series is given by W_t for $t = m+1, \ldots, m+n$. Some particular applications of Approach A and Approach B are next examined.

2. WARMING UP

2.1 A Finite Approximation

One version of Approach A utilizes a finite order moving average approximation of the general ARMA(p, q) model,

$$W_t = \sum_{j=0}^{m} \psi_j A_{t-j} \tag{8}$$

where m is chosen sufficiently large. (We assume $\theta_0 = 0$ in this section.) The algorithm consists of the following steps:

ALGORITHM A

1. Determine m.

2. Generate $m + p$ pseudo-random numbers $A_{1-(m+p)}$, $A_{2-(m+p)}, \ldots, A_0$ from the innovations distribution.

3. Construct p series start values $W_{1-p}, W_{2-p}, \ldots, W_0$ using the MA(m) model (8) and the innovations from Step 2. Discard the innovations.

Often m is chosen arbitrarily and the simulation may be warmed up longer than necessary.

2.2 Determination of Induction Period

A "precise" method of determining the optimal length of the induction period was proposed by Anderson (1979). Consider the AR(1) model

$$W_t = \phi W_{t-1} + A_t, \qquad |\phi| < 1 \tag{9}$$

and its associated MA(m) approximation

$$W_t = \sum_{j=0}^{m} \phi^j A_{t-j}. \tag{10}$$

Anderson (1975) states that m should be chosen such that the variance of the transient bias

$$\left(\sum_{j=m+1}^{\infty} \phi^{2j} \right) \sigma_A^2 \tag{11}$$

is sufficiently small. However, this approach may be deceptive and lead to an excessively long induction period (Anderson, 1979).

Instead, Anderson considers $\hat{W}_t = W_t - \overline{W}$ where W_t is the AR(1) series generated by ALGORITHM ARMA and

$$\begin{aligned} \overline{W} &= \frac{1}{n} \sum_{t=m+1}^{m+n} W_t \\ &= \frac{\phi^{m+1}(1-\phi^n)}{n(1-\phi)} W_0 + g(m,n,\phi,A) \end{aligned} \tag{12}$$

where A represents the innovations $A_1, \ldots, A_{m+n}$. Then

$$\hat{W}_t = H_t(m,n,\phi)W_0 + h_t(m,n,\phi,A) \tag{13}$$

where

$$H_t(m,n,\phi) = \phi^{m+1} \left[\phi^{t-1} - \frac{(1-\phi^n)}{n(1-\phi)} \right] \tag{14}$$

and the function $h_t(m,n,\phi,A)$ is a not affected by the behavior of the series start value W_0. Hence, the dependence of the simulated series upon the initial series value is minimized by selecting m such that

$$|H_t(m,n,\phi)| < \epsilon, \qquad \epsilon > 0. \tag{15}$$

Since $H_t(m,n,\phi)$ is maximized at $t = 1$, the induction period m is determined by the inequality

$$m > \frac{\ln \left\{ \epsilon \left| 1 - \frac{(1-\phi^n)}{n(1-\phi)} \right| \right\}}{\ln|\phi|} - 1, \qquad \epsilon > 0. \tag{16}$$

For a given $\epsilon > 0$, $m = m(\phi, n)$ is a function of the parameter ϕ and the length of the simulated series n.

Anderson (1979) states that this method extends to the general ARMA(p, q) model. Similar to (13), we have

$$\begin{aligned} \hat{W}_t &= \sum_{j=1}^{p} H_{t,j}(m,n,\phi_1,\ldots,\phi_p)W_{1-j} \\ &\quad + \sum_{k=0}^{q} h_{t,k}(m,n,\phi_1,\ldots,\phi_p,\theta_1,\ldots,\theta_q,A) \end{aligned} \tag{17}$$

where A represents the initial moving average innovations A_{1-q}, $A_{2-q}, \ldots, A_0$ in addition to the innovations $A_1, \ldots, A_{m+n}$. The induction period m is dependent upon the length of the simulated series n through the functions $H_{t,j}(m,n,\phi_1,\ldots,\phi_p)$, and is minimized when their sum is negligible. However, the form of this dependence is quite complicated for models with $p > 1$.

The need to warm up the simulation, and hence the need to select an optimal value of the induction period, are artifacts of the moving average approximation of the series start values. To bypass both of these problems, we derive a method of constucting the initial values of the series directly from the joint distribution of the time series.

3. THE JOHNSON SYSTEM

3.1 Methods of Translation

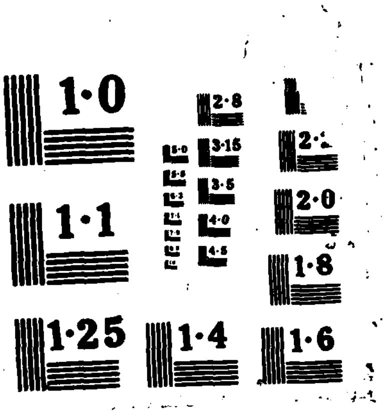
To provide a mathematical representation of a wide variety of statistical distributions, Johnson(1949) proposed a family of frequency curves generated by methods of translation. Let x denote a random variable whose distribution we wish to model, and let z represent a standard normal random variable. Consider the *infinite class of transformations*

$$z = \gamma + \delta f\left(\frac{x-\xi}{\lambda}\right)$$

where f is a monotone function of x and is dependent only upon fixed parameters. Define the standard form of this transformation to be

$$z = \gamma + \delta f(y)$$

AD-A191 296 COMPUTER SCIENCE AND STATISTICS PROCEEDINGS OF THE 4/5
SYMPOSIUM ON THE INTER..(U) COLORADO STATE UNIV FORT
COLLINS T J BOARDMAN 26 AUG 87 AFOSR-TR-88-0153
UNCLASSIFIED AFOSR-86-0070 F/G 12/6 NL


1·0
2·8
1·1
1·8
2·0
3·5
4·0
4·5
1·25
1·4
1·6

where

$$y = (z - \xi)/\lambda.$$

Then the density function of y is given by

$$\begin{aligned} p(y) &= \delta f'(y) p(z)\Big|_{z=\gamma+\delta f(y)} && (18) \\ &= \frac{\delta}{\sqrt{2\pi}} f'(y) \exp\left\{-\tfrac{1}{2}[\gamma + \delta f(y)]^2\right\} && (19) \end{aligned}$$

and has the same shape as the density function of z. The parameters ξ and λ are location and scale factors, respectively, and the parameters δ and γ affect the skewness and kurtosis, respectively.

3.2 Systems of Interest

The special systems described by Johnson are

1. Lognormal, S_L

$$z = \gamma + \delta \ln\left(\frac{x - \xi}{\lambda}\right) \qquad \xi < x < \infty.$$

2. Bounded, S_B

$$z = \gamma + \delta \ln\left(\frac{x - \xi}{\xi + \lambda - x}\right), \qquad \xi < x < \xi + \lambda.$$

3. Unbounded, S_U

$$z = \gamma + \delta \sinh^{-1}\left(\frac{x - \xi}{\lambda}\right), \qquad -\infty \le x \le \infty.$$

These three systems encompass most of the distributions common to statistical analysis. For information concerning alternative systems of distributions, see Johnson (1949), Elderton and Johnson (1965), Kendall and Stuart (1969), and Ord (1972).

3.3 Fitting by Moments

To fit a Johnson curve, we derive the mean, standard deviation, coefficient of skewness, and coefficient of kurtosis of the specified distribution. Next, the appropriate system and parameters of the Johnson curve representation of the specified distribution are determined using algorithm AS 99 (Hill, Hill, and Holder, 1985). Pseudo-random observations from the fitted Johnson curve are obtained by transforming a pseudo-random standard normal variate to a Johnson variate using algorithm AS 100 (Hill, 1985).

For the simulation experiment, we require only the first four moments of the theoretical distribution to determine a corresponding representation within the Johnson system. Since the moments of the distribution to be modelled are calculated theoretically, no sampling error is introduced. Also, we are mainly interested in first and second order properties of the simulated time series. Hence, we view the method of moments approach to fitting a Johnson curve to be acceptable. A number of alternative methods of estimating the parameters of a Johnson curve are discussed by Johnson (1949), Elderton and Johnson (1965), and Ord (1972).

3.4 Series Moments

The relationship between the central moments of the time series and the central moments of the innovations process is given by

$$\begin{aligned} \mu_1(W) &= \mu_1(A) \sum_{j=0}^{\infty} \psi_j \\ \mu_2(W) &= \mu_2(A) \sum_{j=0}^{\infty} \psi_j^2 \\ \mu_3(W) &= \mu_3(A) \sum_{j=0}^{\infty} \psi_j^3 \\ \mu_4(W) &= \mu_4(A) \sum_{j=0}^{\infty} \psi_j^4 + 3[\mu_2(A)]^2 \sum_{j \neq k} \psi_j^2 \psi_k^2 \\ &= \left\{\mu_4(A) - 3[\mu_2(A)]^2\right\} \sum_{j=0}^{\infty} \psi_j^4 + 3[\mu_2(A)]^2 \left(\sum_{j=0}^{\infty} \psi_j^2\right)^2 \end{aligned}$$

where $\mu_i(W)$ and $\mu_i(A)$ denote the ith central moment of W_t and A_t respectively, for $i = 1, 2, 3, 4$. The mean, standard deviation, coefficient of skewness, and coefficient of kurtosis of the distribution of W_t are defined by

$$\begin{aligned} \mu &= \mu_1(W) \\ \sigma &= [\mu_2(W)]^{1/2} \\ \sqrt{\beta_1} &= \mu_3(W)/[\mu_2(W)]^{3/2} \\ \beta_2 &= \mu_4(W)/[\mu_2(W)]^2. \end{aligned}$$

Hence, for independent and identically distributed $(0, \sigma_A^2)$ innovations,

$$\begin{aligned} \mu &= 0 && (20) \\ \sigma &= \sigma_A \left(\sum_{j=0}^{\infty} \psi_j^2\right)^{1/2} && (21) \\ \sqrt{\beta_1} &= \sqrt{\beta_1(A)} \frac{\sum_{j=0}^{\infty} \psi_j^3}{\left(\sum_{j=0}^{\infty} \psi_j^2\right)^{3/2}} && (22) \\ \beta_2 &= [\beta_2(A) - 3] \frac{\sum_{j=0}^{\infty} \psi_j^4}{\left(\sum_{j=0}^{\infty} \psi_j^2\right)^2} + 3. && (23) \end{aligned}$$

The stationarity and invertibility assumptions imply that the infinite sums of the ψ_j are absolutely convergent. Davies, Spedding, and Watson (1980) state that for low order ARMA models, at most 30 of the ψ_j are required to compute the above moments.

4. GENERATION OF START VALUES

4.1 A Simple Example

We introduce the Johnson Curve approach to generating the initial values of a time series in the context of the AR(1) model

$$W_t = \phi W_{t-1} + A_t, \qquad |\phi| < 1 \qquad (24)$$

and its equivalent random shock form

$$W_t = \sum_{j=0}^{\infty} \phi^j A_{t-j}, \qquad |\phi| < 1. \qquad (25)$$

Setting $\psi_j = \phi^j$ in (21) through (23) gives

$$\sigma = \sigma_A \left(\frac{1}{1 - \phi^2}\right)^{1/2}$$

$$\begin{aligned}\sqrt{\beta_1} &= \sqrt{\beta_1(A)}\left[\frac{(1-\phi^2)^{3/2}}{1-\phi^3}\right]\\ \beta_2 &= [\beta_2(A)-3]\left(\frac{1-\phi^2}{1+\phi^2}\right)+3.\end{aligned}$$

The start value of the AR(1) model may then be obtained by fitting a Johnson curve and generating a Johnson variate as previously described.

4.2 A Joint Distribution

Let $P(W_{i+1}, W_{i+2}, \ldots, W_{i+p})$ represent the joint probability of p consecutive elements of the time series W_t. Let w_t denote a particular realization of the element W_t. The definition of conditional probability implies

$$\begin{aligned}&P(W_{i+1}, W_{i+2}, \ldots, W_{i+p}) = \\ &\qquad P(W_{i+1})P(W_{i+2} \mid w_{i+1})P(W_{i+3} \mid w_{i+1}, w_{i+2}) \cdots \\ &\qquad \cdots P(W_{i+p} \mid w_{i+1}, w_{i+2}, \ldots, w_{i+p-1}).\end{aligned}$$

Each term on the right hand side corresponds to a univariate distribution which may be approximated by a member of the Johnson system. The observed values w_{i+j} for $j = 1, \ldots, p$ may therefore be obtained from the joint distribution of W_t by constructing Johnson curve approximations to successive distributions, each conditional on the previously generated observed values w_t.

4.3 The General Algorithm

We now develop a procedure to generate the series start values for the general ARMA(p, q) model. Define

$$\theta_{0k} = \begin{cases}\theta_0 + \sum_{j=k+1}^{p} \phi_j w_{1-j}, & k = 1, 2, \ldots, p-1 \\ \theta_0, & k = p\end{cases} \tag{26}$$

and

$$\mu_k = \frac{\theta_{0k}}{1 - \sum_{j=1}^{k} \phi_j}. \tag{27}$$

Let $1 \le k \le p$ and consider the general ARMA(k, q) model

$$\phi_k(B)W_t = \theta_{0k} + \theta_q(B)A_t \tag{28}$$

and its equivalent random shock form

$$W_t = \mu_k + \psi_k(B)A_t \tag{29}$$

where $\phi_k(B)$, $\theta_q(B)$, and $\psi_k(B)$ are defined as in (2), (3), and (5) respectively. The algorithm to generate the initial values of the time series for the general ARMA(p, q) model via Approach B consists of the following steps:

ALGORITHM B

1. Determine $\sqrt{\beta_1(A)}$ and $\beta_2(A)$ for a specified innovations distribution with mean 0 and variance σ_A^2.

2. Set $k = p$.

3. Compute μ_k using (27).

4. Compute σ, $\sqrt{\beta_1}$, and β_2 using (21) through (23) and the result of Step 1.

5. Determine the type of Johnson curve and the parameters ξ, λ, δ, and γ using algorithm AS 99 given the results of Step 4.

6. Generate a pseudo-random number from the standard normal distribution.

7. Apply the inverse transformation of algorithm AS 100 to compute a pseudo-random Johnson variate using the Johnson curve determined in Step 5 and the result of Step 6.

8. Compute w_{1-k} by adding μ_k from Step 3 to the pseudo-random Johnson variate from Step 7.

9. Set $k = k - 1$.

10. Repeat Step 3 through Step 9 until $k = 0$.

The values w_{1-p}, w_{2-p}, …, w_0 constitute an observation from the joint distribution of p consecutive elements of the time series W_t.

5. DISCUSSION

We have considered two approaches to warming up time series simulations. Approach A, as implemented through ALGORITHM A, approximates the general ARMA(p, q) model by a finite order moving average in order to determine the p start values of the series. The transient bias introduced by beginning the simulation run with approximate start values requires an induction period of m observations. To avoid unnecessary warming up of the time series, an optimal value of m may be determined using the method of Anderson (1979). This value is dependent upon the length of the time series to be simulated as well as the autoregressive parameters of the model. Although this method extends to the general ARMA(p, q) model, the complexity of the expression for the minimal induction period may prohibit its use in practice.

Since the source of the difficulty with Approach A lies in the approximation used to obtain the start values of the series, we proposed a method to directly generate these observations from the joint distribution of the time series, called Approach B. This approach may be implemented using the strategy of ALGORITHM B. For a specified innovations distribution, the mean, standard deviation, skewness, and kurtosis of the time series are computed for general ARMA(k, q) models of decreasing order in k. At each value of k, the distribution of the time series is approximated by a Johnson curve and a start value is generated. Using the definition of conditional probability, the start values are generated in succession, and together constitute an observation from the joint distribution of the time series. In practice, ALGORITHM B may be applied with a moderate amount of warming up to compensate for fitting the distribution based on the first four moments.

The heuristic descriptions of Approach A and Approach B involve approximations of the time series model and of the joint distribution of the time series, respectively. Comparison of these approaches may be performed with respect to their implementation in ALGORITHM A and ALGORITHM B. In particular, ALGORITHM B

- requires no explicit warming up period;

- does not depend on the length of the simulated series;

- readily extends to the general ARMA(p, q) model.

Further research into the properties of both approaches and algorithms is in progress.

REFERENCES

Anderson, O. D. (1975), "On the Collection of Time Series Data," *Operations Research Quarterly*, **26**, 331–335.

Anderson, O. D. (1979), "On Warming-Up Time Series Simulations Generated by Box-Jenkins Models," *Journal of the Operations Research Society*, **30**, 587–589.

Bacon-Shone, J. (1985), "Algorithm AS 210: Fitting Five Parameter Johnson S_B Curves by Moments," *Applied Statistics*, **34**, 95–101.

Cheng, R. C. H. (1976), "A Note on the Effect of Initial Conditions on a Simulation Run," *Operations Research Quarterly*, **27**, 467–470.

Davies, N., Spedding, T., and Watson, W. (1980), "Autoregressive Moving Average Processes With Non-normal Residuals," *Journal of Time Series Analysis*, **1**, 103–109.

Elderton, W. P. and Johnson, N. L. (1969), *Systems of Frequency Curves*, Cambridge: Cambridge University Press.

Hill, I. D. (1985), "Algorithm AS 100: Normal-Johnson and Johnson-Normal Transformations," in *Applied Statistics Algorithms*, eds. P. Griffiths and I. D. Hill, Chichester: Ellis Horwood, pp. 183–187.

Hill, I. D., Hill, R., and Holder, R. L. (1985), "Algorithm AS 99: Fitting Johnson Curves by Moments," in *Applied Statistics Algorithms*, eds. P. Griffiths and I. D. Hill, Chichester: Ellis Horwood, pp. 171–182.

Johnson, N. L. (1949), "Systems of Frequency Curves Generated by Methods of Translation," *Biometrika*, **36**, 149–176.

Johnson, N. L. (1965), "Tables to Facilitate Fitting S_U Frequency Curves," *Biometrika*, **52**, 547–550.

Johnson, N. L. and Kotz, S. (1970a), *Continuous Univariate Distributions–1*, New York: John Wiley.

Johnson, N. L. and Kotz, S. (1970b), *Continuous Univariate Distributions–2*, New York: John Wiley.

Kendall, M. G. and Stuart, A. (1969), *The Advanced Theory of Statistics* (Vol. 1; 3rd ed.), New York: Hafner.

Leslie, D. C. M. (1959), "Determination of Parameters in the Johnson System of Probability Distributions," *Biometrika*, **46**, 229–231.

Ord, J. K. (1972), *Families of Frequency Distributions*, New York: Hafner.

Patel, J. K., Kapadia, C. H., and Owen, D. B. (1976), *Handbook of Statistical Distributions*, New York: Marcel Dekker.

Piccolo, D. and Wilson, G. T. (1984), "A Unified Approach to ARMA Model Identification and Preliminary Estimation," *Journal of Time Series Analysis*, **5**, 183–204.

Censored Discrete Data and Generalized Linear Models

Douglas B. Clarkson, IMSL Inc.

Abstract

Generalized linear models in discrete data encompasses, among other models, logistic regression models, probit models, and Poisson regression models. This paper discusses an algorithm for computing parameter estimates in such models when interval (and other) censoring is present in the data. Also discussed are some solutions to problems encountered in the algorithm, along with the statistical implications of some forms of model degeneracy. Finally, censored data analogues of some common non-censored data graphical techniques and statistics are given.

1.0 Introduction

This paper discusses some experiences gained when implementing subroutine CTGLM (CaTegorical Generalized Linear Models) for inclusion in the IMSL libraries. Although concern will be with linear models in the sense of Nelder and Wedderburn (1972) or McCullagh and Nelder (1983), their terminology will not be used. The main advantage that CTGLM seems to offer over similar subroutines is the ability to handle censored (right, left, or interval) data directly. This ability causes some problems in the usual algorithms and in the usual analysis. Discussion will center around how these problems can be resolved.

2.0 The Models

CTGLM handles four discrete distributions and a total of six models. User specified models (without censoring) are handled by another routine. Let x_i denote a row vector of covariates, β denote a column vector of parameters, n_i, denote the binomial sample size, r_i denote the number of successes in the negative binomial, λ_i denote the Poisson parameter, θ_i denote the probability of success in a single Bernouli trial, y_i denote the realization of the random variable, and let Φ denote the cumulative standard normal probability distribution function. Then the possible models are given as:

1. Binomial, $f(y_i/n_i, \theta_i, x_i)$, with three models for θ_i:
   (a) logistic: $\theta_i = \frac{\exp(x_i\beta)}{1+\exp(x_i\beta)}$.
   (b) Probit: $\theta_i = \Phi(\exp(x_i\beta))$.
   (c) Log-log: $\theta_i = \exp(-\exp(x_i\beta))$.
2. Poisson: $f(y_i/\lambda_i, x_i)$, $\lambda_i = \exp(x_i\beta)$.
3. Negative Binomial: $f(y_i/\theta_i, r_i, x_i)$, $\theta_i = \frac{\exp(x_i\beta)}{1+\exp(x_i\beta)}$.
4. Logarithmic: $f(y_i/\theta_i, x_i)$, where $\theta_i = \frac{\exp(x_i\beta)}{1+\exp(x_i\beta)}$.

In all models, left censoring is the same as interval censoring with a left endpoint of zero. In the binomial models, right censoring is the also the same as interval censoring with a right endpoint of n_i. Note that the covariates x_i may be (and usually are) vector valued.

3.0 Example

The following table helps to illustrate the logistic model for interval censored data. An interval for the number of deaths at a given dose level and sample size is given, along with the maximum likelihood estimates of the binomial probabilities, $\hat{\theta}_i$, and the estimated probability of the observation (censoring interval) $\hat{f}_i$. Throughout this paper it is assumed that censoring mechanism operates independently of the binomial probability and of the outcome. Other censoring mechanisms may be possible. See, e.g., Kalbfleisch and Prentice (1980).

Table 1
An Example

Dose	Number Deaths	Sample Size	Censoring Type	$\hat{\theta}_i$	$\hat{f}_i$
1	0-3	100	Left	.0124	0.964
2	7-15	100	Interval	.1021	0.849
3	40-60	100	Interval	.5068	0.963
4	80-100	100	Right	.9028	0.999

When a binomial model is fit to the data with a simple linear logistic model on x_i = dose, one obtains the maximum likelihood estimate for the intercept as $\hat{\beta}_o = -6.576$ with slope estimate $\hat{\beta}_1 = 2.201$. The usual asymptotic statistics may also be computed. The estimates $\hat{f}_i$ of the estimated 'cell' probabilities are obtained from the maximum likelihood estimates.

The data for this example may arise, for example, in an experiment on an insecticide. Insects may be censored because they die for reasons unrelated to the insecticide, and before the effect of the insecticide can be assessed.

4.0 The Algorithm

Let $\eta_i = x_i\beta$ and note that the derivatives of f_i in the following are with respect to η_i, and that f_i and its derivatives are evaluated at $\hat{\eta}_i$. When an observation is censored, the function f_i is the sum over the censoring interval of the probability distribution. Otherwise, f_i is the probability of the single observed outcome. The log-likelihood and derivatives are computed as follows:

$$\begin{aligned} l &= log(L) = \sum_{i=1}^{N} log(f_i) \\ l'_\beta &= \sum_{i=1}^{N} \frac{f'_i x_i}{f_i} = \sum_{i=1}^{N} w_{2i} x_i \\ l''_\beta &= \sum_{i=1}^{N} \left[\frac{f''_i}{f_i} - \left(\frac{f'_i}{f_i} \right)^2 \right] x_i x_i^t = \sum_{i=1}^{N} w_{1i} x_i x_i^t \end{aligned}$$

where N is the number of observations (rows in X).

Newton-Raphson iteration can be implemented via weighted least squares with

1. weights w_{1i},
2. independent variable x_i, and
3. dependent variable $\frac{w_{2i}}{w_{1i}}$.

The Newton-Raphson step is obtained as the vector of estimated regression parameters. Alternatively, scoring can be implemented via weighted least squares with

1. weights w_{2i}^2,
2. independent variable x_i, and
3. dependent variable $\frac{1}{w_{2i}}$.

CTGLM always begins with scoring. When the relative change in the likelihood from one iteration to the next is small enough, the algorithm switches to Newton-Raphson iteration. Step halving is used for the line search whenever the likelihood does not increase with the initial step. The weighted least squares estimates are computed via Givens rotations. McCullagh and Nelder (1983), and Stirling (1984), among others, discuss the same or similar algorithms. See also Green (1984).

5.0 A Convergence Problem

A problem with the algorithm which is more common in censored data is that one or more of the estimated parameters $\hat{\eta}_i = x_i\hat{\beta}$ may be infinite. As an example, consider the logistic model for a single observation. If the observation is such that $y_i = n_i$ (i.e., if the number of successes equals the number of trials), or if the observation is censored and the censoring interval contains n_i, then the maximum likelihood estimate of the binomial probability is $\hat{\theta}_i = 1$. In the logistic model this corresponds to $\hat{\eta}_i = x_i\hat{\beta} = \infty$, obviously an extreme situation. Alternatively, if the observation (or the censoring interval) contains 0, then $\hat{\eta}_i = -\infty$ is obtained. These types of extremes are more common in censored data, but they can occur in uncensored data.

With more than one observation, this same type of extreme will occur if the $i-$th observation is n_i (or 0) and, additionally, the i-th row of the design matrix X (x_i) is linearly independent of the remaining rows in X. More generally, if all rows in X corresponding to a group of right (or left) censored observations are linearly independent of the remaining rows in X, then the $\hat{\eta}_i$ corresponding to these rows will be ∞ ($-\infty$).

To see how easily such extremes can occur in practice, consider the example in section 3.0 with a one one-way ANOVA model replacing the simple linear regression model for the parameter η_i. In this model, the four observations are linearly independent of each other, so the estimated parameter at dose 1 will be $\hat{\theta}_1 = 0$ which corresponds to $\hat{\eta}_1 = -\infty$, while dose 4 will have parameter estimate $\hat{\theta}_4 = 1$ which corresponds to $\hat{\eta}_4 = \infty$. Estimates for the regression parameters β will depend upon the particular parameterization used in the ANOVA model, but regardless of the parameterization used, most, if not all, of the parameters estimates $\hat{\beta}_i$ will be infinite and the iterative algorithm will fail to converge. Note, however, that maximum likelihood estimates for the observation probabilities, θ_i, are well defined.

To account for the possibility of infinite $\hat{\beta}$, CTGLM uses restricted maximum likelihood estimation as follows:

> Each observation (censored or otherwise) is restricted such that its estimated probability ($\hat{f}_i$) is less than 0.9999. Whenever $\hat{f}_i$ becomes 0.9999 or greater then the observation is omitted from the likelihood (until its probability becomes less than 0.9999).

Note that restricting the f_i also restricts the parameters β_i through η_i. In effect the norm of the β_i's is not allowed to become too large. While one could restrict the β_i's directly, it is more natural to restrict f_i. Moreover, the statistical properties of the resulting estimaters are clearer.

It is important to note that it is the probability of the observation, and not the binomial parameter θ_i, which is being restricted. Indeed, in a binomial model a right censored observation may have a current estimate for θ of 0.7 or less, but because of the censoring, the probability of the observation (i.e., the sum of the binomial probabilities) can be very close to one.

In the following the log-likelihood in which observations with probabilities near one have been eliminated is called the 'reduced likelihood'. The 'restricted likelihood' is the log-likelihood one obtains when the restrictions on the f_i are applied. CTGLM optimizes the reduced likelihood, not the restricted likelihood. However, it is easy to show that a local optimum of the reduced likelihood is also a local optimum for the restricted problem. To see this, let $l_i(\beta) = log(f_i)$, and denote the constraints as $l_i(\beta) \leq log(1-\epsilon) = -\delta$, for $\epsilon, \delta > 0$. Define 'Lagrange multipliers' $\mu_i = 1$ if the i-th observation is restricted, with $\mu_i = 0$ otherwise. The restricted log-likelihood, l_R, involves the μ_i and is given as

$$l_R = \sum_{i=1}^{N} l_i(\hat{\beta}) - \sum_{i=1}^{N} \mu_i(l_i(\hat{\beta}) + \delta).$$

In the restricted log-likelihood, both β and μ must be estimated. With the choice for the μ_i above, the restricted log-likelihood, l_R, yields the same estimates as the reduced likelihood since the restricted observations are eliminated from ∇L (see below) in both. All that remains is to show that the μ_i chosen above yield a local optimum for the restricted likelihood. Using the Kuhn-Tucker second order sufficiency conditions (Luenberger, 1984, pages 316-317), this amounts to showing that the chosen μ_i, $i = 1, \ldots, N$, are such that

$$\mu_i(l_i + \delta) = 0,$$

that

$$\nabla L = \sum_{i=1}^{N} \nabla l_i(\hat{\beta}) - \sum_{i=1}^{N} \mu_i \nabla l_i(\hat{\beta}) = 0,$$

and that the Hessian matrix

$$H = \sum_{i=1}^{N} \nabla^2 l_i(\hat{\beta}) - \sum_{i=1}^{N} \mu_i \nabla^2 l_i(\hat{\beta})$$

is of full rank on the space orthogonal to $\sum_{i=1}^{N} \mu_i \nabla^2 l_i(\beta)$. Because of the form of the Hessian matrix, this latter assumption is equivalent to restricting β to a space orthogonal to the rows x_i in X for which $\mu_i = 1$.

Because of the choice for the μ_i's, and because a local optimum for the reduced likelihood is assumed, the last two conditions above are clearly satisfied. It remains only to show that $\mu_i(l_i + \delta) = 0$ for all i. This is trivial to show if $\mu_i = 0$. If $\mu_i = 1$ then $l_i(\hat{\beta})$ is restricted and again $l_i + \delta = 0$. Thus, for probabilities near one, the restricted and the reduced likelihoods are identical. (Note, however, that in both the $\hat{\beta}_i$ are not uniquely defined because the Hessian is singular.)

As an example of estimates obtained from the algorithm, parameterize the ANOVA model for the example above as follows: Let $\eta_i = \beta_0 + \gamma_j$ for j = doses 1, 2, and 3, while $\eta_4 = \beta_0$. Then the estimated parameters are $\hat{\beta}_0 = 2.8$, and $(\hat{\gamma}_1, \hat{\gamma}_2, \hat{\gamma}_3) = (-10.1, -4.9, -2.8)$. These estimates give estimated observation binomial parameters of (0.0007, 0.1072, 0.5000, 0.9422).

6.0 Some Considerations about the Deviance

The deviance d_i (Nelder and Wedderburn, 1972, page 375) for each observation is given as

$$d_i = 2(max_\beta l_i(x_i\beta) - l_i(x_i\hat{\beta})),$$

i.e., it is twice the maximizing log-likelihood of the observation with respect to the parameter η_i $(= x_i\beta_i)$ minus the likelihood of the observation as obtained from $\hat{\beta}$. In the absence of restrictions, the deviance has $N - p$ degrees of freedom, where p is the number of parameters in β. The total deviance is given as the sum of the d_i, i.e., as:

$$D = \sum_{i=1}^{N} d_i = \sum_{i=1}^{N} 2(sup_\beta l_i(x_i\beta) - l_i(x_i\hat{\beta}_i)).$$

The deviance of each observation may be used as a 'residual' (see below). The total deviance may be used in an asymptotic chi-squared goodness of fit test of the model (see McCullagh and Nelder (1983)), with $N - p$ degrees of freedom.

The definition of the deviance need not change in censored data. The d_i will become, however, more difficult to compute because a closed form solution for the optimizing $l_i(x_i\beta)$ is usually not available in censored data.

The degrees of freedom for the deviance when there are restricted observations must also be adjusted. Clearly one degree of freedom should be subtracted from the total degrees of freedom, N, for each restriction applied. However, because the restrictions on l_i also restrict the parameters β, the degrees of freedom, p, for the number of estimated parameters may also decrease. Let r denote the number of linearly independent rows in the matrix formed from the restricted x_i. Then r is the number of restrictions placed on the parameters β, and the total degrees of freedom in the deviance is $N - p - q + r$, where q is the number of restrictions placed upon the observations. (Note that in the binomial distribution each Bernouli trial is an observation and contributes to N.)

The degrees of freedom in the deviance is data dependent and thus a random variable. More restrictions will be applied in some samples than in others. These changing degrees of freedom will affect the chi-squared goodness of fit test. Whether the restricted degrees of freedom are a better predictor of the adequacy of the adequacy of the asymptotic approximations used throughout the analysis is another question needing study.

7.0 Residual Plots

Pregibon (1981) gives two methods for defining residuals which are of interest here. The first, involving the deviances, is given as:

$$r_{1i} = \text{sign}(\hat{\eta}_i - x_i\hat{\beta})\sqrt{d_i},$$

where d_i is the component of the deviance as discussed above, while $\hat{\eta}_i$ is the optimizing η_i for the single observation in question. Clearly, $r_{1i} = 0$ for restricted observations. In CTGLM, it would be desirable if one could avoid computing the d_i since their computation may be expensive, especially in censored data where the computation of $\hat{\eta}_i$ may require iteration for each observation.

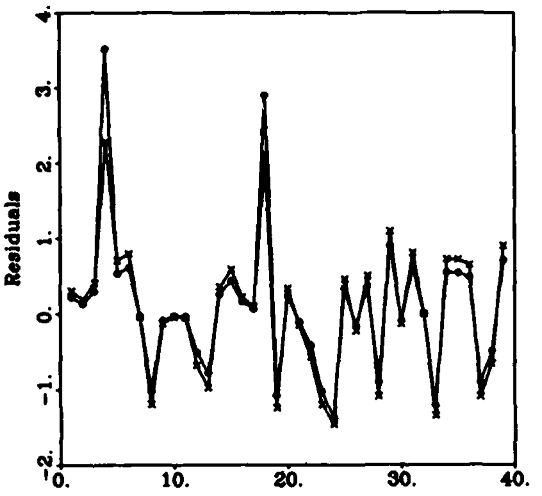
The second method of defining residuals is given by Pregibon (1981, pages 708-709) as:

$$r_{2i} = \frac{l_i'(\hat{\beta})}{\sqrt{l_i''(\hat{\beta})}}.$$

Clearly r_{2i} is easy to compute with censored as well as 'exact' data. Indeed, r_{2i}, is obtainable from quantities already computed in the iterative algorithm. Moreover, it is possible to show that r_{2i} is a 'one-step' approximation to r_{1i}. Thus, one would expect r_{1i} and r_{2i} to be very close for 'small' residuals, and not so close for 'large' residuals. This is born out in the figure below, in which both types of residuals are plotted versus the index number for the data. The data is taken from Pregibon (1981), who attributes it to Finney (1947), and gives similar plots. In this figure, at least, the two types of residuals seem almost equivalent.

Because r_{2i} is cheap to compute, while r_{1i} is not, r_{2i} is the residual used in CTGLM. Similar figures were obtained for other data sets.

Figure 1
The Residuals (x-r_1, o-r_2)



8.0 A Monte Carlo Study

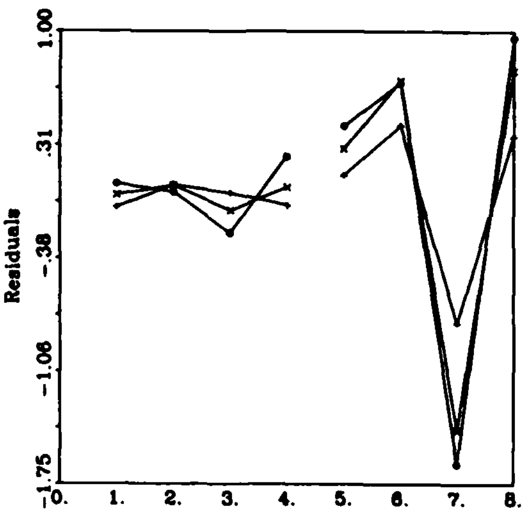
The effect of censoring on the estimated parameters and the resulting residuals is also of interest, as is whether one can compare 'censored' residuals with 'uncensored' residuals. In an attempt to answer these questions, a small Monte Carlo study was performed using a simple linear logistic regression model. Two factors were studied. The first factor was the censoring level. Three levels of censoring (0%, 10%, and 30%) were used. The second factor was concerned with the appropriateness of the simple linear model. Data generated for level one of this factor fit the model for covariate values of (-2, -0.5, 0.5, and 2.0) corresponding to binomial probabilities of (0.119, 0.378, 0.622, and 0.881). In the second level the same covariate values were used, but the probability 0.662 for the third covariate was changed to 0.20. Thus, in the second level, the simple linear logistic regression model did not fit. A balanced factorial design was used.

All binomial observations involved 10 Bernouli trials. Censoring was incorporated as follows: For each Bernouli trial a uniform (0,1) random deviate was generated using function GGUBFS of the IMSL (1984) library. If the generated deviate was greater than the censoring probability then a second deviate was generated. If the second deviate was less than the logistic probability at the given dose, then the observations count was incremented by one. If the first deviate was less than the censoring probability, then the length of the censoring interval was increased by one. All computations were performed on a Data General MV 10000 computer and coded in FORTRAN.

Thirty replications at each combination of the two factors were performed. For each sample of four observations in each replication, the simple linear logistic model was fit involving parameters β_0 and β_1 according to the methods discussed above. The results are discussed below.

The expected (based upon the 30 replications) regression slope, β_1 decreased ($p \leq 0.0001$) as the censoring increased. As one would expect, the slope also changed if the model was not correctly specified (p=.0963), although the effect was not as dramatic. The intercept (β_0) was significantly different ($p \leq 0.0001$) when the model was not correctly specified. It was not affected (very much) by the censoring level. As the censoring level increased, the size of the average residual decreased, but the decrease was not always significant. To get a feel for the variation in the residuals, consider the figure below. In this figure the average residuals for the ANOVA cells which fit the model are plotted at points (1, 2, 3, and 4) on the x-axis, with all three censoring levels present, while the average residuals for the ANOVA cells which did not fit the model are plotted at x-axis points (5, 6, 7, and 8). The censoring levels are (0-0%, X-10%, +-30%). Note in the figure that the ordering of the residuals tends to be +-smallest, X-second smallest, and 0-largest. The residuals at x axis point 7 do not fit the model (the actual binomial probability was changed from 0.622 to 0.20). This fact is clear from the size of the residuals at this point. Also note that the residuals tend to be smallest when the data fits the model (i.e., at x-axis points 1, 2, 3, and 4).

Figure 2
The Expected Residuals (o-0%, x-10%, +-30%)



The residual plots seem to indicate that one can compare 'censored' residuals with residuals from observations which are not censored. One should expect, however, that residuals from censored observations will tend to be smaller in magnitude.

References

Finney, D. J., (1947), The estimation from individual records of the relationship between dose and quantal response, *Biometrika*, **34**, 320-334.

Green, P. J, (1984), Iteratively Reweighted Least Squares for Maximum Likelihood Estimation, and some Robust and Resistant Alternatives, *Journal of the Royal Statistics Society, Series B*, **46**, 149-192.

Kalbfleisch, John D. and Prentice, Ross, L., *The Statistical Analysis of Failure Time Data*, John Wiley and Sons, New York.

Nelder, J. A and Wedderburn, R. W. M., (1972), Generalized Linear Models, *Journal of the Royal Statistics Society, Series A*, **135**, 370-384.

McCullagh, P. and Nelder, J. A., (1983), *Generalised Linear Models*, Chapman and Hall, London.

Pregibon, Daryl, (1981), Logistic Regression Diagnostics, *The Annals of Statistics*, **9**, 705-724.

Stirling, W. Douglas, (1984), Iteratively reweighted least squares for models with a linear part, *Applied Statistics*, **33**, 7-17.

HOTELLING'S T^2, ROBUST ꟳRINCIPAL COMPONENTS, AND GRAPHICS FOR SPC

David Coleman, RCA Laboratories

Abstract

Correlated variables in the same physical unit are commonly measured for purposes of statistical process control (SPC) - for example, dimensions of a manufactured part. The multivariate control chart using Hotelling's T^2 statistic is an effective process control technique for this multivariate situation. A common problem, however, has been how to make practical use of such control charts. It is often hard to interpret "out-of-control" declarations so as to produce control actions which correct the problem. Robust principal components analysis captures the covariance structure of subgroups of multivariate observations. Corresponding graphical techniques help us interpret bias and variability problems much more effectively than can a multivariate control chart, alone. These methods can be used to develop a diagnostic SPC system.

1. Main Results and Conclusions, In Brief

A common SPC and engineering diagnostic problem is to interpret and use multiple measurements of a product taken in the same physical unit, but at different locations on the product (Figure 1). These measurements are often highly correlated. A procedure for SPC is motivated and described in this paper, using as an example the relative misalignment of photolithographic grids on (integrated circuit) semiconductor wafers. The procedure can be summarized as follows:

(1) Use robust principal components to: (a) Select a "process base sample," of a large number of typical multivariate observations, (b) Estimate a "process covariance matrix," S^*, (c) Compute process principal axes from S^*.

(2) Periodically sample subgroups of product for routine SPC.

(3) For each subgroup: (a) [Optional] Compute Hotelling's T^2 statistics for subgroup bias and variability in the principal axis space of the "process" database (Equation (3)), (b) Interpret a potential variance problem using a robust "Principal Axis Plot" (Figures 7a and 7b) for the subgroup, (c) Interpret a potential bias problem by using both a "Spider Plot" (Figure 6) - which shows an exaggerated representation of measurements taken on subgroups of product, and by using plots of the subgroup principal components (Figure 10).

2. Introduction

2.1 Statement of the Problem

A common problem in manufacturing is that of maintaining a process under statistical control when it has several correlated process or performance control variables, in the same physical units. The prototypical case for this paper will be the physical dimensions of a manufactured part. For example, suppose two oxide grids are supposed to be applied to a semiconductor wafer (perhaps a "monitor wafer") such that one falls exactly upon the other, but manufacturing variability causes misalignment. Figure 1 shows a simple but extreme case of misalignment. Measurements for the purposes of SPC and diagnosis of manufacturing engineering problems might be taken on sample wafers at a small subset of the grid nodes, such as the 5-by-5 array shown in Figure 1. These measurements might be of misalignment or of som other variable, such as thickness or sheet resistivity. Commercial systems are currently available to take and display various measurements at many locations on a wafer - in the form of a contour plot. As far as the author can determine, however, none of them has the statistical capabilities described in this paper.

In one of the simplest statistical models of misalignment - the physical displacement of grid B relative to grid A - a simple analytic expression would allow us to express the transformations associated with Euclidean geometry: rotation about some point (perhaps not the grid center, but constrained to be within a limited domain), then horizontal and vertical translation:

$$x^{obs}_{i,j,k} = x_i^{rot} + r_{i,j,k} \cos(\theta_i + \phi_{i,j,k}) + x_i^{trans} + \epsilon^x_{i,j,k}$$

$$(1)$$

$$y^{obs}_{i,j,k} = y_i^{rot} + r_{i,j,k} \sin(\theta_i + \phi_{i,j,k}) + y_i^{trans} + \epsilon^y_{i,j,k}$$

$i=1,...,n$ refers to the wafer number within a sample
$j=1,...,5$ (the horizontal position on the measurement array)
$k=1,...,5$ (the vertical position on the measurement array)

$r_{i,j,k} = \sqrt{(x^{actual}_{i,j,k} - x_i^{rot})^2 + (y^{actual}_{i,j,k} - y_i^{rot})^2}$ = the Euclidean distance of each of the 25 locations of the measurement array from the point of rotation, (x_i^{rot}, y_i^{rot}).

θ_i = the angle of rotation about the point of rotation.

$\phi_{i,j,k} = e.g. ,\sin^{-1}(\frac{x^{actual}_{i,j,k} - x_i^{rot}}{r_{i,j,k}})$ = the angle position of the 25 locations of the measurement array with respect to the point of rotation $(r_{i,j,k} \neq 0)$.

x_i^{trans} , y_i^{trans} is the amount of translation in x and y.

$\epsilon^x_{i,j,k}$; and $\epsilon^y_{i,j,k}$ are measurement errors in the x and y directions.

[Note that the contraint on the location of the point of rotation forces us to also allow a translation; any (Euclidean) isometry can be described by a rotation about a point, if the location of such a point is unconstrained].

We might assume a model such as this and use a procedure such as nonlinear least squares to estimate the unknown parameters of the transformations.

Unfortunately, model (1) is too simple for many applications, because rotation and physical translation may be inadequate to describe the possible patterns of measurements, such as grid misalignment. We may want to allow additional transformations: shearing, projections, inversions in circles, reflections in lines, and other one-to-one, differentiable mappings. See Figure 2 (as suggested in [1]). In our specific wafer example, if the grids are applied by projection photo-lithography, there may be additional grid distortions due to process or wafer irregularities, such as vertical or horizontal stretch (or compression = "negative stretch"), diagonal stretch, radial stretch, saddle-shaped wafer, or local distortion (e.g., a local blemish). Transformations as diverse as this can occur in measurements of all kinds on processes of all kinds; they are not peculiar to wafer fabrication.

The direct mathematical modeling approach is always appealing. We could, in principle, extend model (1) to include various additional types of transformations. However, we may not be able to state, a priori, in what geometry we should be working - more specifically, what all of the plausible patterns of distortion might be. Also, the patterns may be too complex or diverse for us to be confident that we can wisely spend degrees of freedom for estimation of linear and non-linear distortion parameters. In addition, we are likely to be interested in patterns of systematic VARIATION in the measurements, as well as patterns of BIAS (measurements made on individual samples of product). These variational patterns may be even more difficult to specify, a priori. Hence, the potential complexity of the direct mathematical modeling approach leads us to consider an indirect approach. This paper describes such an indirect approach: a diagnostic SPC system for wafer misalignment or some other similar performance variable.

2.2 Hotelling's T^2 Statistic(s)

A standard textbook strategy for SPC when there are correlated process variables (see, for example, [2]) is to use Hotelling's T^2 statistic,

$$T_j^2 = (x_j - \bar{x})'S^{-1}(x_j - \bar{x}) = y_j{}'y_j. \qquad (2)$$

where x_j is the j^{th} p-dimensional observation which we want to assess for control, $\bar{x}$ is the p-dimensional vector of sample means of the p (correlated) process variables taken on n parts (wafers, in our example), S is a sample covariance matrix, and $y_j = V\,'D^{-1}(x_j - \bar{x})$, the j^{th} observation in principal axis space (V is from the eigen-decomposition, $S = VD^2V\,'$). It is known that $F = \frac{n-p}{p(n-1)}T^2$ has the F distribution with degrees of freedom p and $(n-p)$. This implicitly defines the Hotelling's T^2 "α-confidence ellipse."

$$F_{p,n-p,\alpha/2} \leqslant \frac{(n-p)}{p(n-1)} (x-\bar{x})'S^{-1}(x-\bar{x}) \leqslant F_{p,n-p,1-\alpha/2},$$

which is quadratic in the vector x (see Figure 3).

2.2.1 Use of Hotelling's T^2

Hotelling's T^2 statistic correctly handles the correlation structure of the p process variables in that it gives true α-level (Type I) confidence in declaring "out-of-control" for simultaneous values of the p variables. It would seem to be the method of choice for the multivariate SPC situation. As J.S. Hunter has repeatedly commented (e.g., [3]),

> "If bivariate charts are so valuable, why, one might ask, haven't such charts found wider use? Arithmetic is the answer. Hotelling's T^2 statistic must be calculated to establish the bivariate [and more generally, *multivariate*] control boundaries. This expression and its associated arithmetic may appear formidable, but they are not. Today's hand-held calculator or desk-top computer is easily programmed to complete the necessary arithmetic and graphics within a few seconds.... In practice, the factory worker would place the several measured responses into the hand-held or desk-top calculator. The calculator would calculate T^2, and could be programmed to 'beep' whenever an unusual value of T^2 was obtained... Monitoring today's processes with one-variable-at-a-time methods is to throw away information."

It is hard to disagree with Hunter. The many companies now supplying SPC software have perhaps not yet attained this level of sophistication, but it is only a matter of time.

The real impediment to multivariate SPC, other than computational complexity, has been the difficulty of interpretation. However, for the SPC situations described in this paper, the Hotelling's T^2 approach can be usefully augmented with robust methods and with statistical graphics so as to ease interpretation, especially for manufacturing engineers. Some motivation is given in the next section.

2.2.2 Limitations, Especially When p is Large

While technically correct, and usable in practice, Hotelling's T^2 suffers from problems of interpretation, particularly when p is large.

Consider our wafer example, when the causes of misalignment might be complex. Succbut it might be hard to interpret the out-of-control declarations, and to decide what to do to get the process back in control. Indeed, it has been the author's experience that the same reaction is obtained again and again, after describing how the low Type I error (false reject) rate of control chart mein the declaration that SOMETHING is wrong with the process: "All right, WHAT is wrong with the process; what do I fix?"

This is not an easy question to answer. Indeed, it may seem to be better answered by an engineer who is more familiar with the process. However, more information CAN be gleaned from a control chart than just a control declaration. For the simplest case, the univariate control chart, a collection of rules of thumb can be usefully developed. These can be based upon the standard (or any specialized) conditions for declaring "out-of-control." For example, such rules for the $\bar{X}$ chart can take the form: (a) one point more than 3σ from the mean indicates a sudden, extreme departure from control, or (b) eight points in a row on the same side of the mean indicates a trend or a slight but persistent shift. Once users become familiar and comfortable with these rules (and the construction and philosophy of the control charts), more can be learned. The multiple rules serve to narrow the field of possible problems in that it is more likely that (a) is a bolt that snapped and (b) is tool wear, than vice versa. Of course, engineering interpretation is always needed to identify the specific problem at hand.

Similar rules of thumb can be developed for Hotelling's T^2 control charts - but the difficulties of interpretation are compounded. In the bivariate case, the simplest condition is one extreme point beyond the 99+% level. It can be due to extreme values of both variables, or just one of the two variables, or a pair of values of the two variables that is unusual - though neither value may be extreme in itself. In Figure 3, point A is likely to be easy to interpret, but points B and C, and especially D are likely to be difficult to interpret. The difficulties of interpretation are far greater when we consider more than two, say 50 variables, as in the wafer grid misalignment measurements. With a problem of this size and potential complexity, Hotelling's T^2 is not informative enough. We need effective interpretation, not just correct α-level declaration of "out-of-control."

3. Principal Components Analysis on Successive Rational Subgroups

Jackson advocates the use of principal components and a form of Hotelling's T^2 statistic for rational subgroups ([4], [5]). Instead of the simple T^2 statistic for a single part, given above, we compute (following Jackson) for n observations and $k \leqslant min(n ,p)$ principal components (ideally, $n \gg p$),

$$T_{O\ i}^2 = T_{M\ i}^2 + T_{D\ i}^2 \tag{3}$$

where $T_{O\ i}^2 = \sum_{j=1}^{n} T_{j(i)}^2 = \sum_{j=1}^{n} y_{j(i)}' y_{j(i)}$ is the overall variability of subgroup i about the grand mean, $\bar{\bar{x}}$ (note that we replace the mean in (2) with the grand mean). It has an asymptotic χ^2_{kn} distribution under the null hypothesis that the observations in this subgroup are not significantly different from the grand mean (usually we do hypothesis testing with $T^2_{M\ i}$ and$T^2_{D\ i}$, instead of with $T^2_{O\ i}$).

$T^2_{M\ i} = n\bar{y}_{(i)}'\,\bar{y}_{(i)}$ (where $\bar{y}_{(i)} = \sum_{j=1}^{n} \frac{y_{j(i)}}{n} = V\,'D^{-1}(\bar{x}_{(i)}-\bar{\bar{x}})$) is the squared bias of the subgroup mean from the grand mean. It has an asymptotic χ^2_k distribution under the null hypothesis that the bias is not significantly different from zero.

$T^2_{D\ i}$, computed by $T^2_{D\ i} = T^2_{O\ i} - T^2_{M\ i}$, is the variability of the subgroup observations about their own mean, $\bar{x}_i$. It has an asymptotic $\chi^2_{k(n-1)}$ distribution under the null hypothesis that the subgroup variability is not significantly different from the variability from which the principal components were derived.

Paired control charts of $T^2_{M\ i}$ and $T^2_{D\ i}$ comprise a *multivariate analog* of $\bar{X}$ and R (or s) charts, as briefly mentioned by Jackson. A perusal of textbooks and current literature indicates that these paired charts are rarely used. When computed robustly, and supplemented with good graphical displays, paired T^2 charts can be very effective.

3.1 Computations

A classic issue in principal components analysis is whether to do an eigendecomposition of the covariance matrix, S, or the correlation matrix, R. For the case of interest in this article - correlated variables in the same physical unit - the use of S is appropriate, since we have no a priori reason to scale measurements at one location differently than at another location.

Using S rather than R also allows us to make simple quantitative statements of interest about the principal component analysis. For example, suppose that we take that the variance in the original n by p matrix of observations, X, is $trace(S)$. Since the matrix of eigenvectors, $V\,'$, is orthonormal, the total variance of X remains the same after pre-multiplication ($V\,'X$ corresponds to a rigid rotation of the basis vectors). Hence, we can make statements of the form, "m% of the total variance in X is accounted for by the first k principal axes, which are: ..." (as stated in [6]).

3.2 Graphical Displays

As stated, the wafer example is prototypical for this paper. However, the graphical tools described below are applicable to any situation where dimensions or some other process-related or performance-related variable of a part are measured in the same physical unit, at different locations. Two graphical tools are presented for interpreting bias problems in misalignment. One of the tools is the Exaggerated-Measurement Plot - which is similar to a contour plot. For subgroups of wafers rather than individual wafers, we can use the related Spider Plot. Another graphical tool presented below is the Principal Axis Plot. It aids in interpreting variability.

3.2.1 Exaggerated-Measurement Plots and Spider Plots

The Exaggerated-Measurement Plot is a representation of the part on which measurements have been made - but with exaggerated representations of the measurements displayed (see Figures 4a and 4b).

Specifically, an Exaggerated-Measurement Plot is constructed by drawing a line segment for each location of measurement. One end of the line segment is at the nominal location of measurement. From that point, the line segment is drawn in the same direction as the misalignment at that location, but the length of the segment is scaled (up) so as to make it easier to identify patterns of misalignment, perhaps such as shown in Figure 1. Thus, the axes on the plot implicitly carry two scales: the scale of the part being measured - in which we can find the locations of measurement, and the misalignment error scale - which might be several orders of magnitude less.

The Exaggerated-Measurement Plot is analogous to a caricature, in that enough of the nominal features of the part are shown so that the part can be recognized for what it is, but the measured features unique to that particular part are coded into the graphical representation so as to highlight those features. This makes it easier to discriminate among parts. We may especially want to tell good parts from bad parts. Figures 4a and 4b show Exaggerated-Measurement Plots for misalignment of grids on two wafers. The wafer scale might be 10^{-4} meters, while the misalignment scale might be 10^{-6} meters.

By overlaying Exaggerated-Measurement Plots from sample members of a subgroup (see Figure 5) we can see the bias and variability of subgroups on a per-location basis. However, this method of graphical presentation can be improved.

R. Barton ([7]) and (implicitly) J. E. Jackson ([4]) have used a superior means of graphical presentation for subgroups of data of this form - the Spider Plot. The Spider Plot helps to separate bias from variability. Jackson illustrated it as a means for geometrically representing the relationships between the T^2 statistics of (3). For applications such as our wafer example, it can be used directly at each measurement location, as seen in Figure 6. It is constructed as follows: (a) Exaggerate all misalignment errors by the same amount, e.g., by a factor of 100. Do the following for each measurement location: (b) Compute the subgroup mean exaggerated error. (c) Draw a line segment from the nominal measurement location to the subgroup mean. (d) Draw one line segment per wafer from the subgroup mean- to the exaggerated error position - as would be obtained in an Exaggerated-Measurement Plot representation for that wafer at that location.

Another way to conceptualize how a Spider Plot is constructed is to suppose, for a moment, that the subgroup mean is zero at all measurement locations. Then, the Spider Plot would be identical to overlaid Exaggerated-Measurement Plots. Each spider could then be displaced from the nominal position to the actual subgroup mean, per location. This per-location subgroup bias is also represented by a line segment - with a solid dot at each end.

The Spider Plot helps us to make qualitative assessments of the form, "Is the within-subgroup variability sufficient to disregard the bias from the nominal?" and "Is there a pattern in the bias which is independent of the magnitude or shape of the variability?" and "Are all the spiders of the 'same species?' "

Either of the above plots is the natural companion to either the T_j^2 (for individual parts) or the $T_{M\ i}^2$ (for subgroups of parts) statistic. They each show departure from nominal which is in the form of bias. Exaggerated-Measurement Plots or Spider Plots could be examined routinely, or a control procedure could be designed so that when T_j^2 or $T_{M\ i}^2$ (whichever was being used) was declared out of control, an exaggerated-measurement or Spider Plot could be produced (by computer, of course) to help diagnose the problem. See Section 5.1 for such a procedure.

3.2.2 Principal Axis Plots ("Major Motion Pictures")

Principal Axis Plots address the systematic-variability side of the control question. A limitation of the exaggerated-measurement and the Spider Plots is that though they highlight part-to-part or subgroup-to-subgroup differences, it is hard to see PATTERNS of differences between parts or subgroups. This limitation holds also for the classical statistics we might compute on the X matrix: $\overline{X}$ and S; $\overline{X}$ is computed per measurement location, and S shows only pairwise relationships. Though we can compare bias and variability, location-to-location, in Figures 5 and 6, we cannot readily perceive whether or not the variation is systematic. In our assumed multivariate SPC situation, we know there is high correlation - so much of the variation IS systematic. A principal components analysis gives how much variation is systematic, and a Principal Axis Plot shows the pattern of systematic variation.

The Principal Axis Plot, such as shown in Figures 7a and 7b, is constructed as follows: (a) Do principal components analysis of the subgroup, resulting in (usually) a few major principal axes (that is, axes associated with relatively large principal values), (b) Scale each principal axis to be displayed (perhaps by a fixed value, or proportional to the associated principal value), (c) For each principal axis to be represented, place a limeasurement location with one end positioned at the nominal measurement location, (d) Draw the line segment as in an Exaggerated-Measurement Plot - treating the scaled principal axis as a vector of misalignment errors, with a horizontal and a vertical component for each measurement location, (e) Put "arrowheads" on the ends of the line segments, to help distinguish the Principal Axis Plot from an Exaggerated-Measurement Plot in appearance.

Figure 7a is a Principal Axis Plot for the first principal axis for a subgroup of wafers. It is apparent that much (nearly 88%) of the variation in misalignment is along a 30°—45° (diagonal) direction. The second principal axis is shown in Figure 7b, and it appears to be compression along one Cartesian axis, and expansion along the other, with an origin near the array point: (row 2, column 4). The orthogonality of principal axes lessens the likelihood that patterns of systematic variability will be confounded in either of the two ways: two sources of variability captured in one major principal axis, or one source of variability split into two major principal axes. As with Exaggerated-Measurement Plots and Spider Plots, we search for patterns such as illustrated in Figure 2 when we examine Principal Axis Plots.

The Principal Axis Plot is the natural companion to the $T_{\bar{D}\,i}^2$ statistic, which is just the sum of the T_j^2 statistics of (2) if there is only one subgroup (in which case the grand mean is identical to the mean). Principal Axis Plots could be examined routinely, or a control procedure could be designed so that when $T_{\bar{D}\,i}^2$ was declared out of control, a Principal Axis Plot could be produced of the offending subgroup, as given in Section 5.1.

3.3 Interpretation

In Section 2.2.2, the motivation for providing a means of interpretation of T^2 statistics was given. For SPC in the multivariate situation discussed in this paper, principal components analysis plus the three graphical techniques described in Section 3.2 help with engineering interpretation. This section is brief, because general discussion of interpretation is necessarily limited in scope. Section 5.2 has a more detailed example.

3.3.1 Classification

A good way to interpret correlated process variables in the same physical unit is to look for patterns among the values of the variables - particularly when those values have been declared out of control by a multivariate control procedure, such as Hotelling's T^2. This is especially effective for correlated process-related or performance-related variables measured at different locations of a manufactured part. For what types of patterns should we look? Probably the same types whether we are examining bias (using a Spider Plot or Exaggerated-Measurement Plot) or systematic variability (using a Principal Axis Plot) - though the interpretations would naturally differ. Typically, we should look for: (1) "global" patterns associated with the types of one-to-one, differentiable transformations listed in Section 2, such as: translation, rotation (not necessarily about the center), stretch in one dimension (not necessarily the original measurement dimensions), radial stretch (not necessarily centered), corner effects (such as "sag"), edge effects (such as a "rim"), and (2) "local" phenomena, such as isolated outliers or small clusters of outliers. When a process is declared out of control in subgroup variability, the type of thing that we do NOT want to see in an a Principal Axis Plot is seen in Figure 8. Such a pattern of systematic variability would be difficult to interpret - especially if it were associated with a large principal value (that is, it "accounted for" a lot of variability in the raw measurement data). If we were to get an Exaggerated-Measurement Plot with line segments such as shown in Figure 8, it would imply local problems rather than global problems.

3.3.2 Manufacturing Diagnostic Tables

The pragmatic advantage to classifying patterns is the potential for developing rules of thumb for engineering action. These might be analogous to (but more complex than) those developed for $\overline{\overline{X}}$ and R charts. Two basic methods might be used for tracing back performance patterns of bias or variability to their root causes: (1) physical phenomena affected by the geometry, and (2) pattern preservation.

Physical phenomena affected by the geometry include such processes as:

- film deposition - which might depend on distance from the source of deposition material.
- thermal, chemical, or electrical processing - which might be more extensive along the edges and/or corners because they are more exposed.
- imaging of light or electrons - which might have distortion approximately proportional to the angle of refraction or deflection.

Pattern preservation is a conservation law, which states that if a certain type of deviation from nominal (e.g., a lateral shift) is introduced during manufacturing, it will be preserved in the performance variables of the final product, unless specifically taken out. When process steps are more or less independent of one another, common sense tells us that patterns of deviations will be preserved. But even though this is often the case, pattern preservation is not often exploited in multivariate SPC. Examples are: lateral shift of a part relative to a fixture, rotation of part relative to fixture, optical process errors in setup leading to stretch or shrinkage, or other one-to-one, differentiable transformations of many kinds.

Departures from the assumption that the process steps are independent can take two basic forms. Further process steps can reduce patterns, or can magnify patterns. Reduction can occur when processes do relative tolerancing or relative alignment, rather than absolute. Or, a later process step may remove the deviation from nominal altogether (e.g. incorrect seating in a clamp might be deliberately corrected, or inadvertently corrected by handling). Magnification can occur when a small deviation from nominal can propagate, or have effects which propagate (e.g. a small burr in a milling process can cause extensive milling irregularities, or an off-trajectory laser beam may get further and further off-trajectory). After-the-fact SPC techniques are most likely to help identify the cause when independence or magnification holds.

4. Robust Principal Components Analysis

To guard against distortions due to atypical data values, Devlin, Gnanadesikan, and Kettenring ([8]) suggest several methods for computing robust principal components. Additionally, these methods can be used to identify such atypical values. They may be the most informative data.

4.1 Alternatives

Devlin et al. recommend the use of three robust techniques for principal component analysis. One is iterative multivariate trimming (MVT) based on trimming a fixed proportion, α, of observations with extreme (Mahalanobis) distances, $d_j^* = \sqrt{(x_j - m^*)' S^{*-1} (x_j - m^*)}$, where m^* is the current measure of location, and S^* is the current robust estimate of S (m and S are used as the starting-points for M^* and S^*). Other recommended approaches are based on "maximum likelihood t" (MLT), for only the Cauchy case (1 degree of freedom), and a Huber-weights-based method (HUB) designed to make S^* asymptotically unbiased for the multivariate normal. MLT and HUB were judged superior to MVT with regard to estimation of correlations, eigenvalues, and eigenvectors, but MVT converges faster, and is recommended by the authors for large values of p - which is typical of the SPC situation considered in this paper.

For this reason, and because MVT is invariant under all nonsingular linear transformations of the observations, the author has used and recommends the use of MVT. Analysis of rejected observations is also recommended, and is part of the procedure given in Section 5.1 .

5. A Working Procedure for SPC, and A Wafer Example

5.1 A Working Procedure

Below is summarized a procedure for SPC with strong engineering support (a more detailed and thorough form of this procedure has been developed, but is not given in this paper). It is assumed that the engineers have good analytic and interpretive skills, and that the product design is stable enough so that a body of process knowledge can be accumulated and refined.

[Note that a subtlety of the multivariate situation is that either or both bias and variability can be declared out of control when neither bias nor variability has statistically significantly increased - but the covariance structure has sufficiently changed.]

ROBUST, MULTIVARIATE SPC PROCEDURE

1) [GET S^*] Compute a robust sample covariance matrix, S^*, using MVT on a N by p matrix, X_{start} , of selected parts - resulting in trimmed matrix, X_{base}^* (N is assumed large), and associated major principal axis space, V_{base}^{major}.

2) [SAMPLE FOR ROUTINE SPC] Sample subgroup i of n parts, resulting in n by p matrix, X_i.

3) [GET T^2's] Compute Hotelling's $T_{D\ i}^2$, and $T_{M\ i}^2$, in V_{base}^{major}. Plot the Hotelling's T^2 values on separate, parallel, Hotelling's T^2 control charts, and apply the appropriate control chart rule(s).

4) [INTERPRET VARIABILITY PROBLEMS] If $T_{D\ i}^2$ is out of control, examine X_i in V_{base}^{major}, look for outliers which might inflate the subgroup variance, and seek to correct. If not present, compute robust principal components of X_i. Examine the major Principal Axis Plots for systematic patterns of variability, and compare them to those produced for X_{base}^*. Diagnose, record, and go back to 2.

5) [INTERPRET BIAS PROBLEMS] If $T_{M\ i}^2$ is out of control, produce the Spider Plot for X_i. Compare to the Spider Plot of X_{base}^*. Look for outliers and systematic bias, and seek to correct. Optionally, examine X_i in V_{base}^{major}, look for outliers, and seek to correct. Diagnose, record, and go back to 2.

5.2 A Wafer Example

Suppose that a projection photo-lithographic process applies two rectangular grids for oxide deposition on semiconductor wafers: (1) apply grid A, (2) change projection source and wafer fixture (reflecting process differences in applying the two different grids), (3) apply grid B. As described above, the nominal design calls for grid B to fall exactly upon grid A, but there is always some misalignment.

To carry out SPC on wafers, both vertical and horizontal misalignment of grids is measured at 25 positions on a 5 by 5 array, as shown in Figure 1. The measurements are highly correlated, because any one grid of vertical and horizontal lines is applied in a single process step. Differences from the nominal location within a process step are not trivial, however (correlations are not +/- 1), because they can be due to a variety of deviations from the ideal process setup - which do not necessarily affect misalignment uniformly over the gridded wafer.

Hotelling's $T_M^2{}_i$ and $T_D^2{}_i$ can be used on production wafer subgroups. Classical and robust sample covariance matrices can be estimated based on many wafers sampled uniformly over a considerable period of production time. After resolving any concerns about outliers in the sample used to compute S, subgroups of wafers can be then sampled for routine SPC. Figure 6 shows the Spider Plot for a sample from a subgroup which was "out of control," according to its Hotelling's $T_D^2{}_i$ statistic, indicating a variability problem. Figures 7a and 7b show the first two major Principal Axis Plots. We can see that our chief concern should be variable diagonal translation - perhaps due to problems with wafer or projection lens fixtures. Secondarily, we may choose to try to reduce an expansion/compression problem - perhaps due to projection lens distortions or wafer distortions.

Another subgroup would have been declared "out-of-control" due to high values of both of its Hotelling's T^2 statistics, had not robust principal components been used. It was contaminated by an outlying observation. Robust principal components rejected observation #7, as shown in Figures 9 and 10. When observation #7 was replaced by one taken from another wafer, the subgroup was no longer out of control, according to T^2 statistics.

The example could be continued to illustrate the complexity and subtlety of possible multivariate phenomena and to provide illustration of the various ways that the proposed control machinery could be used. Instead, the reader can study the different paths of the procedure provided in Section 5.1 .

6. Summary

Hotelling's T^2, robust principal components, and appropriate graphical displays can be combined to form a sophisticated system for statistical process control when many correlated performance variables are measured in the same physical unit. We need not be so concerned about the computational complexity or burden of such a SPC system. Rather, the harder task is to develop systems which are not only statistically sound, but can lead to meaningful, interpretable results which guide corrective action. In brief: (1) Hotelling's $T_M^2{}_i$ and $T_D^2{}_i$ statistics are the *multivariate analog* of $\bar{X}$ and R charts, and should be used rather than a multitude of simultaneous $\bar{X}$ and R charts or the standard T^2; (2) Use of robust principal components analysis helps us to avoid misinformation due to atypical data - such as outliers and departures from the standard process correlation structure. It also makes possible more subtle interpretation of multivariate data; (3) High-resolution computer graphics are widely available, and graphical tools for SPC, such as Spider Plots and Principal Axis Plots, can be built into an interactive SPC system. Lastly, the only way such a complex SPC system can work is to have well-trained process engineers who are dedicated to improving the process, and who have good analytic and interpretive skills.

The application of this control methodology to wafer grid misalignment is only one example of its potential use. With slight modification it can be applied to the case of any measurement taken at different locations on a wafer - sheet resistivity, thickness, inductance, etc. More important, it also applies to other manufactured products which have many measurements taken in the same physical units. For example, it could be applied to registration patterns on PC boards, emulsion thickness or purity on photographic film, thickness or density of "uniform" sheets of steel or some other material, diameters of spheres or cylinders, such as ball bearings or rods, or physical dimensions of arbitrarily-shaped parts. The Exaggerated-Measurement Plot, Spider Plot, and Principal Axis Plot can be generalized for these other areas of application. The Exaggerated-Measurement Plot and Principal Axis Plot can be replaced by contour or 3D plots. The Spider Plot is harder to generalize, but a glyph plot can be used.

Areas for further research include distributional theory for small sample sizes, analogs to the L_2-norm-based covariance matrix and principal components - perhaps based on other norms, and ways to catalog process-related patterns.

REFERENCES

[1] Lord, E.A., and Wilson, C.B., *The Mathematical Description of Shape* and *Form*, Halsted Press, division of John Wiley & Sons, New York (1984), p. 35.

[2] American Society for Quality Control Statistics Division, "Glossary and Tables for Statistical Quality Control, 2nd edition," ASQC, 1983, p. 100.

[3] Hunter, J.S., "The Technology of Quality," RCA Engineer, Vol. 30, No. 3, May/June, 1985, p. 10.

[4] Jackson, J.E., "Principal Components and Factor Analysis: Part II - Additional Topics Related to Principal Components," Journal of Quality Technology, Vol. 13, No. 1, January, 1981, pp. 46-58.

[5] Jackson, J.E., and Bradley, R.A., "Sequential χ^2- and T^2 Tests and Their Application to an Acceptance Sampling Problem," Technometrics, Vol. 3, No. 4, November, 1961, pp. 519-534.

[6] Seber, G.A.F., "Multivariate Observations," John Wiley & Sons, New York (1984), pp. 181-187.

[7] Barton, R., personal memos at the RCA Laboratories, David Sarnoff Research Center, dated 1984 and 1985.

[8] Devlin, S.J., Gnanadesikan, R., and Kettenring, J.R., "Robust Estimation of Dispersion Matrices and Principal Components," Journal of the American Statistical Association, June 1981, Vol. 76, No. 374, pp. 354-362.

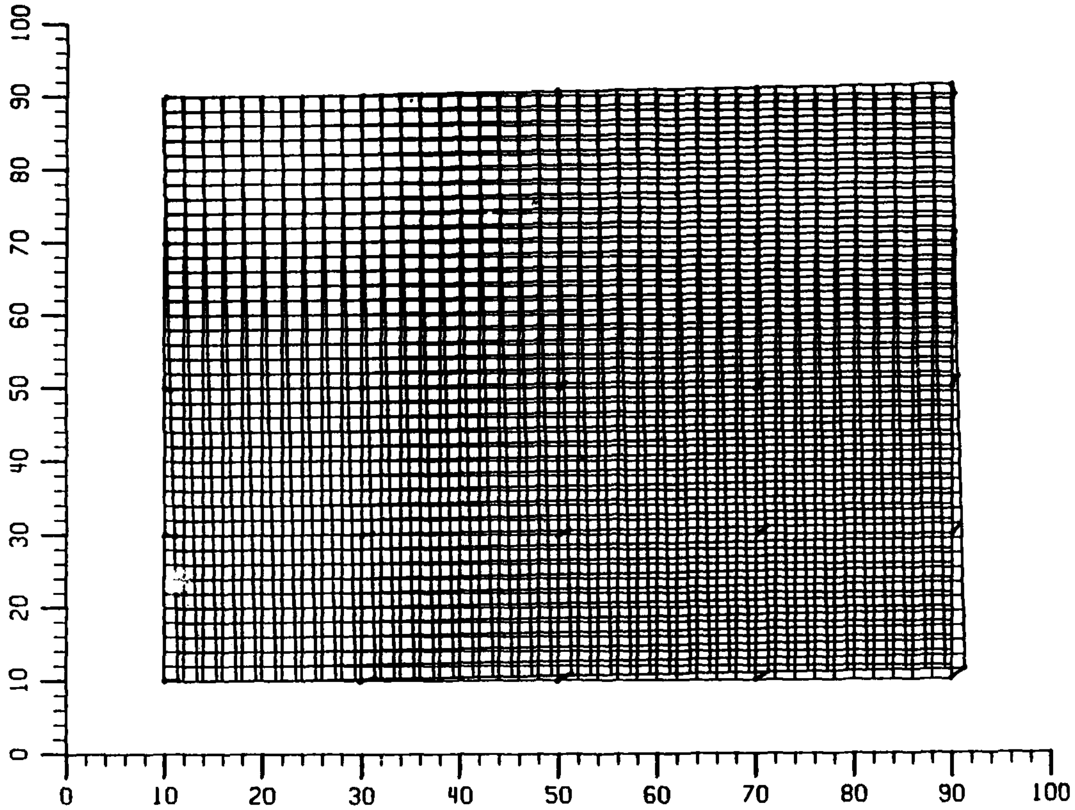


Figure 1

An extreme case of grid misalignment, due to manufacturing variations, on a (perhaps monitor) semiconductor wafer. A 5-by-5 array of misalignment measurements is taken as a relative displacement of grid A to grid B. The locations of the measurements can either be absolute or determined, de facto, by one of the grids.

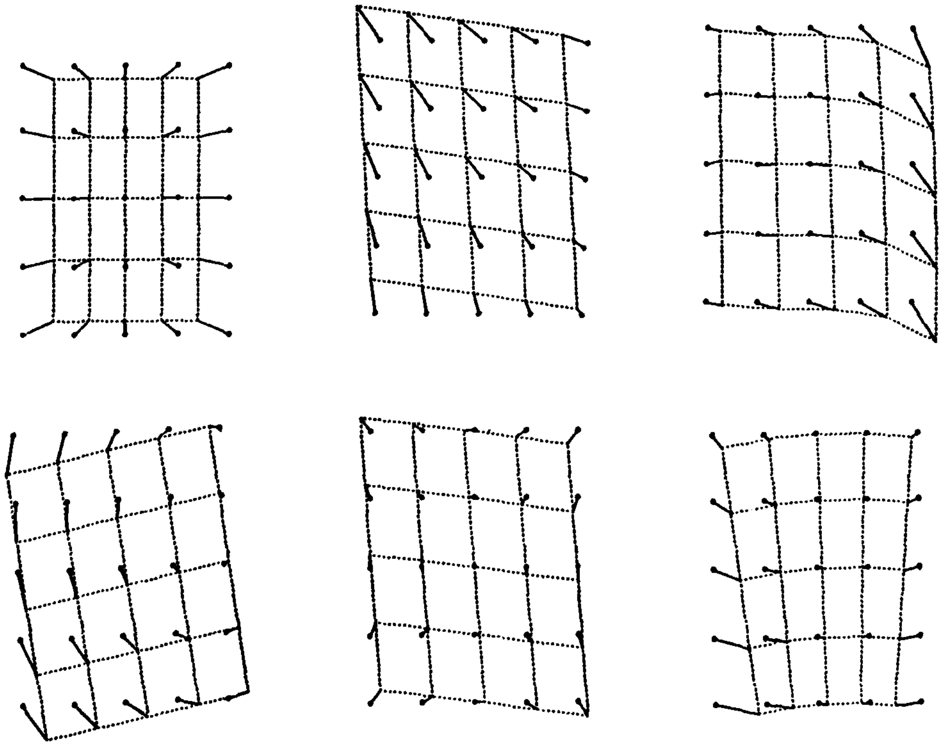


Figure 2

Various one-to-one mappings (groups of motions) of grids: (a) rotation (and translation). (b) dilation, (c) shearing, (d) perspective, (e) inversions in circles (and reflections in lines). (f) differentiable mappings (diffeomorphisms), (g) continuous (topological) mappings

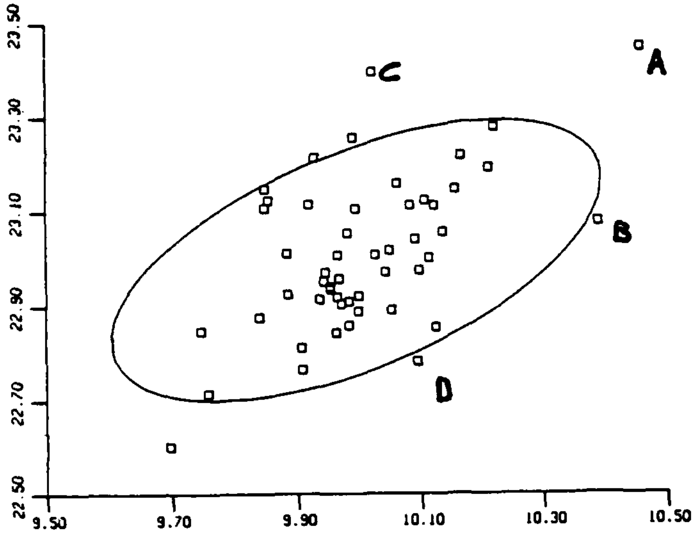
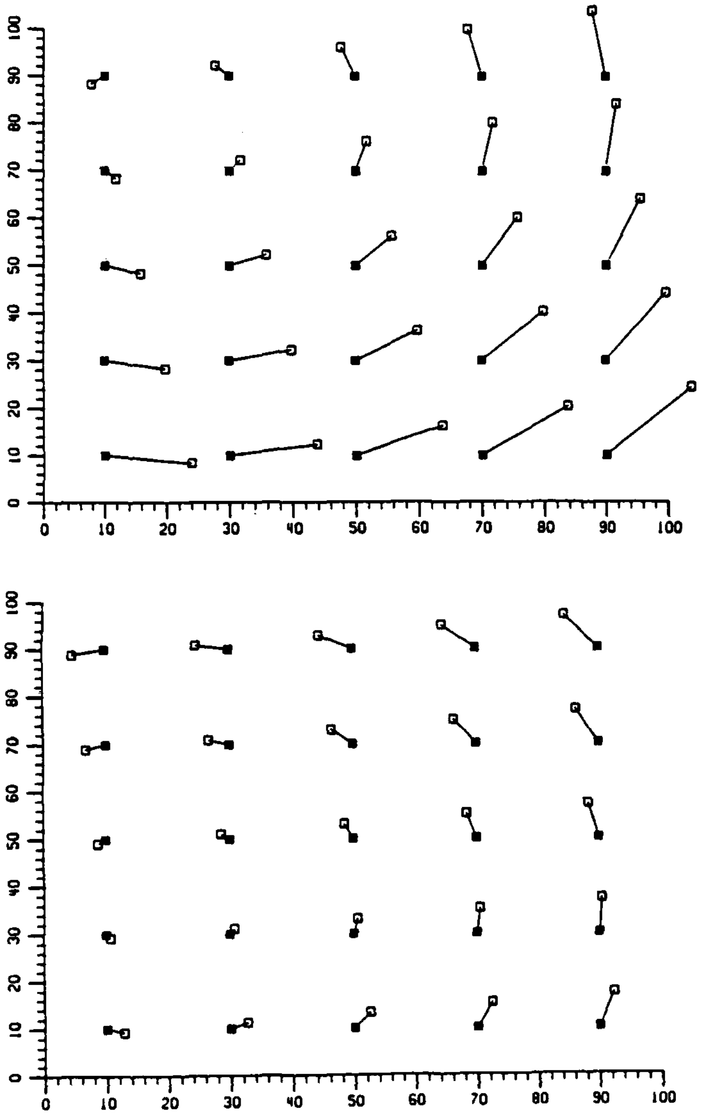


Figure 3

A Hotelling's T^2 control ellipse for the bivariate control case. Point A is extreme in both factors. Points B and C are extreme in only Factors 1 and 2 respectively. Point D is not extreme in any *one* factor, but *nonetheless* falls outside the control ellipse.



Figures 4a and 4b

Two Exaggerated-Measurement Plots for wafer grid misalignment. The direction of measured misalignment is preserved, but the magnitude is exaggerated to make it easier to identify patterns of misalignment. Both show an apparent rotation.

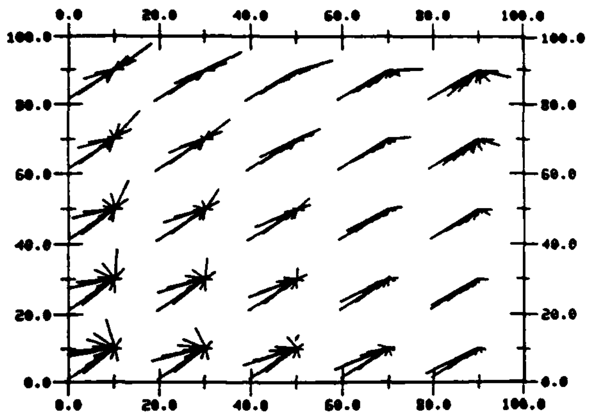


Figure 5

Overlaid Exaggerated-Measurement plots enable us to see subgroup bias and variability on a per-location basis. A superior means for presenting the same data is the Spider Plot, shown in Figure 6.

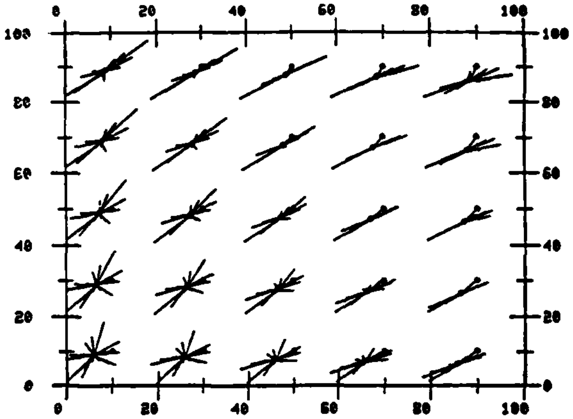


Figure 6

The Spider Plot enables us to qualitatively separate subgroup bias from variability on a per-location basis, using exaggerated misalignment errors. The nominal measurement locations and subgroup means are indicated by solid dots.

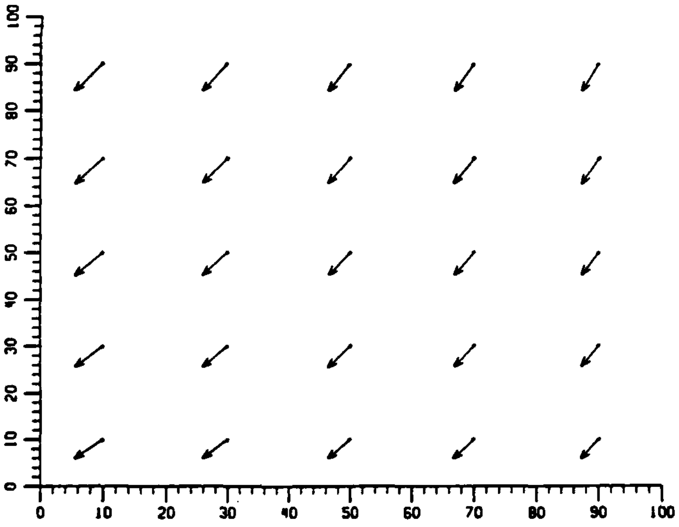


Figure 7a

The First Principal Axis Plot for measurements on the wafer subgroup displayed in the Spider plot of Figure 6. The dot indicates the nominal measurement location, and the arrow indicates the direction and relative magnitude of variation at that measurement location - as "accounted for" by this principal axis.

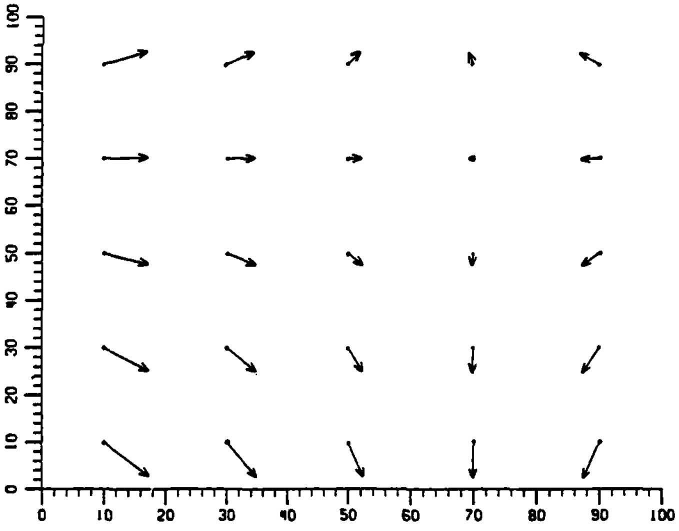


Figure 7b

The Second Principal Axis Plot for the same wafer subgroup. The first principal axis accounted for nearly 88 per cent of the total variance in the raw misalignment data; this one accounts for nearly 12 per cent.

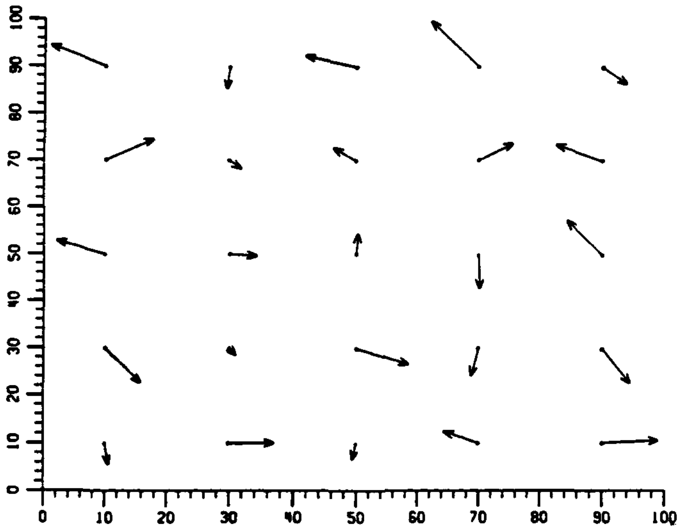


Figure 8

A "worst-case" Principal Axis Plot - in the sense of identifying patterns of variability.

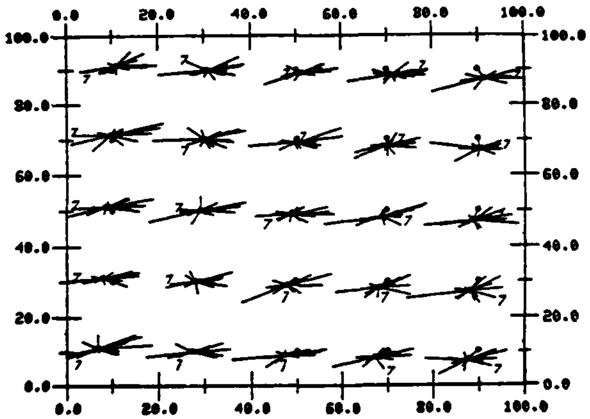


Figure 9

The Spider Plot for a sample of wafers containing one outlier, wafer #7. Wafer #7 cannot be easily selected from this plot as an outlier, because its variability does not follow the covariance structure in S^*. See Figure 10.

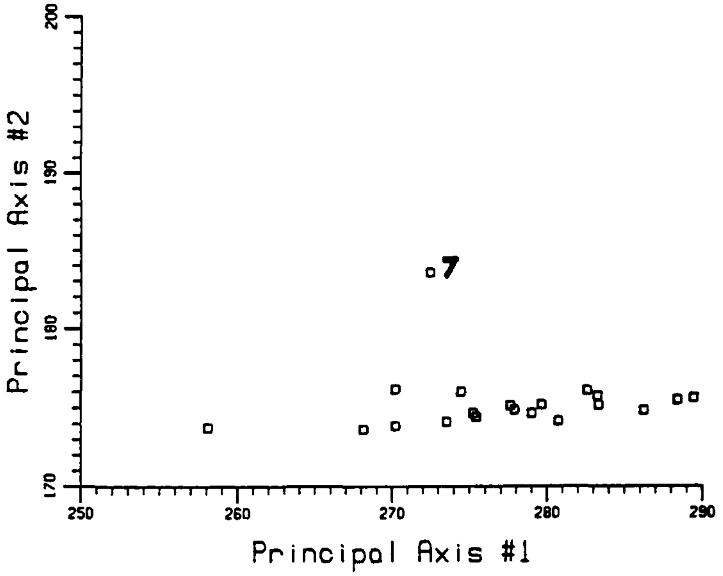


Figure 10

The first two principal components for the same wafer sample as illustrated in Figure 9. Wafer #7 stands out in this plot as an outlier. The complex multivariate nature of the measurements renders it nearly impossible to identify outliers of this type without principal components analysis.

COLLINEARITY AND POINTS OF EXPANSION IN POLYNOMIAL REGRESSION
(Interim Report)

Michael F. Driscoll, Arizona State University

ABSTRACT

Frequent use is made of single- and multi-variable polynomial regression models in situations for which prior knowledge fails to suggest a specific response function. In such models, the numerical and statistical stability of the least-squares estimators are highly dependent on the point of expansion, or origin, used for the underlying predictor variables. Results of some mathematical analyses of this phenomenon are given. Related issues in variable selection and diagnostic checking are also considered.

1. INTRODUCTION

The purpose of this paper is to give a overview of issues which arise when doing ordinary least-squares fitting of functional relationships described only as polynomials in one or more variables. This method, herein called the <u>polynomial approach</u>, is commonly used in pilot studies and is necessary whenever prior knowledge is not detailed enough to postulate a more specific relational form for examination.

In Section 2, the polynomial approach to a regression problem is described very generally as a problem of estimating the coefficients in a power series approximation to a smooth function. Assumptions and difficulties inherent in this use of power series are discussed in Section 3. The question of selecting an optimal point about which to express the series expansion is considered in Sections 4 and 5. Some final remarks are made in Section 6.

This paper is designated an interim report since the research summarized in it is still incomplete (especially in Section 5). I have proofs for some results; sketches of the more detailed of these are given in the Appendix. Statements which I believe to be true but have not yet proven are offered as conjectures.

2. THE POLYNOMIAL APPROACH

The aim of multiple least-squares linear regression analysis is to determine a suitable model

(2.1) $$Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_t X_t + \varepsilon$$

for describing the relationship between a <u>response variable</u> Y and several <u>predictor variables</u>

(2.2) $$X_i = f_i(U_1,\ldots,U_b)$$

defined from certain underlying or <u>basic variables</u> $U_1,\ldots,U_b$ available to the analyst. It is often not clear at the outset what form the predictors (2.2) should have, so a common method is to use predictors of the form

(2.3) $$X(\underline{p}|\underline{c}) = (U_1-c_1)^{p_1}\,(U_2-c_2)^{p_2}\cdots(U_b-c_b)^{p_b},$$

whose definition depends on the vectors $\underline{p} = (p_1,\ldots,p_b)'$ and $\underline{c} = (c_1,\ldots,c_b)'$. The effect of the polynomial approach is that the unknown <u>true response function</u>,

(2.4) $$E(Y) = g(U_1,\ldots,U_b)$$

(say), is replaced by an unknown <u>approximate response function</u>

(2.5) $$E(Y) \approx \Sigma\, \beta(\underline{p}|\underline{c})\, X(\underline{p}|\underline{c})$$

which is a partial power series expansion of (2.4) about the point $\underline{c}$ in the space of the basic variables. The need is then to obtain a <u>fitted response function</u>

(2.6) $$\hat{Y} = \Sigma\, \hat{\beta}(\underline{p}|\underline{c})\, X(\underline{p}|\underline{c}),$$

that is, to obtain the least-squares estimates $\hat{\beta}(\underline{p}|\underline{c})$ of the parameters $\beta(\underline{p}|\underline{c})$. The highest value of

$$p_1 + \cdots + p_b$$

among the predictors used is called the <u>order</u> of the model, and is herein denoted by P.

3. CONSEQUENCES OF SERIES APPROXIMATION

The polynomial approach to a regression problem entails several conceptual and practical concerns which can be illuminated by emphasizing the effect of the series approximation of (2.4) by (2.5). One is the tacit assumption that the true response function has only very mild discontinuities, if any, in the pertinent part of the domain of the basic variables. This assumption is in fact unavoidable: if the analyst's knowledge is insufficient to suggest an approach more specific than one based on power series approximation, then he is unlikely to have much information on continuity properties.

Choice of the order of the series approximation (2.5) is a more immediate concern. If (2.4) is assumed to be continuous in its several arguments, then there is no question but that a large enough value of P for (2.5) will produce an approximate response function which is practically indistinguishable from the true response function. In single-variable problems (those with just one basic variable U) one has considerable flexibility in selecting P. But in multi-variable problems, one's choice of P is limited by computational requirements and, more importantly, by the difficulty of interpreting interaction terms. The larger the number of basic variables, the less workable are the higher order approximations. This aspect of the polynomial approach is well understood, so models of low order are often used.

The most important indeterminate in (2.5) is the point of expansion or centering, $(c_1,\ldots,c_b)$. It plays a pivotal role in the expression of the series approximation, and therefore can have an overwhelming impact on the fitted response function. However, it is absent in the true response function (2.4).

There is some controversy over the idea of changing scale or location in regression predictors, due to the possible affect on diagnostics for predictor ill-conditioning. The paper by Belsley (1984) and the attendant comments by others discuss some fundamental advantages and disadvantages of mean-centering the predictors. The centering being considered in this paper is the selection of a point of expansion in the space of the basic variables, around which the predictors are to be constructed; centering of the predictors themselves is not being treated.

The true response function does not depend on the point of expansion, so the description of the response function implied by its approximate form (2.5) and obtained from the regression fit (2.6) must be invariant to the point of expansion used. In other words, the fitted response functions obtained from various points of expansion must be equivalent descriptions of the true response function. This requirement is met only if (2.6) is complete in the sense that it contains all predictors through a given order, P — that is, includes all terms for which the elements of p satisfy the condition

$$0 \le p_1 + \cdots + p_b \le P$$

— or in the weaker sense that it contains all terms which can be produced by striking one or more factors from any predictor present.

Completeness in polynomial models is discussed in detail in Driscoll and Anderson (1980). The need for it is not widely enough appreciated by practitioners. Predictor selection done under the polynomial approach is often flawed, resulting in misapplications which are analogous to (if less transparent than) using a zero intercept model without intending to do so. Griepentrog, Ryan, and Smith (1982) allude to this issue in discussing the affect which changes of location and scale in the basic variables can have on the t-tests for lower-order coefficients in the polynomial model. The example they give is meant to illustrate that such t-tests may be meaningless. It is more properly viewed as an argument in favor of a logically prior requirement: that of respecting any hierarchy present among the predictors used. This is more easily accomplished today than in the past, although it still requires some user effort, especially if an "all" subsets analysis is desired. At least program BMDP2R (Dixon, 1983, Appendix F.4) now has facilities for entering and removing predictors in a specified sequence or by defined groups; and, of course, there is program BMDP5R for single-variable models. Procedure STEPWISE (SAS Institute, Inc., 1985, p. 764) has no such facilities, but the manual does refer the user to a survey article by Hocking (1976).

On the other hand, the fact that the point of expansion is indeterminate has its advantages. In particular, it allows the analyst to choose the point of expansion so as to satisfy other needs. One auxiliary goal is to mitigate the numerical and statistical instabilities arising from the collinearity pandemic in polynomial regression models. Another is to enhance the interpretability of the resulting fitted model (2.6). Snee and Marquardt (1984), in their comment on Belsley's paper, give an analysis of some tree-volume data which nicely illustrates this benefit of point of expansion indeterminacy.

Discussion now turns to criteria for selection of the point of expansion. The obvious starting point is with one basic variable, that is, with single-variable polynomial regression.

4. ONE BASIC VARIABLE

In the single-variable polynomial regression model, the predictors (2.3) are

$$(4.1) \qquad X(p|c) = (U - c)^p \ , \ p = 0,1,\ldots,P \ ,$$

where P is the order of the approximate and fitted response functions. The investigation into the choice of the point of expansion will be done in sample terms to avoid the need for distributional assumptions. A sample u_1 , u_2 , ... , u_n of observed values of the basic variable provides samples

$$(4.2) \qquad x_i(p|c) = (u_i - c)^p \ , \ i = 1,2,\ldots,n \ ,$$

on each of the predictors. To avoid annoying qualifications in what follows, it is assumed throughout that the u_i's contain at least three distinct values. [The case of just two distinct values is trivial. Also, the purposes of this paper do not require reference to observed response values.]

4.1. Scalar Moments

The usual sample moments of the predictors have strong properties. The means

$$(4.3) \qquad m(p|c) = \Sigma_i \ (u_i - c)^p \ / \ n$$

have first and second derivatives $-pm(p-1|c)$ and $p(p-1)m(p-2|c)$ with respect to c. [The symbol $m(\cdot|c)$, and others like it which appear below, is to be interpreted as zero if the argument is negative.]

It follows immediately that: (1) if p is odd, the mean is monotone decreasing in c with a unique zero; (2) if p is even, the mean is convex, so it uniquely attains an minimum; (3) these zeros and minima occur between $\min_i u_i$ and $\max_i u_i$. If the u_i's are symmetric about their mean $\bar{u} = m(1|0)$, then the zeros and minima all occur at $\bar{u}$, suggesting that this is a good point of expansion.

The predictor covariances

$$(4.4) \qquad s(p,q|c) = m(p+q|c) - m(p|c) \ m(q|c)$$

have first derivatives

$$(4.5) \qquad -p \ s(p-1,q|c) - q \ s(p,q-1|c)$$

and second derivatives

$$(4.6)\quad \begin{aligned} &p(p-1)\ s(p-2,q|c)\\ &\quad + 2pq\ s(p-1,q-1|c)\\ &\quad + q(q-1)\ s(p,q-2|c) \end{aligned}$$

with respect to c. Since (4.4) is zero if $pq = 0$ and positive when $pq > 0$ and $p+q$ is even (see the Appendix for a proof), these covariances have monotonicity and convexity properties analogous to those of the predictor means. In particular, if the u_i's are symmetrically placed then, at $c = \bar{u}$, $s(p,q|c)$ is zero for $p+q$ odd and minimized for $p+q$ even.

The behavior of the predictor correlations is more intriguing. In the interests of tractability, only the symmetric case is considered here, and in terms of the coefficients of determination

$$(4.7)\quad r^2(p,q|c) = s^2(p,q|c) \ / \ s(p,p|c)\ s(q,q|c)\ .$$

For $pq = 0$ or $p+q$ odd, (4.7) clearly has an absolute minimum at $c = \bar{u}$. In the case that $p+q$ is even (ignoring the case that $p = q$), $r^2(p,q|c)$ appears to have a relative maximum at $c = \bar{u}$. I can prove this if p and q are odd (see the Appendix), and I conjecture it to be true when p and q are even.

Specific illustrations of the behavior of these coefficients of determination are available in Bradley and Srivastava (1979) and Hackney and Mohammad (1978). These similar papers include formulas expressing (4.7) as a rational function with coefficients written in terms of the means $m(\cdot|\bar{u})$, and give a graphs of $r^2(1,2|c)$ for two symmetric situations. The graphs were produced by assigning to the $m(\cdot|c)$'s the values of the corresponding population moments first from the standard normal and second from the rectangular distribution. Hackney and Mohammad also plot $r^2(1,3|c)$ for these situations and note that $c = \bar{u}$ is a point of relative maximum. Further, they note in these situations the patterns

$$(4.8)\quad r^2(p-2,q|\bar{u}) \le r^2(p,q|\bar{u}) \le r^2(p+1,q+1|\bar{u})$$

(for $p+q$ even) among the correlations of the first 12 powers of the centered basic variable, and suggest that it may therefore suffice in non-symmetric situations to choose c so as to achieve a minimal value for the highest-order ($p+q$ even) determination among the predictors being used. Since I think the nature of (4.7) needs to be further investigated before this suggestion can be evaluated, I have not yet tried to prove (4.8).

4.2. Matrix Moments

Although the mathematical analysis of the scalar moments of the predictors (2.3) is incomplete, the results obtained thus far do indicate that a more encompassing approach would be valuable. What is required is to study the predictors as a total ensemble rather than one or two at a time, that is, to consider matrix moments. Such analysis should provide guidance for selecting the point of expansion in a way which better controls the covariance structure of the least-squares estimators.

The purpose of this subsection is to present a particular anatomy of the predictors which seems to promise results of this kind. The anatomy has a matrix formulation.

The design matrix of the order P single-variable polynomial regression model is, from (4.1) and (4.2), the $n\times(1+P)$ matrix

$$(4.9)\qquad \underline{X}(c) = [(u_i-c)^p]\ ,$$

where $i=1,\dots,n$ and $p=0,1,\dots,P$. This matrix has first derivative (see, for example, Rogers, 1980)

$$(d/dc)\ \underline{X}(c) = [-p(u_i-c)^{p-1}]$$

with respect to c. Let $\underline{J}$ be the $(1+P)\times(1+P)$ matrix with $-p$ as element $(p-1,p)$ for $p=1,\dots,P$ and other elements zero, that is, the super-diagonal matrix $\underline{J} = \text{supdiag}\{-1,\dots,-P\}$. Then

$$(4.10)\qquad (d/dc)\ \underline{X}(c) = \underline{X}(c)\underline{J}\ .$$

Using results from elementary matrix differential equations (see, for example, Finkbeiner, 1966, Chapter 10) on (4.10), or by direct matrix calculation, one can show that $\underline{X}(c)$ satisfies

$$(4.11)\qquad \underline{X}(c) = \underline{X}(0)\exp(\underline{J}c)\ ,$$

where $\exp(\underline{J}c)$ denotes the matrix exponential function

$$(4.12)\qquad \begin{aligned} \exp(\underline{J}c) &= \Sigma_p\ \underline{J}^p c^p/p!\\ &= \underline{I} + \underline{J}c + \underline{J}^2c^2/2! + \cdots + \underline{J}^P c^P/P!\ , \end{aligned}$$

the series being finite here because $\underline{J}$ is nilpotent of order $1+P$.

The analogy between (4.12) and the scalar exponential function is obvious, as is that of (4.10) and (4.11). In further analogy, the inverse of (4.12) exists and is given by

$$(4.13)\qquad \{\exp(\underline{J}c)\}^{-1} = \exp(-\underline{J}c)\ .$$

Using (4.12) and (4.13) it is readily shown via mathematical induction that the elements of these matrices are, for $p,q=0,1,\dots,P$,

$$(4.14)\qquad \{\exp(\underline{J}c)\}(p,q) = {}_qC_p\ (-c)^{q-p}$$

and

$$(4.15)\qquad \{\exp(-\underline{J}c)\}(p,q) = {}_qC_p\ c^{q-p}$$

where ${}_qC_p$ denotes the binomial coefficient q-choose-p. In particular, these matrices are upper-triangular.

One can also express the relation between the parameter estimates in (2.6) by the algebraic equality

$$(4.16)\qquad \underline{\beta}\hat{}(c) = \exp(-\underline{J}c)\underline{\beta}\hat{}(0)\ .$$

[Reference here is to the least-squares estimators; computational instabilities in producing numerical estimates are for the moment being ignored.] It is apparent from (4.15) and (4.16) that $\beta\hat{}(P|c) = \beta\hat{}(P|0)$. This is a reflection of the well known fact that changing the location of the basic variables in a polynomial model does not alter the estimates of the highest-order parameters.

Equations (4.11) and (4.16) succinctly state the effect of taking c rather than zero as the point of expansion for the polynomial approach. Using (4.11) first with c itself and then with $c=\bar{u}$ gives

$$\underline{X}(c) = \underline{X}(\bar{u})\exp\{\underline{J}(c-\bar{u})\} ,$$

which after pre-multiplication by a vector of (1/n)'s yields a matrix form of the equations Bradley and Srivastava (1979) gave for computing the means (4.3). If one changes scale by a factor d after changing location by −c, then the right-hand side of (4.11) should be post-multiplied by the diagonal matrix $D=\text{diag}\{1,d,d^2,\ldots,d^P\}$. If scale is changed first then a pre-multiplication is required, as in

$$\underline{X}(d,c) = \underline{X}(1,0)\underline{D}\exp(\underline{J}c) ,$$

which is the general form of the triangular linear transformation which Griepentrog, Ryan, and Smith (1982) considered for the case P=2.

The normal-equations coefficient matrix and its inverse are of prime interest at this point. The anatomy summarized by (4.11) gives them the forms

$$\begin{aligned}(\underline{X}'\underline{X})(c) &= \{\underline{X}(c)\}'\{\underline{X}(c)\} \\ &= \exp(\underline{J}'c)(\underline{X}'\underline{X})(0)\exp(\underline{J}c)\end{aligned}$$

and

$$(\underline{X}'\underline{X})^{-1}(c) = \exp(-\underline{J}c)(\underline{X}'\underline{X})(0)\exp(-\underline{J}'c) .$$

The fact that these matrix functions have such similar forms is promising for further mathematical analysis. Investigations are incomplete as yet, but a few initial indications can be given.

An easy result of negative nature is the following one. Since $\underline{J}$ is nilpotent, its eigenvalues are all zero, those of $\exp(\underline{J}c)$ are all $\exp(0c)=1$, so $|\exp(\underline{J}c)|=1$ and $|(\underline{X}'\underline{X})(c)| = |(\underline{X}'\underline{X})(0)|$ for all c. This shows that D-optimality is a worthless criterion in the single-variable polynomial approach, which is in marked contrast to other regression situations (see, for example, Bates, 1983).

It appears that, as is the case for $(\underline{X}'\underline{X})(c)$, the diagonal elements of $(\underline{X}'\underline{X})^{-1}(c)$ are convex in c; the first P of them strictly so, the last being constant. In the symmetric case, the minima would occur at $c=\bar{u}$. These results would be rather useful, since the diagonal elements of $(\underline{X}'\underline{X})^{-1}(c)$ appear in the variances of the least-squares estimators, and although I have not yet achieved proof of them I do conjecture them to be true. Their proof and the discovery of other facts about this inverse matrix will likely use some properties of Vandermonde, Hankel, and Toeplitz matrices.

5. SEVERAL BASIC VARIABLES

The case of several basic variables obviously awaits solution of the single-variable case. Nonetheless it is clear from results obtained so far and from general principles that an optimal situation is that the basic variables be uncorrelated and symmetric around their respective means.

6. FINAL REMARKS

Bradley and Srivastava (1979) recommend that non-essential collinearity in a regression model be reduced by appropriate choice of a point of expansion and that, when possible, collinearity inherent in the basic variables be removed by good experimental design. That goal — joined with a belief that mathematical analysis can give insight into the effect of the point of expansion in the polynomial approach — is the motivation for the present work.

The disagreements about the benefits and methods of data centering, in particular Belsley's (1984) paper and the comments on it, may have some relevance to selecting a point of expansion. It is not clear what conclusions will ultimately derive from such debates, not for centering of predictors and certainly not for centering of basic variables. What is clear is that losing sight of the distinction between the model and any given description of it is disastrous to understanding. Herr (1980) has made this latter point by comparing the geometric (or coordinate-free) and algebraic approaches to linear models. Jacobowitz and Driscoll (1980) give a more mathematically abstract presentation which distinguishes between the properties inherent in the model, those in the model under a particular parametrization, and those in the parametrized model with an explicit coordinate representation. Such distinctions are especially needed when discussing and using the polynomial approach to linear regression analysis.

REFERENCES

Bates, Douglas (1983), "The Derivative of |X'X| and Its Uses," Technometrics 25, 373-376.

Belsley, David A. (1984), "Demeaning Regression Diagnostics through Centering," The American Statistician, 38, 73-77, with comments and reply, 78-93.

Bradley, Ralph O. and Srivastava, S.S. (1979), "Correlation in Polynomial Regression," The American Statistician, 33: 11-14.

Dixon, Wilfrid J., ed. (1983), BMDP Statistical Software Manual (1983 and 1985 printings), Berkeley: University of California Press.

Driscoll, Michael F. and Anderson, Dennis J. (1980), "Point-of-Expansion, Structure, and *Selection in Multivariable* Polynomial Regression," Communications in Statistics: Theory and Methods, A9 (8), 821-836.

Finkbeiner, Daniel T., II (1966), Introduction to Matrices and Linear Transformations, 2nd ed., San Francisco: W.H. Freeman.

Griepentrog, Gary L., Ryan, J. Michael, and Smith, L. Douglas (1982), "Linear Transformations of Polynomial Regression Models," The American Statistician, 36: 171-174.

Gurland, John (1967), "An Inequality Satisfied by the Expectation of the Reciprocal of a Random Variable," *The American Statistician*, 21: 24-25.

Hackney, O.P. and Mohammad, L.A. (1978), "Multicollinearity in Polynomial Regression," *1978 Proceedings of the Statistical Computing Section of the American Statistical Association*, 160-164.

Herr, David G. (1980), "On the History of the Use of Geometry in the General Linear Model," *The American Statistician*, 34, 43-47.

Hocking, R.R. (1976), "The Analysis and Selection of Variables in Linear Regression," *Biometrics*, 32: 1-40.

Jacobowitz, Ronald, and Driscoll, Michael F. (1980), "Estimability, Testability, and Parametrization in Fixed-Effects Linear Models," Technical Report Number 47, Department of Mathematics, Arizona State University, Tempe, Arizona, 141 pp.

Patel, Jagdish K., Kapadia, C.H., and Owen, D.B., (1976), *Handbook of Statistical Distributions*, New York: Marcel Dekker.

Rogers, Gerald S. (1980), *Matrix Derivatives*, New York: Marcel Dekker.

SAS Institute, Inc. (1985), *SAS User's Guide: Statistics, Version 5 Edition*, Cary, N.C.: SAS Institute, Inc.

Sclove, S.L., Simons, G., and van Ryzin, John, (1967), "Further Remarks on the Expectation of the Reciprocal of a Positive Random Variable," *The American Statistician*, 21, 33-34.

Snee, Ronald D. and Marquardt, Donald W. (1984), "Collinearity Diagnostics Depend on the Domain of Prediction, the Model, and the Data" (comment on "Demeaning Conditioning Diagnostics through Centering" by David A. Belsley), *The American Statistician*, 38, 83-87.

APPENDIX

This appendix sketches the more detailed of the proofs for results stated in the body of the paper. I would appreciate receiving information about alternate or expanded proofs.

A.1. Covariances Positive

When p and q are positive and $p+q$ is even, $s(p,q|c)$ is bounded below by the covariance between powers p and q of $|u_i-c|$. Applying the lemma in Gurland (1967) with $f(\cdot)$ and $g(\cdot)$ as the corresponding power functions and X as a random variable taking values $|u_i-c|$ with probability proportional to the multiplicity of u_i among $u_1,\dots,u_n$, one sees that $s(p,q|c)$ is nonnegative. The value can be zero only if the $|u_i-c|$ are equal for all i, which does not occur since (by assumption) $u_1,\dots,u_n$ contain at least three distinct values.

A.2. Relative Maxima in $r^2(p,q|c)$

The case to be considered is pq positive, $p+q$ even, and $u_1,\dots,u_n$ symmetric about $\bar{u}$. Use the derivatives (4.5) and (4.6) to show that $r^2(p,q|c)$ has a critical point at $c = \bar{u}$ and that its behavior there is the same as that of the function

$$h(c) = s^2(p,q|c) - s(p,p|c) - s(q,q|c) \ .$$

To show that $r^2(p,q|c)$ has a relative maximum at $\bar{u}$, it now suffices to prove that

$$\begin{aligned} h''(\bar{u}) = 2\, & s(p,q|\bar{u})\, s''(p,q|\bar{u}) \\ & - s(p,p|\bar{u})\, s''(q,q|\bar{u}) \\ & - s(q,q|\bar{u})\, s''(p,p|\bar{u}) \end{aligned} \tag{A.1}$$

is negative. Here, $''$ denotes the second derivative with respect to c, so that $s''(\cdot,\cdot|\bar{u})$ is given by (4.6).

Expand (A.1) using (4.6) to express $h''(\bar{u})$ entirely in terms of undifferentiated covariances. Then use the positive definiteness of the covariance matrix $\{s(p,q|\bar{u})\}$ to show that the three terms having covariances between powers $p-1$ and $q-1$ as factors in their coefficients have a sum which is negative. This step reduces the problem to showing that

$$\begin{aligned} & p_0\, s(p-2,q|\bar{u})\, s(p,q|\bar{u}) \\ & + q_0\, s(p,q-2|\bar{u})\, s(p,q|\bar{u}) \\ & - p_0\, s(p-2,p|\bar{u})\, s(q,q|\bar{u}) \\ & - q_0\, s(q,q-2|\bar{u})\, s(p,p|\bar{u}) \ , \end{aligned} \tag{A.2}$$

in which p_0 denotes $2p(p-1)$ and q_0 denotes $2q(q-1)$, is nonpositive.

Now, reduce to the subcase that p and q are odd and, without loss of generality, take p less than q. The resulting simplification in (A.2) is that all occurrences of "s" are replaced by "m," so that one may directly deal with predictor means. At this point, use

$$\begin{aligned} & m(p+q|c)\, m(p+q-2|c) \\ & \le m(2p|c)\, m(2q-2|c) \\ & \le m(2p-2|c)\, m(2q|c) \end{aligned} \tag{A.3}$$

to show that (A.2) is nonpositive, completing the proof.

The moment inequalities in (A.3) can be established by judicious use of a result of Sclove, Simons, and van Ryzin (1967) which is listed in Patel, Kapadia, and Owen (1976, p.47). The facts that p and q are positive and that p is less than q are important.

In the subcase that p and q are even, one can of course use (4.4) to express (A.2) explicitly in terms of predictor means. But I have not been able to prove in this subcase that (A.2) is nonpositive. While I believe it is, I also believe that the proof will be delicate.

A WORKSTATION-BASED ENVIRONMENT FOR STATISTICAL ANALYSIS OF SET-VALUED DATA

Lionel Galway, Carnegie-Mellon University

Abstract

Although set-valued data exists in many fields, statistical techniques for analyzing such data are not well-developed. One problem with set-valued data is that almost any non-trivial analysis makes heavy computational demands and requires computer graphics; such usage of central mainframe computers can be quite expensive. We report on the current state of development of an integrated software package to analyze two-dimensional set-valued data: it includes routines for manipulating and doing calculations on sets, facilities for generating pseudo-random sets, and provisions for graphical output and user interaction. Although much of the package has been designed to be system-independent, it takes full advantage of the unique facilities of the ANDREW window manager and the VICE distributed file system, plus the availability at CMU of powerful workstations with graphical input devices and high-resolution bit-mapped graphics displays (ANDREW and VICE were developed by the Information Technology Center at CMU to support a campus-wide network of personal workstations).

1. Introduction

Statistics is concerned with random quantities; traditionally these have been random numbers, vectors or functions. A natural extension is to study random sets in n-dimensional Euclidean space. This is important from a theoretical point of view and from a practical one as well: data from a variety of fields is naturally expressed in terms of sets. Examples can be found in geology (using sand grains' shape and size distribution to determine its provenance (Ehrlich,et.al., 1980)), stereology (determining a three-dimensional structure from two dimensional slices (Jensen, et.al, 1985)), and from fields such as computed tomography, granulometry, etc. (Trader, 1981). A theory of random sets would make possible statistical modeling and analysis of this data in a natural way.

Significant computing facilities are required to collect, store, and analyze set-valued data; lack of such facilities has been a significant obstacle to empirical work with such data. This paper describes the design of a package of computer programs to analyze and display both random and deterministic sets in the real plane. Attention is restricted here to the two-dimensional case for two reasons: the geometrical and graphics software is simplified, and a larger amount of theory is available for two-dimensional random sets. In the future the algorithms and data structures will be generalized to higher dimensions and more complex set structures.

2. Computational Requirements for Analysis of Set-valued Data

The statistical analysis of real- and vector-valued data has a long history of collection, plotting and tabulation which predated and prepared the way for probability modeling of such data. In contrast, set-valued data requires fairly sophisticated computing resources to collect and store, and to do almost any non-trivial analysis; there is virtually no history of data analysis on set-valued data and so the theory has not been data-driven.

For example, in a set of data on sand grain shapes, an average grain profile consists of about 350 points. Automated scanning equipment is clearly needed to digitize any useful number of grains. In addition, a calculation of the storage requirements for a small sample of 100 grains (assuming 32 bit floating point numbers for each coordinate) gives 100 (grains) x 350 (points/grain) x 2 (x,y) x 4 (bytes/coordinate) = 273.5 Kbytes, approximately the capacity of a floppy disk. The cost of disk storage on mainframe computers for set-valued data sets of any significant size would be prohibitive.

Analysis of a data set also requires substantial cpu time. Algorithms that manipulate geometric objects such as the digitized sand grains must access each point of each set at least once (and sometimes more often). Finally, analysis of set-valued data requires graphics input and output, and high resolution graphics devices have typically not been available even on large mainframe computers except at high costs. Even then, the use of timesharing environments has severely limited the performance of generally-available graphics systems.

The lack of such equipment has meant that most work on random sets has been restricted to theoretical studies (e.g. (Matheron, 1975, Artstein and Vitale, 1975, Cressie, 1979, Eddy, 1980, Trader, 1981, Eddy, 1982)) and in turn that little intuition or experience with real set-valued data has driven the attempts at statistical modeling. However, the recent advent of powerful personal workstations at a fairly low cost (e.g. (Crecine, 1986)) has brought together adequate computing power, disk storage, and high resolution graphics I/O together in a compact package which can be dedicated to one person. These developments suggest that for the first time an empirical or heuristic approach to the statistical analysis of set-valued data is feasible.

3. S^3: Set Statistical System

By utilizing a computer to do tedious geometrical operations on sets, a user can quickly and easily carry out set operations such as union, intersection or Minkowski addition, much as data analysts in the early part of this century used paper, pencil, and desk calculators to construct statistics for real-valued data. We have designed a software system that will act as a framework for experimenting with the analysis of set-valued data. The central goal is to provide the user with a set of algorithms which operate on planar sets, together with graphics support and a flexible user interface that will allow exploration of set-valued data and construction and evaluation of appropriate statistics suggested by the exploration. These three components are discussed in more detail in the next three sections; this is followed by a section discussing ANDREW, a set of software enhancements to UNIX^{TM} which makes development of this system feasible on a workstation.

3.1. Set Manipulation Subroutines

The statistical analysis of any set of data typically requires the computation of statistics by combining elements of the data with appropriate arithmetic operations. For sets, this could involve taking unions, intersections, or Minkowski sums, for example, or transforming the sets in various ways. The lowest level of the system is a library of programs that implement operations on geometric objects in the plane (and a set of data structures for representing those geometric objects). These routines will form the basis for computing statistics on set-valued data. Careful design and implementation of these basic routines is essential, since their efficiency will determine the size of the sample which can be processed in a reasonable time.

3.2. Graphics Routines

The second part of S^3 is a library of routines to support graphical display of planar sets using a high-resolution bit-mapped display and graphical user input. Computer graphics allows visualization of random sets in the plane and the display of associated function representations such as the support function and the boundary function (Valentine, 1964).

The graphics programs fall into two categories:

1. Routines which build and manipulate an "abstract" display environment. These programs maintain lists of geometric objects to be displayed and information on the current real coordinate system and its relation to the graphics device in use. These routines are device-independent.

2. Programs which are specific for the Andrew programming environment and its graphics facilities.

3.3. User Interface

The final goal is a package, much like current statistical packages for real- and vector-valued data, which will allow easy entry of data and convenient user interaction. It should also be extensible (like S or ISP) to allow users to conveniently compute tentative statistics and test their performance on real and simulated data.

3.4. Andrew

Andrew is a system of hardware-independent extensions to the UNIX™ operating system, written by the Information Technology Center (Crecine, 1986, Morris,et.al., 1986) at Carnegie-Mellon University, which runs on several high performance workstations which are now available. It allows a user application program to manage several windows on a high-resolution bit-mapped display. For example, in one window a user could be viewing a realization of a set-valued random process in the plane, another window could be displaying set-valued statistics for the process as they are computed, still another window could be displaying the process in some appropriate function space, while a fourth window could be used to issue commands to affect the process or statistical computations. ANDREW also provides support for menus and for the easy implementation of graphical input with a three-key mouse. Although all of the set-manipulation routines and most of the graphics routines are written to be independent of a particular graphics environment, much of the utility of the system will come from the features of the Andrew software. In particular, the hardware-independent nature of Andrew will allow use of these programs on different machines. Finally, network support in ANDREW provides access to large amounts of disk storage from an individual workstation, freeing it from the necessity of storing all of the needed files on a local disk.

4. Current Status and Future Plans

The geometrical routines are well-advanced, consisting of about 100 routines that range from simple coordinate transformations (e.g. cartesian to polar) to generating pseudo-random star-shaped sets. Data-structures and conventions are fairly stable and well-defined. Programmming effort continues in extending the functions available and programming more efficient algorithms based on research results in computational geometry (e.g. (Preparata and Shamos, 1985)).

The graphics routines have a core which is used to test the geometrical routines, but are under active development as we experiment with various ways of using the graphical interface facilities provided by ANDREW. In particular, we are planning to implement multiple windows in our programs to allow a user to view the same data set simultaneously in several different ways.

The development of the user interface is still in the planning stage, since we are just beginning to get experience with exploring set-valued data in a workstation-type environment. One option under investigation is to attempt to integrate the routines more closely with an existing extensible package such as S (Becker and Chambers, 1984) to take advantage of its user interface (which already accomodates graphical I/O to some extent).

Finally, we have at hand some samples of set-valued data (such as the sand grain data mentioned above) which we intend to explore from this new perspective using these new tools.

The accompanying figure is a screen dump of the Andrew workstation "pitneyfork" running a prototype of S^3. The screen displays a plot of 50 elements from a set-valued random process of star-shaped sets, plus a plot of the distribution of the number of vertices per set using S. In addition, the text editor EMACS is being used to modify the main routine of S^3 and the window labeled console is monitoring the performance of the workstation.

5. REFERENCES

Artstein, Z., and Vitale, R.A. (1975), "A Strong Law of Large Numbers for Random Compact Sets," *Annals of Probability*, **3**, 879-882.

Becker, R.A., and Chambers, J.M. (1984), *S: An Interactive Environment for Data Analysis and Graphics*, Belmont, California: Wadsworth Advanced Book Program.

Crecine, J.P. (1986), "The Next Generation of Personal Computers," *Science*, **231**, 935-943.

Cressie, N. (1979), "A Central Limit Theorem for Random Sets," *Z. Wahrscheinlichkeitstheorie verw. Gebiete*, **49**, 37-47.

Eddy, William F. (1980), "The Distribution of the Convex Hull of a Gaussian Sample," *Journal of Applied Probability*, **16**, 686-695.

Eddy, W.F. (1982), "Laws of Large Numbers for Intersection and Union of Random Closed Sets," Report 227, Carnegie-Mellon University, Department of Statistics.

Ehrlich,R., et. al. (1980), "The Origin of Shape Frequency Distributions and the Relationship between Size and Shape," *Journal of Sedimentary Petrology*, **50**, June 1980, 475-484.

Jensen, E.B., Baddeley, A.J., *et.al.* (1985), "Recent Trends in Stereology," *International Statistical Review*, **53** (1), April, 99-108.

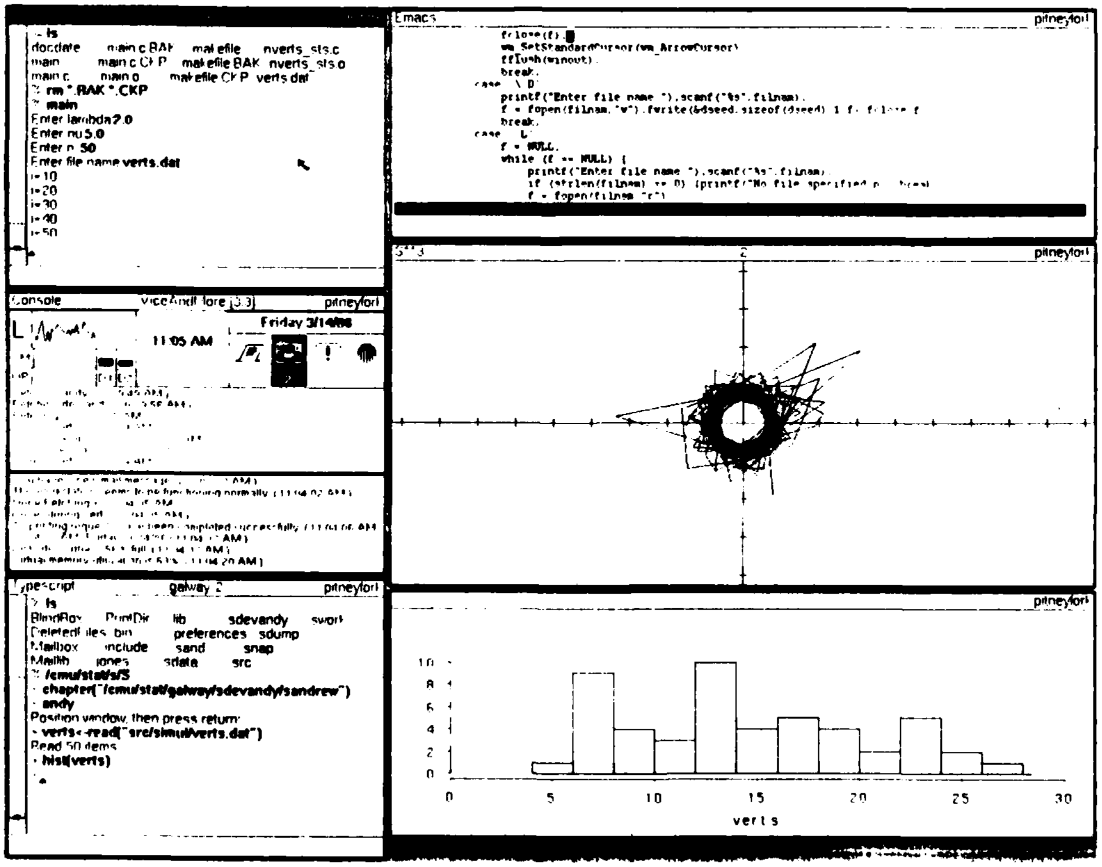
Matheron, G. (1975), *Random Sets and Integral Geometry*, New York: John Wiley & Sons.

Morris, J.H., et. al. (1986), "Andrew: A Distributed Personal Computing Environment," *Communications of the ACM*, **29**, 184-201.

Preparata, F.P. and Shamos, M.I. (1985), *Computational Geometry*, New York: Springer-Verlag.

Trader, D.A. (1981), *Infinitely Divisible Random Sets*, unpublished Ph.D. thesis, Carnegie-Mellon University, Department of Statistics.

Valentine, F.A. (1964), *Convex Sets*, New York: McGraw-Hill.



```
% ls
docdate      main.c.BAK   makefile       nverts_sts.c
main         main.c.CKP   makefile.BAK   nverts_sts.o
main.c       main.o       makefile.CKP   verts.dat
% rm *.BAK *.CKP
% main
Enter lambda 2.0
Enter nu 5.0
Enter n 50
Enter file name verts.dat
i=10
i=20
i=30
i=40
i=50
```

```
Emacs                                                        pitneyfort
            fclose(f);
            wm_SetStandardCursor(wm_ArrowCursor);
            fflush(winout);
            break;
        case '\D':
            printf("Enter file name "),scanf("%s",filnam);
            f = fopen(filnam,"w");fwrite(&dseed,sizeof(dseed),1,f); fclose(f)
            break;
        case 'L':
            f = NULL;
            while (f == NULL) {
                printf("Enter file name "),scanf("%s",filnam);
                if (strlen(filnam) == 0) {printf("No file specified\n"); break}
                f = fopen(filnam,"r");
```

```
Console        viceAndrew fore (3.3)        pitneyfort
11:05 AM                         Friday 3/14/86
```

```
[illegible] normally (11:04:02 AM)
[illegible] completed successfully (11:04:06 AM)
[illegible] memory [illegible] (11:04:20 AM)
```

```
Typescript                  galway 2                pitneyfort
% ls
BlindBox      PrintDir    lib           sdevandy   swork
DeletedFiles  bin         preferences   sdump
Mailbox       include     sand          snap
Maillib       jones       sdata         src
% /cmu/stat/s/S
> chapter("/cmu/stat/galway/sdevandy/sandrew")
> andy
Position window, then press return:
> verts<-read("src/simul/verts.dat")
Read 50 items
> hist(verts)
>
```

Translating Fortran Programs to C
Should You Do It?

David Gray
Statistics Department
Univeristy of Kentucky

ABSTRACT

This paper grew out of my experiences in converting some nonlinear minimization subroutines written in Fortran to C. I wanted to port these routines to a microcomputer on which a C compiler was available but no Fortran compiler. In the course of this I tried to develop some automatic translation tools. I am going to discuss some of the issues involved in this process. Hopefully this will help others make their decision as to whether they wish to do this.

1. Why Convert a Fortran Program to C?

The programming language C is very popular and its popularity seems to be increasing at a rapid rate. This probably stems as much as anything from the thousands of computer science students trained in the UNIX* environment. Many of the large software houses are using C and others are converting to it. Here is a short list of reasons why people are converting their old Fortran programs to C.

1) It's possible that you may only have a C compiler available on your microcomputer.

2) C compilers are typically the first language available on a new machine (along with Basic). This is usually because the the software firm that wrote the operating system software did it in C. For example, the Atari 520ST had a C compiler available from at its introduction but after almost a year on the market, no Fortran compiler is available. A firm that wants to get a jump on its competition or needs to port software to a new machine will have an advantage if its software is written in C.

3) Most large software projects are being written in C. There are compiler families that allow Fortran and C code to be linked together. However, this can be tricky business and if all the code is written in one language things will go smoother.

4) Many new programmers know C from university experience with UNIX. Rather than training them in a new language, there may be an advantage in using them in their 'natural' environment. I have been told stories of freshly minted programmers balking at programming in anything but C. Of course this a management problem, and not a pure programming issue, but it is important.

5) C libraies usually offer interfaces to operating system and hardware services such as graphics, memory management etc.

6) And of course there's the bandwagon effect: everybody else is doing it.

The advantages of C include:

1) Speed. In some cases, it has been claimed that C is only 2 to 3 times slower than programming in assembly language. However, this is clearly dependent on the quality of code generated by a particular C compiler. In the microcomputer world compilers tend to produce flat footed, unimaginative code. Also, as the programs grow larger, it is not clear that the claimed C to assembly language speed

* UNIX is a trademark of AT&T.

ratio is maintained.

2) Portability. Portable programs are possible, but access to operating system resources usually results in non-portable programs. The use of conditional compilation can help but writing portable code is an art that few of us are good at. It is also debatable whether portability is always desirable.

3) Flexibility. It seems to be true that if it can be done, you can do it in C. The richness of data structures may be the most important reason.

The advantages of Fortran include:

1) Speed. Fortran compilers have a general reputation for being fast with respect to execution, especially for numerical work. Again, as with C, in the micro world Fortran compilers may not deserve that reputation.

2) Portability. Fortran is extremely portable if you stay away from compiler specific extensions.

3) Flexibility. This is Fortran's real weakness. Fortran is poor at handling character data, it has limited access to OS resources and very limited data structures.

So it appears the real difference is the flexibility of C over Fortran. I believe most professional programmers will agree that this is one of the chief reasons if not the main reason for using C.

2. What makes the translation from Fortran to C difficult?

If were trivial to convert Fortran programs to C, it would have all been done by now. Unfortunately, it isn't all sweetness and light. There are many problems in going from Fortran to C, some easy and others subtle and difficult. What follows is a by no means complete list of some of the problems involved.

1) Fortran is context sensitive. If you come across a '(' in a Fortran program you don't know whether you're dealing with a function or an array without extra information. It's more work to keep track of the needed information to make these decisions. Another problem is that blanks are not significant in Fortran. It's possible to write totally unreadable Fortran programs. Fortunately, people don't write such programs, at least not intentionally.

2) Fortran is column major and C is row major with respect to arrays. Numerical algorithms tend to take advantage of Fortran's column ordering. If such an algorithm were literally translated into C on a machine virtual memory, substantial performance degradation due to page faults can occur.

3) Not all functions available in Fortran are available in C. This is a fairly straight forward problem to solve but it does entail extra work and testing.

4) Fortran makes no distinctions between parameters and variables. That is, the variable declartions in a subroutine don't give any information whether a value or an address is being passed. For functions and arrays it is clear that they are addresses. This is a difficult problem.

5) Not all C and Fortran data types match up. For example, there are no logical or complex types in C. They can usually be simulated however.

6) C promotes single precision floating point variables to double precision when performing arithmetic operations. This is an important problem for the numerical analyst because in many problems most of the computation is done in single precision and only such things as residuals are kept in double precision. The speed advantage of single precision is lost. Some C compilers optionally do not promote and the ANSI C draft may make this standard.

7) C does not guarantee the order of evalutation of terms in an arithmetic expression. For example,

$$(a + b) + (c + d)$$

could be translated by the compiler to

$$(a + d) + (b + c)$$

If the order of evaluation is important, e.g. to avoid overflow or in a calculation where small elements must be added before large elements to avoid cancellation, accuracy may be lost. Order would have to be forced by using temporary variables to hold intermediate quantities.

8) Arrays in C start with index 0 while in Fortran they start with 1. This seems simple to fix, but with the wild subscripting schemes used in some programs, it can be confusing.

9) In Fortran subroutines, arrays may be given dimensions passed to the subroutine. In C, arrays must have fixed dimension, except for the last dimension. One way around this is to write your own matrix allocation subroutine and call it with the passed dimensions. But then the reference of elements of the allocated array can become messy.

Some of the problems in this list are easily amenable to automatic translation, at least of the kind I mean where you are not going to spend a year writing your translator. Some of them appear to require an in depth understanding of the original Fortran code and will have to be resolved manually.

3. *Writing your own translation tools.*

In attempting to write some tools, I wanted to take advantage of existing tools to make my life easier. Fortunately, I had access to a Vax 11/750 running 4.2BSD UNIX. UNIX contains many tools for character and string manipulation, which much of translation process is. It also has a little known utility, struct, which is of great value. Struct takes a Fortran program and translates it into a Ratfor program. Ratfor is a dialect of Fortran with many C like constructs. After using struct, further processing was done with awk, a pattern matching and string substitution language that can be programmed similarly to C (I have been told that sed, the stream editor, can do some of the things that I was using awk and much more quickly). I also wrote some C programs in those cases where awk wasn't powerful and/or fast enough.

The awk and C programs did such things as:

1) Fix goto labels

2) Change Ratfor switch statements to C switch statments

3) Change do statements to for statements

4) Change parentheses to brackets. This required knowing in advance the function and array names. I used grep for this purpose.

5) Add semicolons, fix up comments, etc.

None of the above touch on the more difficult problems mentioned in the previous section. And with good reason, as I didn't want to spend a year writing my own translator. Also, it is important to note that I've relied on the good coding practices of the people who wrote the Fortran programs. For the type of code I deal with the programs are well written and with care. This makes it easier to write translation tools and certainly makes my simplistic tools work.

A neater and more powerful way to accomplish the above and even more is to use the UNIX utilities yacc and lex. Yacc is a parser generator that has been used in the construction of compilers and other software. Lex is a lexical analyzer. These tools along with struct and awk could be used to build a powerful translator. However, I wouldn't go so far as to try and write a Fortran to C compiler. Fortran must be one of the hardest languages to write a compiler for and you'll easily be spending your year (or more) doing this.

There are commercial programs and services available for Fortran to C source translation. They tend to be expensive and a recent rewiew of one of these programs reported that the resulting C code was a literal translation of the Fortran source code and that the translator program broke on large programs. It appears that the people who do this for a living have problems too.

4. An example of simple translation.

In this example, a Fortran fragment will be translated into C code. The orginal fragment was:

```
      DO 20 I=1,L
        II=L - I + 1
 20     S(I)=R(II,L)
```

After applying struct we get,

```
      do i = 1,l {
            ii=l-i+1
            s(i)=r(ii,l)
            }
```

After running an awk program to convert do statements to for statements, we get

```
for(i=1-1;i<l;i++) {
  ii=l-i+1
  s(i)=r(ii,l)
  }
```

Note that the '1-1' in the for statement is obnoxious, but I am taking advantage of the compiler to fold the term into '0'. Next using tools to change brackets into parentheses and add semicolons, the final version of the fragment is,

```
for(i=1-1;i<l;i++) {
  ii=l-i+1;
  s[i]=r[ii][l];
  }
```

Rather than run these programs individually, they can be put into a shell script and executed as a single command.

5. What Do You Have When You're Done?

Typically, what you have when you're done is a Fortran program written in C. This is especially true when the process is highly automated. None of the unique C constructs will be used. In some cases this is all you can ask for as some programs can really only be written one way, Fortran or C. But you may be disappointed as the resulting code is not very exciting. You get what you pay for.

At worst, you may have an inefficient program that doesn't produce the same results as the original. This too may be a result of highly automated translation or mistakes made in manual translation.

The moral of the story is to TEST the resulting C program. A comment was made at the conference that one reason NOT to translate Fortran programs to C is that the Fortran routines have withstood the test of time. A new C version would also have to undergo the same tests.

To improve performance, accuracy or take advantage of C feature, fine tuning by hand will be necessary. This means

1) You know Fortran.

2) You know C.

3) You understand the program or underlying algorithm.

You might decide it is better to start from scratch in C.

6. Conclusions

After first being very enthusiastic on the idea of automatic translation of Fortran to C, I am now more cautious. You can translate some Fortran programs or subroutines into good quality C programs or functions. What I've learned is that you can't be too greedy. My conclusions come down to the following:

1) It is possible to automate much of the tediousness out of the Fortran to C translation process without too much effort. With a lot of effort you can just about automate the whole process.

2) You should have a good reason to convert a Fortran program to C. Fortran will often out perform C in execution time and Fortran is very portable. And there are compiler families that allow Fortran, C and other languages to be mixed.

3) You cannot treat this as a black box, especially with numerically sensitive routines. The resulting C program may perform unacceptably and you will have to fix it manually. I would never take a large Fortran program, automatically translate it and assume it will perform correctly without testing.

4) The typical size of routine that I translate semi-automatically is 100 to 300 lines. This is a size that allows me to understand what is going on inside the program and also feel confident that I can do the necessary manual labor involved. Fortunately, this is a 'standard' size routine that is found in numerical work.

UPPER AND LOWER PROBABILITY: A GENERAL FRAMEWORK FOR MODELING UNCERTAINTY

Yves L. Grize, AT&T Bell Laboratories, Holmdel, NJ 07733

Abstract

This paper reviews the theory of upper and lower probability, also called interval-valued probability. Interest in this theory has recently been stimulated by the attempts to apply the theory of belief functions to artificial intelligence, especially expert systems designs. Upper and lower probabilities provide an attractive general framework for modeling uncertainty because of their large scope of interpretations. They include not only conventional probabilities, belief functions and envelopes of measures but also new uncertainty functions that cannot be related to conventional probabilities. The problem of the numerical complexity of the theory is discussed. A family of lower probabilities that are easy to use and should be large enough for most practical applications is described. Some limitations of the theory will also be discussed.

I. INTRODUCTION

The need of mathematical models for reasoning under uncertainty has been emphasized in the recent literature in artificial intelligence and related fields (e.g., see IJCAI [1985]). Useful abstract models for reasoning under uncertainty, namely probabilistic reasoning models, should be based on a concept of probability that is supported by a mathematical structure.

In this paper we review the mathematical structure of upper and lower (U/L) probability also called interval-valued probability and describe different uncertainty models that can be constructed from this structure. Because of their large scope of interpretations U/L probabilities provide a general framework for modeling uncertainty. They include not only conventional probabilities, belief functions and envelopes of measures but also new uncertainty functions that cannot be related to conventional probabilities.

Interest in U/L probabilities has recently been stimulated by the attempts to apply the theory of belief functions to the design of expert systems.

II. THE THEORY OF UPPER AND LOWER PROBABILITY

II.1 The Basic Theory:

Throughout the paper Ω denotes a finite set. Two numbers, $\underline{P}(A)$ and $\overline{P}(A)$, called respectively the lower and upper probability of A, are assigned to each subset A of Ω. These two set functions must satisfy the following axioms (e.g., see Good [1962]):

- Axiom 1 (Normalization): $\underline{P}(\Omega) = 1$
- Axiom 2 (Nonnegativity): $(\forall A)\ \underline{P}(A) \geq 0$
- Axiom 3 (Conjugacy): $(\forall A)\ \underline{P}(A) + \overline{P}(\overline{A}) = 1$ ($\overline{A}$ denotes the complement in Ω of A)
- Axiom 4 (Sub- and Superadditivity):
  $(\forall A,B\ A \cap B = \emptyset)$
  $\overline{P}(A) + \overline{P}(B) \geq \overline{P}(A \cup B)$ (sub)
  $\underline{P}(A) + \underline{P}(B) \leq \underline{P}(A \cup B)$ (super)

Elementary consequences of these axioms include the following:

1. $(\forall A)\ \underline{P}(\emptyset) = \overline{P}(\emptyset) = 0 \leq \underline{P}(A) \leq \overline{P}(A) \leq 1 = \underline{P}(\Omega) = \overline{P}(\Omega)$
2. $(\forall A,B)\ A \subset B \Rightarrow \underline{P}(A) \leq \underline{P}(B)$ and $\overline{P}(A) \leq \overline{P}(B)$

Observe that if $\underline{P} = \overline{P} = P$ then P is a (finitely additive) probability measure.

Because of Axiom 3 one set function on 2^{Ω} completely determines the other. Therefore, without loss of generality, the entire theory can be expressed in terms of only one of them. From now on our discussion will be phrased in terms of the lower probability only, as it is customary in the literature.

A lower probability can be defined independently of its associated upper probability as follows:
A lower probability on 2^{Ω} is a normalized, non-negative set function such that $\underline{P}(\emptyset)=0$ and $\underline{P}(A) + \underline{P}(B) \leq \underline{P}(A \cap B) + \underline{P}(A \cup B)$ for all pairs (A,B) of sets such that $A \cap B = \emptyset$ or $A \cup B = \Omega$.

II.2 Motivation for the Axioms:

A natural motivation for the axioms 1-4 comes from the behavior of the relative frequency f_n of occurrence of an event E in a sequence of n independent repetitions of an experiment (Walley & Fine [1982]). Since $\{f_n\}$ is a bounded sequence of real numbers it always has an inferior and a superior limit. One then defines $\underline{P}(E) = \liminf f_n$ and $\bar{P}(E) = \limsup f_n$. The assumption about the convergence of the relative frequency to the probability, needed to justify the axioms of conventional probability, is no longer necessary.

II.3 A Simple Example:

To illustrate how a lower probability could be used to model uncertainty let us consider the following simplified situation of a medical diagnosis: Let $\Omega = \{a,b,c\}$ where a, b and c are three symptoms: a: to have a runny nose, b: to have irritated eyes, c: to sneeze once in a while. Further, suppose that an expert tells us that:
- one symptom alone does not indicate an allergy,
- the three symptoms together surely indicate an allergy,
- two symptoms indicate a middle state of indecision about the presence of an allergy (i.e. in common language there is a 50% "chance" of an allergy).
How can we model the uncertainty in an allergy diagnosis based on an observation A in Ω?

A natural answer is the set function $\underline{P}$ defined by:

$$\underline{P}(\Omega) = 1,$$
$$\underline{P}(\{a,b\}) = \underline{P}(\{b,c\}) = \underline{P}(\{a,c\}) = \frac{1}{2},$$
$$\underline{P}(\{a\}) = \underline{P}(\{b\}) = \underline{P}(\{c\}) = \underline{P}(\varnothing) = 0.$$

Observe that $\underline{P}$ is *not* a probability (in fact it is not even a belief function, as defined in III.1.f). However, it is easy to see that $\underline{P}$ is a lower probability.

III. UNCERTAINTY MODELS BASED UPON LOWER PROBABILITIES:

The mathematical structure of U/L probabilities can be used as a basis for various uncertainty models each one corresponding to a different type of lower probability. We first define these different types of lower probabilities and then discuss the corresponding models and mention some of their applications.

III.1 Classification of Lower Probabilities:

Definition: Let $\underline{P}$ be a lower probability on 2^{Ω}.

a. A probability measure μ *dominates* $\underline{P}$ if $(\forall A)$ $\mu(A) \geq \underline{P}(A)$. It follows that $\mu(A) \leq \bar{P}(A)$.

b. The class M_P of all probabilities that dominate $\underline{P}$ is called the *class of dominating probabilities* of $\underline{P}$.

c. If M_P is empty, $\underline{P}$ is called *undominated*, otherwise it is *dominated*.

d. $\underline{P}$ is a *lower envelope* if
$$(\forall A)\ \underline{P}(A) = \inf\{\mu(A) : \mu \in M_P\}.$$

e. A lower probability $\underline{P}$ is *monotone of order k* if: $(\forall A_1, \ldots, A_k)$
$$\underline{P}(\bigcup_{j=1}^{k} A_j) \geq \sum_{\varnothing \neq I \in \{1,2,..k\}} (-1)^{|I|+1} \underline{P}(\bigcap_{i \in I} A_i).$$

In particular $\underline{P}$ is monotone of order 2 (or 2-monotone) if:
$$(\forall A,B)\ \underline{P}(A) + \underline{P}(B) \leq \underline{P}(A \cup B) + \underline{P}(A \cap B).$$

f. $\underline{P}$ is a *belief function* if it is monotone of order k for all k. Every set function $\underline{P}$ on 2^{Ω} can be written as $(\forall A)$ $\underline{P}(A) = \sum_{B \subset A} m(B)$ for some function m. It turns out that $\underline{P}$ is a belief function if and only if m is non-negative with $m(\varnothing) = 0$ and $\sum_{B \subset \Omega} m(B) = 1$. m is called *the basic probability assignment* of the belief function $\underline{P}$.

Denote by *P, B, M_2, LE, D, U* and *LP* the classes of probabilities, belief functions, 2-monotone lower probabilities, lower envelopes, dominated lower probabilities, undominated lower probabilities and lower probabilities on 2^{Ω}. It has been shown (e.g., see Walley & Fine [1982]) that:

$$P \subset B \subset M_2 \subset LE \subset D \subset D \cup U = LP$$

and that if $|\Omega| \geq 7$ all these inclusions are strict.

We now discuss the uncertainty models resulting from this hierarchy of lower probabilities.

III.2 Probability-Based Models (P-Models):

We have already pointed out that probabilities are a special case of lower probabilities. P-models are well-known and will not be further discussed.

III.3 Belief Function-Based Models (B-Models):

B-models are the lower probability based models that have received the most attention. They arose from the work of Dempster [1967] on multivalued mapping and were later extended by Shafer [1976] (see also Shafer [1982a]).

Belief functions are interpreted through their basic probability assignment function m: in light of a piece of evidence, m(A) is that portion of a person's total belief (of value 1) exactly committed to A and to none of the proper subsets of A ("intrinsic" belief). A reason for the popularity of B-models is that they can be combined together through their m-functions as different pieces of evidence are collected using the so-called "Dempster's rule of combination" (see Shafer [1976]). B-models have been used in a variety of fields such as psychology (e.g., Krantz and Miyamoto [1983]), statistics (e.g., Shafer [1982b]), computer vision, risk assessment and medical expert systems (e.g., Gordon & Shortliffe [1984]).

III.4 Lower Envelope-Based Models (LE-Models):

LE-models arise whenever the uncertainty is conveniently described by a class of probability measures M. Indeed any such M induces a lower envelope by $\underline{P}(A) = \inf \{\mu(A) : \mu \epsilon M\}$. In general the set M_P of dominating measures will be larger than the class M that induces the lower envelope. Some applications of LE-models are mentioned:

1. In the theory of robust statistics LE-models have been used to describe neighborhoods of probability distributions (Huber [1981]).
2. When expert opinions are represented by probabilities, lower envelopes provide a simple way to aggregate these opinions into one set function (Walley [1982]). Thorp et al. [1982] give an example of a LE-model used to forecast production cost in an electric utility.
3. Finally a personalistic account of uncertainty, based on a notion of "coherency", has been developed using lower envelopes, in a similar way as it is done using probabilities (see Walley [1981] and the references therein). This approach allows to model the inherent imprecision in a person's beliefs.

III.5 Dominated Lower Probability-Based Models (D-models):

All the models discussed so far are based on dominated lower probabilities but the class of D-models itself has not yet been studied specifically. It is difficult to interpret dominated lower probabilities that are not lower envelopes, except as a vague description of an underlying probability measure μ such that $\underline{P} \leq \mu \leq \tilde{P}$.

III.6 Undominated Lower Probability-Based Models (U-models):

Undominated lower probabilities provide a completely new framework for modeling uncertainty since they cannot be related to usual probability measures. Grize and Fine [1986/7] have shown that U-models can be constructed to describe stationary processes with bounded and divergent (i.e. fluctuating) time averages, while the modeling of such processes is impossible in standard probability theory (contradiction with the ergodic theorems). The fact that processes with the above-mentioned properties seem to exist, as data on the frequency fluctuations of quartz crystal oscillators show (see Grize [1984] for details), strongly motivates the study of U-models.

The use of U-models remains so far conceptual. The interpretation of undominated lower probabilities in terms of observable data is still an open problem.

IV. THE COMPLEXITY OF THE THEORY - PRACTICAL CONSIDERATIONS:

Let $|\Omega| = n$. To define a probability μ on Ω it suffices to specify the values of μ on the n atoms of Ω, but for a lower probability $\underline{P}$ the values of $\underline{P}$ must be given for each of the 2^n subsets of Ω.

The classes P and LP are closed and bounded convex polyhedrons in the 2^n dimensional space R^{2^n}. P has n extreme points. An idea of the complexity or richness of the structure of lower probabilities is gained by examining the extreme points of LP. The number of extreme lower probabilities grows very rapidly with n and already exceeds 10 million when $n=10$ (see Grize [1984] for details). If $n=3$, there are 8 extreme lower probabilities: 7 belief functions and the lower probability of paragraph II.3.

To have a useful theory, a way must be found to avoid having to define $\underline{P}$ for every set. A large class of lower probabilities that include all the types discussed above and that is easy to use has been proposed in Grize [1984]. Such lower probabilities are defined by way of a family G of sets with the property that any collection of 2m-2 elements of G has a non-empty intersection, where m is a given integer greater than 1. For a set A, $\underline{P}(A)$ is determined by the smallest number of sets in G whose intersection lies in A. More precisely:

$$(\forall A \subset \Omega) \;\; \underline{P}(A) =$$

$$\begin{cases} 1 \text{ if } A = \Omega \\ 1 - \dfrac{1}{m} \text{ if } A \neq \Omega \text{ and } (\exists B \epsilon G_1)\, B \subset A \\ 1 - \dfrac{2}{m} \text{ if } (\forall B \epsilon G_1)\, B \not\subset A \text{ and } (\exists B \epsilon G_2)\, B \subset A \\ \quad \dots \\ \dfrac{1}{m} \text{ if } (\forall B \epsilon G_{m-2})\, B \not\subset A \text{ and } (\exists B \epsilon G_{m-1})\, B \subset A \\ 0 \text{ otherwise} \end{cases}$$

where: $G_1 = G$ and

$$G_k = \cap \, (G)_i$$
$$= \{B \subset \Omega : (\exists B_1, B_2, .., B_k \epsilon G)\; B = B_1 \cap B_2 \cap \dots \cap B_k\}.$$

It is easy to check that $\underline{P}$ is a lower probability. This class of lower probabilities should be large enough for most practical applications. The lower probability of section II.3 is an example of a lower probability defined in this fashion with $G = \{ \{a,b\}, \{b,c\}, \{a,c\} \}$ and $m = 2$.

V. CONDITIONAL LOWER PROBABILITIES:

A satisfactory answer to the question of defining conditional lower probabilities is yet to be found. It is beyond the scope of this paper to present a full discussion of this issue and we shall limit ourselves to briefly mention some of the various forms of conditioning that have been proposed so far (see Walley ([1981] for more details):

- For probabilities define:

$$P(A|B) = \frac{P(A \cap B)}{P(B)}.$$

- For belief functions the conditioning is expressed in terms of the upper probability (Shafer [1976]):

$$\overline{P}(A|B) = \frac{\overline{P}(A \cap B)}{\overline{P}(B)}$$

or:

$$\underline{P}(A|B) = \frac{\underline{P}(A \cup \bar{B}) - \underline{P}(\bar{B})}{1 - \underline{P}(\bar{B})}.$$

- For a lower envelope $\underline{P}$ given by $\underline{P}(A) = \{ \inf \mu(A) : \mu \epsilon M_P\}$ define (e.g., Walley[1981]):

$$\underline{P}(A|B) = \{ \inf \mu(A|B) : \mu \epsilon M_P\}.$$

If $\underline{P}$ is 2-monotone $\underline{P}(A|B)$ can be written as:

$$\underline{P}(A|B) = \frac{\underline{P}(A \cap B)}{\underline{P}(A \cap B) + \overline{P}(B \cap \bar{A})}.$$

This lack of consensus on how to define conditional lower probabilities is disturbing. There is little doubt that this question needs to be solved for the theory to be successful in areas such as expert systems or artificial intelligence. In our opinion the issue of conditioning is the major limitation of the theory of lower probability today.

VI. CONCLUSION:

- The mathematical structure of upper and lower probabilities unifies various uncertainty models and is an elegant generalization of the classical theory of probability,
- Upper and lower probabilities provide a general framework for modeling uncertainty where the uncertainty described by conventional probability is only a degenerate case ($\underline{P} = \overline{P}$),
- Upper and lower probabilities have solid mathematical foundations hence are adequate for rigorous developments,
- Although upper and lower probabilities are, in general, difficult to specify numerically, a class that is easy to use and large enough for most practical purposes has been identified.

However:

- An intuitive interpretation of upper and lower probabilities that are not envelopes of probabilities is still missing (especially for undominated lower probabilities),
- The methods for conditioning upper and lower probabilities are still unsatisfactory (What is the right way to do it?)

REFERENCES

□ Good, I. J. [1962], "Subjective Probability as the Measure of a Non-Measurable Set", in E. Nagel, P.Suppes, A. Tarski eds., *Logic, Methodology and Philosophy of Sciences*, Stanford University Press.

□ Gorgon, J. and Shortliffe, E. H. [1984], "The Dempster-Shafer Theory of Evidence", in B. G. Buchanan and E. H. Shortliffe eds., *Rule-Based Expert Systems*, Addison-Wesley.

□ Grize, Y. L. [1984], *Towards a Stationary Continuous Lower Probability-Based Model For Flicker Noise*, Ph.D. Thesis, Cornell Univ., Ithaca, NY.

□ Grize, Y. L. & Fine, T. L. [1986/7], "Continuous

Lower Probability Based Models for Stationary Processes with Bounded and Divergent Time Averages", *Annals of Probability* (to appear).

□ Huber, P. [1981], *Robust Statistics*, Wiley.

□ IJCAI [1985], *Proceedings of the Ninth International Joint Conf. on Artificial Intelligence*, 18-23 August, Los Angeles, CA.

□ Krantz, D. H. and Miyamoto, J. [1983], "Priors and Likelihood Ratios as Evidence", *J. Am. Stat. Assoc.***78**, pp. 418-423.

□ Shafer, G. [1976], *A Mathematical Theory of Evidence*, Princeton Univ. Press.

□ Shafer, G. [1982a], "Belief Functions and Parametric Models", *J. Royal Stat. Soc. Ser. B* **44**, pp. 1-25.

□ Shafer, G. [1982b], "Lindley's Paradox", *J. Am. Stat. Assoc.* **77**, pp. 325-351.

□ Thorp, J. S., McClure, J. and Fine, T. L. [1982], "The Use of Expert Opinion in Forecasting Production Cost for an Electric Utility", *Proc. of the 1982 IEEE Inter. Large Scale Systems Symp.*, Virginia Beach, VA, pp. 126-131.

□ Walley, P. [1981], *Coherent Lower (and Upper) Probabilities*, Techn. Report, Dept. Statistics, Univ. of Warwick, Coventry, England.

□ Walley, P. [1982], *The Elicitation and Aggregation of Beliefs*, Techn. Report, Dept. Statistics, Univ. of Warwick, Coventry, England.

□ Walley, P. & Fine, T. L. [1982], "Towards a Frequentist Theory of Upper and Lower Probabilities", *Annals of Statistics* **10**, pp. 741-761.

THE NUMERICAL SOLUTION OF A SYSTEM OF ORDINARY STOCHASTIC DIFFERENTIAL EQUATIONS ON THE CYBER 205 SUPERCOMPUTER

Tim Haas, Colorado State University

Summary

A numerical method using an Ornstein-Uhlenbeck approximation to the Wiener Process coupled to a Runge-Kutta algorithm has been programmed on the Cyber 205 supercomputer to solve systems of ordinary stochastic differential equations (OSDE'S). Each equation may be any programmable function of the independent and dependent variables optionally multiplied by a stochastic process and/or with an additive stochastic process. Equations with analytical solutions are solved and errors presented. Cyber 205 timings are presented along with remarks concerning the vectorization of the code.

I. Introduction

Although the theory of stochastic differential equations (SDE'S) has received much attention [1, 2], the numerical solution of these equations having no known analytical solution has not been subject to similar effort. One reason may be the expensive computation that solutions of these equations require.

Many models in the physical, life and social sciences may be beneficially recast in an SDE form. This form may, however, be analytically intractable, thus a numerical approach is the only recourse. The program SDESS (Stochastic Differential Equation System Solver) was developed as an attempt to answer this need.

Specifically a program written by M. Elrod at the University of Georgia on a CDC 7600 [3] to solve a single, particular SDE was modified to solve a general system and adapted to the Cyber 205.

II. Equation Form

The type of equation system solvable by SDESS is of the form

$$\frac{dy_i}{dt} = \xi_i(t)F_i(\bar{y},t) + \psi_i(t)$$

where,

y_i is the i th dependent variable

t is the independent variable

ξ_i, ψ_i are stochastic processes which may be nonstationary.

F_i is any FORTRAN programmable function of the *vector of* dependent variables and the independent variable.

Since a fourth order Runge Kutta method with no variable step size capability is used, the SDE's should not be "stiff" although this may be difficult to predetermine without knowing the effect of the stochastic processes on the solution.

III. Input for SDESS

Input for SDESS consists of two types; source code modification and input file creation.

The first type consists of programming the statement functions for the covariance matrices for the two stochastic processes, and the statement functions for the equation functions. This may seem unnecessarily cumbersome, but is crucial for efficient use of the 205 as described below.

The second type of input consists of the input file which contains a) the number of realizations of the solution desired, b) number of independent variable steps before the first four moments of $\bar{y}$ are computed, c) the number of covariances (number of values of the independent variable equally spaced at which covariances are to be computed), d) the sizes of the covariance matrices for the stochastic processes.

SDESS is written as a subroutine so that the input file is passed to SDESS through a "CALL" statement.

IV. Output From SDESS

SDESS will output the following at the requested time points, $t_1, t_2, \ldots, t_{final}$: a) the mean value of y_i, b) the covariance matrix of y_i $(cov(y_i(t_j), y_i(t_k)))$, c) the mean value for each of the stochastic processes ξ_i and ψ_i, d) the covariance matrix for each stochastic process and e) the mean square error of the estimated mean and covariance of each stochastic process.

V. Verification and Test Runs.

Two different equations have been run using SDESS. The first is the equation (equation A) used by Elrod in his dissertation to verify the program. In SDESS a two equation system was created by coding this equation twice. The second equation (equation B) was selected because of its oscillating and analytical solution. Here also, the same equation was coded twice to create a two equation system.

The equation used by Elrod is:

$$\frac{dy}{dt} = \xi(t)y + \psi(t)$$

where

$$\langle\xi(t)\rangle = .5$$
$$\langle\psi(t)\rangle = .5+\sin(2\pi t)$$
$$y(o) \text{ is } N(1,1/3)$$

The covariance matrices of $\xi(t)$ and $\psi(t)$ are given by,

$$cov(t,t') = \exp(-|t-t'|)$$

Elrod ran the above using 10^6 realizations. During initial development of the code on a VAX-11/750 with a floating point accelerator, running in core this solution was essentially duplicated with a double precision version of the code. The run time was about 160 hours at about 95% cpu usage. In order to save computation expense the two equation system of the above was run on the 205 using only 10^5 realizations. The results were within expected accuracy, see figs. 1 and 2 and Table 1 for solution and error values from Elrod's, the VAX and 205 runs.

The second equation was chosen to verify that SDESS would not always allow a solution to "blow up", a behavior suspected due to the potentially imperfect simulation of a white noise process inherent to the program. Also, an analytical solution had to exist. Thus, for simplicity,

$$\frac{dy}{dt} = -\sin(t) + \sigma \frac{dw(t)}{dt}, \quad y(o) = 1, \ \sigma = 1,4$$

was chosen. The solution is given in Gihman and Skorohod [1] as,

$$y = \text{cost} + \int_0^{t_2} \sigma^2 dW(S), \text{ where W is a Wiener Process}$$

$$\langle y \rangle = \text{cost}$$

$$\text{Var }(y) = \sigma^2 t_{\text{final}}$$

Here, the initial value of y was chosen to be nonrandom to simplify the variance calculation.

As can be seen in fig. 3, the numerical solution of the mean value is quite close to the analytical - this using a step size of .01 and 50,000 realizations.

In order to test the validity of the white noise simulation, the equation was run first with $\sigma = 1$ and then with $\sigma = 4$.

The Ornstein-Uhlenbeck process used in SDESS converges to a white noise process (dW(t)) as a,b → ∞, (a/b) → 1/2 where the covariance function of the Ornstein-Uhlenbeck process is given by:

(1) $\text{cov}(t,t') = ae^{b|t-t'|}$ ([6], pg. 55).

Specifically, SDESS uses (1) to calculate adjustment vectors which modify the Gaussian random variables generated each realization. To avoid underflow on the computer, b was fixed at 1000 and a was calibrated using the $\sigma = 1$ equation. Since a linear relationship was found to exist between a and the Var (Y) at t_{final}, interpolation was arbitrarily stopped when the calculated Var(Y) was close to the analytical. To check the validity of the simulation, σ was then set to 4 and run, the Var (Y) calculation was again quite close (see Table 2). Ideally, analytically determined values of a and b could be given which would yield accurate results (such as $a = 10^5$, $b = 2 \times 10^5$) However, it is suspected the time step would have to be quite small to make use of this finer approximation - the run times then would probably be impractical. The calibration method although not a general result, does appear to allow accurate simulation of stochastic processes.

The 205 cpu timings for each equation system are given in Table 3.

VI. Remarks on Vectorization of the Code.

The Cyber 205, although a fast sequential machine, achieves most of its speed via vectorization of code segments (usually "DO" loops) [4]. Thus to take advantage of this capability the DO loops in the source code must have certain characteristics which will allow the computer to vectorize. Unfortunately, the Runge-Kutta algorithm wherein SDESS spends about 90% of its run time, is by nature a recursive process which is not vectorizable. Surprisingly, the seemingly time consuming collection of "IF" statements required to compute a value of each stochastic process each time step does not require a relatively large amount of time. Thus, in order to increase the speed as much as possible, all subroutine and function calls within the Runge-Kutta "DO" loops were replaced by in-line code. Also, the random number generation using the function "URAND" [5] in early code development was replaced by calls to the 205 random number generator, "RANF" [4]. These two modifications resulted in the modified SDESS running about 3 times faster over the unmodified code on the 205. However, SDESS on the 205 is only about 30 times faster than the VAX-750 implementation - low efficiency for a vector machine.

If a parametric study of a particular system of SDE's were desired (as in nonlinear regression), it should be mentioned that saving the values of the stochastic processes each realization and using these on subsequent iterations would result in a speed increase. However, judging from experience with "turning off" one stochastic process the speed increase will probably be modest.

VII. Conclusions

A solution of a general system of OSDE's appears to be feasible using somewhere in the neighborhood of 10^5 realizations. However, even using a supercomputer, the computation is still quite expensive. In general, run-times would be decreased if new results could produce algorithms which converge with fewer realizations and particular to vector processing computers if a nonrecursive numerical method could be used to solve the SDE.

Incorporating this solver into a nonlinear regression routine using absolute differences is planned.

VIII. Acknowledgements

This work has been supported by the Institute for Computational Studies at Colorado State University.

IX. References

[1] Gihman, T.T. and Skorohod, A.V., Stochastic Differential Equations, New York, Springer-Verlag, 1972.

[2] Ikeda, N. and Watanabe, S., Stochastic Differential Equations and Diffusion Processes, Amsterdam, North-Holland, 1981.

[3] Elrod, M., Numerical Methods for the Solution of Stochastic Differential Equations, Ph.D dissertation, University of Georgia (Athens), Ann Arbor, Michigan, University Microfilms, 1973.

[4] Fortran 200 Version 1, Reference Manual, Control Data Corporation, 1985.

[5] Forsythe, G. E., Malcolm, M.A. and Moler, C.B., Computer Methods for Mathematical Computations, Englewood Cliffs, New Jersey, Prentice-Hall, 1977.

[6] Arnold, L., Stochastic Differential Equations: Theory and Applications, New York, Wiley, 1974.

	<Y>	VAR Y	% error <Y>	% error VAR Y
Elrod	3.035	123.33	21.	-.5
VAX	3.014	118.5	20.	-4.4
205	3.044	156.2	21.	26.
Exact	2.507	124.0	0	0

Table 1. Percent error from exact solution at t_{final} (equation A).

σ	Numerical	Analytical	% error
1	2.483	2.5	.68
4	39.72	40.	.70

Table 2*. Percent error from analytical for Var Y at t_{final} for equation B (-sint).

Equation	Time (seconds)	Cost
1 (Elrod's) 10^5 realizations	1884.77	$1889.
2 (-sint) 50,000 realizations	711.30	$715.

*Based on running equation B with 10^5 realizations 3 separate times, a rough standard deviation for the <Y> at t_{final} was found to .0107. A standard deviation for the Var Y at t_{final} was likewise found to be .0136. These will decrease in proportion to the square root of the number of realizations. Similar values would be found for Equation A.

Table 3. 205 central processing unit time and cost for each solution.

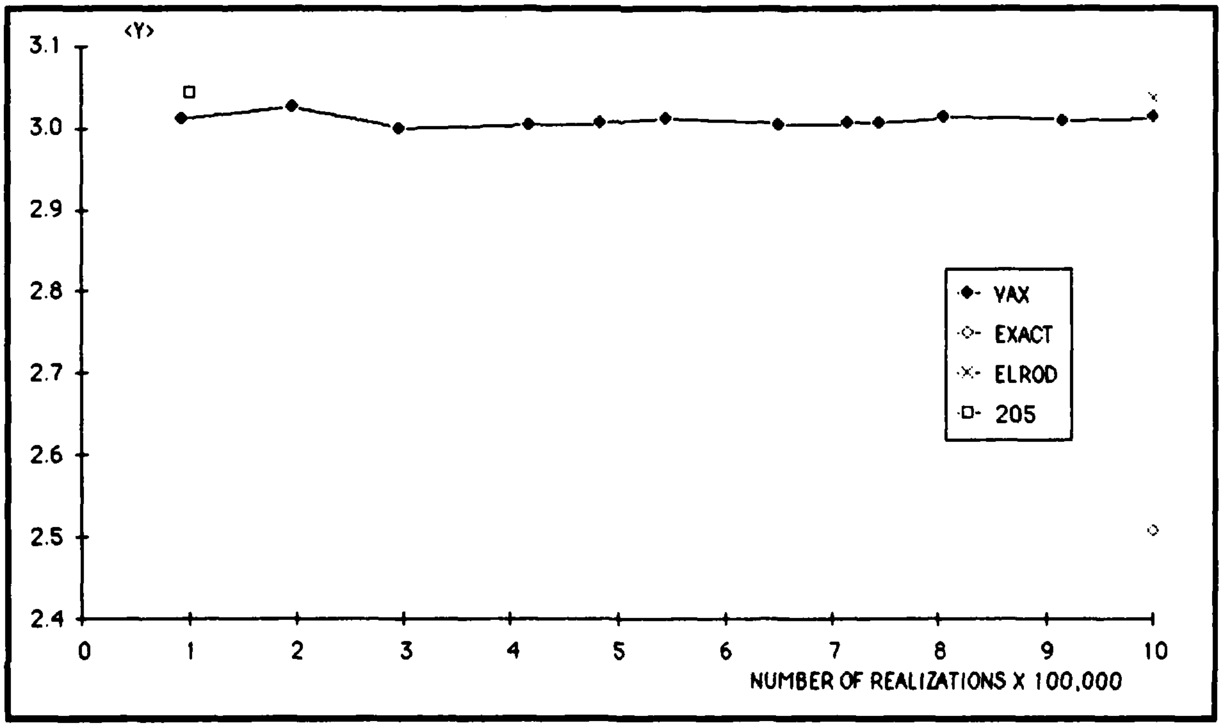


FIGURE 1: <Y> OF EQUATION A AT T = 2.5

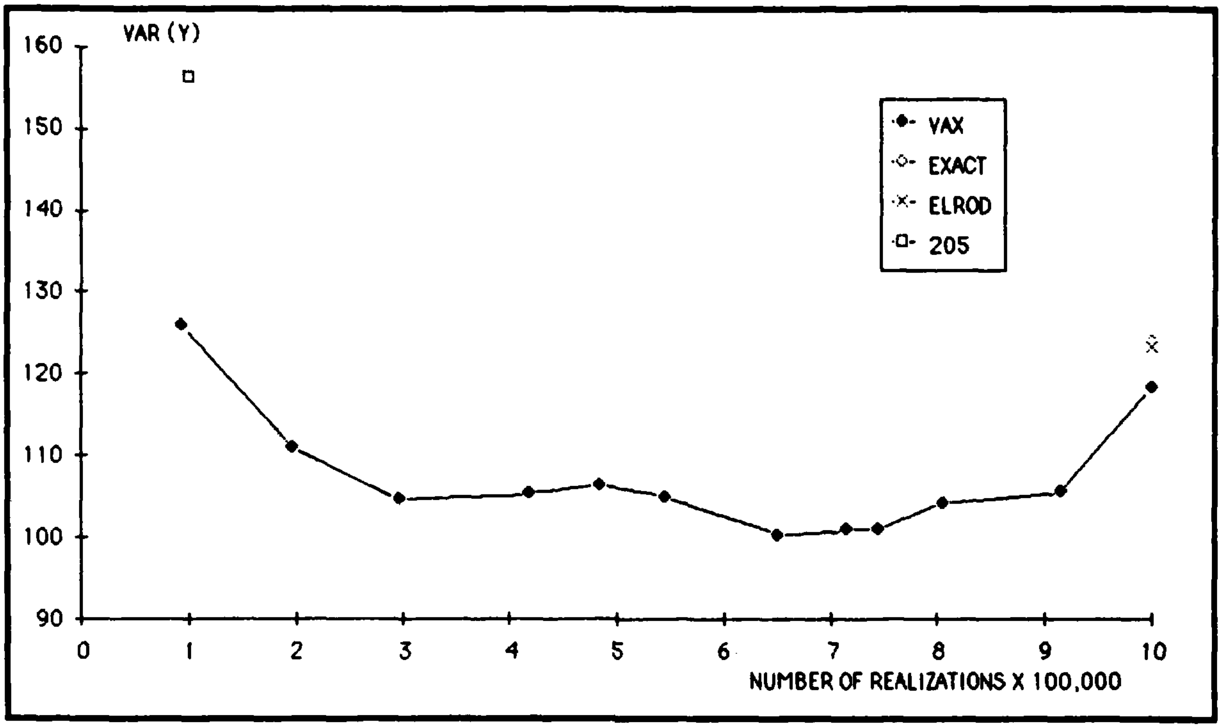


FIGURE 2: VAR (Y), T = 2.5 (EQUATION A)

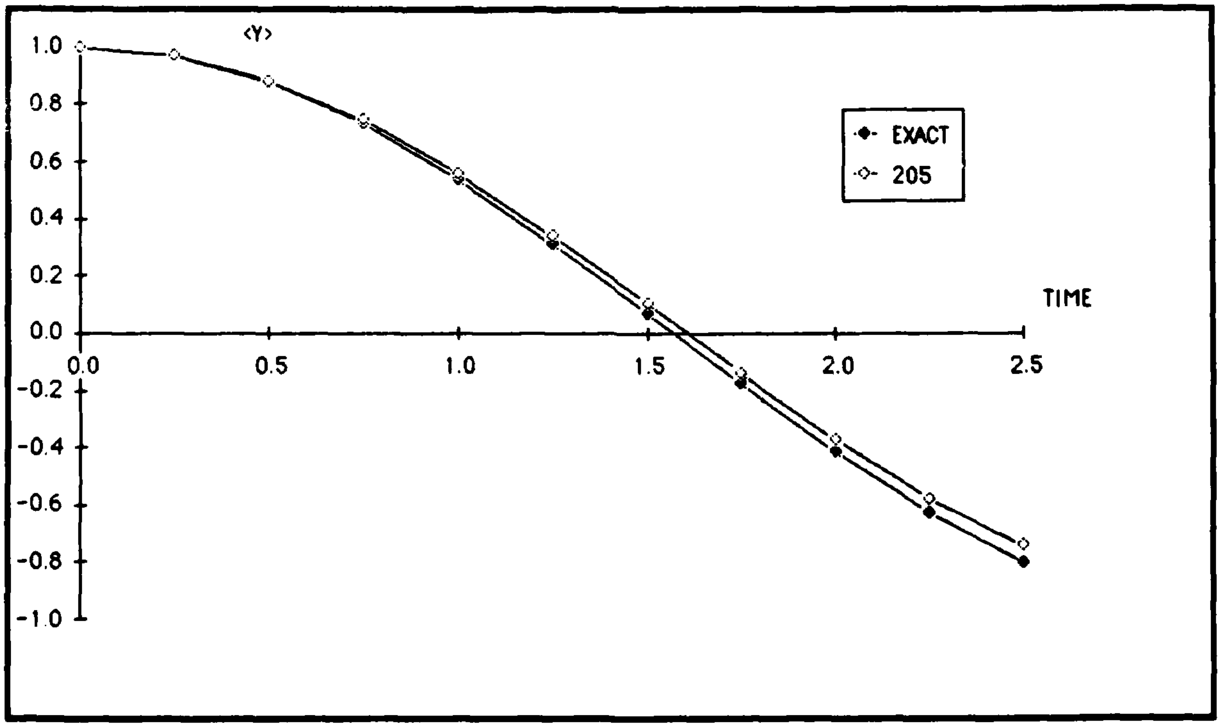


FIGURE 3: < Y > OF EQUATION B (-SIN T)

SOLVING NONLINEAR ECONOMETRIC MODELS USING VECTOR PROCESSORS

Patrick J. Hénaff, Massachusetts Institute of Technology
Alfred L. Norman, University of Texas

1. INTRODUCTION

This paper reports on the design and implementation of a reduced Newton algorithm for solving large non-linear econometric models on a vector processor such as the CYBER 205 or a CRAY X/MP machine. To take full advantage of the vector processing capabilities, one needs to organize the computations such that operations are carried on vectors. For the price of a small start-up time, which varies with machines, a vector processor performs a floating point operation on a component of a vector in a fraction of the time needed to perform the same operation on an isolated scalar. Up to now, numerical methods have been optimized for scalar processors; this involved:

- minimizing the storage requirement. For Newton's algorithm, this was achieved by developing software for operating on matrices stored in sparse format.
- minimizing the number of floating point operations by, for example, computing a row and column permutation that reduces the amount of fill-in during matrix factorization.

Vector processors, however, suggest a different set of criteria for optimizing the implementation of numerical methods:

- Because of dramatic reduction in cost and progress in miniaturization of RAM components, the memory available on recent computers, especially on the vector processor, is very large. As a result, core requirement minimization is not as strong an imperative as it used to be.
- computation should be organized as vector operations as much as possible, even if this leads to some redundant computation.

Let us now turn to econometric models. To solve such models by Newton's method on a vector processor, one approach has been to rewrite sparse matrix code to take advantage of vector processing. However, that approach may be limited by the fact that sparse matrix techniques were designed according to criteria not entirely relevent to vector processing. In contrast, our approach is to restructure the problem at hand so that computations can be naturally expressed as vector operations, and sparse matrix storage schemes avoided altogether. The rest of the paper is organized as follows: Section 2 presents the solution algorithm and how it leads to vector processing. Several technical features of the method are discussed in Section 3, and the coding is considered in Section 4. Timing of vector versus scalar processing solutions for a medium scale econometric model follows.

2. THE SOLUTION ALGORITHM AND HOW IT CAN BE VECTORIZED

2.1 The solution method

We consider the set of simultaneous equations

$$f(z) = 0 \qquad [1]$$

where f is a n-component function and z a vector of n endogenous variables. Predetermined variables are omitted for clarity. At a given point z_k, the residual of the system is $f(z_k) = d_k$. The system's Jacobian is $J_k = [\partial f/\partial z]$.

A reduced system equivalent to [1] is derived by using some equations to eliminate endogenous variables, thus reducing the dimension of the system. First, a set of loop variables is identified in the system. A set of loop variables is such that if the variables in this set were predetermined, then the other variables in the system could be computed recursively. After an appropriate permutation of variables and equations, the system of equations is partitioned into 2 blocks, called the core and the loop block:

$$\begin{aligned} g(x,y) &= 0 \\ h(x,y) &= 0 \end{aligned} \qquad [2]$$

where g and h are vector-valued functions with respectively (n-s) and s components, y a vector of s loop variables and x a vector of n-s core variables. Finally, at a point (x_k, y_k), let the equation residuals be similarly partitioned (b_k, c_k). For each equation in the core we define the function f'_i, which is the original function f_i, solved for variable x_i:

$$x_i = f'_i(x_1,\dots,x_{i-1},y) \quad i = 1,\dots,n-s \qquad [3]$$

The error function is defined as:

$$\Phi(y) = h(x(y),y)$$

with x(y) defined by the core equations [3]. The original system is equivalent to:

$$\Phi(y) = 0 \qquad [4]$$

which is called the reduced problem. Its Jacobian is $T^* = [\partial\Phi/\partial y]$. Newton's algorithm applied to problem [1] will be referred to as the Global Newton's algorithm (GN), while the same method applied to problem [4] will be the Reduced Newton's algorithm (RN). One iteration of RN involved 4 steps:

1. Evaluate the error function at the current point, y_k:

$$\Phi(y_k) = d_k \qquad [5]$$

2. Compute T^*_k at y_k.

3. Solve Newton's equation for p_k:

$$T^* \, p_k = - d_k \qquad [6]$$

4. Convergence test: Stop or go to step 1 with

$$y_{k+1} = y_k + p_k$$

Corresponding to the partitioning of variables, the Jacobian of the original system can be similarly partitioned as:

$$J = \begin{vmatrix} G & R \\ S & T \end{vmatrix} \qquad [7]$$

The Jacobian of the reduced system is then:

$$T^* = T - S\,G^{-1}R \qquad [8]$$

A finite difference approximation to T^* is easily computed because of the recursive structure of the core equations. The jth column of T^* is obtained by:

$$\frac{\partial \Phi}{\partial y_j} \sim \frac{1}{h_j} \left| \Phi(y + e_j h_j) - \Phi(y) \right| \qquad [9]$$

where e_j is the jth unit vector and h_j a small scalar. The determination of h_j is discussed in the next section. The Jacobian of the reduced problem is usually dense and of small size, so that the solution of Newton's equation for the reduced problem can be easily obtained. This solution method has been described in (1).

2.2 How it can be vectorized

Vectorization is achieved at two steps of the algorithm. First, in evaluating the Jacobian of the reduced model by [9] and second in the solution of Newton's equation [6] for the reduced model. The vectorization of the second step is by now a standard procedure, since the Jacobian of the reduced system is dense and stored in full format. Library subroutines are available for carrying such computation (see, for example, (2)). The vectorization of the first step is made possible by computing in sequence the perturbed values of each core and loop variables corresponding to the perturbations of each loop variable. Each core and loop equation is now a vector expression where each endogenous variable is a vector of length s. At the beginning of the computation, the vectors of loop variables are initialized to the base values of these loop variables. Then, for each loop variable j, the jth element of the corresponding vector is perturbed by h_j. The core equations are then evaluated recursively, then the loop equations. The result is a $s \times s$ matrix of error terms, where entry (i,j) represents the error on loop equation i corresponding to the perturbation of the jth loop variable. The Jacobian of the reduced system is immediately obtained by dividing each column by the corresponding h_j.

3. TECHNICAL ISSUES

The method raises several technical issues which are now addressed. Let us first consider the relationship of the solution algorithm to Newton's method applied to the original problem.

The original and reduced system are equivalent in the sense that a solution to the reduced system is also a solution to the original system. Moreover, the following fact can be established:

Fact: Let the notation be as in section 2. If the functions g and h are linear with respect to x, then RN and GN, started from a common value y_0 will generate the same sequence $\{y_k\}$ for the loop variables.

Proof: Dropping the k subscript for clarity, let the current point be (x, y) and the corresponding residuals be (b, c). The Global Newton's step for the loop variables is

$$t = T^{-1}[-c + S\,G^{-1}b] \qquad [10]$$

Let us consider RN applied to the same point. First, x is evaluated so that [3] holds. Since g is linear with respect to x, the solution is:

$$x^+ = x - G^{-1}b$$

The new residuals of the loop equations are then:

$$h(x^+, y) = c - S\,G^{-1}b$$

Since S and G are constant, T^* evaluated at (x^+, y) is the same as T^* evaluated at (x, y). The Reduced Newton step for the loop variables is then:

$$t^+ = -T^{-1}\,h(x^+, y) = -T^{-1}(c - S\,G^{-1}b) \qquad [11]$$

which is the same as the GN step in [10].

In the general nonlinear case, however, the paths to a same solution will be different. In the case where multiple solutions to the original problem exist, it is even conceivable that RN and GN, started from the same point, would yield different solutions.

Two convergence tests are applied at each iteration. Iterations terminate if the maximum relative error on the loop equations is less than ε_1:

$$\max_i \left| \frac{h_i(x_k, y_k)}{x_i} \right| \le \varepsilon_1 \qquad [12]$$

Iterations also terminate if the maximum relative change in the variables from one iteration to the next is less than ε_2:

$$\max_i \left| \frac{y_{ik} - y_{ik-1}}{y_{ik-1}} \right| \le \varepsilon_2 \qquad [13]$$

Note that the core equations are by construction exactly verified at each iteration, so that criterium [12] implies that the maximum relative error over all equations is less than ε_1. Well conditioned problems should terminate with criterium [12]. Criterium [13] is tested after [12] to terminate iterations when iterates stabilize at a point away from a local solution.

The selection of the perturbation scalar h_j is now considered. The problem is to choose for each variable y_j a perturbatin h_j such that it minimizes truncation and cancellation errors on the evaluation of g and h.

At this point, a trial-and-error approach has been used, with the selected h_j's corresponding to the perturbations yielding the best convergence properties. A more systematic approach is the object of current research.

4. CODING

The coding is best explained through an example, for which pseudo-code will be shown. Consider the system arranged in quasi-triangular order:

$$
\begin{aligned}
z_1 &= f_1(z_{11}, \ldots, z_{15}) \\
z_2 &= f_2(z_1, z_{11}, \ldots, z_{15}) \\
z_3 &= f_3(z_1, z_2, z_{11}, \ldots, z_{15}) \\
&\ldots\ldots\ldots\ldots \\
z_{10} &= f_{10}(z_1, \ldots, z_9, z_{11}, \ldots, z_{15}) \\
z_{11} &= f_{11}(z_1, \ldots, z_{15}) \\
&\ldots\ldots\ldots\ldots \\
z_{15} &= f_{15}(z_1, \ldots, z_{15})
\end{aligned}
$$

the core variables are z_1 through z_{10}, and the loop variables z_{11} to z_{15}. Let z_k be the current point. It is stored in row 6 of array Z. The error at the current point is stored in row 6 of array PHI. The reduced Jacobian at z_k is evaluated as follows.

1. Initialize the array of loop variables to current point value

```
for i = 1 to 5
    for j = 11 to 15
        Z(i, j) = Z(6, j)
```

2. Set perturbed values of loop variables

```
for i = 1 to 5
    Z(i,i + 10) = Z(i,i + 10) + H(i)
```

3. Execute subroutine REDUC. A 5×5 array PHI is returned. Each row is the perturbed value of the error vector for the corresponding perturbed loop variable.

4. Compute the transposed reduced jacobian by dividing each row i of PHI by the corresponding perturbation H(i)

```
for i = 1 to 5
    for j = 1 to 5
        PHI(i, j) = (PHI(i, j) - PHI(6, j)) / H(i)
```

The relevant portion of subroutine REDUC is as follows:

```
nloop = 5
for i = 1 to nloop
    Z(i,1) = F1(Z(i,11), . . ., Z(i,15)
for i = 1 to nloop
    Z(i,1) = F2(Z(i,1), Z(i,11), . . ., Z(i,15))
    . . . .
for i = 1 to nloop
    Z(i,10) = F10(Z(i,1), Z(i,9), Z(i,11, . . ., Z(i,15))
for i = 1 to nloop
    PHI(i,1) = F11(Z(i,1) . . ., Z(i,15))
    . . . .
for i = 1 to nloop
    PHI(i,5) = F15(Z(i,1), . . ., Z(i,15))
```

5. NUMERICAL RESULTS

The authors selected for experimentation the Texas Econometric Model (version M5) developed by the Bureau of Business Research of The University of Texas in Austin. With 293 equations, this model is characteristic of intermediate size econometric models. It can be partitioned into a 59 equation recursive prologue, a 201 equation simultaneous block, and a 33 equation recursive epilogue. The three algorithms considered were the reduced Newton (RN), the modified reduced Newton (mRN), and the Gauss Seidel (GS). The mRN algorithm was obtained by using the LU decomposition of T* computed at the first iteration in all subsequent iterations. The GS algorithm employed the ordering and normalization of the RN algorithm with each loop equation normalized on a loop variables.

The algorithms were coded in FORTRAN 77 and compiled with the FTN200 compiler on the CYBER 205 at Purdue University. For the scalar runs a scalar version of LEQFIT subroutine from the IMSL subroutine library was employed to solve the linear system. For the vectorized runs the GEL subroutine from the MAGEV library (2) was employed. The results for solving the model for one period (1970), with a convergence criterion ε_2 = .1e-3 are displayed in the following table:

Table 1

Number of iterations and time (sec) to solve model M5 for 1970

Algorithm	No Vectorization	Vectorization
GS	19 (.0365)	19 (.0318)
RN	4 (.1454)	4 (.0546)
mRN	4 (.0586)	4 (.0200)

For the GS code the recursive nature of the algorithm prohibits any significant vectorization. The only operation which can be vectorized is the storing of the current point at a given iteration for comparison with the result of the next iteration. In the RN algorithm, most of the steps can be vectorized. In the mRN code, the GEL subroutine solves the linear system [6] in less the 10% of the time required to obtain a single scalar solution to the core and loop equations. This means that in the mRN algorithm the second and subsequent iterations are obtained at a cost only slightly greater than the cost of a Gauss Seidel iteration. With vectorization, mRN achieves a saving of about 1/3 over GS. On the CYBER 205 setting up the vector pipeline has a substantial overhead which can be seen in Table 2. The table displays the time needed to execute the subroutine REDUC for various values of parameter nloop (see Section 4).

Table 2

Time to solve the core and loop

nloop	1	6	11	16	21
scalar code	.0018	.0078	.0142	.0204	.0268
vector code	.0053	.0062	.0079	.0097	.0103

Given the overhead in setting up the pipeline, the time to solve the model using the scalar processor is less than the time for the vector processor for nloop less than 8. Thus, when using the CYBER 205, trying to minimize the number of loop variables is not too relevant, at least when intermediate size models are considered. On a CRAY machine, however, some increase in speed occurs even with a vector of length two. Hence where the number of loop variables is small one would expect the RN and mRN algorithms to be much more effective on CRAY machines.

6. CONCLUSION

A Newton-type algorithm adapted to vector processing has been described. Preliminary numerical results are encouraging: An m-step Newton's method was found to be 33% faster than the Gauss-Seidel algorithm in solving an econometric model of intermediate size.

Further research is planned along two axis. The first one is to perform more experiments, this time using a much larger multicountry model with a loop of over 100 variables. It is also intended to carry the same computations on a CRAY X/MP-24, soon to be installed at the University of Texas in Austin. The other axis is to develop a systematic approach to some aspects of the algorithm, in particular to the choice of the perturbation values used in the computation of the reduced Jacobian by finite differencing.

7. BIBLIOGRAPHY

(1) Nepomiastchy et al. "Adapted Methods for Solving Optimizing Quasi-Triangular Econometric Models," *Annals of Economic and Social Measurement*, Vol. 12, 1978.

(2) "The Math-Geophysical Vector Library", document 22-MAGEV, Purdue University Computing Center.

B-SPLINE ESTIMATION OF THE HAZARD FUNCTION IN PERIOD ANALYSIS

John J. Hsieh, University of Toronto

This article develops a precise method for estimating the hazard function, survival function and density function for period analysis using B-splines. Explicit expressions for the representation of the hazard function as linear combination of quadratic B-splines are obtained. Accuracy is achieved by using three overlapping consecutive age-specifc death rates covering three years each, as coefficients of the B-spline basis defined on a single-year uniformly-spaced knot sequence. The exact expressions of various functions describing the probability distribution of the lifetime are derived from the hazard function. The methods are illustrated using 1981 Canadian male population and death data.

1. INTRODUCTION

The intent of this article is to employ the B-spline basis to estimate the hazard function (force of mortality) as well as its associated survival and density functions using death and population data from period analysis.

Data from government publications available for the study of the distribution of the human life length come from two sources: Those classified by age-of-death come from vital registration and yield counts of occurences of deaths grouped by single-year age intervals while those pertaining to age-still-alive come from census and give rise to counts of population size grouped in five-year age intervals.

A well-known method of estimation as well as approximation used in mortality analysis is to compute the age-specific death rate as the ratio of the number of deaths to the number of person-years of exposure in a given age interval and to estimate the hazard function at every age within the age interval by the death rate for that interval. The difficulty with this procedure is that the hazard function so estimated is constant for every age within an age interval and jumps at every division point between two age intervals, so that the hazard function is estimated by a step funrtion with step width normally five years long.

In this article we shall improve this time - honoured procedure by smoothing out the step function estimate using spline functions. This is accomplished by combining the death rate in the age interval within which the hazard function is sought with the death rates from the left and the right adjacent intervals and by redistributing them in a quadratic fashion among these consecutive overlapping age intervals. The technique with which this is effected is the representation of the hazard function as a linear combination of the quadrrtic B-splines on single-year uniformly-spaced knot sequences with the overlapping three-year age-specific death rates as the coefficients. To compute the death rates, the populations in five-year age groupings are cumulated and then interpolated into single years using a complete cubic spline. Once the hazard function is obtained, the survival function and the death density function are evaluated directly from the hazard function by exact integration.

The spline functions employed in this paper both for interpolation and for estimation enjoy **minimum** norm, best approximation and fast convergence properties. The set of the B-spline basis is a generalization of the "hat" functions and is a well-conditioned basis for spanning spline functions. As a Peano Kernel, it provides a local partition of unity on the entire agespan with small supports.

Since the behavior of the hazard function (in particular, its speed of decline) during the first year of life differs from that for the remainder of the lifespan and since the infant population tends to be underenumerated, the method and data for estimating the hazard function for ages under one should differ from the methods of estimation for the remaining life. An infant mortality law and a method for estimating the hazard and other related functions for the first year of life have recently been provided by the author (see Hsieh, 1985). We shall make use of the results from that work and in this paper concentrate on the agespan $[t_1,t_n]$, where t_1 = 1 year and t_n may be taken as 85 or 90 years or whatever advanced age depending on the availability and reliability of data at these ages.

In Section 2, explicit expressions are derived for the quadratic B-splines as well as their derivatives and integrals. Several useful simple properties of B-splines are also discussed. Section 3 describes the method of estimation of the hazard function using B-splines. Explicit formulas are given for estimates of the hazard function, survival function and the death density function based on single-year uniformly-spaced knot sequences. Section 4 provides an example of estimation using 1981 Canadian male population and death data. A comparison is made with other two existing methods of estimation for period analysis.

2. B-SPLINES AND THEIR PROPERTIES

There are several ways of defining a B-spline (B stands for basis). We shall use a definition consistent with current usage and appropriate for our application (For a survey and list of references about B-splines see de Boor, 1976). For a chosen sequence of knots $t_{-r+2} \leq \ldots < t_1 < t_2 \ldots < t_n \leq \ldots \leq t_{n+r-1}$ $(r=1,2,\ldots)$, the i-th B-spline of order r, denoted by B_{ir} or simply by B_i with the order r understood, is a piece-wise polynomial of degree r-1 defined as the product of $(t_{i+r}-t_i)$ and the i-th divided difference of the truncated power function $(t-x)_+^{r-1} \equiv \max(0,(t-x)^{r-1})$ with respect to t at the knots $t_i, t_{i+1}, \ldots, t_{i+r}$ $(i=-r+2,\ldots,n-1)$. In symbols,

$$B_{ir}(x) = (t_{i+r}-t_i) f[t_i, t_{i+1}, \ldots, t_{i+r}; x], \quad (1)$$

where $f[t_i,\ldots,t_{i+r};x]$ is the r-th divided difference of $f(t;x)=(t-x)_+^{r-1}$ at the points $t_i, t_{i+1}, \ldots, t_{i+r}$. In other words, $f[t_i,\ldots,t_{i+r};x]$ is the leading coefficient of the polynomial of degree r which agrees with the function $f(t;x) \equiv (t-x)_+^{r-1}$ at the points $t_i, t_{i+1}, \ldots, t_{i+r}$.

Since published data on populations and deaths are given in single-year or five-year age groupings, to construct a B-spline basis for the purpose of estimating the hazard function, we shall partition the agespan $[t_1, t_n]$ by choosing the exact integral ages as the division points of the age axis and place one knot t_i on each of the interior division points, $i=2,3,\ldots,n-1$, and r knots on each of the two boundary points so that $t_1=t_0=\ldots=t_{-r+2}$ are the initial knots and $t_n=t_{n+1}=\ldots=t_{n+r-1}$ are the final knots.

Both theoretical and empirical considerations indicate that choice of r=3 or 4 will produce optimal results. For estimating the hazard function, quadratic B-splines (r=3) are preferable to cubic B-splines (r=4). The use of the former will avoid possible undue undulations caused by the use of cubic or higher-degree splines, at the same time retain the smoothness of a spline function, with the added benefit of simpler expressions for the hazard function as well as the survival function and the death density function derived therefrom.

Using the definition of $B_{ir}(x)$ given above and the recursive relations of the divided difference given below:

$$f[t_i,\ldots,t_{i+r};x] = \frac{f[t_{i+1},\ldots,t_{i+r};x]-f[t_i,\ldots,t_{i+r-1};x]}{t_{i+r}-t_i}, \quad (2)$$

(when multiple knots occur, then the derivatives naturally enter (2), so that

$$f[t,t;x] = f'(t;x)$$

$$f[t,t,s;x] = \frac{f'(t;x)-[f(t;x)-f(s;x)]/(t-s)}{t-s} \quad (3)$$

$$f[t,t,t;x] = f''(t;x)/2! \text{ etc.,})$$

explicit formulas for B-splines of order r=3 are derived and given in Column 2 of Table 1. From the expressions in Column 2, the formulas for the first order derivatives and integrals are derived and shown in Columns 3 and 4, respectively (where we have used the notation $\lambda_i=t_{i+1}-t_i$ and $\psi_{i,k}(x)=\int_{t_{i+k}}^{x} B_i(y)dy$ for $x\in[t_{i+k},t_{i+k+1})$, $k=0,1,2$). Notice that, for $i=1,2,\ldots,n-3$, $B_i(x)$ is a three piece quadratic with support (t_i,t_{i+3}) and is continuously differentiable at each of its four knots t_i, t_{i+1}, t_{i+2} and t_{i+3}. For i=0, $B_i(x)$ is a two-piece quadratic with support (t_{i+1},t_{i+3}) and is continuously differentiable only at the two knots t_{i+2} and t_{i+3} but not at the boundary point x=1 because two knots t_i and t_{i+1} are placed there. For i=-1, $B_i(x)$ consists of only one quadratic function with support $[t_{i+2},t_{i+3})$ and is neither continuous nor continuously differentiable at x=1 because three knots are placed there. Similarly, the same may be said of the case with i=n-2 and n-1 on the opposite end of the agespan in a symmetrical fashion. The expression for the value of the $B_i(x)$ and its derivative at each knot is also shown in Table 1.

B-splines have many desirable properties, some of which we will be making use of are listed below:

(i) For each i, $B_{ir}(x)$ is a spline function of degree r-1 on the real line if no multiple knots are involved and hence that $\Phi(x) = \int_{-\infty}^{x} B_{ir}(y)dy$ is a monotone increasing spline function of order r+1 with $\Phi(x)=0$, for $x \leq t_i$.

(ii) The area under the B_i curve is given by

$$\int_{-\infty}^{\infty} B_i(x)dx = (t_{i+r}-t_i)/r. \quad (4)$$

Thus, for single-year uniformly-space knot sequences, $t_i=i$, all i, this integral becomes unity so that $B_i(x)$ represents a probability density on the real line.

(iii) The support of the $B_i(x)$ functions is restricted to (t_i,t_{i+r}), i.e.

$$B_i(x) \begin{cases} > 0 \text{ for } x\in(t_i,t_{i+r}), \\ = 0 \text{ for } x\notin(t_i,t_{i+r}). \end{cases} \quad (5)$$

(iv) The sum of the B_i's at a given x is unity,

TABLE 1. EXPRESSIONS FOR QUADRATIC B-SPLINES, THEIR DERIVATIVES AND INTEGRALS

Xε	$B_i(x) =$	$B_i'(x) =$	$\psi_{ik}(x) =$
$(-\infty, t_i]$	0	0	
	$B_i(t_i)=0$	$B_i'(t_i)=0$	
$[t_i, t_{i+1}]$	$\frac{(x-t_i)^2}{(\lambda_i+\lambda_{i+1})\lambda_i}$	$\frac{2(x-t_i)}{(\lambda_i+\lambda_{i+1})\lambda_i}$	$\frac{(x-t_i)^3}{3(\lambda_i+\lambda_{i+1})\lambda_i}$
	$B_i(t_{i+1})= \frac{\lambda_i}{\lambda_{i+1}+\lambda_i}$	$B_i'(t_{i+1})= \frac{2}{\lambda_{i+i}+\lambda_i}$	
$[t_{i+1}, t_{i+2}]$	$1-\lambda_{i+1}^{-1}\left[\frac{(x-t_{i+1})^2}{\lambda_{i+1}+\lambda_{i+2}} + \frac{(t_{i+2}-x)^2}{\lambda_i+\lambda_{i+1}}\right]$	$2\lambda_{i+1}^{-1}\left[\frac{t_{i+1}-x}{\lambda_{i+1}+\lambda_{i+2}} + \frac{t_{i+2}-x}{\lambda_i+\lambda_{i+1}}\right]$	$-(3\lambda_{i+1})^{-1}\left[\frac{(x-t_{i+1})^3}{\lambda_{i+1}+\lambda_{i+2}} - \frac{(t_{i+2}-x)^3-\lambda_{i+1}^3}{\lambda_i+\lambda_{i+1}}\right] +(x-t_{i+1})$
	$B_i(t_{i+2})=\frac{\lambda_{i+2}}{\lambda_{i+1}+\lambda_{i+2}}$	$B_i'(t_{i+2})=-\frac{2}{\lambda_{i+1}+\lambda_{i+2}}$	
$[t_{i+2}, t_{i+3}]$	$\frac{(t_{i+3}-x)^2}{(\lambda_{i+1}+\lambda_{i+2})\lambda_{i+2}}$	$\frac{2(x-t_{i+3})}{(\lambda_{i+1}+\lambda_{i+2})\lambda_{i+2}}$	$\frac{\lambda^3_{i+2}-(t_{i+3}-x)^3}{3(\lambda_{i+1}+\lambda_{i+2})\lambda_{i+2}}$
	$B_i(t_{i+3})=0$	$B_i'(t_{i+3})=0$	
$[t_{i+3}, \infty)$	0	0	

i.e., for $x\varepsilon[t_j, t_{j+1}], j=1,\ldots,n-1,$

$$\sum_i B_i(x) = \sum_{i=j-r+1}^{j} B_i(x) = 1. \tag{6}$$

and hence for single-year knot sequence,

$$\sum_{i=j-r+1}^{j} \int_j^{j+1} B_i(x)dx = 1, \tag{7}$$

so that $\sum_{i=j-r+1}^{j} B_i(x)$ represents a probability density on $[j,j+1]$.

(v). At a given division point on the age axis, the number of continuity conditions plus the number of knots equals the order of the B-splines.

(vi). Every spline function f(x) of degree r-1 based on the above sequence is uniquely represented by a linear combination of the B-splines basis. Thus, for $x\varepsilon[t_j, t_{j+1}]$, for some $j=1,\ldots,n-1$,

$$f(x) = \sum_{i=j-r+1}^{j} a_i B_i(x) . \tag{8}$$

Furthermore, the linear span is strictly convex.

Properties (i) through (iv) are direct consequences of the definition of B-splines and/or Peano's theorem. Properties (v) and (vi) were shown by Curry and Schoenberg (1966). Allthese properties can be verified from the explicit formulas for the quadratic B-splines and their integrals given in Table 1. There are other mathematical properties most of which are derivable using the total positivity property of B-splines due to Karlin (1968). As more properties are uncovered about the B_i basis splines, their importance in both theoretical and applied works will become evident.

3. ESTIMATION OF HAZARD FUNCTION, SURVIVAL FUNCTION AND DEATH DENSITY FUNCTION

We shall represent the hazard function h(x) over the agespan $[t_1,t_n]$ by a spline function and, in accordance with equation (8), express it as a linear combination of the B-splines with the coefficients a_i to be estimated from the observed population and death data. Accordingly, the estimate of h(x) on $[t_1,t_n]$ may be written as

$$\hat{h}(x) = \sum_{i=-r+2}^{n-1} \hat{a}_i B_i(x). \qquad (9)$$

With quadratic splines, the explicit expressions for $B_i(x)$ are given in Table 1.

Determination of the coefficients of the B_i-splines in (9) requires information about h(x). If we are given a set of $h(x_i)$ values at some n-1 age points $x_i \varepsilon [t_i,t_{i+1})$, i=1,...,n-1, plus r-1 values of h(x) and its derivatives at the two boundaries, say, then we may solve (9) as a linear interpolation problem to obtain estimates of the (n+r-2) coefficients $\hat{a}_i$. However, it is difficult to obtain accurate values of h(x) at so many age points, much more so with its derivatives. This approach is, therefore, not feasible. Even though we do not know h(x), the population and death data do provide us with certain average values of h(x) over relatively small age intervals. These are weighted averages of h(x) with age distribution as weights and are known as age-specific death rates (see (10) below). Computationally, the death rate over the age interval [x,x+y], denoted by ${}_yM_x$, is the ratio of the number of deaths to the person-years of exposure in the age interval [x,x+y], the numerator coming from death (age-of-death) data while the denominator from population (age-still-alive) data. We shall now derive our estimates of the a_i's in terms of the ${}_yM_x$'s with specific choices of x and y and show that they are indeed good estimates.

For every sequence of r+1 knots, t_i, $t_{i+1},\ldots,t_{i+r}$, for i=-r+2,...,n-1, define the truncated mean $\bar{t}_i=(t_{i+1}+\ldots+t_{i+r-1})/(r-1)$ and the symmetrical range $2d_i$ where $d_i=\min(\bar{t}_i-t_i,t_{i+r}-\bar{t}_i)$ is the minimum length of age intervals from the truncated mean $\bar{t}_i$ to the two extreme knots. Then, the death rate over the age interval $[\bar{t}_i-d_i,\bar{t}_i+d_i]$ associated with these r+1 knots is given as the weighted average

$${}_{2d_i}M_{\bar{t}_i-d_i} = \frac{\int_{\bar{t}_i-d_i}^{\bar{t}_i+d_i} h(x)p(x)dx}{\int_{\bar{t}_i-d_i}^{\bar{t}_i+d_i} p(x)dx}, \qquad (10)$$

where p(x) is the population profile function. The numerator in (10) represents the number of deaths per unit time and the denominator the number of persons, both for the age interval $[\bar{t}_i-d_i,\bar{t}_i+d_i]$. It follows from (10) that if the hazard function is constant over $[\bar{t}_i-d_i,\bar{t}_i+d_i]$, then, $h(x)={}_{2d_i}M_{\bar{t}_i-d_i}$. This result implies the validity (but not accuracy) of the conventional method of estimation mentioned in Section 1.

To derive an estimate for a_i, we assume h(x) and p(x) to have derivatives up to the second order and expand them around $\bar{t}_i$ as follows:

$$h(x) = h(\bar{t}_i)+h'(\bar{t}_i)(x-\bar{t}_i)+o((x-t_i)\ ), \qquad (11)$$

$$p(x) = p(\bar{t}_i)+p'(\bar{t}_i)(x-\bar{t}_i)+o((x-\bar{t}_i)\ ). \qquad (12)$$

Then, substitute (11) and (12) into (10) and integrate out to obtain

$${}_{2d_i}M_{\bar{t}_i-d_i} = \frac{h(\bar{t}_i)p(\bar{t}_i)+o(d_i^2)}{p(\bar{t}_i)+o(d_i^2)}$$

$$= h(\bar{t}_i)+o(d_i^2). \qquad (13)$$

On the other hand, following the arguments used in the quasi-interpolant approximation of de Boor and Fix (1973), one easily obtains the following approximation for the B-spline coefficients:

$$a_i = h(\bar{t}_i)+O(d_i^2) \qquad (14)$$

Substituting (13) in (14), we have

$$a_i = {}_{2d_i}M_{\bar{t}_i-d_i}+O(d_i^2) \qquad (15)$$

Notice that the order of approximation obtained in (15) is independent of the order r of the spline used as long as r≥3. Equation (15) suggests that ${}_{2d_i}M_{\bar{t}_i-d_i}$ is a good estimator for a_i for small spacings between the knots such as single-year uniformly-spaced knot sequences. Substituting this estimate

$$\hat{a}_i = {}_{2d_i}M_{\bar{t}_i-d_i} \qquad (16)$$

into (9) yields

$$\hat{h}(x) = \sum_{i=j-r+1}^{j} {}_{2d_i}M_{\bar{t}_i-d_i}B_i(x), \qquad (17)$$

for $x \varepsilon [t_j,t_{j+1}]$, j=1,2,...,n-1, since, by Property (iii) of Section 2, each B_i has a support covering only r consecutive age intervals. Formula (17) is the estimator of the hazard function we propose. From Property (i) of Section 2, $\hat{h}$ is of continuous class C^{r-2} and its integral of continuous class C^{r-1} over (t_1, t_n). Because of (13), the estimator (17) is sharper than Schoenberg's shape-preserving or variation diminishing approximation (Schoenberg 1967). It can also be seen as a modification as well as generalization of Breslow's (1974) estimator for cohort analysis which employs constant splines over random death times. If h(x) is constant, then, in view of equation (6), equation (17) reduces to $\hat{h}(x) = {}_{2d_i}M_{\bar{t}_i - d_i}$ so that the estimation is exact following the arguments immediately below equation (10).

For quadratic splines with single year knot spacing and n=90, we have $r=3, t_i=i$, $i=-1,0,1,\ldots,90,91$ and 92. The area under the $B_i(x)$ courve for each single year age interval is: $\psi_{-1,2}(2)=\psi_{89,0}(90)=1/3, \psi_{0,1}(2)=\psi_{88,1}(90)=1/2$, and $\psi_{i,0}(i+1)=\psi_{i,2}(i+3)=1/6, \psi_{i+1}(i+2)=2/3, i=1,\ldots,87$. The estimators of the a_i's are obtained from the definition of $\bar{t}_i$ and d_i and equation (16) to be: $\hat{a}_{-1}=\hat{h}(1), \hat{a}_0={}_1M_1, \hat{a}_i={}_3M_i, i=1,\ldots,87$, $\hat{a}_{88}={}_1M_{89}, \hat{a}_{89}=\hat{h}(90)$. To obtain explicit formulas for $\hat{h}(x)$, we substitute into (17) the above estimates for a_i's and the expressions for $B_i(x)$ for single year knot spacing derived from Table 1. Integration of $\hat{h}(x)$ then leads to estimatesof the survival function and density function. Hence, the hazard function is estimated by

$$\hat{h}(x) = \hat{h}(1)(2-x)^2 - {}_1M_1(3x^2-10x+7)/2 + {}_3M_1(2-x)^2/2, \text{ for } 1 \le x \le 2;$$

$$\hat{h}(x) = {}_1M_1(3-x)^2/2 - {}_3M_1(2x^2-10x+11)/2 + {}_3M_2(x-2)^2/2, \text{ for } 2 \le x \le 3;$$

$$\hat{h}(x) = {}_3M_{j-2}(j+1-x)^2/2 - {}_3M_{j-1}[2x^2-2(2j+1)x+2(j-1)(j+2)-3]/2 + {}_3M_j(x-j)^2/2, \; j \le x \le j+1, \; j=3,\ldots,87,$$

$$\hat{h}(x) = {}_3M_{86}(89-x)^2/2 + {}_3M_{87}[1-((x-88)^2+(89-x)^2)/2] + {}_1M_{89}(x-88)^2/2, \text{ for } 88 \le x \le 89;$$

$$\hat{h}(x) = {}_3M_{87}(90-x)^2/2 + {}_1M_{89}[1-(x-89)^2-(90-x)^2/2] + \hat{h}(90)(x-89)^2, \text{ for } 89 \le x \le 89. \qquad (18)$$

The survival function is estimated by

$$\ln \hat{\bar{F}}(x) = \ln \hat{\bar{F}}(1) - \int_1^x \hat{h}(y)dy, \qquad (19)$$

where $\int_1^x h(y)dy = \sum_{i=1}^{j-1} H_i(i+1) + H_j(x), j=1,\ldots,89$, and $x \varepsilon [j, j+1]$, and $H_j(x) = \int_j^x \hat{h}(y)dy$ is given by

$$H_1(x) = \hat{h}(1)[(x-2)^3+1]/3 + {}_1M_1(3-x)(x-1)^2/2 + {}_3M_1(x-1)^3/6,$$

$$H_2(x) = {}_1M_1[(x-3)^3+1]/6 + {}_3M_1(2-x)(2x^2-11x+11)/6 + {}_3M_2(x-2)^3/6,$$

$$H_j(x) = {}_3M_{j-2}\psi_{j-2,2}(x) + {}_3M_{j-1}\psi_{j-1,1}(x) + {}_3M_j\psi_{j,0}(x), \text{ for } j=3,\ldots,87;$$

$$H_{88}(x) = {}_3M_{86}[1-(89-x)^3]/6 + {}_3M_{87}[[(88-x)^3+(89-x)^3-1]/6-(88-x)] + {}_1M_{89}(x-88)^3/6,$$

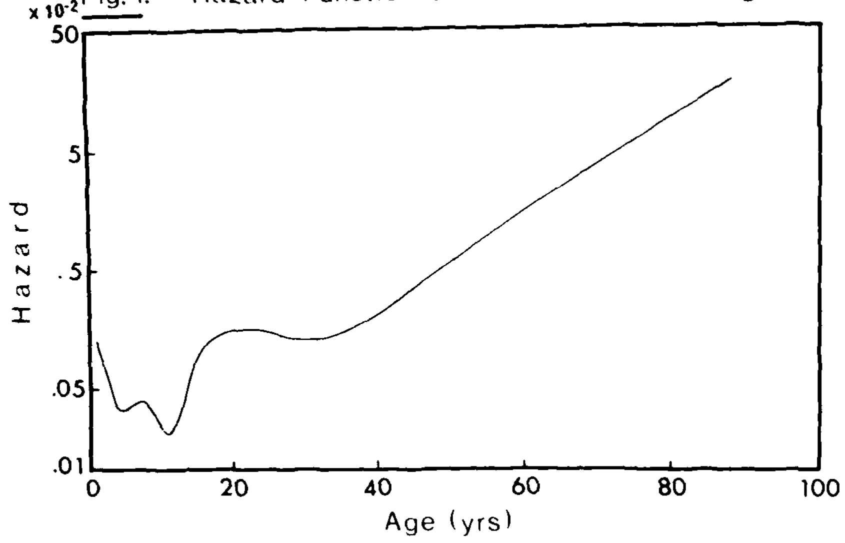
$$H_{89}(x) = {}_3M_{87}[1-(90-x)^3]/6 + {}_1M_{89}[x-89-(x-89)^3/3-[(x-90)^3+1]/6] + \hat{h}(90)(x-89)^3/3. \qquad (20)$$

To compute with the above formulas, three values $\hat{h}(1)$, $\hat{h}(90)$ and $\hat{\bar{F}}(1)$ are still needed. The values of $\hat{h}(1)$ and $\hat{\bar{F}}(1)$ are obtained from the method described in Hsieh (1985) and $\hat{h}(90)$ is obtained by fitting Gompertz curve through the last three age-specific death rates. Estimate of the death density function f(x) is obtained by substituting (18) and (19) into the following formula

$$\hat{f}(x) = \hat{h}(x)\hat{\bar{F}}(x). \qquad (21)$$

Because of age heapings and reporting and other errors, population data are normally published in five-year age groupings. To compute the death rates ${}_1M_1, {}_3M_i, i=2,\ldots,87$, ${}_1M_{89}$, we interpolate the five-year cumulated

Fig. 1. Hazard Function Canadian Males 1981 (Log Scale)



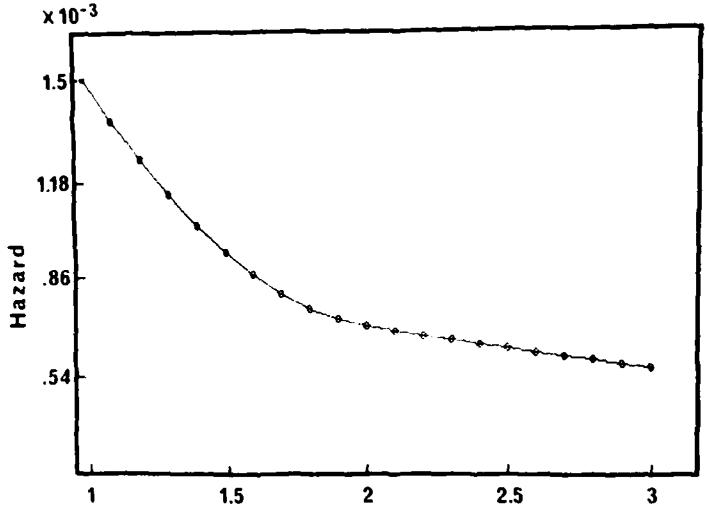


Fig. 1a. Hazard Function Canadian Males 1981 (Age 1-3 yrs)

Fig. 2. Survival Function Canadian Males 1981

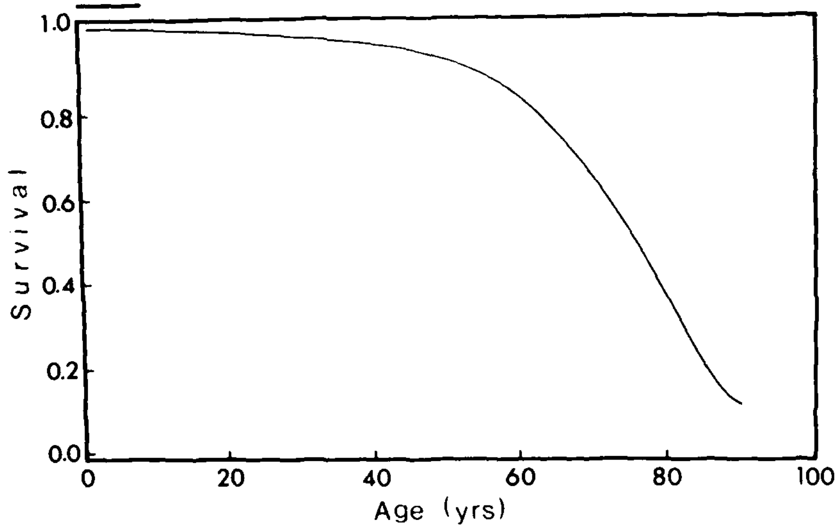
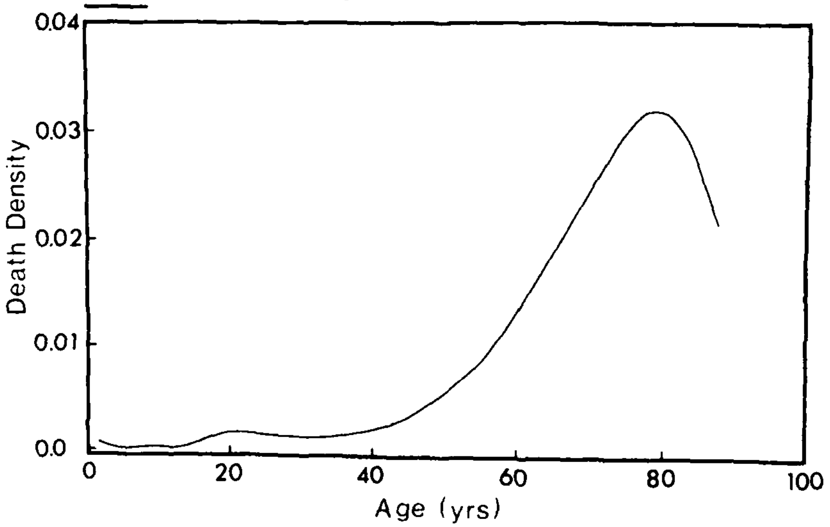


Fig. 3. Death Density Function Canadian Males 1981



population to the exact integral ages using a complete cubic spline function. Subtraction then yields the required population for the denominator of the death rates.

In conclusion, by using quadratic B-splines, we have considerably improved the accuracy of the hazard function estimate over the conventional method in which zero-degree B-splines are in effect used.

4. AN EXAMPLE

We have employed the tabulated registered deaths and census populations for Canadian males, 1981, to estimate the hazard function, using (18), (19), (20) and (21). Death rates are computed as the ratio of deaths to the mid-year population in the indicated age intervals. To obtain the population for these age intervals, the tabulated populations in five-year age groupings are cumulated backward and then interpolated to exact integral ages using complete spline interpolation using either the procedure given in Ahlberg, et.al. (1967) or that given in Schoenberg (1973). The initial endslope for the complete spline is estimated by subtracting the number of deaths under age one from the births and the final endslope by adding one-half the deaths in the last age interval to the population in that age interval. The estimates of the three functions h(x), $\bar{F}(x)$ and f(x) are shown in Figures 1, 2 and 3. Figure 1a magnifies $\hat{h}(x)$ for ages from 1 to 3 years. A comparison with the other two spline methods of estimating the hazard function and its derived functions, namely, those of Hsieh (1979) and Okamoto (1979), shows practically no differences in the results and the graphs obtained from the latter two methods are virtually indistinguishable from those shown in Figures1, 1a, 2 and 3.

The principal advantage of the present method lies in the fact that the hazard function can be integrated out exactly, resulting in a low degree polynomial spline.

REFERENCES

1. Ahlberg, J.H., E.N. Nielsen, and J.L. Walsh, The Theory of Splines and their Applications, Academic Press, New York, 1967.

2. Breslow, N. Covariance analysis of censored survival data. Biometrics 30, 1974, 89-99.

3. Curry, H.B. and I.J. Schoenberg. On Polya frequency functions, IV. The fundamental spline functions and their limits, J. d'Analyse Math. 17, 1966, 71-107.

4. de Boor,C. Splines as linear combinations of B-Splines: A Survey. Approximation Theory II (ed. Lorentz, G.G.) Academic Press, 1976, 1-47.

5. de Boor, C. and F. Fix. Spline approximation by quasi-interpolants, J. Approximation Theory 7, 1973, 19-45.

6. Hsieh, J.J. Use of cardinal splines in the construction of life tables, Proc. Comput. Sci. Statist. 12, 1979, 327-331.

7. Hsieh, J.J. Construction of expanded infant life tables: A method based on a new mortality law, Math Biosci 76, 1985, 221-242.

8. Karlin, S. Total Positivity and Applications to Analysis Vol. 1. Standford University Press,.Standford, CA. 1968.

9. Okamoto, D.M. On Approximating the Force of Mortality by Taking the Ratio of Histosplines and Its Application to the Life Table. Ph.D. Thesis, University of Toronto, Toronto, 1979.

10. Schoenberg, I.J. On spline functions,, in inequalities (ed.O. Shisha), Academic Press, New York, 1967, 255-291.

11. Schoenberg, I.J. Cardinal Spline Interpolation, CBMS Vol. 12, SIAM,. Philadelphia, 1973.

A LEISURELY LOOK AT THE MODELS OF UNCERTAINTY IN EXPERT SYSTEMS

Syni-An Hwang, SUNY Albany

§1. Introduction

What is an Expert System? Broadly speaking, we say that an Expert System is a computer program that uses explicitly represented knowledge and computational inference procedures to solve problems normally thought to require human expertise. More precisely, its purpose is to obtain the knowledge of experts in a particular domain, represent it in an expandable knowledge base, and transfer it to users for solving other problems in the same problem domain.

Currently, there is a great deal of interest in introducing uncertainty into an Expert System. Given a simple rule: If E, then H, the expert actually expressed the rule as: If E, then H with p. The user actually provides the information as: E is true with p_E. p is a measure of strength, with higher strength indicating greater power the evidence to confirm the hypothesis.

In this paper, we will study some of the best-known approaches to model the uncertainty in an Expert System. There have been papers devoted to similar topics, Bonissone (1982), and Black and Eddy (1985). One of the aims of this paper is to discuss the points raised by previous papers.

§2. Bayesian Probabilistic Models

In the Bayesian probability model, it is assumed that probability measures the degree of belief. Let P(H) denote the prior belief in H. When new evidence E is obtained, the posterior, P(H|E), denotes revised belief in H upon learning that E is true. Bayes' rule can be expressed in odds-likelihood form as:

$$O(H|E) = LR(E|H) \times O(H) \qquad (2.1)$$

where O(H), O(H|E), and LR(E|H) are prior odds, posterior odds, and likelihood ratio. To implement such a rule, the expert provides the likelihood ratio and the prior odds. Then by eq. (2.1), the system updates it to the posterior odds.

There exist potential problems of incoherency in this approach. For example, in theory, $LR(E|H)$ and $LR(E|\overline{H})$ satisfy

$$LR(\overline{E}|H) = (1-LR(E|H)\times P(E|\overline{H}))/(1-P(E|\overline{H})) \qquad (2.2)$$

In practice, the elicited probabilities, provided by the expert often violate eq. (2.2). Biases do exist even among well-trained experts (Kahneman, Slovic and Tversky (1982), and Shafer and Tversky (1984)).

Another type of incoherency, the difference between the uncertainty of user and expert could exist. When implementing an Expert System, the user makes relevant observations and provides $P_u(E|O)$ - the user's probability that E is true given O. Previous approaches make the implicit assumption, given O, $P_u(E|O) = P(E|O)$. In practice, it is less likely that the user and the expert will be coherent with each other.

There are other problems in applying a Bayesian probabilistic model to the Expert System, for example, the independence assumptions needed for reducing the computing complexity. Suppose we have n rules which say - If E_i, then H with probability P_i, i=1,...n,; we can construct a combined rule which says - If E, then H with probability P, where $E = E_1 \cap E_2 \cap \ldots \cap E_n$ and $P = P(H|E)$. Theoretically, we can compute P by Bayes' rule. However, in practice the likelihood ratio $LR(E_1 \cap E_2 \cap \ldots \cap E_n|H)$ needed in Bayes' rule is not provided in the Expert System. Therefore, we need the likelihood ratios for $\bigcap_{i=1}^{k} E_i$, k = 1,...,n, given H. This makes the model extensively complicated and hence inapplicable. To simplify the computation, two conditional independence assumptions have been made,

$$P(E_i \cap E_j|H) = P(E_i|H) \times P(E_j|H) \qquad (2.3)$$

$$P(E_i \cap E_j|\overline{H}) = P(E_i|\overline{H}) \times P(E_j|\overline{H}) \qquad (2.4)$$

Under these assumptions, we can compute the posterior odds as

$$O(H|E_1 \cap E_2 \cap \ldots \cap E_n) = (\prod_{i=1}^{n} LR(E_i|H)) \times O(H) \qquad (2.5)$$

The assumptions in eq. (2.3) and (2.4) had been given in the Expert System PROSPECTOR and were extensively criticized. One of the most notable papers is given by Pednault, Zucker and Muresan (1981). Using a result from Hussian (1972), they claim that under the above assumptions, no updating can take place. This result has been blunted by Glymour (1985), who points out an algebraic error in Hussian's derivation and hence invalidates the result of Pednault, et al.

Overall, the researchers of Expert System seem to understand that we need some independency assumptions to reduce the computing load. They also agree that the independence assumptions are unrealistic. To reconcile these conflicts, their attitude is to try to make the assumption of independence as realistic as possible (through the design of the Expert System). Hence, we can at least approximate the ideal Bayesian probabilistic model.

There has been a different approach to problem-solving systems generated mainly by statisticians. The most notable program is Kadane, et al. (1980). The main difference between this approach and the rule-based Expert System is in the assumption of the underlying statistical model and the existence of a prior to represent the knowledge of the expert.

Philosophically, there are different attitudes toward the statistically-based and rule-based Expert System. That is, does one want an increasingly large (due to addition of the new rule) and essentially deterministic rule-based Expert System, or a concise and probabilistic statistically-based Expert System? A formal comparison between the statistically-based and rule-based Expert System should be attempted.

In summary, the Bayesian probabilistic model has a concrete theoretical foundation. However, such practical problems as computational burdens and incoherent probability assessment make it less applicable in Expert System.

§3. Certainty Factors

The certainty factors approach originates from Carnap's confirmation theory (Carnap (1950)). Instead of saying, E implies H or E refutes H, the probability expresses the degree of implication of H afforded by E. According to Carnap's concept, "probability is much like personal probability, except that here it is argued or postulated that there is one and only one opinion justified by any body of evidence, so that probability is an objective logical relationship between an event A and the evidence B" (Savage (1961)).

The design of the MYCIN system (Shortliffe and Buchanan (1975)) is an implementation of Carnap's concept. They define a measure of belief (MB), and a measure of disbelief (MD) as the percentage of increases (decreases) on P(H) to P(H|E) relative to what is possible. By definition, we can prove that MD(H,E) is equal to MB($\overline{H}$,E). The overall certainty factor (CF) is defined as

$$CF(H,E) = MB(H,E) - MD(H,E) \qquad (3.1)$$

Four combining functions have been used in the MYCIN and cause a lot of criticism. Adams (1976) has shown that the first combining function (incrementally acquired evidence) implicitly assumes the independence of the evidence, a questionable assumption as we stated in section 2. The second and third combining functions are the controversial minimum and maximum rules borrowed from the fuzzy set theory. To apply the fourth combining function (strength of evidence) we have to assume the coherency between the user and the expert, a property that is doubtful in a real world.

In conclusion, we can view the certainty factors model as a Bayesian model with some ad hoc combining of rules. From a theoretical viewpoint, it does not appear useful, but it does have value from a practical point of view. Perhaps this could be best expressed by quoting Shortliffe and Buchanan: "The justification of our approach therefore rests not with a claim of improving on Bayes' theorem but rather with the development of a mechanism whereby judgmental knowledge can be efficiently represented and utilized for the modeling of medical decision making,..."

§4. Belief Function

Shafer's belief function originates from Dempster's upper and lower probabilities (e.g. Dempster (1967)). Assuming there is a set θ (frame of discrement) of n mutually exclusive and exhaustive propositions, $A_1, A_2, \ldots, A_n$. Shafer assigns probability mass on the power set of θ according to a basic probability function m(·), m: $2^\theta \rightarrow [0,1]$. A subset A of a frame θ is called a focal element if m(A) > 0. The belief function Bel: $2^\theta \rightarrow [0.1]$ is defined, on A, a subset of θ, as the sum of m(B) over all subsets B of A. Note there is a 1-1 relation between Bel(·) and m(·) given by

$$m(A) = \sum_{B \subset A} (-1)^{\backslash A-B \backslash} Bel(B) \qquad (4.1)$$

where \A-B\ is the cardinality of A-B. The plausibility, P*(A), is defined as 1 - Bel($\bar{A}$). By definition, Bel(A) ≤ P*(A). When Bel(A) = P*(A) for all subsets in 2^θ, the belief function reduces to the conventional probability. This occurs only if m(·) distributed all the mass on the singletons $A_1, \ldots, A_n$.

Dempster's rule of combination has been used to compute the combining belief functions. Let m_1, m_2 be two basic probability assignments over the same frame θ, with focal elements $A_1, \ldots, A_I$ and $B_1, \ldots, B_J$, respectively. If the normalized factor K = $1 - \sum_{A_i \cap B_j = \phi} m_1(A_i)\, m_2(B_j) > 0$, then the

Dempster's rule defines their orthogonal sum as

$$m_{12}(A) = (\sum_{A_i \cap B_j = A} m_1(A_i)\, m_2(B_j))/K \quad (4.2)$$

for all nonempty subsets A of θ, $m_{12}(\emptyset) = 0$.

To implement belief function in the rule-based Expert System, the user provides his belief on the evidence, and the expert provides his belief on each rule in the system. Dempster's rule of combination is then used to combine them.

There are several advantages in using the belief function in the Expert System.

1. Ignorance: When an expert(user) has complete confidence he can express his opinion as a probability. But when he is unable to commit all of his belief, he can choose to ignore the noncommitable part.

2. Ability to handle conflicting evidence: Except for the case of totally contradiction, Dempster's rule of combination provides a way to combine expert's and user's belief even when their beliefs are incoherent.

However, there are questions raised about the implementation of belief function.

1. Computational problem: The evaluation of the degree of belief is time-exponential in the cardinality of the propositions set, a problem for which real-time calculations in real-time situations are not possible on today's computers.

2. Normalization process: The normalization process used in Dempster's rule can lead to incorrect results. An instructive example has been given by Zadeh (1984) to show that the normalization process can produce counter-intuitive results when dealing with conflicting evidence.

In summary, in theory, the belief function models has a solid foundation but lacks empirical support. In practice, overloaded computing time makes the belief function model almost inapplicable at present.

§5. Possibility Theory

Possibility theory originates from Zadeh's fuzzy set theory. The formal definition and basic operator of fuzzy set theory were given in Zadeh's 1965 paper. Zadeh argued that the probability theory may be appropriate for problems involving the measure of information. It is inappropriate, however, for problems with the meaning of information. To overcome such problems caused by fuzziness (vagueness) of definition, Zadeh proposed the fuzzy set theory, which provides a formalism for treating such vagueness. A new terminology has been introduced by Zadeh named "membership function", which can be interpreted as a measure of fuzziness for inclusion of an object in a set. Let X be the space of points (or objects) of interest, say, X = {x}. A fuzzy set A in X is characterized by a membership function $f_A(\cdot)$, with $f_A(x)$ representing the grade of membership of x in A. The basic assumptions given by Zadeh can be expressed as follows:

$$\text{(i)} \qquad 0 \le f_A(x) \le 1 \qquad (5.1)$$

$$\text{(ii)} \quad f_{A\cup B}(x) = \max\{f_A(x),\ f_B(x)\} \qquad (5.2)$$

$$\text{(iii)} \quad f_{A\cap B}(x) = \min\{f_A(x),\ f_B(x)\} \qquad (5.3)$$

$$\text{(iv)} \qquad f_{\bar{A}}(x) = 1 - f_A(x) \qquad (5.4)$$

A, B are subsets of X; $\bar{A}$ is the complement of A.

The assumption in eq. (5.1) is not necessary, but it is convenient. When A is completely specified, $f_A(x)$ takes only the value of 1 or 0, respectively, according to whether x does or does not belong to A. Thus $f_A(x)$ reduces to the ordinary indicator function of a set A.

The second and third assumptions are the minimum and maximum rules for conjunction and disjunction. The last assumption is the complement rule. The example below, given by Black and Eddy (1985), shows the shortcomings of the above rules. Applying A and $\bar{A}$ to rules (5.2), (5.3) and (5.4) we have

$$f_{A\cup\bar{A}}(x) = \max\{f_A(x),\ 1 - f_A(x)\} \qquad (5.5)$$

and

$$f_{A\cap\bar{A}}(x) = \min\{f_A(x),\ 1 - f_A(x)\} \qquad (5.6)$$

The left hand sides of eq. (5.5) and (5.6) are definitely equal to 1 and 0, respectively, but the right hand side of the equations are not.

There are other critiques of the fuzzy set theory. French (1984) argued against the fuzzy set theory from the philosophical point of view. Two key points raised by French are: (i) Why should we believe (or assume) that the fuzziness in our perception is well (or precisely) modeled by the abstract concept of a fuzzy model? (ii) Since emphasizing imprecision does not seem to help us to understand the model better, why bother to bring in another level of reasoning about fuzziness?

The other question often raised is, "How can the grade of membership be determined?". In all, the fuzzy set theory does not seem to contain a rational or an empirical method for determining the value of the membership function.

So far, we have introduced the foundations of fuzzy set theory and mentioned various critiques of it, but it must be said that we might expect too much from a field with only twenty years of history. There has been much exciting research surrounding the field of fuzzy set theory. We will briefly mention some of it below.

A tremendous amount of effort has been put forth to combine the probability (or belief function) and fuzzy set theory. Zadeh points out that the concepts of possibility and necessity are the same as the concepts of support and plausibility in Shafer's belief function. Actually, if we discard the normalized factor in the Dempster rule of combination, (as suggested by Zadeh), the theory of belief function is exactly the same as the theory of possibility (Zadeh (1984)).

Another area of interest is the use of fuzzy set theory in linguistic approximation to the true qualification. Zadeh suggests a fuzzy-set theoretic interpretation of linguistic variables. That is to say, if the assertion of a fact is not known with precision, then it may be characterized linguistically as, say, true, not true, very true, etc., with each of the linguistic expressions representing a fuzzy subset of the unit interval. Zadeh treated such fuzzy reasoning of linguistics as an approximate reasoning.

It has been argued that it is more appropriate to present the conclusions in natural language form than in numerical form (e.g. Bonissone (1979)). Also, it has been argued that the people prefer to express their beliefs linguistically, rather than numerically. If the above arguments are true, then it is very natural to implement linguistic approximation for fuzzy reasoning in the Expert System. However, there appears to be very little psychological or theoretical evidence to support the arguments made above.

In summary, fuzzy set theory has a sound theoretical foundation, but it lacks normative justification as a belief function. It appears, at the moment, the inclusion of fuzzy logic in models of inexact reasoning adds an unnecessary extra complication.

§6. Conclusion

As we can see none of the above models seems to be better than the other in all applications. Moreover, the previous sections suggest that there is more than one type of uncertainty. This result suggests that we should use the different types of uncertainty in different situations and hence different models in different types of problems. Further study of the feasibility of the multi-model approach to uncertainty in the expert system is needed. Other interesting research topics can be found in Hwang (1986).

REFERENCES

Adams, J.B. (1976), "A Probability Model of Medical Reasoning and the MYCIN Model," Mathematical Biosciences, 32, 177-186.

Black, P.K., and Eddy, W.F. (1985), "Models of Inexact Reasoning", Technical Report #351, Dept. of Stat., Carnegie-Mellon U.

Bonissone, P.P. (1979), "A Pattern Recognition Approach to the Problem of Linguistic Approximation in System Analysis", Proceedings of the IEEE International Conference on Cybernetics and Society, pp. 793-798.

Bonissone, P.P. (1983), "Coping With Uncertainty in Expert Systems: A Comparative Study", Proceedings of the American Control Conference.

Carnap, R. (1950), Logical Foundations of Probability, Chicago, IL: University of Chicago Press.

Dempster, A.P. (1967), "Upper and Lower Probabilities Induced By a Multivalued Mapping", Annals of Mathematical Statistics, 38, 325-339.

French, S. (1984), "Fuzzy Decision Analysis: Some Criticisms", in Fuzzy Sets and Decision Analysis, TIMS Studies in the Management Sciences, eds. H.J. Zimmerman, L.A. Zadeh and B.R. Gaines, 29-44.

Glymour, C. (1985), "Independence Assumptions and Bayesian Updating", Artificial Intelligence 25, 95-99.

Hussian, A. (1972), "On the Correctness of Some Sequential Classification Schemes in Pattern Recognition", IEEE Trans. Comput. *21, 318-320.*

Hwang, S.A. (1986), "Models of Uncertainty in Expert Systems: A Review", Technical Report, Dept. of Math. & Stat., SUNY Albany.

Kadane, J.B., Dickey, J.M., Winkler, R.L., Smith, W.S., and Peters, S.C. (1980), "Interactive Elicitation of Opinion for a Normal Linear Model", JASA, 75, 845-854.

Kahneman, D., Slovic, P., and Tversky, A. (eds.) (1982), Judgment Under Uncertainty: Heuristics and Biases, New York: Cambridge University Press.

Pednault, E.P.D., Zucker, S.W., and Muresan, L.V. (1981), "On the Independence Assumption Underlying Subjective Bayesian Updating", Artificial Intelligence, 16, 213-222.

Shafer, G. (1976), A Mathemati .l Theory of Evidence, Princeton, NJ: Princeton University Press.

Shafer, G., and Tversky, A. (1984), "Weighing Evidence: The Design and Comparison of Probability Though Experiments", unpublished manuscript.

Shortliffe, E.H., and Buchanan, B.G. (1975), "A Model of Inexact Reasoning in Medicine", Mathematical Biosciences, 23, 351-379.

Zadeh, L.A. (1965), "Fuzzy Sets", Information and Control, 8, 338-353.

Zadeh, L.A. (1984), "Review of Shafer's A Mathematical Theory of Evidence", Artificial Intelligence Magazine, 5, 81-83.

BOOTSTRAPPING KOLMOGOROV-SMIRNOV STATISTICS, II

Alan Julian Izenman, Temple University

Summary. This is the second part of an empirical investigation into bootstrapping the two-sample Kolmogorov-Smirnov statistic D(m,n). The first part (ASA Proc. of the Statistical Computing Section, 97-101, 1985) dealt with estimating the standard error of D(m,n) by bootstrap methods and showed that the bootstrap performs very well, especially when compared with standard asymptotic approximations. In this paper, we carry out an empirical study of percentile estimation for the Kolmogorov-Smirnov statistic using the bootstrap procedure. As noted for other situations in which percentiles are estimated by the bootstrap, the number of bootstrap replications has to be large to obtain reasonable estimates, and even then those estimates are slightly on the low side. We also consider a logarithmic transformation of D(m,n), which has been suggested in the literature as having an approximately normal distribution for large m and n.

1. Introduction. The usual nonparametric two-sample problem can be stated in the following manner. *Given two independent random samples,*

$$X_1,X_2,\ldots,X_m \sim F, \quad Y_1,Y_2,\ldots,Y_n \sim G,$$

where F and G are both continuous, but unknown, distribution functions, and $m \leq n$, we are interested in comparing F with G to see whether they are in fact the same. The statistic that we consider here is the classical two-sample Kolmogorov-Smirnov distance,

$$D(m,n) = \sup_{x\in\mathbf{R}}|\hat{F}_m(x) - \hat{G}_n(x)|,$$

where $\hat{F}_m$ and $\hat{G}_n$ are the respective sample distribution functions obtained by placing mass 1/m on each X_i (i=1,2,...,m) and mass 1/n on each Y_j (j=1,2,...,n). Large values of D(m,n) suggest evidence against

$$H_0: F = G,$$

while small values of D(m,n) favor H_0 over the alternative, that $F \neq G$. Because of the global nature of the alternative, the Kolmogorov-Smirnov statistic has been criticised as not having high sensitivity to detect specific types of departure between the two distribution functions. However, the statistic is used sufficiently to warrant an investigation as presented here.

The distribution theory associated with the two-sample Kolmogorov-Smirnov statistic D(m,n) has been well documented. See Izenman (1985) for a summary, where it was pointed out that most of the results are complicated algebraically and are unsuitable for computation. All too often authors resort to using asymptotic approximations (for large m and n) to moments, critical values, or percentage points when, in fact, the sample sizes are small. Even then, the asymptotic distribution of D(m,n) is not normal, but involves an infinite summation whose value is approached most erratically.

In the first part of this empirical investigation (see Izenman 1985), the bootstrap procedure due to Efron (1979, 1982) was applied to evaluate the standard error of D(m,n) under the null hypothesis. Two possible bootstrap sampling procedures were compared for the two-sample problem:

(a) "separate" bootstrapping, in which a bootstrap sample was drawn (sampling with replacement) from the first sample, and then an independent bootstrap sample was drawn from the second sample;

(b) "combined" bootstrapping, in which the two original samples were pooled to form a combined sample of size m+n, and then two bootstrap samples were drawn (with replacement) from the combined sample.

In both cases, two bootstrap samples, one of size m and the other of size n, were generated and the statistic D*(m,n) was computed, where

$$D^*(m,n) = \sup_{x\in\mathbf{R}}|\hat{F}^*_m(x) - \hat{G}^*_n(x)|,$$

$\hat{F}^*_m$ and $\hat{G}^*_n$ being the respective sample distribution functions of the two bootstrap samples. This procedure was repeated a large number B times, yielding B bootstrap replications

$$D^*_1(m,n), D^*_2(m,n), \ldots, D^*_B(m,n).$$

These B values could then be used to estimate functionals of F and G. It was shown that the standard deviation of these B values, namely,

$$\{(B-1)^{-1}\Sigma_{b=1}^{B}(D^*_b - D^*_.)^2\}^{\frac{1}{2}},$$

where

$$D^*_. = B^{-1}\Sigma_{b=1}^{B} D^*_b,$$

is an excellent estimator of the standard error of D(m,n) under the null hypothesis when sampling is carried out using the "combined" bootstrap procedure. The "separate" bootstrap procedure is uniformly poorer for estimating those same standard errors. Simulations were carried out in each case by sampling with replacement from the uniform distribution on [0,1] according to the values of m and n, carrying out the appropriate bootstrap recipe described above, and repeating the procedure T (= # trials) times. The resulting standard deviations were then averaged over all T trials and compared with the exact standard error and an asymptotic approximation. The simulation parameters chosen were:

$$n = 25,50, \quad m = 5(5)n, \quad B = 100, \quad T = 100.$$

As a footnote to part one of this investigation, we have recomputed the simulations for n = 25 and m = 5(5)25 using B = 1000 and T = 20. The results are given in Table 1 for the "combined" bootstrap procedure only. It appears that accuracy in estimating standard errors of D(m,n) is improved by increasing B, the number of bootstrap replications, and not the number of trials T. In fact, variability of the bootstrap standard deviation has sharply decreased by going from B = 100, T = 100 to B = 1000, T = 20. It is, therefore, clear that the "combined" bootstrap can be used with a high degree of confidence to estimate the standard error of D(m,n) under the null hypothesis.

In Section 2, we consider the problem of estimating percentiles of the distribution of D(m,n) under the null hypothesis using the "combined" bootstrap procedure. Then, in Section 3, we consider a suggested logarithmic transformation of D(m,n). Some further directions of research in this area are discussed in Section 4 with particular reference to situations involving censored data.

2. Percentile estimation.

Estimating percentiles of a distribution is a much harder problem than estimating standard errors. This is especially true when using the bootstrap. The bootstrap method assumes implicitly that the true distribution is supported on the observed data points. Hence, the number of bootstrap replications, B, has to be larger to obtain reasonable accuracy in the tails of the distribution.

The "percentile method" (Efron 1979, 1982) is a straightforward procedure for estimating percentiles (and confidence intervals) from the results of bootstrap sampling by finding the appropriate percentiles of the bootstrap distribution. To be specific, let

$$\widehat{CDF}(t) = \mathrm{Prob}_*\{D^*(m,n) \le t\}$$

be the cumulative distribution function of the bootstrap distribution of $D^*(m,n)$. For $0 < \alpha < \frac{1}{2}$, the $(1-\alpha)\times 100$-th percentile of the distribution of D(m,n) is estimated by

$$PP(100(1-\alpha)) = \widehat{CDF}^{-1}(1-\alpha).$$

As in the previous Section, two bootstrap simulations were compared with n = 25 and m = 5(5)25:

PM1: B = 100, T = 100
PM2: B = 1000, T = 20.

For the purposes of simulation, CDF(t) was approximated by $\#\{D^*_b(m,n) \le t\}/B$ for each trial, and averaged over all T trials. A plus-or-minus figure was also calculated using the standard deviation of a specific percentile estimate over the T trials. The results are given in Table 2. "True" values were obtained by linearly interpolating in the tables of Kim and Jennrich (1970).

The bootstrap estimates of percentage points of D(m,n) were found to be slightly on the low side, as would be expected. In the simulations, we only considered 90, 95, and 99 percent points, and these were estimated between three and 14% too low. More centrally located percentiles, such as the 68th percentile, should be estimated better; however, such lower percentiles do not appear in any published set of tables for ready comparisons. Table 2 shows that even with B=1000, the bootstrap percentile method is still quite low in estimating true percentiles of the distribution of D(m,n). Even allowing for variability over the T trials, percentiles were not estimated anywhere as well as were standard errors. The "bias-corrected percentile method" (Efron 1982) did not appear to improve the estimates significantly and the results are not given here.

3. Transformation of D(m,n).

The normal distribution is not the standard large-sample approximation to D(m,n). However, Kim (1969) has suggested on the basis of empirical simulation studies that, for large m and n,

$$U(m,n) = \log_e\{D(m,n)/E_{(F,F)}D(m,n)\}$$

has approximately a normal distribution with mean -0.0450 and variance 0.0898. The drawback to using this transformation was that "one has to have the exact mean of D(m,n), an awesome task ... for n > 50," according to Kim. If we bootstrap D(m,n) and replace the exact mean by D(m,n) itself, then we have B bootstrapped versions of U(m,n), namely,

$$U^*_1(m,n), U^*_2(m,n), \ldots, U^*_B(m,n).$$

The assertion regarding the approximate normality of U(m,n) can be checked via a normal probability plot of the $U^*_b(m,n)$ values.

As an example, we used the stamp thickness data from Izenman and Sommer (1985); see also Example 2 in Izenman (1985). The data consist of two samples of measurements, one on m = 24 stamps watermarked "Papel Sellado" and the other on n = 289 unwatermarked stamps, both sets part of the 1872 Hidalgo Issue of Mexico. The "combined" bootstrap procedure was applied to the two samples, and the B = 1000 values of $U^*_b(m,n)$ were obtained. The mean and variance of those 1000 values were -0.054 and 0.110 respectively, and the normal probability plot exhibited a clear linear configuration.

Simulation results, not shown here, showed this transformation to be reasonable.

4. Censored data.

So far, discussion in this paper, and in Izenman (1985), has been confined to the complete sample situation. Recently, a number of papers have appeared in the literature in which the Kolmogorov-Smirnov statistic is used to compare two survival curves (or, distribution functions) for right-censored data. We refer the reader to Barr and Davidson (1973), Koziol and Byar (1975), Dufour and Maag (1978), Fleming et al (1980), Breslow et al (1984), and Sandford (1985).

There is some controversy regarding the suitability of the Kolmogorov-Smirnov statistic for comparing censored survival data. Certain authors (such as Fleming et al 1980) prefer the Kolmogorov-Smirnov statistic over the logrank and Gehan-Wilcoxon statistics in such situations. As Fleming et al remark, "it has been our frequent experience that substantial differences between two survival distributions may be apparent at one point in time, but fail to exist elsewhere. For example, certain treatments for coronary heart disease yield remarkably improved long-term survival, even though survival immediately following onset of treatment may be worse than that obtained with less aggressive alternative treatments." For detecting these types of 'crossed-hazards departures' from the null hypothesis, a modified version of the Kolmogorov-Smirnov statistic is preferred to the logrank or Gehan-Wilcoxon statistics, which appear to be insensitive to such departures. An alternative view is given by Breslow et al (1984), who develop a complementary criterion to be used in conjunction with the logrank procedure.

Comparisons between these different methods of studying differences between two survival curves have involved the use of asymptotic theory and, for small and medium sized samples, Monte Carlo simulations. Certain of the asymptotic results lead to normal approximations of the distributions of the statistics considered. The references listed above also include real data studies for purposes of illustration of the statistics.

It seems, therefore, that bootstrapping Kolmogorov-Smirnov statistics (or, possible modifications) can also be applied in the presence of censored data. In fact, Efron (1981) has investigated the use of the bootstrap for the Kaplan-Meier product-limit estimated survival curve. The bootstrap was used to assess the standard error of the Kaplan-Meier curve, functions (such as location estimates) of the Kaplan-Meier curve, and associated confidence intervals. Similar questions can be asked of the Kolmogorov-Smirnov and related statistics, based on bootstrap considerations. These questions will be addressed elsewhere.

References.

Barr, D.R. and Davidson, T.G. (1973), "A Kolmogorov-Smirnov Test for Censored Examples," Technometrics, 15, 739-757.

Breslow, N.E., Edler, L., and Berger, J. (1984), "A Two-Sample Censored-Data Rank Test for Acceleration," Biometrics, 40, 1049-1062.

Dufour, R. and Maag, U.R. (1978), "Distribution Results for Modified Kolmogorov-Smirnov Statistics for Truncated or Censored Samples," Technometrics, 20, 29-32.

Efron, B. (1979), "Bootstrap Methods: Another Look at the Jackknife," The Annals of Statistics, 7, 1-26.

Efron, B. (1981), "Censored Data and the Bootstrap," Journal of the American Statistical Association, 76, 312-319.

Efron, B. (1982), The Jackknife, The Bootstrap, and Other Resampling Plans. Society for Industrial and Applied Mathematics: Philadelphia, PA.

Efron, B. and Tibshirani, R. (1986), "Bootstrap Methods for Standard Errors, Confidence Intervals, and Other Measures of Statistical Accuracy," Statistical Science, 1, 54-77.

Fleming, T.R., O'Fallon, J.R., and O'Brien, P.C. (1980), "Modified Kolmogorov-Smirnov Test Procedures with Applications to Arbitrarily Right-Censored Data," Biometrics, 36, 607-625.

IMSL, Subroutines GGUD and NKS2.

Izenman, A.J. (1985), "Bootstrapping Kolmogorov-Smirnov Statistics," ASA Proceedings of the Statistical Computing Section, 97-101.

Izenman, A.J. and Sommer, C.J. (1985), "Philatelic Mixtures and Multimodal Densities," To appear in the Journal of the American Statistical Association.

Kim, P.J. (1969), "On the Exact and Approximate Sampling Distribution of the Two-Sample Kolmogorov-Smirnov Criterion D_{mn}, $m \leq n$," The Journal of the American Statistical Association, 64, 1625-1637.

Kim, P.J. and Jennrich, R.I. (1970), "Tables of the Upper α Points of the Exact Sampling Distribution of the Two Sample Kolmogorov-Smirnov Criterion, D_{mn}, $m \leq n$," In Biometrika Tables for Statisticians, Vol. II. Cambridge University Press: London.

Koziol, J.A. and Byar, D.P. (1975), "Percentage Points of the Asymptotic Distributions of One and Two Sample K-S Statistics for Truncated or Censored Data," Technometrics, 17, 507-510.

Sandford, M.D. (1985), "Nonparametric One-Sided Confidence Intervals for an Unknown Distribution Function Using Censored Data," Technometrics, 27, 41-48.

				"COMBINED" BOOTSTRAP SEs			
				B = 100, T = 100		B = 1000, T = 20	
n	m	true SE	ASE	Exp.	StDev.	Exp.	StDev.
25	5	0.121	0.128	0.122	0.010	0.121	0.003
	10	0.095	0.098	0.096	0.007	0.096	0.002
	15	0.083	0.085	0.084	0.006	0.084	0.002
	20	0.077	0.078	0.078	0.007	0.077	0.002
	25	0.073	0.074	0.074	0.006	0.074	0.002

Table 1. Comparison of true standard error, asymptotic standard error (ASE) using Smirnov approximation, and two bootstrap estimates of standard error using the "combined" procedure. Computations were carried out on a CDC Cyber 170/750 mainframe computer using Fortran programs and IMSL calls to subroutines GGUD and NKS2.

n	m	Method	PP90	PP95	PP99
25	5	"true" PP	0.572	0.631	0.748
		PM1	0.532 ± 0.031	0.583 ± 0.038	0.676 ± 0.057
		PM2	0.528 ± 0.016	0.596 ± 0.018	0.700 ± 0.024
	10	"true" PP	0.439	0.487	0.580
		PM1	0.407 ± 0.017	0.453 ± 0.024	0.525 ± 0.041
		PM2	0.415 ± 0.009	0.464 ± 0.011	0.562 ± 0.016
	15	"true" PP	0.384	0.426	0.510
		PM1	0.362 ± 0.020	0.403 ± 0.026	0.467 ± 0.037
		PM2	0.367 ± 0.010	0.407 ± 0.010	0.487 ± 0.012
	20	"true" PP	0.353	0.392	0.470
		PM1	0.335 ± 0.018	0.369 ± 0.021	0.429 ± 0.037
		PM2	0.337 ± 0.007	0.378 ± 0.006	0.455 ± 0.009
	25	"true" PP	0.349	0.386	0.461
		PM1	0.318 ± 0.020	0.350 ± 0.024	0.406 ± 0.036
		PM2	0.320 ± 0.000	0.358 ± 0.009	0.430 ± 0.022

Table 2. Comparison of true 90, 95, and 99 percent points (PP) of the Kolmogorov-Smirnov statistic with bootstrap estimates using the percentile method (PM). The "true" values were linearly interpolated from Kim and Jennrich (1970) tables. The first percentile method (PM1) used B = 100 and T = 100; the second percentile method (PM2) used B = 1000 and T = 20. Entries shown are mean ± stdev over the T trials. Computations were carried out as for Table 1.

CALCULATING IMPROVED BOUNDS AND APPROXIMATIONS FOR MULTIPLE COMPARISONS

James R. Kenyon, University of Connecticut

When applying multiple comparison procedures to a particular problem, the simultaneous confidence intervals usually are conservative. This occurs because procedures either use bounds in the coverage probability statement, or project a multivariate region to a multivariate rectangle (i.e., expand a confidence region to a confidence rectangle) (Miller 1981). Even conditional confidence intervals may not have the coverage claimed (Meeks and D'Agostino 1983).

Recently, there has been interest in obtaining improvements in bounds for multivariate probabilities (Games 1977, Glaz and Johnson 1984, Miller 1981, and Worsley 1982). It has been shown that the "usual," e.g., Bonferroni type bounds, often are not very useful, particularly when there are many events, B_i, and the $P(B_i)$ are not "small," or when there is a strong dependence structure in the multivariate distribution (Glaz and Johnson 1984, Miller 1981, Schwager 1984, and Worsley 1982).

Let us consider three problems for simultaneous confidence intervals and develop methods to calculate the coverage probability, or improved bounds for this probability. To obtain the approximation for the coverage probability, we will be using several bounds previously developed but not evaluated for these problems, including bounds using a conditional probability approach developed by Glaz and Johnson (1984).

These bounds are (where A_i is any event):

1) 1st Bonferroni bound

$$P\left(\bigcup_{i=1}^{t} A_i\right) \le \sum_{i=1}^{t} P(A_i)$$

2) 2nd Bonferroni bound

$$P\left(\bigcup_{i=1}^{t} A_i\right) \ge \sum_{i=1}^{t} P(A_i) - \sum_{i<k}^{t} P(A_i \cap A_k)$$

3) 1st Worsley bound

$$P\left(\bigcup_{i=1}^{t} A_i\right) \le \sum_{i=1}^{t} P(A_i) - \sum_{i=1}^{t-1} P(A_i \cap A_{i+1})$$

4) 2nd Worsley bound

$$P\left(\bigcup_{i=1}^{t} A_i\right) \le \sum_{i=1}^{t} P(A_i) - \frac{2}{t} \sum_{i<k}^{t} P(A_i \cap A_k)$$

5) Galambos bound

$$P\left(\bigcup_{i=1}^{t} A_i\right) \ge \frac{2}{k} \sum_{i=1}^{t} P(A_i) - \frac{2}{k(k-1)} \sum_{i<h}^{t} P(A_i \cap A_h)$$

for $k \ge 2$ and optimal

$$k = \left[2 \sum_{i<h}^{t} P(A_i \cap A_h) \Big/ \sum_{i=1}^{t} P(A_i)\right] + 2$$

6) Sidak bound

$$P\left(\bigcup_{i=1}^{t} A_i\right) \le 1 - \prod_{i=1}^{t} P(A_i)$$

7) Glaz and Johnson bounds and approximations:

a) $$P\left(\bigcup_{i=1}^{t} A_i\right) \le 1 - \prod_{k=1}^{t} P(A_k)$$

b) $$P\left(\bigcup_{i=1}^{t} A_i\right) \le 1 - P(A_1) \prod_{k=2}^{t} P(A_k | A_{k-1})$$

c) $$P\left(\bigcup_{i=1}^{t} A_i\right) \le 1 - P(A_1 \cap A_2) \times \prod_{k=3}^{t} P(A_k | A_{k-1} \cap A_{k-2})$$

d) $$P\left(\bigcup_{i=1}^{t} A_i\right) \le 1 - P(A_1 \cap A_2 \cap A_3) \times \prod_{k=4}^{t} P(A_k | A_{k-1} \cap A_{k-2} \cap A_{k-3})$$

e) $$P\left(\bigcup_{i=1}^{t} A_i\right) \le 1 - P(A_1 \cap A_2 \cap A_3 \cap A_4) \times \prod_{k=5}^{t} P(A_k | A_{k-1} \cap A_{k-2} \cap A_{k-3} \cap A_{k-4})$$

Note Glaz and Johnson have only obtained their third bound, c), for special cases. Calculations for d) and e) have not been obtained previously. Additionally, the conditional approximations b) - e) of Glaz and Johnson are not always guaranteed to be bounds. A sufficient, but not necessary, condition for these approximations to be bounds is that the multivariate distribution be multivariate totally positive of order 2 (MTP_2). For the multivariate normal, MTP_2 is equivalent to all the partial correlations being ≥ 0. Note that in the calculation of the bounds, one does not have a t-dimensional multivariate normal or multivariate t, but 2-dimensions in most cases and 5-dimension for the worst case.

First, consider a simple control problem for a one-way analysis of variance with a balanced design. Let Y_{ik} be distributed as independent normal random variables with mean μ_i and variance σ^2 for $i = 0,1,2,\ldots,t$ and $k = 1,2,\ldots,n$. Y_{0k}, $k = 1,2,\ldots,n$, is the control group and the remaining t groups, each of n observations, are the treatment groups. Thus, the $\bar{Y}_i$'s are distributed as independent normals with mean μ_i and variance σ^2/n for $i = 0,1,2,\ldots,t$. In this problem, one is usually interested in comparing the t treatments to the control. The resulting contrasts are $|\bar{Y}_0 - \bar{Y}_i|/S\sqrt{2/n} < c$ for $i = 1,2,\ldots,t$, where S^2 is the usual pooled estimate of σ^2.

$$1 - \alpha = P\left(\frac{|\overline{Y}_0-\overline{Y}_i|/\sigma^2}{S\sqrt{2/n}/\sigma^2} < c;\ i = 1,2,\ldots,t\right).$$

Without loss of generality, we can assume that $\sigma^2 = 1$ and $\mu_i = 0$ for $i = 0,1,2,\ldots,t$.

$$1 - \alpha = \int_0^\infty P(|\overline{Y}_0 - \overline{Y}_i| < cS\sqrt{2/n}\ ;\ i = 1,2,\ldots,t|S = s)\ f_S(s)\ ds$$

$$= \int_0^\infty \int_{-\infty}^\infty P(a < \overline{Y}_i < b;\ i = 1,\ldots,t|S = s,\ \overline{Y}_0 = y) \times f_{\overline{Y}_0}(y)\ f_S(s)\ dy\ ds$$

$$= \int_0^\infty \int_{-\infty}^\infty [P(a < \overline{Y}_i < b|S = s,\ Y_0=y)]^t \times f_{\overline{Y}_0}(y)\ f_S(s)\ dy\ ds$$

where $a = \overline{Y}_0 - cS\sqrt{2/n}$ and $b = \overline{Y}_0 + cS\sqrt{2/n}$.

Since $\overline{Y}_i$, $i = 0,1,2,\ldots,t$, are independent and identically distributed, and each is independent of S, $f_{\overline{Y}_i}(y) = \sqrt{n/2\pi}\ e^{-\frac{1}{2}ny^2}$. Let v = d.f. = $(t+1) \times (n-1)$, then vS^2/σ^2 is distributed Chi-Squared$_{(v)}$. From this, the density for S is $f_S(s) = \dfrac{v^{v/2}\ S^{v-1}\ e^{-\frac{1}{2}vS^2}}{2^{\frac{1}{2}v-1}\ \Gamma(\frac{1}{2}v)}$.

Now let $z = y\sqrt{n/2}$, and $w = \frac{1}{2}vS^2$. These transformations yield

$$1 - \alpha = 1/\sqrt{\pi} \int_0^\infty \frac{w^{\frac{1}{2}v-1}}{\Gamma(\frac{1}{2}v)}\ e^{-w} \int_{-\infty}^\infty [\Phi_{0,1/n}(b) - \Phi_{0,1/n}(a)]^t\ e^{-z^2} dz\ dw$$

where $\Phi_{\mu,\sigma^2}(x)$ is the c.d.f. at x for a normal distribution with mean μ and variance σ^2, $a = (z - c\sqrt{2w/v}) \times \sqrt{2/n}$ and $b = (z + c\sqrt{2w/v}) \times \sqrt{2/n}$.

Now, recall the Gauss-Laguerre formulas:

$$\int_0^\infty e^{-x}\ f(x)\ dx \simeq \sum_{i=1}^n A_i\ f(x_i)$$

and the Gauss-Hermite formulas:

$$\int_{-\infty}^\infty e^{-x^2} f(x)\ dx \simeq \sum_{i=1}^n B_i [f(x_i) + f(-x_i)]$$

$$\simeq \sum_{i=1}^n 2B_i f(x_i)$$

if $f(x)$ is symmetric about 0.

Thus, $$1 - \alpha \simeq (1/\sqrt{\pi}) \sum_{i=1}^{n_w} A_i \frac{w_i^{\frac{1}{2}v-1}}{\Gamma(\frac{1}{2}v)} \times \sum_{k=1}^{n_z} 2B_k [\Phi_{0,1/n}(b) - \Phi_{0,1/n}(a)]^t.$$

Where n_w is the number of points used in the Gauss-Laguerre approximation, n_z is the number of points used in the Gauss-Hermite approximation, $a = (z_k - c\sqrt{2w_i/v}) \times \sqrt{2/n}$, $b = (z_k + c\sqrt{2w_i/v}) \times \sqrt{2/n}$ and $\Phi_{\mu,\sigma^2}(x)$ is as above.

Since the variables used for conditioning have infinite limits of integration and the weight function appears in the density, the Gaussian methods do have a natural advantage. Other methods require either truncation of the integral besides performing an approximation of this truncated integral, or usage of a transformation of the variable of integration which yields finite limits of integration. In addition, Stroud and Secrest give a comparison of Gaussian quadrature with other methods. For an equal number of points the Gaussian quadrature error is comparable, even for cases where it is not believed the "best." They also compare different approaches to calculating some specific integrals with infinite limits, including transformations to integrals with finite limits. Moreover, Gaussian quadrature absorbs part of the densities in its weight function along with cancellation of other terms in the transformation to this form. All these results appear to indicate this approach favored for these particular densities.

Notice that the above formula is for the exact coverage probability and the same techniques can be applied to calculating probabilities for any number of events in this problem that we wish. Consequently, we can easily compute and compare the bounds with the exact probability. Also, the multivariate density in the control problem is MTP_2, thus guaranteeing the conditional bounds, but there is not much dependence structure for the conditional probability bounds to take advantage of. See Table 1 for a comparison of the bounds and exact probability for some particular cases that have previously been presented in the literature.

Secondly, let us consider the same problem, but for unbalanced data. That is, n is not necessarily the same for each group. Thus, we replace n above with n_i. We also allow c to vary for each group and replace it with c_i. The coverage probability $1 - \alpha$ is:

$$P(|\overline{Y}_0-\overline{Y}_i|/S\ \sqrt{1/n_0+1/n_1} < c_i; i = 1,2,\ldots,t)$$

$$= 1/\sqrt{\pi} \int_0^\infty \frac{w^{\frac{1}{2}v-1}}{\Gamma(\frac{1}{2}v)}\ e^{-w} \int_{-\infty}^\infty \prod_{i=1}^t [\Phi_{0,1/n_i}(b) - \Phi_{0,1/n_i}(a)]\ e^{-z^2} dz\ dz\ .$$

Where in the above, $b = (z + c_i\sqrt{2w/v})$

$\times \sqrt{1/n_0+1/n_1}$, $a = (z - c_i\sqrt{2w/v}) \times \sqrt{1/n_0+1/n_i}$, and $\Phi_{\mu,\sigma^2}(x)$ is the same as earlier. Thus

$$1 - \alpha \simeq (1/\sqrt{\pi}) \sum_{m=1}^{n_w} A_m \frac{w_i^{\frac{1}{2}v-1}}{\Gamma(\frac{1}{2}v)}$$

$$\times \sum_{k=1}^{n_z} 2B_k \prod_{i=1}^{t} [\Phi_{0,1/n_i}(b) - \Phi_{0,1/n_i}(a)]$$

where

$$a = (z_k - c_i\sqrt{2w_m/v}) \times \sqrt{1/n_0+1/n_i} ,$$

and

$$b = (z_k + c_i\sqrt{2w_m/v}) \times \sqrt{1/n_0+1/n_i} .$$

There is more computing to be done here, but for many applications the exact probability can usually be calculated along with the bounds. This is particularly interesting for the control problem since many practitioners often allocate more units to the control group. Dunnett (1964) has given tables and for optimal allocation suggests $n_0/n_i \simeq \sqrt{t}$ holds when the joint confidence level is ≥ 0.95. Refer to Table 2 for a comparison of the bounds and exact probabilities for several cases previously presented in the literature.

For the last problem, let the random variable Y have a normal distribution with a mean of XB and a variance of $\sigma^2 I$, where X is a known design matrix and B is a vector of unknown parameters. For t arbitrary contrasts, k_iB which must be estimable, the $P(|k_i\hat{B}|/s(k_i\hat{B}) < c_i$ for $i = 1,2,\ldots,t)$ is the multivariate probability of interest, where $s(k_i\hat{B})$ is the sample standard deviation of $k_i\hat{B}$ and $\hat{B}$ is an estimate of B.

For arbitrary contrasts only conditioning on S readily applies. This gives

$$P(|k_i\hat{B}|/S\sqrt{k_i(X'X)^-k_i'} < c_i;\ i=1,2,\ldots,t)$$

$$= \int_0^\infty f_S(s)P(|k_i\hat{B}| < c_iS\sqrt{k_i(X'X)^-k_i'} ;\ i = 1,2,\ldots,t|S = s)\, ds$$

where

$$P(|k_i\hat{B}| < c_iS\sqrt{k_i(X'X)^-K_i'} ;\ i = 1,2,\ldots,t|S = s)$$

is a multivariate normal probability. This will require much more computational work since in general there is no dependence structure that can readily be taken advantage of as was done in the earlier cases to obtain conditional independence. Additionally, there is no guarantee that $(k(X'X)^-k')^-$ has nonnegative off diagonal elements for the MTP_2 property.

Lastly, Gauss-Legendre formulas will be used to calculate the probabilities necessary for the bounds under study. For computational accuracy and efficiency, matrix decompositions and numerical methods for linear algebra must be utilized.

By taking advantage of any dependence structure and the form of the density, improved and even exact coverage probabilities can be calculated for multiple comparison problems or more generally for simultaneous confidence rectangles. With these capabilities, the inverse problem of determining c given α, n, and t can be performed as was done for Table 1.

REFERENCES

Dunnett, C. W. (1964). New Tables for Multiple Comparisons with a Control. Biometrics 20:482-491.

Dunnett, C. W. (1955). A Multiple Comparison Procedure for Comparing Several Treatments with a Control. Journal of the American Statistical Association 50:1096-1121.

Galambos, Janos. A New Bound on Multivariate Extreme Value Distributions.

Games, Paul A. (1977). An Improved t Table for Simultaneous Control on g Contrasts. Journal of the American Statistical Association 72:531-534.

Glaz, Joseph and Johnson, Bruce McK. (1984). Probability Inequalities for Multivariate Distributions with Dependence Structures. Journal of the American Statistical Association 79: 436-440.

Meeks, S. L., and D'Agostino, R. B. (1983). A Note on the Use of Confidence Limits Following Rejection of a Null Hypothesis. The American Statistician 37:134-136.

Miller, Rupert G. (1981). Simultaneous Statistical Inference. New York: Springer-Verlag.

Schwager, Stephen J. (1984). Bonferroni Sometimes Loses. The American Statistician 38:192-197.

Sidak, Zbynek (1968). On Multivariate Normal Probabilities of Rectangles: Their Dependence on Correlations. The Annals of Mathematical Statistics 39: 1425-1434.

Stroud, A. H., and Secrest, D. (1966). Gaussian Quadrature Formulas. Englewood Cliffs, New Jersey: Prentice-Hall.

Worsley, K.J. (1982). An Improved Bonferroni Inequality and Applications. Biometrika 69:297-302.

TABLE 1a

Games (1977) used the Sidak product bound to produce tables for contrasts. The following values for c were selected from this table for the degrees of freedom for error, number of contrasts, and alpha level given. Gamma 1 through 5 are the Glaz and Johnson bounds.

t	2	5	9	9
n	2	5	4	4
d.f.	3	24	30	30
confidence level	0.80	0.99	0.90	0.80
c	2.294	3.465	2.687	2.369
	Lower Bounds on the Confidence Level			
1st Bonferroni	0.78886	0.98973	0.89521	0.77976
Sidak (Gamma 1)	0.80001	0.98973	0.89996	0.80013
1st Worsley	0.83165	0.99046	0.90729	0.81441
2nd Worsley	0.83165	0.99046	0.90729	0.81441
Gamma 2	0.83165	0.99048	0.91014	0.82302
Gamma 3	---	0.99088	0.91648	0.83680
Gamma 4	---	0.99109	0.92063	0.84561
Gamma 5	---	0.99118	0.92341	0.85138
	Upper Bounds on the Confidence Level			
2nd Bonferroni	0.83165	0.99155	0.94956	0.92238
Galambos	0.83165	0.99155	0.94826	0.90070
Exact level	0.83165	0.99118	0.92735	0.85937
Correct c	2.1365	3.4027	2.5391	2.1867

TABLE 1b

Dunnett (1955, 1964) presented tables for the first problem presented here. The following values for c were taken from the 1964 article and thus should be exact.

t	2	5	5	9
n	3	3	5	4
d.f.	6	12	24	30
confidence level	0.95	0.95	0.99	0.95
c	2.86	2.90	3.40	2.86
	Lower Bounds on the Confidence Level			
1st Bonferroni	0.94240	0.93337	0.98821	0.93124
Sidak (Gamma 1)	0.94323	0.93512	0.98827	0.93331
1st Worsley	0.94982	0.94253	0.98908	0.93828
2nd Worsley	0.94982	0.94253	0.98908	0.93828
Gamma 2	0.94982	0.94326	0.98910	0.93955
Gamma 3	---	0.94724	0.98957	0.94351
Gamma 4	---	0.94915	0.98982	0.94615
Gamma 5	---	0.94989	0.98992	0.94793
	Upper Bounds on the Confidence Level			
2nd Bonferroni	0.94982	0.95627	0.99037	0.96290
Galambos	0.94982	0.95627	0.99037	0.96290
Exact level	0.94982	0.94989	0.98992	0.95050

TABLE 2

As given in Table lb, Dunnett (1955, 1964) presented tables for the first and second problem presented here. The following values for c were taken from the 1964 article and were corrected for unequal sample size as instructed in the article. These sample sizes were selected to provide allocations as given in Dunnett (i.e., $n_0/n_i \approx \sqrt{t}$) and thus should be exact for these cases even though the method described here is more general.

t	5	5	9	9
n_0, n_i	4, 2	10, 4	6, 2	6, 2
d.f.	8	24	14	14
confidence level	0.95	0.99	0.95	0.99
c	3.2000425	3.43876	3.1764266	4.0217
	Lower Bounds on the Confidence Level			
1st Bonferroni	0.93609	0.98307	0.93908	0.98879
Sidak (Gamma 1)	0.93765	0.98935	0.94070	0.98885
1st Worsley	0.94235	0.98958	0.94219	0.98917
2nd Worsley	0.94235	0.98958	0.94219	0.98917
Gamma 2	0.94311	0.98961	0.94332	0.98921
Gamma 3	0.94616	0.98978	0.94529	0.98949
Gamma 4	0.94776	0.98988	0.94675	0.98969
Gamma 5	0.94841	0.98993	0.94783	0.98985
	Upper Bounds on the Confidence Level			
2nd Bonferroni	0.95182	0.99000	0.95308	0.99050
Galambos	0.95182	0.99000	0.95308	0.99050
Exact level	0.94841	0.98993	0.94960	0.999011
c (from Games)	3.342	3.465	3.261	4.084

MEASURES OF JOINT INFLUENCE IN REGRESSION: ORTHOGONAL DECOMPOSITION, DISTRIBUTION, AN APPLICATION TO DATA ANALYSIS

Lillian Kingsbury, Villanova University Richard M. Heiberger, Temple University

ABSTRACT

Diagnostic measures of the joint influence of subsets of data points in regression are easy to define but have lacked an intuitive interpretation. Measures of influence for a single case are easy to interpret in terms of the position of the observation in the spaces spanned by the columns of the X matrix and the (orthogonal) residual column $E=Y-\hat{Y}$. We present an orthogonal decomposition of the joint influence measures in terms of the single case measures of an equivalent set of orthogonal pseudopoints. The decomposition allows an intuitive interpretation of the joint influence measures and leads to the derivation of the distribution of the measures under the usual assumptions. We illustrate the use of the decomposition in characterizing data configurations which contain influential points and in analysis of data.

1. Introduction with Example.

In recent years, methods to diagnose influential observations in regression analysis have received much attention in the Statistics literature. In data analysis we often encounter subsets of data points which are jointly influential, although not individually so. That is, simultaneous perturbation or deletion of all the cases in the subset leads to substantial changes in the estimated regression coefficients, although perturbation or deletion of single cases from the subset leads only to small changes in the results. This can be easily illustrated with the adaptive scores data, initially reported by Mickey, Dunn, and Clark (1967), and more recently analyzed in Cook and Weisberg (1980). These data contain at least three unusual points: cases 19, 2, and 18. Cases 2 and 18 are a highly influential pair, as evidenced by the change in the estimated regresion line due to deleting these two cases (see Figure 1).

Several diagnostic measures of joint influence have been developed (Cook and Weisberg 1982; Belsley, Kuh, and Welsch 1980) as generalizations of single-point diagnostics. The usual presentations of these measures, however, lack an intuitive interpretation that leads to an understanding of what kinds of arrangements of points are highly influential.

The joint influence of cases 2 and 18 in this data set is further illustrated by the individual and joint statistics for these two observations, summarized in Tables 1 and 2 below. From Table 1 we notice that, although the value of Cook's distance is highest for case 18, it is well below the "flag" value of 1.0. These cases are likely to go unnoticed on the basis of the individual case statistics.

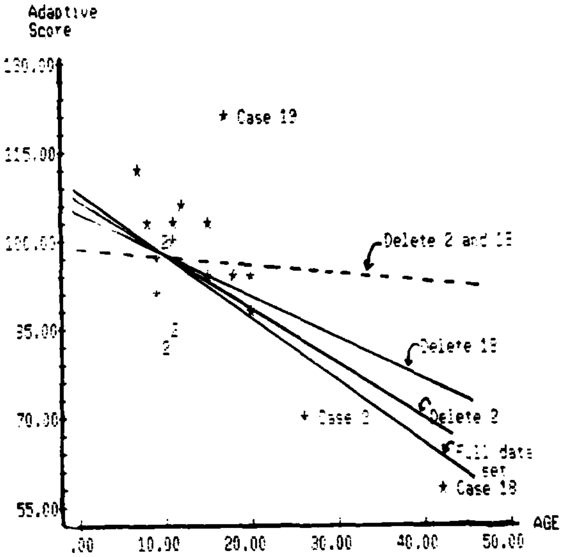


Figure 1. Scatterplot of the Adaptive Scores Data

Table 1
Case Statistics for Observations 2 and 18. Adaptive Scores Data. Full Data Set.

Case Number	h_{ii}	$(\underline{e})_i$	$d_{(i)}$	Cook's D_i
2	0.15	-9.57	-11.26	0.08
18	0.65	-5.54	-15.83	0.68

$s^* = 11.02$

Table 2
Case Statistics for Observations 2 and 18, Adaptive Scores Data. Reduced Data Sets.

	h_{ii}	$(\underline{e})_i$	$d_{(i)}$	D_i
Case 2 (delete case 18)	0.42	-14.37	- 24.78	1.02
Case 18 (delete case 2)	0.76	-9.96	- 37.33	4.30

Cook's D_I (Cases 2 and 18 jointly) = 6.37

The value of Cook's distance for each of these cases when the other one is deleted (Table 2) is much higher, further evidencing that these cases reinforce each other in their effect on the estimated regression line. There is no simple way to combine the individual case statistics to obtain the joint value of Cook's distance, D_I=6.37.

We present a decomposition of several joint influence measures as the sum of the case influence measures of a set of orthogonal pseudopoints. The pseudopoints are equivalent to the points under investigation in the OLS sense. That is, if the observations under investigation are replaced with the pseudopoints the results of the analysis remain the same. We continue the discussion of the adaptive scores data by giving the pseudopoints equivalent to cases 2 and 18 (labeled Z_1° and Z_2°), and the value of their case influence measures, in Table 3 below.

Table 3

Individual Case Statistics for Observations 2 and 18 and for Equivalent Pseudopoints.

Case	Wt.	X_0	X	Y	h_{ii}	e_i	$d_{(i)}$	D_i
18	1	1	42	57	0.65	-5.54	-15.82	0.079
2	1	1	26	71	0.15	-9.57	-11.26	0.67
Z_1°	1	1.331	49.08	81.85	0.794	-9.09	-44.12	6.365
Z_2°	1	0.478	5.58	39.89	0.012	-6.34	-6.36	0.002
Q_1	$(1.331)^2$	1	36.87	61.49	0.794	-6.83	-32.97	6.365
Q_2	$(0.478)^2$	1	11.67	83.45	0.012	-13.26	-13.54	0.002

The most important feature of Table 3 is that the values of Cook's distance for the pseudopoints, denoted Z_1° and Z_2°, add up correctly to 6.37. Also note that the large value of D_i for the pseudopoint Z_1° implies that this point would be identified by common, single row diagnostic techniques. Since the X_c column entries for the pseudopoints are not 1 these two points cannot be plotted on the same axes as the remaining observations. We have therefore appended two more rows to Table 3, labeled Q_1 and Q_2, also equivalent to the pseudopoints and to cases 2 and 18. The equivalence is in the sense that, if Q_1 and Q_2 replace cases 2 and 18 in the original data set and WLS is applied (with weights as indicated for Q_1 and Q_2, and weight 1 for the remaining cases) the same estimates are obtained. This algebraic identity, as well as the individual case statistics in Table 3, were verified by computation. The interpretive value of this approach is illustrated in Figure 2. This figure is a plot of the data with cases 2 and 18 replaced by Q_1 and Q_2. We can clearly see in this picture that Q_1, with weight $(1.331)^2$=1.9, would control the fit.

Subsets of data points which are jointly influential will usually be equivalent to a set of pseudopoints of which at least one is individually influential.

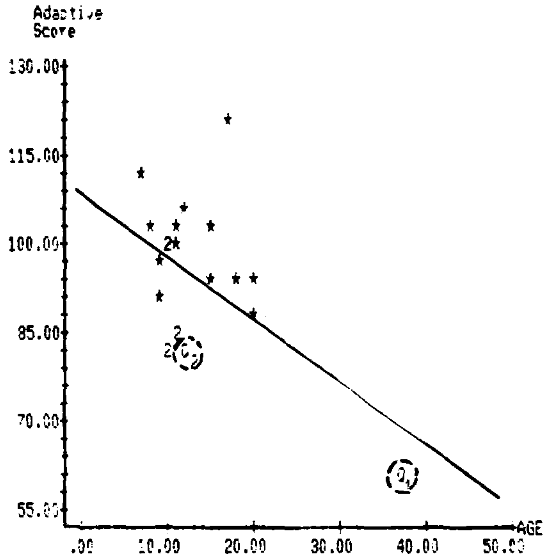


Figure 2. Scatterplot of Adaptive Scores Data. The equivalent Q_1 and Q_2 Pseudopoints Replacing Cases 2 and 18.

2. Notation.

We define an augmented data matrix, V*, of dimensionality (n+k)xp. The first k rows are assigned to the points whose joint influence is being investigated. We refer to these points as <u>keypoints</u>, and denote them by Z. The remaining data points, not under investigation, are the n rows of the reduced data matrix, V. In a data analysis situation we would permute the rows of the data to the appropriate position.

$$V^* = \begin{bmatrix} Z \\ V \end{bmatrix} = \begin{bmatrix} X_z & | & Y_z \\ X & & Y \end{bmatrix} = [X^* | Y^*] \quad (1)$$

We let $\hat{\beta}^*$ denote the regression estimator operating on the full data matrix, V*, and $\hat{\beta}$ denote the regression estimator operating on the data points in V. Therefore, using the notation introduced above, we have

$$\hat{\beta}^* = (X^{*\prime}X^*)^{-1}(X^{*\prime}Y^*) \qquad (2)$$

$$\hat{\beta} = (X'X)^{-1}(X'Y) \qquad (3)$$

The vector of residuals from OLS, $\underline{e}_{n+k}$, on the augmented data matrix, V*, is

$$\underline{e}_{n+k}' = (\underline{e}_k' \ \ \underline{e}_n') = (Y^* - X^*\hat{\beta}^*)'. \quad (4)$$

We define the vector of predicted residuals, $\underline{d}_{n+k}$, as residuals from the fit based on $\hat{\beta}$,

$$\underline{d}_{n+k}' = (\underline{d}_k' \ \ \underline{d}_n') = (Y^* - X^*\hat{\beta})' \quad (5)$$

We let s^{*2} denote the estimated residual variance from OLS in the augmented data matrix, V^*, and s^2 denote the estimated residual variance from OLS on V. That is,

$$s^{*2} = \frac{1}{(k+n-p)} \underline{e}_{k+n}' \underline{e}_{k+n} \qquad (6)$$

$$s^2 = \frac{1}{(n-p)} \underline{d}_n' \underline{d}_n \qquad (7)$$

Two frequently used matrices are:

$$G = X^*(X'X)^{-1}X^{*\prime} \qquad (8)$$

and

$$G_z = X_z(X'X)^{-1}X_z' = T\Theta T' \qquad (9)$$

The matrix G is an augmented hat matrix. Partitioning it as

$$G = \begin{bmatrix} G_z & G_Z' \\ G_Z & G_v \end{bmatrix} \qquad (\text{Note that } G_Z \cdot G_Z' = G_Z)$$

shows that $G_v = X(X'X)^{-1}X$ is the hat matrix for the data points in V. The elements of G_v and G_Z, g_{ij} for i=1 to n+k and j=1 to n, are interpretable as the rate of change in the i^{th} OLS fitted value based on $\hat{\beta}$, $\hat{y}_i = x_i\hat{\beta}$, with respect to y_j. The diagonal elements of G_z provide a measure of the distance, in X-space, between the keypoints and the centroid of the *remaining observations relative to a scale and* orientation determined by the remaining observations.

The reader can verify that

$$\underline{d}_k = (I+G_z)\underline{e}_k \qquad (11)$$

We stress that T, Θ and D are functions of the keypoints.

3. A Brief Review of Joint Influence Diagnostics.

Three approaches have been used in the development of multiple row diagnostics:

a. differencing of $\hat{\beta}$ with respect to the presence or absence of the keypoints (traditionally labeled "case-deletion"),

b. differentiation of $\hat{\beta}$ with respect to *weights assigned to the keypoints* (traditionally labeled "differentiation"),

c. ratio of data space volumes calculated in the presence and absence of the keypoints (traditionally labeled "geometric interpretation").

The differencing and ratio approaches are case deletion methods, the differentiation approach is a continuous form ofcase deletion.

After a healthy dose of algebra we have obtained general forms for measures based on each of these approaches in terms of the predicted residuals for the keypoints and the eigenvalue decomposition of G_z. These forms, as well as example of well-known diagnostics of each type are summarized in Table 4.

4. *Orthogonal Pseudopoints*

The consistent appearance of $T\underline{d}_k$ in the expressions in Table 4 suggested a data transformation as indicated below:

$$V^{*o} = \begin{bmatrix} T & 0 \\ 0 & I \end{bmatrix} \cdot \begin{bmatrix} Z \\ V \end{bmatrix} = \begin{bmatrix} Z^o \\ V \end{bmatrix} = \left[\begin{array}{c|c} X_z & Y_z \\ X & Y \end{array}\right]$$

This transformation leaves the reduced data matrix, V, unchanged and replaces the keypoints, Z, with a set of pseudopoints, $Z^o = TZ$. The pseudopoints, Z^o, are equivalent to the keypoints, Z, in the OLS sense. That is, the transformation does not change the results of the OLS analysis.

Table 4

Selected Joint Influence Diagnostic Measures

	Examples
Differencing:	
$C_k = (\hat{\beta}^*-\hat{\beta})'(X^{*\prime}X^*)(\hat{\beta}^*-\hat{\beta})$	$D_i = (1/(ps^{*2}))C_i$
$= \underline{d}_k'T\Theta(I+\Theta)^{-1}T'\underline{d}_k \qquad (13)$	
$W_k = (\hat{\beta}^*-\hat{\beta})'(X'X)(\hat{\beta}^*-\hat{\beta})$	MDFFIT = W_k
$= \underline{d}_k'T\Theta(I+\Theta)^{-2}T'\underline{d}_k \qquad (14)$	
Differentiation:	
$R_k = \left.\frac{(\partial(\hat{\beta}(w)))'}{\partial w}(X^{*\prime}X^*)\frac{(\partial(\hat{\beta}(w)))}{\partial w}\right\rvert_{w=1}$	BKW(1981,p.36)
$= \underline{d}_k'T\Theta(I+\Theta)^{-3}T'\underline{d}_k \qquad (15)$	
Ratio of Data Space Volumes:	
$L_k = \frac{\det(V^{*\prime}V^*)}{\det(V'V)}$	Andrews and Pregibon statistic
$= (1 + \frac{\underline{d}_k'T\Theta(I+\Theta)^{-1}T'\underline{d}_k}{\underline{d}_n'\underline{d}_n})\cdot\det(I+G_z) \qquad (16)$	

The pseudopoints, Z^o, are orthogonal with respect to both the $(X'X)^{-1}$ and $(X^{*\prime}X^*)^{-1}$ innerproducts. Also,

$$G_z^o = TG_zT' = D \qquad (17)$$

$$\underline{e}_k^o = T\underline{e}_k \qquad (18)$$

$$\underline{d}_k^o = T\underline{d}_k\,. \qquad (19)$$

Substitution of (17) to (19) into the equations in Table 4 leads to further re-expression of the joint influence diagnostics for the keypoints as weighted sums of the case influence diagnostics for the pseudopoints. These re-expressions are given in Table 5.

Table 5
Joint Influence Measures as Functions of the Orthogonal Pseudopoints

$$\left(\frac{1}{k\cdot s^2}\right)C_k = \left(\frac{1}{k\cdot s^2}\right)\sum \phi_i(1+\phi_i)^{-1}\cdot d_i^{\circ 2}$$

$$= \left(\frac{1}{k\cdot s^2}\right)\sum C^{\circ}_{1,i} \qquad (20)$$

$$\left(\frac{1}{k\cdot s^2}\right)W_k = \left(\frac{1}{k\cdot s^2}\right)\sum \phi_i(1+\phi_i)^{-2}\cdot d_i^{\circ 2}$$

$$= \left(\frac{1}{k\cdot s^2}\right)\sum W^{\circ}_{1,i} \qquad (21)$$

$$\left(\frac{1}{k\cdot s^2}\right)R_k = \left(\frac{1}{k\cdot s^2}\right)\sum \phi_i(1+\phi_i)^{-3}\cdot d_i^{\circ 2}$$

$$= \left(\frac{1}{k\cdot s^2}\right)\sum R^{\circ}_{1,i} \qquad (22)$$

$$\frac{(n-p')}{k} L_k = \frac{1}{k}\left(n-p' + \frac{\sum (1+\phi_i)^{-1} d_i^{\circ 2}}{s^2}\right)\prod (1+\phi_i) \qquad (23)$$

All sums and products are for i=1 to k.

The expressions given in Table 5 differ slightly from those given in Table 4 since they are rescaled (dividing by $k\cdot s^2$) to make them unitless and independent of the number of keypoints, k. These measures have several important properties:

- Each numerator is the sum of the single keypoint diagnostics for the k pseudopoints.
- Each numerator is the weighted sum of the single-point predicted residuals for the pseudopoints. The weights are simple functions of the augmented leverages of the pseudopoints. These measure the distance of the pseudopoints from the centroid of the initial points. The distinctions between the diagnostic measures are in the functional form of the weights.
- The weights are of the form $(\phi_i\cdot(1+\phi_i)^{-m})$, where the values of m generate a ladder of powers. It thus appears that the choice of crossproduct matrix, M, and the use of a differentiation vs a differencing approach, result in changing the power of the $(1+\phi_i)$ terms. This suggests that other values of m, not necessarily integers, may provide useful diagnostics. This needs to be studied further.
- The denominators of the measures define a scale based on only the initial data points.

It follows from the first two properties that a jointly influential set of points will typically have at least one equivalent pseudopoint that is individually influential.

5. The Distribution of Joint Influence Measures.

Under the usual normality and independence assumptions of regression analysis, the vector of residuals, $\underline{e}_{n+k}$ follows a normal distribution with mean zero, $\underline{0}$, and variance covariance matrix $(I-H)\sigma^2$. From equation (11) it follows that the vector of predicted residuals, $(\underline{d}_n, \underline{d}_k^{\circ})$, follows a normal distribution with mean zero, $\underline{0}$, and variance covariance matrix W, given by

$$W = \begin{bmatrix} I-G_1 & 2G_2T' \\ 2TG_2' & I+\Phi \end{bmatrix}. \qquad (24)$$

The numerators of each of the measures (13) through (15) in Table 4 are of the form $\underline{d}_k^{\circ\prime}M\underline{d}_k^{\circ}$, therefore each of these numerators can be written as the sum of k independent chi-squares with one degree of freedom. That is,

$$\underline{d}_k^{\circ\prime}M\underline{d}_k^{\circ} = \sum a_i \chi^2_i(1) \qquad (25)$$

where the a_i's are the non-zero eigenvalues of $M(I+\Phi)$ and $\chi^2(r)$ denotes a chi-square variable with r degrees of freedom. For measure (13) $M = \Phi(I+\Phi)^{-2}$ and $a_i = \phi_i$, for measure (14) $M = \Phi(I+\Phi)^{-3}$ and $a_i = \phi_i(1+\phi_i)^{-1}$, for measure (15) $M = \Phi(I+\Phi)^{-1}$ and $a_i = \phi_i(1+\phi_i)^{-2}$. Similarly, the denominator of each of these measures, $\underline{d}_n'\underline{d}_n$, is a chi-square with n-p degrees of freedom.

The measures (13) through (15) are appealing in that they are "F=like", however, since W, given in (24) is not block-diagonal, the numerator and the denominator are not independent.

Work by Gurland (1953) and others on the distribution of indefinite quadratic forms can be used to determine the cumulative distribution of measures of the form given in (13) through (15) in Table 4, as follows. We have

$$\Pr\left(\frac{\underline{d}_k'M\underline{d}_k}{\underline{d}_n'\underline{d}_n} > t\right) \qquad (26)$$

$$= \Pr\left(\frac{\underline{d}_{n+k}'M_1\underline{d}_{n+k}}{\underline{d}_{n+k}'M_2\underline{d}_{n+k}} > t \right)$$

$$= \Pr\left(\underline{d}_{n+k}'\{M_1 - t\cdot M_2\}\underline{d}_{n+k} > 0\right) \qquad (27)$$

where

$$M_1 = \begin{bmatrix} 0 & 0 \\ 0 & M \end{bmatrix}, \qquad (28)$$

and

$$M_2 = \begin{bmatrix} I_n & 0 \\ 0 & 0 \end{bmatrix}. \qquad (29)$$

The matrix $\{M_1 - t\cdot M_2\}$ is non-definite. The probability in (26) can be written, again following Baldessari (1967), as the

$$\Pr\left(\sum a_j \chi^2_j(r_j) > 0\right) \qquad (30)$$

where the a_j's are the s distinct eigenvalues of $W(M_1 - t\cdot M_2)$, the chi squares are mutually independent, and r_j is the multiplicity of a_j. Work along these lines, aimed at calculating the cumulative probability distributions of the measures in Table 4,is being pursued.

6. Bibliography

Andrews, D.F. and Pregibon, D. (1978), "Finding Outliers that Matter", J. Royal Statistical Society, Series B,40, 85-93.

Baldessari,B.,(1967) "The Distribution of a Quadratic Form of Normal Random Variables", Ann. Math. Stat., 1700-1704.

Belsley,D., Kuh, E. and Welsch,R. (1980), Regression Diagnostics, New York: John Wiley.

Cook, D.R. and Weisberg, S. (1980), "Characterizations of an empirical influence function for detecting influential cases in regression", Technometrics, 22, 495-508.

Cook, D.R. and Weisberg, S. (1982), Residuals and Influence in Regression, New York: Chapman and Hall.

Denby, L. and Larsen, W., (1977), "Robust Regression estimators compared via Monte Carlo", Communications in Statistics, A6(4), 335-362.

Gurland, J. (1953), "Distribution of Quadratic Forms and Ratios of Quadratic Forms", Ann. Math. Stat., 416-427.

Heiberger, R., Velleman, P., and Ypelaar, A. (1983), "Generating test data with independently controllable features for multivariate general linear forms", J. of the American Statistical Association, 78, 585-595.

Hill, R.W. (1977), "Robust regression when there are outliers in the carriers", Ph.D. Thesis, Department of Statistics, Harvard University.

Hill, R. and Holland, P. (1977), "Two robust alternatives to least squares regression". J. of American Statistical Association, 72, 828-833.

Hoaglin, D., Mosteller, F. and Tukey, J. (1983) Understanding Robust and Exploratory Data Analysis, New York:John Wiley and Sons.

Holland, P. and Welsch, R. (1977) "Robust regression using iteratively reweighted least squares", Communications in Statistics, A6(9), 813-827.

Huber, P. J. (1981), Robust Statistics, New York: John Wiley.

Kingsbury, L. and Heiberger, R. (1985), "Evaluating Robust Regression M-Estimators"' Proceedings of the ASA - Statistical Computing Section.

Velleman, P. and Ypelaar, J. (1980), "Constructing regressions with controlled features: A method of probing regression performance", J. American Statistical Association, 75, 839-844.

A GENERAL TECHNIQUE FOR GENERATING PSEUDO MULTIVARIATE RANDOM VARIABLES: A MARKOV PROCESS APPROACH

Subhash R. Lele and M. T. Boswell*, The Pennsylvania State University

Key Words and Phrases:
Multivariate Generators, Simulation

ABSTRACT

We present a very easy-to-program method of generating random samples from arbitrary discrete or continuous multivariate distributions. A Markov chain is constructed such that its equilibrium distribution is the desired distribution. The observations come from running the process until equilibrium is reached.

1. INTRODUCTION

Generating random samples from a given distribution is an essential part of any simulation study. There are many very general methods of generating random samples from univariate distributions. However, generalizing these techniques to multivariate distributions is difficult, and few algorithms have been developed. Hidden in applied fields and Monte Carlo literature is a very easy to program method of generating random samples from arbitrary multivariate distributions (see references 1,2,5,6,7 and 8). This method has been used to generate observations from certain spatial stochastic processes (Ripley 1979), although it has not been documented in its full generality with a careful proof in the statistical literature.

The idea behind this approach is to construct a discrete or continuous state space Markov chain such that its equilibrium distribution is the desired distribution. Each observation of the desired multivariate distribution is then an observation from the equilibrium distribution of this Markov chain.

2. THE BASIC METHOD

The method described below works for either a continuous or discrete distribution. For a continuous distribution the Markov chain is unusual in that the state space is continuous, but there are atoms of probability corresponding to no change. This requires a more complicated proof; the discrete case is a simpler case. The statements and proofs given below are for continuous distributions. Let $f(\underset{\sim}{r})$, $\underset{\sim}{r}$ in R^n be the probability density from which the observations are desired, and let $T(\underset{\sim}{r}|\underset{\sim}{s})$ be the transition probability distribution function from $\underset{\sim}{s}$ to $\underset{\sim}{r}$ of a Markov chain. The standard condition for such a chain to have $f(\underset{\sim}{r})$ as its equilibrium distribution is that

$$f(\underset{\sim}{r}) = \int..\int_{\underset{\sim}{s}\varepsilon S} f(\underset{\sim}{s})dT(\underset{\sim}{r}|\underset{\sim}{s}) \qquad (2.1)$$

where $\underset{\sim}{r}$ is fixed and integration is over $\underset{\sim}{s}$ in the state space S. For a continuous distribution, the distribution function for transitions from $\underset{\sim}{s}$ to $\underset{\sim}{r}$ has an atom of probability at $\underset{\sim}{s} = \underset{\sim}{r}$. In this case the distribution function for transitions from $\underset{\sim}{s}$ to $\underset{\sim}{r}$ satisfies

$$T(\underset{\sim}{r}|\underset{\sim}{s}) = p_{\underset{\sim}{s}}T_c(\underset{\sim}{r}|\underset{\sim}{s}) + (1-p_{\underset{\sim}{s}})\ I_{\underset{\sim}{s}}(\underset{\sim}{r})$$

where T_c is a continuous distribution function and $I_{\underset{\sim}{s}}(\underset{\sim}{r}) = 1$ if $R \geqq S_i$ for all i and $I_{\underset{\sim}{s}}(\underset{\sim}{r}) = 0$ otherwise.

Lemma 2.1. A sufficient condition for $f(\underset{\sim}{r})$ to be the equilibrium distribution of the Markov chain described above is that

$$t_c(\underset{\sim}{r}|\underset{\sim}{s})f(\underset{\sim}{s})p_{\underset{\sim}{s}} = t_c(\underset{\sim}{s}|\underset{\sim}{r})f(\underset{\sim}{r})p_{\underset{\sim}{r}} \qquad (2.2)$$

for all $\underset{\sim}{r}$ and $\underset{\sim}{s}$ in S, where

$$t_c(\underset{\sim}{r}|\underset{\sim}{s}) = \frac{d}{d\underset{\sim}{r}}\ T_c(\underset{\sim}{r}|\underset{\sim}{s})$$

Proof. Consider

$$\int...\int_{\underset{\sim}{s}\varepsilon S} f(\underset{\sim}{s})dT(\underset{\sim}{r}|\underset{\sim}{s})$$

$$= \int...\int_{\underset{\sim}{s}\varepsilon S} f(\underset{\sim}{s})t_c(\underset{\sim}{r}|\underset{\sim}{s})p_{\underset{\sim}{s}}d\underset{\sim}{s} + (1-p_{\underset{\sim}{r}})f(\underset{\sim}{r})$$

$$= \int...\int_{\underset{\sim}{s}\varepsilon S} f(\underset{\sim}{r})t_c(\underset{\sim}{s}|\underset{\sim}{r})p_{\underset{\sim}{r}}dS + (1-p_{\underset{\sim}{r}})f(\underset{\sim}{r})$$

$$=f(\underset{\sim}{r})$$

We will now create a Markov chain satisfying (2.2) for which the equilibrium distribution is the desired distribution. Suppose for each $\underset{\sim}{s}$ in S, it is easy to generate an observation $\underset{\sim}{r}$ with probability density $g(\underset{\sim}{r}|\underset{\sim}{s})$ which has a support containing S. We then generate an observation $\underset{\sim}{r}$ with density $g(\underset{\sim}{r}|\underset{\sim}{s})$ and move from $\underset{\sim}{s}$ to $\underset{\sim}{r}$ with probability $a(\underset{\sim}{r}|\underset{\sim}{s})$; otherwise we remain at $\underset{\sim}{s}$ with probability

$$1-\int...\int g(\underset{\sim}{r}|\underset{\sim}{s})a(\underset{\sim}{r}|\underset{\sim}{s})d\underset{\sim}{r} = 1-p_{\underset{\sim}{s}}$$

This is similar to a rejection technique; however, if the generated observation is not accepted, then the chain remains at its current location.

Theorem 2.1. For the processes described above, let

$$a(\underset{\sim}{r}|\underset{\sim}{s}) = \min\{q(\underset{\sim}{r}|\underset{\sim}{s}),1\}$$

where

$$q(\underset{\sim}{r}|\underset{\sim}{s}) = g(\underset{\sim}{s}|\underset{\sim}{r})f(\underset{\sim}{r})/[g(\underset{\sim}{r}|\underset{\sim}{s})f(\underset{\sim}{s})].$$

Then the resulting transition distribution for the Markov chain satisfy (2.2) and the equilibrium distribution is $f(\underset{\sim}{r})$.

Proof. Observe that $q(\underset{\sim}{r}|\underset{\sim}{s}) = 1/q(\underset{\sim}{s}|\underset{\sim}{r})$. Thus if $q(\underset{\sim}{r}|\underset{\sim}{s}) \geq 1$, then $q(\underset{\sim}{s}|\underset{\sim}{r}) \leq 1$. Consider, without loss of generality, the case where $q(\underset{\sim}{r}|\underset{\sim}{s}) \geq 1$. Then $a(\underset{\sim}{r}|\underset{\sim}{s}) = 1$ and $a(\underset{\sim}{s}|\underset{\sim}{r}) = q(\underset{\sim}{s}|\underset{\sim}{r})$. Also

$$t_c(\underset{\sim}{r}|\underset{\sim}{s}) = g(\underset{\sim}{r}|\underset{\sim}{s}).1/p_{\underset{\sim}{s}}$$

and

$$t_c(\underset{\sim}{s}|\underset{\sim}{r}) = g(\underset{\sim}{s}|\underset{\sim}{r}).q(\underset{\sim}{s}|\underset{\sim}{r})/p_{\underset{\sim}{r}} .$$

Thus

$$\begin{aligned} p_{\underset{\sim}{r}}t_c(\underset{\sim}{s}|\underset{\sim}{r})f(\underset{\sim}{r}) &= g(\underset{\sim}{s}|\underset{\sim}{r})q(\underset{\sim}{s}|\underset{\sim}{r})f(\underset{\sim}{r}) \\ &= g(\underset{\sim}{s}|\underset{\sim}{r}) \frac{g(\underset{\sim}{r}|\underset{\sim}{s})f(\underset{\sim}{s})}{g(\underset{\sim}{s}|\underset{\sim}{r})f(\underset{\sim}{r})} f(\underset{\sim}{r}) \\ &= g(\underset{\sim}{r}|\underset{\sim}{s})f(\underset{\sim}{s}) \\ &= p_{\underset{\sim}{s}}t_c(\underset{\sim}{r}|\underset{\sim}{s})f(\underset{\sim}{s}) \end{aligned}$$

The conclusion follows by Lemma 2.1.

3. THE ALGORITHM

The desired observation has density $f(\underset{\sim}{s})$. It is easy to generate observations with a density $g(\underset{\sim}{r}|\underset{\sim}{s})$ for each value of S. Equlibrium is assumed to be reached after NEQ number of steps of the Markov chain.

```
STEP0. Set COUNTER equal to 0.
STEP1. Generate an observation
       $\underset{\sim}{s}$ from some initial distribution.
STEP2. Generate an observation $\underset{\sim}{r}$ with density
       $g(\underset{\sim}{r}|\underset{\sim}{s})$
STEP3. Increment the COUNTER
STEP4. Calculate $q(\underset{\sim}{r}|\underset{\sim}{s})$
       $= g(\underset{\sim}{s}|\underset{\sim}{r})f(\underset{\sim}{r})/[g(\underset{\sim}{r}|\underset{\sim}{s})f(\underset{\sim}{s})]$
STEP5. If $q(\underset{\sim}{r}|\underset{\sim}{s}) \geq 1$ then set $\underset{\sim}{s} = \underset{\sim}{r}$ and
       go to step 8.
STEP6. Generate $U$ from the uniform
       distribution on $(0,1)$.
STEP7. If $U \leq q(\underset{\sim}{r}|\underset{\sim}{s})$ set $\underset{\sim}{s} = \underset{\sim}{r}$.
STEP8. If COUNTER $\leq$ NEQ then go to step 2.
STEP9. Deliver $\underset{\sim}{s}$
```

The resulting observation $\underset{\sim}{s}$ has density $f(\underset{\sim}{s})$ (approximately). To generate more observations the COUNTER is set equal to 0, but the algorithm is started again at Step 2.

4. PERFORMANCE OF THE ALGORITHM

In practice, the process is run a finite number of times to yield an observation which has approximately the equilibrium distribution. However, the rate of convergence of Markov chains is known to be geometric and therefore, only a few steps in the Markov chain will be necessary. We have tried running the process 5 to 15 times for the bivariate dirichlet, the bivariate binomial, the bivariate log-series and the bivariate normal distributions, and have found that ten observations are enough to reach equilibrium. The location (centering of $g(\underset{\sim}{r}|\underset{\sim}{s})$ is much more important than the shape. We have used a distribution $g(\underset{\sim}{r}|\underset{\sim}{s}) = \Pi g(R_i)$ of the product of independent random variables (not depending on $\underset{\sim}{s}$).

We used the algorithm to generate observations from the bivariate binomial, the bivariate log-series and the bivariate Dirichlet distributions. In each case we generated 10 sets of 1000 observations, and used the chi-square goodness-of-fit test. In every case the number of steps to reach equilibrium was taken to be 10. There were no samples rejected at a 5% level.

Additionally 10,000 observations were generated for each of these distributions, and the chi-square goodness-of-fit test was done. The results given below show the algorithm works quite well.

4.1 The Bivariate Binomial Distribution.

Let Y_1, X_2 and X_3 be independent identically distributed Bernoulli random variables with $p = 1/3$, and let $X = X_1 + X_2$ and let $Y = X_1 + X_3$. Then (X,Y) has a bivariate binomial distribution. We used the uniform distribution on the 7 possible values for the initial distribution as well as for $g(\underset{\sim}{r}|\underset{\sim}{s})$. The observed and (expected) frequencies for 10,000 observations are given in the following table:

x\y	0	1	2
0	2973 (2962)	1513 (1482)	---
1	1497 (1482)	2232 (2222)	680 (741)
2	---	734 (741)	371 (370)

The chi-square value is 5.98.

4.2. The Bivariate Log-Series Distribution.

This distribution is a discrete distribution with unbounded support. Its density is

$$P(X_1 = x_1, X_2 = x_2)$$

$$= \frac{\theta_1^{x_1} \theta_2^{x_2}}{-\ln(1-\theta_1-\theta_2)} \frac{(x_1 + x_2 - 1)!}{x_1!\, x_2!}, \quad x_1+x_2 \geq 1,$$

where $x_i = 0,1,\ldots;\ 0 < \theta_i < 1$ and $\theta_1 + \theta_2 < 1$. We generated observations from the bivariate log-series distribution with $\theta_1 = \theta_2 = 0.2$, using the product of independent Pascal distributions with parameters $k = 1$ and $p = 1-1/e$ for the initial distribution as well as for $g(\underset{\sim}{r}|\underset{\sim}{s})$. The observed and (expected) frequencies for 10,000 observations are given in the table below.

$x_1 \backslash x_2$	0	1	2	3	4	≥5
0	---	3994 (3915.2)	368 (391.5)	53 (52.2)	10 (7.83)	
1	3894 (3915.2)	791 (783.1)	174 (156.6)	27 (31.3)	1 (6.3)	
2	373 (391.5)	141 (156.6)	47 (46.9)	8 (12.5)		
3	43 (52.2)	29 (31.3)	13 (12.5)	9 (13.4)		
4	13 (7.8)	5 (6.3)				
≥5						7 (9.7)

The chi-square value is 22.49.

4.3 The Bivariate Dirichlet Distribution.

This distribution is continuous with bounded support. Its density function is

$$f(y_1,y_2) = c\, y_1^{\theta_1-1}\, y_2^{\theta_2-1}\, (1-y_1-y_2)^{\theta_3-1},$$

where $0 < y_i < 1$, $y_1 + y_2 < 1$, and $\theta_i > 0$. We generated observations from the bivariate Dirichlet distribution with $\theta_1 = \theta_2 = \theta_3 = 2.0$ the uniform distribution on the triangle bounded by $X = 0$, $Y = 0$ and $X + Y = 1$ for the initial distribution as well as for $g(\underset{\sim}{r}|\underset{\sim}{s})$. Five equiprobable strips with sides paralles to $X + Y = 1$ were divided into 10 equiprobable regions by the line $X = Y$. The observed and expected frequencies for these 10 regions are given below.

Region	1	2	3	4	5
observed frequency	1059	1012	1012	991	1003
expected frequency	1000	1000	1000	1000	1000

Region	6	7	8	9	10
observed frequency	937	976	999	978	1033
expected frequency	1000	1000	1000	1000	1000

The chi-square value is 9.98.

6. TIMINGS OF THE ALGORITHM

In order to have some idea of the efficiency of the algorithm we timed the algorithm used to generate observations from the bivariate normal distribution, and timed the generation of the bivariate normal distribution generated as a linear combination of independent normals each of which was generated using the ratio of uniforms technique (Kinderman and Monahan 1977). The uniform generator is a version of the generalized feedback shift register (GFSR) algorithm, Lewis and Payne (1973) The routines were written in FORTRAN and compiled using a MICROSOFT FORTRAN compiler on a ZENITH Z152 (Z100-PC) computer. These timings and the timings to generate 1000 observations of the three distributions, discussed in the previous sections, are given below.

TIMINGS TO GENERATE 1000 OBSERVATIONS (MIN;SEC)

Bivariate:	I	II	III
Normal(RU)	1;21.46	1;20.69	1;20.52
Normal(MC)	2;31.21	2;31.65	2;35.99
Binomial(MC)	0;47.35	0;47.24	0;49.10
Log-Series(MC)	1;58.20	1;59.07	2;08.63
Dirichlet(MC)	0;52.18	0;52.45	0;54.81

I: 1 call of 1000 observations
II: 100 calls of 10 observations
III: 1000 calls of 1 observation

7. COMMENTS AND DISCUSSION

(1). There were two problems that occurred during the use of this algorithm. Both are basically computational problems.

(a) If the initial distribution $g(\underset{\sim}{r}|\underset{\sim}{s})$ is too far away from the desired distribution, then a larger number of steps to reach equilibrium is needed. It is desirable to have $g(\underset{\sim}{r}|\underset{\sim}{s})$ and the desired distribution as close to each other as possible. The shape of $g(\underset{\sim}{r}|\underset{\sim}{s})$ is not as important as the location.

(b) While calculating $q(\underset{\sim}{r}|\underset{\sim}{s})$, we need to take the ratio of two quantities. In practice, if these quantities are too small or too large, the problem of overflow and underflow occurs. We suggest taking the natural logarithm of $q(\underset{\sim}{r}|\underset{\sim}{s})$ in this case. Thus we have $\ln a(R|S) = \min\{0, \ln q(\underset{\sim}{r}|\underset{\sim}{s})\}$ and then find $a(\underset{\sim}{r}|\underset{\sim}{s})$.

(2) To implement this algorithm we need to know the density only up to a constant.

8. ACKNOWLEDGEMENTS

The authors wish to thank Mr. Venkat Bhethanabotla for many useful discussions.

References:

(1) Hammersley, J. M., Handscomb, D. C. (1965). *Monte Carlo Methods.* Methuen.

(2) Hastings, W. K. (1970). Monte Carlo Sampling Methods Using Markov Chains and Their Applications. *Biometrika,* 57, 27-109.

(3) Kinderman, A. J. and Monahan (1977). Computer Generation of Random Variables Using the Ratio of Uniform Deviates. *ACM Trans. on Mathematical Software,* 3, 257-260.

(4) Lewis and Payne (1973). Generalized Feedback Shift Registrar Pseudorandom Number *Algorithm JACM,* 20(3), 456-468.

(5) Metropolis, Rosenbluth, Rosenbluth, Teller and Teller (1953). Equation of State Calculations by Fast Computing Machines. *Journal of Chem. Phy.*, 21(6), 1087-1092.

(6) Peskun, P. H. (1973). Optimal Monte Carlo Sampling Using Markov Chains. *Biometrika*, 60, 607-612.

(7) Rao, Pangali and Berne (1979). On Force Bias Monte Carlo Simulation of Water: Methodology, Optimisation and Comparison with Molecular Dynamics. *Molecular Physics*, 37(6), 1773-1798.

(8) Ripley, B. D. (1979). Simulating Spatial Patterns: Dependent Samples from a Multivariate Density. *Appl. Statist.*, 28, 109-112.

*Research carried out at the Center for Statistical Ecology and Environmental Statistics, Department of Statistics, The Pennsylvania State University with the partial support from the Northeast Fisheries Center, Woods Hole, MA. under Grant. NA 84-EAH-00011.

INFERENCE IN ROBUST DISCRIMINANT ANALYSIS

Vesna Lužar
University Computing Centre, Zagreb

In this paper a procedure for determining asymptotic distribution of the test statistic in a model for robust discriminant analysis has been proposed. The expressions, derived for the general class of elliptical populations, were found to be asimptotically distributed as χ^2. A Monte Carlo experiment has been conducted in order to study sampling distribution of the proposed statistic.

KEY WORDS: robust discriminant analysis; asymptotic distribution of covariances; elliptical distributions; Monte Carlo.

1. INTRODUCTION

A very simple model for Robust Discriminant Analysis (RDA) has been proposed by Štalec and Momirović (1984). The method is based on maximisation of variances of among group means on mutually orthogonal latent dimensions, defined in the space of standardized group means of a set of not necessarily normally distributed variables. Furthermore, for applying the method, the condition of regularity of initial variables need not be fulfilled.

Discriminant functions produced by the proposed algorithm are correlated, but for interpretational purposes can be transformed into orthogonal variables, using some factor analysis technique.

In this paper, a procedure for determining the number of significant discriminant functions, defined by RDA, is suggested. The expressions for test statistics are obtained for the general class of elliptical populations and for normally distributed variables, as a special case.

Finally, a Monte Carlo experiment was conducted in order to study the sampling distributions of the proposed statistics for normal and contaminated normal populations and for 3 different sample sizes.

2. MODEL OF ROBUST DISCRIMINANT ANALYSIS

Let

$$Z = (z_{ij}) \quad i = 1, \dots n; \quad j = 1, \dots m$$

be the data matrix in standard normal form, obtained by the description of the set of subjects $E = \{e_i,\ i=1, \dots n\}$ on the set of quantitative variables $V = \{v_j,\ j=1, \dots m\}$.

Let

$$S = (s_{ik}) \quad i = 1, \dots m; \quad k = 1, \dots g$$

be the selector matrix, obtained by the description of the set E on the nominal variable $N = \{n_k,\ k=1, \dots g\}$, where g denotes the number of groups.

By the operation

$$Q = S\ (S^T S)^{-1}\ S^T Z = PZ,$$

where P is a projector matrix, a matrix of standardized group means of variables from V, centered to common zero mean, is defined.

Matrix of covariances of variables in Q is:

$$G = Q^T Q/n = Z^T PZ/n,$$

where G is m x m matrix of rank q = min (g-1,m).

In order to obtain robust discriminant variables, covariances between linear composites

$$L_i = Qy_i \qquad (1)$$

and $\qquad i = 1,\ldots q$

$$K_i = Zy_i$$

have to be suceesivelly maximized:

$$\Omega_i = L_i^T K_i/n = y_i^T Gy_i = \max,$$

under the constraints of orthonormality of transformation vectors:

$$y_i^T y_i = 1$$
$$i, j = 1,\ldots q \qquad (2)$$
$$y_i^T y_j = \emptyset, \ i \neq j$$

It is easy to see, that this formulation leads to the characteristic equation:

$$G\,Y = Y\Omega \qquad (3)$$

where

$$Y = (y_1, y_2 \ldots y_q)$$

and

$$\Omega = \text{diag}\ (\Omega_1, \Omega_2 \ldots \Omega_\Omega),\ \Omega_1 > \Omega_2 > \ldots > \Omega_\Omega$$

are matrices of eigenvectors and nonzero eigenvalues, respectively.

Robust discriminant variables are given by:

$$K = Z\,Y,$$

with the covariance matrix:

$$U = K^T K/n = Y^T R\,Y \qquad (4)$$

which is not orthogonal, in general.

3. TESTING HYPOTHESES IN ROBUST DISCRIMINANT ANALYSIS

In order to test a series of hypotheses

$$H_{q-k}: \quad \tilde{\Omega}_{k+1} = 0$$
$$\tilde{\Omega}_{k+2} = 0$$
$$\vdots$$
$$\tilde{\Omega}_q = 0, \quad k = 1,\ldots q-1$$

where $\tilde{\Omega}_i$ i = 2,... q are the first q roots of the population matrix $\tilde{G}$, a modification of a procedure proposed by Steiger & Browne (1984) for testing correlations beetween optimal linear composites was applied. The procedure gives, in fact, the conditions under which optimal correlations can be tested in exactly the same manner as simple correlations by replacing original observations by optimal composite scores. Using the following, well known proposition, the procedure can be applied olso in the case of covariances and for the nonnormal populations:

Proposition 1 (Cook, (1951))

If $T = n^{1/2} (S - \Sigma)$, where S is the sample covariance matrix constructed from sample of N = n + 1 i.i.d. m-vectors $x_1 \ldots x_N$ with finite fourth moments, then the asymptotic distribution of T is normal with mean zero and covariances expressed in terms of cumulants of the distribution of x_i as

$$\psi_{ij,kh} = \text{cov}\ (t_{ij}, t_{kh}) = \kappa_{1111}^{ijkh} + \kappa_{11}^{ik}\ \kappa_{11}^{jh} + \kappa_{11}^{ih}\ \kappa_{11}^{jk}$$

In the case of elliptical populations, the expression for $\psi_{ij,kh}$ can be simplified using the standardized kurtosis parametar 3κ of the marginal distributions of X_i:

$$\psi_{ij,kh} = t_{ik} t_{jh} + t_{ih} t_{jk} + K\ (t_{ij} t_{kh} + t_{ik} t_{jh} + t_{ih} t_{jk}) \qquad (5)$$

For normal populations $\kappa = 0$ and covariances of the elements of T are given using only first two terms:

$$\psi_{ij,kh} = t_{ik}t_{jh} + t_{ih}t_{jk}$$

The requirements of proposition given by Steiger & Browne (1984) are satisfied if and only if the population vector $\tilde{y}_i$, $i = 1,\dots q$ is differentiable i.e. when the population root $\tilde{\Omega}_i$ $(i = 1,\dots q)$ is distinct. Is must be noted that the proposition is not applicable if all the roots of $\tilde{G}$ are zero. Since the distribution of the elements of the matrix G is known from proposition 1, asymptotic distribution of covariances between L = ZY and K = PZY can be found from the sample covariance matrix

$$C = \begin{bmatrix} L^T\ L/n & L^T\ K/n \\ K^T\ L/n & K^T\ K/n \end{bmatrix} = \begin{bmatrix} \Omega & \Omega \\ \Omega & U \end{bmatrix}$$

of

$$F = \begin{bmatrix} L \\ K \end{bmatrix} = \begin{bmatrix} Z\ Y \\ P\ Z\ Y \end{bmatrix} \quad .$$

From matrix C and proposition 1 with the use of some algebra, following result can be obtained:

PROPOSITTION 2

If $\underline{\Omega} = \text{vec}\ (\Omega_1, \Omega_2 .. \Omega_\Omega)$ is the vector of covariances between

$$L_i = Z\ y_i$$

and $\qquad$ (6)

$$K_i = P\ Z\ y_i, \quad i = 1,\dots q\ ,$$

where y_i are eigenvectors of G, and Z is data matrix from elliptical population, then the asymptotic distribution of

$n^{1/2}\ (\underline{\Omega} - \tilde{\underline{\Omega}})$, where $\tilde{\underline{\Omega}}$ is q x 1 vector of population covariances, is normal with mean zero and covariances given by q x q matrix ψ:

$$\psi = \psi_1 + \kappa\psi_2\ ,$$

where

$$\psi_1 = \Omega * U + \Omega * \Omega \quad (7) \quad (\Omega = \text{diag}(\Omega_1, \Omega_2 \dots \Omega_q))$$

$$\psi_2 = \psi_1 + \underline{\Omega}\underline{\Omega}^T\ , \qquad (8)$$

* denotes Hadamard's product, and 3 κ is the standardized kurtosis of the marginal distributions of vectors from Z.

Finaly, in order to test hypotesis H_{q-k}, a matrix

$$M_{q-k} = \begin{bmatrix} \emptyset & I_{q-k} \end{bmatrix}$$

is formed, where $\emptyset$ is q-k x k null matrix and I_{q-k} is (q-k) x (q-k) identity matrix. Using M_{q-k}, hypothesis H_{q-k} can be expressed in the equivalent form:

$$M_{q-k}\tilde{\underline{\Omega}} = \emptyset .$$

This, together with the proposition 2 leads to the final result:

PROPOSITTION 3

If $\underline{\Omega}$ and $\tilde{\underline{\Omega}}$ are q x 1 vectors of sample and population covariances between L_i and K_i as given by (6), then the asymptotic distribution of statistic

$$T_{q-k} = n\ (M_{q-k}\underline{\Omega})^T\ (M_{q-k}\ \hat{\psi}\ M_{q-k})^{-1}\ (M_{q-k}\underline{\Omega}), \quad (9)$$

where $\hat{\psi}$ is consistent estimate of ψ, is χ^2 with q-k degrees of freedom. In the case of normal populations, (9) can be expressed as:

$$T^N_{q-k} = n\ (M_{q-k}\underline{\Omega})^T\ (M_{q-k}\ \hat{\psi}_1\ M_{q-k})^{-1}\ (M_{q-k}\underline{\Omega})\ ,$$

where $\hat{\psi}_1$ is consistent estimate of ψ_1, as given by (7).

4. MONTE CARLO EXPERIMENT

In order to study the sampling distribution of the proposed test statistic, a Monte Carlo experiment was conducted for two types of elliptical distributions: normal and contaminated normal (ε =.1, σ = 3). Number of variables and number of groups were fixed to five and four, respectively. All four groups were of the same size. Three diferent sample sizes per group were used: n_i = 20, 50, 100. For each case, 200 replications were generated.

Model matrix of group means was:

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 & 2 \\ 3 & 3 & 3 & 3 & 3 \end{bmatrix}$$

The experiment was conducted through the use of package for generating random matrices developed by Lužar (1981).

Table 1 shows empirical type I error rates according to T_1^N and T_1, as compared to nominal rates α = .050, .025 and .01, for all 6 analysed cases. In general, empirical rates for α = .05 and α = .025 for normal populations are higher than the nominal ones. On the other hand, sampling distribution of T_1 statistic ajusted for nonzero kurtoses,is shorter tailed than the χ^2 (1) distribution. Deviations from the theoretical distribution could be due to truncation from below of the normal distribution of the smallest nonzero root Ω_q of the positive semidefinite matrix G. Unfortunately, distribution of the quadratic forms in truncated normal variables is not easy to obtain. Anyway, in using T_{q-k}^N (T_{q-k}) statistic in determining the number of significant robust discriminant variables, one should bear in mind that the sampling distribution in question is shorter tailed than the χ^2 distribution.

Olso, it should be pointed out, that additional experiments, with different model matrices should be conducted, in order to come to general conclusions concerning sampling distribution of T_{q-k}^N (T_{q-k}) statistic.

5. CONCLUSIONS

The intent in this paper has been to suggest the method for determining the number of significant discriminant variables in RDA. Proposed statistic, obtained for general class of elliptical populations, is asymptotically distributed as χ^2.

A Monte Carlo experiment brought to some conclusions concerning the sampling distribution of the proposed statistic.

TABLE 1. Number of times, out of 200, T_1^N (T_1) exceeded the α-th percentile of $\chi^2(1)$ (empirical type I error rate)

			α = .050	α = .025	α = .010
n_i=20	NORMAL	T_1^N	17 (.085)	7 (.025)	1 (.005)
	CONTAMINATED NORMAL (κ = 1.78)	T_1	7 (.035)	2 (.010)	0 (.000)
n_i=50	NORMAL	T_1^N	10 (.050)	3 (.015)	0 (.000)
	CONTAMINATED NORMAL (κ= 1.78)	T_1	7 (.035)	1 (.005)	0 (.000)
n_i=100	NORMAL	T_1^N	12 (.060)	7 (.035)	1 (.005)
	CONTAMINATED NORMAL (κ= 1.78)	T_1	5 (.025)	1 (.005)	0 (.000)

REFERENCES:

1. Cook, M. B. Bivariate k-statistics and cumulations of their joint sampling distribution. Biometrika, 38, 1951, 179-195.

2. Devlin, S. J., R. Granadesikan, J. R. Kettenring. Some multivariate applications of elliptical distributions, Essays in Prob. and Stat. S. Ikeda and others (ed.). 1976, 365-393.

3. Dobrić V., K. Momirović. An algoritam and program for robust discriminant analysis. In "Computer programs for classification, selection, programming and control of training, Institute of Kinesiology, Zagreb, 1984.

4. Lužar, V. A system of programs for generating pseudorandom matrices in multivariate analysis, Proc. Symp. "Jahorina 81", 1981, 359.1-359.9. (in croatian)

5. Muirhead R. J. Aspects of multivariate statwstical theory. J. Willey & Sons, 1982.

6. Steiger J. H., M. W. Browne. The comparison of interdependent correlation between optimal lineat composities. Psychometrika, Vol 49, No. 1, 1984, 11-24.

7. Steiger J. H., A. R. Hakstian. The asymptotic distribution of elements of a correlation matrix: Theory and application. British Journal of Math and Statist. Psych. 35, 1982, 208-215.

8. Štalec J., K. Momirović: On a very symple method for robust discriminant analysis. Unpublished manuscript, 1984.

EXACT CONFIDENCE BOUNDS FOR PROPORTIONS

Donald L. Marx, University of Alaska, Anchorage

INTRODUCTION

The construction of confidence interval estimates for proportions based on a random sample from an infinite population is one of the earliest inferential techniques learned by statistics users. The normal approximation is so well known that it's accuracy is seldom questioned for samples larger than Size 30. Charts and tables are available for constructing exact confidence bounds for smaller samples. A BASIC computer program for constructing exact confidence bounds is presented in this paper.

There are several advantages of computerizing the exact confidence bounds procedures. For example, any sample sizes and confidence levels can be considered; one is not restricted to only those values appearing on a chart or in a table. Inaccuracies inherent in reading graphical charts are avoided. There is less need to rely upon the approximate procedures based upon the standard normal distribution.

Following a brief discussion of accepted methods for constructing confidence bounds for proportions, this paper describes a computer algorithm for implementing the exact procedure. Inaccuracies of normal approximations are illustrated by several examples presented in the final section. In many cases the approximate procedures realize only one decimal accuracy. Very rarely is the accuracy greater than two decimal places for confidence levels of 95% or more and samples of up to 100 observations.

DISCUSSION

At a recent ASA Short Course on nonparametric statistics, Conover and Iman (1981) suggested three methods for constructing confidence intervals for proportions. Method A requires the use of specially designed charts, method B uses exact binomial distributions tables, and method C is based upon the normal approximation of the binomial distribution. Iman and Conover recommend that one use method A for 95% and 99% symmetric confidence intervals, use method B for other confidence levels and samples not greater than 20, and use method C for large samples.

The charts required for method A are available in Pearson and Hartley (1976). These charts were constructed for two confidence levels (95% and 99%) and selected samples sizes from 8 to 1000. Each confidence bound was constructed by calculating sufficient points (Θ, x) implicit in the cumulative binomial distribution function $F(x;n,\Theta)$ = constant to draw smooth curves for the upper (UB) and lower (LB) bounds when x 'successes' are observed in a sample of size n. As stated by the authors (p. 84) "The charts cannot and are not intended to provide very precise readings." Although graphical interpolation can be used to approximate bounds for sample sizes greater than eight that are not explicitly included in the charts, there is no way to use the charts for sample sizes less than eight.

Binomial probability mass function tables $f(x;n,\Theta)$ are readily available in elementary and intermediate textbooks and reference books. Method B requires that the cumulative binomial distribution function, $F(x;n,\Theta)$, for the observed number of 'successes' x in a sample of size n be provided for incremental values of Θ between 0 and 1. Typically interpolation on Θ is required to obtain UB and LB. For accurate determination of the bounds, $F(x;n,\Theta)$ must be available in very small increments of Θ . Adequate tables for method B are seldom found in books, but simple computer or programmable calculator routines can be easily developed for this purpose.

The construction of confidence intervals for proportions based upon the standard normal distribution is well known and widely used. This approximate procedure is based on the central limit theorem and estimating the standard deviation of the sample proportion in the implementation of the normal approximation. Many elementary textbook authors suggest this approximation is adequate for sample sizes greater than thirty. Fleiss (1981) documents two refinements of the normal approximation method. One refinement, which he recommends for sample proportions between 0.3 and 0.7, is the familiar continuity correction factor. The other, which Fleiss suggests for extreme sample proportions, includes the continuity correction factor and using the standard deviation of the sample proportion rather than its estimate in the normal approximation. The effect of the latter is that quadratic equations must be solved for the confidence bounds.

More central to the development of this paper is the distinction between Iman and Conover's methods B and C (and Fleiss' refinements to method C). Method B is called the exact method because it is based upon the true, exact sampling distribution for the sample proportion. Except for the cumbersomeness of implementing it, method B is preferred over any of the approximations. Method A, incidentally, is nothing more than an attempt to reduce the cumbersomeness of method B by using graphical charts. The difficulties of accurately reading charts notwithstanding, method A is properly described as an exact method for constructing confidence intervals for proportions.

DESCRIPTION OF ALGORITHM

The algorithm described here implements Conover and Iman's method B. It was coded in MBASIC to generate the example results below on the Osborne I computer. Suppose we wish to construct a $(1 - \alpha)$ 100% upper bound for Θ based on an observed sample result, say x 'successes' in n independent Bernoulli trials. Assume $0 < x < n$. An initial estimate for the upper confidence bound (UB) is constructed using Fleiss' quadratic function form of the normal approximation. Appropriate terms of the probability mass function, $f(x;n,UB)$, and cumulative distribution function, $F(x;n,UB)$, are computed for the binomial model with parameters n and UB. The exact confidence level for the upper bound UB is $[1 - F(x;n,UB)]100\%$. Based on the difference between the exact and desired confidence levels, $F(x,n,UB) - \alpha$, the value for the upper bound is revised, and the above procedure is repeated. The procedure continues to be repeated until either the difference between the exact and desired confidence levels is acceptably small or an established maximum number of iterations is executed. Default values are 0.00001 for the accuracy specification and 10 iterations.

The $(1 - \alpha)100\%$ lower bound is constructed in a similar fashion. Fleiss' quadratic form of the normal approximation provides the initial estimate LB. $f(x-1;n,LB)$ and $F(x-1;n,LB)$ are computed. The exact confidence level for the lower bound LB is $F(x-1;n,LB)100\%$. Based upon the difference $F(x-1;n,LB) - (1-\alpha)$, the value for the lower bound is revised, and the process is repeated as indicated in the previous paragraph.

The binomial probability mass function $f(x;n,\Theta)$, for given sample data, n and x, is constructed using the recursive relationship

$$f(0;n,\Theta) = (1 - \Theta)^n$$

$$f(k;n,\Theta) = \frac{(n-k+1)\Theta}{k(1-\Theta)} f(k-1;n,\Theta),\ k=1,2,\ldots,x.$$

The cumulative distribution function $F(x;n,\Theta)$ is constructed using

$$F(0;n,\Theta) = f(0;n,\Theta)$$

$$F(k;n,\Theta)=F(k-1;n,\Theta) + f(k;n,\Theta),\ k=1,2,\ldots,x.$$

Revision of the boundary estimate in the iterative procedure described above is accomplished in either of two ways. The first way uses the first two terms of the Taylor's series expansion for $F(x;n,\Theta)$ about the current value of the bound, say B. That is, the revised value for the bound is

$$B - \frac{1 - B}{(nB - x)f(x;n,B)} [\alpha - F(x;n,B)]$$

The second way, which is implemented only if the first way produces a revised value that is outside the interval *(0,1)*, is the *simple average* of B and the interval boundary that is exceeded by the first way. That is, B/2 is used if the revised value above is less than or equal to zero, and (1+B)/2 is used if the revised value above is greater than or equal to one.

The normal approximations used to obtain estimates for confidence boundaries require standard normal quantiles. Quantiles, accurate to four decimal places, from the standard normal distribution for selected probabilities are contained in the computer code. Twenty value for tail probabilities from 0.0001 to 0.5 were selected. Linear interpolation is implemented for intermediate probability values.

Efficiency and accuracy in computing exact confidence levels for current values of bounds are enhanced by taking advantage of the complimentarity of variables in the binomial model. That is, if random variable X has the binomial distribution with parameters n and Θ, then n-X is binomial with parameters n and $1-\Theta$. The computations of $f(x;n,\Theta)$ and $F(x;n,\Theta)$ are implemented so that the iterative formulas above are taken from zero to x or n-x, whichever is smaller. Additional *efficiency is accomplished by handling the cases* x = 0 and x = n separately from the general case $0 < x < n$. Symmetric confidence intervals do not exist for these special cases. When x = 0, the exact $(1 - \alpha)100\%$ upper bound is $\alpha^{1/n}$ and there is no $(1 - \alpha)100\%$ lower bound. When x = n, the exact $(1 - \alpha)100\%$ lower bound is $(1 - \alpha)^{1/n}$ and there is no $(1 - \alpha)100\%$ upper bound.

In the MBASIC implementation on the Osborne I, the algorithm produces exact confidence bounds, either symmetric (when they exist) or one sided as specified by the user, for specified confidence levels from 50% to 99.99% and samples of up to 126 observations. Accuracy is within the specified default value (0.0001) for each bound; consequently the guaranteed accuracy of symmetric confidence intervals is 0.00002. Larger sample sizes can be accomodated, but the accuracy cannot be guaranteed due to possible rounding problems. If the iterations limit is reached, the algorithm displays the normal approximation bound and the message: MAXIMUM ITERATIONS COMPLETED. If the sample is too large to accurately implement the iterative formulas for $f(x;n,\Theta)$, the normal approximation is displayed with the message:

SAMPLE SIZE TOO LARGE OR SAMPLE PROPORTION TOO NEAR 1/2 TO COMPUTE EXACT BINOMIAL DISTRIBUTION

For large samples and proportions near 1/2, the initial term $f(0;n,\Theta)$ in the iterative formulas underflows and is set to zero.

EXAMPLES

A specific sample result is considered first to illustrate the operation of the algorithm. Attention is then focused on the inaccuracies of the approximate procedures. Inaccuracies in confidence bounds estimators for the example introduced in the following paragraph and several additional example are discussed.

Suppose we want to construct a symmetric 99% confidence interval estimate for the proportion of 'successes' in an infinite population based on a random sample of size 20. Let random variable Y denote the number of 'successes' in random samples of size 20 from the population. Then Y has the binomial distribution with unknown parameter Θ and n = 20. For Y observed 'successes', the upper bound, UB, is the value of Θ such that $Prob(Y \leq y) = 0.005$; the lower bound, LB, is the value of Θ such that $Prob(Y \geq y) = 0.005$. UB is given implicitly in terms of the cumulative distribution function as $F(y;20,UB) = 0.005$. LB is implicit in $F(y-1;20,LB) = .995$.

Now suppose that six 'successes' are observed; i.e. y = 6. The MBASIC algorithm implements an iterative procedure to solve for UB. The search is initiated by approximating UB with 0.6064, Fleiss' quadratic equation approximation. $F(6;20,\Theta)$ where $\Theta = 0.6064$ is calculated and compared with 0.005. If the difference is within the specified tolerance, the search terminates. Otherwise, the value for Θ is revised using the first two terms of the Taylor's series expansion for Θ. $F(6;20,\Theta)$ using the revised value of Θ is calculated and again compared with 0.005. The search continues until the tolerance specification is satisfied. The result is UB = 0.6096. Fleiss' approximation is correct to two decimal places. The tail probability, $F(6;20,0.6064)$, beyond Fleiss' UB is

0.0055 rather than the desired value, 0.005. The search for LB is similar. LB is given implicitly by $F(5;20,\theta) = .995$. Fleiss' method gives LB = 0.1013; the solution given by the MBASIC algorithm is LB = 0.0846. Fleiss' LB is accurate to only one decimal place. The tail probability, $1-F(5;20,0.1013) = 0.0119$, is more than double the desired value. Table I lists 99% symmetric confidence limits for this example using the exact method as well as the various normal approximations. Tail probabilities for the various bounds estimators are also listed.

Exact and both simple and quadratic approximations for 99% confidence intervals were constructed for sample results from x = 1 'success' to x = 10 'successes'. The simple approximation produced negative lower bounds estimates for samples with five or fewer 'successes'. That is, in terms of the sample proportion $\hat{p}$, lower bounds when $\hat{p} \leq 0.25$ are negative. Only one decimal place accuracy is achieved for $0.30 \leq \hat{p} \leq 0.45$. The greatest accuracy achieved is only two decimal places when $\hat{p} = 0.50$. Similar inaccuracies are realized using the quadratic approximation except that negative lower bounds are averted. Only one decimal accuracy is achieved for $\hat{p} \leq 0.45$, and the greatest achieved accuracy is only two decimal places when $\hat{p} = 0.50$.

Additional examples were considered using samples of sizes 30, 50, and 100 observations. Exact and both simple and quadratic approximations for 95% and 99% confidence interval bounds were calculated. Achieved accuracies for the simple approximation method are summarized in Figures I and II. Using the simple approximation for 99% confidence level bounds, only one decimal place accuracy is achieved for samples of 100 observations and $\hat{p} \leq 0.27$. The greatest accuracy of two decimal places is realized when $\hat{p} \geq 0.30$. Reducing the confidence level to 95% provides little improvement in the accuracy of the simple approximation bounds; two decimal accuracy is realized for $\hat{p} \geq 0.27$.

TABLE I
99% Confidence Interval for θ Based
on 6 'successes' in 20 Trials
(Tail probability beyond bound)

Method	LB	UB
Exact	0.0846 (0.0050)	0.6096 (0.0050)
Iman & Conover's Method C	0.0361 (0.0001)	0.5639 (0.0157)
Method C with cont. corr.	0.0611 (0.0010)	0.5889 (0.0086)
Method C with quadratic eqn.	0.1013 (0.0119)	0.6064 (0.0055)

The quadratic approximation method for 99% confidence interval bounds provides two decimal place accuracy for samples of size 50 and $\hat{p} \geq 0.16$ and for all samples of size 100. (See Figure III.) When the confidence level is reduced to 95% (Figure IV), at least two decimal place accuracy is realized everywhere except for $\hat{p} \leq 0.67$ and samples of size 30. Three decimal accuracy is achieved for $\hat{p} \geq 0.42$ and samples of 100 observations.

CONCLUSIONS

The accuracy of confidence interval estimators for proportions based upon the simple normal approximation are limited to two decimal places for samples up to size 100 and confidence levels of at least 95%. Two decimal place accuracy is the greatest achievable, and in many cases only one decimal place accuracy is realized. The quadratic approximation provides some improvement, but the accuracy is still limited to two decimal place accuracy except for sample proportions very near one half and sample sizes of at least 100 observations.

The alternative to using the approximate procedures is to construct exact confidence intervals for proportions. This paper describes an implementation of the exact procedure in an interactive computer algorithm. Programmed in MBASIC code on the Osborne I computer, five place accuracy is realized for samples of up to 126 observations.

REFERENCES

Conover, W.J. and R.I. Iman, Nonparametric Statistics, notes for a short course presented at the annual meeting of the American Statistical Association at Detroit, August 8-9, 1981.

Fleiss, J.L., Statistical Methods for Rates and Proportions, 2nd ed., John Wiley and Sons, 1981.

Pearson, E.S. and H.O. Hartley, Biometrika Tables for Statisticians, Vol. 1, 2nd ed., Charles Griffin & Co., 1976.

FIGURE I

ACCURACY OF SIMPLE APPROXIMATION FOR
99% CONFIDENCE INTERVALS FOR PROPORTIONS

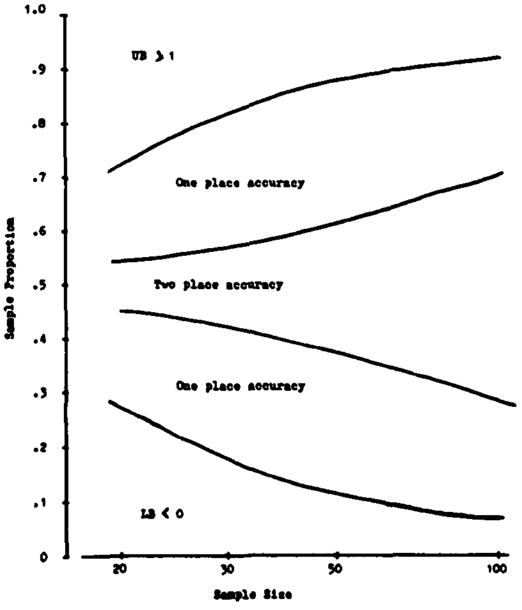


FIGURE II

ACCURACY OF SIMPLE APPROXIMATION FOR
95% CONFIDENCE INTERVALS FOR PROPORTIONS

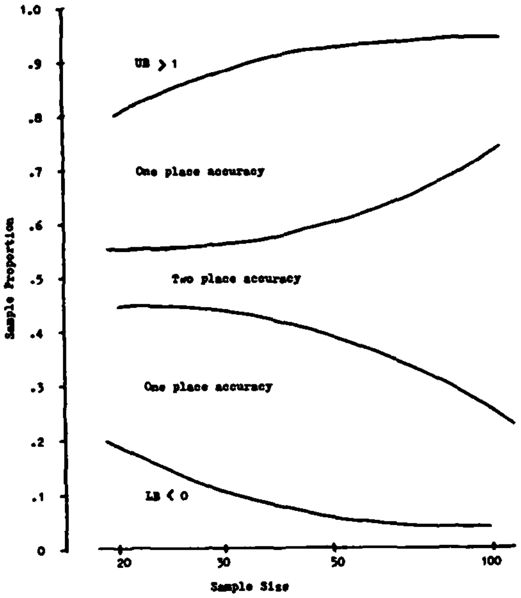


FIGURE III

ACCURACY OF QUADRATIC APPROXIMATION FOR
99% CONFIDENCE INTERVALS FOR PROPORTIONS

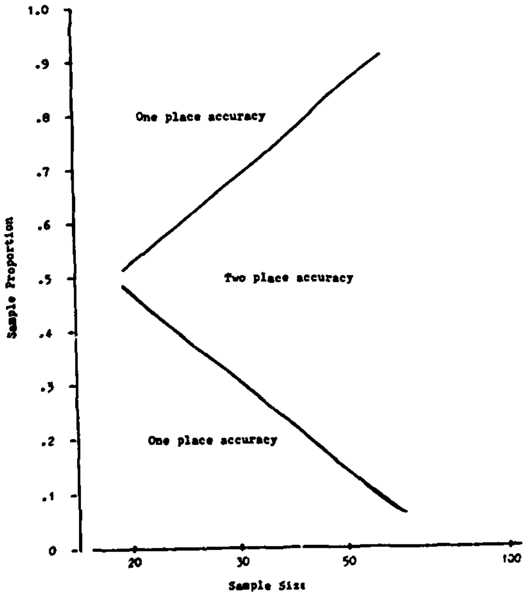
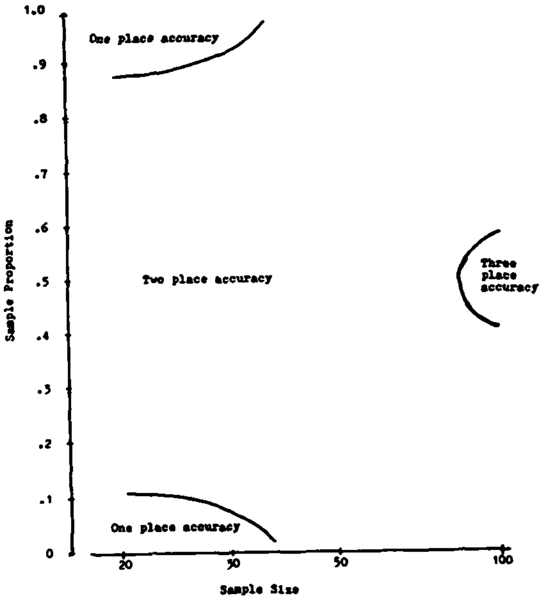


FIGURE IV

ACCURACY OF QUADRATIC APPROXIMATION FOR
95% CONFIDENCE INTERVALS FOR PROPORTIONS



TUNING COMPUTER SYSTEMS FOR MAXIMUM PERFORMANCE : A STATISTICAL APPROACH.

William A. Nazaret
William J. Klingler

AT&T Bell Laboratories
Holmdel, New Jersey 07733

In this paper we discuss a statistical approach to the problem of setting the tunable parameters of an operating system to minimize response time for interactive tasks. This approach is applicable to both tuning and benchmarking of computer systems. It is based on the use of statistical experimental design techniques and represents an inexpensive, systematic alternative to traditional ways of improving the response of systems. We illustrate the method by means of experiments performed on VAX* minicomputers running under the UNIX* System V Operating System.

1. INTRODUCTION.

The work described in this paper has been motivated by two different, but related problems, that arise in the analysis of computer performance. The first is how to set an operating system's tunable parameters to achieve the best response time for interactive tasks, given the computer's load conditions. The second is how to map the relationship among the different tunable parameters of the system and their impact on response time.

Although the second problem appears to be a generalization of the first in practice the two appear in different contexts. The first problem is normally confronted by system administrators in their attempt to get the most performance out of the system on behalf of the users. In contrast, the second question is tackled mostly by system designers and performance analysts who are charged with modeling system performance under a variety of loads before the systems are actually handed to the customers. This activity is sometimes called "benchmarking".

The system administrator goal is to optimize the response for important tasks in his/her organization under the particular load conditions Therefore we could say imposed on the system by the users. that this problem is "local" by nature. On the other hand, the responsibility of the system designer and performance analyst is to understand how the system reacts to changes in the tunable parameters for each of many different loads that are likely to be encountered on the field. In this sense the problem is rather "global".

A very important consequence of this distinction is that the measurements used for benchmarking are usually made under "simulated" loads that are designed to exercise the system in a given manner and on which the experimenter has total control. Tuning, instead, is done using data generated by the actual load of users on the system. This type of load is not under the complete control of the administrator or the experimenter. Despite the above differences, tuning and benchmarking have something in common : the necessity to experiment with different settings for the parameters in search for a configuration that yields the best results.

In this paper we present a systematic, cost effective approach to conducting these experiments. This approach makes use of statistical techniques to design experiments which yield, in many cases, information nearly equivalent to the one obtained by performing a complete exhaustive test. Our approach although not new to statisticians is becoming popular among systems managers and performance analysts as an alternative to more traditional methods of experimentation.

Throughout the paper we use the UNIX* operating system as an example of a tunable operating system. However, the method is applicable to any operating system (or system in general) which allows the user the freedom to adjust its operating characteristics. In Section Two we present an overview of UNIX tunable parameters and their potential impact on system's response. Section Three introduces the experimental problem by describing three experiments carried out on VAX 780, 785 and 8600 machines respectively. Section Four explains our statistical strategy to estimate the effect of the parameters on response time for certain tasks. In Section Five we analyze the results of the experiments and show the improvement achieved after adjustment of the parameters according to these results. Finally, in Section Six a critique of our approach is given together with some extensions.

2. *WHAT DOES TUNING A COMPUTER MEAN ?.*

Tuning a computer system is, in principle, not very different from tuning any system in general. It amounts to finding the setting of certain parameters to satisfy performance requirements. Hence, the performance requirements determine which settings are "optimal" for an application. For instance, of a typical passenger car will be tuned to optimize fuel consumption and reduce emissions. In contrast, a racing car will be tuned to maximize speed at the expense of fuel economy.

The UNIX system allows fine tuning by giving the administrator the freedom to set the values of some kernel parameters at boot time. Additionally, one can exercise control over options that are not part of the kernel. Some of them relate to the hardware and some to software. Examples of these parameters and their significance for the run time environment are :

A. **System Buffers :** These are chunks of physical memory, typically 1024 kilobytes in size, which are used by the operating system to keep recently used data in hope that it might be used shortly afterwards. Increasing the number of these buffers improves the "hit ratio" on this cache up to a point. An excessive amount of these buffers will hurt performance for it takes away memory space from the users.

B. **Sticky Processes :** There is a bit associated with the permissions on an executable file that will cause its text segment to be stored in contiguous blocks on the swapping device. Commands which are frequently invoked (specially those with large images) ought to have the sticky bit set so that every time they are invoked their code can be brought into memory as easily as possible by the system. Systems in which this is not done usually suffer from chronic disk I/O bottleneck and the resulting degradation ın response time. The number and kind of commands with the sticky bit set is a tunable parameter.

C. **Paging Daemon Parameters :** In virtual memory implementations of UNIX, memory used by processes is assigned on a per-page basis. A page is just a piece of the code usually 512 or 1024 Kilobytes long. The paging daemon is a system process whose responsibility is to free up memory by reclaiming space occupied by pages which are no longer in use. A process can also be stripped of its pages if its total CPU time exceeds a given value. How often the daemon runs, how many simultaneous active pages a process can have and the maximum CPU time quota before a process is swapped out are tunable parameters.

D. **File System Organization :** This is a highly installation dependent parameter. The idea is to distribute the system and user files among the available disks in a way that the load on each of them is approximately the same. When one of the disks is overloaded I/O waits increase and response is degraded.

E. **CPU Assist Devices :** Some types of hardware allow the possibility of adding coprocessors or add-on boxes to relieve the CPU of mundane chores. Some examples are terminal I/O assist devices and troff coprocessors. The use and number of such devices can be subjected to tuning.

F. **Main Memory :** By allowing minor changes in the amount of physical memory on the system it is possible to detect whether increasing memory size will help to enhance the performance of the system. This is helpful to know before committing any resources into buying the additional boards.

3. *THREE CASE STUDIES.*

To illustrate the methodology we will describe three experiments carried out on VAX* 780, 785 and 8600 systems respectively, running under the UNIX System V Operating System at the Quality Assurance Center of AT & T Bell Laboratories.

The first of the three was conducted with almost an entirely tuning orientation, and it is somewhat similar to the one reported in [1]. Our goals were to improve response on a system whose performance was becoming unbearable and to overcome some questionable aspects of the experience in [1]. Among these aspects we targeted :

- Duration of the experiments : The experiment described in [1] lasted more then three months. We believe that any approach to tuning which takes this long to produce results has very limited practical value. Therefore, one of our goals was to find ways to obtain useful results in reasonable amount of time.
- Stationarity of the load : A second goal was to ensure that the load conditions were reasonably stationary. We wanted to avoid a situation in which it was not possible to detect whether improvements in performance were due to the tuning or rather due to a decreased level of load on the system (this is essentially what happened in [1]).

The tunable parameters considered in this experiment were : file system organization, memory size, system buffers, sticky processes, KMC's (terminal I/O processors) and PDQ's (troff coprocessors).

The second experiment on a VAX 785 system included only 5 of the 6 parameters considered in the previous one. Sticky processes were dropped out of consideration because we already had considerable prior knowledge about how to handle them. The experiment was designed to allow us to estimate, in addition to the main effects of the factors, some of the interactions among them. Therefore it resulted in a larger number of trials. Our intentions here went beyond tuning the system. We also wanted to assess the merits of fancier (more expensive) design plans relative to simple plans like the one used in Experiment One. The precise meaning of this will be given in next section when we discuss the experimental strategy.

The third experiment differed from the previous ones in a very important characteristic. The load used was not a "live" load but rather a simulated load built using a strategy to imitate the essentials aspects of the type of load we expected this system to be subjected to. The factors considered were the same as in the second experiment except that KMC's were not included for they are not necessary in this new VAX 8600 model. By using a simulated load, the experimental conditions were completely under our control and thus we expected to get more statistically reliable results. However, there was no previous experience in using our method in this context and little was known about the validity of the conclusions under actual load conditions. What we learned from this is discussed in Section Six.

4. *THE EXPERIMENTAL STRATEGY.*

Before basic planning for the experiment can be done, we need to choose the levels at which each of the factors is going to be tried. Since the set of factors considered for Experiment One (VAX-780) contain the ones for the other two systems it will suffice to describe the levels chosen in that case. Figure 1 shows the levels for each one of the six factors.

In Figure 1 the amount of system buffers space allocated depends on the total size of memory. Hence, Low, Medium and High represent a different fraction of memory for each of the three memory sizes. We determined the amount of memory assigned to these buffers using a formula of the form

$$Sysbuff = C_L + .2 \cdot K_L$$

where C_L is 1.0, 1.2 and 1.4 Megabytes when L is Low, Medium and High respectively. Similarly, K_L is 0,1 or 2.

Figure 1

VAX-780

	LEVELS		
FACTORS	**1**	**2**	**3**
File System Distribution	**A**	**B**	**C**
KMC's	**NO**	**YES**	**-**
Memory Size	**6 Mb**	**7 Mb**	**8 Mb**
System Buffers	**Low**	**Medium**	**High**
Sticky Processes	**10**	**20**	**30**
PDQ's	**None**	**One**	**Two**

The choice of levels for the factors is highly installation dependent and must be done taking into account both the characteristics of the load and prior knowledge about how changes in these parameters are supposed to affect the system's response ([4] is an excellent reference for this). A common strategy starts from the current settings and introduces some variations around them. The size of this variation ranges from modest to large. Minor variations defeat the purpose of the experiment. One exception to this strategy occurs when the current setting of one of the parameters is clearly wrong (non-optimal). In Figure 1 we have such an example. The choice of setting for sticky processes was to set the 10, 20 and 30 most popular commands. The number of sticky processes before the experiment was about 40 but they were not chosen according to frequency of invocation. We therefore ignored them.

Next we had to decide on the measures to assess the performance of the system under the different experimental conditions. A common choice is the time the system needs to execute a script containing tasks important to the organization . For instance, in a text processing organization such a script would consist of formatting a document. The four measures we typically use are : trivial time, edit time, troff time and c-compile time. Among those only whose name is not self-explanatory is "trivial" time. This is just the response time for a command that involves no interaction between the user and the system (e.g. the "date" command). It gives a measure of instantaneous response time.

Deciding on a sampling plan for the experiment is a crucial and difficult task. As we noted above, we wanted to reduce the total time for the experiment, as much as possible, without compromising the integrity of the methodology. After careful study of the load in the VAX-780 and VAX-785 systems we decided that it was safe to use a day as the basic duration of a run. A day is a natural unit because it allows you to set up the system from one run to the next during off-hours therefore sparing the users any inconvenience. This choice is also minimal in the sense that to have the runs last less than a day will force you to interfere with the normal functioning of the system. More importantly, it will make comparisons among runs invalid due to the within day variations of the load (peak hours). We must note that a day may even troublesome in some installations were the behavior of the load depends, say, on the day of the week.

In the third of the experiments a whole run took only about an hour as opposed to a whole day. This, of course, was due to the use of a simulated load.

Response times were measured at evenly spaced intervals during the run. It is important not to oversample since the load caused by the timing programs and the timed scripts could interfere negatively with the users.

The most important aspect of the whole experimental strategy is the choice of combinations to be tried during the experiment. A complete exhaustive search will most likely give us the right answer. However, it is not hard to realize that the time this would take is prohibitive. For instance, from Figure 1 we gather that it will take about 729 days to run an all-combinations experiment !. Even in the case of the third experiment (with simulated loads) the administrative overhead is overwhelming. It requires re-booting the system 81 times !. Our strategy is to run a fraction of all possible runs following an array of combinations that allows us to test all factors simultaneously in a fair way. Arrays with this property are documented extensively in the statistical literature (See [2] and [3]). For instance, the design used in Experiment 1 (VAX 780) is given by Figure 2. It was constructed using an orthogonal array known as the L_{18}, consisting of only 18 runs.

For the second experiment we used another orthogonal array known as the L_{27} (see [2]), consisting of 27 runs. The increased size of the experiment, as we mentioned previously, was deliberatedly planned to allow us to estimate together with the main effects, the interactions between memory size and the other four factors. Finally, the controlled experiment (VAX 8600) with only four factors was run following a plan based on a fraction of a 3^4 array consisting of just 9 runs.

The advantages of using design plans like the ones above are :

- They provide an average picture over the whole parameter space.
- The estimate of the effect of any of the factors is orthogonal with respect to those of the other factors.

FIGURE 2

VAX 780

Run #	File Sys	KMC's	Memory Size	Buffers Space	Sticky Process	PDQ's
1	A	No	7 Mb	1.4	10	None
2	A	No	6 Mb	1.0	20	One
3	A	No	8 Mb	1.8	30	Two
4	A	Yes	7 Mb	1.2	30	One
5	A	Yes	6 Mb	1.4	10	Two
6	A	Yes	8 Mb	1.6	20	None
7	B	No	7 Mb	1.4	20	One
8	B	No	6 Mb	1.0	30	Two
9	B	No	8 Mb	1.8	10	None
10	B	Yes	7 Mb	1.6	30	None
11	B	Yes	6 Mb	1.2	10	One
12	B	Yes	8 Mb	1.4	20	Two
13	C	No	7 Mb	1.6	10	Two
14	C	No	6 Mb	1.2	20	None
15	C	No	8 Mb	1.4	30	One
16	C	Yes	7 Mb	1.2	20	Two
17	C	Yes	6 Mb	1.6	30	None
18	C	Yes	8 Mb	1.6	10	One

- Under certain conditions they yield information approximately equivalent to what you would obtain by using a much larger experiment.

There is of course a price to pay for this. These plans achieve the reduction in size of the experiment by deliberatedly confounding the main effect of the factors with the "joint" effect or interaction of some of the other factors. Therefore, they rely on the size of the main effects to be dominant.

In spite of the above we advocate a strategy based on choosing a highly fractioned array because, at the very least, it provides a most inexpensive starting point from which we can always obtain very useful information about the parameters. In particular, these experiments lend themselves to be extended, if necessary, to allow the estimation of higher order effects (interactions) if the data suggest they might be important. In practice, we have seldomly had to go pass on iteration in this cycle. In most cases the information provided by the data is such that the improvement achieved by the predicted optimal setting (as verified by a confirmatory run) makes the extra effort involved in conducting additional experiments unattractive.

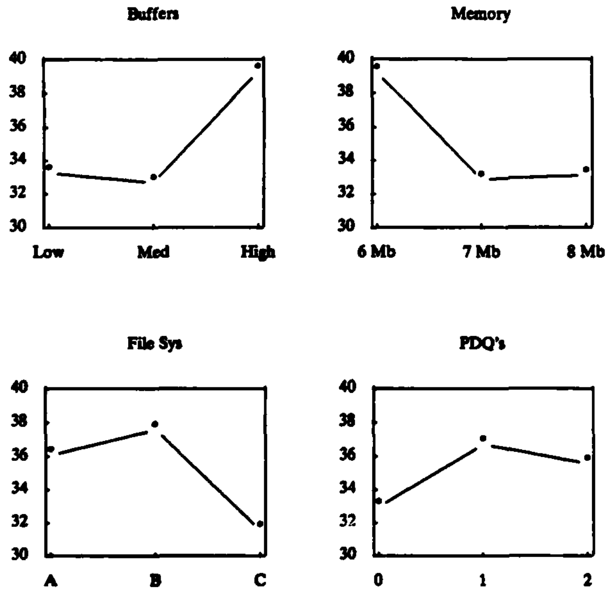
5. *DATA ANALYSIS*.

Due to space constraints we can not present summaries and analyses of the data for each of the three experiments. We can however show a selected subset of plots summarizing the information provided by the data with respect to how changes in the factors affect performance. We will also show plots to illustrate what was achieved by re-setting the parameters to the levels suggested by the experimental data.

Figure 3 shows the estimated effects that each of the four parameters in Experiment number 3 (VAX 8600) had on c-compile time. The performance measure in this case was mean response time. Another sensible choice would be mean square average. This latter criterion has the advantage of picking the setting that minimizes a sum of the variability and the square of the average. The time scale is seconds.

FIGURE 3

C-compile



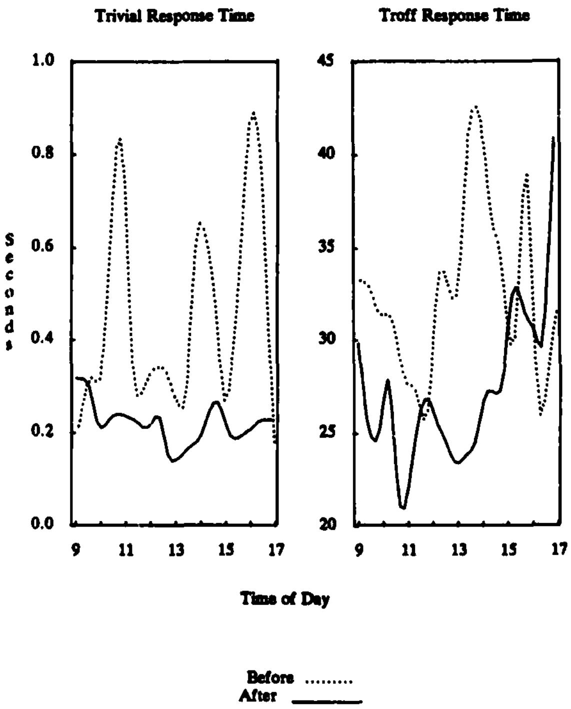
A summary of the conclusions extracted from these data is :

- The machine should be run at its current level of 8 Megabytes, the gradient information in Figure 3 does not suggest additional gains if another 4 Megabytes are added (memory can be bought in 4 Mb units).
- The number of system buffers can be set to low which means about 1.4 Megabytes of memory for the system and the rest for the users.
- File system organization C is advantageous over either A or B.
- No Pdq's should be used.

These conclusions are valid only for this particular performance measure and for this system. There is no reason why they have to hold for another response measure (like edit time) even under the same load. For instance, one could argue that not adding PDQ's to the system could hurt text processing performance and this could very well be the case. In general the answer depends on the relative level of capability of the processor to handle the load. As a matter of fact, to our surprise we have seen cases in which adding PDQ's hurts "troff" response time. An explanation to this puzzling event can be found in the fact that the VAX processor is several times more powerful that the microprocessor which drives the PDQ's. Therefore every time a text processing job is sent to the PDQ when the CPU could indeed had handled it, a loss in performance is sure to occur. Indeed ,only in the first experiment the results suggested the inclusion of one or more PDQ's. For this system the load was so heavy, that there was little change in troff performance by adding or excluding the PDQ's, while there was a positive effect on edit and trivial response times upon adding them.

Figure 4

COMPARISON OF RESPONSE TIMES



A way to check the gains in performance after tuning is to run back to back confirmatory runs under the old and new settings. The results for one of the three systems is given in Figure 4. In the graphs the dotted curve represents the response time before the experiment and the solid curve represents the response after the tuning. We see that both trivial and response times were reduced considerably after tuning the system.

6. CONCLUSIONS

We were able to reduce response time for several typical tasks in all three systems. For the first one, in average we reduced response by 38%, an important gain in a system that was considered hopeless. In the second system, we additionally discover that blind use of PDQ's could lead to loss in performance for text processing jobs. The cvaluation of the results for the third system is still under way. Early data seem to indicate that the configuration recommended by the simulated load experiment enables the machine to handle the real load rather easily.

We also have a much better assessment of the real usefulness of these experiments for both tuning and benchmarking. The results of back to back confirmatory runs (a week each) showing substantial reductions in response time for both of the tuning experiments, indicate that it is possible to use this approach successfully for periodical system tuning. We can not however claim that it will succeed in general. Instead we can say that as long as attention is confined to only a few factors (therefore keeping the duration short) and the load is relatively stable, the method will help to run your system better. The experience with simulated loads convinced us that the usefulness of this approach in benchmarking studies is even greater. In fact, we are currently using our approach to find the functional relation between the parameters of the UNIX System V Virtual Memory Management scheme and response time.

Finally, in response to questions about the dangers involved in ignoring the interaction among some of the factors we would like to point out that, yes, there is a risk. However, even in cases were we had (upon analysis of the data) second thoughts about the absence of such interactions, the improvements over the previous system configuration achieved by using information generated during the experiments made the risk worth taking.

* UNIX is a trademark of AT&T Bell Laboratories

* VAX is a trademark of Digital Equipment Corporation

REFERENCES

[1] Pao, T.W., Phadke, M.S. and Sherrerd C.S., "Computer Response Time Optimization Using Orthogonal Array Experiments", IEEE International Communications Conference, Chicago IL, 1985. Conference Record, Vol. 2, pp 890-895.

[2] Taguchi, G and Wu, Y, OFF-LINE QUALITY CONTROL (1979). Central Japan Quality Control Association.

[3] Box, G.E.P, Hunter, W and Hunter, J.S, STATISTICS FOR EXPERIMENTERS (1978). John Wiley and Sons.

[4] UNIX System V, Tuning and Configuration Guide. AT & T, Information Systems.

FITTING PARAMETRIC AND SEMI-PARAMETRIC PROBIT FORMS WITH NON-ZERO BACKGROUND

Haiganoush K. Preisler, USDA-Forest Service,Berkeley

__1.Introduction:__ In a commonly used quantal response bioassay model for toxicological experiments, the binomial response probabilty p is set to $p=\pi+(1-\pi)F(\eta)$,where $0\leqslant\pi<1$ is a *parameter corresponding to the proportion of* background responses (natural mortality), F is the probit (logit or some other mathematical) function and η is a smooth function of covariates. Parametric probit regression models are obtained by substituting a known parametric function of the covariates for η . Estimation in the parametric probit regression models with zero background (i.e. $\pi=0$) can be handled directly by the GLIM statistical package (Baker and Nelder,1978). Models with $\pi>0$ require special treatment because they do not fit within the framework of generalized linear model of GLIM. Hasselblad et al. (1980) describe a fitting procedure for this model that employs the EM-algorithm. Cox (1984) uses the derivative free program BMDPAR and a short FORTRAN program to obtain the estimates. Russell et al. (1977) have written a FORTRAN program specifically to handle the above model with $\pi\geqslant0$. It produces maximum likelihood estimates of the parameters and performs tests of parallelism and equality among treatment groups. In section 2 of this paper I demonstrate how the *GLIM package can be* modified to allow the fitting of the parametric model with non-zero background.

An estimation procedure for the non-parametric model, with η an unspecified smooth function of the covariates and π zero, is discussed in Hastie and Tibshirani (1985). In sections 4 and 5 of this paper I present an estimation procedure for the semi-parametric case were η is an unspecified function and π is an unkown parameter. The procedure utilizes the specific ACE algorithm of Breiman and Friedman (1985). It produces estimates of the functions of the covariates that minimize a weighted residual error criterion. Uses of these procedures are illustrated by examples from insecticide bioassay studies. The GLIM macros employed to perform the parametric estimation are listed in the appendix. The semi-parametric fitting was done via ACE implemented within S (Becker and Chambers 1984).

2. Parametric probit with background.

Consider the fitting of the model

$$E(y_{ij})=\mu_{ij}=\pi_i+(1-\pi_i)F(x_{ij}\beta)u_{ij} \quad . \quad . \quad . \quad (2.1)$$

where y_{ij}, $i=1,..,I,j=1,...,J_i$, is the proportion of responses out of a total of n_{ij}. β is a p-vector of parameters, x_{ij} is a p-vector of covariates and u_{ij} is an indicator variable that is set to zero for the background responses and to one otherwise. If we assume the number of responses to be binomial, then

$$var(y_{ij})=\sigma^2_{ij}=\mu_{ij}(1-\mu_{ij})/n_{ij}$$

and y_{ij} may be written as $\mu_{ij}+\sigma_{ij}e$, with e an error variate.

The algorithm used by GLIM for a generalized linear model with a link function $\eta=g(\mu)$ is as follows, for a given current estimate η_o of the linear predictor, regress

$$\eta_o+(y-\mu_o)(d\eta/d\mu)_o$$

on the vector of covariates x with weights defined by $w_o{}^{1}=(d\eta/d\mu)^2\sigma^2$

In order to use the GLIM algorithms for the model in (2.1) with unknown π the fo!lowing procedure may be followed. Linearize μ_{ij} around a previous estimate to obtain

$$y_{ij}=\hat{\mu}_{ij}+(\pi_i-\hat{\pi}_i)\left[1-F(x_{ij}\hat{\beta})u_{ij}\right]$$

$$+(\beta-\hat{\beta})'(1-\hat{\pi}_i)x_{ij}u_{ij}f(x_{ij}\hat{\beta})+\hat{\sigma}_{ij}e$$

Here f is the derivative of F. Next, rearrange the terms in the equation above and set $f_{\bullet}(x\beta)=uf(x\beta)+(1-u)$ and $m=(1-\pi)f_{\bullet}(x\beta)$ to obtain

$$\hat{\pi}_i m_{ij}+\hat{\beta}'x_{ij}u_{ij}+ (y_{ij}-\hat{\mu}_{ij})/(1-\hat{\pi}_i)f_{\bullet}(x_{ij}\hat{\beta})=$$

$$\pi_i m_{ij}+\beta'x_{ij}u_{ij}+ \left[\sigma^2_{ij}/(1-\pi_i)f_{\bullet}(x_{ij}\beta)\right] e \; .$$

Thus, by defining the linear predictor with a new explanatory variable m, i.e. $\eta=\pi m+\beta' xu$, and setting $(d\eta/d\mu)=1/(1-\pi)f_*$, the GLIM algorithms can be used for estimating the parameters π and β. Since m is a function of the fitted values, it must be recalculated at each iteration. Scallan (1982) uses a similar technique to find maximum likelihood estimates for some other models with 'parametric' link functions such as the hyperbolic curve with $E(y)=(x+A)/(\alpha+\beta x)$ where A is the extra parameter to be estimated.

3.A toxicological example with non-zero background. Data from an insecticide bioassay experiment is analyzed in this section. The experiment consisted of treating samples of male and female larvae from a particular insect population with 2 different chemicals. A fixed amount of insecticide was applied to the surface skin (topical application) of each insect, (Robertson and Kimball,1979). A control group was treated with solvent only. Mortality was tallied after 7 days.

A number of probit models were fitted to the data using the GLIM package in conjunction with the macros in the appendix. All models included one covariate, the logarithm of the concentration of insecticide used. The most general model included separate background rates, intercepts and slopes for each sex and treatment while the simplest model included the same (common) background with the same intercept and slope for each sex and treatment (table 1). The normal quantiles of the data, corrected for background using the common estimate, are plotted in figure 1 along with the fitted model B.

Table 1

Fits of various models to the insect data

Model	scaled deviance	d.f.
A. Common π, slope and intercept	118	21
B. Common π, slope, separate intercepts	23	18
C. Common π, separate slopes, intercepts	21	15
D. Separate π's, slopes and intercepts	17	12

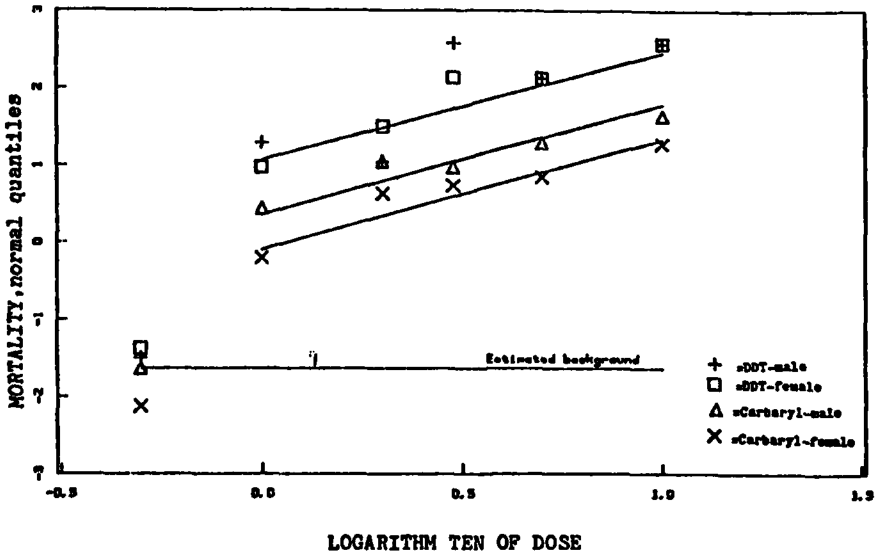


Figure 1. Mortality of larvae treated with insecticides and the fitted probit lines for model B.

4. Non-parametric probit predictor. In the non-parametric probit model $E(Y)=F(\eta)$, where Y is a binary response variable, F is the probit function, $\eta=\sum t_k(x_k)$ and $t_1,\dots,t_p$ are arbitrary smooth functions of the covariates, x_k. We would like to estimate the functions t that minimize the weighted square residual of the probit regression

$$S^2=\sum_{i=1}^{n}\left[y_i-F(\eta_i)\right]^2 w_i \qquad (4.1)$$

where $w=1/F(\eta)(1-F(\eta))$. This can be accomplished as follows. Given initial estimates η_o, (4.1) can be approximated by

$$\sum_{i=1}^{n}\left[y_i-F(\eta_{io})-f(\eta_{io})\sum_{k=1}^{p}\Delta(x_k)\right]^2 w_{io}$$

$$=\sum_{i=1}^{n}\left[(y_i-F(\eta_{io}))/f(\eta_{io})-\sum_{k=1}^{p}\Delta_k(x_k)\right]^2 w^*_{io} \qquad \dots(4.2)$$

where f is the derivative of F. Where $\sum\Delta=\eta_1-\eta_o$ is the correction needed to update the estimate of η, and where $w^*=wf^2(\eta)$.

Now the problem is reduced to calculating Δ_k's that minimize (4.2). This may be accomplished by using the ACE (Alternating Conditional Expectation) algorithm of Breiman and Friedman (1985) with the adjusted variable $z=(y-F)/f$ as the dependent variable fitted by a linear transformation and with weights given by wf^2.

It is to be noted that the values of the corrections $\Delta_1,\dots\Delta_k$ as evaluated by ACE are scaled to have mean zero and variance one. In order to obtain the updated estimates of η the coefficients $a,b_1,\dots,b_p$ in the equation

$$\eta_1=\eta_o+a+\sum_{k=1}^{p}b_k\Delta_k(x_k)$$

need to be calculated. This is done by regressing the adjusted variables z on $\Delta_1,\dots\Delta_p$.

5. Semi-parametric probit model. This model extends the one discussed in section 4 in that the extra parameter π for the background response is included. The mean square error that is minimized in this case is

$$\sum_{i=1}^{n}\left[y_i-\pi-(1-\pi)F(\eta_i)\right]^2 w_i \quad \dots \quad (5.1)$$

where y, F, η and w are as defined before.

For a given set of weights, w_{io}, and probit predictors, η_{io}, minimizing (5.1) with respect to π yields

$$\hat{\pi}=\frac{\sum_{i=1}^{n}\left[(y_i-F(\eta_{io}))(1-F(\eta_{io}))w_{io}\right]}{\sum_{i=1}^{n}(1-F(\eta_{io}))^2 w_{io}.} \qquad (5.2)$$

The next step is to minimize (5.1) with respect to $t_1(x_1),\dots t_p(x_p)$ given π. This is the same as minimizing the function

$$\sum_{i=1}^{n}\left[y^c_i-F(\eta_i)\right]^2 w_i \quad \dots \quad (5.3)$$

where now $y^c_i=(y_i-\hat{\pi})/(1-\hat{\pi})$.

The function (5.3) is of the same form as (4.1), and so the same procedure can be used to obtain estimates for $t_1,\dots,t_p$ once π is calculated using (5.2). One could then proceed by iterating back and forth between determining π and determining η.

6. A toxicological example with two covariates. Robertson et al. (1981) presented a group of experiments that tested the effects of weight on the response of the western spruce budworm (Choristoneura occidentalis Freeman) to insecticides. The data analyzed in this section is from one of these experiments wherein each insect was weighted and then treated with a fixed concentration of DDT. The response of each insect to the chemical was recorded after 7 days with the response variable $y_i=1$ if the insect was dead by the seventh day and $y_i=0$ otherwise. Plots of the binary response data versus the two covariates, dose per weight and weight, are shown in figure 2.

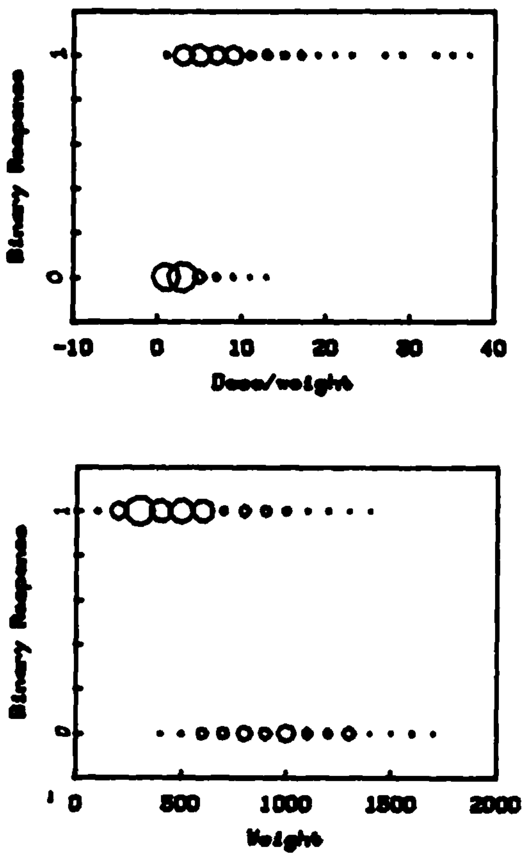


Figure 2. Mortality of insects, treated with DDT, versus dose per weight and weight. The area of the circles is proportional to the number of observation.

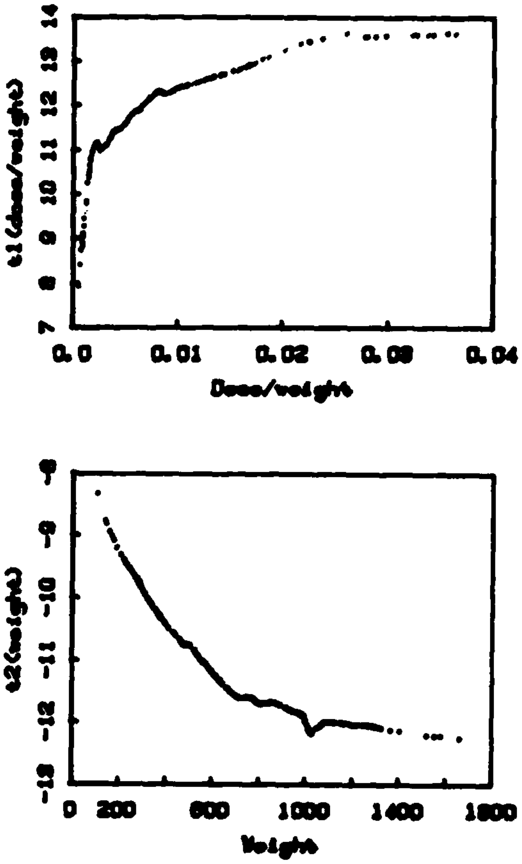


Figure3. Non-parametric transformations of covariates for *DDT data*.

The objectives of this study were two-fold. The first was to determine whether insects respond to toxicants in proportion to their body weight. The second was to decide on a mathematical form for the predictor η as a function of dose and weight. The plots in figure 3 are the transformed variables, $t_1(x_1)$ and $t_2(x_2)$ versus the original variables x_1=dose per weight and x_2=weight. The initial values used were obtained by fitting a parametric probit model using the GLIM package. The plots in figure 4 are the transformed variables versus the logarithms of the dose per weight and weight respectively. The nearly linear shapes of these graphs suggest that a logarithmic transformation might be appropriate for these variables. In order to determine whether both variables are needed in the model we also ran a probit regression, using GLIM, with the transformed variables as covariates. The resulting scaled deviances for the models with the nonparametric functions and the model with a parametric predictor using a logarithmic transformation are given in table 2.

The large and significant decrease in the deviance between models (A) and (B) is a substantial indication that the weight covariate is needed in the model. The similarity in the deviances for models (B) and (C) is an indication that the logarithmic transformation is most probably the appropriate transformation to be used for these covariates.

In conclusion, it was found that the commonly used logarithmic transformation remains appropriate when consideration is extended to a broad family of transformations.

AD-A191 296 COMPUTER SCIENCE AND STATISTICS PROCEEDINGS OF THE 5/5
SYMPOSIUM ON THE INTER..(U) COLORADO STATE UNIV FORT
COLLINS T J BOARDMAN 26 AUG 87 AFOSR-TR-88-0153
UNCLASSIFIED AFOSR-86-0070 F/G 12/6 NL

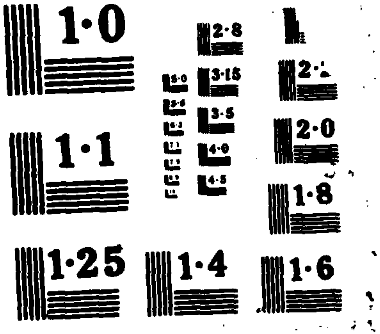

1·0
2·8
2·0
1·1
1·8
1·25
1·4
1·6

Table 2

Comparison of parametric and non-parametric fits of the model $F(\eta(x_1,x_2))$ with x_1=dose per weight and x_2=weight.

η	scaled deviance	d.f.
A. $\beta_o+\beta_1 t_1(x_1)$	228	257
B. $\beta_o+\beta_1 t_1(x_1)+\beta_2 t_2(x_2)$	177	256
C. $\beta_o+\beta_1 \log(x_1)+\beta_2 \log(x_2)$	184	256

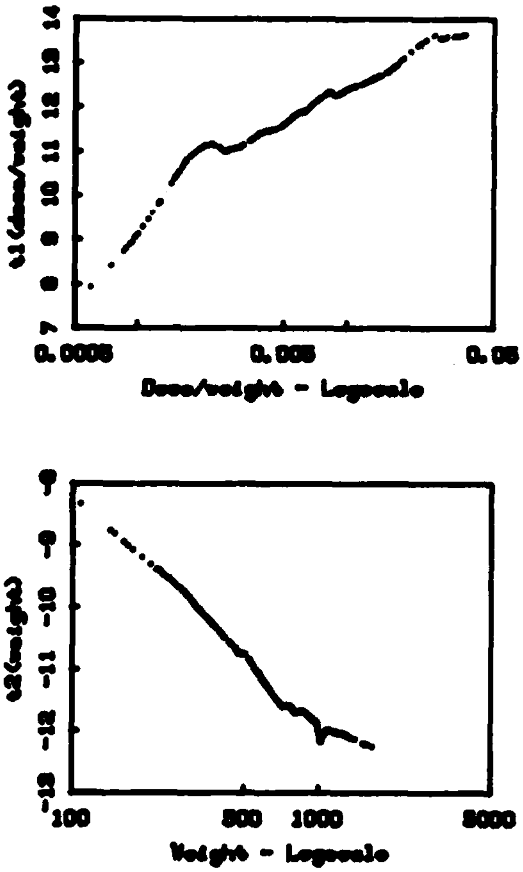


Figure 4. Non-parametric transformations versus covariates, on logarithmic scale.

Acknowledgements. I would like to thank Dr. Jacqueline Robertson for providing the data sets used in the examples and for reading the manuscript. I would also like to thank Kristen Rusk for helping in the formating and printing of this paper.

References

Baker,R.J. and Nelder,J.A. (1978). The GLIM system, Release 3. Oxford: Numerical Algorithms Group.

Becker, R.A. and Chambers, J.M. (1984). An interactive environment for data analysis and graphics. Wadsworth Advanced Book Program. Belmont, California.

Breiman,L. and Friedman,J.H. (1985). Estimating optimal transformations for multiple regression and correlation. J. Am. Statist. Assoc., 80, 580-598.

Cox, C. (1984). Generalized linear models - The missing link. Appl. Statist. 33, no.1, 18-24.

Hasselbald,V., Stead,A.G. and Creason,J.P. (1980). Multiple probit analysis with a nonzero background. Biometrics 36, 659-663.

Hastie,T. and Tibshirani,R. (1984). Generalized additive models. Technical report no.2. Laboratory of computational statistics. Stanford Univ.

Robertson,J.L. and Kimball,R.A. (1979). Effects of insect growth regulators on the western spruce budworm (Christoneura occidentalis) (Lepidoptera:Torticidae): I. Lethal effects of last instar treatments. Can. Ent. 111, 1361-1368.

Scallan,A. (1982). GLIM 82. Proceedings of the international conference of generalized linear models. ed. R.Gildhirst. N.Y. Springer-Verlag. 122-127.

Appendix

The following macros are used together with the OWN facility of GLIM to fit the parametric probit model of section 2. In these macros lp represents the original linear predictor $\beta'xu$ and %lp represents the modified linear predictor with the new explanatory variable m.

```
$c GLIM MACROS
$subfile probit
$mac fv
$switch %s mext
$calc %s=1 : phe=%np(lp)*u:
dphe=(1-u)+u*(%exp(-.5*lp**2)/%sqrt(2*%pi))
$calc m(i)=(1-phe)/((1-%pe(a))*dphe)
$calc %fv(i)=%pe(a)+(1-%pe(a))*phe
$endm
$mac dr $calc %dr(i)=1/((1-%pe(a))*dphe)
$endm
$mac va $calc %va=%fv*(1-%fv)/n
$calc %va=%if(%le(%va,0),.0000001,%va)
$endm
$mac di
$calc %di=-2*n*(%yv*%log(%fv/%yv)+
          (1-%yv)*%log((1-%fv)/(1-%yv)))
$endm
$mac mext $extract %pe
$calc lp(i)=%lp-%pe(a)*m
$endm
$return
!
!
$c A GLIM SESSION TO FIT PROBIT REGRESSION
   WITH NON-ZERO BACKGROUND TO INSECTICIDE
   DATA IN SECTION 3.
$calc %i=24         !SETS UP SAMPLE SIZE.
$units %i $data d r n tr sex !READS IN DATA.
$dinput 7$
$input 8 probit$       !READS IN OWN MACROS.
$calc a=1 $fac a 1 !SETS UP NUMBER OF DIFFERENT
                   !BKG. PARAMETERS TO ESTIMATE.
$fac tr 2 sex 2
$calc ld=.4343*%log(d)     !LOGARITHM TEN OF DOSE
$calc i=%gl(%i,1)  : u=%ne(d,0)
----- invalid function/operator arguments(s)
$c IGNORE THE INVALID FUNCTION ERROR MESSAGES.
$yvar r  $err b n $link p $wei u
$fit ld    ! THIS PRODUCES  INIT. ESTIMATES USING
           ! DATA WITH DOSE GREATER THAN ZERO.
$
          scaled
cycle      deviance        df
  4          114.7         18
$calc lp=u*%lp  :y=r/n
$yvar y
----- current display inhibited
$calc %pe(1)=.05  ! INIT. VALUE FOR BKG.
$calc %lp=m=0  ! SETS INIT. VALUE FOR %lp AND m.
$calc b=1 $fac b 1 ! A DUMMY FACTOR NEEDED TO
                   ! HAVE CONTROL OVER %pe ORDER
                   ! IT WOULD NOT BE NEEDED WERE
                   ! 'a' NOT A FACTOR.
$own fv dr va di
$wei $scale 1
$c THE FOLLOWING FIT IS FOR MODEL B OF TABLE 1.
$fit a.m+sex.tr.u+b.ld-%gm$d e$
----- invalid function/operator argument(s)
         scaled
cycle    deviance       df
   5       23.08        18

      estimate        s.e.        parameters
  1   0.5305e-01    0.1500e-01    a(1).m
  2    1.438        0.1685        b(1).ld
  3    1.056        0.1393        sex(1).tr(1).u
  4   -0.1006       0.1123        sex(1).tr(2).u
  5    1.071        0.1401        sex(2).tr(2).u
  6   0.3486        0.1141        sex(2).tr(2).u
 scaled parameter taken as 1.0000

$stop
```

ON THE FITTING AND FORECASTING OF VECTOR TIME SERIES MODELS

B.L. Shea, Numerical Algorithms Group Ltd., U.K.

1. INTRODUCTION

Let $W_t = (w_{1t}, w_{2t}, \ldots, w_{kt})'$, $(t = 1,2,\ldots,n)$, denote a vector of k time series assumed to be jointly stationary and generated by the model

$$\tilde{W}_t - \phi_1\tilde{W}_{t-1} - \cdots - \phi_p\tilde{W}_{t-p} = \epsilon_t - \theta_1\epsilon_{t-1} - \cdots - \theta_q\epsilon_{t-q} \qquad (1)$$

where $\tilde{W}_t = W_t - \mu$ denotes the deviation of W_t from its mean μ and $\epsilon_t = (\epsilon_{1t}, \epsilon_{2t}, \ldots, \epsilon_{kt})'$, $(t=1,2,\ldots,n)$, denotes a vector of k residual series assumed to have a multivariate normal distribution with zero mean and positive definite covariance matrix $\Sigma = \sigma^2 Q$. We shall also assume that $E(\epsilon_t\epsilon_s') = 0$ for $t \neq s$.

(1) is called a vector autoregressive-moving average (VARMA) process of order (p,q). $\phi = (\phi_1, \phi_2, \ldots, \phi_p)$ are the p k x k matrices of autoregressive parameters and $\theta = (\theta_1, \theta_2, \ldots, \theta_q)$ are the q k x k matrices of moving average parameters. (1) may be written in the state space form

$$h'\alpha_t = \tilde{W}_t - \epsilon_t$$

$$\alpha_t = A\alpha_{t-1} + R\tilde{W}_{t-1}$$

where α_t is the state vector of length kr with $r = \text{maximum}\,(p,q)$ and

$$A = \begin{bmatrix} \theta_1 & I & & \\ \theta_2 & & I & \\ \cdot & & \cdot & \\ \cdot & & & \cdot \\ \theta_{r-1} & & & I \\ \theta_r & 0 & & 0 \end{bmatrix}, \quad R = \begin{bmatrix} \phi_1 - \theta_1 \\ \phi_2 - \theta_2 \\ \cdot \\ \cdot \\ \cdot \\ \phi_r - \theta_r \end{bmatrix}, \quad h = \begin{bmatrix} 1 \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \end{bmatrix}$$

(Note that $\phi_i = 0$ for $i > p$ and $\theta_j = 0$ for $j > q$). Let $a_{t|t-1}$ denote the linear minimum mean square error (MMSE) estimate of α_t given data up to time t-1 and $\sigma^2 P_{t|t-1}$ the covariance matrix of this estimation error. Similarly let a_t denote the MMSE estimate of α_t given data up to time t and $\sigma^2 P_t$ the covariance matrix of this estimation error. Then it can be shown that the log likelihood function is given by the expression

$$\log L(\phi,\theta,\mu,\sigma^2,Q) = -\frac{nk}{2}\log(2\pi\sigma^2) - \frac{1}{2}\log\prod_{t=1}^{n}|F_t| - \frac{1}{2\sigma^2}\sum_{t=1}^{n} V_t^1 F_t^{-1} V_t \qquad (2)$$

For $t=1,2,\ldots,n$ the one step ahead prediction errors V_t and the corresponding covariance matrices $\sigma^2 F_t = E(V_tV_t')$ are generated by the recursive equations

$$a_{t|t-1} = Aa_{t-1} + R\tilde{W}_{t-1}$$

$$V_t = \tilde{W}_t - h'a_{t|t-1}$$

$$P_{t|t-1} = AP_{t-1}A'$$

$$F_t = h'P_{t|t-1}h + Q$$

$$a_t = a_{t|t-1} + P_{t|t-1}hF_t^{-1}V_t$$

$$P_t = P_{t|t-1} - P_{t|t-1}hF_t^{-1}h'P_{t|t-1}$$

Starting values are given by setting $a_{1|0} = 0$ and calculating $P_{1|0}$ using the method described by Jones (1980). Ansley and Kohn (1983) discuss a similar procedure for a different state space representation which is less efficient whenever $p \leqslant q$. If we have no missing observations then the following recursive equations should be used (see Shea (1986))

$$V_t = \tilde{W}_t - ha'_{t|t-1}$$

$$a_{t|t-1} = Ta_{t-1|t-2} + K_{t-1}F_{t-1}^{-1}V_{t-1}$$

$$K_t = K_{t-1} + TL_{t-1}M_{t-1}(h'L_{t-1})'$$

$$F_t = F_{t-1} + h'L_{t-1}M_{t-1}(h'L_{t-1})'$$

$$M_t = M_{t-1} - M_{t-1}(h'L_{t-1})'F_t^{-1}h'L_{t-1}M_{t-1}$$

$$L_t = TL_{t-1} - K_{t-1}F_{t-1}^{-1}h'L_{t-1}$$

where $T = A + Rh'$ is just the matrix A with θ's replaced by ϕ's. Initial conditions are given by $a_{1|0} = 0$, $K_1 = L_1 = TP_{1|0}h + RQ$, $F_1 = h'P_{1|0}h + Q$, $M_1 = -F_1^{-1}$. The matrices M_t (k x k) and L_t (kr x k) unlike σ^2F_t and the well known Kalman gain matrix $\sigma^2K_t = E(\alpha_{t+1}V_t')$ have no physical interpretation as covariance matrices.

For a stationary (and invertible) process $E\{(V_t - \epsilon_t)(V_t - \epsilon_t)'\}$ tends to zero as t becomes large. Thus the V_t's are the linear MMSE estimates of the residual series.

σ^2 (which is typically taken to be the top left hand element of Σ so that $Q(1,1) = 1$) can be differentiated out of (2) to yield a concentrated likelihood function which can be re-arranged as a sum of squares. Thus maximising (2) can be shown to be equivalent to minimising

$$\left(\prod_{1}^{n}|F_t|\right)^{1/nk}\sum_{1}^{n} V_t'F_t^{-1}V_t \qquad (3)$$

To avoid problems of underflow or overflow in calculating $\prod_1^n |F_t|$ the product can be stored in the form $a2^b$ (Martin and Wilkinson (1965)). It follows that if the model (1) is stationary then the F_t's will be positive definite and thus in calculating F_t^{-1} the Choleski factorization of F_t (C_tC_t' with C_t lower triangular) will be obtained as a by-product of the inversion process. Thus we have

$$\left(\prod_1^n |F_t|\right)^{1/nk} = D^2(n,k)$$

where $D(n,k) = \prod_{t=1}^{n} \prod_{i=1}^{k} C_t(i,i)$

If we let $v_t^* = D(n,k)C_t^{-1}V_t$ and $a_{(t-1)k+j}$ denote the jth component of v_t^* ($t=1,2,\ldots,n$, $j=1,2,\ldots,k$) then (3) reduces to

$$\sum_{s=1}^{nk} a_s^2 \qquad (4)$$

Computation of $C_t^{-1}V_t$ is speedily carried out using back substitution. A non-linear least squares algorithm such as that of Marquardt (1963) may be used to search for the maximum likelihood estimates of ϕ, θ, Q and μ. Using such an algorithm has an advantage over just using a general purpose optimization routine in that such routines are numerically more stable and generally converge to the minimum more quickly. Another advantage is that a reliable estimate of the Jacobian matrix for calculating asymptotic standard errors of parameter estimates is usually obtained as a by-product.

2. FINITE SAMPLE PREDICTION

MMSE forecasts of future series values are easily computed as follows. Let us assume we wish to forecast from time origin n and let $a_{n+1|n} = E(\alpha_{n+1}|W_1,W_2,\ldots,W_n)$ and $\sigma^2P_{n+1|n} = E\{(\alpha_{n+1}-a_{n+1|n})(\alpha_{n+1} - a_{n+1|n})'\}$. If $\hat{W}_n(1)$ denotes the linear MMSE estimate of W_{n+1} given $W_1,W_2,\ldots,W_n$ then $\hat{W}_n(1) = E(W_{n+1}|W_1,W_2,\ldots,W_n) = h'a_{n+1|n}$. It follows that σ^2F_{n+1}

$$= \sigma^2(h'P_{n+1|n}h + Q),$$ called the mean square error of prediction matrix

$$= E\{(W_{n+1} - \hat{W}_n(1)) \ (W_{n+1} - \hat{W}_n(1))'\}.$$ Since

$\alpha_{n+1} = T\alpha_{n+1-1} + R\epsilon_{n+1-1}$ and $h'\alpha_{n+1} = W_{n+1} - \epsilon_{n+1}$ we have on taking expectations conditional on $W_1,W_2,\ldots,W_n$

$$a_{n+l|n} = Ta_{n+l-1|n} + R\,E(\epsilon_{n+l-1}|W_1,W_2,\ldots,W_n)$$
$$= Aa_{n|n} + RW_n, \qquad l = 1$$
$$= Ta_{n+l-1|n}, \qquad l \geq 2$$

We also have

$$P_{n+l|n} = AP_{n|n}A', \qquad l = 1$$
$$= TP_{n+l-1|n}T' + RQR', \quad l \geq 2$$

$a_{n|n}$ and $P_{n|n}$ are easily recovered from the equations

$$a_{n|n} = a_{n|n-1} + P_{n|n-1}hF_n^{-1}V_n$$

$$P_{n|n} = P_{n|n-1} - P_{n|n-1}hF_n^{-1}h'P_{n|n-1}$$

with $P_{n|n-1} = P_{1|0} + \sum_{j=1}^{n-1} L_jM_jL_j'$

Probability limits for forecasts are calculated as follows.

Let $V_{n+1} = W_{n+1} - h'a_{n+1|n}$, then

$$V_{n+1} \approx N(0,\sigma^2F_{n+1})$$

Now $\frac{1}{\sigma^2}\sum_1^n V_t'F_t^{-1}V_t$ has a X^2 distribution (on nk degrees of freedom) independent of V_{n+1}.

Suppose interest centres on the jth time series. Let V_{n+1}^* denote the vector V_{n+1} where the jth and kth components of V_{n+1} have been interchanged. Also let F_{n+1}^* denote the matrix F_{n+1} with the jth and kth row and column interchanged. If $C_{n+1}^*C_{n+1}^{*\prime}$ is the Choleski decomposition of F_{n+1}^{*-1} with C_{n+1}^* lower triangular then

$$\frac{1}{\sigma}C_{n+1}^{*\prime}V_{n+1}^* \approx N(0,I)$$

Let d_{n+1}^* be the (k,k)th element of C_{n+1}^*, then

$$\frac{d_{n+1}^*\left(W_{n+1}^{(j)} - \hat{W}_n^{(j)}(1)\right)}{\hat{\sigma}} \sim t_{nk}$$

so that $\hat{W}_n^{(j)}(1) \pm \frac{\hat{\sigma}}{d_{n+1}^*} t_{\frac{1}{2}\alpha,nk}$ are 100(1-α) % probability limits for

$W_{n+1}^{(j)}$ where $\hat{\sigma}^2 = \frac{1}{nk}\sum_1^n V_t'F_t^{-1}V_t$.

REFERENCES

ANSLEY, C.F and KOHN, R. (1980) Exact likelihood of a vector autoregressive moving average process with missing or aggregated data . Biometrika, 70, 275-278.

JONES, R.H. (1980) Maximum likelihood fitting of ARMA models to time series with missing observations . Technometrics, 22, 389-395.

MARQUARDT, D.W. (1963) An algorithm for least squares estimation of non-linear parameters . Journal.Soc,Ind.Appl.Math.11, 431.

MARTIN, R.S. and WILKINSON, J.H. (1965) Symmetric decomposition of positive definite band matrices . Num.Math., 7, 355-361

SHEA, B.L. (1986) Estimation of multivariate time series . Journal of Time Series Analysis (to appear).

PREDICTION INTERVALS FOR THE GAMMA DISTRIBUTION

Wei-Kei Shiue, Southern Illinois University-Edwardsville
Lee J. Bain, University of Missouri-Rolla

ABSTRACT

Approximate prediction intervals for a single future observation or for the average of m future observations are developed for the two-parameter gamma distribution where both parameters are considered unknown. The methods are illustrated with three examples.

INTRODUCTION

The Pearson Type III or gamma distribution is a classical distribution which provides a useful model in many fields of application. Statistical methods for this distribution when both parameters are unknown have been slow to be developed, primarily because the parameters are not in the convenient location-scale form. Consider the gamma density function given by

$$g(x;\theta,\kappa) = \frac{1}{\Gamma(\kappa)\theta^{\kappa}} x^{\kappa-1}e^{-x/\theta}, \ x > 0; \ \kappa, \ \theta > 0.$$

The mean is $\mu = E(X) = \kappa\theta$. The parameters κ and θ are referred to as shape and scale parameters, respectively. Optimum tests for θ with κ as an unknown nuisance parameter are derived by Engelhardt and Bain (1977), based on the conditional distribution of $\bar{X}$ given $\tilde{X}$, in which $\bar{X}$ and $\tilde{X}$ denote the arithmetic and geometric sample means respectively. Tests for κ with θ unknown may be based on the maximum likelihood estimator (m.l.e.) $\hat{\kappa}$ or, equivalently, on $S = \ln(\bar{X}/\tilde{X})$. Bain and Engelhardt (1975) provide approximate distributional results for S. Grice and Bain (1980) provide approximate tests or confidence limits for the mean when both parameters are assumed unknown, and this method is extended to the two-sample case by Shiue and Bain (1983). Some related discussion concerning tolerance limits is given by Bain, Engelhardt and Shiue (1984). The purpose of this paper is to extend these results to obtain prediction intervals for a future observation or for the average of m future observations.

Suppose $x_1,\ldots,x_n$ denotes a random sample of size n from a gamma distribution, then a lower $1 - \alpha$ level prediction limit for a future observation, y, is a function of the sample, say $L_y = L_y(x_1,\ldots,x_n;\alpha)$, such that

$$P[L_y(X_1,\ldots,X_n;\alpha) \leq Y] = 1 - \alpha.$$

It can also be shown (in general) that

$$P[L_y(x_1,\ldots,x_n;\alpha) \leq Y] = E[1 - G_Y(L_y)] = 1 - \alpha,$$

and therefore a $1 - \alpha$ level prediction interval is also a $(1 - \alpha)$-expectation tolerance interval.

A lower $1 - \alpha$ level prediction limit for $\bar{y}$, the average of m future observations, would be $L_{\bar{y}}(x_1,\ldots,x_n;\alpha)$ where

$$P[L_{\bar{y}}(X_1,\ldots,X_n;\alpha) \leq \bar{Y}] = 1 - \alpha$$

An upper $1 - \alpha$ level prediction limit is obtained by replacing $1 - \alpha$ with α,

$$U_{\bar{y}}(x_1,\ldots,x_n;\alpha) = L_{\bar{y}}(x_1,\ldots,x_n;1 - \alpha).$$

A prediction limit for the total of m future observations, such as the total amount of rainfall, may be useful. In reliability applications a lower prediction limit for the total operating time which will be realized from a component with $m - 1$ spares would be meaningful, for example. It is clear that for a total $T = m\bar{Y}$,

$$L_T(x_1,\ldots,x_n;\alpha) = m\, L_{\bar{y}}(x_1,\ldots,x_n;\alpha).$$

APPROXIMATE PREDICTION INTERVALS

For a random sample of size n from a gamma distribution

$$\frac{2n\bar{X}}{\theta} \sim \chi^2(2n\kappa).$$

For m future observations, $y_1,\ldots,y_m$,

$$\frac{2m\bar{Y}}{\theta} \sim \chi^2(2m\kappa)$$

and it follows that

$$\frac{\bar{X}}{\bar{Y}} \sim F(2n\kappa, 2m\kappa)$$

where F(a,b) denotes Snedecor's F-distribution with a and b degrees-of-freedom. Note that letting m = 1 gives the important special case of a single future observation, $\bar{y} = y$.

Prediction intervals could be easily computed if κ were known since, for example,

$$P[\frac{\bar{X}}{\bar{Y}} \leq f_{1-\alpha}(2n\kappa, 2m\kappa)]$$

$$= P[\bar{Y} \geq X/f_{1-\alpha}(2n\kappa, 2m\kappa)]$$

$$= 1 - \alpha,$$

and $\bar{X}/f_{1-\alpha}(2n\kappa, 2m\kappa)$ would be a lower $1 - \alpha$ level prediction limit for $\bar{Y}$. Following the procedure of Grice and Bain (1980) the unknown κ is replaced by the m.l.e. $\hat{\kappa}$, and then the probability level actually achieved is studied. The true probability level in this case will be a function of κ, and it may differ substantially from the nominal level for small n, but it is again found that the achieved level is fairly constant over κ. Thus, it is possible to adjust the initial level to more nearly achieve the desired level when $\hat{\kappa}$ is used. That is, we propose a lower prediction limit of the form

$$L_{\bar{y}}(x_1,\ldots,x_n;\alpha) = \bar{X}/f_{1-\beta}(2n\hat{\kappa}, 2m\hat{\kappa}),$$

where β is adjusted to give approximately the correct $1 - \alpha$ level. The good rational approximation for $\hat{\kappa}$ given by Greenwood and Durand (1960) is used in the study, where $M = \ln(\bar{X}/\tilde{X})$ and

$$\hat{\kappa} = \frac{.5000876 + .1648852M - .0544274M^2}{M}, \quad 0 \leq M \leq .5772,$$

$$\hat{\kappa} = \frac{8.898919 + 9.059950M + .9775373M^2}{M(17.79728 + 11.968477M + M^2)}, \quad .5772 < M \leq 17,$$

$$\hat{\kappa} = 1/M, \quad M > 17.$$

Now, let

$$\alpha_1 = P_1(\kappa,\beta) = 1 - P[\bar{Y} \leq \bar{X}/f_\beta(2n\hat{\kappa}, 2m\hat{\kappa})]$$

and

$$\alpha_2 = P_2(\kappa,\beta) = 1 - P[\bar{X}/f_{1-\beta}(2n\hat{\kappa}, 2m\hat{\kappa}) \leq \bar{Y}],$$

then

$$U_{\bar{y}}(x_1,\dots,x_n;\alpha_1) = \bar{X}/f_\beta(2n\hat{\kappa}, 2m\hat{\kappa})$$

and

$$L_{\bar{y}}(x_1,\dots,x_n;\alpha_2) = \bar{X}/f_{1-\beta}(2n\hat{\kappa}, 2m\hat{\kappa}).$$

The values of α_1 and α_2 approach β for large n, but as noted they differ from β and depend on κ for small n. Since the dependence on the unknown κ is small, our approach is to determine guidelines for selecting a value of β which will approximately yield a desired specified α value.

Values of α_1 and α_2 were estimated by Monte Carlo simulation for several β values over a range of κ values, and the results are given in Table 1. The Monte Carlo values are based on 10,000 gamma variates generated using the IMSL subroutine GGAMR. Asymptotic values were derived mathematically for the $\kappa = 0$ and $\kappa = \infty$ lines in a manner similar to that followed in the papers cited earlier. In particular, the $\kappa = \infty$ values are the same as those obtained by Grice and Bain (1980). The $\kappa = 0$ values may be obtained in a manner similar to that used in Shiue and Bain (1983) except $\hat{\kappa}$ is now based on only a sample size n. In this case letting

$d = \frac{n}{n+m}$ and $1 - d = \frac{m}{n+m}$, we have

$$\alpha_1 = p_1(0,\beta) = \begin{cases} (1-d)\{1 - \frac{1}{n}\ln[\beta/(1-d)]\}^{-n+1}, & \beta < 1 - d \\ 1 - d\{1 - \frac{1}{n}\ln[(1-\beta)/d]\}^{-n+1}, & \beta > 1 - d \end{cases}$$

and

$$\alpha_2 = p_2(0,\beta) = \begin{cases} d\{1 - \frac{1}{n}\ln[\beta/d]\}^{-n+1}, & \beta < d \\ 1-(1-d)\{1 - \frac{1}{n}\ln[(1-\beta)/(1-d)]\}^{-n+1}, & \beta > d. \end{cases}$$

We now observe from Table 1 that the actual levels are approximately constant over the entire range of possible κ values even for moderately small sample sizes of 10, or 20. The Monte Carlo values may not be totally accurate to the three digits shown in the table, but they show the general change in the α values between the limiting values at $\kappa = 0$ and $\kappa = \infty$.

Thus for an initial β value, entering Table 1 with $\kappa = \hat{\kappa}$ would give a close estimate of the actual α value associated with the prediction limit. It would ordinarily be more helpful to know what initial β value is necessary to approximately provide a specified α level. As suggested in the earlier cited references, the simple procedure of inverting the infinity values is again recommended. Table 2 gives the value of β which yields $p_i(\infty,\beta) = \alpha$ for the commonly used values of α, and this value of β is then used to compute the prediction limits. For $n > 40$ interpolation on 1/n may be followed. Note that $p_1(\infty,\beta) = p_2(\infty,\beta)$, and also that the infinity values do not depend on m, so that only a small simple table is required. This simple adjustment should be adequate for practical applications for any range of κ values. The approximate probability levels are less accurate at very small sample sizes, but the inherent sampling variation will be relatively larger for small samples and this cause for lack of precision in the results will generally be of relatively greater importance than the small inaccuracies in the stated probability levels.

Use of Table 2, of course, gives more accurate results with problems concerning larger κ values. For example, in reliability problems in most cases $\kappa > 1$, since $\kappa > 1$ corresponds to having an increasing failure rate with age. In other applications small κ values may sometimes occur, and with very small κ and small n a closer approximation would be obtained by inverting the $p_i(0,\beta)$ values. These values depend on i and m, but this is not a difficulty since they can be inverted in closed form. For specified α_1 or α_2 we have

$$\beta_1 = (1-d)\exp\{n[1 - (\frac{\alpha_1}{1-d})^{1/(-n+1)}]\}, \quad \alpha_1 < 1 - d$$

$$\beta_1 = 1 - d\exp\{n[1 - (\frac{1-\alpha_1}{d})^{1/(-n+1)}]\}, \quad \alpha_1 > 1 - d,$$

$$\beta_2 = d\exp\{n[1 - (\frac{\alpha_2}{d})^{1/(-n+1)}]\}, \quad \alpha_2 < d,$$

$$\beta_2 = 1 - (1-d)\exp\{n[1 - (\frac{1-\alpha_2}{1-d})^{1/(-n+1)}]\}, \quad \alpha_2 > d,$$

where $d = n/(n+m)$.

Improved results are obtained if the above values are used for problems concerning small values of κ, say if $\hat{\kappa} < 1$, and Table 2 is used if $\hat{\kappa} \geq 1$. Note that $\beta_1 = \alpha_1$ when $m/(n+m) = \alpha_1$, so little adjustment of the nominal level is needed in this general range of values for small κ. This primarily applies to the case of upper prediction limits for a single future observation.

It is clear that the outlined procedure will usually require interpolation on both degrees-of-freedom and the probability level in the F tables. This is inconvenient, but it can be carried out in a few minutes on a calculator. Also using the nearest integer degrees-of-freedom should be acceptable in most cases at least for preliminary work. It is also possible to obtain a degree-of-freedom less than 1 if say $m = 1$ and $\hat{\kappa} < .5$, which would require numerical integration or some approximation to obtain the critical value. The reader may refer to Pearson (1968) for more accurate interpolation procedures. These inconveniences would, of course, be removed if the procedure is computerized. It may also be worth noting that by the inherent nature of the problem one will necessarily have very wide intervals for the $m = 1$ and small κ case. For example if the parameters are assumed known with $\kappa = .5$ then

$$P[(\theta/2)\chi^2_{.025}(1) < Y < (\theta/2)\chi^2_{.975}(1)] = .95,$$

which gives the interval $(.0005\theta,\ 2.51\theta)$.

NUMERICAL EXAMPLES

Mielke and Johnson (1974) consider a gamma model for the following accumulated streamflow data from a U.S. Geological Survey station in Colorado.

46.65	29.96	25.49	11.85
41.01	23.64	30.90	19.51
9.06	41.06	57.04	30.93
15.94	29.70	37.51	38.78
17.25	50.80	31.93	15.31
47.11	75.24	25.39	14.69
18.54	45.80	39.64	38.14
53.93	39.84	14.40	28.24

In this case $\hat{\kappa} = 4.5$ and $\bar{x} = 32.67$. A 90% two-sided prediction interval for a future accumulated streamflow reading is given by

$$(L_y(x_1,\ldots,x_n;\alpha_2),\ U_y(x_1,\ldots,x_n;\alpha_1))$$

$$= (\bar{x}/f_{1-\beta}(2n\hat{\kappa}, 2m\hat{\kappa}),\ \bar{x}/f_{\beta}(2n\hat{\kappa},\ 2m\hat{\kappa})),$$

where $\alpha_1 = \alpha_2 = .05$, $n = 32$, $m = 1$, and from Table 2, $\beta \doteq .042$. Thus

$$(L_y(x_1,\ldots,x_n;.05),\ U_y(x_1,\ldots,x_n;.05))$$

$$= (32.67/f_{.958}(288,9),\ 32.67/f_{.042}(288,9)$$

$$= (32.67/2.93,\ 32.67/.504)$$

$$= (11.2,\ 64.8).$$

In reliability applications this data could represent the times-to-failure of a certain type component, *and one would wish to predict the* time-to-failure of a new component being placed in service.

EXAMPLE 2

Lieblein and Zelen (1956) present $n = 23$ values of the endurance, in millions of revolutions, of deep-groove ball bearings. A gamma distribution is suggested as an appropriate model by Bain and Engelhardt (1980). For this data $\bar{x} = 72.22$, $\tilde{x} = 63.46$ and $\hat{\kappa} = 4.025$. Suppose a lower 90% prediction limit is desired for the endurance of such a bearing. We have $\alpha = .10$, $n = 23$, $m = 1$ and $\beta \doteq .088$ from Table 2. Thus

$$L_y(x_1,\ldots,x_n;.10) \doteq \bar{x}/f_{1-\beta}(2n\hat{\kappa},\ 2m\hat{\kappa})$$

$$= 72.22/f_{.912}(185.2,\ 8.05)$$

$$= 72.22/1.77$$

$$= 40.8.$$

Suppose a lower 90% prediction limit is desired for the total life time of a bearing and 2 spares. *We have $m = 3$,*

$$L_{\bar{y}}(x_1,\ldots,x_n;.10)$$

$$\doteq 72.22/f_{.912}(185.2,\ 24.15)$$

$$\doteq 49.47,$$

and a lower prediction limit for the total lifetime of 3 bearings is
$L_y(x_1,\ldots,x_n;.10) = 3(49.47) = 148.4.$

EXAMPLE 3

Crow (1977) considers a gamma model for hail data measured by hail/rain separators which were reported in Crow, et. al., (1976). For 17 seeded days he obtains the estimates $\hat{\kappa} = .466$ and $\bar{x} = 13.249$. Suppose a lower 95% prediction interval is desired for the total amount of hail measured on five days. We have

$$L_{\bar{y}}(x_1,\ldots,x_n;\alpha_2) = \bar{x}/f_{1-\beta_2}(2n\hat{\kappa},\ 2m\hat{\kappa}),$$

where $\alpha_2 = .05$, $n = 17$, $m = 5$, $d = 17/22 = .773$, and $\beta_2 = .773$

$$\exp\{17[1 - \left(\frac{.05}{.773}\right)^{-1/16}]\} = .035.$$

$$L_{\bar{y}}(x_1,\ldots,x_n;.05) = 13.249/f_{.965}(15.84,\ 4.66)$$

$$= 13.249/6.17$$

$$= 2.1.$$

For the 5 day total, $T = 5\bar{y}$, and a 95% lower prediction limit for T is $5(2.1) = 10.5$.

REFERENCES

Bain, Lee J., and Engelhardt, Max (1975). "A Two-moment Chi-Square Approximation for the Statistic Log ($\bar{x}/\tilde{x}$)," Journal of the American Statistical Association, 70, 948-950.

Bain, L.J. and Engelhardt, M. (1980). "Probability of Correct Selection of Weibull Versus Gamma Based on Likelihood Ratio," Communications in Statistics, A9(4), 375-381.

Bain, Lee J., Engelhardt, Max, and Shiue, Wei-Kei (1984). "Approximation Tolerance Limits and Confidence Limits on Reliability for the Gamma Distribution,' IEEE Trans. on Reliability, R-33, No. 2, June 1984, 184-187.

Crow, E.L. (1977). "Minimum Variance Unbiased Estimators of the Ratio of Two Lognormal Variates and Two Gamma Variates," Communications in Statistics, A6, 967-975.

Crow, E.L., Summers, P.W., Long, A.B., Knight, C.A., Foote, G.B., and Dye, J.E., (1976). Final Report-National Hail Research Experiment Randomized Seeding Experiment 1972-74, Vol. 1, Experimental Results and Overall Summary. National Center for Atmospheric Research, Boulder, Colorado.

Engelhardt, M., and Bain, L.J., (1977). "Uniformly Most Powerful Unbiased Tests on the Scale Parameter of a Gamma Distribution with a Nuisance Shape Parameter," Technometrics, 19, 77-81.

Greenwood, J.A. and Durand, D., (1960). "Aids for Fitting the Gamma Distribution by Maximum Likelihood," Technometrics, 2, 55-65.

Grice, John, V. and Bain, Lee J., (1980). "Inferences Concerning the Mean of the Gamma Distribution," Journal of the American Statistical Association, 75, 929-933.

Lieblein, J. and Zelen, M. (1956). "Statistical Investigation of the Fatigue Life of Deep-groove Ball Bearings," J. Res. Nat. Bur. Stand., 57, 273-316.

Mielke, P.W. Jr., and Johnson, E. (1974). "Some Generalized Beta Distributions of the Second Kind Having Desirable Application Features in Hydrology and Meteorology," Water Resources Research, 10, 223-226.

Pearson, E.S. (1968). "Lagrangian Coefficients for Interpolating Between Tabled Percentage Point," Biometrika, 55, 19-28.

Shiue, Wei-Kei and Bain, Lee J., (1983). "A Two-Sample Test of Equal Gamma Distribution Scale Parameters with Unknown Common Shape Parameter," Technometrics, 25, 377-381.

ACKNOWLEDGMENT

Research sponsored by the Air Force Office of Scientific Research, Air Force Systems Command, USAF, under Grant Number AFOSR 84-0164. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon.

TABLE 1

			$\alpha_1 = P_1(\kappa,\beta)$				$\alpha_2 = P_2(\kappa,\beta)$			
			β				β			
n	m	κ	.01	.025	.05	.10	.01	.025	.05	.10
5	1	0	.028	.046	.070	.113	.066	.099	.139	.203
		.2	.028	.046	.071	.117	.063	.092	.128	.184
		.5	.034	.054	.080	.128	.066	.094	.129	.183
		1.0	.038	.059	.086	.134	.063	.090	.123	.175
		4.0	.045	.068	.097	.146	.058	.084	.116	.167
		∞	.053	.077	.108	.158	.053	.077	.108	.158
10	1	0	.015	.030	.054	.099	.032	.057	.092	.151
		.2	.015	.031	.056	.105	.030	.054	.086	.140
		.5	.018	.036	.062	.111	.033	.057	.089	.143
		1.0	.021	.039	.065	.115	.033	.056	.086	.139
		4.0	.024	.043	.071	.121	.031	.052	.082	.134
		∞	.027	.048	.077	.128	.027	.048	.077	.128
20	1	0	.011	.026	.050	.097	.019	.040	.070	.125
		.2	.012	.027	.052	.102	.019	.038	.067	.120
		.5	.013	.029	.055	.105	.020	.039	.068	.121
		1.0	.014	.031	.056	.106	.019	.038	.066	.118
		4.0	.016	.033	.059	.109	.018	.037	.065	.116
		∞	.018	.036	.063	.113	.018	.036	.063	.113
5	5	0	.049	.076	.110	.164	.022	.042	.070	.120
		.2	.042	.065	.095	.146	.020	.038	.065	.115
		.5	.046	.070	.100	.151	.023	.042	.070	.121
		1.0	.048	.072	.102	.152	.024	.044	.072	.122
		4.0	.051	.075	.105	.156	.026	.046	.075	.128
		∞	.053	.077	.108	.158	.027	.048	.077	.128
10	5	0	.022	.042	.070	.120	.028	.052	.084	.140
		.2	.020	.038	.065	.115	.024	.044	.073	.124
		.5	.023	.042	.070	.121	.026	.047	.076	.128
		1.0	.024	.044	.072	.122	.027	.047	.076	.127
		4.0	.026	.046	.075	.126	.028	.047	.077	.128
		∞	.027	.048	.077	.128	.027	.048	.077	.128
10	20	0	.028	.052	.084	.140	.022	.042	.070	.120
		.2	.023	.042	.070	.122	.020	.038	.065	.115
		.5	.025	.045	.074	.126	.023	.042	.070	.121
		1.0	.026	.047	.075	.126	.024	.044	.072	.123
		4.0	.027	.048	.077	.128	.026	.047	.075	.126
		∞	.027	.048	.077	.128	.027	.048	.077	.128
20	10	0	.015	.033	.060	.110	.018	.037	.066	.119
		.2	.014	.031	.057	.107	.016	.033	.060	.110
		.5	.015	.033	.059	.110	.016	.034	.061	.112
		1.0	.016	.033	.059	.110	.016	.034	.061	.111
		4.0	.017	.034	.061	.112	.017	.035	.062	.113
		∞	.018	.036	.063	.113	.018	.036	.063	.113

Table 2. Values of β for $\alpha = p_i(\infty,\beta)$

				α			
n	.005	.01	.025	.050	.075	.100	.250
5	.0000	.0000	.0010	.0086	.0234	.0432	.2038
10	.0003	.0015	.0086	.0267	.0486	.0724	.2294
20	.0017	.0046	.0159	.0380	.0619	.0866	.2403
40	.0030	.0070	.0203	.0440	.0685	.0934	.2453
∞	.0050	.0100	.0250	.0500	.0750	.1000	.2500

COMPUTER-AIDED SURVEY METHODS

Robert D. Tortora and Frederic A. Vogel
U.S. Department of Agriculture
J. Merrill Shanks, University of California at Berkeley

Abstract

This paper describes computer-assisted survey methods that are being planned, tested, and implemented by the Statistical Reporting Service (SRS) of the U.S. Department of Agriculture for its 44 data collection offices across the United States. The major activities include developing a Data Management System along with Computer-Aided Sampling Frame Maintenance, Computer-Aided Survey Management, and Computer Assisted Telephone Interviewing Systems.

Introduction

The Statistical Reporting Service (SRS) administers the United States Department of Agriculture's program of collecting and publishing current national and state agricultural statistics. SRS is totally dependent upon computer technology to carry out this service. The following paragraphs outline characteristics of the SRS survey and estimating program that affect its use and development of computer methodology.

Estimates for about 120 crops, 45 livestock, and 50 farm economic items are published in about 300 national reports each year. In addition, estimates for many of these items are also published at the state and county level. The reports vary in frequency by item, but the frequency varies from weekly to monthly, to quarterly and annually.

The dynamic agricultural markets rely upon very timely information. Data collection periods are generally limited to 10-15 days with the published estimates following within 2-3 weeks. To illustrate, six major surveys that included about 150,000 farm operators were conducted during the December 1 - January 10 time period. Reports were published as early as December 22 and all were published by February 10. The actual release dates for all reports is announced about one year in advance.

The need for timeliness has been met by developing parameter driven application programs for data edit, analysis, and summary. The data collection and data capture activities are distributed to 44 State Statistical Offices. However, the current mode of data processing is batch oriented. The 44 data collection offices communicate with the Washington, D.C. headquarters over a leased communications network. The IBM mainframe, used for the majority of presurvey, survey, and post-survey processing, is also leased and is located in Orlando, Florida.

To improve the timeliness and quality of the agricultural data that SRS collects, this paper will describe the four systems: Data Management System, Computer-Aided Sampling Frame Maintenance System, Computer-Aided Survey Management System, and Computer-Assisted Telephone Interviewing (CATI) that will create an on-line real time processing environment.

Data Management System

The Data Management System affects all areas of SRS's work. It includes the basic sampling frames, raw survey data, sample estimates and their measures of error, and administrative data, such as budgets, salaries, equipment inventories, etc. It also includes all of the functions that SRS uses to design, implement, collect, and publish survey data. Figure 1 illustrates the relationship between the sets of data SRS uses to conduct its day-to-day operations and functional activities.

The major feature of the data management system is that it will efficiently integrate the data sets with the functional activities. To implement this integration, the data management system development is divided into three parts. These include the data dictionary/data directory; i.e., the metadata for SRS, the logical design and physical implementation of the data base management system, and the applications programs. A data base management system called ADABAS and a 4th generation language, called NATURAL, for application program development are being used. Instead of developing the data management system for all activities at once, the project has been subdivided into subject matter areas. These areas will be modeled to describe the data elements required by the data users and to define the relationships between the data elements. Table 1 lists the 16 subject matter areas. Work has begun on the budget model and the equipment and supply model. As application programs are being developed for these two models, modeling will begin for specialty crops.

The remainder of this paper discusses three functional activities associated with the data management system; viz., computer-aided survey management, computer-aided sampling frame maintenance, and computer-assisted telephone interviewing.

Computer-Aided Sampling Frame Maintenance

SRS maintains data on three frames: A list frame, area frame, and release frame. The first two frames are used for survey design, data collection and analysis. The release frame contains information on nonfarmers that should receive survey results. The list frame contains 1.8 million records. The area frame contains over 65,000 records and the release frame contains over 50,000 names nationwide. Of course, by data collection site the number of records may vary from less than 10,000 to more than 100,000. All of the processes associated with frame maintenance are currently conducted in a batch environment with transactions being hand coded to forms and those forms then being key entered. The computer-aided frame maintenance activities will allow SRS to search and display records, add new records, and maintain and change records in an on-line environment. Search and display will be used for on-line overlap/nonoverlap determinations and duplicate record detection. Improved overlap/nonoverlap determination procedures reduce a major source of nonsampling errors in dual frame sampling (Vogel, 1975). Improved detection of duplicate records reduces the possibility of duplicate records existing in the list frame. This ensures that correct probabilities of selection are used for the list frame estimators. Duplicate records can occur because many farms and ranches are operated as partnerships or have operation names that can occur in the frame along with individual names. The add record function will be used to add new records to the frames. Examples of the use of this function include adding the new names associated with the 20 percent rotation of the area frame and the continual addition of new names to the list frame as new sources of farm and ranch names are found. The maintenance and change function includes updating the names and addresses of farmers and ranchers as well as adding, deleting, or changing the auxiliary information describing a farm operation such as number of acres, cattle and hog inventories, etc., used for sampling purposes.

All of the data used in this system will be accessible on-line. Besides the name, address, and phone number of each farm or ranch, it will include: Up to 100 items of auxiliary information for sampling purposes; identifiers indicating the surveys in which the unit was selected; information about the publications the sample unit should receive; alternative names for the sampling unit; and, finally, a "comments" section. This system is also being implemented with ADABAS.

Computer-Aided Survey Management

As described in the Introduction, SRS has a critical need to keep track of and be able to report the status of several surveys being conducted either simultaneously, or with considerable overlap in data collection activity. The Computer-Aided Survey Management system will meet this need. This system will provide ad hoc inquiries about the survey process or produce aggregate reports describing survey status.

Information the system will produce will include reports about the location of sample units by geographic area--useful in interviewer assignments. Other reports will categorize sample units by surveys--useful in grouping questionnaires that have to be completed by the same respondent for several simultaneous surveys. Reports that describe sample units by mode of data collection--mail, telephone, and face-to-face--will also be produced. And, finally, reports on survey status from presurvey activities through data collection and data edit will be produced to help each field office manage the survey process.

A prototype system has been implemented on PC's using a DBASE III. It will eventually be integrated into the main frame environment and ADABAS.

Computer-Assisted Telephone Interviewing

CATI replaces the paper and pencil questionnaire historically being used for telephone data collection. The telephone interviewers now have the questionnaire displayed on a cathode ray tube and respondents' answers are keyed directly into the machine for editing and retention. SRS has been working jointly with the University of California at Berkeley since 1980 to test, develop, and implement the Berkeley/USDA CATI system. 1/ SRS currently has 11 field offices operational using CATI on super micros operating in a UNIX environment. 2/ Late this summer, four more field offices will receive the hardware necessary for CATI operations. The major development effort in this area for SRS will be the testing and implementation of CATI in a MS-DOS environment using a Local Area Network (LAN). The CATI software has been successfully loaded on a LAN. Operational testing will begin this summer.

Summary

The distributed processing requirements described above are essential to operate an efficient statistical organization. To improve the timeliness and quality of agricultural data in the U.S. the Statistical Reporting Service is testing, developing, and implementing new systems to process in an interactive environment. These include a Data Management, Computer-Aided Frame Maintenance, Computer-Aided Survey Management, and Computer-Assisted Telephone Interviewing Systems.

Table 1. - Subject Matter Areas for Data Modeling

Field Crops
Specialty Crops
Livestock
Dairy
Poultry
Prices Paid
Prices Received

Labor, et al
Area Frame
List Frame
Release Frame
Personnel
Budget
Equipment/Supplies

Footnotes

1/ For a more complete description of the SRS CATI environment, see Tortora (1985). For a more complete description of the CATI environment in general, see the paper by Nicholls and Groves (1986) in these proceedings.

2/ UNIX is a trademark of AT&T Bell Labs.

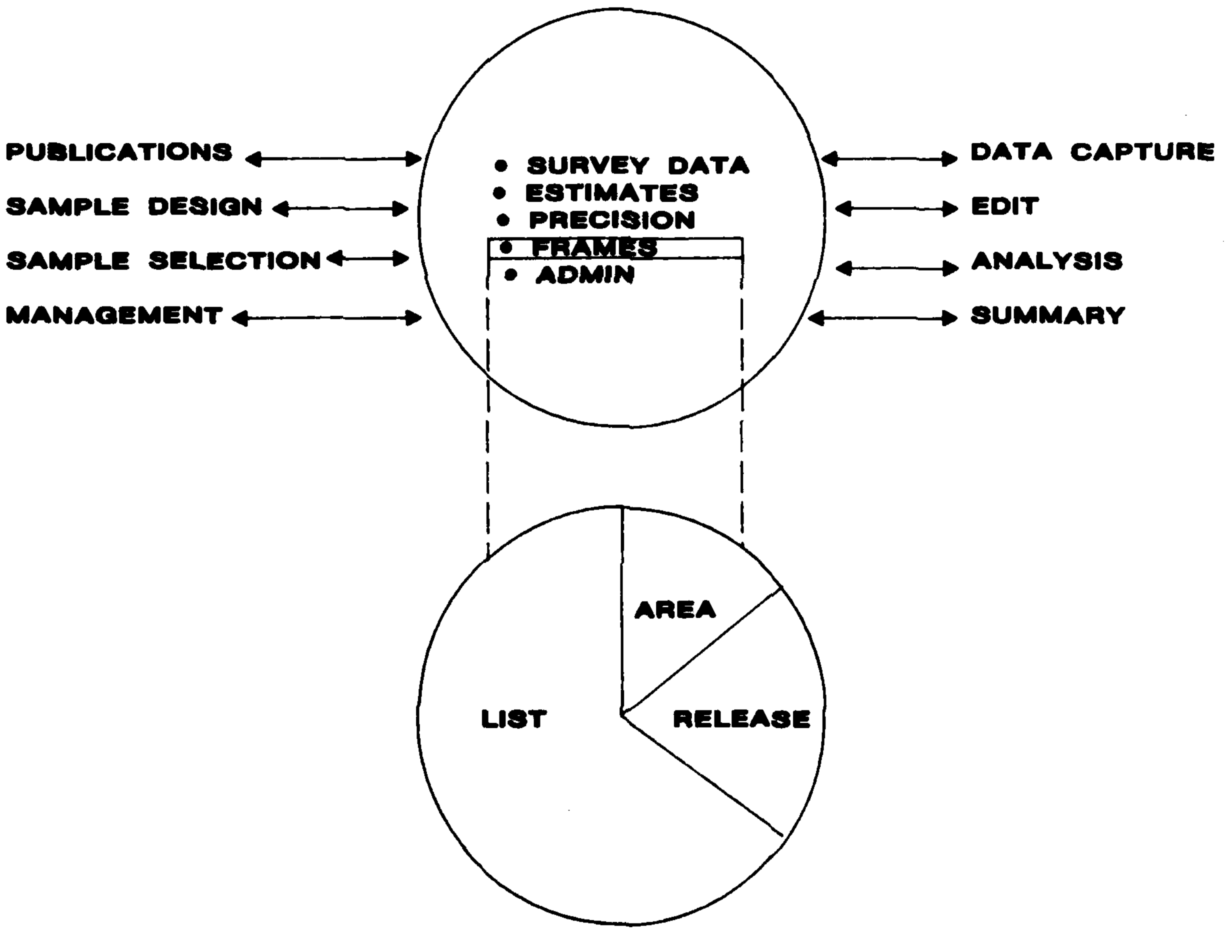
References

Nicholls, W. II and R. Groves (1986). The Status of Computer-Assisted Telephone Interviewing. Proceedings of Computer Science and Statistics: 18th Symposium on the Interface. Washington,DC

Tortora, R.D. (1985). CATI in an Agricultural Statistical Agency. Journal of Official Statistics, Vol. 1, No. 3, pp. 301-314.

Vogel, F.A. (1975). Surveys with Overlapping Frames - Problems in Application. Proceedings of the Social Statistics Section, ASA Annual Meeting, Washington, D.C.

DATA MANAGEMENT SYSTEM



TWO MRPP RANK TESTS AND THEIR SIMULATED POWERS FOR SOME ASYMMETRIC POPULATIONS

Derrick S. Tracy, University of Windsor

ABSTRACT

Two rank tests, based on multiresponse permutation procedure, are compared with respect to their empirical powers, when sampling from underlying Weibull and gamma populations, for different parameters. Simulated powers of the test statistics against various location shifts are examined under Pearson Type III and Type VI distribution approximations. The relative gains in power depend upon the values of the parameter.

INTRODUCTION

For most tests of hypotheses, the test statistic is derived under several assumptions, such as normality and homogeneity of variances. To avoid making such assumptions when analyzing multiresponse data, Mielke, Berry and Johnson (1976) proposed an exact permutation procedure and introduced the MRPP (multiresponse permutation procedure) test statistic. It is optimal when responses are made commensurate with each other. It is applicable to data at ordinal level or higher, as encountered often in social and biological sciences. The exact permutation procedure requires very heavy computations; hence certain approximations are considered.

MRPP STATISTICS

Let Ω be a population of observations $X_1, \ldots, X_N$. Let K of them be classified according to some *a priori* classification scheme into g mutually exclusive subgroups $S_1, \ldots, S_g$, with n_i observations in S_i, leaving $N-K = n_{g+1}$ observations in the excess subgroup S_{g+1}. The MRPP test statistic δ is weighted average of distances between all pairs of observations within each of the classified subgroups. Thus $\delta = \sum_1^g C_i \xi_i$, where $C_i > 0$ are weights with $\sum_1^g C_i = 1$. And

$$\xi_i = \binom{n_i}{2}^{-1} \sum_{I<J}^{N} \Delta_{IJ} \; S_i(X_I) S_i(X_J)$$

is the average of a distance measure Δ_{IJ} between X_I and X_J, where $S_i(X_I)$ is an indicator function, taking the value 1 if X_I is in S_i and 0 otherwise. When the classification is random, each of the $N!/\Pi_1^{g+1} n_i!$ permutations is equally likely to occur. Thus the value of δ is likely to be higher than when the classification is done according to some *a priori* scheme. Therefore an α level test rejects 'H_0: Classification is random' if $\delta \leq \delta_\alpha$.

Mielke, Berry, Brockwell and Williams (1981) consider special cases of δ with $C_i = n_i/K$ and $\Delta_{IJ} = |R(X_I)-R(X_J)|^\nu$, $R(X_I)$ being the rank of X_I. For $\nu=1$ and 2, they denote the test statistic by δ_1 and δ_2 respectively. With $n_{g+1} = 0$, $g = 2$, $\nu = 2$, $n_1 = n_2$, the MRPP test based on δ_2 is equivalent to the two-sided Wilcoxon test. For $g \geq 2$, $C_i = \frac{n_i - 1}{K-g}$, $N=K$, δ_2 is equivalent to the Kruskal-Wallis test. But when $\nu \neq 2$, $N = K$, $r = 1$ and $C_i = n_i/K$, Brockwell, Mielke and Robinson (1982) show that δ has a non-normal non-invariant distribution, and its asymptotic distribution depends on the underlying distribution of observations.

For $\nu = 1$, Tracy and Tajuddin (1986) study the distribution of δ_1 for large samples when $g = 2$, and $n_1 = n_2 = N/2$ for several underlying symmetric populations. Here we consider asymmetric underlying populations, taking them to be Weibull and gamma with different parameters. We conduct an extensive simulation study based on 10,000 samples from the underlying populations.

Using Mielke *et al.* (1981) results for the first three moments of δ_1, and those of Tracy and Tajuddin (1985) for the fourth moment, we obtain β_1 and β_2 of δ_1, and the Pearson criterion $2\beta_2 - 3\beta_1 - 6$ for various values of N. This indicates (Tracy and Tajuddin, 1986) that for $N > 34$, the Pearson Type VI distribution is a better approximation. We obtain powers of δ_1 both under Pearson Type VI and Type III approximations, and compare with the powers of δ_2 using Pearson Type III approximation, which is known to be its asymptotic distribution.

THE METHOD

We consider 10,000 independent samples of 80 observations from Weibull $W(\beta)$ populations, with $\beta = 0.5$, 0.67, 0.8, 1.0, 1.5 and 2.0. The case of $\beta = 1.0$ is the case of the exponential population. Similarly we consider gamma $G(\beta)$ populations, with $\beta = 0.3$, 0.5, 2.0 and 3.0. The case of $G(1.0)$ is again the exponential.

We shift the last 40 of the 80 observations by $k\sigma$, where k proceeds from 0 to some appropriate value so that power curves can be drawn. The number of rejections was counted for the choice of $\alpha = 0.01$, 0.05 and 0.10.

We present our results for empirical power in Table I for Weibull and Table II for gamma underlying populations. We also present power plots for these cases, obtained by using the cubic spline method of interpolation. The standard error of any estimated power is bounded by $\sqrt{(0.5)(0.5)10000} = 0.005$. Thus any difference in power of more than $2(0.005) = 0.01$ is significant at least at the 5% level of significance.

The samples were drawn using IMSL subroutines for the respective populations.

TABLE I

Empirical Powers of δ_1 and δ_2 for Weibull Populations

Statistic	δ_1		δ_2	δ_1		δ_2	δ_1		δ_2	δ_1		δ_2
Type	VI	III	III	VI	III	III	VI	III	III	VI	III	III
α↓						$\beta = 0.5$						
Shift→		0.0			0.1σ			0.2σ			0.3σ	
0.01	.0088	.0085	.0087	.7345	.7315	.4789	.9833	.9823	.8534	.9987	.9987	.9637
0.05	.0448	.0444	.0480	.9343	.9332	.7141	.9990	.9990	.9501	1.000	1.000	.9912
0.10	.0927	.0924	.0953	.9774	.9770	.8081	.9998	.9998	.9760	1.000	1.000	.9965
						$\beta = 0.67$						
Shift		0.0			0.1σ			0.2σ			0.3σ	
0.01	.0088	.0085	.0087	.1546	.1530	.1268	.5967	.5933	.4420	.8878	.8861	.7309
0.05	.0448	.0444	.0480	.3929	.3906	.3157	.8508	.8492	.6844	.9790	.9784	.8856
0.10	.0927	.0924	.0953	.5571	.5560	.4325	.9345	.9339	.7818	.9945	.9944	.9368
						$\beta = 0.8$						
Shift		0.1σ			0.3σ			0.5σ			0.7σ	
0.01	.0572	.0561	.0599	.6119	.6086	.4983	.9546	.9538	.8744	.9969	.9968	.9792
0.05	.1821	.1805	.1783	.8496	.8478	.7339	.9932	.9930	.9624	.9997	.9997	.9964
0.10	.2959	.2951	.2773	.9286	.9281	.8233	.9987	.9987	.9814	1.000	1.000	.9982
						$\beta = 1.0$						
Shift		0.1σ			0.3σ			0.5σ			0.7σ	
0.01	.0255	.0252	.0301	.3057	.3034	.2877	.7734	.7715	.7079	.9690	.9682	.9332
0.05	.1032	.1022	.1106	.5682	.5648	.5300	.9319	.9308	.8792	.9956	.9956	.9833
0.10	.1776	.1767	.1862	.7013	.7009	.6500	.9684	.9683	.9325	.9990	.9990	.9927
						$\beta = 1.5$						
Shift		0.1σ			0.2σ			0.3σ			0.4σ	
0.01	.0146	.0152	.0159	.0462	.0458	.0509	.1131	.1123	.1236	.2315	.2298	.2477
0.05	.0695	.0685	.0754	.1482	.1467	.1613	.2813	.2792	.3082	.4654	.4628	.4909
0.10	.1311	.1309	.1372	.2354	.2346	.2547	.4042	.4030	.4281	.5964	.5958	.6139
Shift		0.5σ			0.6σ			0.7σ			0.9σ	
0.01	.4008	.3976	.4196	.5898	.5868	.6006	.7489	.7472	.7562	.9475	.9463	.9418
0.05	.6510	.6486	.6641	.8054	.8043	.8109	.9117	.9102	.9090	.9887	.9885	.9869
0.10	.7622	.7609	.7726	.8870	.8869	.8880	.9564	.9562	.9530	.9960	.9960	.9945
						$\beta = 2.0$						
Shift		0.1σ			0.2σ			0.3σ			0.4σ	
0.01	.0134	.0133	.0144	.0407	.0402	.0431	.0943	.0930	.1028	.1882	.1860	.2026
0.05	.0656	.0648	.0711	.1320	.1305	.1428	.2421	.2398	.2632	.4003	.3981	.4310
0.10	.1248	.1245	.1311	.2128	.2125	.2274	.3539	.3530	.3844	.5258	.5250	.5602
Shift		0.5σ			0.6σ			0.7σ			0.9σ	
0.01	.3269	.3246	.3553	.4956	.4935	.5295	.6588	.6567	.6897	.9018	.9007	.9164
0.05	.5723	.5701	.6053	.7224	.7204	.7574	.8536	.8515	.8745	.9730	.9725	.9790
0.10	.6822	.6817	.7168	.8238	.8232	.8463	.9145	.9144	.9307	.9883	.9883	.9909

TABLE II

Empirical Powers of δ_1 and δ_2 for Gamma Populations

Statistic	δ_1		δ_2	δ_1		δ_2	δ_1		δ_2	δ_1		δ_2
Type	VI	III	III	VI	III	III	VI	III	III	VI	III	III
α↓						β = 0.3						
Shift→		0.0			0.1σ			0.13σ			0.2σ	
0.01	.0124	.0120	.0118	.5666	.5616	.3238	.7206	.7176	.4344	.9120	.9104	.6638
0.05	.0492	.0488	.0520	.8512	.8488	.5642	.9320	.9306	.6804	.9878	.9876	.8472
0.10	.0994	.0990	.1024	.9402	.9394	.6812	.9752	.9750	.7776	.9968	.9968	.9036
Shift		0.3σ										
0.01	.9812	.9808	.8512									
0.05	.9976	.9976	.9514									
0.10	.9990	.9990	.9752									
						β = 0.5						
Shift		0.0			0.1σ			0.2σ			0.3σ	
0.01	.0094	.0092	.0102	.1234	.1226	.1018	.4632	.4600	.3360	.7732	.7712	.5914
0.05	.0454	.0444	.0432	.3348	.3322	.2650	.7432	.7408	.5702	.9420	.9414	.8042
0.10	.0922	.0916	.0946	.4918	.4908	.3670	.8700	.8696	.6832	.9784	.9776	.8830
Shift		0.4σ			0.5σ							
0.01	.9284	.9276	.7880	.9824	.9820	.9116						
0.05	.9878	.9876	.9264	.9978	.9978	.9736						
0.10	.9976	.9976	.9614	.9998	.9998	.9868						
						β = 2.0						
Shift		0.0			0.1σ			0.2σ			0.3σ	
0.01	.0100	.0096	.0096	.0200	.0196	.0216	.0636	.0630	.0690	.1496	.1488	.1664
0.05	.0472	.0464	.0518	.0838	.0824	.0884	.1768	.1762	.1934	.3456	.3426	.3628
0.10	.0974	.0966	.1002	.1458	.1446	.1560	.2778	.2766	.2956	.4778	.4764	.4900
Shift		0.4σ			0.5σ			0.6σ			0.7σ	
0.01	.3084	.3052	.3176	.5138	.5114	.5114	.7020	.7002	.6908	.8456	.8438	.8360
0.05	.5618	.5598	.5686	.7438	.7424	.7396	.8816	.8802	.8776	.9534	.9526	.9454
0.10	.6774	.6774	.6812	.8378	.8374	.8320	.9376	.9370	.9282	.9790	.9790	.9730
						β = 3.0						
Shift		0.0			0.1σ			0.2σ			0.3σ	
0.01	.0112	.0110	.0098	.0192	.0188	.0212	.0506	.0498	.0556	.1190	.1178	.1298
0.05	.0484	.0474	.0488	.0716	.0710	.0762	.1516	.1496	.1640	.2900	.2880	.3044
0.10	.0970	.0968	.1022	.1240	.1236	.1334	.2386	.2378	.2522	.4064	.4058	.4274
Shift		0.4σ			0.5σ			0.6σ			0.7σ	
0.01	.2472	.2456	.2584	.4130	.4118	.4274	.5986	.5960	.6120	.7706	.7684	.7696
0.05	.4696	.4652	.4950	.6588	.6556	.6722	.8200	.8174	.8250	.9210	.9206	.9230
0.10	.5972	.5966	.6080	.7732	.7730	.7808	.8926	.8922	.8968	.9584	.9584	.9560

POWER OF MRPP TESTS

UNDERLYING DISTRIBUTION: W(0.5)

$N_1 = N_2 = 40$

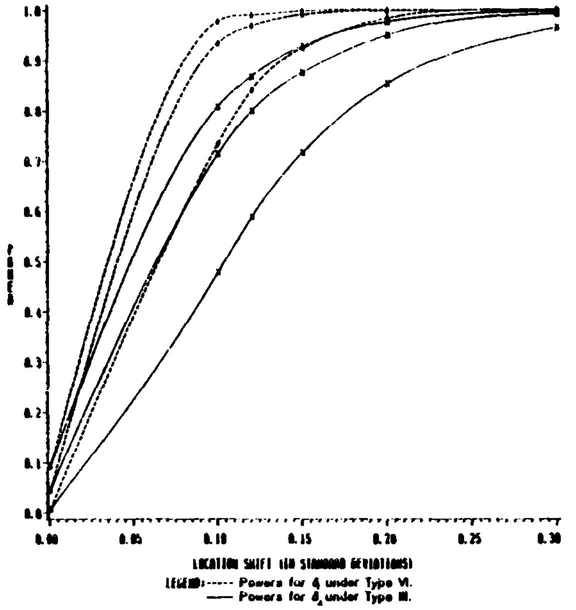


FIGURE 1

POWER OF MRPP TESTS

UNDERLYING DISTRIBUTION: W(0.67)

$N_1 = N_2 = 40$

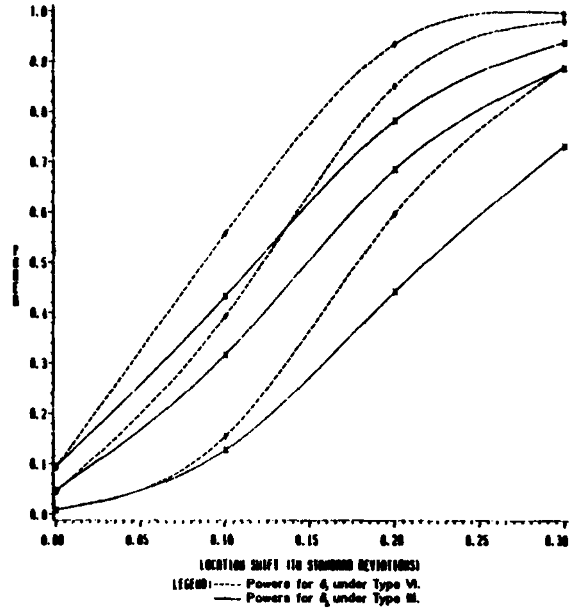


FIGURE 2

POWER OF MRPP TESTS

UNDERLYING DISTRIBUTION: W(0.8)

$N_1 = N_2 = 40$

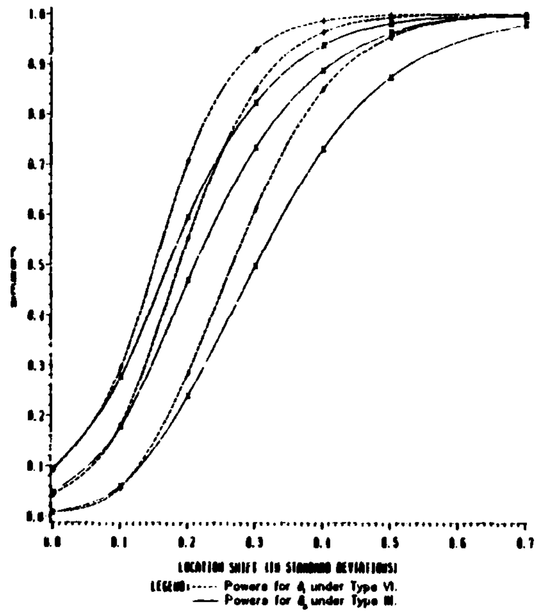


FIGURE 3

POWER OF MRPP TESTS

UNDERLYING DISTRIBUTION: EXPONENTIAL=W(1.0)

$N_1 = N_2 = 40$

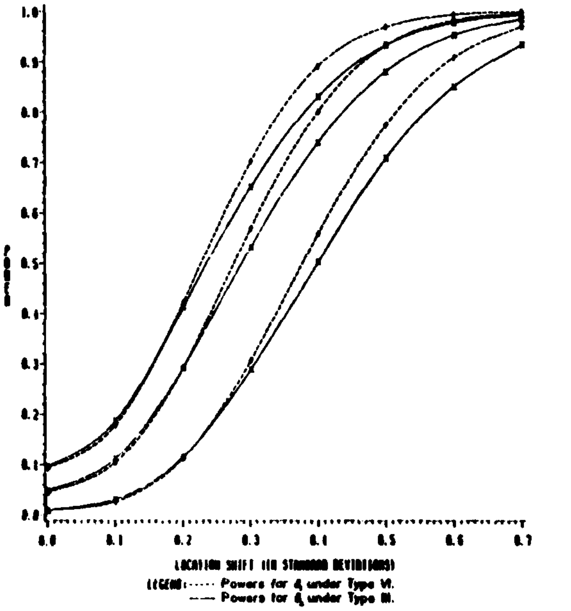


FIGURE 4

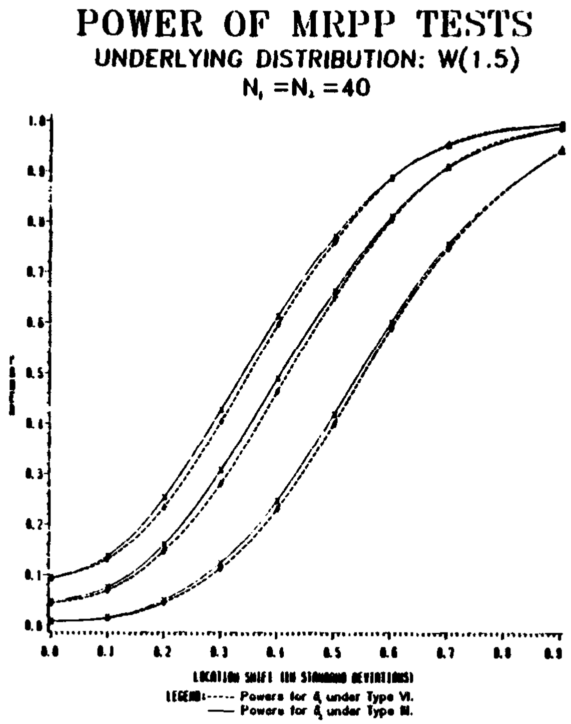

POWER OF MRPP TESTS
UNDERLYING DISTRIBUTION: W(1.5)
$N_1 = N_2 = 40$

FIGURE 5

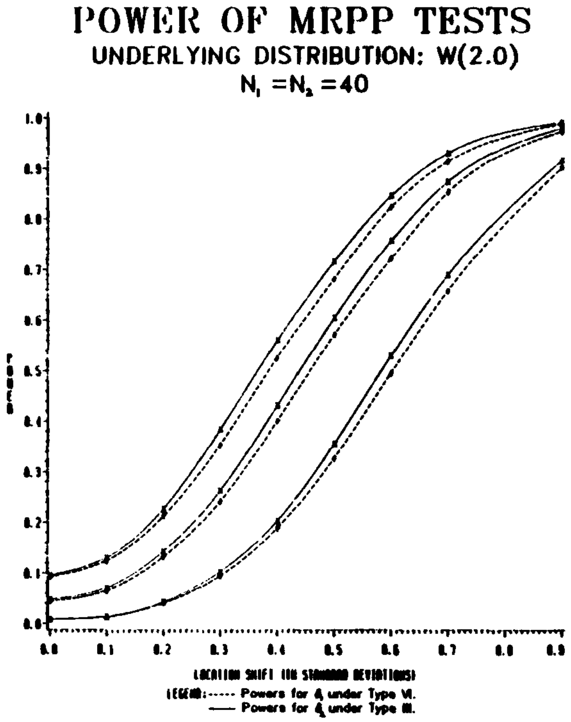

POWER OF MRPP TESTS
UNDERLYING DISTRIBUTION: W(2.0)
$N_1 = N_2 = 40$

FIGURE 6

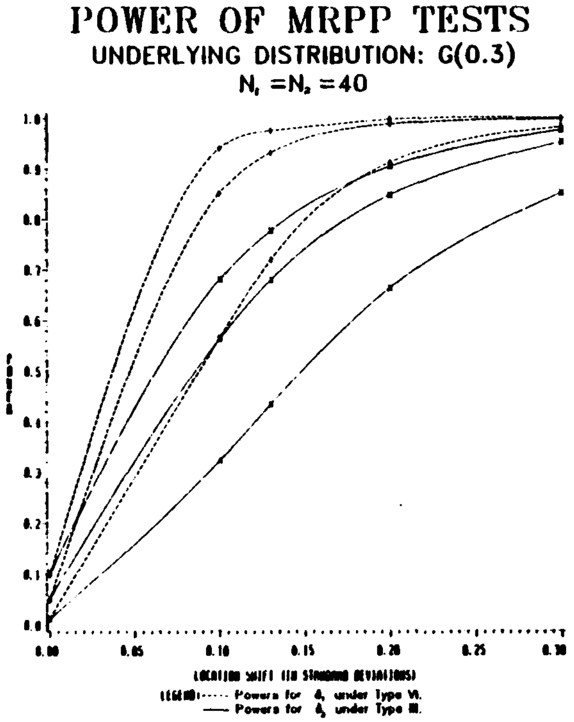

POWER OF MRPP TESTS
UNDERLYING DISTRIBUTION: G(0.3)
$N_1 = N_2 = 40$

FIGURE 7

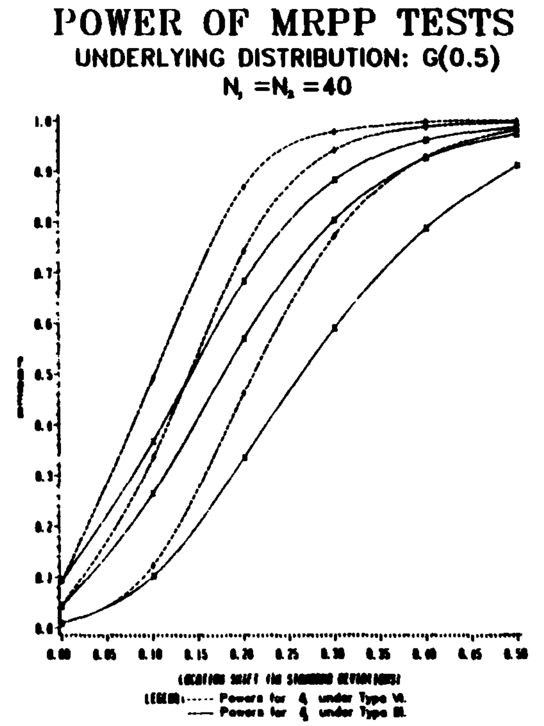

POWER OF MRPP TESTS
UNDERLYING DISTRIBUTION: G(0.5)
$N_1 = N_2 = 40$

FIGURE 8

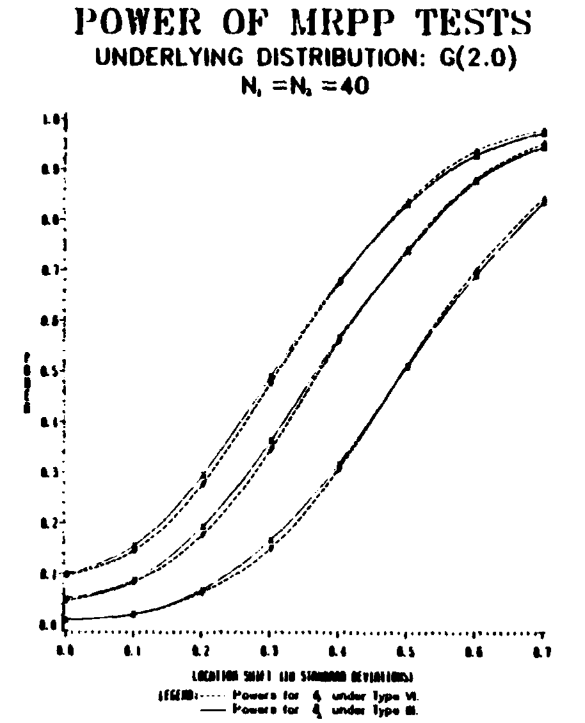


FIGURE 9

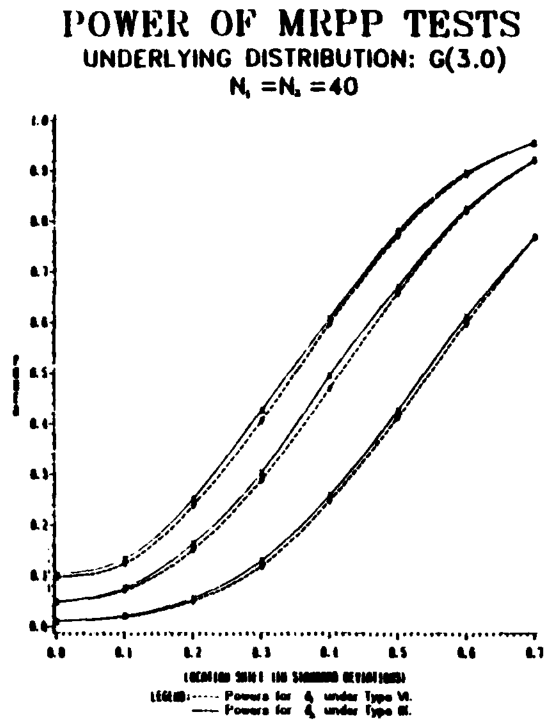


FIGURE 10

CONCLUSIONS

We observe from Tables I and II that for all underlying Weibull and gamma populations considered, the empirical power of δ_1 under Pearson Type VI approximation is always greater than that under the Type III approximation. We therefore plot the power curves of δ_1 under the Type VI approximation and those of δ_2 under the Type III approximation. These are shown by a dotted line and a solid line respectively in Figures 1 - 10. From the tables and the plots, we draw the following conclusions.

For low values of β, i.e., for very skewed populations, the powers of δ_1 and δ_2 increase very sharply. The powers get close to 1 by the time the location shift is 0.3σ for W(0.5) and G(0.3). The power of δ_1 is much greater than that of δ_2 for small location shifts. As β approaches 1, the sharpness in the increase of powers of δ_1 and δ_2 reduces gradually, with δ_1 always having greater power than δ_2.

For $\beta=1$ (the exponential population), the power of δ_1 is greater than that of δ_2 for location shifts larger than 0.2σ.

For Weibull populations, as β increases to 1.5, δ_2 has greater power than δ_1 for location shifts $\leq 0.6\sigma$, but for bigger location shifts δ has slightly more power. By the time β is 2, δ_2 has consistently higher power than δ_1.

For gamma populations, as β increases to 2, δ_2 has greater power than δ_1 for location shifts $\leq 0.4\sigma$, whereas δ_1 has greater power for larger location shifts. By the time β is 3, δ_2 has consistently higher power than δ_1.

Overall, it seems that δ_1 performs better for more skewed Weibull and gamma populations, but as the parameter increases and the distribution tends towards symmetry, δ_2 begins to perform better than δ_1.

ACKNOWLEDGMENT

The author is indebted to I. H. Tajuddin and S. K. Siddiqui for assistance in computation.

REFERENCES

Brockwell, P. J., Mielke, P. W. and Robinson, J. (1982). On non-normal invariance principles for multiresponse permutation procedures. Aust. J. Stat. 24, 33-41.

Mielke, P. W., Berry, K. J., Brockwell, P. J. and Williams, J. S. (1981). A class of nonparametric tests based on multiresponse permutation procedures. Biometrika 68, 720-724.

Mielke, P. W., Berry, K. J. and Johnson, E. S. (1976). Multiresponse permutation procedures for a priori classifications. Comm. Statist. A - Theor. Meth. 5, 1409-1424.

Tracy, D. S. and Tajuddin, I. H. (1985). Extended moment results for improving inferences based on MRPP. Comm. Statist. A - Theor. Meth. 14, 1485-1496.

Tracy, D. S. and Tajuddin, I. H. (1986). Empirical power computations of two MRPP rank tests. Comm. Statist. A - Theor. Meth. 15.

SOLUTION OF LEAST SQUARES EQUATIONS BY INCOMPLETE QR FACTORIZATION

J. W. Trigg, University of Michigan-Flint
A. K. Rigler, University of Missouri-Rolla

Introduction

Iterative procedures for solving large sets of sparse linear equations have been known for a very long time but only since the highspeed digital computer became available in the 1950's have these techniques been in popular use. The need to solve discrete analogs of partial differential equations of elliptic and parabolic type motivated much of the research directed toward analyzing and improving iterative schemes [3,12]. The least squares solution of rectangular systems of linear equations has received considerable attention also, but relatively little of the research has been focused upon iterative methods. Our purpose here is to suggest that the discretization of the variational problem associated with the differential equation leads to a natural formulation of a rectangular problem solved by an iterative procedure. Finally, the process may be applied to more general regression problems and can provide an "inner" iteration for the nonlinear case.

Iterative Methods for Solving Difference Equations

Our discussion starts with the solution of Laplace's equation. We propose first to solve $\nabla^2 W=0$ in a closed region with W taking on prescribed values on the boundary. In particular, we choose the "Model Problem", $\nabla^2 W=0$ on a square with W fixed on all four sides. Upon the square, a uniform grid of mesh length 1.0 is imposed.The grid points in the interior are numbered in a regular fashion; in our case, we use the left to right, top to bottom ordering of English text. The nine interior points shown below is the smallest number that includes all of the properties we wish to illustrate; the numerical examples to follow will be based upon Figure 1.

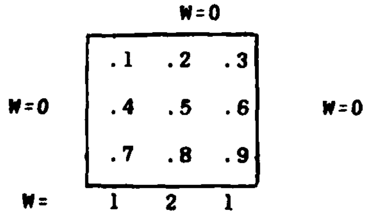


Figure 1

The finite difference stencil

$$\begin{array}{|c|c|c|}\hline & 1 & \\ \hline 1 & -4 & 1 \\ \hline & 1 & \\ \hline\end{array}$$

that produces a discrete approximation to the Laplace operator is centered over each mesh point, thus producing a set of nine linear equations in nine unknowns; AW=k.

$$\begin{bmatrix} 4 & -1 & & -1 & & & & & \\ -1 & 4 & -1 & & -1 & & & & \\ & -1 & 4 & & & -1 & & & \\ -1 & & & 4 & -1 & & -1 & & \\ & -1 & & -1 & 4 & -1 & & -1 & \\ & & -1 & & -1 & 4 & & & -1 \\ & & & -1 & & & 4 & -1 & \\ & & & & -1 & & -1 & 4 & -1 \\ & & & & & -1 & & -1 & 4 \end{bmatrix} \begin{bmatrix} W_1 \\ W_2 \\ . \\ . \\ . \\ . \\ . \\ . \\ W_9 \end{bmatrix} = \begin{bmatrix} 0 \\ . \\ . \\ . \\ . \\ 0 \\ 1 \\ 2 \\ 1 \end{bmatrix}$$

The matrix A is banded with five diagonals, it is tridiagonal by blocks associated with rows of the grid points, the diagonal blocks are tridiagonal in themselves, and finally the matrix A and each of its diagonal blocks are positive definite. The sparseness of the matrix is evident in the 9x9 example; for a grid with many points, our coefficient matrix A is mostly zero.

The sparsity of nonzero elements of a very large matrix leads one to use iterative methods of solution. The matrix A is split into two, A=D-C, so that the linear equations become DW=CW+k. If D^{-1} exists, as when D is the block diagonal partition,

$$W = D^{-1}CW + D^{-1}k.$$

The classic Jacobi methods have this form; point Jacobi has a strictly diagonal D while one line Jacobi uses the block diagonal form.

Several variations of this splitting technique have been studied such as Gauss-Seidel, Successive Overrelaxation, and Alternating Direction Implicit Methods. Much of the acceleration obtained by these variations depends upon the block tridiagonal nature of the difference equations and is not guaranteed in general. It is true, however, that for the most part, the value of the iteration method comes from reduced storage requirements and fewer arithmetic operations.

If the original, AW=k, were to be solved by direct elimination methods, one would ordinarily decompose A into triangular factors; for symmetric A, the Choleski method would be suitable. For banded matrices, factoring destroys sparsity between the extreme bands but does not disturb zeroes outside of the band. If $AW=k$ is written as $DW=CW+k$, the iteration would be carried out by a direct solution process where Choleski factoring is applied to D. For this reason, the iteration has been called Incomplete Choleski[7,9].

One point of view suggests that the iteration is successful because of the dominance of D. With this in mind, the splitting process may replace elements of R by zero when they are identified with small elements of A. The effect is to maintain a dominant and easily solved D (in factored form) at the expense of increased density of nonzero elements of C. The meaning of "smallness" is a matter of judgment[9].

Symmetric Matrices and Normal Equations

It is desirable that the coefficient matrix be symmetric and positive definite. This property has important consequences affecting the performance of the iterative procedure and the amount of computer memory and arithmetic operations required for its execution. Two methods of deriving the difference stencil that ensure symmetry have been proposed. Varga[11] applied Green's theorem to a mesh box with the grid point in its interior and mimicked by differences the normal derivatives that occur in the theorem. When applied to each grid point, a symmetric system of equations is generated.

A macroscopic application of Green's theorem was suggested by Engli, et al, and by Forsythe and Wasow[2,3]. The original differential equation is the Euler equation associated with a variational principle and they suggest that the functional be discretized before minimization. Then the set of linear equations to be solved is simply the set of normal equations associated with an overdetermined linear system to be solved by least squares. Thus, the coefficient matrix is sure to be symmetric and at least semidefinite. The positive definite property is obtained separately when using Varga's derivation.

For our problem, the basic functional is

$$J = 0.5\iint[(\partial W/\partial x)^2+(\partial W/\partial y)^2)]ds.$$

When this expression is discretized on the same grid as before, we have a quadratic form in the variables W_i, $i=1,2,\ldots,9$, whose minimum is located by solving $\partial J/\partial W_i=0$, $i=1,2,\ldots,9$; that is, the normal equations of an overdetermined linear system, $AW=k$. Because $A=F^tF$, we are guaranteed symmetry and semidefiniteness. In addition, if Choleski factoring is applied to A producing $A=R^tR$, it is a fact that orthogonal decomposition of F into QR, where $Q^tQ=I$, produces exactly the same R[8].

The derivation of a symmetric positive definite coefficient matrix from a variational principle is by no means limited to our Model Problem. Other boundary conditions applied to all or part of the boundary, for example a symmetry condition, will be treated in the discretization of the functional and symmetric coefficient matrices will always be the result. A more complicated functional with concomitant complications in the boundary conditions arises in mechanics [10,13]. The biharmonic equation $\nabla^4W = 0$ is the Euler equation associated with

$$J = \iint\{0.5[\partial^2W/\partial x^2+\nu\partial^2W/\partial y^2]^2$$
$$0.5(1-\nu^2)(\partial^2W/\partial y^2)^2$$
$$(1-\nu)(\partial^2W/\partial x\partial y)^2\}dA$$

$(0 < \nu < 1)$

The discretization of J is again a quadratic form that may be interpreted as normal equations of an overdetermined linear system.

Incomplete Factoring and Incomplete Orthogonalization

Choleski factoring of normal equations and orthogonal factoring of the overdetermined system have been successful as direct methods for linear least squares problems. Furthermore, an incomplete Choleski factoring avoids the "fill-in" of nonzero entries between the bands, locations which are occupied by zeroes in the unfactored matrix. This incompleteness induces a splitting of A to produce an iterative scheme. It is only natural to wonder if an incomplete orthogonalization of F might produce a successful iterative method to find the least squares solution of the overdetermined system.

A splitting of the rectangular matrix $F=H-G$ produces a normal equation matrix of the form

$$A=F^tF=(H^tH+G^tG)-(H^tG+G^tH).$$

A natural splitting of $A=D-C$ is to let $D=H^tH+G^tG$ and $C=H^tG+G^tH$. If D and C have suitable properties, e.g. form a regular splitting, then an iteration of the form

(Eq. 1) $DW_{m+1}=CW_m+k$

will converge [12]. If in addition, D is sparse and easily solved, for example diagonal or tridiagonal, the calculation is inexpensive.

Unfortunately, an iteration of the same nature applied to the rectangular system *will not solve the problem* unless the splitting is "proper"[1]. An easy way to reproduce (Eq. 1) in rectangular form is to double the row dimension. Write F=H-G and set HW=GW+k, GW=HW-k to be solved simultaneously. Thus the rectangular problem

$$\begin{bmatrix}H\\G\end{bmatrix} W_{m+1} = \begin{bmatrix}G\\H\end{bmatrix} W_m + \begin{bmatrix}k\\-k\end{bmatrix}$$

can be solved repeatedly. The normal equations for W_{m+1} have D for the coefficient matrix, etc. but can be solved by applying the QR algorithm to $\begin{bmatrix}H\\G\end{bmatrix}$.

The R produced in this manner is *exactly the Choleski factor of D*. For this reason we can call this an incomplete QR method.

A Specific Splitting for $\nabla^2 W=0$

On the 9x9 grid of Figure 1, the discrete analog of the functional is a set of linear equations FW=b.

$$\begin{bmatrix}
1 &&&&&&&& \\
-1 & 1 &&&&&&& \\
& -1 & 1 &&&&&& \\
&& -1 &&&&&& \\
&&& 1 &&&&& \\
&&& -1 & 1 &&&& \\
&&&& -1 & 1 &&& \\
&&&&& -1 &&& \\
&&&&&& 1 && \\
&&&&&& -1 & 1 & \\
&&&&&&& -1 & 1 \\
&&&&&&&& -1 \\
-1 &&&&&&&& \\
& -1 &&&&&&& \\
&& -1 &&&&&& \\
1 &&& -1 &&&&& \\
& 1 &&& -1 &&&& \\
&& 1 &&& -1 &&& \\
&&& 1 &&& -1 && \\
&&&& 1 &&& -1 & \\
&&&&& 1 &&& -1 \\
&&&&&& 1 && \\
&&&&&&& 1 & \\
&&&&&&&& 1
\end{bmatrix} \quad W \quad = \quad \begin{bmatrix}
0 \\ \cdot \\ \cdot \\ \cdot \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ 1 \\ 2 \\ 1
\end{bmatrix}$$

This set of equations consists of 24 rows and 9 columns. Let us choose the diagonal band from F commencing at row 16, column 1 to constitute our matrix G, H being the remaining portion. Then *the 48*9 system*

$$\begin{bmatrix}H\\G\end{bmatrix} W = \begin{bmatrix}G\\H\end{bmatrix} W + \begin{bmatrix}k\\-k\end{bmatrix}$$

may be solved iteratively. Each update is a least squares problem in its own right whose normal equations are

$$(H^tH + G^tG)W = (H^tH + G^tG)W + (H^tk - G^tk)$$

This process is precisely the Jacobi *single line iteration* applied to the square set of normal equations. When the matrix $\begin{bmatrix}H\\G\end{bmatrix}$ is factored by the QR method, the upper triangular matrix R is the Choleski factor of the diagonal partition of the normal equations corresponding with *one line* of grid points.

Nonlinear Least Squares

The solution of a nonlinear least squares problem is often obtained by the Gauss-Newton-Hartley method or Grey's[4] variation of it. Each of the functions in the overdetermined set is linearized at some nominal guess and a correction is obtained by solving the linear approximation as a conventional least squares problem. The coefficient matrix is the Jacobian of the original system and on occasion may be quite sparse. An iterative solution of the *linearized system* may be the most efficient way to proceed. The rectangular formulation as mentioned above might well be considered for the solution of the "inner iteration."

As an example, we might choose a standard test problem generally attributed to C. F. Wood [5]. It is often presented as

$$\text{Min } J = 100(x_2 - x_1^2)^2 + (1 - x_1)^2 + 90(x_4 - x_3^2)^2 + (1 - x_3)^2 + .2(x_2 -1)^2 + .2(x_4-1)^2 + 9.9(x_2 + x_4 - 2)^2$$

The rewriting as a sum of squares gives us the equivalent system of seven *equations in four variables.*

$$\begin{gathered}
10(x_2 - x_1^2) = 0 \\
1 - x_1 = 0 \\
\sqrt{90}\,(x_4 - x_3^2) = 0 \\
1 - x_3 = 0 \\
\sqrt{0.2}\,(x_2 - 1) = 0 \\
\sqrt{0.2}\,(x_4 - 1) = 0 \\
\sqrt{9.9}\,(x_2 + x_4 - 2) = 0
\end{gathered}$$

The Jacobian of this system is

$$\begin{bmatrix}
-20x_1 & 10 & 0 & 0 \\
-1 & 0 & 0 & 0 \\
0 & 0 & -2\sqrt{90}x_3 & \sqrt{90} \\
0 & 0 & -1 & 0 \\
0 & \sqrt{.2} & 0 & 0 \\
0 & 0 & 0 & \sqrt{.2} \\
0 & \sqrt{9.9} & 0 & \sqrt{9.9}
\end{bmatrix}$$

in which 18 of the 28 entries are zero.

The splitting of F is obtained by an incomplete factoring of

$$F = \begin{bmatrix} H \\ G \end{bmatrix} = QR - E,$$

where the r_{24} element of R is replaced by zero. For example, we evaluated the Jacobian at (0,0,0,0,) and carried out the decomposition. The Q, R, and E are shown below.

$$Q = \begin{bmatrix} 0 & .953 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & .948 \\ 0 & 0 & -1 & 0 \\ 0 & .0426 & 0 & 0 \\ 0 & 0 & 0 & .0447 \\ 0 & .299 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & .315 \end{bmatrix}$$

$$R = \begin{bmatrix} 1 & 0 & 0 & 0 \\ & 10.493 & 0 & 0 \\ & & 1 & 0 \\ & & & 10.004 \end{bmatrix}$$

$$E = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3.146 \\ 0 & 10 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 9.486 \\ 0 & 0 & -1 & 0 \\ 0 & .477 & 0 & 0 \\ 0 & 0 & 0 & .447 \\ 0 & 3.146 & 0 & 0 \end{bmatrix}$$

We executed the inner iteration from this starting point within the Gauss-Hartley scheme and found the solution (1,1,1,1) as expected.

Summary

By relating difference equations with normal equations through a variational principle, we have attempted to formulate an iterative procedure for a sparse overdetermined system in terms of conventional methods for solving difference equations. In particular, we demonstrate a Jacobi iteration in rectangular form. For large nonlinear problems with sparse Jacobians, such iterative processes may be useful in obtaining the updates in a Gauss-Newton-Hartley type algorithm.

REFERENCES:

1. Berman, A., and M. Neumann, "Proper Splittings of Rectangular Matrices," SIAM J. Appl. Math, Vol. 31 (1976), 307-312.

2. Engeli, M., Th. Ginsburg, H. Rutishauser, and E. Stiefel, Refined Iterative Methods for Computation of the Solution and the Eigenvalues of Self-Adjoint Boundary Value Problems. Birkhauser Verlag, Basel/Stuttgart, 1959.

3. Forsythe, G. E. and W. R. Wasow, Finite Difference Methods for Partial Differential Equations. John Wiley and Sons, Inc., New York 1960.

4. Grey, D. S, "Aberration Theories for Semi-Automatic Lens Design by Electronic Computers: I Preliminary Remarks; II A Specific Computer Program," J. Opt. Soc. Am. 53 (1963), 672-676, 677-680.

5. Himmelblau, D. M., Applied Nonlinear Programming. McGraw-Hill, New York 1972.

6. Jennings, A. and M. A. Ajiz, "Incomplete Methods for Solving $A^t Ax+b^*$," SIAM J. Sci. Stat. Comput. Vol. 5 (1984), 978-987.

7. Kershaw, D. S., "The Incomplete Choleski-Conjugate Gradient Method for the Iterative Solution of Systems of Linear Equations," J. Comput. Phys., Vol. 26 (1978), 42-65.

8. Lawson, C. L., and R. J. Hanson, Solving Least Squares Problems. Prentice-Hall, Inc., Englewood Cliffs, N. J., 1975.

9. Manteuffel, T. A., "An Incomplete Factorization Technique for Positive Definite Linear Systems," Math. of Comp. Vol. 34 (1980), 473-497.

10. Timoshenko, S. and S. Woinowsky-Krieger. Theory of Plates and Shells. McGraw-Hill, Inc. New York, 1959.

11. Varga, R. S., "Numerical Solution of the Two-Group Diffusion Equation in x-y Geometry," IRE Trans of the Professional Group on Nuclear Science, NS-4, 52-62 (1957).

12. Varga, R. S., Matrix Iterative Analysis. Prentice-Hall, Inc., Englewood Cliffs, N. J. 1962.

13. Weinstock, R., Calculus of Variations. McGraw-Hill, New York, 1959.

POLYHEDRON GRAPHS

Danny W. Turner, John W. Seaman, and Dean M. Young
Baylor University

ABSTRACT

This paper introduces a new multivariate graphical technique known as polyhedron graphics, or P-graphs. The independent variables are displayed as a polar projection in the base plane of a three dimensional coordinate system, forming a base polygon. A dependent variable is graphed above the origin of the polar projection. Lines are projected from the plotted dependent variable to the vertices of the base polygon, forming a polyhedron. An additional variable can be plotted below the base plane. Other dependent variables can be represented by moving the base polygon in the base plane, rotating the base polygon, etc. Results of a three-dimensional graphics implementation are presented along with an example from experimental design.

1. INTRODUCTION

In this paper we present a graphical procedure that can be used to display multidimensional data. A variety of such procedures already exist, and include such things as function plots, linear profiles, polar profiles(k-sided polygons), Chernoff-type faces, and so forth. The common mathematical basis that these specialized graphical methods have is that each can be used to map a point in k-dimensional space (R^k) to a subset of two-dimensional space (R^2) while attempting to preserve in some way the original information. Thus, a set of vectors $X_1,...,X_n$ with each $X_i \in R^k$ is mapped to a collection of subsets $S_1,...,S_n$ with each $S_i \in R^2$. These subsets can then be drawn using appropriate computer graphics devices and studied with the intent of extracting information about the original points.

It is obvious that there are an unlimited number of mappings like those described above. The purpose of this article is to present a new display technique that is essentially a simple generalization of one of the oldest and most popular existing methods, namely polygon graphs.

2. POLYGON GRAPHS

With roots going back to the 1950's, polygon graphs represent one of the earliest graphical attempts at displaying multivariate data. These graphs and numerous variations may be found under many names including k-sided polygons, stars, spiderwebs, polar profiles and sunflowers. Figure 1 shows a typical star representing a point in 12-dimensional space. The values of the twelve variables are transformed into the lengths of the twelve equally-spaced rays emanating from the polar origin. (The use of equally-spaced rays is common but obviously not required.)

Applications using star plots are plentiful. For typical examples refer to Chambers, Cleveland, Kleiner, and Tukey (1983) or Turner and Hall (1983). These plots have many advantages including ease of generation and interpretation. In the next section we introduce a simple generalization of star plots.

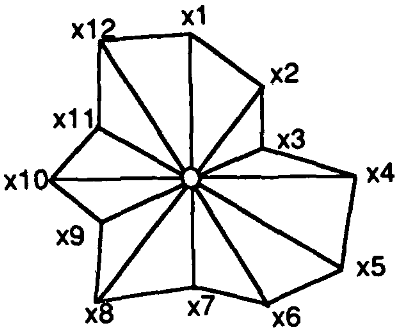


Figure 1. A typical polygon graph (star plot).

3. POLYHEDRON GRAPHS

Building on the idea of star plots (polygon graphs) we propose polyhedron graphs. These displays are formed by using the values of the variables in a k-dimensional vector to control the positions of the vertices of a polyhedron. Figure 2 displays a prototype polyhedron graph or P-graph as we shall refer to them.

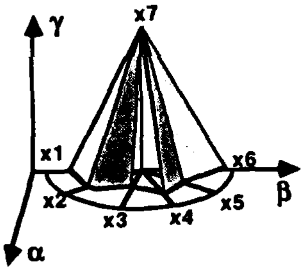


Figure 2. A prototype polyhedron graph (P-graph). Notice that the variables in the αβ plane produce a portion of a star plot. This partial star is called the base polygon. It is hidden somewhat be the shaded faces of the polygon.

Applications that will be particularly suitable for P-graphs will involve situations where there is a natural decomposition of the multiple variables into two groups. One group of variables will be coded into the base polygon and the other group into vertices not in the base polygon. An example would be multiple regression of one response variable on the k independent variables with the k independent variables controlling the base polygon and the response variable controlling one vertex outside the base polygon as in Figure 2. An alternative is to use the value of the residual at each observation to control the length of the ray outside the αβ plane. A second dependent variable can be represented by adding a vertex below the base polygon as shown in Figure 3. Pairs of P-graphs with common base polygons can be used to add more dependent variables.

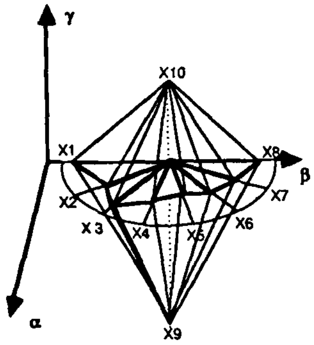


Figure 3. A P-graph with two variables represented outside the base polygon.

4. EXAMPLE

Evolutionary operation (EVOP) is a statistics based methodology for process improvement in the context of an operating full-scale process. A detailed exposition is found in Box and Draper (1969). Within this context consider a 2^3 factorial design with the three factors temperature, concentration and pressure and the response variable average yield.

Figure 4 shows a common way to display the multivariate data. As the produciton runs are made under the various treatment conditions the values of average yield are shown at the appropriate vertices.

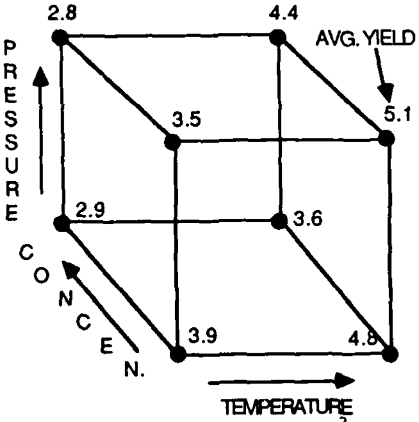


Figure 4. Cube display for a 2^3 factorial design in an EVOP application.

Suppose that EVOP is to be run using a 2^5 factorial design involving five factors and two response variables. The usual EVOP displays are not adequate for this situation. However, P-graphs provide a simple technique for displaying the results of the EVOP runs that is easy to understand by the process operators. A reference P-graph is shown in Figure 5. The independent variables are pressure, temperature, concentration of A, concentration of B, and concentration of C; the response variable are average yield and average tensile strength.

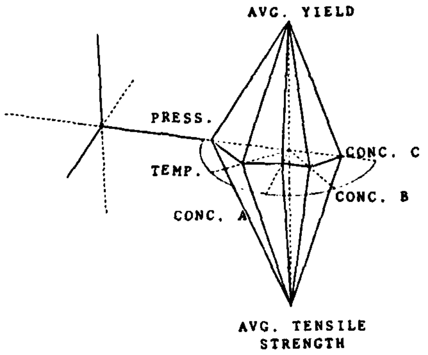


Figure 5. Reference P-graph for the EVOP example involving five independent variables and two response variables.

As each production run in the design is performed the corresponding P-graph would be displayed to the process operator (and updated in the case of replicates). The evolution of the process could easily be tracked and directed by observing the time series of P-graphs. A portion of such a time series is displayed in Figure 6. It is easy to visualize how yield and tensile strength are changing with respect to the changes in the levels of the independent variables. We feel this technique would be quite valuable to a trained process operator.

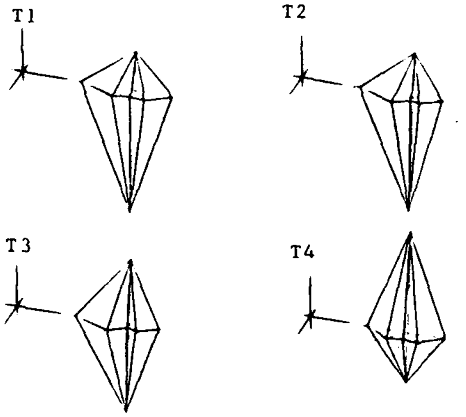


Figure 6. Selected P-graphs for the EVOP example. $T_1 < T_2 < T_3 < T_4$ are the corresponding production run times.

REFERENCES

Box, G.E.P., and Draper, N.R. (1969), Evolutionary Operation: A Statistical Method of Process Improvement, New York: Wiley.

Chambers, J.M., Cleveland, W.S., Kleiner, B., and Tukey, P.A. (1983), Graphical Methods for Data Analysis, California: Wadsworth.

Turner, D.W., and Hall, K.A. (1983), "GRAPHPAK: Interactive Graphics for Analysis of Multivariate Data," Proceedings of the Fifteenth Symposium on the Interface of Computer Science and Statistics, editor J.E. Gentle, pp. 328-332.

EFFICIENT ESTIMATION & TESTING FOR HETEROSCEDASTICITY WITHOUT AUXILIARY VARIABLES

H.D. Vinod (Fordham University) and Aman Ullah (University of Western Ontario)

1. INTRODUCTION

The available methods for estimation and testing in the presence of heteroscedastic errors, Vinod and Ullah (1981), need a specific parametric form of heteroscedasticity. Waldman (1983) shows "algebraic equivalence" of White's(1980) test with certain versions of Godfrey and Breusch-Pagan tests which rely on auxiliary variables. In this paper we propose a new two-step generalized least squares (GLS) estimator which is consistent and asymptotically efficient. In the first step we develop J.N.K. Rao (1973) type modified minimum norm quadratic estimator (MINQE) of the unknown heteroscedastic variances based on replicated observations for the variables in the model. The replicated observations in our paper are created in the framework of Vinod's (1982a,b) use of the additional information contained in the fact that only a certain number of digits in the variable data series are reliable, and beyond which there is fuzziness. Latin squares style replications, which are well-known in statistical design of experiments literature, Kendall and Stuart(1979), are then used. For testing of heteroscedasticity we propose appropriate test statistics.

2. THE MODEL AND ESTIMATORS

Consider the usual regression model

$$y = X\beta + u \qquad (2.1)$$

where y is a Tx1 vector X is a Txp matrix of p regressors, β is a px1 vector of unknown regression coefficients, and u is a Tx1 vector of disturbances such that

$$Eu = 0 \text{ and } Euu' = \text{Diag}(\sigma_{11}, \sigma_{22}, \ldots, \sigma_{TT}) = \Sigma \qquad (2.2)$$

The usual OLS estimator of β in (2.1) is given by

$$b = (X'X)^{-1}X'Y, \qquad (2.3)$$

which is unbiased with the variance-covariance matrix given by

$$V(b) = (X'X)^{-1}X'\ \Sigma X(X'X)^{-1} \qquad (2.4)$$

The GLS estimator of β is

$$\hat{\beta} = (X'\ \Sigma^{-1}X)^{-1}X'\ \Sigma^{-1}\ y \qquad (2.5)$$

with the corresponding variance-covariance matrix

$$V(\beta) = (X'\ \Sigma^{-1}\ X)^{-1}. \qquad (2.6)$$

It is well known that $V(b) - V(\hat{\beta})$ is non-negative definite.

In practice, Σ is rarely, if ever, known hence $\hat{\beta}$ is not operational. The published literature contains several techniques for relating the heteroscedastic variances to some auxiliary variables to estimate Σ, and then use "estimated" GLS estimators. White(1980) proposes a consistent estimator of $X'^{*}\Sigma X$ in the middle of the V(b) expression to develop a test based on OLS residuals. His procedure involves inefficient OLS estimator since no rigorous estimator for Σ is suggested. White's method fails if σ_{tt} may be a cosine or sine wave in a trend variable not included as one of the regressors. Cragg(1983) can handle such arbitrary forms, provided they are explicitly stated in terms of auxiliary variables, not otherwise.

2.1 MINQE Style Estimation of Σ

To obtain efficient estimates of β in (2.1) using (2.5) we need to estimate Σ denoting by σ a Tx1 vector of the diagonal elements of Σ. We have

$$\hat{u} = y - Xb = Mu,\ M = I-H, \text{ and } H = X(X'X)^{-1}X' \qquad (2.7)$$

where $\hat{u}$ is a Tx1 vector of ordinary least squares (OLS) residuals, and H is the hat matrix. From (2.2) note that E(u) = 0, E(uu') = M ∑ M.

Matching the diagonal elements we have the algegraic result

$$E\ \hat{u} = Q\sigma, \qquad (2.8)$$

where û = u * u, Q=M * M are Hadamard products, that is, obtained by replacing each element of u and M by their own squares. Denoting $n = \hat{u} - E\hat{u}$, we have a regression equation:

$$\hat{u} = Q\sigma + n, \qquad (2.9)$$

where σ is the set of T unknown regression coefficients. Although the M matrix is singular, Q may be assumed to be non-singular. Now we will note that the OLS estimator

$$\hat{\sigma}_{OLS} = (Q'Q)^{-1}Q'u = Q^{-1}u \qquad (2.10)$$

is unbiased but is inconsistent even when X is non-stochastic and (X'X)/T is finite. In the case where replicated data are available, there is no inconsistency problem with Rao's (1970) MINQE as noted by Horn, et al (1975) and J.N.K. Rao (1973). This estimator is given below.

Denote y_j a Tx1 vector of elements y_{tj}, where y_{tj} is the t^{th} observation on the dependent variable for the j^{th} replication, j=1, 2,..., J. Define

$$Q_J = (I-J^{-1}H) * (I-J^{-1}H) + (J-1)J^{-2}H*H \qquad (2.11)$$

where * denotes the Hadamard product as before. Now the MINQE [see Horn, et al (1975, p. 381)] is given by

$$\hat{\sigma}_{MQ} = Q_J^{-1}\sum\hat{u}_jJ^{-1} \qquad (2.12)$$

where the summation is from j=1 to j=J, and $\hat{\hat{u}}_j = \hat{u}_j * \hat{u}_j$; $\hat{u}_j = (I-H_j)y_j$ is a Tx1 vector of residuals from the j^{th} replication, with elements $\hat{u}_{tj}$. For J=1, (2.12) reduces to (2.10). The estimator in (2.12) is consistent when J increases without bound.

A well-known problem [Horn, et al (1975)] with MINQE estimator $\hat{\sigma}_{OLS}$ or $\hat{\sigma}_{MQ}$ is that it usually yields at least some negative estimates for σ, which should be strictly positive. J.N.K. Rao's (1973, 1980) modification of MINQE called average of squared residuals (ASR) solves the negative variance problem, and has desirable mean squared error (MSE) properties and rather simple computation. First, we compute for each of the J replicates

$$\hat{u}_j = y_j - Hy_j \qquad (2.13)$$

the Tx1 vector of residuals for the j^{th} replicate. Now we have the ASR estimator immediately as

$$\hat{\sigma}_{ASR} = (1/J) \sum \hat{u}_j. \qquad (2.14)$$

Since σ_{ASR} is based on a sum of squares, it will obviously be positive. Note that (2.14) is an approximation of MINQE in (2.12). This is because $M_j \doteq I$ in (2.12), by neglecting the terms of the lower order in T.

2.2 Latin Square Style Replications and Estimation of ∑

Note that the ASR estimator requires replicated observations which are usually unavailable in econometrics. Any randomized scheme of generating replications is usually unacceptable because each run of the same date then yields a different estimate. Though the choice of our Latin Square style is somewhat arbitrary, it uses the fuzziness range of the data to yield unique results with desirable properties.

Following Vinod (1982a) and Vinod (1982b) we note that the observed values of regressors can be written as

$$|x_{ti}| - 0.5(10)^{h_i} \le |x_{ti}| \le |x_{ti}| + 0.49(10)^{h_i} \qquad (2.15)$$

where h_i represents the number of "significant" or "reliable" digits to the right of the decimal point. For example, if the observed data are 29.7 (h_i = 1) it may have been anywhere in the range 29.65 to 29.749 (= 29.7 + .49 x .1) and rounded to 29.7 by the usual rounding methods. We choose J non-stochastic numbers from this range to construct our replications. Let C_j denote a T x p matrix of corrections, with elements c_{tij} lying in the range: $(-d_i$ to $d_i)$ or

$$c_{tij} - d_i \le c_{tij} \le c_{tij} + d_i$$

where $d_i = 0.49\ (10)^{-h_i}$, and where j= 1,2,..,J. Starting with the observed matrix of regressors X we construct replications $X_j = X + C_j$ of non-stochastic regressor matrices. The range in (2.15) is divided into J equal components, or may be based on quantiles of an appropriate distribution.

In natural sciences the measurements involving temperature, pressure, weight, etc. all have an identifiable "true" value, and there is a clearcut meaning to the word measurement error. In econometrics, for example, for many (aggregative) variables such as gross national product, implicit price deflator, unemployment rate, etc. it is fair to say the "true" value of the variable itself is fuzzy. All the values in a fuzziness range are almost equally feasible, and (market) agents react to the "reported" values of variables with skepticism. By contrast, a natural substance reacts to the true temperature, weight etc. of another substance, and the fact that the engineer makes an error in measuring the temperature is of no consequence to the physical interaction of the substances. This is a major distinction which must be recognized. The true values of the variables could be anywhere in the fuzziness range, and the market agents also treat them as - such. If a fitted regression equation is overly sensitive to changes within the fuzziness range, (i.e., not "smooth") the market agents will reject such models. The discussion of measurement errors from natural sciences needs to be modified in some social science applications.

In spectral analysis of time series assumption that the spectrum should be smooth is similar to Vinod's (1982) assumption that the regression estimates should be "smooth", or not too sensitive in the fuzziness range. One does not reject the spectral analysis or kernel estimates of density functions simply because there is a large variety of plausible "window" specifications. It is well- known that in agricultural experiments the Latin Square design eliminates the "fertility gradient" associated with the rows and columns of agricultural plots. We provide unique estimates for given data, and eliminate the effect of two coordinates associated with the specific observation number (rows), and the specified variable (column) used.

Since the fuzziness in the dependent variable y is essentially similar to regressor fuzziness, it is convenient to augment the X matrix by including the additional column for y. We write $X^a = [\underline{X}:y]$. The choice of J, the number of replications to be created depends on several practical considerations. The J should be larger than p+1, enough to provide consistent estimates, and small enough to impose a reasonable computational burden. If J is an integer multiple of p+1 the construction of (p+1) x (p+1) Latin Square specification is most convenient. We describe the simplest case of p=3, J=4, T=8 for convenience.

The assumed fuzziness range in (2.15) is divided into J equal parts. For our example with J=4, let us choose four constants $k_a = -d_i$, $k_b = -d_i\ /2$, $k_c = +d_i/2$, and $k_d = d_i$. Now the original data of the (augmented) X matrix of dimension 8 x 4 are modified by adding an 8 x 4 matrix to yield the first replication. For convenience we report only the subscripts of k in a 8 x 4 matrix which represents the first replication, j=1.

$$\begin{bmatrix} a & b & c & d \\ b & c & d & a \\ c & d & a & b \\ d & a & b & c \\ a & b & c & d \\ b & c & d & a \\ c & d & a & b \\ d & a & b & c \end{bmatrix}$$

Note that we have used two "standard" 4 x 4 Latin Squares to generate the 8 x 4 matrix. If T=9 it is obvious that an additional row will have (a b c d) as subscripts of k. For any T each column above will be a string of (a b c d) with appropriate starting points. For the second replication j=2 the subscript of k in the top left corner is b and the first row is (b c d a). Each column is now a string of (a b c d) with these starting values. For the third and fourth replication the additive constants start with k_c and k_d. If J=5, we would cycle the subscripts (a b c d e). In general, one can devise such strings for any J. Denoting the j^{th} replication of X by X_j and the j^{th} replication of y by y_j, we can thus generate J sets of replicated data.

Each replication of X_j will imply a new well known idempotent matrix M_j = $[I - H_j]$ with the residual vector u_j = $y_j - H_j y_j$, where $H_j = X_j(X'_j X_j)^{-1} X'_j$ is the hat matrix., and this is used in (2.14) to get

$$\tilde{\sigma}_{ASR} = (1/J)\sum \hat{u}_j^2, \hat{u}_j = y_j - H_j y_j \quad (2.16)$$

An alternative derivation of the proposed estimator in (2.16) is given below.

For the j^{th} replication let us write from (2.9)

$$\hat{u}_j = M_j \sigma + \eta_j. \quad (2.17)$$

Further, for j=1,...,J we can write (2.17) in the pooled regression form as

$$u^* = Q^* \sigma + \eta^* \quad (2.18)$$

where $u^* = [\dot{u}'_1, \ldots, \dot{u}'_J]'$, $Q^* = [Q'_1, \ldots, Q'_J]'$ and $\eta^* = [\eta'_1, \ldots, \eta'_J]'$. The pooled OLS estimator of σ in (2.18) is then

$$\tilde{\sigma} = (Q^{*\prime} Q^*)^{-1} Q^{*\prime} \dot{u}^* \quad (2.19)$$

Now note that $Q_j \doteq I$, by neglecting the terms of the lower order in T. Thus, substituting $M_j \doteq I$, we get an approximation of (2.19) which is given in (2.16).

3. GLS ESTIMATION AND A TEST FOR HETEROSCEDASTICITY

3.1 "Estimated" GLS Estimation

We note that the $\tilde{\sigma}_{ASR}$ in (2.16) generates an estimate $\sum$ of the diagonal matrix of heteroscedastic variances, by substituting $\sigma_{ASR,t}$ value as the t^{th} term along the diagonal. Thus we do not need a further uncertain search for appropriate combination and/or transformation of auxiliary variables to estimate the heteroscedastic variances.

To obtain the GLS estimator we proceed as follows. The J replications of the Txp data matrix X are rearranged such that

$$y_j = X_j \beta + u_j, \; j=1\ldots,J \quad (3.1)$$

where X_j is the j^{th} Txp matrix of J replicated observations on p regressors, and u_j is a Tx1 disturbance vector such that

$$Eu_j = 0 \text{ and } Eu_j u'_j = \sum \quad (3.2)$$

The model in (3.1) can be written in a more compact form as

$$y^* = X^* \beta + u^* \quad (3.3)$$

where $y^* = (y'_1, y'_2, \ldots, y'_J)$, $X^* = [X'_1, \ldots, X'_J]'$, and $u^* = [u'_1, \ldots, u'_J]'$ such that

$$Eu^* = 0 \text{ and } Eu^* u^{*\prime} = \sum \otimes I_J \quad (3.4)$$

The estimated GLS estimator is then given by

$b^* = (X^{*\prime} (I_J \otimes \tilde{\Sigma}^{-1})X^*)^{-1} X^{*\prime} (I_J \otimes \tilde{\Sigma}$
$1)y^*$ (3.5)
where σ_{ASR} is given in (2.16).

It follows from a theorem in Fuller and Rao(1978, p. 1152) that the estimator b* is consistent and that √ (JT) (b* - β) has a limiting normal distribution with mean 0 and covariance matrix for J ≥ 3 given by a limit as T increases without bound

$$\lim JT[(1+2J^{-1} - 8J^{-2}) \quad (X^{*\prime} I_J \otimes \tilde{\Sigma}^{-1} X^*)^{-1} + 4J^{-2}\Omega\ ] \qquad (3.6)$$

where

$$\Omega = (X^{*\prime}X^*)^{-1} X^{*\prime} I_J \otimes \tilde{\Sigma} X^* (X^{*\prime}X^*)^{-1} (3.7)$$

Thus the asymptotic variance-covariance matrix of b* is

$$V(b^*) = (1 + 2J^{-1} - 8J^{-2}) \ (X^{*\prime} \tilde{\Sigma}^{-1} \otimes I_J X^*)^{-1} + 4J^{-2}\Omega \qquad (3.8)$$

and for large J

$$V(b^*) \doteq (X^{*\prime} \tilde{\Sigma}^{-1} \otimes \ I_J X^*)^{-1} \qquad (3.9)$$

One method of checking whether the estimated GLS estimation has improved the matters is to find the eigenvalues of the difference between the asymptotic covariance matrix of the OLS estimator b in (2.4), and that of b*. If the efficiency has been improved the eigenvalues of the estimated difference V(b) - V(b*) should be positive. A more crude assessment may be based on a comparison of standard errors of regression coefficients, if computer programs for eigenvalue computation are unavailable.

3.2A Test for Heteroscedasticity

A simple test is proposed here for the null hypothesis of homoscedasticity

$$H_o: \sigma_{11} = \sigma_{22} = \cdots = \sigma_{TT} = \sigma^2 \quad (3.10)$$

From the normality assumption on u it is clear that $\sum \ u_{tj}^2 \ /\sigma^2$ is a Chi-square random variable χ_J^2 with J degrees of freedom (df). Even under the null hypothesis (3.10) of homoscedasticity, our estimates of σ_{tt} will be random variables and will not all be identical to each other. If σ^2 is known, estimates $J\bar{\sigma}_{ASR,t}/\sigma^2 = \sum u_{tj}^2/\sigma^2$ will be random observations from a χ_J^2 parent distribution for large T. If σ^2 is not known, then again under the null hypothesis the ratios $w_{ASR,t}$ defined by $J\sigma_{ASR,t}/s^2$ can be considered as observations from a random variable which is approximately Chi-square with J df; s^2 is an estimate of the common variance σ^2 given by

$$s^2 = (y-Xb)'(y-Xb)/(T-p). \quad (3.11)$$

Then the empirical distribution function (edf) of these ratios should resemble that of a χ_J^2 variable. A "goodness of fit" test procedure as in Kendall and Stuart (1979, Ch. 30) is to rearrange the above ratios in an ascending order of magnitude:

$$w_{ASR,1} \le \cdots \le w_{ASR,T}. \qquad (3.12)$$

Next we evaluate the cumulative density of X_J^2 variable denoted by $Z_{ASR,t}$ evaluated at the "order statistics" in (3.12).

Now the Cramer-von Mises test statistic is

$$W^2 = (1/12T) + \sum(Z_{ASR,t} - (2t-1)/2T)^2. (3.13)$$

where summation is from t=1 to t=T. The 5% point for this statistic is based on an approximation, and for large samples is 0.461. Thus there is no need to look up any tables. This test is intended to be a refinement to Bartlett's well-known test for homoscedasticity. The W^2 test has higher power than the well-known Kolmogorov statistics based on the largest absolute differences instead of the square terms in (3.13). Clearly, even further refinements to W^2 test are possible, but may not make a practical difference.

AN ILLUSTRATION

The theoretical developments discussed above are illustrataed with the help of an example from Pindyck and Rubinfeld (1981, p. 169). The results show considerable reduction in standard errors with similar magnitudes of regression coefficients. The details can be obtained from the authors upon request.

5. CONCLUSION

We show that a minimum norm quadratic estimator (MINQE) of heteroscedastic variances can be derived by using non-stochastic replications from a fuzziness range discussed in Vinod (1982a,b). Further, using this we provide the efficient "estimated" GLS estimator of the regression coefficients.

The proposed test is based on the fact that the ratio of ASR estimate of heteroscedastic variance (based on our Latin Square type replications of X, y data) to the usual estimate of s^2 of residual variance, is approximately Chi-square variable with J (number of replications) degrees of freedom. We use the Cramer-von Mises W^2 test of "goodness of fit" for observed order statistics of the ratios mentioned above. The proposed estimation is the two-step GLS procedure. This has been introduced in the case of completely unknown form of the heteroscedasticity. We avoid the potential specification error in choosing auxiliary regressor or other variables related to heteroscedasticity testing or estimation

REFERENCES

Cragg, J. (1983), "More Efficient Estimation in the Presence of Heteroscedasticity of Unknown Form," Econometrica, 51, 751-763.

Fuller, W.A. and J.N.K. Rao (1978), "Estimation for a Linear Regression Model with Unknown Diagonal Covariance Matrix," The Annals of Statistics, 6, 1149-1158.

Horn,S.D., R.A. Horn and D. B. Duncan (1975), "Estimating Heteroscedasticity Variances in Linear Models," Journal of the American Statistical Association, 70, 380-385.

Kendall, M.G. and A. Stuart (1979), The Advanced Theory of Statistics, Vol. 2, 4th edition (New York, N.Y.: Macmillan).

Pindyck, R.S. and D.L. Rubinfeld (1981), Econometric Models and Economic Forecasts (New York, N.Y.: McGraw-Hill).

Rao, C.R. (1970), "On the Estimation of Heteroscedastic Variances", Biometrics, 29, 11-24.

Rao, J.N.K. (1973), "On the Estimation of Heteroscedastic Variances," Biometrics, 29, 11-24.

Rao, J.N.K. (1980) "Estimating the Common Mean of Possibly Different Normal Populations: A Simulation Study," Journal of the American Statistical Association, 75, 447-453.

Stephens, M.S. (1974), "EDF Statistics for Goodness of Fit and Some Comparison," Journal of the American Statistical Association, 69, 730, 737.

Vinod, H.D. and A. Ullah (1981), Recent Advances in Regression Methods (New York, N.Y.: Marcel Dekker, Inc.).

Vinod, H.D. (1982a), "Maximum Entropy Measurement Error Estimates of Singular Covariance Matrices in Undersized Samples," Journal of Econometrics, 20, 163-174.

Vinod, H.D. (1982b), "Enduring Regression Estimator," in O.D. Anderson, ed., Time Series Analysis: Theory and Practice 1 (Amsterdam: North Holland), 397-416.

White, H. (1980), "Heteroscedasticity - Consistent Covariance Matrix Estimator and Direct Test for Heteroscedasticity," Econometrica, 48, 817-838.

Waldman, D.M. (1980), "A Note on Algebraic Equivalence of White's Test and a Variation of Godfrey/Breusch-Pagan Test for Heteroscedasticity," Economics Letters, 13, 197-200.

The Selection of a Statistical Data Analysis Computing Environment for an Industrial Research Laboratory

James G. Wendelberger and Michael A. Wincek, General Motors Research Laboratories

1 Introduction

The statistics community is beginning to take advantage of the vast computational ability that is now surfacing in the computer industry. Many new algorithms and techniques now exist that were impractical or impossible to execute on computers of just a few years ago. Moreover, the enormous data acquisition and storage capabilities of the computer have allowed statisticians to confront problems that are increasingly more complex, both in the number and type of questions asked and in the size and structure of the data base used to answer the questions. The pencil-and-paper methods that were characteristic of statistics twenty to thirty years ago and that are still taught in many of today's textbooks are inadequate to deal with such problems because these methods are largely concerned with simple models and small data sets. Computerization and extension of these methods have allowed more complex models and larger data sets to be analyzed, but the line of development characterized by batch computing and the use of statistical packages has been exhausted both in efficiency and capability.

The purpose of this paper is to briefly state the statistical computing environment (hardware, operating system, and application software capabilities) that is necessary for a statistician or an experienced data analyst to efficiently deal with today's problems and to select a currently available environment which we judge best meets our criteria. Because of rapid developments in hardware and software, any selection may very likely be outdated (i.e. able to be improved upon) in three or four years. Hence, it is important to choose extendable software and updateable hardware, both of whose course of growth has been and will continue to be on the state-of-the-art development trajectory.

Some broad requirements of a statistical computing environment are given in Section 2. Operating systems and statistical software are discussed in Section 3. Some techniques that are currently being used by researchers in statistical computation are given in Section 4. The new equipment needed to implement these techniques is given in Section 5. In Section 6 we give our specific choice of available hardware and software for our statistical computing environment. Since we are statisticians and not computer scientists, our comments are focused on the functionality of the computing environment and not on the technical aspects of the software and hardware. References are provided to the technical computer science details.

2 A Statistical Computing Environment

For the purpose of this paper, a computing environment consists of hardware, an operating system, and software; the distinction between the last two is not always clear. A statistical computing environment should aid the model development process and the presentation of intermediate and final results. The model development process includes: model specification, estimation, and criticism. The model building process is iterative, the number and size of the iterations being determined in part by the complexity of the problem and the data set. It requires the ability to cull large data sets to find the relevant data, to display complex structure in multidimensional data, to interactively direct the course of the analysis, and to carry out computationally intensive methods. The computing environment should allow quick and efficient passage through the model building process, especially in the early stages when the problem is not well-defined. Capability, in the diminished sense of just being able to perform a task (irrespective of the amount of time it takes), is not adequate. Fast execution (less than a second) of commands is essential for full productivity of the analyst. Quick response time is needed so as not to inhibit problem solving activity. High speed is also essential for full use of interactive graphics, one of the most powerful tools in exploratory data analysis. McDonald and Pedersen (1985) point out that to draw a three-dimensional scatterplot containing 1000 points could require the graphics processor to draw at a speed of at least 3 million pixels per second. High speed color graphics requires very fast processing and large storage capabilities. McDonald and Pedersen (1985) give guidelines on the computational, graphics processing, and graphics display speeds.

The development, use, and maintenance of software are central issues to the development and application of statistics and data analysis techniques. Solutions to statistical problems often require that standard techniques be put together in slightly new ways tailored to the specific problem of interest. A good statistical computing environment will aid this activity.

A common tool for analysis is a statistical package developed for a batch mode environment (in contrast to an interactive environment). Typically a statistical package provides a set of commands to carry out certain procedures. The commands usually produce a predetermined output and usually cannot be combined with other commands to form new procedures. Flexibility or fine tuning of the command is achieved through a predetermined list of options. In anticipation of various possible outcomes, a typical style of analysis is to request "everything" from the commands in a statistical package, in order to submit the command once. This "shotgun" approach to analysis forces the user to look through the output for the relevant pieces. For the inexperienced analyst or the infrequent user of statistics, such an approach may be beneficial if it teaches the user about a new technique or makes him aware of the limitations of the analysis and the data. However, the experienced analyst (for whom the computing environment described here is intended) generally wants more direct and immediate control over the analysis and does not want to be forced into a particular mode. In the analysis of very large data sets it is impossible to effectively use the shotgun approach. Where the statistician may prefer to interactively direct the course of analysis and select output displays of interest, such capability may be of limited use to the inexperienced analyst. The style of analysis provided by a statistical package may be the best way for such a person to analyze data, but it is only one mode the statistician may choose.

The rather fixed, stand alone nature of the commands in a statistical package does not allow the easy creation of new procedures. One is usually restricted only to what the package can do. An example of a statistical language which does not have many of the limitations of statistical packages is S, an interactive environment for data analysis and graphics. See Becker and Chambers (1985) for a good review of the philosophy and capability of S. S is both a very high-level language for doing computations and is an environment which supports data management, documentation, and graphics. S graphics does not currently make use of the latest high resolution, high speed raster graphics display devices available on workstations, but future plans are aimed in that direction.

Analysis of data also frequently requires that new programs be created, so-called software engineering. Programs can be created by putting together existing software or by writing entirely new software. A statistical computing environment should recognize this need and provide software development and debugging tools to efficiently produce new software. Since a resulting program is often specific to a problem and may not be of general use, it is very important that the development be done quickly. If it takes too much time, it will not be done, possibly lowering the quality of the analysis. Ideally, the new software should be developed in the context of a statistical language, such as S, so that already existing input/output and graphics routines can be easily used and development time can be reduced.

Graphical displays are an absolute necessity for data analysis. As noted by Chambers et al. (1985), "there is no single statistical tool that is as powerful as a well-chosen graph." The human mind is far superior to any computer system in its ability to detect patterns. The use of color, dynamic displays, and other enhancements to a graph have great potential in aiding the analysis of the data. Creating graphics to exploit the great pattern discovery capabilities of the human visual system is an area of current research.

Documentation of results and the ability to produce manuscripts of high typographic quality in a timely fashion are essential to the success of a statistical project. High typographic quality is required since statistical papers typically consist of mathematical equations with Greek letters, special symbols, subscripts, superscripts, arrays, tables, charts, and the like. The production of such papers have typically been done by secretaries. As capable as they may be, the system of first writing the paper by hand and then giving it to a secretary to type is inefficient. Often, especially in mathematical writing, new errors are introduced into the paper in transcription. Much time may be consumed in explaining what is meant and how it should be typed. The researcher may be reluctant to revise because of increased burden to the secretary who must typically service many people. A statistical computing environment must include interactive tools that enable the analyst to document results and produce high quality papers.

Coincident with the documentation of results is the presentation of results. Written material and graphical displays may be presented on paper, transparencies (foils), slides, movie film, or some other media. Access to appropriate output devices is necessary.

Since many statistical projects are interdisciplinary, e.g. involving engineers, statisticians, and other scientists, data and results must be easily communicated. This can easily be done by means of a computer network. Data can be transferred via tape, but the time, bother, and inaccessability of tape inhibits its use. Networks encourage frequent communication of rather small messages and allow for the rapid dissemination of information. Networks are also important in the configuration of hardware. This will be discussed in Section 5.

A good statistical computing environment must allow the statistician to interactively formulate models, to efficiently and quickly analyze data, to document and display results, and to communicate these results at all stages of the analysis. Such an environment would provide fast execution time and fast program development time. As has been noted, a fast machine is not enough. The hardware and software must be considered together.

3 Operating Systems and Software

A statistical research group often develops software to develop and implement new techniques. A statistical consulting group often needs to modify or combine standard programs to solve a particular problem. Both groups need an operating system with tools that facilitate such activities. Since the operating system runs programs, manages the computer's resources, and provides an interface between the computer and the user, the choice of an operating system is crucial to the efficiency and productivity of the statistician. An excellent and, in our opinion, the best available operating system for statistical work is the multi-user, multi-tasking UNIX operating system. UNIX was developed by a group of programmers at Bell Labs for their own use. In contrast to operating systems developed by computer vendors, UNIX had a rather long gestation period in academic and research oriented environments before it became commercial. The long period of development in a protected environment promoted the development of new ideas and the discarding of bad ones. The result was a very flexible environment rich in utilities and tools. For example, the UNIX system comes with text editors, graphics routines, pattern scanners, languages such as C, FORTRAN, and PASCAL, and document preparation utilities, to name a few. The modular design of UNIX and the fact that it is written in the C language, allow it to be customized to meet the user's needs. The statistical language S was developed in a UNIX environment.

At present, there are two versions of UNIX: UNIX 4.2BSD with enhancements and improvementsfrom University of California, Berkeley, and SYSTEM V, the version provided by AT&T's Bell Labs. The Berkeley version is suited to researchers, while the Bell Labs' version is aimed at the commercial market. Work is presently underway to merge these two versions into one version that will be called SYSTEM V. Presently, the Berkeley version is preferred for a statistical research computing environment.

Aside from the operating system other software mainstays of a statistical computing environment are: **S**, APL, IMSL, a graphics package, the linear algebra subroutines in Linpack and Eispack, and a document preparation utility such as troff in UNIX or TEX . This software can be easily installed on a UNIX system.

4 Current Techniques

Some of the most well-known applications of graphics workstations have been provided by the Prim-9, Prim-H, and Orion I projects. These projects and the computationally intensive Interactive Projection Pursuit Regression project (McDonald. 1982), feature graphics with real time motion. The latter project is a sophisticated example of interactive model fitting. These techniques make extensive use of the workstation's fast numerical and graphics processing capabilities. The success of the technique is dependent on interaction with the user. Hence, the feasibility and the success of these techniques are dependent on the computing environment.

The statistical computing environment is important not only for the above applications, but also for commonly used techniques such as regression. Much is known about the pitfalls and limitations of regression. For example, in the analysis of linear models good statistical practice dictates that one examine residuals, search for outliers and influential observations, detect exact and near multicollinearies, and, in general, subject the model to severe criticism. Since these techniques cannot be totally automated, the computing environment must assist in carrying out these tasks with a high degree of interaction. Thus, in order to fully apply known statistical techniques to even conventional procedures, a powerful, highly interactive system is needed.

A multivariate plotting routine at General Motors Research Laboratories (GMR) provides an example of the increase of power and usefulness that interactive graphics capability can add to a static graphics display. The intended residence of such interactive graphics software is a professional workstation. The multivariate plotting routine had its origins in a procedure which plots multivariate data as a matrix of pairwise scatterplots. The use of color, smoothing of data, choice of plotting symbols and their sizes, and other graphical enhancements make the graph very useful for detecting patterns. A similar capability exists with the **pairs** command in **S**. Interactive graphics capability allows the user to immediately modify, highlight, select, etc., as he thinks of them. In a static situation, one must do all thinking prior to reissuing the command to produce the new graph.

5 Hardware

The statistical computing environment described in Section 2 is quite specialized to the needs of the professional statistician and the experienced data analyst and so may not meet the needs of the general computing community. The computing tasks of the general community are usually done on a mainframe computer with a time-sharing operating system. A mainframe must have the capacity and auxiliary devices to meet the needs of its user community, and the time-sharing environment must allocate these resources in an efficient and equitable way. Compromises are inevitable. The environment described in Section 2 will sometimes require that all the available resources be uncompromisingly given to one person. Since by design a time-sharing environment does not let that happen, it is not the appropriate environment for the statistical tasks described here. What one needs is one's own computer. Fortunately, the cost and architecture of today's super microcomputing systems, or graphics workstations, provide just that. The philosophy is not to build bigger and bigger computers to meet the ever expanding specialized needs of different user groups, but to build specialized machines and link them via a network. The network may contain mainframe computers, multi-user computers, personal computers, midicomputers, super computers, printers, tape drives, fileservers, and gateways to other networks. McDonald and Pedersen (1985) give the hardware requirements of a statistical graphics workstation. Joy and Gage (1985) give an overview of the impact of the new hardware on scientific computing.

The hardware requirements needed to support the statistical computing environment being described rule out the personal computer as a possible computing device. The personal computer is characterized by slow processor speed, limited addressable memory, and primitive (proprietary) operating systems. However, these characteristics are changing, so that the capabilities of the personal computer are evolving at different rates towards those of the workstation. The current graphics of the personal computer have too few colors and too coarse of a resolution. Much of the software written for mainframes will not run on personal computers. Workstations do not have these hardware and software limitations. Since the workstation has only one user, the human interface can be designed to increase the productivity of the user and help him concentrate on his problem rather than on computer science.

That one person at a time uses a workstation does not imply that the user is isolated. As noted previously, workstations can be configured in a network. Since a network allows the fusion of different machines, it is important that the network be designed following industry standards so that the greatest number and variety of the available computer hardware are compatible with it. If a vendor proprietary networking system is chosen, then one may be locked into that vendor's hardware. An open architecture philosophy allows one to more readily acquire the latest hardware and, likewise, to more easily sell it. It also encourages competition among the silicon valley upstarts, thereby improving quality and reducing prices.

6 Vendors for Statistical Computing Environment Hardware and Our Final Selection

The criteria for the choice of a microcomputing network for statistical computing may be summed up as follows:

- high resolution, high speed graphics;
- UNIX operating system;
- a rich environment for program development and maintenance;
- large random access memory and disk storage to handle very large data sets, (one data set which we will analyze consists of a million observations with about 15 variables);
- networking capability;
- multi-color graphics (at least enough for shading).

The vendors that we personally contacted were DEC with microVAX II and VAXSTATION 520, APOLLO with 560 and 660, and SUN with SUN-3. Other candidate vendors, such as Chromatics, Iris, Ridge 32, and Symbolics 3600, were not investigated due to local unavailability and the limited time of the search team. The final decision was to choose SUN-3.

The DEC offerings were rejected because of the slower processing speed and the diminished graphics capability relative to SUN. DEC's graphics workstation is really a Tektronix color monitor attached to a monochrome VAXstation. The graphics workstation does not support a diskless node and it does not support ULTRIX-32m, DEC's version of UNIX 4.2BSD. Remote login from a microVAX with ULTRIX to a machine with a VMS operating system (DEC's proprietary operating system) was not possible at the time of our investigation. Networking would be accomplished with DEC's proprietary networking system DECNET.

APOLLO was a closer contender to SUN. At the time of our visit to APOLLO, the company did not have the Motorola 68020 chip (MC68020) in its high performance color graphics workstation. APOLLO was rejected because their UNIX operating sytem was not completely independent of their proprietary operating system AEGIS. As described in their manual (APOLLO, 1985), one "should be familiar with both UNIX and AEGIS software, as well as DOMAIN networks" in order to use the system. The top of the line color graphics workstation, the DN660, had a 32-bit bit-slice processor and could support a maximum of 8 Mb of Random Access Memory (RAM). At the time of our visit, this machine, though still being sold, was in the process of being phased out in favor of a machine built on the 68020 chip. The DN460, the high performance monochrome workstation, could support only 3 Mb of main memory. Warranty analysis at GMR requires at least 15 Mb of RAM. A paint attribute surface representation program at GMR can easily consume 16 Mb of RAM. The APOLLO equipment does not match SUN's speed and memory capacity. A further disadvantage of the APOLLO equipment is that it is designed to run on its own proprietary network.

The SUN equipment most fully met our criteria. The SUN workstations are based on the 68020 chip and can support up to 16 Mb of main memory. The SUN operating system is an extension of UNIX 4.2BSD (the extension allowing for such innovations as windowing). The design philosophy of the equipment is to closely follow industry standards (or working standards) as much as possible. Hence, SUN uses the ETHERNET as its network, in contrast to APOLLO who developed their own DOMAIN network. In comparison with APOLLO's products, the SUN workstations are faster, have more main memory, conform more closely to industry standards, and are totally based on UNIX.

Future developments in SUN workstations include a HYPERchannel connection in May 1986. SUN already has a connection to the Cray, the so called "Craystation."

The current graphics on the SUN is called SunCore and SunCGI. SunCGI complies with the ANSI and ISO's draft of the Computer Graphics Interface for fast two-dimensional graphics. SunCore follows the SIGGRAPH de facto standard for two- or three-dimensional graphics. SunGKS is available for more advanced graphical capabilities.

Some pertinent technical facts about the SUN stations are given by Sun (1985). The high resolution screens on the SUN are 1152 (h) by 900 (v) pixels. The refresh is at 66Hz non-interlased. The color pallet has eight planes with 256 simultaneously displayed colors. The SUN requires no special temperature (0°C – 40°C), humidity (5-95%), or altitude (0-3,000m) environments. The Motorola 68020 chip run at 16.67MHz is included in each station along with the 68881 coprocessor which is run at 12.5MHz. Spanier (1985) provides some timing comparisons between DEC machines with either a "UNIX or a UNIX-like operating system" and the SUN-3. The SUN-3 without a floating point processor is 1.6 times as fast as a VAX 780 in doing floating point computations. It is about 1.8 times as fast in integer computations. These figures are just slightly better when compared to a microvax II. The SUN-3 with a floating point processor is 4.6 times as fast as a VAX 780 in doing floating point computations. This figure is just slightly better when compared to a microvax II. Due to the unavailability of the new APOLLO MC68020-based system, a comparison is not available at this time.

Figure 1 contains a listing of the system hardware that will meet our research needs. The system is configured for six users. Figure 2 is a schematic representation of the subnetwork with the system configuration of the hardware described in Figure 1. The above selection of hardware and the final system configuration is intended to be used for statistical computing. Though this system may be appropriate for other groups, it should not be viewed as prototypical. A big advantage of a subnetwork of workstations is the ability to create a specific computing environment customized to the users' needs. Hence, one should carefully assess one's needs when designing and acquiring a computing environment. In particular, the following (somewhat obvious) steps should be taken:

QUANTITY	DEVICE	HARDWARE
1	3/160C	68020,68881,FPA,GB,GP,16MEG,Color,High Res.
2	3/160C	68020,68881,4MEG,Color,High Res.
1	3/160M	68020,68881,4MEG,High Res., File Server
2	3/50	68020,68881,4MEG,Monochrome,High Res.
1	Ethernet	Cable and Terminators
1	Communication Box	Connection to the GMR Ethernet
1	Tape Drive	1600 BPI Tape Drive
2	2 Eagle Disks	Mass Storage 760 MBytes
1	Laser Writer	Monochrome High Res. Hard Copy
1	Benson Printer	High Res. Color Ink Jet

Figure 1: The SUN Hardware.

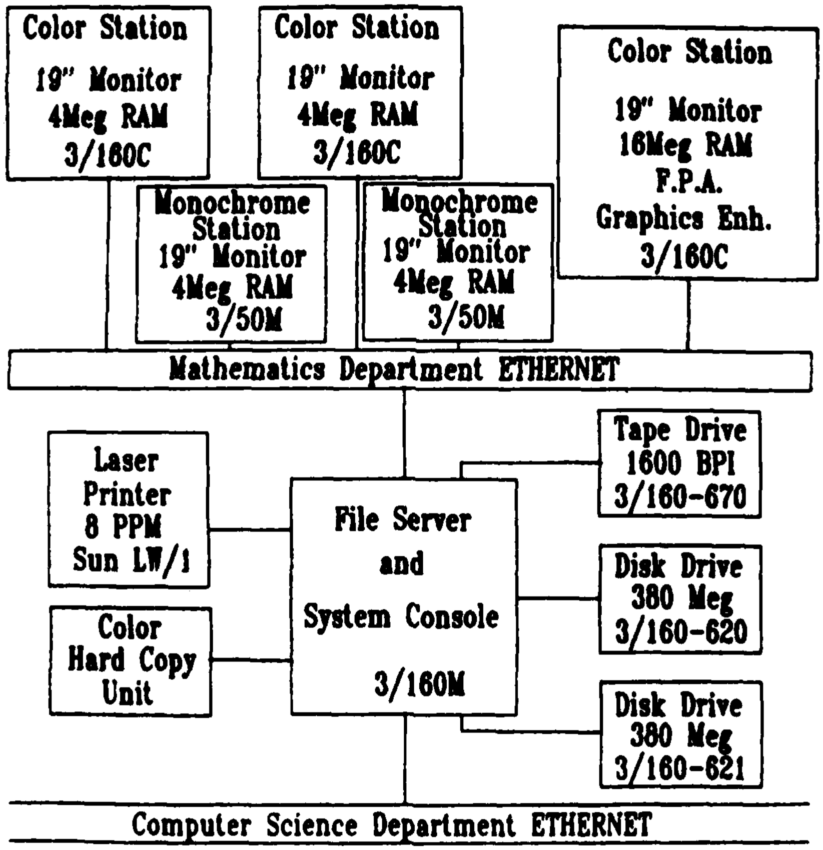


Figure 2: The SUN System Configuration.

1. State the computing requirements (e.g. number of users, type of network, graphics capabilities, output devices, floating point speed, graphics processing speed, level of hardware and software maintenance and support, and operating system characteristics). Scott (1985) gives a checklist of variables to consider in general, but especially when the computing environment is to be used for code development and production work in scientific computation.
2. Identify candidate vendors for the equipment and software.
3. Compare vendor capabilities and screen out vendors not meeting the requirements.
4. Identify possible system configurations among the contending vendors.
5. Select the system which will best satisfy the requirements. Scott (1985) suggests that the problem may be formulated as a linear program.

Taking these steps is important in order to make full use of the new technology provided by a distributed computing network, where each node (here, a subnetwork of workstations) on the backbone network should offer an environment customized to enhance the efficiency, productivity, and capabilities of the particular user group. Moreover, the selection of a workstation should help one to go in new research directions and not merely to mimic or expedite current capabilities.

7 Acknowledgements

We gratefully acknowledge the technical assistance provided by the Computer Science Department of the General Motors Research Laboratories. In particular, Jim Marselle was always available to help answer our many questions and Myron Ginsberg helped interrogate the vendors. Thanks to Jim Ayers and to Diana Wegner for their advice and support.

8 References

1. APOLLO (1985), "DOMAIN/IX Command Reference for 4.2BSD," Apollo Computer Inc., Chelmsford, MA.
2. Becker, R. A., and Chambers, J. M. (1985), "Data Analysis Through Interaction," *UNIX Review*, Vol. 3, No. 11, 46 ff.
3. Chambers, J. M., Cleveland, W. S., Kleiner, B., and Tukey, P. (1983), **Graphical Methods for Data Analysis**, Wadsworth International Group and Duxbury Press.
4. Joy, W., and Gage, J. (1985), "Workstations in Science," *Science*, Vol. 228, 467-470.
5. McDonald, J. A. (1982), "Interactive Graphics for Data Analysis," Stanford Linear Accelerator Center, Stanford University, Report 253.
6. McDonald, J. A., and Pedersen J. (1985), "Computing Environments for Data Analysis I. Introduction," *SIAM J. Sci. Stat. Comput.*, Vol. 6, No. 4, 1004-1012.
7. McDonald, J. A., and Pedersen J. (1985), "Computing Environments for Data Analysis II. Hardware," *SIAM J. Sci. Stat. Comput.*, Vol. 6, No. 4, 1013-1021.
8. Scott, L. R. (1985), "What Kind of Workstation Should I Buy? Seven-fold Criteria for Hardware and Software," Mathematics Research Center, University of Wisconsin - Madison, Technical Summary Report #2840.
9. Spanier, S. (1985), "Sun-3 Benchmarks," Sun Microsystems, Inc., Mountain View, California.
10. SUN (1985), "The Sun-3 Product Family: A Technical Overview; Open Systems for Open Minds," Sun Microsystems, Inc., Mountain View, California.

AN EXPERT SYSTEM FOR TESTING STATISTICAL HYPOTHESES

Knut M. Wittkowski, University of Tuebingen, Fed. Rep. Germany

ABSTRACT

Until now, most approaches for building expert systems with applications in statistics have concentrated on the area of generating hypotheses (RX, GUHA-80, REX, STUDENT, GLIM-Front-End). These expert systems make decisions on the basis of the empirical distribution of the data, subjective opinion, and some a-priori knowledge. The present paper proves some concepts underlying these systems to be inappropriate for testing statistical hypotheses.

Based on a new rating and a new classification of knowledge on statistical concepts, problems and methods and on rules for checking the appropriateness of sub-problems, selection of statistical methods is formalized as a special pattern recognition process. It is demonstrated, how an expert system can support the user in choosing methods and interpreting results.

KEY WORDS: artificial intelligence, experimental design, generalized linear model, nonparametric statistics, multiple comparisons, confirmatory data analysis

1. INTRODUCTION

Statistical methods are frequently used

- to identify criteria that allow for discrimination between groups or to predict the outcome of some event,
- to generate hypotheses that provide an explanation of some biological, sociological or economical process.
- to test some of these hypotheses on an observed set of data.

Each methods is (implicitly) based on a mathematical model, so that on the one hand a decision on a model can be made observing the result of different methods and on the other hand a method can be selected according to a predefined model.

If a model is build for the purpose of prediction or discrimination, there is no need that its parameters represent concepts that have an interpretation in reality. The major goal is to provide a "black box" that gives a good prediction or few mis-classifications, respectively. In medicine, for instance, a model containing some unrealistic parameters might lead to correct diagnoses and a treatment might be useful, even if the underlying mechanism is not known (until recently, no one knew, how aspirin stills pain).

In generating hypotheses it is not sufficient to find a model that allows prediction of an outcome, but it is also necessary that this model can be interpreted, e.g. that it shows which parameters influence this outcome.

In testing hypotheses, the requirements are even more restrictive, because the purpose is not only to look for a model that provides an explanation, but to compute the probability of erroneously choosing "significant" parameters for this model.

This paper discusses implications of these special demands in the field of testing hypotheses on expert systems. Section 2 discusses some concepts used in expert system approaches for generating hypotheses. It is demonstrated, why most of these concepts are not applicable for testing hypotheses. A new concept is introduced, that is based neither on data nor on assumptions or knowledge but on the interest in the analysis. In Section 3 a classification of relevant criteria is introduced and the area of applicability is defined. Section 4 gives a solution for the problem of multiple analyses on subsets of the same set of data and a representation of the process of selecting statistical methods as a special pattern recognition process. An example is given in Section 5 and in Section 6 some consequences of these new concepts are outlined.

2. CRITERIA FOR MODEL SELECTION

2.1. Generalized linear models

The process of building a model starts with formalizing the information available prior to observing the data. This a-priori knowledge and the data remain unchanged in the following process of fitting different models to the data. This process will be discussed in the context of generalized linear models:

$$L(Y_{ijk}) = f_0 + f_1(a_i)+f_2(b_j)+\ldots + f_n(ab_{ij})+\ldots + e_{ijk}$$

where observations Y_{ijk} are decomposed into main effects a_i, b_j, ... , interactions ab_{ij}, ac_{ik}, bc_{jk}, ... , and an error term e_{ijk} representing residuals that cannot be explained by main effects or interactions.

After all possible terms for the model equation have been identified, a set of terms (main effects and interactions), a set of functions f_t , and a "link"-function L are selected to find a model that fits "best" (e.g. in terms of least squares). For prediction and discrimination criteria for selection are typically based on the distribution of the data. For hypotheses generation some a-priori knowledge or assumptions on the area of application may also be used to mark all terms that are known to be necessary or undesired for interpretation. Examples of expert systems for these purposes are RX (BLUM 1978), REX / STUDENT (GALE and PREGIBON 1984), GUHA 80 (HAJEK and IVANEK 1982), and the front-end for GLIM (NELDER and WESTENHOLM 1986). In the following, these systems' sources for information (data, assumptions, knowledge) will be discussed with respect to applicability in the field of testing hypotheses and a new concept will be introduced. To simplify terminology, prediction and discrimination will be treated as special cases of hypothesis generation.

2.2 Rules based on data

Consider the case where the influence of a treatment is to be tested in a (generalized) linear model with several factors and the experimenter is not sure which terms are to be included into the model equation. Suppose the expert system decides on the basis of rules like

```
IF      inclusion of term $f_t(x_{ijk})$
        leads to a higher F-ratio for
        the c-th factor,
THEN    include this term into the
        model equation.
```

Although rules might often look different, they may have a similar effect on the decision. The following rule, for instance, is taken from REX (GALE and PREGIBON 1984):

```
RULE-1
IF      the distribution of  y  is
        unduly skew
AND     the sign of  y  is positive
THEN    assert that logarithms of
        the response variable  y
        should be used.
```

In generating hypotheses these rules might be very useful, because in that context minimizing variability of the residuals is most important and RULE-1 might be a valuable suggestion. Whether or not the variability was actually reduced can be checked on the data. In testing hypotheses, however, the following argument proves rules based on the empirical distribution of the data to be inapplicable: The more rules are available and the more functions $L(y_{ijk})$ and $f_t(x_{ijk})$ are considered, the greater is the probability of finding a model that leads to a statistic with a p-value less than a given alpha.

If the p-value is of interest, i.e. if the decision was intended to be made with a limited probability of an error of first kind, the model that was determined to be the "best" fit must under no circumstances be used for testing the hypothesis of no treatment effects !

If an expert system is to be used for testing hypotheses, it must not derive its information from the data. (There are only few exceptions, like adaptive rank tests, where looking at the data does not affect the conservatism of the test procedure. These exceptions will not be considered here.) This argument rules out application of the above mentioned expert systems for the purpose of testing statistical hypotheses.

2.3. Rules based on assumptions

Many "rules" in common textbooks are based on assumptions on the distribution of the random variables:

```
RULE-2
IF      the distribution of  Y
        is log-normal,
THEN    logarithms of the response
        variable  y  should be
        used.
```

There is no doubt that RULE-2 is true: If two log-normally distributed groups are to be compared with respect to the location of the random variable, taking logarithms provides for a higher efficiency, i.e. a smaller number of observations is necessary to achieve a significant result for groups differing in location.

However, how should one know that the assumption is true ? Assumptions on distributions are not very realistic for many applications: Log-normal distribution of residuals, for instance, can only be guaranteed, if all unknown sources of variation are both multiplicative and independent ! The (asymptotic) relative efficiency (ARE) of two methods, however, depends heavily on the distribution of the residuals in the model chosen: The t-test, for instance, is more efficient than the U-test only for some distributions (e.g. Gaussian); for other distributions (e.g. logistic) the converse might be true.

In some applications the data is used to "prove" that an assumption is met. This approach is based on a common misunderstanding of the concept of "significance": A non-significant test for deviation from normality does not mean that the distribution is proven to be normal. Even if alpha is chosen to be as high as .20, .50, or .80, it is the unknown error of second kind (beta) that counts. The effect of an error of second kind in "proving" the appropriateness of some assumption on the error of first kind in testing a hypothesis is not predictable.

Again some information currently used in hypothesis generation has to be reevaluated for hypothesis testing: Assumptions on the distribution of residuals are helpful only in rare occasions, where all sources of variation are known and their independence is actually proven.

2.4. Rules based on knowledge

Where information from the data itself cannot be used and assumptions are not realistic, knowledge that is independent on the observed data has to be considered as a basis for selection of statistical methods. Because so far most expert systems have been developed for hypothesis generation, knowledge has been given little attention. If a-priori information is considered at all, knowledge bases typically contain rules as in the following example from the STATPATH knowledge base (PORTER and LAI, 1983):

RULE - 3
IF the scale is ordinal
THEN use a rank test

It is well known, that a rank test is more efficient for detecting differences in location than a chi-square test. However, if the purpose of the analysis is to detect any difference in distribution (location, skewness, number of modes, scale, etc.) a rank test might be extremely insensitive.

Consider the following example, where the status after a treatment (placebo or verum) was measured in terms of "better" (+), "unchanged" (0), and "worse" (-):

	-	o	+
placebo	0	20	0
verum	10	0	10

A rank test (U-test corrected for ties) on the difference between placebo and verum leads to a test statistic of $U = 0$, i.e. the test would not be sensitive for the observed type of difference in effects. A rank test is only appropriate, if the "tendency" in effect is of interest (i.e. whether the probability of a preferable result is different for the treatments). On the other hand, the chi-square test is appropriate, if any difference in the effects is of interest.

The scale level, however, must not be ignored, either. Even if the data in the example above had been numerically coded (0:="-", 1:="o", 2:="+"), a t-test would typically give a meaningless result for ordinal or nominal variables. It follows that the scale level restricts the set of possible methods but does not determine the method (except for nominal variables, where means and standard deviations are meaningless, even if common analysis systems like BMDP, P-STAT, SAS, and SPSS typically provide these measures as defaults for "descriptive" analysis.)

2.5. Rules based on interest

As demonstrated in the examples above, for testing hypotheses the rating of efficiency and consistency as criteria for selection of models (or methods) has to be re-evaluated. **Efficiency** was defined as a measure of the number of observations necessary to achieve a significant result under a given alternative. **Consistency** of a method against a certain type of alternative means that any alternative of this type will lead to a significant result for (almost) any underlying distribution, provided the sample size is big enough.

Until now, efficiency was (implicitly) considered the more important criterion for selecting statistical methods. This view resulted from the wealth of results concerning efficiency of test statistics in the field of mathematical statistics, where the set of distributions is typically restricted (e.g. to Gaussian distributions) to ensure that all methods are consistent against the same alternatives.

Assumptions on distributions, however, have been demonstrated to be neither realistic nor provable on the basis of the observations. On the other hand, the assumption of normality often taken as a reason for transforming data is relatively unimportant even to analysis of variance procedures (including the well-known t-test) as far as consistency is concerned. As a consequence, the asymptotic relative efficiency, though important in the field of theoretical statistics, is of little value in the field of applied statistics.

Comparing these concepts it seems more reasonable to base a decision on consistency rather than on efficiency. Rules based on efficiency might result in the "best" solution for the wrong problem, i.e. a completely misleading decision, while decisions based on consistency will give results for the problem of interest, even if this solution is "not optimal", i.e. given with a p-value that might be not exact.

In terms of consistency, there is a second reason, why RULE-1 is not applicable for testing hypotheses: It is a well-known fact that (1) log-normally distributed data are skewed to the left and positive and (2) the geometric mean is the most efficient estimator of the median for this type of data. This implication, however, cannot be inverted: A distribution that is skewed to the left and positive need not be lognormal. Because geometric mean and median estimate different parameters for most other distributions, they are not comparable in terms of efficiency. Taking logarithms and computing means may lead to completely different results than computing medians, because it leads not only to a transformation of the residuals but also to a to a transformation of the problem.

The empirical distribution in all four groups of the following example is skewed to the left and all observations are positive. Means and standard deviations are even positively correlated, as they should be in log-normally distributed data. The example proves that differences in logarithms are sensitive to differences both in location and in scale of the original data (B vs. C). Differences in scale might neutralize or even reverse differences in location (A vs. B and C vs. D, respectively). Moreover, variances are not "stabilized" by computing logarithms of the data ($y = \log(x)$).

Group	A		B		C		D
	3.00		3.60		3.68		5.28
	2.20		2.32		1.93		2.50
	2.00		1.58		1.89		1.00
$x_{.}$	2.40	$<$	2.50	$=$	2.50	$<$	2.93
s_x	.53	$<$	1.02	$=$	1.02	$<$	2.17
	1.10		1.28		1.30		1.66
	.78		.84		.66		.92
	.69		.46		.64		.00
$y_{.}$	.86	$=$	.86	$-$	.87	$>$	.86
s_y	.21		.41	$\neq$	.38		.83

An expert system for testing hypotheses needs rules that depend on the interest (type of influence), i.e. whether a reduction by 50% is as relevant as an increase by 100% (RATIO: compute logarithms) or by 50% (EXPECT: estimate expectation of random variables using the original data):

IF *relative differences are of interest,*

THEN logarithms of the response variable y should be used.

3. FORMAL RELATIONS

The examples above demonstrate that common "heuristics" are for different reasons often inapplicable as rules for expert systems in the area of testing hypotheses: Literature on theoretical aspects of statistical models is typically based on assumptions that are not realistic. Literature on applied statistics often contains misleading or even wrong recommendations.

Evaluation of expert systems in different areas of application shows that "heuristics" are not an economical way to represent knowledge (see WITTKOWSKI 1986 for a more detailed discussion) but that it is desirable to structure knowledge in order that the description of an object or relation can be inherited from the object- or relation-types it belongs to.

Until recently, there was no unique classification of objects and relations, although this lack of meta-knowledge has already been recognized e.g. by MOLENAAR (1984): " If the statistical community succeeds in producing a workable classification of all or most data sets, a statistical expert system could be very helpful in assessing the adequacy and robustness of some statistical techniques for the particular data set considered. "

Because statistical analysis is predetermined by the way an experiment is planned or data are collected in a (retrospective) study, looking for such a classification is much more promising in the field of statistical analysis than, for instance, in the field of medical diagnosis. The following concept for structuring knowledge was introduced by WITTKOWSKI (1984a).

Data are typically arranged as rectangular tables, where rows correspond to observational units (days, patients, rats etc.) and columns to a set of variables associated with each type of observational unit. If all those tables are joined according to the observable relations defined by the observational units, the resulting universal relation describes all **observed relations**.

The structure defined by the a-priori knowledge on the variables will be referred to as **theoretical relations** including classification of dependent, nuisance, and independent variables, strategy of sampling, SI-units, format of data, level and type of scale, etc. . All non-observable, but testable declarations of relevant types of influence on dependent variables (differences in distribution, expectation, tendency, or dispersion) will be referred to as **hypothetical relations**. Requirements on the representation of the results (e.g. tables, plots, test statistics) will be referred to as **output types**. Observed relations will also be called actual relations, while observable, theoretical and hypothetical relations will be referred to as **formal relations**

As proven in Wittkowski (1985), formal relations are sufficient for choosing appropriate statistical methods and interpreting their results, as far as consistency is concerned, provided that a suitable class of statistical methods is selected. Currently, this concept has been proven to be sufficient for linear models (analysis of variance and covariance), tendency models (several non-parametric models based on ranks), "semi"-parametric models (ranking after alignment), log-linear models (analysis of contingency tables), and several graphical and tabular techniques. It can easily be generalized e.g. to principal component analysis, analysis of dispersion etc. .

4. MULTIPLE ANALYSES

In generating hypotheses, the same set of data is typically split into overlapping subsets and analysed with various methods to find a model that fits best. For the reason outlined in Section 2.2 multiple analyses of the same set of data may cause serious difficulties in testing hypotheses. For instance, it is neither possible, to try a t-test on logarithms, a t-test on the original data, a U-test and a chi-square test for the same set of data nor trying out both the paired and unpaired design prior to deciding which result is to be published. It is obvious, that the probability of an error of the first kind will be much higher than the p-value of the result chosen on the basis of this " principle of most significance ". In the terminology of Section 3, confirmatory analyses require that observable and hypothetical relations must not be modified during analysis.

Therefore, an expert system for testing hypotheses should know not only the original (conceptual) theoretical relations but also the original (conceptual) observable and hypothetical relations. This knowledge on formal relations will be referred to as **conceptual problem type**. Based on this knowledge, the system can decide which modifications of the model are allowed in the process of defining a derived **external problem type** (e.g. defining a projection on the data-set for uni- or bi-variate statistics, defining a restriction for simple main effects and a-posteriori multiple comparisons) and which are not (e.g. testing both the original and transformed data, including terms in the conceptual model equation or excluding terms from it). For the special cases given above (c.f. WITTKOWSKI 1985) the system can even compute the adjustment necessary to derive the "global" p-value from the "local" p-value given by the method.

Based on this new concept of rating and structuring knowledge on statistical concepts, problems, and methods, selection of appropriate methods can be treated as a special pattern recognition process (see WITTKOWSKI 1986 for details), which consists of

1) **representing** problems and methods based on data-independent relations (conceptual and implicit problem types, respectively),
2) choosing sub-designs by **projecting** and **restricting** the conceptual problem type,
3) **normalizing** the external problem type and **selecting** a method with a corresponding implicit problem type, and
4) **verifying** assumptions of the method on the data.

5. THE USER INTERFACE

During this pattern recognition process knowledge acquisition and knowledge application can be supported at several stages:

- Acquisition of knowledge (from experts in applied and theoretical statistics) on conceptual and implicit problem types can be facilitated by fast dialogue procedures and be verified by testing its consistency (not shown in this paper).
- By deduction from the conceptual problem type the amount of input necessary to define a sub-problem is reduced, inconsistencies in the non-deducible information are explained or, alternatively, pop-up menus containing only consistent alternatives are presented, and the user is given hints for interpretation (see the example below). The set of necessary parameters may be explained by means of intelligent tutoring.
- The expert system automatically chooses and calls an appropriate statistical method with a corresponding implicit problem type.

Consider, for example, an experiment where each of 10 patients is given three doses of a medication (BETADOSE) subsequent to two different techniques of operation (OPERTYPE) applied after cardiac infarction. Note that patients are nested within factór OPERTYPE. Bodyweight (BODYWGHT) was measured at entry in the study, vigour (ERGOMETR) for each dose. Suppose that the goal of the study was to measure the effect on ERGOMETR in terms of expectation and that a linear relation between BODYWGHT and ERGOMETR is assumed. Input of "ERGOMETR" would result in the following mask on the display:

NAM	OPERTYPE	(3)	BETADOSE	BODYWGHT	ERGOMETR
MIN MAX	(1)		(1)	(2)	(2)
I/O	EXPECT		EXPECT	LINEAR	TEST

Without a modification the system calls automatically an analysis of covariance. If OPERTYPE or BETADOSE is restricted to one or two categories (1) , the necessary modifications of the analysis system's (BMDP, P-STAT, SAS, SPSS, etc.) control language are generated and the' user is given the information, how to compute a global p-value from the local p-value given by the analysis system. If BODYWGHT or ERGOMETR are restricted (2) , the influence types are modified (e.g. LINEAR also for BETADOSE), or a variable PATINGRP with categories 1-5 is introduced (3), the appropriate methods are called as well, but the user is given the information that the result has to be interpreted as being exploratory.

6. CONCLUSIONS

The first statistical expert systems were designed for applications in the field of generating hypotheses. In the field of testing hypotheses, new concepts for knowledge representation are proven to be necessary. A formal description of interest in the result of an analysis (hypothetical relations) is proposed and demonstrated to be sufficient for a wide range of methods.

This expert system approach significantly reduces the amount of information to be entered during analysis of subproblems so that fast dialogue procedures can facilitate access to analysis systems. Because the user formulates questions (restrictions of projections on the data set and hypothetical relations) instead of choosing some test procedures, this concept leads to fewer erroneous applications of statistical methods. Knowledge on conceptual and external problems can also be used to explain results ("significances") to the experimenter and thus help him to interpret these results in terms of hypothetical relations. Automatic adjustment of p-levels in multiple analyses of the same data set reduces the frequency of misinterpreting "significant" results.

The distinction of formal and actual relations not only simplifies access to the data and interpretation of results but is also essential to the concept of knowledge engineering (WITTKOWSKI 1986): Actual integrity constraints can be checked either by data base management systems or inside the methods, while formal integrity constraints can be checked by the expert system. Because access to the data is not necessary to compare the definition of an (external) problem with the knowledge bases, data and knowledge may be stored on distant computer systems (c.f. WITTKOWSKI 1985).

Although this concept has been originally developed for applications in the field of testing hypotheses, it has implications also on generating hypotheses: Criteria for selection of methods in the search for a model that fits to the data can be based on the knowledge on the theoretical relations. The description of implicit problem types associated with each method can be used to help interpreting models in the area of application.

The following consequences are related to more insight into statistical methods: Common concepts underlying these methods will become explicitly defined, so that they can be discussed among experts, and some "heuristics" can be replaced by deterministic rules. This new insight to meta-knowledge will allow to concentrate on relatively few concepts instead on various methods so that unnecessary technical details can be omitted in teaching statistics.

REFERENCES

BLUM RL (1982)
Discovery and representation of causal relationships from a large time-oriented clinical database: The RX project. Berlin: Springer

GALE WA, PREGIBON D (1984)
Constructing an expert system for data analysis by working examples. In: HAVRANEK T, SIDAK Z, NOVAK M. (ed.) COMPSTAT 84. Wien: Physica, 227-236.

HAJEK P, IVANEK J (1982),
Artificial intelligence and data analysis. In: CAUSSINUS H, ETTINGER P, TOMASSONE R (ed.), COMPSTAT 1982 - Part I, proceedings in computational statistics. Wien: Physica, 54-60.

MOLENAAR IW (1984),
Discussion on the paper by professor Victor. Statistical Software Newsletter 10, 121-122.

NELDER JA, WESTENHOLM D (1986)
An expert system for GLIM. In: HAUX R (ed.) Statistical expert systems. Stuttgart, F.R.G.: Fischer

OLDFORD RW, PETERS SC (1985),
Implementation and study of statistical strategy. Workshop on Artificial Intelligence and Statistics, 11-13 April 1985, Princeton, NJ.

PORTER KM, LAY PY (1983)
A statistical expert system for analysis determination. Proc ASA Stat Comp Sect: 309-311

WITTKOWSKI KM (1984a),
On the use of structural information for a statistical expert system in medical research. In: VAN EIMEREN W, ENGELBRECHT R, FLAGLE CD (ed.), 3rd int conf on system science in health care. Berlin: Springer, 1140-1143.

WITTKOWSKI KM (1984b),
Künstliche Intelligenz und verteilte Intelligenz: Implikationen für Sicherheit, Interpretation und Schutz von Krankendaten im Rahmen klinischer Studien. In: ABT K, GIERE W, LEIBER B (ed.) Krankendaten, Krankheitsregister, Datenschutz. Berlin: Springer, 145-149.

WITTKOWSKI KM (1985)
Ein Expertensystem zur Datenhaltung und Methodenauswahl für statistische Anwendungen. Stuttgart, F.R.G.: Dissertation.

WITTKOWSKI KM (1986)
Generating and testing statistical hypotheses: Strategies for knowledge engineering. In: HAUX R (ed.) Statistical expert systems. Stuttgart, F.R.G.: Fischer

AUTHOR'S ADDRESS:
Dept. Med. Biometry
Westbahnhofstr. 55
D-7400 Tuebingen
Fed. Rep. GERMANY

AN EVALUATION OF TIME SERIES ANALYSIS PROGRAMS AVAILABLE IN THREE MAJOR STATISTICAL COMPUTER PACKAGES

Terry J. Woodfield, Arizona State University

1. INTRODUCTION

Three popular statistical software packages, BMDP, SAS, and SPSS^x, have implemented programs to perform what is commonly referred to as Box-Jenkins Time Series Modeling. Many other software products exist that perform calculations related to time series modeling, such as RATS and SCA, but it appears that BMDP, SAS, and SPSS^x are packages likely to be encountered in an academic computing environment. In this paper, we consider the time series features of BMDP, SAS, and SPSS^x and compare the packages using simulated data. The results of a small Monte Carlo study comparing three parameter estimation techniques are also presented.

The implementations we will examine involve batch processing on an IBM 3081 mainframe computer at Arizona State University. While interactive processing is preferred for many applications, it typically requires greater overhead and is not usually feasible for large academic computing systems. The new generation of supermicro's and mini's will clearly alter this situation, and in fact has already had an impact on statistical software vendors as is evident from the availability of versions of BMDP, SAS, and SPSS^x for microcomputers. Large data sets and intensive computational overhead make time series analysis more appropriate for larger, faster computers. However, rapid changes are occurring in both hardware and software. Our evaluations must be judged in that context.

There are many criteria that one could use to evaluate packages. This paper will consider how flexible, up-to-date, and accurate the three packages are with respect to time series analysis. Here flexibility may refer to two aspects: how easy output may be manipulated to produce displays and further analyses, and how many methods of analysis are available. All packages appear to be accurate within the limitations of the floating point arithmetic used and the algorithms employed. Accuracy with respect to the estimation algorithm employed will be emphasized in this work.

2. TIME SERIES MODELING

Most packages use Box and Jenkins (1976) as the primary reference for univariate time series modeling. Judge, et. al., (1985) provide a useful summary of the available theory related to univariate and multivariate time series modeling. The so-called Box-Jenkins modeling strategy is incorporated into the design of the programs of BMDP, SAS, and SPSS^x. This strategy is summarized as: identification → estimation → diagnostic checking. After a model passes the diagnostic checking step, one may produce forecasts using the model.

For convenience, we will restrict attention to autoregressive integrated moving average (ARIMA) models that incorporate a transfer function component. The general form of the model is

$$\phi(B)[Y_t - \mu_t - \sum_{i=1}^{k} \beta_i(B)X_{it}] = \theta(B)\varepsilon_t, \qquad (2.1)$$

where B is the backshift operator defined by $B^k Y_t = Y_{t-k}$, Y_t is the original series or a transformation of the original series, μ_t is a mean or trend term, the polynomial $\phi(B)=1-\phi_1 B-\phi_2 B^2-\cdots-\phi_r B^r$ may be the product of stationary seasonal and non-seasonal autoregressive components and nonstationary differencing components, the polynomial $\theta(B)=1-\theta_1 B-\theta_2 B^2-\cdots-\theta_s B^s$ may be the product of invertible seasonal and non-seasonal moving average components, ε_t is an independent Gaussian white noise process, $X_{1t},\ldots, X_{kt}$ are exogenous variables, and the transfer functions $\beta_i(B)=\omega_i(B)/\delta_i(B)$ are ratios of polynomials of possibly varying orders in general given by $\omega(B)=\omega_0-\omega_1 B-\omega_2 B^2-\cdots$, $\delta(B)=1-\delta_1 B-\delta_2 B^2-\cdots$. See Box and Tiao (1975) for a complete description of transfer function models with ARIMA errors.

Given realizations $y_1, y_2,\ldots, y_n$ of a time series Y_t, one may estimate the parameters of the model (2.1) using nonlinear least squares or exact maximum likelihood. Kohn and Ansley (1985) provide one of the most recent algorithms for evaluating the likelihood function. Ansley and Newbold (1980) compare the unconditional least squares, conditional least squares, and maximum likelihood techniques for parameter estimation. The algorithm of Kohn and Ansley appears to be the best available method for parameter estimation based on numerical and statistical criterion. Newton (1981) provides a useful discussion of available estimation techniques.

The identification stage of the modeling process relies on subjective examination of sample functions. Many sources restrict attention to the sample autocorrelations and partial autocorrelations. Frequency domain quantities such as periodograms and sample spectral densities may also be employed. Transfer function models are identified using cross-correlations and cross-spectral densities. Recent diagnostic tools include the shifted S-array of Woodward and Gray (1981), objective order determining criteria such as AIC (Akaike 1974) and CAT (Parzen 1977), and canonical correlation diagnostics suggested by Priestley, Rao, and Tong (1974), Akaike (1976), and Tsay and Tiao (1985).

3. DESCRIPTION OF THE PACKAGES

We will address some of the basic features of each package in this section. The next section will provide comments and comparisons related to univariate analysis of a time series.

BMDP has two programs for time series analysis: BMDP1T and BMDP2T. BMDP1T performs univariate and bivariate spectral analysis. BMDP2T performs Box-Jenkins time series analysis including transfer function models. BMDP provides a basic user's manual and numerous technical reports. The documentation provided by the basic manual is adequate for an experienced time series analyst but probably difficult for the student or novice. BMDP anticipates many of the needs of time series analysts and has implemented techniques that are current up to about 1982.

BMDP has the most primitive data handling capabilities of the three packages. The TRANSFORM paragraph is used to transform response or predictor variables or to create indicator variables. No date/time functions are provided to label the time frame of the data. BMDP provides few features for saving or merging data sets. For example, producing residuals from a regression analysis using BMDP1R and analyzing them using BMDP2T is more difficult than using comparable procedures in SAS or SPSS^x.

BMDP1T is the most comprehensive frequency domain implementation of the three packages. It calculates periodograms, cross-periodograms, and smoothed or filtered versions of these. BMDP1T has options for replacing missing values of a series. Parametric spectral estimation is available using autoregressive filters.

BMDP2T is a complete implementation of Box-Jenkins ARIMA modeling that includes capabilities for handling transfer function models. No multivariate capabilities beyond the inclusion of multiple transfer functions are available.

SAS provides a library of procedures in the SAS/ETS product that perform a wide variety of time domain computations. SAS provides numerous manuals, instructional texts, and technical reports that document its features. The SAS/ETS User's Guide is the basic manual for the SAS/ETS product. The documentation in the SAS/ETS guide is adequate for experienced time series analysts but perhaps somewhat confusing to the novice. The documentation is more comprehensive and easy to follow than the comparable BMDP product. SAS anticipates many of the needs of time series analysts and has implemented techniques that are current up to about 1983.

SAS has the most advanced data management features of the three packages. It has a large number of built in date/time functions for labeling the time frame of the data. Variable transformations are handled by the DATA step. In fact, the wide range of built-in mathematical and statistical functions make it possible to use the DATA step to program simple applications such as multiplicative decomposition seasonal adjustment.

SAS/ETS has ⌐ number of procedures to perform univariate and multivariate forecasting. These include ARIMA, AUTOREG, FORECAST, and STATESPACE. Other linear and nonlinear modeling procedures are also available. SPECTRA calculates periodograms, cross-periodograms, and smoothed versions of these. Kernel estimates and parametric estimates of a spectral density are not provided. No filtering other than moving average filtering is available.

SPSS^x has one procedure, BOX-JENKINS, for performing univariate time series computations. SPSS^x provides numerous manuals and instructional texts that document its features. The SPSS^x User's Guide is the basic manual for users of SPSS^x. The documentation in the SPSS^x guide is adequate for experienced time series analysts but perhaps somewhat confusing to the novice. The documentation is more comprehensive and easy to follow than the comparable BMDP product (univariate modeling only), but inferior to the corresponding SAS product. SPSS^x anticipates many of the needs of time series analysts who use only Box-Jenkins univariate ARIMA models and has implemented techniques that are current up to about 1976.

SPSS^x has advanced data management features comparable to earlier versions of SAS. It has a large number of built in date/time functions for labeling the time frame of the data. Variable transformations are handled by the COMPUTE command. Like SAS, SPSS^x has a wide range of built-in mathematical and statistical functions.

SPSS^x has no multivariate time series capabilities and no frequency domain capabilities. Univariate transfer function modeling is not available from SPSS^x. The BOX-JENKINS procedure implements the basic ARIMA modeling strategy described in Box and Jenkins (1976).

4. UNIVARIATE IMPLEMENTATION OF BOX-JENKINS TIME SERIES MODELING

Consider model (2.1) without the transfer function component. To identify To identify the nature of the polynomials $\phi(B)$ and $\theta(B)$, BMDP2T, SAS PROC ARIMA, and SPSS^x procedure BOX-JENKINS all provide the sample autocorrelation and partial autocorrelation functions. For identifying seasonal components, BMDP and SAS permit calculation of the periodogram and various smoothed versions of the periodogram.

The display of the sample autocorrelations and partial autocorrelations is very similar for all three packages. None of the packages permit alternate forms of plotting the sample autocorrelations, and none allow the autocorrelations to be saved for later use. Consequently, more desirable pen plotter versions cannot be obtained, even though SAS and SPSS^x have advanced routines for accessing sophisticated plotting hardware. Thus, publication quality plots of the sample autocorrelations must be obtained using other resources. One approach is to read the appropriate output page using a program written in a lower level language like C or FORTRAN that reconstructs the desired sample function, and then use SAS to read in the reconstructed function and plot it using SAS/GRAPH.

If the identification stage reveals that the series is nonstationary, various transformations or differencing operations may be performed. All packages have the ability to perform transformations "outside of" the procedures used to carry out the analysis. In addition, SPSS^x procedure BOX-JENKINS permits logarithmic or power transformations as options within the procedure. The advantage of the SPSS^x implementation is that the user need not worry about untransforming the forecasts. All programs allow differencing of the input series to be performed within the procedure being used.

When a model has been identified, the parameters may be estimated using conditional least squares (CLS), unconditional least squares (ULS, or the backforecasting approach described in Box and Jenkins, 1976), or maximum likelihood

(ML). Neither BMDP2T nor SPSS^x BOX-JENKINS provide maximum likelihood estimates. SAS PROC ARIMA uses one of the more efficient algorithms for obtaining ML estimates. CLS and ULS estimates are obtained using a nonlinear least squares algorithm. BMDP2T and SAS PROC ARIMA seem to use a more efficient algorithm (Gauss-Marquardt) for nonlinear least squares than does SPSS^x procedure BOX-JENKINS (pattern search), although numerical results using the same estimation technique are usually in close agreement for the three packages.

SAS PROC ARIMA has a unique feature that allows alternate parameterization for a transfer function model. ARIMA also produces an estimate of $\mu_t=\mu$ labeled MU, and for models with AR components, ARIMA provides an estimate labeled CONSTANT which is the stable mean typically called θ_0 by some authors and defined by $\theta_0=\mu(1-\Sigma\phi_k)$. BMDP and SPSS^x provide only the estimate of θ_0. When no AR components are present, $\mu=\theta_0$.

The three packages have similar options for obtaining forecasts. Options include specifying at what time point the forecasts should begin and how many time points beyond the end of the series the forecasts should extend. BMDP2T does not display confidence intervals for the forecasts, but does provide standard errors that may be used to compute such intervals. SAS PROC ARIMA provides 95% confidence intervals. SPSS^x BOX-JENKINS provides confidence intervals and allows the user to specify the desired confidence coefficient. Note that such confidence intervals are usually valid only when derived using a long series of data, i.e., the intervals are based on asymptotic theory and not on exact distribution theory.

All three packages display forecasts and confidence intervals and/or standard errors in a column listing. SPSS^x BOX-JENKINS also displays forecasts in a table similar to that found in Box and Jenkins (1976, Table 5.2, page 136). In addition, SAS PROC ARIMA allows the forecasts and related statistics to be saved in an output data set.

All plotting of sample functions, forecasts, and related statistics is "internal" and beyond the control of the user in BMDP2T and SPSS^x BOX-JENKINS. Alternately, while SAS PROC ARIMA plots sample autocorrelations and partial autocorrelations (and inverse autocorrelations), the PLOT and GPLOT procedures are employed in SAS to customize desired plots of original data, forecasts, and confidence intervals. Thus, SAS seems more ideally suited for time series analyses that are to be published.

Few modern diagnostic tools are made available by the three packages. SAS PROC ARIMA produces the inverse autocorrelation function in the IDENTIFY step. PROC ARIMA also provides the value of Akaike's AIC criterion for a fitted model. BMDP2T, SAS PROC ARIMA, and SPSS^x procedure BOX-JENKINS provide a chi-square test of the residuals for white noise. All three packages produce t-ratios for estimated parameters and sample autocorrelations for the residuals. Again, note that the t-ratios correspond to asymptotic theory, since no exact distribution theory is known. For this reason, the packages wisely refuse to print p-values whose interpretation would be questionable for smaller series.

All packages also report the residual variance and other statistics related to the original series and the residuals. The terminology used by SAS, e.g., "VARIANCE ESTIMATE" instead of "RESIDUAL VARIANCE", may be confusing to some. All packages require some investigation to determine the divisor employed to obtain the residual variance. The divisor appears to be the degrees of freedom formed by subtracting degrees of differencing and number of parameters estimated from the total series length.

None of the packages have options within the procedures or programs to carry out a complete residual analysis. Reliance on a composite chi-square test for white noise is inadequate for many situations. However, SAS has the greatest flexibility in retaining the residuals and carrying out a residual analysis using other SAS procedures, such as PROC MEANS, PROC UNIVARIATE, and PROC PLOT. With a little more effort, SPSS^x can achieve similar results, while BMDP requires much greater effort to carry out a complete residual analysis.

Finally, note that SAS PROC AUTOREG and SAS PROC FORECAST have some features that may be useful in Box-Jenkins modeling. Primarily, these features are designed for purely autoregressive processes.

Display 1 summarizes the features of each package.

5. EXAMPLES OF UNIVARIATE TIME SERIES MODELING

Three models and two series lengths (n=50 and n=100) were used to simulate data in order to compare the estimation algorithms of the three packages. The three models employed were:

(1) $y_t - 0.95y_{t-1} = \varepsilon_t$;

(2) $y_t = \varepsilon_t - 0.95\varepsilon_{t-1}$;

(3) $y_t - 1.5y_{t-1} + 1.21y_{t-2} - 0.455y_{t-3}$
$= \varepsilon_t + 0.2\varepsilon_{t-1} + 0.9\varepsilon_{t-2}$.

Model (3) was suggested by Woodward and Gray (1981).

Initially, one series was generated for each model and sample size. Results were different across all packages, although in many cases results were very similar. The differences can be attributed to the following differences in implementation:

1. Default values for convergence of parameter estimates differ. SAS and SPSS^x use a stopping criterion of 0.001, while BMDP uses 0.0001.
2. The stopping rules vary across packages, e.g., BMDP has a stopping rule related to the relative change in the residual sum of squares, while SAS and SPSS^x appear to use only the relative difference in consecutive parameter estimates.
3. The three packages use different numerical optimization routines.
4. Only SAS restricts estimates by default to fall within the stationary or invertible region.

5. SAS appears to use double precision for all computations, while SPSS* uses single precision for some computations. BMDP does all computations in single precision.

While one can use certain options to control some of the above factors, it appears that the packages cannot be made to provide identical results. SAS PROC ARIMA can eliminate the stationary and invertible restriction. SPSS* and BMDP do not have an option to force stationarity or invertibility. All packages allow one to fine tune convergence criteria, and all packages allow control of starting values for estimation.

We decided that simulations generated to compare the packages might unfairly favor the capabilities of one or more packages. For example, the first two models considered have roots near the unit circle. The ULS method appears to perform better for models with roots near the unit circle, while otherwise it appears to be inferior to the CLS and ML methods (Ansley and Newbold 1980). Thus, the package with the "best" implementation of ULS might appear superior to the others even if it had an inferior implementation of the other techniques.

Display 2 gives some results using n=50 for all three models. Note that in some cases results are quite different and that choice of package and estimation technique may be very important. Unfortunately, no guidelines are evident other than those suggested by Ansley and Newbold (1980), namely to use maximum likelihood if possible over ULS and CLS. Since neither BMDP nor SPSS* provide ML estimates, this advice favors SAS.

A comprehensive study is warranted to determine what set of estimation algorithms and default tuning parameters may be preferred for given types of models. Experience suggests that no given technique or set of defaults will be a clear winner.

In order to gain some insight into the validity of the Ansley and Newbold (1980) findings, we ran a small simulation study using five replicates for each model/sample size combination. We used SAS PROC ARIMA with its default settings to compare the three estimation techniques. The design was a 3 by 2 factorial repeated measures experiment with response variable MSE defined by

$$MSE = \Sigma\ [parameter–estimated]^2/(no.\ parameters)$$

We define N as the sample size classification variable and MODEL as the classification variable identifying the model used to simulate the data. The repeated measure was MSE taken over the three techniques, CLS, ULS, and ML, for each data set (experimental unit). The analysis clearly revealed the presence of interaction between MODEL and N. Univariate corrected F-tests indicated that the MODEL means were significantly different for all three repeated measures. In addition, N and MODEL*N means were significantly different for the CLS and ML methods. Statistical significance was judged at the 10% level.

Given the presence of interaction, we proceed to examine the value of MSE for each technique averaged over MODEL, N, and MODEL*N to gain insight into the behavior of the techniques. Display 3 provides summary tables giving the relevant averages. Also given are averages using MAD rather than MSE, where MAD is defined by

$$MAD = \Sigma\ |parameter–estimate|/(no.\ parameters).$$

Note the cases where MSE and MAD provide different orderings of the average values.

Our results seem to agree with those obtained by Ansley and Newbold (1980). In particular, the ML method seemed to work best for small samples and for more complicated models, while ULS seemed to perform well for the two models with roots of the characteristic polynomials near the unit circle. Ansley and Newbold (1980) observe that ULS estimates tend to be poor in the sense that they often give estimates yielding characteristic polynomials with roots near the unit circle even when the underlying model does not exhibit roots near the unit circle.

6. CONCLUDING REMARKS

If one had to choose a single package for time series analysis, SAS would probably be the choice because it appears to provide the most options and flexibility. On the other hand, with access to all three packages, there are situations where BMDP or SPSS* might be used instead of or along with SAS. I have seen many clever things done with the SAS DATA step or with SAS PROC MATRIX, including seasonal adjustment, state space modeling, and kernel spectral estimation. (In some cases, the use of a lower level language would have been preferred, but the motivation was more to show that "SAS could do it" rather than "this is the way it should be done".)

Historically, BMDP seems to have been the first of the three packages to provide a fully implemented version of Box-Jenkins transfer function modeling. SAS followed with an upgraded version of PROC ARIMA a few years later that matched the capabilities of BMDP2T. SPSS* currently has no transfer function capabilities. SAS has yet to match BMDP's frequency domain capabilities, and SPSS* has no frequency domain capabilities. On the other hand, the SAS/ETS product provides a comprehensive collection of tools useful to time series analysts, although, as the name implies, economic applications dominate.

Finally, recalling the brief discussion above on the changing technology, the future looks to more sophisticated interactive time series analysis programs. One expects to see significant changes in the three products mentioned over the next few years to keep pace with the statistical theory and the computer technology.

The information contained in this paper was obtained from the manuals listed in the references and from the author's experience with the packages. None of the software vendors were contacted to confirm this information. Hence, care should be exercised in using this information to help select a package for performing time series analysis.

REFERENCES

Akaike, H. (1974). "Markovian Representation of Stochastic Processes and Its Application to the Analysis of Autoregressive Moving Average Processes." Annals of the Inst. of Stat. Math., 26, 363-387.

Akaike, H. (1976). "Canonical Correlations Analysis of Time Series and the Use of an Information Criterion," in Mehra, R. and Lainiotis, D.G. (eds.), Advances and Case Studies in System Identification. New York: Academic Press.

Anderson, T.W. (1977). "Estimation for Autoregressive Moving Average Models in the Time and Frequency Domains". Annals of Stat., 5, 842-865.

Ansley, C., and Newbold, P. (1980). "Finite sample properties of estimators for autoregressive moving average models". J. of Econometrics, 13, 159-183.

BMDP Statistical Software Manual, 1985 Reprinting (1983). Los Angeles: University of California Press.

Harvey, A.C. (1982). "The Kalman Filter and Its Application in Econometrics and Time Series Analysis." Methods of Operations Research, 44, 3-18.

Jazwinski, A.H. (1970). Stochastic Process and Filtering Theory. New York: Academic Press.

Judge, G., Griffiths, W., Hill, R., Lutkepohl, H., and Lee, T-C. (1985). The Theory and Practice of Econometrics. New York: Wiley.

McDowall, D., McCleary, R., Meidinger, E., and Hay, R. (1980). Interrupted Time Series Analysis. Sage Publications.

Parzen, E. (1977). "Multiple time series: Determining the order of approximating autoregressive schemes," in Multivariate Analysis IV, edited by P. Krishnaiah. Amsterdam: North Holland, 283-295.

Priestley, M.B., Subba Rao, T. and Tong, H. (1974). "Applications of principal components analysis and factor analysis in the identification of multivariable systems." IEEE Trans. Auto. Ctrl., AC-19, 730-734.

SAS Institute Inc. SAS/ETS User's Guide, Version 5 Edition (1984). Cary, NC: SAS Institute Inc.

SPSS* Statistical Algorithms (1983). Chicago: SPSS Inc.

SPSS* User's Guide, 2nd Edition (1986). New York: McGraw-Hill.

Tsay, R.S., and Tiao, G.C. (1985). "Use of canonical analysis in time series model identification." Biometrika, 72, 299-316.

Woodward, W.A., and Gray, H.L. (1981). "On the relationship between the S array and the Box-Jenkins method of ARMA model identification". JASA, 76, 579-587.

Display 1. Box-Jenkins Time Series Features of the Three Packages

STEP	BMDP	SAS	SPSS*
	2T	ARIMA	BOX-JENKINS
Identification			
Autocorrelations	X	X	X
Partial Autoc.	X	X	X
Inverse Autoc.		X	
Transformations	X	X	X
Estimation			
ULS	X	X	X
CLS	X	X	X
ML		X	
t-ratios	X	X	X
Correlations of estimates	X	X	X
Diagnostic Checking			
Residual Autoc.	X	X	X
White noise test	X	X	X
Plots	X	PLOT,GPLOT	X
AIC		X	
Spectra	1T	SPECTRA	
Forecasting			
k-step ahead forecasts	X	X	X
Confidence Intervals		X (95%)	X
Plots	X	PLOT,GPLOT	X
"Undo" transformations	2-step	OUTPUT+DATA STEP	X
Transfer Functions	X	X	

NOTE: An X indicates that the feature is present within the procedure or program. Any other non-blank entry indicates that the feature is available in another procedure or requires some additional programming considerations.

Display 2. Results for Simulated Models, N=50

Model	Method	Estimate	BMDP	SAS	SPSS*
1	ULS	AR1	0.9422/19.19	0.9347/18.16	0.9338/18.51
		Res. Var.	1.1102	1.1051	1.0595
	CLS	AR1	0.9424/19.18	0.9370/18.37	0.9328/18.27
		Res. Var.	1.1103	1.1072	1.0808
2	ULS	MA1	0.9940/82.74	0.9999/3.45	1.0234/16.57
		Res. Var.	1.2676	1.2605	1.1558
	CLS	MA1	1.0070/61.57	0.9999/14.39	1.1156/11.87
		Res. Var.	1.2642	1.3136	1.1215
3	ULS	MA1	-0.3691/-3.77	-0.3666/-4.30	-0.2988/-23.63
		MA2	-0.9059/-11.17	-0.9999/-3.30	-1.0399/-33.91
		AR1	1.4760/13.36	1.5260/11.15	1.4986/14.19
		AR2	-1.2260/-8.91	-1.3716/-7.47	-1.3828/-11.48
		AR3	0.4895/5.48	0.6128/4.72	0.6708/7.64
		Res. Var.	1.1770	1.3336	1.0197
	CLS	MA1	-0.3732/-3.32	-0.2901/-1.69	-0.5219/-2.14
		MA2	-0.8533/-8.67	-0.6183/-4.27	-1.2165/-5.39
		AR1	1.3330/9.21	1.5877/8.74	1.1511/10.51
		AR2	-1.0910/-5.43	-1.3380/-5.07	-1.0625/-7.09
		AR3	0.4704/3.33	0.5715/3.45	0.4854/4.29
		Res. Var.	1.4960	1.8471	1.1651

Entries are: estimate/t-ratio

Display 3a. Overall Averages

Response(method)	n	MEAN	ST. DEV.
MSE(CLS)	30	0.03418720	0.10957495
MSE(ML)	30	0.02956176	0.10990708
MSE(ULS)	30	0.02993844	0.11025630
MAD(CLS)	30	0.10693617	0.11400497
MAD(ML)	30	0.09227277	0.10990816
MAD(ULS)	30	0.09164743	0.11121834

Display 3b. Cell Averages

Response(method)	n	MEAN	ST. DEV.
MODEL=1 N=50			
MSE(CLS)	5	0.00317783	0.00547312
MSE(ML)	5	0.00395779	0.00799318
MSE(ULS)	5	0.00321348	0.00571738
MAD(CLS)	5	0.04081240	0.04347671
MAD(ML)	5	0.04078180	0.05355648
MAD(ULS)	5	0.03903800	0.04595537
MODEL=1 N=100			
MSE(CLS)	5	0.01460279	0.02593161
MSE(ML)	5	0.01599610	0.02668808
MSE(ULS)	5	0.01445261	0.02557444
MAD(CLS)	5	0.08491980	0.09612111
MAD(ML)	5	0.09650740	0.09139488
MAD(ULS)	5	0.08667780	0.09313678
MODEL=2 N=50			
MSE(CLS)	5	0.00529622	0.00625272
MSE(ML)	5	0.00248983	0.00000631
MSE(ULS)	5	0.00249826	0.00000111
MAD(CLS)	5	0.06567540	0.03505294
MAD(ML)	5	0.04989820	0.00006322
MAD(ULS)	5	0.04998260	0.00001110
MODEL=2 N=100			
MSE(CLS)	5	0.00441814	0.00422492
MSE(ML)	5	0.00340524	0.00203887
MSE(ULS)	5	0.00301214	0.00114943
MAD(CLS)	5	0.06094560	0.02965998
MAD(ML)	5	0.05674320	0.01522531
MAD(ULS)	5	0.05422320	0.00948585
MODEL=3 N=50			
MSE(CLS)	5	0.14264180	0.25853880
MSE(ML)	5	0.14355716	0.25902080
MSE(ULS)	5	0.14719005	0.25802466
MAD(CLS)	5	0.23572824	0.20096962
MAD(ML)	5	0.23331799	0.20664777
MAD(ULS)	5	0.24280711	0.20164603
MODEL=3 N=100			
MSE(CLS)	5	0.03498640	0.03017236
MSE(ML)	5	0.00796441	0.00756469
MSE(ULS)	5	0.00926409	0.01050028
MAD(CLS)	5	0.15353561	0.07943903
MAD(ML)	5	0.07638804	0.03287154
MAD(ULS)	5	0.07715584	0.04366065

Display 3c. Model Averages

Response(method)	n	MEAN	ST. DEV.
MODEL=1			
MSE(CLS)	10	0.00889031	0.01866648
MSE(ML)	10	0.00997695	0.01962674
MSE(ULS)	10	0.00883305	0.01844740
MAD(CLS)	10	0.06286610	0.07407325
MAD(ML)	10	0.06864460	0.07648430
MAD(ULS)	10	0.06285790	0.07365030
MODEL=2			
MSE(CLS)	10	0.00485718	0.00505210
MSE(ML)	10	0.00294754	0.00144234
MSE(ULS)	10	0.00275520	0.00081274
MAD(CLS)	10	0.06331050	0.03071304
MAD(ML)	10	0.05332070	0.01077235
MAD(ULS)	10	0.05210290	0.00670723
MODEL=3			
MSE(CLS)	10	0.08881410	0.18256961
MSE(ML)	10	0.07576079	0.18695201
MSE(ULS)	10	0.07822707	0.18687692
MAD(CLS)	10	0.19463192	0.15043879
MAD(ML)	10	0.15485301	0.16217373
MAD(ULS)	10	0.15998148	0.16291455

Display 3d. Sample Size Averages

Response(method)	n	MEAN	ST. DEV.
N=50			
MSE(CLS)	15	0.05037195	0.15388066
MSE(ML)	15	0.05000160	0.15452066
MSE(ULS)	15	0.05096727	0.15489169
MAD(CLS)	15	0.11407201	0.14307293
MAD(ML)	15	0.10799933	0.14645329
MAD(ULS)	15	0.11060924	0.14698484
N=100			
MSE(CLS)	15	0.01800244	0.02510749
MSE(ML)	15	0.00912192	0.01581345
MSE(ULS)	15	0.00890961	0.01556248
MAD(CLS)	15	0.09980034	0.07964706
MAD(ML)	15	0.07654621	0.05517159
MAD(ULS)	15	0.07268561	0.05698740

Algorithms for Nonlinear Generalized Cross–Validation

Brian S. Yandell, University of Wisconsin–Madison

A variety of penalized nonlinear problems can be expressed as the iterated solution to a nonlinear minimization, in which the inner step involves minimizing a penalized weighted least squares expression. We propose algorithms when matrices in the least squares problem may depend on the unknown parameters. The problems in increasing complexity are (a) generalized linear models, (b) iterated reweighted least squares, and (c) general nonlinear problems. The algorithms are built around GCVPACK (Bates, Lindstrom, Wahba and Yandell, 1985), a package for generalized cross-validation, using a balance of Cholesky and singular value decompositions which is adjusted depending on the type of problem.

1. Introduction

A variety of penalized nonlinear problems can be expressed as the iteration to a solution of a nonlinear minimization, in which the inner step involves minimizing a quadratic form such as

$$\frac{1}{n}\ \| \mathbf{W}^{\mathrm{T}}(\mathbf{y}-\mathbf{S}\alpha-\mathbf{T}\beta-\mathbf{K}\delta) \|^2 + \lambda\delta^{\mathrm{T}}\mathbf{K}_U\delta \qquad (1.1)$$

in which S, T and K are the design matrices for the covariates, polynomial and "smooth" part of the model, and y and W are the responses and the weights. The simplest form is the partial spline model, or semi-parametric linear model,

$$y_i = S_i^{\mathrm{T}}\alpha + f(\mathbf{x}_i) + \varepsilon_i \ , i=1,\cdots,n \ , \qquad (1.2)$$

in which $f(\cdot)$ is some "smooth" function and $\varepsilon=(\varepsilon_1,\cdots,\varepsilon_n)^{\mathrm{T}}$ has covariance matrix $(\mathbf{WW}^{\mathrm{T}})^{-1}$ which is usually diagonal. We present three situations and proposed computational solutions when matrices in the above linearized problem may depend on the unknown parameters. The problems in increasing degree of complexity are:

(1) Semi-parametric generalized linear models, in which S, T, K and $\mathbf{K}_U$ are constant, while W and y may change with each iteration.

(2) Iteratively reweighted least squares, in which only $\mathbf{K}_U$ remains constant.

(3) General nonlinear problems (remote sensing, for example), in which all matrices may change with each iteration.

Different compromises are suggested by each problem. Clearly, one would like to decompose the constant matrices exactly once and would like to keep decompositions of the changing matrices as cheap as possible. The method proposed here combines the advantages of SVD in locating the generalized cross validation choice of λ with Cholesky decompositions which are relatively cheap once λ is fixed. While the decompositions suggested are not new, the combination of approaches appears to be an unexplored area. The basic strategy is as follows:

(1) guess at initial λ (=∞) and $(\beta^{\mathrm{T}},\alpha^{\mathrm{T}},\delta^{\mathrm{T}})$
(2) CD: iterate (part-way) to solution for fixed λ
(3) linearize the problem as in (1.1)
(4) SVD: pick optimal λ via GCV
(5) iterate (2)-(4) to convergence

Convergence criteria can include absolute or relative convergence of the regularization functional and/or the parameter estimates, and absolute convergence of $\log(n\lambda)$. The number of iterations in (2) may be restricted, leading to rough estimates which are fed into (3).

We do not assume any special structure to the design or the matrices, except that we suppose that W is of full rank, and computationally invertible. In many cases, W is actually diagonal, but this will not be explicitly used in the linear algebra.

Algorithms for the linear model (1.2) have been given by many authors, most recently in the multivariate form by Bates et al. (1985). The algorithms below are extensions of Bates et al. (1985), building on their Fortran77 package, GCVPACK.

2. Semi-Parametric Generalized Linear Models

For semi-parametric generalized linear models (SGLM), one has a parameter vector θ which consists of a parametric piece and a "smooth" nonparametric piece,

$$\theta_i = \mathbf{S}_i^{\mathrm{T}}\alpha + f(\mathbf{x}_i) \ , \ i=1,\cdots,n \ .$$

One can formulate the problem as minimizing, for fixed λ,

$$S_\lambda(\theta) = L(\theta) + \lambda J(\theta) \ .$$

in which L is the log likelihood and J is the smoothing penalty (see Good and Gaskins (1971); Leonard (1982); Green, Jennison and Seheult (1983); O'Sullivan, Yandell and Raynor (1986); Green and Yandell (1985)). We know from O'Sullivan (1983) that if $L(\theta)$ is suitably convex and $J(\theta)$ is a quadratic form (*e.g.*, the squared norm of a projection), then $S_\lambda(\theta)$ has a unique minimum for each λ. These conditions appear to hold for many generalized linear models.

One can choose λ to minimize the GCV criterion (Craven and Wahba, 1979). which is "close" *to minimizing the predictive* mean square error (see Craven and Wahba (1979); Speckman (1985); Cox (1983)). What we propose to do here is to iterate on θ *and* λ, to find the $\hat\lambda$ which is the GCV minimizer and the $\hat\theta$ which minimizes $S_{\hat\lambda}(\hat\theta)$. It is not known whether such a procedure will converge, but we conjecture that, if the GCV minimizer is bounded away from 0 and ∞ and L is suitable convex, then it does converge.

The log likelihood can be written in an iterative form using pseudo-values y and pseudo-weights W,

$$\begin{aligned} \mathbf{WW}^{\mathrm{T}} &= E\left[-\frac{\partial^2 L}{\partial\theta\partial\theta^{\mathrm{T}}}\right]_{\theta^o} \\ \mathbf{y} &= \theta^o + (\mathbf{WW}^{\mathrm{T}})^{-1}\left[\frac{\partial L}{\partial\theta}\right]_{\theta^o} \ , \end{aligned} \qquad (2.1)$$

based on θ^o from the previous iteration. Note that for the independent normal model, $\mathbf{W}^{-1}$ is a diagonal matrix of the standard deviations and y is the vector of observed responses. The linearized log likelihood is

$$L(\theta) = \frac{1}{n}\ \| \mathbf{W}^{\mathrm{T}}(\mathbf{y}-\theta) \|^2 \ .$$

The penalty J can often be written in a nonnegative definite quadratic form in δ (see Green and Yandell (1985)). We follow the spline literature and formulate it as

$$J(\theta) = J(\delta) = \delta^T \mathbf{K}_U \delta \text{ subject to } \mathbf{T}_U^T \delta = \mathbf{0} \ .$$

Typically the $k \times k$ matrix $\mathbf{K}_U$ and $k \times t$ matrix $\mathbf{T}_U$ are either derived from the unique design points or from a set of user-supplied basis nodes (see Appendix 2 of Bates et al. (1985)). If we write the parameter vector as

$$\theta = \mathbf{S}\alpha + \mathbf{T}\beta + \mathbf{K}\delta$$

in which $\mathbf{S}$ is the $n \times c$ covariate matrix, $\mathbf{T}$ is the $n \times t$ polynomial matrix, and $\mathbf{K}$ is the $n \times k$ smooth matrix, the linearized problem becomes (1.1).

We can locate the unique design points $\mathbf{T}_U$, and the corresponding unique covariates $\mathbf{S}_{1U}$, and form a QR decomposition

$$[\mathbf{T}_U : \mathbf{S}_{1U}] = \tilde{\mathbf{F}}\tilde{\mathbf{G}} = \tilde{\mathbf{F}}_1 \tilde{\mathbf{G}}_1 \ .$$

From this we construct the (unweighted) design

$$\mathbf{X} = [\mathbf{T} : \mathbf{S} : \mathbf{K}\tilde{\mathbf{F}}_2] \tag{2.2}$$

and penalty

$$\Sigma = \begin{bmatrix} 0 & 0 \\ 0 & \tilde{\mathbf{F}}_2^T \mathbf{K}_U \tilde{\mathbf{F}}_2 \end{bmatrix} \ . \tag{2.3}$$

We decompose Σ using a pivoted Cholesky followed by a Householder,

$$\mathbf{E}^T \Sigma \mathbf{E} = \mathbf{L}^T \mathbf{L} \quad \text{and} \quad \mathbf{L}^T = \mathbf{Q}\mathbf{R} = \mathbf{Q}_1 \mathbf{R}_1 \ , \tag{2.4}$$

and construct

$$\mathbf{Z} = [\mathbf{Z}_1 : \mathbf{Z}_2] = \mathbf{XEQ} \begin{bmatrix} \mathbf{R}_1^{-T} & 0 \\ 0 & \mathbf{I} \end{bmatrix} \ . \tag{2.5}$$

Finally, the original parameters are transformed to

$$\begin{bmatrix} \beta \\ \alpha \\ \tilde{\mathbf{F}}_2^T \delta \end{bmatrix} = \mathbf{E}\,\mathbf{Q} \begin{bmatrix} \mathbf{R}_1^{-T} & 0 \\ 0 & \mathbf{I} \end{bmatrix} \begin{bmatrix} \gamma \\ \omega \end{bmatrix} \ . \tag{2.6}$$

In the usual case that $\tilde{\mathbf{F}}_2^T \mathbf{K}_U \tilde{\mathbf{F}}_2$ is full rank, $\mathbf{E}\,\mathbf{Q}_2$ is an $n \times (c+t)$ matrix which permutes the coefficients α and β, i.e., $\omega^T = (\beta^T : \alpha^T : \mathbf{0})\,\mathbf{E}\,\mathbf{Q}_2$. The objective functional can now be reparameterized as

$$\frac{1}{n} \ \| \mathbf{W}^T(\mathbf{y} - \mathbf{Z}_2\omega - \mathbf{Z}_1\gamma) \|^2 + \lambda \gamma^T \gamma \ . \tag{2.7}$$

At this point, we have done all the "one-time" decompositions. The following steps must be redone each time $\mathbf{W}$ and $\mathbf{y}$ change, or simply once for the linear (normal) model. We form a QR decomposition of

$$\mathbf{W}^T \mathbf{Z}_2 = \mathbf{FG} = \mathbf{F}_1 \mathbf{G}_1 \ ,$$

and create

$$\mathbf{J} = [\mathbf{J}_1 : \mathbf{J}_2] = [\mathbf{F}_1^T : \mathbf{F}_2^T] \mathbf{W}^T \mathbf{Z}_1 \ ,$$

leading to the minimization of

$$\frac{1}{n} \| \mathbf{F}_1^T \mathbf{W}^T \mathbf{y} - \mathbf{G}_1 \omega - \mathbf{J}_1 \gamma \|^2 \tag{2.8}$$

$$+ \frac{1}{n} \| \mathbf{F}_2^T \mathbf{W}^T \mathbf{y} - \mathbf{J}_2 \gamma \|^2 + \lambda \gamma^T \gamma \ .$$

The first term can be made zero by solving for ω, with any given γ,

$$\mathbf{G}_1 \omega = \mathbf{F}_1^T \mathbf{W}^T \mathbf{y} - \mathbf{J}_1 \gamma \ . \tag{2.9}$$

The estimate of γ is found by solving

$$\mathbf{M}\gamma = \mathbf{J}_2^T \mathbf{F}_2^T \mathbf{W}^T \mathbf{y} \ , \tag{2.10}$$

with

$$\mathbf{M} = \mathbf{J}_2^T \mathbf{J}_2 + n\lambda \mathbf{I} \ .$$

The "hat" matrix can be formally written as

$$\mathbf{A}(\lambda) = \mathbf{W}^{-T}\mathbf{F} \begin{bmatrix} \mathbf{I} & 0 \\ 0 & \mathbf{J}_2 \mathbf{M}^{-1} \mathbf{J}_2^T \end{bmatrix} \mathbf{F}^T \mathbf{W}^T \tag{2.11}$$

provided we can invert $\mathbf{M}$. Naturally, one would iterate to new pseudo-values and pseudo-weights using (2.1) and repeat the minimization of the objective function (2.7). At convergence, one can obtain the estimates of the original parameters via (2.6).

One may approach the above solution for γ and the "hat" matrix $\mathbf{A}(\lambda)$ in different ways, depending on whether one wishes to choose a new λ, say via generalized cross validation, or whether one wishes to leave λ fixed.

2.1. SVD approach

One way to choose a new λ is based on generalized cross validation for the linearized problem (2.7). This is basically the ridge regression problem of Golub, Heath and Wahba (1979). Form a singular value decomposition of

$$\mathbf{J}_2 = \mathbf{UDV}^T \ ,$$

where $\mathbf{U}$ and $\mathbf{V}$ are orthogonal and $\mathbf{D}$ is diagonal, to get

$$\hat{\gamma} = \mathbf{V}(\mathbf{D}^2 + n\lambda \mathbf{I})^{-1} \mathbf{D} \mathbf{U}^T \mathbf{F}_2^T \mathbf{W}^T \mathbf{y} \ .$$

The "hat" matrix is

$$\mathbf{A}(\lambda) = \mathbf{W}^{-T}\mathbf{F} \begin{bmatrix} \mathbf{I} & 0 \\ 0 & \mathbf{U}\mathbf{D}^2(\mathbf{D}^2 + n\lambda \mathbf{I})^{-1} \mathbf{U}^T \end{bmatrix} \mathbf{F}^T \mathbf{W}^T \ .$$

One can choose λ to minimize the GCV criterion (Craven and Wahba, 1979)

$$V(\lambda) = \frac{n \ \| \mathbf{W}^T(\mathbf{I} - \mathbf{A}(\lambda))\mathbf{y} \|^2}{[tr(\mathbf{I} - \mathbf{A}(\lambda))]^2} \ , \tag{2.12}$$

or as some intermediate value if this is seen as being too "far" from the previous value.

2.2. Cholesky approach

If we choose to leave λ fixed, one can take the cheaper approach of a Cholesky decomposition of

$$\mathbf{M} = \mathbf{J}_2^T \mathbf{J}_2 + n\lambda \mathbf{I} = \mathbf{C}^T \mathbf{C} \ ,$$

leading to the estimate of γ by solving

$$\mathbf{C}^T \mathbf{C} \hat{\gamma} = \mathbf{J}_2^T \mathbf{F}_2^T \mathbf{W}^T \mathbf{y} \ .$$

The "hat" matrix becomes

$$\mathbf{A}(\lambda) = \mathbf{W}^{-T}\mathbf{F} \begin{bmatrix} \mathbf{I} & 0 \\ 0 & \mathbf{J}_2 \mathbf{C}^{-1} \mathbf{C}^{-T} \mathbf{J}_2^T \end{bmatrix} \mathbf{F}^T \mathbf{W}^T \ . \tag{2.13}$$

This route was followed by O'Sullivan, Yandell and Raynor (1986), iterating to a solution for fixed λ. The "optimal" λ was chosen by minimizing $V(\lambda)$ over a grid of $\log(\lambda)$.

3. Iteratively Reweighted Least Squares Models

Iteratively reweighted least squares (IRLS) models differ from semi-parametric GLMs in that only the penalty matrix remains fixed (Green, 1984). The log-likelihood parameter θ can be locally linearized, but the $\mathbf{S}$, $\mathbf{T}$, and $\mathbf{K}$ matrices are no longer fixed:

$$\mathbf{S} = \frac{\partial L}{\partial \alpha} \, , \ \mathbf{T} = \frac{\partial L}{\partial \beta} \, , \ \mathbf{K} = \frac{\partial L}{\partial \delta} \ .$$

We still only need form and decompose Σ as in (2.3) and (2.4) exactly once. However, the (unweighted) design (2.2) may change with each iteration. Hence, the remaining computations need to be done at each iteration. One could proceed in the same manner as for the generalized linear models, but reconstructing **X**, and hence **Z** and **J**, each time.

4. General Nonlinear Models

General nonlinear problems could proceed in the same manner as for IRLS, except that $\mathbf{K}_U$ changes each time. Thus most computations need to be redone. It may be possible for some nonlinear problems to reparameterize them as SGLM or IRLS problems to eliminate this difficulty.

5. Diagnostics

The diagonal elements of the "hat" matrix have been used for diagnostics in generalized linear models (Pregibon, 1981) as well as in smoothing spline models (Eubank 1984, 1985). It is natural to think of extending these uses to the present array of models (Green and Yandell, 1985; Green, 1985). The diagonal elements can be computed as

$$\{\mathbf{A}(\lambda)\}_{ii} = \| \mathbf{F}_1^T \mathbf{e}_i \|^2 + \| \mathbf{M}^{-1/2} \mathbf{F}_2^T \mathbf{e}_i \|^2$$

in which $\mathbf{e}_i$ is the n-vector with a 1 in the *i-th* position and 0's elsewhere. For the SVD approach this is simply

$$\{\mathbf{A}(\lambda)\}_{ii} = \| \mathbf{F}_1^T \mathbf{e}_i \|^2 + \| D(D + n\lambda \mathbf{I})^{-1/2} \mathbf{U}^T \mathbf{F}_2^T \mathbf{e}_i \|^2 \ ,$$

and for the Cholesky approach (*cf.* O'Sullivan (1985)),

$$\{\mathbf{A}(\lambda)\}_{ii} = \| \mathbf{F}_1^T \mathbf{e}_i \|^2 + \| \mathbf{C}^{-T} \mathbf{J}_2^T \mathbf{F}_2^T \mathbf{e}_i \|^2 \ .$$

Covariance matrices can be computed by noting that $COV(\mathbf{y}) = \mathbf{W}^{-T}\mathbf{W}^{-1}$. We find from (2.11) that

$$COV(\hat{\theta}) = \mathbf{W}^{-T}\mathbf{F} \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{J}_2\mathbf{M}^{-1}\mathbf{J}_2^T\mathbf{J}_2\mathbf{M}^{-T}\mathbf{J}_2^T \end{pmatrix} \mathbf{F}^T\mathbf{W}^{-T} \ .$$

Hence, the variances are

$$VAR(\theta_i) = \| \mathbf{F}_1^T \mathbf{W}^{-1} \mathbf{e}_i \|^2 + \| \mathbf{J}_2 \mathbf{M}^{-T} \mathbf{J}_2^T \mathbf{F}_2^T \mathbf{W}^{-1} \mathbf{e}_i \|^2 \ .$$

Noting the relation

$$\mathbf{M}^{-1}\mathbf{J}_2^T\mathbf{J}_2\mathbf{M}^{-T} = \mathbf{M}^{-1}(\mathbf{I} - n\lambda\mathbf{M}^{-1}) \ ,$$

the variances can be written as

$$VAR(\theta_i) = \| \mathbf{F}_1^T \mathbf{W}^{-1} \mathbf{e}_i \|^2 + \| \mathbf{C}^{-T} \mathbf{J}_2^T \mathbf{F}_2^T \mathbf{W}^{-1} \mathbf{e}_i \|^2 - n\lambda \| \mathbf{C}^{-1} \mathbf{C}^{-T} \mathbf{J}_2^T \mathbf{F}_2^T \mathbf{W}^{-1} \mathbf{e}_i \|^2$$

for the Cholesky approach. For the SVD we have

$$VAR(\theta_i) = \| \mathbf{F}_1^T \mathbf{W}^{-1} \mathbf{e}_i \|^2 + \| \mathbf{D}^2(\mathbf{D}^2 + n\lambda\mathbf{I})^{-1} \mathbf{U}^T \mathbf{F}_2^T \mathbf{W}^{-1} \mathbf{e}_i \|^2 \ .$$

The covariance among the coefficients can be derived, using (2.9), (2.10) and (2.6), as

$$COV \begin{pmatrix} \beta \\ \alpha \\ \bar{\mathbf{F}}_2^T\delta \end{pmatrix} = \mathbf{E}\mathbf{Q}_2\mathbf{G}_1^{-1}\mathbf{G}_1^{-T}\mathbf{Q}_2^T\mathbf{E}^T +$$

$$\mathbf{E}\mathbf{Q} \begin{bmatrix} \mathbf{R}_1^{-T} \\ -\mathbf{G}_1^{-1}\mathbf{F}_1^T\mathbf{W}^T \end{bmatrix} \mathbf{M}^{-1}\mathbf{J}_2^T\mathbf{J}_2\mathbf{M}^{-T} \begin{bmatrix} \mathbf{R}_1^{-T} \\ -\mathbf{G}_1^{-1}\mathbf{F}_1^T\mathbf{W}^T \end{bmatrix}^T \mathbf{Q}^T\mathbf{E}^T \ .$$

In many situations we may be only interested in $COV(\alpha)$. Further, if the penalty Σ is of the proper rank, then the QR decomposition of (2.4) should simply permute the indices for the coefficients. In other words, $\mathbf{EQ}_2$ often simply permutes the coefficients α (and β) into ω. In this case, let $\bar{e}_i$ denote the permutation for α_i, $i = 1, \cdots, c$. For the SVD approach,

$$VAR(\alpha_i) = \| \mathbf{G}_1^{-T} \bar{e}_i \|^2 + \| \mathbf{D}(\mathbf{D}^2 + n\lambda\mathbf{I})^{-1} \mathbf{V}^T \mathbf{W} \mathbf{F}_1 \mathbf{G}_1^{-T} \bar{e}_i \|^2 \ .$$

For the Cholesky approach,

$$VAR(\alpha_i) = \| \mathbf{G}_1^{-T} \bar{e}_i \|^2 + \| \mathbf{C}^{-T} \mathbf{W} \mathbf{F}_1 \mathbf{G}_1^{-T} \bar{e}_i \|^2 - n\lambda \| \mathbf{C}^{-1} \mathbf{C}^{-T} \mathbf{W} \mathbf{F}_1 \mathbf{G}_1^{-T} \bar{e}_i \|^2 \ .$$

Joint work is in progress with Peter J. Green (Green and Yandell, 1985) on analogues to diagnostic tools for generalized linear models along the lines of Pregibon (1981, 1982) and Nelder and Pregibon (1986).

6. Numerical Comparisons

We focus our investigations upon the Poisson and binomial special cases of the semi-parametric generalized linear model as these are potentially of wide interest and easy to formulate. We allowed up to c initial iterations of the Cholesky decomposition (CD) for $\lambda = \infty$ (perfectly smooth case), and up to c CDs following each SVD, where c was 1, 2, or 10. No case required more than 7 CD following an SVD, or more than 7 SVD overall.

We examined some real data on leafhopper oviposition and potato pathogen in a field, both Poisson, and data on rat survival, which was binomial. In addition we simulated data which we thought might be "cumbersome" for the numerical algorithms. The simulations were Poisson with a normal shaped carve of θ = log(mean value), with peak height of between $\theta = 1.5$ and 20. Binomial simulations used a similar normal shaped curve for θ = logit(mean value), with peak height of between θ = logit(.01) and logit(.3). Simulations were conducted for $n = 50$ and 100.

The Cholesky steps in the real examples increased the run time by 20-35%, including one-time costs and construction of the diagonals of the "hat" matrix (see Tables 1-3). This occurred because the number of SVDs was not reduced by more intermediate CDs, nor were the sequences of optimal λ's for the linearized problems markedly altered by the CDs. In addition, each CD took about 10% of the time for an SVD. In these examples, the signal was fairly apparent, indicating that the linear approximation was adequate using the SVD iterations alone.

Table 1. Poisson Oviposition Data (n=27)

task	c=0	c=1	c=2	c=10
one-time	4.40	4.40	4.43	4.50
cholesky	0.78	4.22	7.78	11.92
svd	24.93	25.02	24.73	24.78
hat	2.20	2.22	2.23	2.22
total	31.07	34.57	37.85	42.10
no. svd	5	5	5	5
no. chol	1	6	11	19

Table 2. Binomial Rats Data (n=127)

task	c=0	c=2	c=10
one-time	34.6	33.9	35.0
cholesky	7.5	58.7	74.2
svd	245.0	245.6	243.9
hat	34.6	34.8	35.2
total	312.8	364.2	379.3
no. svd	5	5	5
no. chol	1	9	12

Table 3. 2-D Poisson Fungi (n=400, k=100)

task	c=0	c=2	c=10
one-time	279	279	283
cholesky	140	1004	1475
svd	4486	4413	4425
hat	594	598	598
total	5354	6150	6637
no. svd	7	7	7
no. chol	2	16	26

The simulations showed that when the "signal" is small relative to the "noise", the CDs seem to stabilize the minimization problem, reducing the number of SVDs required and cutting the run time. Table 4(a-b) present the combined CD and SVD run times, while Table 4(c-d) present the numbers of SVDs and CDs. As the height of the Poisson peak rises, the CD iterations have a reduced impact on convergence. However, note that on several occassions iteration with only one CD increased the number of SVDs required. Allowing more than 2 CD steps only seemed to increase the overall run time; the number of SVDs was reduced in only a few instances. In addition, a few simulations, not shown here, converged when up to 2 CDs per SVD were allowed, but did not converge when 0 or up to 10 were allowed. Similar statements can be made about the binomial simulations (Table 5(a-b)).

Table 4(a). Poisson Run Times (n=50)

peak	c=0	c=1	c=2	c=10
1.5	134	120	94	103
2	163	150	130	141
2.5	134	148	126	134
3	132	148	125	138
4	159	178	155	142
5	158	180	157	144
6	131	173	155	120
7	133	159	127	161
8	131	175	157	141
9	135	178	158	144
10	157	204	188	174
15	134	180	187	181
20	158	207	189	175

Table 4(b). Poisson Run Times (n=100)

peak	c=0	c=1	c=2	c=10
1.5	974	848	885	904
2	950	834	880	933
2.5	1149	1051	1098	932
3	759	824	659	718
4	956	1048	882	967
5	955	1069	1100	988
6	970	1244	915	1006
7	938	1038	873	970
8	939	1053	1105	1043
9	955	1280	1138	1026
10	1129	1245	1106	1371
15	941	1252	1109	762
20	962	1276	1131	1143

Table 4(c). Poisson Runs (n=50)

	no. SVD / no. CD iterations			
peak	c=0	c=1	c=2	c=10
1.5	5/0	4/4	3/5	3/10
2	6/1	5/6	4/8	4/12
2.5	5/0	5/5	4/7	4/12
3	5/0	5/5	4/7	4/13
4	6/0	6/6	5/8	4/15
5	6/0	6/7	5/9	4/15
6	5/0	6/6	5/9	3/16
7	5/0	5/5	4/7	4/19
8	5/0	6/6	5/9	4/14
9	5/1	6/6	5/9	4/15
10	6/0	7/7	6/10	5/16
15	5/1	6/7	6/10	5/18
20	6/0	7/7	6/10	5/16

Table 4(d). Poisson Runs (n=100)

	no. SVD / no. CD iterations			
peak	c=0	c=1	c=2	c=10
1.5	5/0	4/4	4/6	4/10
2	5/0	4/4	4/7	4/12
2.5	6/0	5/5	5/8	4/11
3	4/0	4/4	3/6	3/11
4	5/0	5/5	4/7	4/13
5	5/0	5/6	5/8	4/14
6	5/1	6/6	4/9	4/15
7	5/0	5/5	4/7	4/14
8	5/0	5/6	5/9	4/17
9	5/0	6/7	5/9	4/16
10	6/0	6/6	5/9	5/23
15	5/0	6/6	5/9	3/13
20	5/0	6/6	5/9	4/19

Table 5(a). Binomial Run Times (n=100)

size	prob	c=0	c=1	c=2	c=10
10	.3	108	87	90	91
	.2	106	118	125	131
	.1	133	118	92	97
	.05	135	148	130	135
20	.3	109	91	92	96
	.2	137	119	123	127
	.1	109	120	124	129
	.05	165	151	159	168

Table 5(b). Binomial Run Times (n=100)

size	prob	c=0	c=1	c=2	c=10
10	.3	943	827	671	692
	.2	970	829	858	882
	.1	968	860	885	937
	.05	1171	1064	898	977
	.01	1166	1046	1097	935
20	.3	743	604	632	635
	.2	760	617	636	645
	.1	780	838	650	680
	.05	795	849	681	742
	.01	1351	1261	1103	1225
	.005	1513	1676	1536	1683

Table 5(c). Binomial Runs (n=50)

		no. SVD / no. CD iterations			
size	prob	c=0	c=1	c=2	c=10
10	.3	5/0	4/4	3/5	3/8
	.2	4/0	4/4	4/6	4/9
	.1	4/0	3/3	3/5	3/6
	.05	5/0	5/5	4/8	4/11
20	.3	4/0	4/4	4/6	4/9
	.2	5/1	4/4	4/6	4/8
	.1	4/1	3/4	3/5	3/7
	.05	6/0	5/5	5/8	5/12

Table 5(d). Binomial Runs (n=100)

		no. SVD / no. CD iterations			
size	prob	c=0	c=1	c=2	c=10
10	.3	5/0	4/4	3/6	3/8
	.2	5/0	4/4	4/6	4/8
	.1	5/0	4/5	4/7	4/11
	.05	6/0	5/5	4/7	4/12
	.01	6/0	5/5	5/8	4/11
20	.3	4/0	3/3	3/5	3/6
	.2	4/1	3/3	3/5	3/7
	.1	4/1	4/4	3/5	3/8
	.05	4/0	4/4	3/6	3/10
	.01	7/0	6/6	5/8	5/14
	.005	8/0	8/8	7/11	7/20

Since we know that the estimates converge for fixed λ (O'Sullivan, Yandell and Raynor, Jr., 1986), a few iterations for fixed λ may guard against nonlinearity in the penalized likelihood. It is not known at this time what conditions are required on the penalized likelihood, as a function of λ, to insure convergence in the SVD-only approach.

If one follows Elden (1984) to stop the singular value decomposition after the bidiagonalization, considerable time can be saved since the effort to diagonalize is magnified by the number of iterations. Earlier work on GCVPACK (Bates et al., 1985) indicated that half of the singular value decomposition time may be spent on bidiagonalization. Of course, once convergence is reached, one could complete the diagonalization, doing this only once, to easily derive the diagonal of the "hat" matrix. Such a savings in computation would further reduce the advantage of iterating via Cholesky with fixed λ.

Acknowledgements

This research has been supported in part by United States Department of Agriculture CSRS grant 511-100, and National Sciences Foundation grant DMS-8404970. Computing was performed on the UW-Madison Statistics VAX 11/750 Research Computer.

References

Bates, D. M., Lindstrom, M. J., Wahba, G. and Yandell, B. S. (1985) GCVPACK – Routines for Generalized Cross Validation. Technical Report#775, Dept. of Statistics, U. of Wisconsin.

Cox, D. D. (1983) Gaussian approximation of smoothing splines. Technical Report#743, Dept. of Statistics, U. of Wisconsin.

Craven, P. and Wahba, G. (1979) Smoothing noisy data with spline functions: estimating the correct degree of smoothing by the method of generalized cross-validation. *Numer. Math.*, **31**, 377-403.

Elden, L. (1984) A note on the computation of the generalized cross-validation function for ill-conditioned least squares problems. *BIT*, **24**, 467-472.

Eubank, R. L. (1984) The hat matrix for smoothing splines. *Statist. and Prob. Letters*, **2**, 9-14.

Eubank, R. L. (1985) Diagnostics for smoothing splines. *J. Roy. Statist. Soc. Ser. B*, **47**. (to appear)

Golub, G. H., Heath, M. and Wahba, G. (1979) Generalised cross validation as a method for choosing a good ridge parameter. *Technometrics*, **21**, 215-224.

Good, I. J. and Gaskins, R. A. (1971) Non-parametric roughness penalties for probability densities. *Biometrika*, **58**, 255-277.

Green, P. J. (1984) Iteratively reweighted least squares for maximum likelihood estimation and some robust and resistant alternatives (with discussion). *J. Roy. Statist. Soc. Ser. B*, **46**, 149-192.

Green, P. J. (1985) Penalized likelihood for general semi-parametric regression models. Technical Report#2819, Math. Research Center, U. of Wisconsin.

Green, P. J., Jennison, C. and Seheult, A. H. (1983) Contribution to the discussion of the paper by Wilkinson et al. *J. Roy. Statist. Soc. Ser. B*, **45**. 193-195.

Green, P. J. and Yandell, B. S. (1985) Semi-parametric generalized linear models. In *GLIM85: Proceedings of the International Conference on Generalized Linear Models, September 1985* (R. Gilchrist, ed.) Lecture Notes in Statistics, Springer-Verlag. (Technical Report#2847, Math. Res. Cen., U. of Wisconsin)

Leonard, T. (1982) An empirical Bayesian approach to the smooth estimation of unknown functions. Technical Report#2339, Math. Research Center, U. of Wisconsin.

Nelder, J. A. and Pregibon, D. (1986) An extended quasi-likelihood function. Unpublished manuscript.

O'Sullivan, F. (1983) The analysis of some penalized likelihood schemes. Technical Report#726, Dept. of Statistics, U. of Wisconsin.

O'Sullivan, F. (1985) Contribution to the discussion of the paper by Silverman. *J. Roy. Statist. Soc. Ser. B*, **47**, 39-40.

O'Sullivan, F., Yandell, B. S. and Raynor, Jr., W. J. (1986) Automatic smoothing of regression functions in generalized linear models. *J. Amer. Statist. Assoc.*, **81**, 000-000. (Technical Report#734, Statistics Dept., U. of Wisconsin)

Pregibon, D. (1981) Logistic regression diagnostics. *Ann. Statist.*, **9**, 705-724.

Pregibon, D. (1982) Score tests in GLIM. In *Proc. GLIM82 Conf.* (R. Gilchrist, ed.) New York: Springer-Verlag.

Speckman, P. (1985) Spline smoothing and optimal rates of convergence in nonparametric regression models. *Ann. Statist.*, **13**, 970-983.

APPENDIX A. SYMPOSIUM ATTENDANTS

JIM ALLEN
JOINER ASSOCS.
P.O. BOX 5445
MADISON, WI 53705

DAVID ALLEN
UNIV. OF KY
DEPT. OF STATISTICS
LEXINGTON, KY 40506

JOAN M. ALSTER
UNIV OF IL COMPUTING SVCS CTR
1304 W. SPRINGFIELD AVE.
URBANA, IL 61801

JANA ANDERSON
CSU
STATISTICS DEPT.
FORT COLLINS, CO 80523

SCOTT ATKINSON
UNIV. OF WYOMING
P.O. BOX 3925
LARAMIE, WY 82071

RAY BAILEY
US PUBLIC HEALTH SERVICE
P.O. BOX 2087
FORT COLLINS, CO 80522

ALFRED BALCH
COLORADO STATE UNIV.
STATISTICS DEPT.
FORT COLLINS, CO 80521

DAVID BALSIGER
JOINER ASSOCS.
P.O. BOX 5445
MADISON, WI 53705

JOHN BARKMEIER
COLORADO SCHOOL OF MINES
MATH DEPT.
GOLDEN, CO 80401

RICHARD BARNES
PACIFIC GAS AND ELECTRIC CO.
77 BEALE ST. RM 1113
SAN FRANCISCO, CA 94106

JIM BAYLIS
EASTMAN KODAK
BLDG. C-42
WINDSOR, CO 80551

RICHARD BECKER
AT&T BELL LABORATORIES
600 MOUNTAIN AVE.
MURRAY HILL, NJ 07974

MAX BENSON
UNIV. OF MN
10 UNIV. DR.
DULUTH, MN 55812

JON BENTLEY
AT&T BELL LABS
RM. 2C-317
MURRAY HILL, NJ 07974

KEN BERK
DEPT. OF MATH
IL STATE UNIV.
NORMAL, IL 61761

KENNETH BERRY
DEPT. OF SOCIOLOGY
CSU
FT. COLLINS, CO 80523

LYNNE BILLARD
UNIVERSITY OF GEORGIA
DEPT. OF STATISTICS
ATHENS, GA 30602

RON BING
IBM
5600 N. 53RD ST.
BOULDER, CO 80314

RON BIONDINI
TRW
14241 E. 4TH AVE.
AURORA, CO 80011

THOMAS BOARDMAN
COLORADO STATE UNIVERSITY
207 OLD ECON BLDG.
FORT COLLINS, CO 80523

DUANE BOES
DEPT. OF STATISTICS
COLORADO STATE UNIV.
FORT COLLINS, CO 80523

DANKMAR BOHNING
215 POND LABORATORY
UNIVERSITY PARK, PA 16802

M.T. BOSWELL
PSU
301 POND LAB
UNIV. PARK, PA 16802

DAVID BOWDEN
COLORADO STATE UNIV.
216 OLD ECON BLDG.
FORT COLLINS, CO 80523

CELEDONIO A. BRAVO
COLORADO STATE UNIV.
DEPT. OF FOREST & WOOD
FT. COLLINS, CO 80523

LEO BREIMAN
STATISTICS
UNIV. OF CA - BERKELEY
BERKELEY, CA 94707

V.N. BRINGI
COLORADO STATE UNIV.
DEPT. OF ELECTRICAL ENGR.
FORT COLLINS, CO 80523

PETER BROCKWELL
COLORADO STATE UNIV-STAT DEPT
RM. 100 OLD ECON. BLDG.
FORT COLLINS, CO 80523

BARBARA G. BROWN
OREGON STATE UNIV. & NCAR
NCAR P.O. BOX 3000
BOULDER, CO 80307

MARK BRYAN
IOWA STATE UNIVERSITY
102 SWEDECOR HALL
AMES, IA 50011

PETER BRYANT
UNIV. OF COLORADO
1475 LAWRENCE ST.
DENVER, CO 80202

G. REX BRYCE
BYU
216 TMCB
PROVO, UT 84602

JUDITH A. BUCHINO
10305 TURNSTILE CT.
LOUISVILLE, KY 40223

DAVID BUNCH
UNIVERSITY OF CALIFORNIA
308 VOORHIES
DAVIS, CA 95616

DAVID BURN
IMSL, INC.
2500 CITYWEST BLVD.
HOUSTON, TX 77042

CAL BUTLER
COLLEGE OF S. ID
P.O. BOX 1238
TWIN FALLS, ID 83301

RICHARD BYRD
DEPT. OF COMPUTER SCIENCE
UNIV. OF COLO.-BOX 430
BOULDER, CO 80309

H. CALLAERT
LIMBURGS UNIVERSITAIR CENTRUM
UNIVERSITAIRE CAMPUS
B-3610 DIEPENBEEK,BELGIUM,

JAMES CALVIN
UNIV. OF IOWA
DEPT. OF STATISTICS
IOWA CITY, IA 52242

PATRICIA CAMP
THE RAND CORP.
1700 MAIN ST.
SANTA MONICA, CA 90406

DANIEL CARR
PACIFIC N.W. LAB.
P.O. BOX 999
RICHLAND, WA 99352

V. CHANDRASEKAR
CSU
DEPT. OF ELEC. ENGR.
FT. COLLINS, CO 80523

NANCY CLARK
LANGUAGE OF DATA PROJECT
P.O. BOX 673
SAUSALITO, CA 94966

WILLIAM CLARKE
UNIVERSITY OF IOWA
DEPT. OF PREV. MEDICINE
IOWA CITY, IA 52242

DOUGLAS CLARKSON
IMSL, INC.
2500 CITYWEST BLVD.
HOUSTON, TX 77042

TIM COBURN
MARATHON OIL CO.
P.O. 269
LITTLETON, CO 80122

DAVID COLEMAN
RCA
W-217 DSRC RCA LABS
PRINCETON, NJ 08540

WILLIAM COMMINS
NATL. SCIENCE FOUNDATION
1800 E. ST. NW-RM. 425
WASHINGTON, DC 20817

RICHARD COOK
UNIV. OF COLORADO
DATA ANALYSIS CENTER
BOULDER, CO 80309-0486

PAULA J. COWLEY
BATTELLE NW LABORATORIES
PO BOX 999-MATH/1129/3000
RICHLAND, WA 99352

DENNIS D. COX
UNIV. OF IL-DEPT. OF STAT.
1409 W. GREEN ST.
URBANA, IL 61801

CSU
COLORADO STATE UNIVERSITY
204 OLD ECON BLDG.
FORT COLLINS, CO 80523

REE DAWSON
MIT
11-317, MIT
CAMBRIDGE, MA 02139

HARI H. DAYAL
FOX CHASE CANCER CENTER
7701 BURHOLME AVE.
PHILADELPHIA, PA 19111

BRUCE DE BLOIS
USAF ACADEMY
COLORADO SPRINGS, CO 80840

LORRAINE DENBY
AT&T BELL LABS.
600 MT. AVE. RM. 2C273
MURRAY HILL, NJ 07922

DAVID DOEHLERT
EDGEWORK, INC.
16514 N.E. 44TH WAY
REDMOND, WA 98052

JANET DONALDSON
NATL. BUREAU OF STANDARDS
325 BROADWAY
BOULDER, CO 80303

DAVID DONOHO
UNIV. OF CALIFORNIA
DEPT. OF STATISTICS
BERKELEY, CA 94720

MARY DOWNTON
NATL. CENTER FOR ATMSPH. RSCH.
P.O. BOX 3000
BOULDER, CO 80307

MICHAEL F. DRISCOLL
AZ STATE UNIV.
DEPT. OF MATH
TEMPE, AZ 85287

BONNIE DUMAS
WESTVACO CORP.
P.O. BOX 1950
SUMMERVILLE, SC 29484

WILLIAM H. DUMOUCHEL
HARVARD UNIV.
77 SNAKE HILL RD.
BELMONT, MA 02178

MARK DURST
LAWRENCE LIVERMORE LAB
P.O. BOX 808
LIVERMORE, CA 94550

WILLIAM EDDY
CARNEGIE-MELLON UNIV.
DEPT. OF STATISTICS, CMU
PITTSBURGH, PA 15213

BEN EISEMAN
UNIV. OF CO HEALTH-SURGERY DEP
4200 E 9TH
DENVER, CO 80220

LASZLO ENGELMAN
BMDP STATISTICAL SOFTWARE, INC
1964 WESTWOOD BLVD.,#202
LOS ANGELES, CA 90025

MARK FELTHAOSER
COLORADO STATE UNIV.
DEPT. OF STATISTICS
FORT COLLINS, CO 80523

RONALD FICHTNER
CENTERS FOR DISEASE CONTROL
1600 CLIFTON RD. N.E.
ATLANTA, GA 30307

LYNDA FINN
JOINER ASSOCS.
P.O. BOX 5445
MADISON, WI 53705

PHILIP FLEMING
AT&T
1100 E. WARRENVILLE RD.
NAPERVILLE, IL 60566

ED FORGY
1024 GRETNA GREEN WAY
LOS ANGELES, CA 90049

VIKKI FRENCH
ASA
COMM DEPT-HERSHEY CORP.
HERSHEY, PA 17033

DENNIS FRIDAY
NATL. BUREAU OF STANDARDS
325 BROADWAY
BOULDER, CO 80303

WAYNE A. FULLER
IA STATE UNIV.
STAT LAB 221 SNEDECOR HAL
AMES, IA 50011

DAVID FURMAN
MINITAB, INC.
3081 ENTERPRISE DRIVE
STATE COLLEGE, PA 16801

RUBEN GABRIEL
UNIVERSITY OF ROCHESTER
ROCHESTER, NY 14627

LIONEL GALWAY
DEPT. OF STATISTICS
CARNEGIE-MELLON UNIV.
PITTSBURGH, PA 15213

FAH FATT GAN
UNIV. OF CENTRAL FLORIDA
DEPT. OF STATISTICS
ORLANDO, FL 32816-6990

MIRIAM GASKO
UNIV. OF CHICAGO
GSB DEPT.
CHICAGO, IL 60637

DAVID M. GAY
AT&T BELL LABS
600 MT. AVE.
MURRAY HILL, NJ 07974

DONALD GEMAN
UNIV. OF MASSACHUSETTS
DEPT. OF MATHEMATICS
AMHERST, MA 01003

JAMES E. GENTLE
IMSL, INC.
2500 CITY W. BLVD.
HOUSTON, TX 77042

LEON GILFORD
AMERICAN STATISTICAL ASSOC.
11510 GEORGIA AVE.
WHEATON, MD 20902

BILL GILLETT
UNIV. OF MO - ROLLA
DEPT. OF CSC-UMR
ROLLA, MO 65401

SHERI GILLEY
SPSS INC.
444 N. MICHIGAN AVE.
CHICAGO, IL 60611

JIM GOIN
GEOMETRIC DATA
999 W. VALLEY RD.
WAYNE, PA 19087

KENNETH M. GOLDBERG
WYETH LABS.
P.O. BOX 8299
PHILADELPHIA, PA 19101

HENSON GRAVES
SAN JOSE STATE UNIVERSITY
885 N. SAN ANTONIO RD.
LOS ALTOS, CA 94022

DAVID GRAY
UNIV. OF KENTUCKY
DEPT. OF STATISTICS
LEXINGTON, KY

DANIEL A. GREER
ROCKWELL INTERNATIONAL
P.O. BOX 464
GOLDEN, CO 80402

JOHN GREGO
MINITAB, INC.
3081 ENTERPRISE DR.
STATE COLLEGE, PA 16801

MARTHA J. GREINER
STATISTICIAN
2800 PLYMOUTH RD.
ANN ARBOR, MI 48105

JEFF GRIMM
FRT RSCH LAB-PA STATE UNIV
PO BOX 309-UNIV DR
BIGLERVILLE, PA 17307

YVES GRIZE
AT&T BELL LABORATORIES
CRAWFORDS CORNER RD.
HOLMDEL, NJ 07733

ARTHUR GURNEY
EASTMAN KODAK
1669 LAKE AVE.
ROCHESTER, NY 14650

TIM HAAS
COLORADO STATE UNIV-STAT DEPT
101 OLD ECON BLDG.
FORT COLLINS, CO 80523

EDWARD J. HALTEMAN
ROCKWELL INTERNATIONAL
P.O. BOX 464
GOLDEN, CO 80402

RANDY HAMLIN
BEATRICE GROCERY GROUP
1645 W. VALENCIA DR.
FULLERTON, CA 92633

BOB HAMMOND
ADOLPH COORS CO
NH5200
GOLDEN, CO 80401

JANIS HARDWICK
BMDP STATISTICAL SOFTWARE, INC
1964 WESTWOOD BLVD., #202
LOS ANGELES, CA 90025

PAMELA HARTIGAN
ASA
CSPCC VA MEDICAL CENTER
WEST HAVEN, CT 06516

JOHN HARTIGAN
ASA
STAT. DEPT. YALE UNIV.
NEW HAVEN, CT 06520

H.B. HARVEY
US ARMY DUGWAY PROVING GROUND
P.O. BOX 53
STOCKTON, UT 84071

MAUREEN HASCHKE
UNIV OF COLO HEALTH SCIENCES
4200 E 9 AVE, BOX C245
DENVER, CO 80262

RICHARD HEIBERGER
TEMPLE UNIVERSITY
DEPT. OF STATISTICS
PHILADELPHIA, PA 19122

KARL HEINER
STATE UNIV. OF NEW YORK
HUMAN INFO. SYSTEMS
SCHENECTADY, NY 12308

DANIEL HEITJAN
UCLA
DEPT. OF BIOMATH CHSAV617
LOS ANGELES, CA 90024

PATRICK HENAFF
CCREMS MIT
E40-134 MIT
CAMBRIDGE, MA 02139

MARY ANN HILL
BMDP STATISTICAL SOFTWARE, INC
1964 WESTWOOD BLVD.,#202
LOS ANGELES, CA 90025

ERIC HORVITZ
STANFORD UNIVERSITY
TC-135 SUMC
STANFORD, CA 94305

JAMES HOSKING
UNIV. OF NORTH CAROLINA
DEPT. OF BIOSTATISTICS
CHAPEL HILL, NC 27510

DOUGLAS HOWELL
HEWLETT-PACKARD
3404 E. HARMONY RD.
FORT COLLINS, CO 80525

JOHN HSIEH
UNIVERSITY OF TORONTO
DEPT. OF PREVENTIVE MED.
TORONTO, ONTARIO
CANADA, M5S 1A8

DAVID C. HUANG
THE UPJOHN CO.
BJOST\7293-32,2
KALAMAZOO, MI 49001

EARL HUGHES
COLO. DIV. OF LOCAL GOVERNMENT
1313 SHERMAN ST.
DENVER, CO 80203

FRED HULTING
IOWA STATE UNIVERSITY
117 SNEDECOR HALL
AMES, IA 50011

CATHERINE HURLEY
UNIV. OF WASHINGTON-SEATTLE
GN22 PADLFORD
SEATTLE, WA 98195

TAUKAR HUSSAIN
NY UNIV. STAT. & OPS. RESEARCH
100 TRINIDY PL.
NY, NY 10006

SYNI-AN HWANG
STATE UNIV. OF NEW YORK
DEPT. OF MATH & STATISTIC
ALBANY, NY 12222

PETER B. IMREY
UNIV. OF ILLINOIS
506 S. MATHEWS AVE.
URBANA, IL 61801

HARI IYER
DEPT. OF STATISTICS
COLORADO STATE UNIV.
FORT COLLINS, CO 80523

ALAN J. IZENMAN
TEMPLE UNIV.
DEPT. OF STATISTICS
PHILADELPHIA, PA 19122

JACK JAMES
G.D. SEARLE & CO.
4901 SEARLE PKWY.
SKOKIE, IL 60077

TERRY JEEVES
WESTINGHOUSE R&D CENTER
1310 BEULAH RD.
PITTSBURG, PA 15235

HOLLY JIMISON
STANFORD MEDICAL CENTER
STANFORD MEDL. CNTR.TC135
STANFORD, CA 94305

ROGER W. JOHNSON
ASA, MATH DEPT.
UNIV OF STHN COLORADO
PUEBLO, CO 81001

MARK JOHNSON
THE UPJOHN CO.
HENRIETTA ST.
KALAMZOO, MI 49001

JEFF JOHNSON
EASTMAN KODAK
BLDG. C-42
WINDSOR, CO 80551

RICHARD JONES
UNIV. OF COLORADO
SCHOOL OF MED. BOX B-119
DENVER, CO 80262

LAWRENCE JONES
ITHACA COLLEGE
ACAD. COMPUTING DEPT.
ITHACA, NY 14850

HONGSUK JORN
UNIVERSITY OF WISCONSIN
1210 W. DAYTON ST. #3202
MADISON, WI 53706

STEVE D. JOST
DEPAUL UNIVERSITY
243 S. WABASH AVE.
CHICAGO, IL 60604-2302

ROBERT JUDISH
NATL. BUREAU OF STANDARDS
325 BROADWAY
BOULDER, CO 80303

KAREN KAFADAR
HEWLETT-PACKARD
1501 PAGE MILL RD., 4U
PALO ALTO, CA 94304

WILLIAM KAHN
YALE UNIVERSITY
P.O. BOX 2179
NEW HAVEN, CT 06511

ALEX KASK
ERNST & WHINNEY
153 E. 53RD. ST.
NEW YORK, NY 10022

MOHAMMAD K. KAZEMPOUR
COLORADO STATE UNIV.
DEPT. OF STATISTICS
FORT COLLINS, CO 80521

THOMAS KEEFE
COLORADO STATE UNIV.
SPRUCE HALL
FT. COLLINS, CO 80523

ELIZABETH KELLY
LOS ALAMOS NATL. LAB
P.O. BOX 1663
LOS ALAMOS, NM 87545

WILLIAM KENNEDY
IOWA STATE UNIVERSITY
117 SNEDECOR STAT. DEPT.
AMES, IA 50011

DAVE KENNON
THE FAIR, ISAAC COMPANIES
120 N. REDWOOD DR.
SAN RAFAEL, CA 94903-1996

JIM KENYON
UNIV. OF CONNECTICUT
DEPT. OF STAT. U-120
STORRS, CT 06268

ELAINE M. KERAMIDAS
BELL COMMUNICATIONS RESEARCH
RM. 2N-335, 435 S. ST.
MORRISTOWN, NJ 07960-1961

RUDY M. KING
U.S. FOREST SERVICE
240 W. PROSPECT ST.
FT. COLLINS, CO 80526

LILLIAM KINGSBURY
VILLANOVA UNIV.
MATH SCIENCES DEPT.
VILLANOVA, PA 19085

JON KOGUT
DEPT. OF LABOR-MSHA (SHTC)
P.O. BOX 25367
DENVER, CO 80225-0367

JOHN E. KOONTZ
NBS
MAIL CD 714; 325 BROADWAY
BOULDER, CO 80303

BASIL KORIN
AMERICAN UNIV.
MATH DEPT. CLARK HALL
WASHINGTON, DC 20016

JOHN KORK
U.S. GEOLOCIGAL SURVEY
BX. 25046 MAIL STOP 937
DENVER, CO 80225

ROBERT KOYAK
DEPT. OF MATH SCIENCES
JOHNS HOPKINS UNIV.
BALTIMORE, MD 21218

JOHN LABRECQUE
NATL. JEWISH CTR.
1400 JACKSON ST.
DENVER, CO 80206

RICHARD B. LANE
DEPT. MATHEMATICAL SCIENCES
UNIV. OF MONTANA
MISSOULA, MT 59812

KINLEY LARNTZ
UNIV. OF MN
DEPT. OF APPLIED STAT.
ST. PAUL, MN 55108

BARBARA LAWTON
ROCKWELL INTERNATIONAL
P.O. BOX 464
GOLDEN, CO 80401

RUSSELL LENTH
UNIV. OF WYOMING
P.O. BOX 3332
LARAMIE, WY 82071

PETER LEWIS
NAVAL POSTGRADUATE SCHOOL
DEPT. OR, NAVAL PG SCHOOL
MONTEREY, CA 93943

HSIN-GEE LI
DEPT. OF STATISTICS
COLORADO STATE UNIV.
FORT COLLINS, CO 80523

Y. RICHARD LIN
COLORADO DIV. OF LOCAL GOVT.
1313 SHERMAN ST., RM 520
DENVER, CO 80203

CHARLES LIN
SAS INSTITUTE, INC.
P.O. BOX 8000
CARY, NC 27511

ROBERT LING
CLEMSON UNIVERSITY
DEPT. OF MATH. SCIENCES
CLEMSON, SC 29634

ZEKE LITTLE
UNIV. OF COLORADO
P.O. BOX 486
BOULDER, CO 80309

JANI LITTLE
UNIV. OF COLORADO
P.O. BOX 486
BOULDER, CO 80309

LORA LEE LOFFT
SOCIOLOGY DEPT.
UNIV. OF DENVER
DENVER, CO 80208-0209

ROBERT M. LUCAS
RESEARCH TRIANGLE INST.
P.O. BOX 12194
RESEARCH TRIANGLE PARK, NC 27709

DENNIS LUCKEY
UNIV. OF COLORADO HEALTH SCI
10482 OWENS CIR.
WESTMINSTER, CO 80020

FRANKLIN LUK
CORNELL UNIVERSITY
PHILLIPS HALL
ITHACA, NY 14853-5401

VESNA LUZAR
RESEARCHER
UNIV COMP CTR, ENGELSOVA
ZAGREB, YUGOSLAVIA, 41000

JOSEPH MACHAK
UNIV. OF MICHIGAN
STATISTICS & MAN. SCI.
ANN ARBOR, MI 48109

LYNDA MacKICHAN
TCI SOFTWARE RESEARCH, INC.
1190-B FOSTER RD.
LAS CRUCES, NM 88001

WILLIAM MAKUCH
GENERAL ELEC. R&D
P.O. BOX 8 KWC207A
SCHENECTADY, NY 12301

COLIN MALLOWS
AT&T BELL LAB.
600 MT. AVE.
MURRAY HILL, NJ 07974-2070

LISE MANCHESTER
UNIV OF BC-DEPT OF STATISTICS
2021 W. MALL
VANCOUVER, B.C.
CANADA V6S 1W5

JOE MANDARINO
COLORADO STATE UNIVERSITY
DEPT. OF STATISTICS
FORT COLLINS, CO 80523

DENNIS MAR
NAVAL POSTGRADUATE SCHOOL
W.R. CHURCH COMP CTR-0141
MONTEREY, CA 93943

GEORGE MARSAGLIA
FLORIDA STATE UNIV.
DEPT. OF STATISTICS
TALLAHASSEE, FL 32306

DONALD MARX
UNIV. OF AK
SCHOOL OF BUSINESS
ANCHORAGE, AK 99508

KIRK MATHEWS
FIELD COMMAND, DNA
FCDNA/FCTT
KIRTLAND AFB, NM 87115-5000

DONALD E. MCCLURE
BROWN UNIV.
APPLIED MATHMATICS, BOX F
PROVIDENCE, RI 02912

JOHN ALAN MCDONALD
UNIV. OF WASHINGTON
STAT., GN-22. VOFW
SEATTLE, WA 98195

JOHN D. MCKENZIE
BABSON COLLEGE
BABSON PARK
WELLESLEY, MA 02157

DAVID MELLITS
JOHNS HOPKINS UNIV.
600 N. WOLFE ST.
BALTIMORE, MD 21205

MARK MERTENS
COBRO
12125 WOODCREST EXEC. DR.
ST. LOUIS, MO 63141

MICHAEL MEYER
1210 W. DAYTON ST.-STAT.
MADISON, WI 53706

DOUGLAS MILLS
PRINCETON UNIV. COMPUTING CNTR
87 PROSPECT AVE.
PRINCETON, NJ 08544

KAZEM MIRKHANI
ELECTRONIC DATA SYSTEMS CORP.
P.O. BOX 7019
TROY, MI 48084

TOBY MITCHELL
OAK RIDGE NATL. LABORATORY
P.O. BOX Y BLDG. 9207A
OAK RIDGE, TN 37830

WILLIAM MONSON
COLORADO SCHOOL OF MINES
GOLDEN, CO 81501

ROGER MOORE
BONNEVILLE POWER ADMN.
P.O. BOX 3621
PORTLAND, OR 97208

CARL MORRIS
UNIVERSITY OF TEXAS
DEPT. OF MATHEMATICS
AUSTIN, TX 78712

TERESA MUCINO PORRAS
SEGUROS MONTERREY S.A.
PRESIDENTE MASARIK 8-5° PISO
COL. BOSQUES DE CHAPULTEPEC
11588 MEXICO, D.F.; MEXICO

JERI M. MULROW
NATL. BUREAU OF STANDARDS
325 BROADWAY MC 714
BOULDER, CO 80303

VIJAY NAIR
AT&T BELL LABS
2C262, AT&T BELL LABS
MURRAY HILL, NJ 07974

STEPHEN NASH
MATH. SCIENCES DEPT-JN HOP UNI
34TH & CHARLES STS.
BALTIMORE, MD 21218

JOHN NASH
UNIVERSITY OF OTTAWA
275 NICHOLAS
OTTAWA, ONTARIO CANADA, KIN6N3

SOMANATHAN NATARAJ
FORT HAYS STATE UNIV.
BUS. ADM.: FHSU
HAYS, KS 67601

WILLIAM NAZARET
AT&T BELL LABS.
CRAWFORD CORNER RD-2K-512
HOLMDEL, NJ 07733

J.A. NELDER
IMPERIAL COLLEGE
HUXLEY BLDG., QUEENS GATE
LONDON,SW7 2BZ, ENGLAND

LORENE NELSON
ROCKY MTN. MS CENTER
P.O. BOX B181 UCHSC
DENVER, CO 80206

WESLEY NEWTON
254 N. FIRST ST.
TOOELE, UT 84074

KATHERINE NG
SAS INSTITUTE
P.O. BOX 8000
CARY, NC 27511

WILLIAM NICHOLLS
U.S. CENSUS BUREAU
DEPT. OF COMMERCE
WASHINGTON, DC 20233

W.L. NICHOLSON
BATTELLE-NORTHWEST
P.O. BOX 999
RICHLAND, WA 99352

SVEIN NORDBOTTEN
UNIV. OF BERGEN
N-5000 BERGEN
NORWAY,

ALVIN B. NOWVERL
USDA-HNIS-NMD-SSB
6505 BELCREST RD.
HYATTSVILLE, MD 20782

WILLIAM NUGENT
HARVARD UNIVERSITY
1 OXFORD ST., RM. 609
CAMBRIDGE, MA 02138

R.L. OBENCHAIN
BELL COMMUNICATIONS RESEARCH
NVC3J213
RED BANK, NJ 07701

R. WAYNE OLDFORD
MIT
E40-139, MIT, 1 AMHERST
CAMBRIDGE, MA 02139

GEORGE OSTROUCHOV
OAK RIDGE NATL. LABORATORY
PO.O. BOX Y BLDG.9207A
OAK RIDGE, TN 37831

FINBARR O`SULLIVAN
UNIV. OF CALIFORNIA-BERKELEY
STATISTICS DEPT.
BERKELEY, CA 94720

NELSON PACHECO
USAF ACADEMY
COLORADO SPRINGS, CO 80840

LARRY PALMITER
ECOTOPE, INC.
2812 E. MADISON
SEATTLE, WA 98112

DAVID PATTERSON
UNIV. OF MONTANA
DEPT. OF MATH. SCIENCES
MISSOULA, MT 59812

JON K. PECK
SPSS, INC.
444 N. MICHIGAN AVE.
CHICAGO, IL 60611

ROGER PFAFFENBERGER
TEXAS CHRISTIAN UNIV.
P.O. BOX 32868
FORT WORTH, TX 76129

HAIGANOUSH PREISLER
USDA FOREST SERVICE
1960 ADDISON ST.
BERKELEY, CA 94701

RONALD REGAL
UNIV. OF MN, DULUTH
DEPT. OF MATH SCIENCES
DULUTY, MN 55812

GARY RICHARDSON
COLORADO STATE UNIV.-ARS
STATISTICS DEPT.
FORT COLLINS, CO 80523

BRAD RITTER
HEWLETT PACKARD
3404 E. HARMONY RD.
FORT COLLINS, CO 80525

DAVID ROBINSON
ST. CLOUD ST. UNIV.
DEPT. OF MATH & STAT.
ST. CLOUD, MN 56301

WILLIAM ROGERS
2677 CENTINELA AVE. #103
SANTA MONICA, CA 90405

LARRY ROSENBERGER
THE FAIR, ISAAC COMPANIES
120 N. REDWOOD DR.
SAN RAFAEL, CA 94903-1996

LAURA RYAN
BLUE CROSS OF CALIFORNIA
21555 OXNARD ST.
WOODLAND HILLS, CA 91367

THOMAS A. RYAN
MINITAB, INC.
3081 ENTERPRISE DR.
STATE COLLEGE, PA 16801

BARBARA F. RYAN
MINITAB, INC.
3081 ENTERPRISE DRIVE
STATE COLLEGE, PA 16801

KAY RYSCHON
CREIGHTON UNIV. CARDIOLOGY
28TH & BURT
OMAHA, NE 68178

RICHARD S. SACHER
UNIV. OF DELAWARE
ACADEMIC COMPUTING SVCS.
NEWARK, DE 19716

WILLIAM M. SALLAS
IMSL, INC.
2500 CITY W. BLVD.
HOUSTON, TX 77042

ALEXANDER SAMAROV
UNIV. OF LOWELL
1 AMHERST ST. MIT,E40-145
CAMBRIDGE, MA 02139

STEPHEN SAMUELS
DEPT. OF STATISTICS
PURDUE UNIV.
W. LAFAYETTE, IN 47907

JOHN SAWYER JR.
DEPT. OF PREV. MEDICINE
5113 THOMPSON HALL
LUBBOCK, TX 79430

RICHARD SCHERER
HEWLETT-PACKARD
3404 E. HARMONY
FORT COLLINS, CO 80525

MARK J. SCHERVISH
CARNEGIE-MELLON UNIV.
DEPT. OF STATISTICS
PITTSBURGH, PA 15213

JAMES SCHMIDHAMMER
UNIV. OF TENNESSEE
DEPT. OF STATISTICS
KNOXVILLE, TN 37996

STEPHEN SCHMIDT
USAF ACADEMY
COLORADO SPRINGS, CO 80840

ROBERT SCHNABEL
DEPT. OF COMPUTER SCIENCE
UNIV. OF COLO.-BOX 430
BOULDER, CO 80309

DAN SCHNELL
IA STATE UNIV.
204 SNEDECOR HALL
AMES, IA 50011

LEE SCHRUBEN
CORNELL UNIVERSITY
342 UPSON HALL
ITHACA, NY 14853

DEL SCOTT
BYU
244 TMCB
PROVO, UT 84651

DAVID W. SCOTT
STANFORD UNIV.
DEPT. STATISTICS, SEQUOIA
STANFORD, CA 94305

WOODROW SETZER
DEPT. OF BIOSTATISTICS, UNC
SCH OF PUBLIC HEALTH-UNC
CHAPEL HILL, NC 27514

BRIAN SHEA
NVMERICAL ALGORITHMS GRP, LTD
NAG CEN OF,256 BANBURY RD
OXFORD, ENGLAND, OX2 70E

KEVIN SHEEHAN
ADOLPH COORS CO
NH520
GOLDEN, CO 80401

DAN SHERWOOD
HEWLETT PACKARD
3404 E. HARMONY RD.
FORT COLLINS, CO 80525

WEI-KEI SHIUE
SO. IL. UNIV. AT EDWARDSVILLE
DEPT. OF MATH/STAT SIUE
EDWARDSVILLE, IL 62026

WILLIAM SHULBY
WILAND SERVICES, INC.
1426 PEARL ST. #400
BOULDER, CO 80302

RICHARD SIMON
NATL. CANCER INST., CTEP, BRB
LANDOW BLDG., RM 4B06
BETHESDA, MD 20892

BURT SIMON
AT&T INFORMATION SYSTEMS LAB.
11900 N. PECOS ST.
DENVER, CO 80234

KENNETH SIMONS
IBM
1900 DIAGONAL HWY.
BOULDER, CO 80302

JIM SKULSTAD
CONNEERING
350 OAKDALE, SUITE 308
CHICAGO, IL 60657

STEPHEN SMEACH
G.D. SEARLE & CO.
4901 SEARLE PKWY.
SKOKIE, IL 60077

JOHN SMITH
UNIV. OF COLORADO
P.O. BOX 108
BOULDER, CO 80309

TOM SMITH
LONGMONT FOODS
P.O. BOX 1479
LONGMONT, CO 80502

P.L. SMITH
SHELL DEVELOPMENT CO.
P.O. BOX 1380
HOUSTON, TX 77251-1380

ROBERT SNELL
EASTMAN KODAK CO., M.S.P.
1669 LAKE AVE.
ROCHESTER, NY 14650

BRAD SNOW
ADOLPH COORS CO
NOB520
GOLDEN, CO 80401

STEVEN SOERENS
COLORADO SCHOOL OF MINES
MATH DEPT.
GOLDEN, CO 80401

W. ALLEN SPIVEY
UNIV. OF MI
GRAD. SCHOOL OF BUSINESS
ANN ARBOR, MI 48109

RANDALL SPOERI
AMERICAN STATISTICAL ASSN.
806 15TH ST., N.W., #640
WASHINGTON, DC 20005

JAYA SRIVASTAVA
COLORADO STATE UNIVERSITY
STAT. DEPT.
FORT COLLINS, CO 80523

WILLIAM STANISH
SAS INSTITUTE
P.O. BOX 8000
CARY, NC 27511

FREDERIC STERBENZ
UNIV. OF WYOMING
P.O. BOX 3985
LARAMIE, WY 82071

PETE STEWART
UNIV. OF MARYLAND
DEPT. OF COMPUTER SCIENCE
COLLEGE PARK, MD 20742

WALTER STUDDIFORD
PRINCETON UNIVERSITY
REGISTRAR B10A W. COLLEGE
PRINCETON, NJ 08544

GARY SULLIVAN
IOWA STATE UNIVERSITY
204 SNEDECOR HALL
AMES, IA 50011

TIM SWARTZ
UNIV. OF TORONTO
DEPT OF STATISTICS
TORONTO, ONT, CANADA, M5S 1A1

GARY L. TABBERT
3M/3M CTR.
3M - STAT. CONSULTING
ST. PAUL, MN 55144

CHARLES H. TAYLOR
COLORADO STATE UNIV
DEPT. OF STATISTICS
FORT COLLINS, CO 80523

ROBERT TEITEL
TEITEL DATA SYSTEMS
7315 WI AVE STE 727 E
BETHESDA, MD 20814

NICK K.W. TEOH
THE UPJOHN CO.
UNIT 9161-243-67
KALAMAZOO, MI 49001

RONALD A. THISTED
THE UNIV. OF CHICAGO
5734 UNIV. AVE.
CHICAGO, IL 60637

STEVE THOMAS
CONNEERING
350 OAKDALE, STE 308
CHICAGO, IL 60657

ROB TIBSHIRANI
DEPT OF PREV MED & BIOSTATISTS
UNIV. OF TORONTO
ONTARIO, CANADA, M5S 1A1

LUKE TIERNEY
UNIV. OF MN
SCHOOL OF STATISTICS
MINNEAPOLIS, MN 55455

NAITEE TING
COLORADO STATE UNIV.
DEPT. OF STATISTICS
FORT COLLINS, CO 80523

ROBERT TORTORA
USDA-SRS-RM. 4147
WASHINGTON, DC 20250

DERRICK TRACY
UNIV. OF WINDSOR
DEPT. OF MATH & STATISTIC
WINDSOR ONTARIO CANADA, N9B 3P4

JOSEPH TRIGG
UNIV. OF MI-FLINT
COMPUTER SCIENCE DEPT.
FLINT, MI 48502-2186

JOHN TUKEY
PRINCETON UNIVERSITY
408 FINE HALL
PRINCETON, NJ 08544

PAUL TUKEY
BELL COMMUNICATIONS RESEARCH
435 SOUTH ST.
MORRISTOWN, NJ 07960

DANNY TURNER
BAYLOR UNIVERSITY
BAYLOR UNIV., MATH DEPT.
WACO, TX 76798

DOMINIC VECCHIA
NATL. BUREAU OF STANDARDS
MC 714, 325 BROADWAY
BOULDER, CO 80303

PAUL VELLEMAN
CORNELL UNIV.
358 IVES HALL
ITHACA, NY 14853

JOSE S. VICTORIA
STATISTICS DEPT.
CSU
FT. COLLINS, CO 80521

H. D. VINOD
FORDHAM UNIVERSITY
ECON. DEPT. FORDHAM UNIV.
BRONX, NY 10458

FREDERIC VOGEL
US DEPT. OF AGRICULTURE,SRS
14TH & INDEPENDENCE AVE.
WASHINGTON, DC 20250

GRACE WAHBA
UNIV. OF WISCONSIN-MADISON
1210 W. DAYTON ST.
MADISON, WI 53706

JACK C. MING WANG
GEN MOTORS RSCH LABS, CS DEPT
WARREN, MI 48090

ANTONIA WANG
COLORADO STATE UNIV.
111 OLD ECON BLDG.
FORT COLLINS, CO 80523

BARBARA WARSAVAGE
ROLM
4678 ALPINE MEADOW LN.
COLORADO SPRINGS, CO 80919

EDWARD J. WEGMAN
GEORGE MASON UNIV.
217 THOMPSON HALL
FAIRFAX, VA 22030

NORRIS WEIMER
UNIVERSITY OF ALBERTA
EDMONTON
ALBERTA CANADA, T6G2H1

GUNTER WEISS
UNIV. OF WINNIPEG-DEPT OF STAT
515 PORTAGE AVE.
WINNIPEG MANITOBA CANADA, R3B2E9

DR. WILLIAM J. WELCH
UNIV. OF BRITISH COLUMBIA
FAC. OF COMMERCE, UBC
VANCOUVER CANADA, V6T1Y8

DR. WILLIAM J. WELCH
UNIV. OF BRITISH COLUMBIA
FAC. OF COMMERCE, UBC
VANCOUVER CANADA, V6T1Y8

ROY WELSCH
M.I.T.
50 MEMORIAL DR.
CAMBRIDGE, MA 02139

JAMES WENDELBERGER
MATH DEPT RES LABS GM TECH CTR
GEN MOTORS RESEARCH LABS
WARREN, MI 48090-9057

THOMAS WHITMORE
LANGUAGE OF DATA PROJECT
P.O. BOX 1169
BERKELEY, CA 94701-1169

DAVID A. WHITNEY
TASC
ONE JACOB WAY
READING, MA 01867

ROBERT WIGTON
UNIV. OF NEBRASKA MEDL. CNTR.
42ND & DEWEY
OMAHA, NE 68105

ARI WIJETUNGA
MOORHEAD STATE UNIV.
DEPT. OF MATHS, MSU
MOORHEAD, MN 56560

RICHARD WILDMAN
USAF ACADEMY
COLORADO SPRINGS, CO 80840

LELAND WILKINSON
SYSTAT, INC.
2902 CENTRAL ST.
EVANSTON, IL 60201

ALLAN WILKS
AT&T BELL LAB.
600 MT. AVE.
MURRAY HILL, NJ 07974

JAMES S. WILLIAMS
COLORADO STATE UNIVERSITY
STAT. DEPT. EN 102
FORT COLLINS, CO 80523

STAN WILLIE
AT&T INFORMATION SYSTEMS LAB.
11900 N. PECOS ST.
DENVER, CO 80234

SUSAN A. WINTER
COLORADO STATE UNIVERSITY
DEPT. OF FOREST & WOOD
FT. COLLINS, CO 80523

KNUT M. WITTKOWSKI
WESTBAHUHOFSTE 55
TUEBIUGEN, I.R.G., D-7900

TERRY WOODFIELD
AZ STATE UNIV.
DEPT. OF MATHEMATICS
TEMPE, AZ 85287

BRIAN WOODRUFF
AFOSR
BOLLING AFB
WASHINGTON, DC 20332

BRIAN YANDELL
UNIVERSITY OF WISCONSIN
1210 W. DAYTON ST.
MADISON, WI 53715

YI-CHING YAO
COLORADO STATE UNIV.
104 OLD ECON BLDG.
FORT COLLINS, CO 80523

PAUL ZADOR
INS. INSTITUTE FOR HWY. SAFETY
WATERGATE 600
WASHINGTON, DC 20037

JAMES R. ZUMBRUNNEN
COLORADO STATE UNIVERSITY
STAT-204 OLD ECON BLDG
FORT COLLINS, CO 80523

APPENDIX B. INDEX OF AUTHORS

END
DATE
FILMED
5-88
DTIC