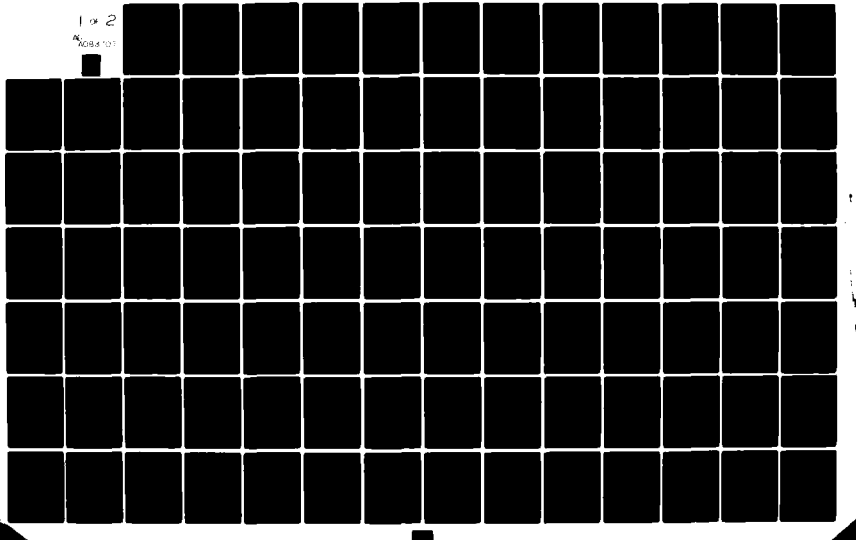
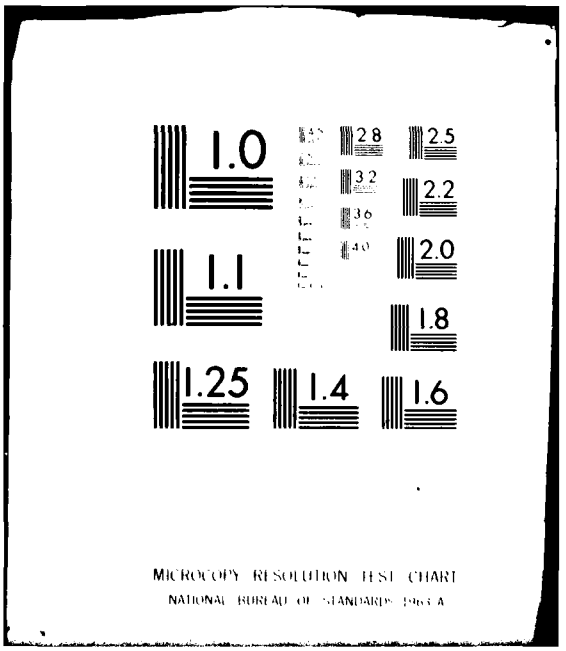
AD-A083 707 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOO--ETC F/6 9/2
LPAFIT, AN INTERACTIVE LINEAR PROGRAMMING PACKAGE DEVELOPED AT --ETC(U)
DEC 79 M A SCHIEFER
UNCLASSIFIED AFIT/GOR/SM/79D-7 NL

1 of 2
AD A083 707




1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963 A

(9) Master's thesis,

(6) LPAFIT,
AN INTERACTIVE LINEAR
PROGRAMMING PACKAGE
DEVELOPED AT
THE AIR FORCE INSTITUTE OF TECHNOLOGY.

THESIS

(14) AFIT/GOR/SM/79D-7

Michael A. Schiefer
Captain USAF

(10)

Approved for public release; distribution unlimited.

012225

JOB

AFIT/GOR/SM/79D-7

LPAFIT

INTERACTIVE LINEAR PROGRAMMING PACKAGE

DEVELOPED AT

THE AIR FORCE INSTITUTE OF TECHNOLOGY

THESIS

Presented to the Faculty of the School of Engineering

of the Air Force Institute of Technology

Air University

In Partial Fulfillment of the

Requirements for the Degree of

Master of Science

by

Michael A. Schiefer

Captain USAF

Graduate Operations Research

December 1979

Preface

The primary purpose of this paper is to document the development of an interactive linear programming package; however, this thesis also provides a variety of spinoff information. It introduces a broad spectrum of advanced programming techniques to those who write computer codes. To those who construct user-oriented computer programs, it offers ideas on creating packages which are easy for others to use. Finally, to managers, this thesis serves as a guage by which to measure other development efforts.

To those who are in need of an interactive, user-oriented, linear programming code, this thesis offers a package which can be readily implemented on a Control Data Corporation 6600. For those with other computer systems, this thesis offers the same package which is documented and structured so that it can be modified to meet individual needs.

Prior to the completion of this thesis, there was no user-oriented linear programming routine at the Air Force Institute of Technology School of Engineering. This thesis effort has filled that void. All students and faculty members can now solve linear programming problems without knowing anything about the computer. The program was designed to provide a linear programming code which made computer operations transparent to the user.

I envision that students will be exposed to LPAFIT in their introductory operations research courses and that they will apply the program to problems which they encounter in more advanced classes. This package will be utilized because it is user-oriented.

I wish to thank my advisor Colonel Charles R. Margenthaler and my readers Lieutenant Colonel Edward J. Dunne, Jr. and Captain R. R. Black for their meticulous review of this text. I also thank my classmates Tilford W. Harp and Thomas L. Wade for their typing and proofreading assistance at the eleventh hour. Finally, I thank my wife who has run the household for the last twenty-two weeks.

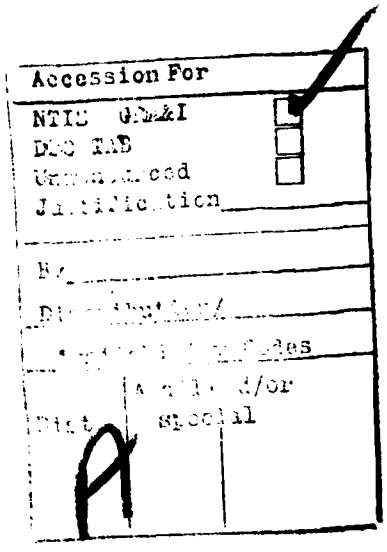


TABLE OF CONTENTS

ABSTRACT

This thesis effort produced a linear programming package, called LPAFIT. Two Fortran programs were written. The first, LPSOLVE, solves linear programming problems using the simplex method. It allows a user to do interactive sensitivity analyses. The second program, LPFRONT, is a preprocessor for LPSOLVE. LPFRONT assists the user in creating the input for LPSOLVE. A procedure controls all file manipulations. The only thing the user must be able to do is to express a problem in terms of a constrained objective function. Both programs are currently formatted to process up to 99 decision variables and 99 constraints. The routines will not solve integer or mixed integer problems. The package runs on a ~~Control Data Corporation~~ CDC 6600.

LPAFIT

AN INTERACTIVE LINEAR PROGRAMMING PACKAGE

DEVELOPED AT

THE AIR FORCE INSTITUTE OF TECHNOLOGY

I. INTRODUCTION

A universal need of graduate schools offering an operations research curriculum is the availability of systems of computer programs that support classroom instruction. Such programs permit the beginning student to focus attention on concepts rather than on the mechanics of problem solving. Computer-based programs allow the advanced student to solve problems more complex than those encountered in introductory courses. In addition, user-oriented computer programs permit the student not majoring in operations research to solve problems without having detailed knowledge of the computer algorithms.

THE LOCAL CAPABILITY

A fundamental member of any operations research package is a linear programming (LP) routine. The Air Force Institute of Technology (AFIT) uses a linear programming code called LPKODE in the graduate operations research curriculum. This widely used computer program is inadequate because of limited user documentation, difficult data entry requirements, inconsistent output, and execution only in a batch mode.

A previous effort to make the input to LPKODE less demanding was undertaken by Robert M. Schumacher (Ref 7), AFIT Operations Research graduate in 1978. He wrote a preprocessor for LPKODE called GORLPP. The

objective of this effort was to build an interactive program to generate the input to LPKODE based upon responses to a set of questions the computer program asked the user concerning a linear programming problem. GORLPP was never completely implemented because it was not documented; it did not work for some types of problems; and it required the user to understand the CDC 6600 file manipulation system. In spite of these problems, GORLPP did provide a useful foundation for part of this thesis effort.

Aside from LPKODE, there are no other operations research programs used by AFIT students to complement their studies.

STATEMENT OF RESEARCH OBJECTIVE

The shortcomings of the local capability and the need to have computer programs to supplement the classroom instruction for AFIT operations research students led to the establishment of this thesis. The original objective was to produce the nucleus for a library of operations research computer routines. The first milestone was to produce an accurate, easy to use linear programming package. Secondly, other packages, such as an integer programming model, were to be written. However, the magnitude of the effort required to build the linear programming routine was quickly realized and the objective of this thesis effort was reduced to the goal of developing an interactive, user-oriented,linear programming model fully documented with a stand-alone user's manual and a program documentation manual.

CRITERIA FOR PACKAGE DEVELOPMENT

The research objective implied several specific criteria for the development of the computer program.

Ease of Use. The requirement that the LP model be easy to use meant that the program had to be written for students who were not comfortable with the computer. This suggested that the model be designed to interactively assist the user in defining a problem. Such a capability would alleviate the difficulties of punching formatted data cards. Ease of use also meant that the computer job control system and file manipulation processes be transparent to the user. Finally, ease of use required that there be a precise set of instructions to prompt the infrequent user.

Ease of Modification. The requirement that the package be easy to use also dictated that the program be written so that others could readily modify it to satisfy their individual needs. Ease of modification meant that the code had to be well documented, both internally and externally. In addition, the package had to be structured in a manner which would facilitate change.

Political Considerations. The final constraint on the package development was political in nature. Since an implied objective was that the new program replace LPKODE, the LP model had to gain the acceptance of current LPKODE users who were accustomed to its particular inputs and outputs. Personal experience has shown that people will often continue to use a computer resource which they understand even if something better becomes available. To entice LPKODE users to try the new LP model, it was required to operate in the batch mode using the same cards as inputs that were being used for LPKODE. The new program was also required to produce the same type of output as LPKODE with the improvements of clarity and correctness. By processing the same input and generating similar output, the new model would offer increased capability to LPKODE

users and would require no change in their procedures.

FEATURES NOT REQUIRED

The replacement for LPKODE was not constrained to have features commonly found in other linear programming computer models. The new program was not required to be able to solve massive LP problems with thousands of constraints or decision variables. This exemption was granted because problems of this magnitude are not commonly presented to the student of linear programming Since large problems did not have to be solved, the new model was not constrained to execute as rapidly as possible. Wherever necessary, efficiency was to be sacrificed to gain effectiveness.

THESIS OVERVIEW

This document describes the development, within the criteria placed upon the effort, of a computer-based, interactive linear programming model called LPAFIT. Chapter II of this thesis discusses how the political constraints placed upon LPAFIT are satisfied. Chapter III documents how LPAFIT is molded to assure that the program is easy to use. Chapter IV reveals how LPAFIT is structured to facilitate modification. The final product, its capabilities, its limitations, and a sample problem are discussed in Chapter V. Finally, Chapter VI contains closing remarks and suggestions for future thesis work. Documentation for the LPAFIT model is contained in two appendices. Appendix A, LPAFIT Manual for the Infrequent User, describes in exact detail what a user who knows nothing about the computer must do in order to solve a problem on the machine. This manual is intended to be a user's guide for those whose primary interest is obtaining answers as

quickly and as easily as possible. Appendix B, LPAFIT Program Documentation for the Advanced User, explains the structure of LPAFIT, potential modifications to the package, and how these changes might be approached. Appendices A and B are suitable for publication as documents which stand independently of this thesis. Appendix C contains program listings and is included because it represents the bulk of the effort behind this thesis.

II. LPAFIT - A LINEAR PROGRAMMING MODEL - DEVELOPMENT AND CONSIDERATIONS

The development process considered the constraints listed in Chapter I. This chapter documents the techniques that assured LPAFIT's compliance with the development criteria. Included is a detailed account of some of the calculations the program performs.

A PREREQUISITE - DECODING LPKODE

Because of imposed constraints, LPKODE had to be deciphered in order to understand the calculations performed to generate output. Therefore, the development of LPAFIT began with a thorough examination of LPKODE. This investigation proved to be non-trivial.

Structure. The primary obstacle to the comprehension of LPKODE was its lack of internal structure. Structure is defined as identifiable portions of the program which perform specific functions. LPKODE is configured to accomplish linear programming calculations in a single, lengthy subroutine. This structure prevents easy determination of program flow.

Variable Names. Variable names impede the understanding of LPKODE. Fourteen letters of the alphabet are stand alone names, and many other names are non-descriptive. Programming techniques such as this make it difficult to decipher a computer program.

Documentation. The absence of documentation for LPKODE compounded the decoding problem. Internal documentation is limited. Only six percent of the cards in LPKODE are comments. The total external documentation for the program is a four page description of its input. There is no record of the calculation techniques the code uses or what the output of the model represents.

OPERATIONAL PROBLEMS DISCOVERED IN LPKODE

Trust in Computers. LPKODE employs the unsound practice of taking a course of action only when two calculated real numbers are equal. Such a practice places undue confidence in the numerical precision of computers. While two real numbers may be theoretically equal, they may not be identically represented in the computer due to rounding errors.

Inefficiency. LPKODE uses central memory inefficiently. All of its program arrays are of fixed dimension. The code makes no attempt to variably dimension arrays based upon the requirements of the problem being solved. Therefore, it contains many sparsely populated matrices.

Calculation Error. LPKODE incorrectly calculates shadow prices for "greater than or equal to" constraints. The program reports the dual of the artificial variable as the shadow price when it should report the dual of the surplus variable.

Summary. Examining and understanding LPKODE were two prerequisites to writing a linear program which would satisfy the imposed constraints. The foregoing problems made it difficult to understand LPKODE. Because of these impediments and deficiencies, the determination was made that such practices be avoided in the construction of LPAFIT.

LPKODE CALCULATION METHODOLOGY

This section documents the linear programming techniques used by LPKODE. Also discussed is LPKODE's impact on the structure of LPAFIT.

LPKODE Techniques. LPKODE is a FORTRAN program which performs calculations using the simplex linear programming method rather than a revised simplex scheme. These LP methods are explained in Hillier and Lieberman (Ref 5). LPKODE uses the "Big M" simplex method to deal with

equality and greater than constraints. The model contains the option of driving artificial variables out of the initial basis first. LPKODE produces solution information in one of five formats selected by the user. The program also prints shadow prices and sensitivity analyses on problem parameters.

LPKODE Techniques Applied to LPAFIT. Like LPKODE, LPAFIT uses the simplex method to solve linear programming problems. This approach was selected because it is a straightforward way to solve resource allocation problems. A revised simplex method could have been written, but it would have required extensive coding to produce required output. Thus,efficiency was sacrificed to gain effectiveness.

As in the case of LPKODE, LPAFIT prints shadow prices and sensitivity analyses. Historically, a point of confusion has been the interpretation of sensitivity output. To clarify this matter, the next section was included to discuss in detail the types of sensitivity analyses which both LPKODE and LPAFIT conduct.

SENSITIVITY ANALYSES

Notation. The following notation developed by Hillier and Lieberman for a final simplex tableau is used to discuss the sensitivity analyses which are common to LPKODE and LPAFIT:

ROW No.	Coefficient of X_1	X_2	X_k	X_n	X_{n+1}	X_{n+2}	X_{n+m}	Right Side
0	$Z_1^*-C_1$	$Z_2^*-C_2$	$\cdots Z_k^*-C_k$	$\cdots Z_n^*-C_n$	Y_1^*	Y_2^*	$\cdots Y_m^*$	Y_0^*
1	A_{11}^*	A_{12}^*	$\cdots A_{1k}^*$	$\cdots A_{1n}^*$	S_{11}^*	S_{12}^*	$\cdots S_{1m}^*$	B_1^*
.			.	.			.	
.			.	.			.	
.			.	.			.	
G	A_{g1}^*	A_{g2}^*	$\cdots A_{gk}^*$	$\cdots A_{gn}^*$	S_{g1}^*	S_{g2}^*	$\cdots S_{gm}^*$	B_g^*
.			.	.			.	
.			.	.			.	
.			.	.			.	
M	A_{m1}^*	A_{m2}^*	$\cdots A_{mk}^*$	$\cdots A_{mn}^*$	S_{m1}^*	S_{m2}^*	$\cdots S_{mm}^*$	B_m^*

where

A_{ij} is the original coefficient of decision variable j in constraint i.

B_i is the original right hand side for constraint i.

C_j is the original coefficient of decision variable j in the objective function.

S_{ij} is a slack variable coefficient.

$X_1 - X_n$ are decision variables.

$X_{n+1} - X_{n+m}$ are slack or artifical variables.

Y_i are shadow prices.

* indicates an entry in the final simplex tableau.

Sensitivity Equations. In terms of the notation just developed, Hillier and Lieberman also derive the following relationships used in sensitivity investigations:

$$\Delta B_k^* = \sum_{i=1}^{m} \Delta B_i S_{ki}^* \quad k=1,m \tag{1}$$

$$\Delta (Z_j^* - C_j) = -\Delta C_j + \sum_{i=1}^{m} \Delta A_{ij} Y_i^* \quad j=1,n \tag{2}$$

Sensitivity Categories. LPKODE and LPAFIT both conduct three types of sensitivity analysis. They study changes in the coefficients (C_i) of the basic decision variables in the objective function, changes in the coefficients (C_i) of the nonbasic decision variables in the objective function, and changes in the right hand sides (B_i). For each of these three cases, both programs assume that only one C_k or one B_k is allowed to vary and that all other C_i's and B_i's retain their initial values.

Changes in the Coefficient of a Basic Decision Variable in the Objective Function. The objective of the first type of analysis is to determine how large or small a C_k could have been before the composition of the final basis would change. The change being sought is in the members of the final basis rather than in the values of the basic variables. Only one C_k is allowed to change. Therefore, the only sensitivity relationship which applies to this situation is

$$\Delta (Z_k^* - C_k) = -\Delta C_k .$$

Since X_k is a basic variable, $(Z_k^* - C_k) = 0$. Now, suppose that C_k varies by ΔC_k. This implies that $(Z_k^* - C_k)$ will no longer be equal to 0. Thus, the final simplex tableau is no longer in correct form because basic variable X_k does not have a 0 entry in ROW 0. Suppose that X_k is the basic variable for ROW G. In order to put the final simplex tableau into proper form, ROW G must be multiplied by ΔC_k and added to ROW 0.

This implies that each entry in ROW 0 becomes

$$(Z_i^* - C_i) + \Delta C_k(A_{gi}) \quad i=1,n.$$

For any nonbasic variable X_i, $(Z_i^* - C_i)$ was greater than or equal to 0 in the final simplex tableau. If this ROW 0 entry becomes negative by adding it to $\Delta C_k(A_{gi})$, then X_i becomes an entering variable and the final basis will change. The automatic sensitivity feature calculates for each basic X_k how large ΔC_k can be before the final set of basic variables changes. An example will now be given to help clairify the technique. Suppose that the final simplex tableau looked like the following:

Basic Variable	Row No.	Coefficient of X_1	X_2	X_3	X_4	X_5	X_6	Right Side
Z	0	9	0	0	0	5	10	45
X_3	1	1	0	1	0	0	4	4
X_2	2	3	1	0	0	1	-2	9
X_4	3	-3	0	0	1	-1	-6	6

Example A.

Sensitivity analysis is now conducted on C_2, the coefficient of basic variable X_2. How large can ΔC_2 be before a ROW 0 entry for a nonbasic variable becomes negative? To answer this question, solve

$$(Z_i^* - C_i) + \Delta C_2(A_{2i}) = 0$$

for each nonbasic variable:

X_1: $9 + \Delta C_2(3) = 0 \rightarrow C_2 = -3$

X_5: $5 + \Delta C_2(1) = 0 \rightarrow C_2 = -5$

X_6: $10 + \Delta C_2(-2) = 0 \rightarrow C_2 = 5.$

Therefore for any value C_2' not in the interval $(-3+C_2 , C_2+5)$, the members of the final basis will change.

Changes in the Coefficient of a Nonbasic Decision Variable in the Objective Function. The objective of this analysis is to determine how much C_k must change before nonbasic variable X_k can enter the final basis. Only C_k is allowed to change. Therefore, the only sensitivity relationship which applies is

$$(Z_k^* - C_k) = -\Delta C_k.$$

Since $(Z_k^* - C_k)$ is greater than or equal to 0, C_k only need be large enough so that $(Z_k^* - C_k) + \Delta(Z_k^* - C_k)$ becomes negative. Example A will be used to illustrate the analysis. Consider C_1, the coefficient of nonbasic variable X_1. In order for $(Z_1^* - C_1)$ to become negative, ΔC_1 must be greater than 9. Therefore, for any value of C_1' not in the interval $(-\infty, C_1+9)$, X_1 will enter the final basis.

Changes in the Right Hand Sides. The objective of this analysis is to determine how large or small a right hand side (B_k) could be before the composition of the final basis would change. Only one B_k is allowed to change. Therefore, the only sensitivity relationship which applies is

$$\Delta B_k^* = \sum_{i=1}^{m} \Delta B_i S_{ki}^*$$

The final basis will change if any one of the B_i's becomes negative due to a change in B_k. Solving the following relationships for ΔB_k

$$\Delta B_1^* = \Delta B_k S_{k1}^* \rightarrow \Delta B_k = B_1^*/S_{k1}^* \quad \text{if } S_{k1}^* \neq 0$$

$$\Delta B_2^* = \Delta B_k S_{k2}^* \rightarrow \Delta B_k = B_2^*/S_{k2}^* \quad \text{if } S_{k2}^* \neq 0$$

$$\vdots \qquad\qquad \vdots$$

$$\Delta B_m^* = \Delta B_k S_{km}^* \rightarrow \Delta B_k = B_m^*/S_{km}^* \quad \text{if } S_{km}^* \neq 0$$

yields a set of $\Delta B_k'$s.

Let $B' = B_k$ + the least negative ΔB_k from the set.

Let $B'' = B_k$ + the least positive ΔB_k from the set.

Then, for any value of B_k not in the interval (B_k', B_k''), a decision variable will change in the final basis. Example A will be used to illustrate the procedure for sensitivity testing on B_3. Suppose that X_4, X_5, and X_6 are the slack variables. Solving for B_3 yields the following:

$$\Delta B_3 = 4/1 = 4$$

$$\Delta B_3 = 9/(-1) = -9$$

$$\Delta B_3 = 6/(-6) = -1$$

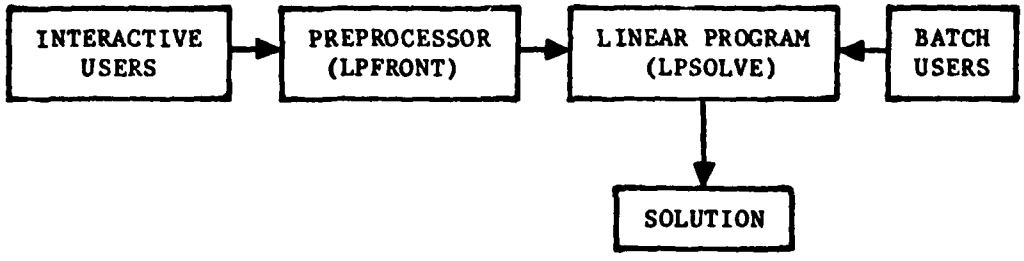
Therefore, for any B_3' outside the interval $(-1+B_3 , B_3+4)$, the members of the final basis will change.

SUMMARY

This chapter began by addressing the problems encountered trying to comprehend LPKODE. This understanding was required so that LPAFIT could be written to satisfy the constraints that it read the same input as LPKODE and produce similar output. To meet these criteria, LPAFIT was constructed to use the simplex and sensitivity analysis methods employed by LPKODE. However, to conform to other criteria, programming techniques not found in LPKODE had to be injected into LPAFIT. The next chapter discusses the methods employed in LPAFIT to produce an interactive model for the infrequent user.

III. LPAFIT - SYSTEM DEVELOPMENT AND CONSIDERATIONS

This chapter discusses the innovations required to make LPAFIT interactive and meaningful to the infrequent user. A variety of devices were employed to satisfy these criteria. One technique was the construction of an interactive preprocessor that makes the computer file manipulation system transparent to the user. Physically, LPAFIT is a package consisting of two FORTRAN programs, and it has the following schematic structure:



LPAFIT SYSTEM STRUCTURE

LPSOLVE is the code which performs linear programming calculations, and LPFRONT is an interactive preprocessor for LPSOLVE. System considerations required that separate programs be written, and user considerations dictated fusing the codes into a single coherent package.

LPFRONT - A PREPROCESSOR FOR LPSOLVE

A preprocessor called LPFRONT was written with the objective of making LPSOLVE easier to use. Before LPFRONT was written, input to LPSOLVE was only possible through punched cards. This was necessarily true because LPSOLVE was constrained to read the same cards used as inputs to LPKODE. The function of LPFRONT, a self-contained FORTRAN

program, is to interactively prompt the user with a series of questions about the problem to be solved. The preprocessor, based upon user responses, then creates properly formatted input for LPSOLVE. Thus, the need for manually punching cards is eliminated. This means that the user is no longer required to be familiar with FORTRAN data formatting or with operating a keypunch machine. In addition, the user need not invest the time required to carefully punch cards. Given this background on the function of the preprocessor, the next section discusses the development methodology followed for LPFRONT.

EVOLUTION OF LPFRONT

Although never implemented as a user program, a preprocessor for LPKODE called GORLPP was available from previous work. An examination of GORLPP indicated that it could be modified and used as a preprocessor for LPSOLVE. Since GORLPP was well-structured, changes to it were straightforward. Two categories of alterations were required: correction of errors and removal of inefficiencies.

Correction of Errors. Errors involving constraints, decision variables and output formats exist in GORLPP.

a. Computer code was written for LPFRONT to modify existing data for a problem by properly adding or deleting constraints or decision variables.

b. LPFRONT also needed a subroutine which would correctly calculate the number of decimal places for data output in a FORTRAN F6 write format. The details of the need for this information is contained in Appendix B, LPAFIT Program Documentation for the Advanced User. A subroutine was developed which performs the correct calculation for all cases. For example, it determines that a number such as 5.12345 can be written in an F6.4 format. Because of the negative sign, the subroutine recognizes

that -5.12345 must be expressed in an F6.3 format. The subroutine accounts for numbers which, when rounded off, would decrease the number of decimal places possible. For example, 9.99997 must be output as an F6.3 because the computer would round it to 10.000 in an F6 write format. Finally, the subroutine avoids the pitfalls associated with numbers such as 654321.78. This number cannot be expressed as an F6.0 because an F format includes the decimal point. The subroutine determines that such numbers be output in an I6 configuration.

Removal of Inefficiencies. In addition to correcting the errors in GORLPP, LPFRONT was written to be more efficient. The primary emphasis was on the use of disk storage space. LPFRONT permits the option of saving user inputs for reuse or modification at a later time. All of the data required to save a problem resides in three arrays whose total size is 10,000 decimal words. The number of these words actually being used varies with the size of the problem being solved. Rather than writing the entire three arrays to a file which could be saved, LPFRONT outputs only those portions of the arrays actually being used. This amounts to a significant savings when one considers that less than 50 words are needed to store the information for a problem with three constraints and two decision variables.

LPAFIT - AN INTERACTIVE MODEL

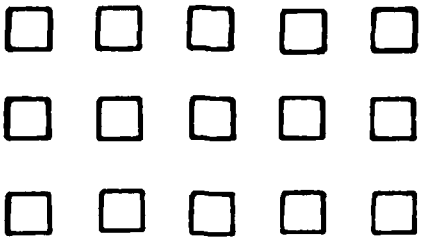
The objective in creating the preprocessor, LPFRONT, was to make LPSOLVE easier to use through an interactive, prompting computer program. The interactive concept is a characteristic which can make programs easy to use.

While LPFRONT and LPSOLVE were being developed, steps had to be taken to insure that they would execute interactively. The primary

obstacle to this objective was a local restriction on the amount of central memory available to an interactive program. Consequently, measures were taken in both LPFRONT and LPSOLVE to reduce the core they required.

LPFRONT - An Interactive Preprocessor. As LPFRONT evolved, it had to seek a narrower scope of objectives than those intended for GORLPP, a preprocessor written for at least five operations research support programs. GORLPP was too flexible for the needs of this effort, and computer code within the program not pertinent to linear programming problems was eliminated. There are two reasons that LPFRONT was restricted to being a preprocessor solely for LPSOLVE: first, this constraint would insure that LPFRONT would execute in the central memory allocated to an interactive program, and second, by not requiring the extra code needed to make LPFRONT a multiple preprocessor, the program would be able to process linear programming problems with a larger number of constraints or decision variables.

LPSOLVE - An Interactive Linear Program. To satisfy the objective of interactive execution, LPSOLVE, like LPFRONT, had to use only as much central memory as was allocated to time sharing programs. A variable dimensioning technique allowed this goal to be accomplished. To understand this approach, first consider the fixed dimension method used by LPKODE. LPKODE allocates a fixed amount of storage for each array used. For example, suppose LPKODE is dimensioned to solve problems with up to 5 decision variables and 3 constraints. The program reserves 15 memory locations to store the coefficients of the technology matrix in the following way:



If a problem arises with 6 decision variables and 2 constraints, only 12 storage locations are needed for the coefficients; however, LPKODE cannot solve the problem because it is structured to process only 5 decision variables.

LPSOLVE uses a different approach. It arranges the same 15 storage locations in the following way:

It uses these locations according to the parameters of the problem input by the user. For example, it can solve a problem with 2 decision variables and 7 constraints by partitioning the storage as follows:

LPSOLVE can also solve a problem with 7 decision variables and 2 constraints, or any other combination which requires less than 15 memory locations. If LPKODE wants the flexibility to solve problems with up to 7 constraints or 7 decision variables, it requires that 49 storage locations be reserved. LPSOLVE uses the variable dimensioning approach to subdivide one large array with 11,000 decimal words into 10 smaller arrays whose dimensions depend upon the particular problem being solved.

The Need for Two Programs. The linear program model and its preprocessor were written to execute interactively; however, some question may exist as to why LPFRONT and LPSOLVE were written as separate

programs. There are two disadvantages of combining the codes into a single, larger program. First, a single program would require the batch user to execute a code containing an unneeded preprocessor. This practice wastes central memory. Second, a single program would require more core for execution than either LPFRONT or LPSOLVE. This increased memory requirement might preclude the package from running interactively. For these reasons, the single program approach was rejected. However, a single program would have been easier for the user to manipulate than two separate codes. Therefore, steps were taken to give the user the illusion that a single routine was being used. The techniques used to accomplish this are discussed in the next section.

NO REQUIREMENT FOR USER KNOWLEDGE OF THE SYSTEM

While being interactive made LPAFIT easier to use, it did not remove all burdens from the user. In order for the linear programming package to be complete, it was also desirable that users not be required to have any knowledge of the CDC 6600 file manipulation and job control systems, that is, that computer system activities be transparent to the user. This desire was satisfied by writing LPAFIT to do all necessary file manipulation for the user. A procedure was written to control both the interface between LPFRONT and LPSOLVE and the communication of batch and interactive users with the two programs. In addition, any other optional file manipulation requested by the interactive user is executed from within LPFRONT. The manipulations performed internally in LPFRONT are discussed in the next subsection.

Internal File Manipulations. The preprocessor LPFRONT allows the user to create data related to a linear programming problem, to save this data for future use, and to modify the data at a later time. To exercise

these options, the CDC 6600 system requires that data files be attached, catalogued, and purged. To ask the interactive user to perform these tasks was viewed as unacceptable. The solution to this problem was a preprocessor that accomplished required file manipulations automatically for the user. When necessary, LPFRONT asks the user if a particular option is to be exercised. A "yes" or "no" answer triggers the appropriate file processing. This ease of use feature was created by writing two subroutines to drive Battelle Disk File Manipulation Computer Routines (Ref 1) available on the CDC 6600. These routines permit file manipulations to be done from within a FORTRAN program. While this capability simplified required file handling, it did not totally shield the user from interacting with the computer system. The next subsection discusses the procedure written to accomplish remaining file manipulations.

External File Manipulations. While the Battelle Disk File Routines accomplish optional file manipulations, they could not interface LPFRONT and LPSOLVE or control communications between users and the two programs. A procedure was written to satisfy these needs. To most users, the procedure creates the illusion that there is a single program solving the linear programming problem. The procedure has bundled two programs into the LPAFIT package. When a user begins the procedure, it attaches necessary files, loads and executes the compiled versions of LPFRONT and LPSOLVE at the appropriate times, and returns files no longer needed.

Battelle Disk File Manipulations Routines and a procedure which interfaces LPFRONT and LPSOLVE make the CDC 6600 job and file control systems transparent to the user of LPAFIT.

SIMPLE INSTRUCTIONS

Even though LPAFIT has been written to process files, the user is still required to perform some activities in order to use the package. To insure that LPAFIT be a viable tool for the infrequent user, a set of instructions which leads users through the solution process was required. Appendix A, LPAFIT Manual for the Infrequent User, contains such a set of directions. It explains, in basic terms, everything from how a user accesses the time sharing system to how to interpret the output. The following partial extract from the Table of Contents of Appendix A is included to indicate the level of knowledge which is required to use the manual:

WHAT LPAFIT CAN DO

WHAT LPAFIT CANNOT DO

WHAT THE USER MUST KNOW

WHAT THE USER DOES NOT HAVE TO KNOW

EXAMPLE PROBLEM IN STANDARD FORM

HOW TO RUN LPAFIT INTERACTIVELY (FROM A TERMINAL)

HOW TO RUN LPAFIT THROUGH BATCH (WITH CARDS)

HANDY INFORMATION

INTERPRETATION OF OUTPUT

Clearly, the manual does not require the user to be familiar with computers or computer jargon.

SUMMARY

This chapter has detailed the structure of LPAFIT and described the techniques designed to facilitate its use. Specific measures employed were the following: developing a preprocessor for LPSOLVE, creating an

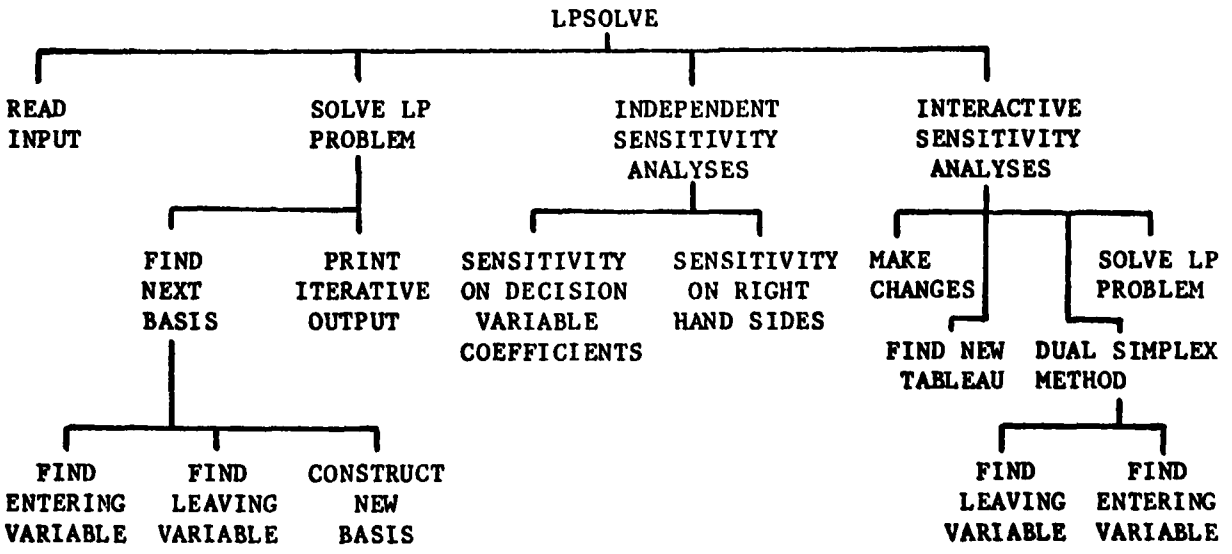
interactive package, freeing the user from knowledge of the system, and writing a simple user's manual.

IV. LPAFIT - SYSTEM STRUCTURE AND DOCUMENTATION

A characteristic common to computer-based systems is that they are often modified. Recognizing this fact, the initial structure of LPAFIT was designed to facilitate program changes. In addition to proper structure, another requirement for system modifications is proper documentation; this need was not compromised in the development of LPAFIT.

STRUCTURE

Structure is defined as identifiable portions of a program which perform specific functions. The existence of structure facilitates recognition of program flow and identification of where specific calculations are being performed. Both LPFRONT and LPSOLVE are highly structured. They have program drivers which control driver subroutines. These subroutines direct other subroutines that actually perform calculations. The following diagram illustrates the structure of LPSOLVE:



If LPSOLVE were to be modified, examination of elements within this structure allows ready identification of where any particular calculation is performed. Structure alone is a great aid to the modification process; however, any change is accomplished more easily when programs are well documented.

DOCUMENTATION

There are two places where LPFRONT and LPSOLVE are documented. They are documented internally through comment cards and externally in Appendix B, LPAFIT Program Documentation for the Advanced User.

External Documentation. The objective of external documentation is to give the user an overview of program structure. This explanation emphasizes why certain techniques were used, and it clarifies calculations not suited to internal documentation. The following is an extract from the Table of Contents of Appendix B:

BASIC PROGRAM STRUCTURE

PROGRAM CONTROLLING PROCEDURES

LINEAR PROGRAMMING PREPROCESSOR (LPFRONT)

- Local Files
- File Manipulations
- Core Requirements

LINEAR PROGRAM (LPSOLVE)

- Local Files
- File Manipulations
- Variable Dimensions

MODIFYING LPAFIT

This list indicates that the external documentation for LPAFIT describes the basic components of the model structure. It gives a macro view of file manipulations and other package functions. This external documentation points out both what LPAFIT does and what it cannot do without modification. Hence, the documentation external to LPAFIT gives a broad view of the package. Describing program calculations is left to internal documentation.

Internal Documentation. Both programs in the LPAFIT package are internally documented to enable a user to understand the function of each subroutine. The following set of comments is typical of the information provided for each subroutine in LPSOLVE:

```
C     ********************************************************************
C     *  SUBROUTINE ANSWER OUTPUTS THE FINAL VALUES OF ALL DECISION      *
C     *  VARIABLES, THEIR SHADOW PRICES, AND THE FINAL VALUE OF THE      *
C     *  OBJECTIVE FUNCTION. ANSWER IS CALLED BY POSTOP.                 *
C     *  LIST OF VARIABLES...COMMON BLOCK VARIABLES DEFINED IN INPUT2    *
C     *  NROWPI....POINTER TO THE PRICE FOR DECISION VARIABLE I          *
C     *  ROWJ......INTEGER. IF DECISION VARIABLE I IS IN THE BASIS,      *
C     *            IT IS IN ROW J.                                       *
C     *  TVALUE....CONTRIBUTION OF A DECISION VARIABLE TO THE VALUE      *
C     *            OF THE OBJECTIVE FUNCTION.                            *
C     ********************************************************************
```

Notice that the information at the front of each subroutine indicates from where the subroutine was called, and it defines variables local to the subroutine. Similar, but less extensive, comments are included in LPFRONT. Time did not permit a proper internal documentation effort in the preprocessor.

LPSOLVE, in addition to having informative headings, is heavily documented within the code. Forty-five percent of the two thousand cards in the program are comments. Such documentation permits users to readily understand what a particular part of the code is accomplishing.

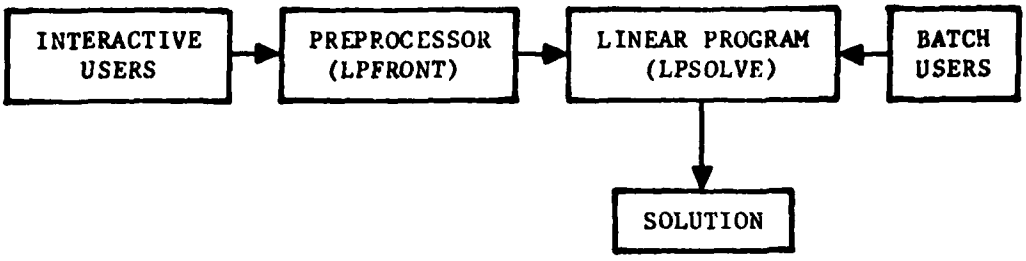
In addition to internal documentation in the form of comments, the readability of the source codes of LPFRONT and LPSOLVE was enhanced with meaningful names for program variables. The average length of names is about five letters, and names were selected to correspond to common linear programming terminology. A final measure to clarify both routines was using statement labels sequentially rather than in a random manner.

SUMMARY

This chapter has explained how structure and documentation insured that the LPAFIT model would be easy to modify. Structure enables ready recognition of program logic flows. External documentation in Appendix B presents a macro view of LPAFIT, and internal documentation identifies individual calculations. Ease of modification was the final development criterion to be satisfied. The next chapter summarizes the final product of this effort, the LPAFIT system.

V. RESEARCH RESULTS - THE LPAFIT SYSTEM

The LPAFIT system, its capabilities and limitations, are discussed in this chapter. The package is represented with the following schematic:



LPAFIT SYSTEM STRUCTURE

In the following section, the two key elements of this diagram, LPFRONT and LPSOLVE, are reviewed; an example problem is presented to demonstrate the package; and, finally, verification and validation procedures are discussed.

LPFRONT

LPFRONT evolved from a preprocessor written by Robert M. Schumacher. It is an interactive FORTRAN program consisting of about a thousand card images, functioning as a preprocessor for LPSOLVE. After posing a series of questions about a linear programming problem to the user, LPFRONT processes the reponses to create properly formatted input to LPSOLVE. Through Battelle Disk File Manipulation Routines, LPFRONT relieves the user of knowing the CDC 6600 file handling system. The program is well documented and highly structured to facilitate future modifications.

LPSOLVE

LPSOLVE is a FORTRAN program containing about two thousand card images. It uses the simplex linear programming method of problem solving. It may be executed by both interactive and batch users. To satisfy the resource constraints established for the LPAFIT system, the program reads the same input that the linear programming routine LPKODE uses, and it produces output which is similar to the output of LPKODE. LPSOLVE correctly calculates shadow prices; it does not allow program flow to depend upon two calculated numbers being equal; it prints output in scientific notation rather than in a FORTRAN F print format; and it prints headings and definitions of variables in an attempt to clarify the meaning of its output.

Interactive Sensitivity Analysis. LPSOLVE can be used to conduct interactive sensitivity investigation. It contains a subroutine which requests from the user the sensitivities to be investigated. A user has the option to simultaneously change coefficients in the objective function (C_j), coefficients in the constraints (A_{ij}), and right hand sides (B_i). The sensitivity relationships in equation (1) and (2), Chapter II, are used to revise the final simplex tableau. When this has been accomplished, the dual simplex and simplex methods are used to determine the new final tableau. LPSOLVE is not equipped to directly deal with sensitivity changes which simultaneously yield negative right hand sides and negative entries in the objective function, that is, infeasibility and non-optimality, respectively. When such situations occur, the program instructs the user to take an indirect route to perform the analysis. The basic approach is to leave LPSOLVE and use LPFRONT to modify the original problem into a problem with the set of

coefficients and right hand sides which caused LPSOLVE to fail. The modified problem is then sent to LPSOLVE and solved outright with the simplex method.

Limitations. In addition to not being able to directly deal with certain sensitivity situations, LPSOLVE, as it is written, cannot be used to solve problems with thousands of constraints or decision variables. Problems of this size should be attacked with more memory efficient revised simplex computer codes.

EXAMPLE PROBLEM

Now that the individual components of LPAFIT have been summarized, an example problem is given to illustrate how the package functions. The problem being solved is

Minimize $Z = 3X_1 + 5X_2$

subject to

$$\begin{aligned} X_1 &\leq 4 \\ 2X_2 &= 12 \\ 3X_1 + 2X_2 &\geq 18 \\ X_1, X_2 &\geq 0 \end{aligned}$$

The listing on the next page is an example of the dialogue which takes place between the computer and the user in order to solve this problem.

```
WELCOME. I AM L.P.AFIT. MY CREATOR WAS MIKE SCHIEFER,GOR79D.

USE THE FOLLOWING METHODS TO ANSWER QUESTIONS
IF THE QUESTION IS A YES/NO TYPE QUESTION, USE
   Y  FOR YES
   N  FOR NO
   Q  TO LEAVE THE PREPROCESSOR IMMEDIATELY AFTER STORING DATA
IF THE QUESTION WANTS A SINGLE INTEGER REPLY, USE
   AN INTEGER FOR THE DESIRED ANSWER
   -999 TO LEAVE THE PREPROCESSOR IMMEDIATELY AFTER STORING DATA

PERIODICALLY, GARBAGE WILL BE PRINTED(E.G. PF CYCLE NO.=999). JUST IGNORE IT.

ARE YOU GOING TO DEFINE A NEW PROBLEM? (Y,N)...Y
HOW MANY DECISION VARIABLES(EXCLUDE ARTIFICAL AND SLACK)? (INTEGER)...2
TOTAL NUMBER OF CONSTRAINTS? (INTEGER)...3
TITLE OF THIS PROBLEM IN 60 SPACES OR LESS?
>EXAMPLE PROBLEM

LPAFIT PRINT OPTION(OPTION=3 TO DISPLAY OPTIONS) OPTION?...3
OPTION  TO PRINT
------  --------
 -2     FIRST TABLEAU, LAST TABLEAU, EACH BASIS
 -1     TABLEAU FOR EACH ITERATION
  0     FIRST AND LAST TABLEAU ONLY
  1     EACH BASIS
  2     LAST BASIS ONLY

OPTION?...-1

DO YOU WANT TO DRIVE ARTIFICAL VARIABLES OUT OF THE BASIS FIRST? (Y,N)...Y

ITERATIVE OUTPUT ON UNIT(2 OR 6, -1 FOR MORE INFORMATION) UNIT...-1

IF YOU WANT THE INFORMATION GENERATED BY THE LPAFIT PRINT OPTION TO BE
PRINTED AT YOUR TERMINAL, UNIT=6. IF THE INFORMATION IS TOO EXTENSIVE
TO PRINT AT THE TERMINAL, UNIT=2 WILL CAUSE IT TO PRINT AT THE AFIT LINE
PRINTER.

ITERATIVE OUTPUT ON UNIT(2 OR 6, -1 FOR MORE INFORMATION) UNIT...6

EXTENSIVE HEADINGS?(Y,N, H FOR MORE INFORMATION)...H

THE OUTPUT FROM THE LINEAR PROGRAM CONTAINS DESCRIPTIVE HEADINGS WHICH ARE
FAIRLY EXTENSIVE. YOU SHOULD SEE THESE HEADINGS AT LEAST ONCE TO INSURE
PROPER UNDERSTANDING OF THE OUTPUT. REPLY Y TO SEE EXTENSIVE HEADINGS.,
IF YOU HAVE SEEN THESE HEADINGS, YOU MAY NOT WANT TO WAIT TO SEE THEM
PRINT. REPLY N FOR ABBREVIATED HEADINGS.
EXTENSIVE HEADINGS?(Y,N, H FOR MORE INFORMATION)...Y
```

ξ user responses are underlined

```
INTERACTIVE SENSITIVITY ANALYSIS?(Y,N, H FOR MORE INFORMATION)...H

AFTER LPAFIT SOLVES YOUR LP PROBLEM, YOU MAY INTERACTIVELY PERFORM
SENSITIVITY ANALYSES.
REPLY Y IF YOU WANT THIS OPTION
REPLY N IF YOU DO NOT WANT IT.

INTERACTIVE SENSITIVITY ANALYSIS?(Y,N, H FOR MORE INFORMATION)...Y

NUMBER OF GREATER THAN (GE) CONSTRAINTS? (INTEGER)...1

DO YOU WANT TO SUPPLY NAMES FOR CONSTRAINTS AND DECISION VARIABLES? (Y,N)...Y
OBJECTIVE NAME(UP TO 8 CHARACTERS)...COST

INPUT  VARIABLE NAMES(UP TO 8 CHARACTERS EACH)
X1: PLANT1
X2: PLANT2
INPUT  CONSTRAINT NAMES(UP TO 8 CHARACTERS EACH)
CONST 1: LAND
CONST 2: CASH
CONST 3: FLOOR

IS THIS A MAX OR A MIN PROBLEM? (MAX,MIN)...MIN

DO YOU KNOW HOW TO RESPOND TO PROGRAM REQUESTS FOR REAL NUMBER DATA?
(I.E. DO YOU KNOW HOW TO ENTER LIST DIRECTED DATA?) (Y,N)...N

EXPLANATION OF HOW TO ENTER REAL NUMBERS (LIST-DIRECTED INPUT)-----
YOU ARE GOING TO INPUT THE REAL VALUED COEFFICIENTS OF THE DECISION
VARIABLES IN THE OBJECTIVE FUNCTION AND IN THE CONSTRAINTS. EVERY COEFFICIENT
MUST BE ENTERED, EVEN IF IT IS 0.
   *COEFFICIENTS MUST BE SEPARATED BY EITHER A BLANK, A COMMA OR A SLASH
   *THE DECIMAL POINT CAN BE OMITTED AND IS ASSUMED TO BE THE RIGHT
    OF THE NUMBER ENTERED. E.G. 34.,6. MAY BE ENTERED AS 34,6
   *TO REPEAT A VALUE, AN INTEGER REPEAT CONSTANT IS FOLLOWED BY AN ASTERISK
    AND THE CONSTANT TO BE REPEATED(DO NOT EMBED BLANKS)
    E.G.  1.5,0,0,0,3.  MAY BE ENTERED AS 1.5,3*0,3
   *IF YOU HAVE TOO MUCH DATA FOR ONE LINE ON THE TERMINAL, HIT THE
    CARRIAGE RETURN AND CONTINUE ON THE NEXT LINE
   *IF AFTER YOU ENTER LIST DIRECTED DATA, THE TERMINAL DOES NOT RESPOND
    FAIRLY QUICKLY, RECOUNT THE # OF DATA POINTS YOU ENTERED. IF YOU SKIPPED
    A POINT, ENTER ENOUGH GARBAGE TO REACH THE REQUIRED POINT TOTAL. YOU
    WILL THEN BE GIVEN A CHANCE TO CHANGE YOUR ANSWER.

DO YOU KNOW HOW TO RESPOND TO PROGRAM REQUESTS FOR REAL NUMBER DATA?
(I.E. DO YOU KNOW HOW TO ENTER LIST DIRECTED DATA?) (Y,N)...Y
```

```
INPUT OBJECTIVE FUNCTION COEFFICIENTS. START WITH FIRST DEC. VAR.,END WITH LAST
3,5

MIN COST     WHICH IS EQUAL TO
 3.00     PLANT1     5.00     PLANT2

IS THIS EQUATION OK? (Y,N)...Y

INPUT COEFFICENTS FOR CONSTRAINT #  1 NOW 1,0
IS THIS AN EQUAL TO,GREATER THAN,OR LESS THAN CONSTRAINT? (EQ,GE,LE)...LE
INPUT RIGHT HAND SIDE NOW. B(1)=  4

INPUT COEFFICENTS FOR CONSTRAINT #  2 NOW
0,2
IS THIS AN EQUAL TO,GREATER THAN,OR LESS THAN CONSTRAINT? (EQ,GE,LE)...EQ
INPUT RIGHT HAND SIDE NOW. B(2)=  12

INPUT COEFFICENTS FOR CONSTRAINT #  3 NOW
3,2
IS THIS AN EQUAL TO,GREATER THAN,OR LESS THAN CONSTRAINT? (EQ,GE,LE)...GE
INPUT RIGHT HAND SIDE NOW. B(3)=  18


ECHO OF THE CURRENT PROBLEM BEING SET UP FOR SOLUTION

MIN COST     WHICH IS EQUAL TO
 3.00     PLANT1     5.00     PLANT2

SUBJECT TO THESE CONSTRAINTS

LAND

 1.00     PLANT1    0.        PLANT2
         LE       4.00

CASH

0.        PLANT1     2.00     PLANT2
         EQ       12.0

SPACE

 3.00     PLANT1     2.00     PLANT2
         GE       18.0

SELECT AN OPTION FOR CHANGES. YOU ARE ALLOWED MORE THAN ONE CHANGE
BUT THEY MUST BE MADE ONE AT A TIME

OPTION? (E.G. 1 FOR NO CHANGES, 3 TO REPEAT OPTION LIST) (INTEGER)...3
```

```
OPTION  ACTION                OPTION  ACTION
------  ------                ------  ------
   1    NO MORE CHANGES         10    CHANGE A VARIABLE NAME
   2    REECHO THE PROBLEM      11    CHANGE A CONSTRAINT NAME
   3    REPEAT OPTION LIST      12    CHANGE OBJECTIVE NAME
   4    ADD A VARIABLE          13    EXCHANGE MAX AND MIN
   5    DELETE A VARIABLE       14    CHANGE OBJECTIVE FUNCTION COEFFICIENTS
   6    ADD A CONSTRAINT        15    CHANGE THE (GE,LE,EQ)
   7    DELETE A CONSTRAINT     16    CHANGE RIGHT HAND SIDE VALUES
   8    RESCALE A VARIABLE      17    CHANGE CONSTRAINT COEFFICIENTS
   9    RESCALE A CONSTRAINT    18    QUIT NOW--DO NOT RUN LPAFIT--SAVE DATA
                                19    CHANGE OUTPUT OR SENSITIVITY OPTIONS

OPTION? (E.G. 1 FOR NO CHANGES, 3 TO REPEAT OPTION LIST) (INTEGER)...1

IF YOU HAVE JUST CREATED A DATA SET, DO YOU WANT TO
SAVE IT (PERMENANT FILE) FOR USE AT A LATER TIME? (Y,N)...Y
UNDER WHAT NAME DO YOU WANT THE DATA SET TO BE SAVED?
UP TO 40 ALPHA CHARACTERS...TEST
UNDER WHAT PASSWORD DO YOU WANT THE DATA SET SAVED?
UP TO 7 ALPHA CHARACTERS(ENTER * FOR NO PASSWORD)...PROB

REMEMBER YOUR DATA SET NAME AND PASSWORD. YOU MUST USE THE DATA SET
AT LEAST EVERY 7 DAYS OR IT WILL BE LOST.

DO YOU WANT TO PUNCH YOUR DATA FOR LATER BATCH INPUT? (Y,N)...Y
WHAT THREE LETTERS DO YOU WANT TO IDENTIFY YOUR PUNCHED DATA DECK?...MAS
YOUR DECK WILL BE PUNCHED AT AFIT WITH IDENTIFIER MAS

LPFAIT PREPROCESSOR SUCESSFULLY TERMINATED


EXAMPLE PROBLEM
BIGM = 1.000E+08


ITERATION NUMBER   0
        VARIABLE      1            2            3            4            5

RIGHT SIDE  BASIS
 3.000E+09    Z   -3.000E+08  -4.000E+08   0.           0.           0.
 4.000E+00    3    1.000E+00   0.          1.000E+00    0.           0.
 1.200E+01    4    0.          2.000E+00   0.           1.000E+00    0.
 1.800E+01    5    3.000E+00   2.000E+00   0.           0.           1.000E+00

TABLEAU CONTINUED PART    1
        VARIABLE      6

RIGHT SIDE  BASIS
 3.000E+09    Z    1.000E+08
 4.000E+00    3    0.
 1.200E+01    4    0.
 1.800E+01    5   -1.000E+00
```

ITERATION NUMBER 1

VARIABLE		1	2	3	4	5
RIGHT SIDE	BASIS					
6.000E+08	Z	-3.000E+08	0.	0.	2.000E+08	0.
4.000E+00	3	1.000E+00	0.	1.000E+00	0.	0.
6.000E+00	2	0.	1.000E+00	0.	5.000E-01	0.
6.000E+00	5	3.000E+00	0.	0.	-1.000E+00	1.000E+00

TABLEAU CONTINUED PART 1

VARIABLE		6
RIGHT SIDE	BASIS	
6.000E+08	Z	1.000E+08
4.000E+00	3	0.
6.000E+00	2	0.
6.000E+00	5	-1.000E+00

ITERATION NUMBER 2

VARIABLE		1	2	3	4	5
RIGHT SIDE	BASIS					
3.600E+01	Z	0.	0.	0.	1.000E+08	1.000E+08
2.000E+00	3	0.	0.	1.000E+00	3.333E-01	-3.333E-01
6.000E+00	2	0.	1.000E+00	0.	5.000E 01	0.
2.000E+00	1	1.000E+00	0.	0.	-3.333E-01	3.333E-01

TABLEAU CONTINUED PART 1

VARIABLE		6
RIGHT SIDE	BASIS	
3.600E+01	Z	1.000E+00
2.000E+00	3	3.333E-01
6.000E+00	2	0.
2.000E+00	1	-3.333E-01

```
**********************************************************
*TERMINAL SOLUTION AFTER    2 ITERATIONS                 *
*                                                        *
* DESCRIPTION OF HEADINGS...                             *
* CURRENT PRICE...THE OBJECTIVE FUNCTION COEFFICIENT     *
*                 OF THE DECISION VARIABLE               *
* ROW ZERO........FINAL TABLEAU ENTRY IN ROW 0.  MUST    *
*                 BE GREATER THAN OR EQUAL TO ZERO.      *
* QUANTITY........TERMINAL VALUE OF THE DECISION         *
*                 VARIABLE.  ONLY BASIC VARIABLES HAVE   *
*                 A NONZERO QUANTITY.                    *
* TOTAL PRICE.....(CURRENT PRICE)*QUANTITY=TOTAL PRICE   *
**********************************************************
```

DECISION VARIABLE	CURRENT PRICE	ROW ZERO	QUANTITY	TOTAL PRICE
1 PLANT1	3.000E+00	0.	2.000E+00	6.000E+00
2 PLANT2	5.000E+00	0.	6.000E+00	3.000E+01

TOTAL VALUE OF OBJECTIVE FUNCTION 3.60000E+01

```
**********************************************************************
* THIS OUTPUT BLOCK IDENTIFIES CONSTRAINT SHADOW PRICES AND THE      *
* AMOUNT OF SURPLUS IN THE CONSTRAINTS                               *
* DESCRIPTION OF HEADINGS...                                         *
* SURPLUS........AMOUNT OF RESOURCE NOT UTILIZED                     *
* VARIABLE.......SLACK OR ARTIFICAL VARIABLE WHICH WAS INTRODUCED    *
*                WITH THE CONSTRAINT                                 *
* SHADOW PRICE...PRICE ONE WOULD BE WILLING TO PAY FOR AN ADDITIONAL *
*                UNIT OF RESOURCE. FOR GE CONSTRAINTS, THE PRICE ONE *
*                WOULD PAY TO RELAX(REDUCE) THE CONSTRAINT BY A UNIT.*
*                EQ CONSTRAINTS MAY HAVE NEGATIVE SHADOW PRICES. IF  *
*                THIS OCCURS, INCREASING THE RESOURCE DRIVES THE     *
*                OBJECTIVE FUNCTION IN THE WRONG DIRECTION.          *
**********************************************************************
```

CONSTRAINT	CONSTRAINT SURPLUS	VARIABLE	SHADOW PRICE
1 LAND	2.000E+00	3	0.
2 CASH	0.	4	-1.500E+00
3 SPACE	0.	5	1.000E+00

DO YOU WANT AUTOMATIC SENSITIVITY ANALYSIS? (Y,N)...Y

```
**************************************************************************
* THIS OUTPUT BLOCK IS A SENSITIVITY ANALYSIS ON THE OBJECTIVE FUNCTION  *
* COEFFICIENT OF EACH DECISION VARIABLE.  THE ANALYSIS IS ONLY FOR CHANGES *
* IN THE OBJECTIVE FUNCTION COEFFICIENTS.  CONSTRAINT COEFFICIENTS ARE NOT *
* PERMITTED TO VARY FROM THEIR INITIAL VALUES.  THE ANALYSIS ON AN INDIVIDUAL *
* DECISION VARIABLE ASSUMES THAT ONLY ITS COEFFICIENT VARIES AND THAT ALL *
* OTHER OBJECTIVE FUNCTION COEFFICIENTS RETAIN THEIR INITIAL VALUES.     *
* DESCRIPTION OF HEADINGS...                                             *
* EV.....................VARIABLE WHICH ENTERS THE FINAL BASIS AS A RESULT OF *
*                        A PRICE CHANGE                                  *
* LV.....................VARIABLE WHICH LEAVES THE FINAL BASIS AS A RESULT OF *
*                        A PRICE CHANGE                                  *
* CURRENT PRICE.........THE INITIAL OBJECTIVE FUNCTION COEFFICIENT OF THE *
*                        DECISION VARIABLE                               *
* MINIMUM PRICE.........IF THE OBJECTIVE FUNCTION COEFFICIENT OF THE DECISION *
*                        VARIABLE FALLS BELOW THIS PRICE, THEN VARIABLE EV WOULD*
*                        ENTER THE FINAL BASIS AND VARIABLE LV WOULD LEAVE THE *
*                        FINAL BASIS.                                    *
* MAXIMUM PRICE.........IF THE OBJECTIVE FUNCTION COEFFICIENT OF THE DECISION *
*                        VARIABLE INCREASES ABOVE THIS PRICE, THEN VARIABLE EV *
*                        WOULD ENTER THE FINAL BASIS AND VARIABLE LV WOULD *
*                        LEAVE THE FINAL BASIS.                          *
* ENTERING QUANTITY....VALUE OF THE ENTERING VARIABLE(EV)                *
**************************************************************************
* THE FOLLOWING CONCLUSIONS MAY BE DRAWN AS THE RESULT OF CHANGING A SINGLE *
* OBJECTIVE FUNCTION PRICE COEFFICIENT                                   *
* 1.  IF THE MAGNITUDE OF THE CHANGE IS SUFFICIENT, THEN A VARIABLE MAY LEAVE *
*     THE FINAL BASIS.                                                   *
* 2.  IF A VARIABLE LEAVES THE FINAL BASIS, THE NEW BASIS WILL STILL BE  *
*     OPTIMAL.                                                           *
* 3.  WHETHER OR NOT A VARIABLE LEAVES THE FINAL BASIS, THE VALUES OF THE *
*     BASIC VARIABLES AND THE OBJECTIVE FUNCTION WILL CHANGE.            *
**************************************************************************
```

DECISION VARIABLE	CURRENT PRICE	MINIMUM PRICE	EV	LV	ENTERING QUANTITY	MAXIMUM PRICE	EV	LV	ENTERING QUANTITY
1 PLANT1	3.000E+00	0.	6	3	6.000E+00	3.000E+08	5	1	6.000E+00
2 PLANT2	5.000E+00					2.000E+08	4	3	6.000E+00

```
************************************************************
* THIS IS A SENSITIVITY ANALYSIS ON THE ORIGINAL RIGHT HAND*
* SIDE(RHS) VALUES. WHILE ONE RHS CHANGES, THE OTHERS KEEP *
* THEIR INITIAL VALUES.                                    *
* DESCRIPTION OF VARIABLES...                              *
*      IF THE VALUE OF AN ORIGINAL RHS FALLS BELOW THE     *
*      MINIMUM VALUE OR RISES ABOVE THE MAXIMUM VALUE,     *
*      THEN VARIABLE LV WOULD LEAVE THE FINAL BASIS.       *
************************************************************
```

CONSTRAINT	ORIGINAL VALUE	MINIMUM VALUE	LV	MAXIMUM VALUE	LV
LAND	4.000E+00	2.000E+00	3		
CASH	1.200E+01	6.000E+00	3	1.800E+01	1
SPACE	1.800E+01	1.200E+01	1	2.400E+01	3

```
************************************************************
* YOU MAY NOW CONDUCT AN INTERACTIVE SENSITIVITY ANALYSIS.*
* YOU MAY SIMULTANEOUSLY CHANGE OBJECTIVE FUNCTION         *
* COEFFICIENTS OF DECISION VARIABLES, CONSTRAINT           *
* COEFFICIENTS OF DECISION VARIABLES, AND RIGHT HAND       *
* SIDES.  YOU MAY CONSIDER AS MANY CASES AS YOU DESIRE.    *
* THE CHANGES FOR CASE 1 CARRY OVER TO CASE 2 AND SO ON.   *
* BASED ON YOUR CHANGES, THE NEW BASIS AND NEW VALUE OF    *
* THE OBJECTIVE FUNCTION WILL BE PRINTED. THE PROGRAM      *
* WILL PROMPT YOU FOR INFORMATION, GIVE YOU A CHOICE OF    *
* RESPONSES FOR ALPHA REPLYS, AND GIVE YOU NUMBER TYPE     *
* FOR NUMERICAL RESPONSES.  REMEMBER TO HIT THE RETURN     *
* BUTTON AFTER EACH RESPONSE.  LET'S GET STARTED.          *
************************************************************

DO YOU WISH TO MAKE ANY CHANGES? (Y,N)...Y
WHAT PRINT OPTION WOULD YOU LIKE? (INTEGER)...2
ANY CHANGES IN THE OBJECTIVE FUNCTION? (Y,N)...N
ANY CHANGES IN THE CONSTRAINTS? (Y,N)...Y
WHICH CONSTRAINT? (INTEGER)...3
CHANGE IN THE DECISION VARIABLE COEFFICIENT OR RIGHT HAND SIDE? (DV,RS)...RS
        OLD RHS=     18.00000           NEW RHS=? (F6)...17
ANY MORE CHANGES IN THE CONSTRAINTS? (Y,N)...N

CASE   1
ITERATION NUMBER   0
```

BASIC VARIABLES	CURRENT PRICE	QUANTITY	TOTAL PRICE
3	0.	2.333E+00	0.
2	5.000E+00	6.000E+00	3.000E+01
1	3.000E+00	1.667E+00	5.000E+00
	TOTAL COST IS $		3.500E+01

```
DO YOU WISH TO MAKE ANY MORE CHANGES? (Y,N)...N
      STOP
```

VERIFICATION

By definition, verification is the process of determining that a computer program is written as intended. In the case of LPAFIT, verification involved testing system options to insure that all data and all files were being properly manipulated. It also involved checking the satisfaction of program development criteria. Input decks for LPKODE were read by LPSOLVE to verify that the output produced by the two codes was similar. To insure that the package fulfilled the interactive concept, LPAFIT was exercised by several subjects who were not familiar with the CDC 6600 computer system. Using feedback from these subjects, the package was iteratively improved by clarifying those features of LPAFIT which created confusion.

VALIDATION

While verification insures that a program is written as intended, the validation process tests a programs's ability to produce correct answers. LPAFIT was validated by using the package to solve numerous linear programming problems with known solutions. Test problems had both negative and positive coefficients, all three types of equality relationships, and various numbers of decision variables and constraints. Problems with no solutions, degenerate solutions, multiple solutions, and unbounded solutions were tested. It must be noted that this validation procedure does not insure that LPSOLVE can successfully solve all linear programming problems; the tests only served to increase confidence in the code.

SUMMARY

This chapter has discussed the components in the LPAFIT package and how they were verified and validated. An example problem was presented to illustrate package use.

VI. CONCLUSIONS AND RECOMMENDATIONS

LPAFIT

The LPAFIT system provides the AFIT CDC 6600 with an interactive and batch mode capability for the solution of linear programming problems of interest to operations research students and faculty. In the interactive mode, the model does the following:

1. It helps the user describe a problem to the computer.
2. It solves the problem.
3. It gives the user the option of performing sensitivity analyses.
4. It controls data storage and retrieval.

By design, LPAFIT enables students with no computer background to use the machine as a tool.

Additional products of this research effort are an independent user's guide and an independent programmer's manual.

LPAFIT IMPROVEMENTS

As is the case with most large programs, there are potential improvements to LPAFIT. The code could be altered to increase its flexibility. Currently, LPAFIT has a fixed program length. This limits the maximum problem size that can be solved. Any problem too large for available space requires that the program source code be read and complied after appropriate arrays are redimensioned. Flexibility would be increased by allowing the batch user to exercise the package on a problem of any size. This feature should be obtainable by rewriting the procedure controlling the LPAFIT package. The desirability of making such a modification would have to weigh the need for increased

flexibility against the requirement of storing and recompiling source code.

FUTURE PROJECTS

LPAFIT is one of the first members of the AFIT analysis library. Anyone contemplating a thesis which adds to this library ought to consider several items. First, the proper program must be selected. An informal survey of the AFIT Systems Management faculty would reveal the needs of particular instructors. Based on personal experience, there is a current need for a user-oriented linear regression package. At the present time at least two linear regression programs exist at AFIT and both of them are inconvenient to use. One of the packages was written locally and runs on a Honeywell computer system. It has no user's manual and data input to it is format sensitive. The program has no interactive data creation feature, and it requires that the user know Honeywell job control or text editing. A thesis project would be to make this program interactive and to implement it on the CDC 6600. The other linear regression code is in the Statistical Package for the Social Sciences (SPSS) which is available on the CDC 6600. Students who use SPSS typically take a course which teaches them to manipulate the package. A partial thesis topic would be to write a manual on using the linear regression routine of SPSS and interpreting the output produced.

PERSONAL INSIGHT

An item a prospective program writer must consider is personal qualifications for such a project. One requirement is to know the computer language in which the program is to be written. Currently, it would be difficult to justify using any language other than FORTRAN.

Past experience in building large packages would also be helpful. Notice that both of these skills are not developed at AFIT. Therefore, they must have been acquired before coming to the School.

Another item to be considered is personal motivation for undertaking a project such as this. The driving motivation must be to produce a useable product. A well written user program should make the exercise look easy to the casual observer. Therefore, do not expect anyone to appreciate a good programming effort. Few people have ever written large computer codes, and an even fewer number recognize the difference between a good and a bad product. I have perceived an attitude which equates ease of task conceptualization with ease of task accomplishment. However, finishing this thesis was like walking from Fox, Arkansas to Timbo, Arkansas, a distance of about five miles as the crow flies. All you have to do is put one foot in front of the other and invest a little time. What the inexperienced "flat-lander" fails to realize is that you have to walk across a thousand acres of poison ivy, that the density of hungry ticks and chiggers is fifty per square foot, and that there are forty-seven snakes waiting to bite you in the leg. When you finally get to where you are going, nobody can understand what all the scratching is about.

Bibliography

1. Battelle Disk File Manipulation Routines User's Guide. Reprinted by ASD Computer Center, July 1978.

2. CDC NOS/BE User's Guide. Printed by ASD Computer Center, January 1978.

3. Cyber Control Language. Reprinted by ASD Computer Center, December 1977.

4. Fortran Extended Version 4 Reference Manual. Sunnyvale: Control Data Corporation, 1977.

5. Hillier, Frederick S. and Lieberman, Gerald J. Operations Research. San Francisco: Holden-Day, Inc., 1974.

6. Intercom Version 4 Reference Manual. St. Paul: Control Data Corporation, 1978.

7. Schumacher, Robert M. Survey and Extension of Computer Resources to Support Graduate Operations Research Education. AFIT/GOR/SM/78D-13.

VITA

Michael A. Schiefer was born on September 22,1950 in San Antonio, Texas. He graduated from high school in Alamogordo, New Mexico in 1969. He completed his Bachelor of Science in Mathematics at the Air Force Academy in 1973. His first tour of duty was at the Theoretical Branch , Technology Division of the Air Force Weapons Laboratory, Kirtland Air Force Base, New Mexico. While there, he studied nuclear blast effects on reentry vehicle trajectories. He was assigned to the Biometrics Division of the School of Aerospace Medicine, Brooks Air Force Base, Texas from 1975 to 1978. During this period, he concentrated on the human elements of weapons systems. He also became skilled in the use of numerically controlled milling machines. He entered the Air Force Institute of Technology in June,1978.

He is married to the former Patricia Lee Carpenter of Johnson City Tennessee. They have a son, Matthew.

Permanent Address: General Delivery
Fox, Arkansas
72051

APPENDIX A

LPAFIT

MANUAL FOR THE

INFREQUENT USER

DECEMBER 1979

LPAFIT - AFIT LINEAR PROGRAMMING PACKAGE

MANUAL FOR THE

INFREQUENT USER

Michael A. Schiefer

TABLE OF CONTENTS

INTRODUCTION

LPAFIT is an interactive linear programming package developed at the Air Force Institute of Technology as a thesis effort. The package is designed to make it very easy for a user to set up and solve a problem. The assumptions which this manual makes are that users of LPAFIT can formulate linear programming problems, that they can read, and that they can follow instructions. It is not necessary that they know anything about the computer in order to use this package.

WHAT LPAFIT CAN DO

LPAFIT solves linear programming problems. The package was designed to illustrate the simplex linear programming method to the student of operations research. LPAFIT runs interactively (from a terminal) and through batch (from cards). In the interactive mode, the package does the following:

1. It helps the user describe a problem to the computer.
2. It solves the problem.
3. It gives the user the option of doing sensitivity analyses.
4. It controls all data storage and retrieval.

LPAFIT was written to replace LPKODE, another linear programming routine in use at AFIT. Data decks which were used to run LPKODE can be run with LPAFIT by modifying one card. LPAFIT offers the users of LPKODE some additional options, and it produces understandable output.

WHAT LPAFIT CANNOT DO

LPAFIT does not do integer or mixed integer problems. It does not do transportation problems which require integer output. It does not allow negative decision variables or negative right hand sides. The

program was not designed to solve large linear programming problems (thousands of decision variables or constraints). There are more memory efficient revised simplex codes available to do this.

WHAT THE USER MUST KNOW

The user of LPAFIT must be able to formulate a linear programming problem in standard form. This is the standard form defined by Hillier and Liberman in Operations Research, Second Edition, p21. This manual contains an example of a problem in standard form. The user must scale all coefficients and right hand sides so that they lie in the range (-99999.499,-.0001) or (.00001,999999.499). All LPAFIT users must have a problem number and interactive users must have a password.

WHAT THE USER DOES NOT HAVE TO KNOW

LPAFIT is very easy to run; it is very smart and does most of the work for the user. The interactive user does not have to know anything about computer job control language, data formats, or keypunch operations. The batch user does not have to know anything about computer job control language.

MAXIMUM PROBLEM SIZE

In its current form, LPAFIT requires that all problems be within the following size limitations:

1. NVAR < 100

2. NCON < 100

3. (NVAR+NCON+NGT)*(3+NCON)+6*NCON+16 < 11000

   if the user supplies names for constraints and decision variables

4. (NVAR+NCON+NGT)*(3+NCON)+5*NCON < 11000

   if the user does not supply names

where

NVAR = Number of decision variables

NCON = Number of constraints

NGT = Number of greater than constraints.

EXAMPLE PROBLEM IN STANDARD FORM

The following example problem is in standard form. It is from Hillier and Lieberman, p65.

Minimize $Z = 3X_1 + 5X_2$

subject to

constraint 1: $X_1 \leq 4$

constraint 2: $2X_2 = 12$

constraint 3: $3X_1 + 2X_2 \geq 18$

$X_1, X_2 \geq 0$

Notice that right hand sides must be positive.

HOW TO RUN LPAFIT INTERACTIVELY (FROM A TERMINAL)

Use the following procedure to run LPAFIT interactively:

1. Arrange for an hour of time on a terminal. Pick a machine which prints on paper (hard copy capability). The first session will last 30-45 minutes. After the experience of one or two

sessions, most users will be able to set up a small problem in 5-10 minutes.

2. Get the computer's attention. This is done by calling the machine with the phone beside the terminal. Accomplish this in the following manner:

a. Pick up the receiver, depress the talk button on the phone, and dial one of the 300 BAUD numbers listed on the terminal.

b. If the computer is going to "talk" to you, it will respond with a high-pitched tone after one ring.

c. Depress the data button on the phone, hang up the receiver, and hit the RETURN key on the terminal.

3. After you dial-up the computer, the following dialogue will take place:

```
Computer: PLEASE LOGIN-
    User: LOGIN (hit the RETURN key after each user response)
Computer: ENTER PROBLEM NUMBER-
    User: (enter problem number, e.g. T790424)
Computer: ENTER PASSWORD-
    User: (enter password, e.g. CIA)
Computer: ENTER 3-DIGIT TERMINAL ID-
    User: (enter terminal id. e.g. 704)
Computer: COMMAND-
    User: ATTACH,PROCFIL,LPAFIT,ID=AFIT.
Computer: COMMAND-
    User: BEGIN,LP.
```

4. After the user responds BEGIN,LP., the hard part is over. The computer will ask a series of questions about the problem and other user desires. After these inquiries, the machine will solve the problem. The next section illustrates some of the questions the machine will ask.

5. After the problem is solved, the following dialogue will occur:

```
Computer: COMMAND-
    User: BEGIN,LP.(use this to solve another problem)
    User: LOGOUT   (use this to quit, remember to hit RETURN)
```

6. After LOGOUT, pick up your material and go have a cup of coffee.

QUESTIONS THE COMPUTER WILL ASK THE INTERACTIVE USER

The computer will assist the interactive user in defining a problem. The machine will first ask which of the following options the user wants:

* DEFINING A NEW PROBLEM
* RECOVERING FROM A MACHINE CRASH OR USER ABORT
* MODIFYING AN EXISTING PROBLEM
* SOLVING AN EXISTING PROBLEM WITHOUT ANY CHANGES

If the user is DEFINING A NEW PROBLEM, the machine will ask about all of the following items. If the user is RECOVERING FROM A MACHINE CRASH OR USER ABORT, the computer will ask about some of the following:

* TOTAL NUMBER OF DECISION VARIABLES
* TOTAL NUMBER OF CONSTRAINTS
* PROBLEM TITLE
* TYPE OF OUTPUT THE USER WANTS TO SEE. OPTIONS ARE:
    .FIRST TABLEAU,LAST TABLEAU,EACH BASIS
    .TABLEAU FOR EACH ITERATION
    .FIRST AND LAST TABLEAU
    .EACH BASIS
    .LAST BASIS ONLY
* WHERE USER WANTS TO PRINT SELECTED OUTPUT OPTION. OPTIONS ARE:
    .PRINT ON HIGH SPEED PRINTER
    .PRINT AT THE TERMINAL
* IF ARTIFICIAL VARIABLES ARE TO BE DRIVEN FROM BASIS FIRST
* IF THE USER WANTS EXTENSIVE DESCRIPTIVE HEADINGS
* IF THE USER WANTS INTERACTIVE SENSITIVITY ANALYSES
* THE NUMBER OF GREATER THAN CONSTRAINTS
* IF THE USER WANTS TO SUPPLY NAMES FOR CONSTRAINTS AND DEC. VARS.
* IF THE PROBLEM IS A MAXIMIZE OR MINIMIZE
* WHAT THE COEFFICIENTS, RESOURCES, AND EQUALITY RELATIONSHIPS ARE

After asking these questions, the computer will give the user the opportunity to modify any response. The machine will start at this

point if the user is MODIFYING AN EXISTING PROBLEM. After modifications are made, the machine will help the user permanently store the problem and/or punch it for batch input. The computer will then solve the linear programming problem. After the problem is solved, the user will be allowed to conduct interactive sensitivity analyses if that option has been selected.

HOW TO RUN LPAFIT THROUGH BATCH (WITH CARDS)

To run LPAFIT through batch input, the user must know more about the computer than if he were solving a problem interactively. Batch input may be advantageous when solving problems with large numbers of decision variables or constraints (in the range of 30 to 99). For problems of this size, an excessive amount of time may be spent printing output at a terminal. Batch has the advantage of high-speed printing. Use the following deck structure to run LPAFIT with cards:

```
JOB CARD                          (use default time and memory)
ATTACH,PROCFIL,LPAFIT,ID=AFIT.    (permanent file)
BEGIN,LPBATCH.                    (procedure)
7/8/9                             (multipunch)
TITLE CARD                        (data)
OPTION CARD                       (data)
COEFFICIENT CARDS                 (data)
RESOURCE CARDS                    (data)
NAME CARDS                        (data)
6/7/8/9                           (multipunch)
```

Input data cards are difficult to correctly generate manually. LPAFIT can be used in the interactive mode to automatically punch input data cards. It is critical that data be punched on cards exactly as it is described here. The next four pages describe in detail how batch input cards must be punched. The batch input example problem illustrates a sample deck setup.

****TITLE CARD****

CARD	COLUMN	OPTIONS	OPTION DESCRIPTION
1	1-60	unlimited	alphanumeric title of problem
	61-80	leave blank	

****OPTION CARD****

CARD	COLUMN	OPTIONS	OPTION DESCRIPTION
2	1-2	positive integer<100	number of constraints
	3-4	positive integer<100	number of decision variables
	5-6		output print options
		-2	first tableau,each basis,last tableau
		-1	tableau for every iteration
		0	first and last tableau
		1	each basis
		2	last basis only
	7-9	MAX	problem being solved is a maximize
		MIN	problem being solved is a minimize
	10	leave blank	
	11	0	user is not supplying names for decision variables and constraints
		1	user is supplying names for decision variables and constraints
	12	0	drive artificials out of basis first
		1	use simplex rules throughout problem
	13-14	positive integer<100	number of greater than constraints
	15	leave blank	this column used by interactive LPAFIT
	16	F	abbreviated descriptive headings
		leave blank	extensive descriptive headings
	17-80	leave blank	used by interactive LPAFIT

****COEFFICIENT CARDS****

Use as many cards as necessary to enter coefficients for decision variables and objective function. Coefficient order does not matter. Seven per data card(7(2I2,F6.0)) Coefficients equal to zero need not be entered. For purpose of illustration, assume two cards are needed.

CARD	COLUMN	OPTIONS	OPTION DESCRIPTION
3	1-2	0	objective function coefficient
		positive integer<100	constraint number of coefficient
	3-4	positive integer<100	decision variable number of coef.
	5-10	properly scaled real	coefficient value
	11-12	0	objective function coefficient
		positive integer<100	constraint number of coefficient
	13-14	positive integer<100	decision variable number of coef.
	15-20	properly scaled real	coefficient value
	.	.	.
	.	.	.
	.	.	.
	61-62	0	objective function coefficient
		positive integer<100	
	63-64	positive integer<100	decision variable number of coef.
	65-70	properly scaled real	coefficient value
	71-80	leave blank	
5	1-4	-1-1	signals end of coefficients
	5-80	leave blank	

****RESOURCE CARDS****

Use as many cards as necessary to enter right hand side values(resources). Order does not matter. Seven right hand sides per card(7(I2,A2,F6.0)). Negative RHS's not permitted. For purposes of illustration, assume three constraints; therefore, one card needed.

CARD	COLUMN	OPTIONS	OPTION DESCRIPTION
6	1-2	positive integer<100	constraint number of RHS
	3-4	LT	less than or equal to constraint
		GT	greater than or equal to constraint
		ET	equal to constraint
	5-10	properly scaled real	right hand side(resource value)
	.	.	.
	.	.	.
	.	.	.
	21-22	positive integer<100	constraint number of RHS
	23-24	LT	less than or equal to constraint
		GT	greater than or equal to constraint
		ET	equal to constraint
	25-30	properly scaled real	right hand side(resource value)
7	1-2	-1	signals end of resource cards
	3-80	leave blank	

****CONSTRAINT NAME CARDS****
Use as many cards as necessary to enter names for constraints. These cards necessary only if user indicated on option card that names would be supplied. Eight names per card (8A8). Every card but the last must have 8 names. Assume three constraints; therefore, 1 card. Order counts!

CARD	COLUMN	OPTIONS	OPTION DESCRIPTION
8	1-8	unlimited	alphanumeric name of first constraint
	9-16	unlimited	alphanumeric name of second constraint
	17-24	unlimited	alphanumeric name of third constraint

****DECISION VARIABLE NAME CARDS****
Use as many cards as necessary to enter names for decision variables. These cards necessary only if user indicated on option card that names would be supplied. Eight names per card (8A8). Every card but the last must have 8 names. Assume three decision variables; therefore, 1 card. Order counts!

CARD	COLUMN	OPTIONS	OPTION DESCRIPTION
9	1-8	unlimited	alphanumeric name of first dec. var.
	9-16	unlimited	alphanumeric name of second dec. var.
	17-24	unlimited	alphanumeric name of third dec. var.

EXAMPLE OF BATCH INPUT DATA

The standard-form problem which follows will be used to illustrate batch input data. The user wants to print each basis, to provide names for constraints and decision variables, to drive artifical variables out of the basis first, and to see extensive headings.

```
Maximize Profit = 1000.3*SALES -503.2*COSTS -10000.*BRIBES
Subject to
      INVENTORY:          SALES              +500.*BRIBES≤998758.
      TOTAL COST:         50.235*SALES +COSTS            ≥1500.73
      BACK ORDERS:        901.479*SALES      +22.*BRIBES ≥40125.8
```

Batch input data cards:

```
PROFIT
 3 3 1MAX 10 2
 0 11000.3 0 2-503.2 0 3-10000 1 11.0000 2 150.235 3 1901.48 2 21.0000
 3 1500.00 3 322.0000
-1-1
 1LT998758 2GT1500.7 3GT40126.
-1
INVNTORYTCOST...B-ORDERS
SALES   COSTS   BRIBES
```

PROBLEMS WITH MORE THAN 99 DECISION VARIABLES OR CONSTRAINTS

The linear programming package LPAFIT, as it now exists, cannot deal with problems that have more than 99 decision variables or 99 constraints. This limitation is due to the input format which LPAFIT expects. A user can solve problems which exceed these size limitations by doing some reprogramming. The procedure for doing this is described in detail in LPAFIT PROGRAM DOCUMENTATION FOR THE ADVANCED USER.

HOW TO RUN LPAFIT WITH AN LPKODE DATA DECK

Persons who have been using LPKODE may wish to try using LPAFIT. This is very easy to do with an existing LPKODE data deck. The only card which must be changed before an LPKODE deck can be run with LPAFIT is card 2, the OPTION CARD. See the description of the OPTION CARD on

page 7 of this manual.

OTHER COMMENTS

If an interactive user is entering data for a large problem or does not have enough time to enter an entire problem, the task can be accomplished in pieces. For example, suppose a problem has 90 decision variables and 10 constraints. It might be advantageous in the DEFINING A NEW PROBLEM stage to tell the computer that the problem has 2 constraints. After data is entered for the first two constraints, the user then has the option of quitting or of adding constraints. If the user quits, the effort can be continued at a later time using the MODIFYING AN EXISTING PROBLEM option.

It is possible that an interactive user will get into an undesirable printing situation. For example, suppose a user selected print option -1 (print complete tableau at each iteration) on unit 6 (print at the terminal). Also suppose that the problem being solved has 70 decision variables and 30 constraints. The user would rapidly discover that an enormous amount of printing at the terminal had been requested. To get out of this situation, the user could do the following:

1. Hit the BREAK key.
2. Type %A (hit the RETURN key)
3. Type BEGIN,LP. (hit the RETURN key)
4. Select the RECOVERING FROM MACHINE CRASH OR USER ABORT option.
5. Select option 19 for changes: CHANGE OUTPUT OR SENSITIVITY OPTIONS
6. Make changes.
7. The computer will rerun the problem.

Interactive user typing errors can be corrected by backspacing and writing over the error. Backspacing is accomplished by holding down the CTRL key, then depressing the H key.

HANDY INFORMATION

The interactive user of LPAFIT is asked to select an option from a list several times while a problem is being set up. The program will print these lists if the user requests it but this takes time. Therefore, the lists are included here for ready reference. The first list covers LPAFIT iterative print options.

OPTION	TO PRINT
-2	FIRST TABLEAU, LAST TABLEAU, EACH BASIS
-1	TABLEAU FOR EACH ITERATION
0	FIRST AND LAST TABLEAU ONLY
1	EACH BASIS
2	LAST BASIS ONLY

This list covers problem modification options.

OPTION	ACTION	OPTION	ACTION
1	NO MORE CHANGES	10	CHANGE A VARIABLE NAME
2	REECHO THE PROBLEM	11	CHANGE A CONSTRAINT NAME
3	REPEAT OPTION LIST	12	CHANGE OBJECTIVE NAME
4	ADD A VARIABLE	13	EXCHANGE MAX AND MIN
5	DELETE A VARIABLE	14	CHANGE OBJECTIVE FUNCTION COEFFICIENTS
6	ADD A CONSTRAINT	15	CHANGE THE (GE,LE,EQ)
7	DELETE A CONSTRAINT	16	CHANGE RIGHT HAND SIDE VALUES
8	RESCALE A VARIABLE	17	CHANGE CONSTRAINT COEFFICIENTS
9	RESCALE A CONSTRAINT	18	QUIT NOW--DO NOT RUN LPAFIT--SAVE DATA
		19	CHANGE OUTPUT OR SENSITIVITY OPTIONS

INTERPRETATION OF OUTPUT

Most of LPAFIT's output is sufficiently explained on the listing. However, some explanation of shadow prices is needed. Shadow price for a resource represents the maximum unit price one would be willing to pay to increase the allocation of that resource. This definition is clear when applied to $\leq$ constraints. However, some elaboration is needed for $=$ and $\geq$ constraints. The following problem taken from p65, Hillier and Lieberman, will be used as an example for this discussion:

Minimize $Z = 3X_1 + 5X_2$
subject to:
constraint 1: $X_1 \leq 4$
constraint 2: $2X_2 = 12$
constraint 3: $3X_1 + 2X_2 \geq 18$ $\quad X_1 , X_2 \geq 0$

Hillier and Lieberman derive the following shadow prices for these resources on p96: $(Y_1, Y_2, Y_3) = (0, -1.5, 1)$. The interpretation of these shadow prices is simplified by examining the graphical solution to the problem in Figure 1. Increasing resource 1 from 4 to 5 units does not change (decrease) the value of the objective function. Therefore, the shadow price is 0 (Figure 2).

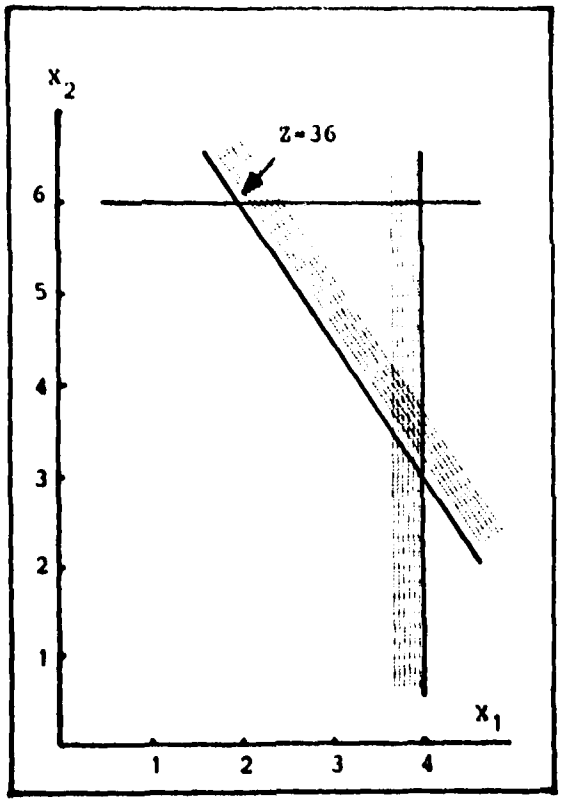


Figure 1. Graphical Solution

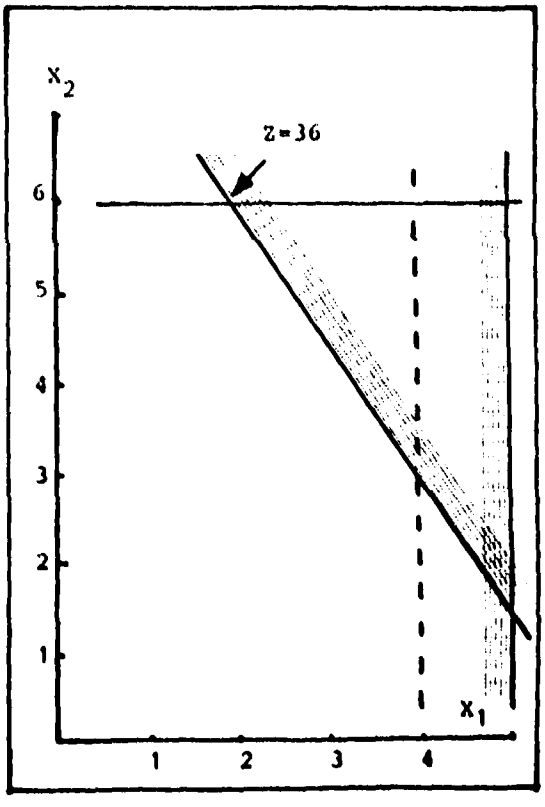


Figure 2. Shadow Price Y

Increasing resource 2 from 12 to 13 causes the value of the objective function to increase by (1.5). Since an increase is contrary to the objective of minimizing Z, resource 2 has a negative shadow price (Figure 3).

The shadow price for resource 3 is 1. To properly interpret this price, constraint 3 must be converted to a $\leq$ constraint. Thus,

constraint 3: $-3X_1 - 2X_2 \leq -18$.

Now, increasing resource 3 by 1 unit from -18 to -17 causes the value of the objective function to decrease to 35 (Figure 4). Since this decrease is in line with the objective of minimizing Z, resource 3 has a positive shadow price.

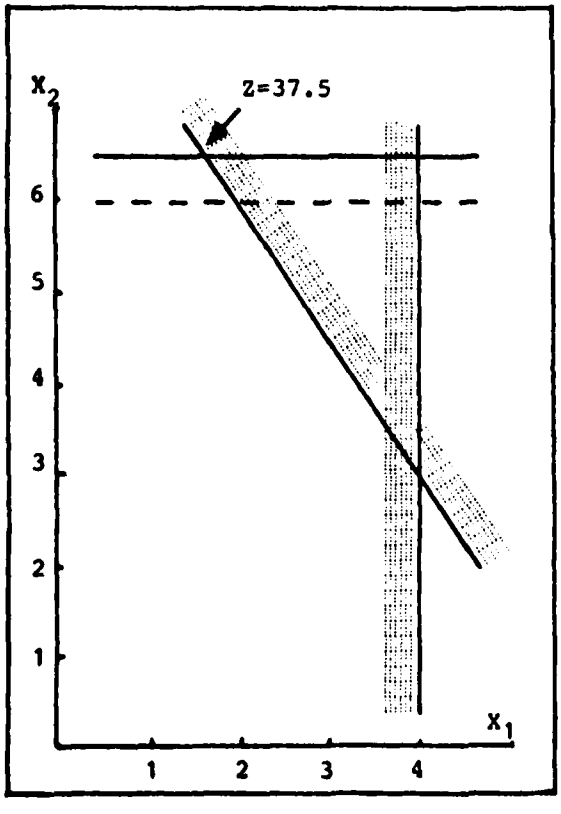


Figure 3. Shadow Price Y

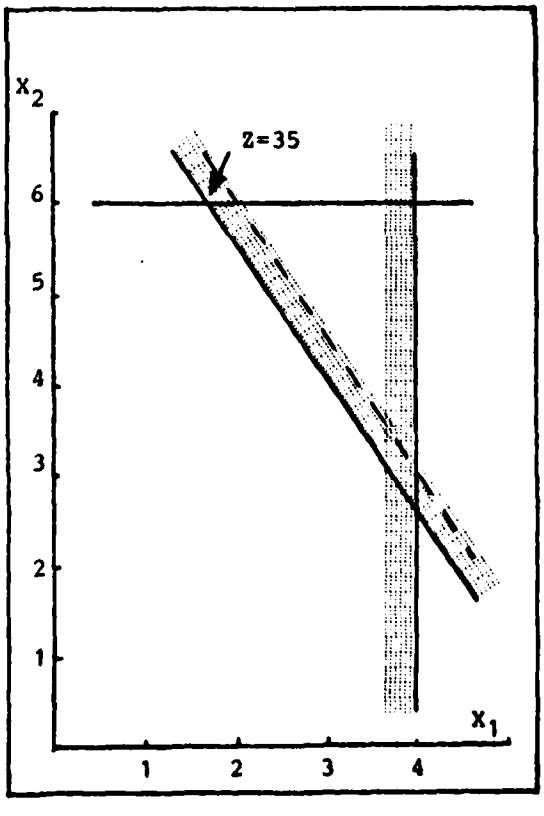


Figure 4. Shadow Price Y

If the example problem were changed to

Maximize $Z = 3X_1 + 5X_2$

with the same constraints as before, then the shadow prices would be

$(Y_1, Y_2, Y_3) = (3, 2.5, 0)$

Figure 5 illustrates the graphical solution. A unit increase in resources 1 and 2 would increase the value of the objective function. Therefore, since an increase is in line with the objective of maximizing Z, resources 1 and 2 have positive shadow prices. The shadow price for resource 3 is 0 because an increase from -18 to -17 in resource 3 would have no effect on the value of the objective function.

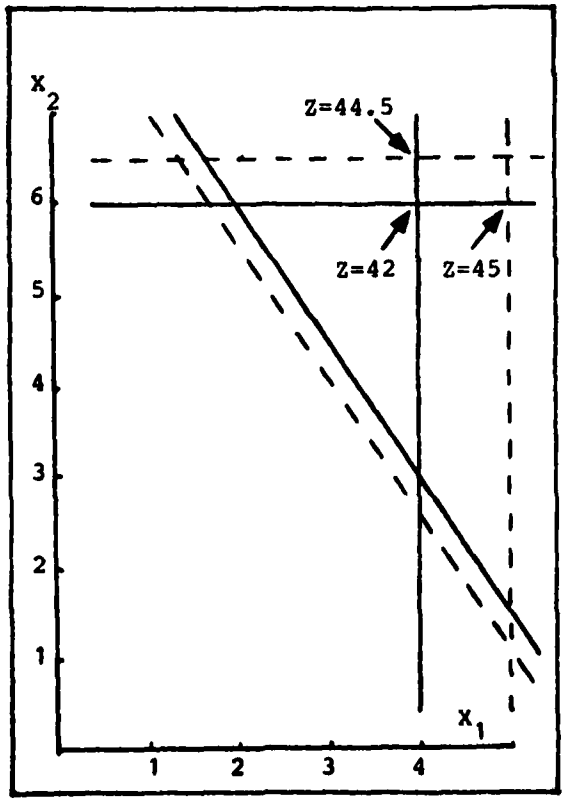


Figure 5. Graphical Solution

One output option is to abbreviate headings which describe the information on the listing. The following headings would have been printed if the option for extensive headings had been selected. They are included for ready reference for users who have abbreviated listings.

```
**************************************************************************
* THIS OUTPUT BLOCK IDENTIFIES CONSTRAINT SHADOW PRICES AND THE          *
* AMOUNT OF SURPLUS IN THE CONSTRAINTS                                   *
* DESCRIPTION OF HEADINGS...                                             *
* SURPLUS.........AMOUNT OF RESOURCE NOT UTILIZED                        *
* VARIABLE........SLACK OR ARTIFICAL VARIABLE WHICH WAS INTRODUCED       *
*                 WITH THE CONSTRAINT                                    *
* SHADOW PRICE...PRICE ONE WOULD BE WILLING TO PAY FOR AN ADDITIONAL     *
*                 UNIT OF RESOURCE. FOR GE CONSTRAINTS, THE PRICE ONE    *
*                 WOULD PAY TO RELAX(REDUCE) THE CONSTRAINT BY A UNIT.   *
*                 EQ CONSTRAINTS MAY HAVE NEGATIVE SHADOW PRICES. IF     *
*                 THIS OCCURS, INCREASING THE RESOURCE DRIVES THE        *
*                 OBJECTIVE FUNCTION IN THE WRONG DIRECTION.             *
**************************************************************************


************************************************************
*TERMINAL SOLUTION AFTER       ITERATIONS                   *
*                                                           *
* DESCRIPTION OF HEADINGS...                                *
* CURRENT PRICE...THE OBJECTIVE FUNCTION COEFFICIENT        *
*                 OF THE DECISION VARIABLE                  *
* ROW ZERO........FINAL TABLEAU ENTRY IN ROW 0.  MUST       *
*                 BE GREATER THAN OR EQUAL TO ZERO.         *
* QUANTITY........TERMINAL VALUE OF THE DECISION            *
*                 VARIABLE.  ONLY BASIC VARIABLES HAVE      *
*                 A NONZERO QUANTITY.                       *
* TOTAL PRICE.....(CURRENT PRICE)*QUANTITY=TOTAL PRICE      *
************************************************************
```

```
**********************************************************************************
* THIS OUTPUT BLOCK IS A SENSITIVITY ANALYSIS ON THE OBJECTIVE FUNCTION          *
* COEFFICIENT OF EACH DECISION VARIABLE.  THE ANALYSIS IS ONLY FOR CHANGES       *
* IN THE OBJECTIVE FUNCTION COEFFICIENTS.  CONSTRAINT COEFFICIENTS ARE NOT       *
* PERMITTED TO VARY FROM THEIR INITIAL VALUES.  THE ANALYSIS ON AN INDIVIDUAL    *
* DECISION VARIABLE ASSUMES THAT ONLY ITS COEFFICIENT VARIES AND THAT ALL        *
* OTHER OBJECTIVE FUNCTION COEFFICIENTS RETAIN THEIR INITIAL VALUES.             *
* DESCRIPTION OF HEADINGS...                                                     *
* EV.....................VARIABLE WHICH ENTERS THE FINAL BASIS AS A RESULT OF   *
*                        A PRICE CHANGE                                          *
* LV.....................VARIABLE WHICH LEAVES THE FINAL BASIS AS A RESULT OF   *
*                        A PRICE CHANGE                                          *
* CURRENT PRICE.........THE INITIAL OBJECTIVE FUNCTION COEFFICIENT OF THE        *
*                        DECISION VARIABLE                                       *
* MINIMUM PRICE.........IF THE OBJECTIVE FUNCTION COEFFICIENT OF THE DECISION    *
*                        VARIABLE FALLS BELOW THIS PRICE, THEN VARIABLE EV WOULD *
*                        ENTER THE FINAL BASIS AND VARIABLE LV WOULD LEAVE THE   *
*                        FINAL BASIS.                                            *
* MAXIMUM PRICE.........IF THE OBJECTIVE FUNCTION COEFFICIENT OF THE DECISION    *
*                        VARIABLE INCREASES ABOVE THIS PRICE, THEN VARIABLE EV   *
*                        WOULD ENTER THE FINAL BASIS AND VARIABLE LV WOULD       *
*                        LEAVE THE FINAL BASIS.                                  *
* ENTERING QUANTITY....VALUE OF THE ENTERING VARIABLE(EV)                        *
**********************************************************************************
* THE FOLLOWING CONCLUSIONS MAY BE DRAWN AS THE RESULT OF CHANGING A SINGLE      *
* OBJECTIVE FUNCTION PRICE COEFFICIENT                                           *
* 1.  IF THE MAGNITUDE OF THE CHANGE IS SUFFICIENT, THEN A VARIABLE MAY LEAVE    *
*     THE FINAL BASIS.                                                           *
* 2.  IF A VARIABLE LEAVES THE FINAL BASIS, THE NEW BASIS WILL STILL BE          *
*     OPTIMAL.                                                                   *
* 3.  WHETHER OR NOT A VARIABLE LEAVES THE FINAL BASIS, THE VALUES OF THE        *
*     BASIC VARIABLES AND THE OBJECTIVE FUNCTION WILL CHANGE.                    *
**********************************************************************************


**************************************************************
* THIS IS A SENSITIVITY ANALYSIS ON THE ORIGINAL RIGHT HAND  *
* SIDE(RHS) VALUES. WHILE ONE RHS CHANGES, THE OTHERS KEEP   *
* THEIR INITIAL VALUES.                                      *
* DESCRIPTION OF VARIABLES...                                *
*      IF THE VALUE OF AN ORIGINAL RHS FALLS BELOW THE       *
*      MINIMUM VALUE OR RISES ABOVE THE MAXIMUM VALUE,       *
*      THEN VARIABLE LV WOULD LEAVE THE FINAL BASIS.         *
**************************************************************
```

APPENDIX B

LPAFIT

PROGRAM DOCUMENTATION

FOR THE ADVANCED USER

DECEMBER 1979

LPAFIT - AFIT LINEAR PROGRAMMING PACKAGE

PROGRAM DOCUMENTATION

FOR THE ADVANCED USER

Michael A. Schiefer

TABLE OF CONTENTS

INTRODUCTION

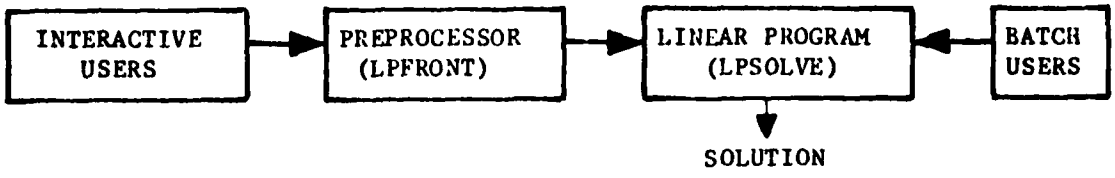
If a user needs this manual, it is assumed that he or she wants to change the LPAFIT linear programming package. Either a mistake has been found in the program which needs to be corrected or the package has to be modified for a special application. The purpose of this manual is to introduce the potential program changer to the overall package structure and to some of the finer programming points. This familiarization will allow the user to make changes more easily. Anyone contemplating a program change will find it necessary to know Fortran. Source decks and listings can be obtained through the AFIT Department of Systems Management.

BASIC PROGRAM STRUCTURE

The interactive linear programming package consists of two Fortran programs and a procedure, named LP, which controls the programs. One of the programs, LPSOLVE, does the linear programming calculations. The second program, LPFRONT, is an interactive preprocessor for LPSOLVE. The batch user does not access LPFRONT and runs LPSOLVE through a procedure called LPBATCH. The linear programming package was split into two programs for the following reasons:

1. It reduces central memory requirements for the batch user, since it is not necessary to load the preprocessor code.
2. It allows the package to run interactively. Central memory limitations imposed on the interactive user would not allow a single massive program to execute. However, several smaller programs, executed one at a time, can accomplish the same task.

The following block diagram illustrates this basic program structure:



LINEAR PROGRAM CONTROLLING PROCEDURES (LP,LPBATCH)

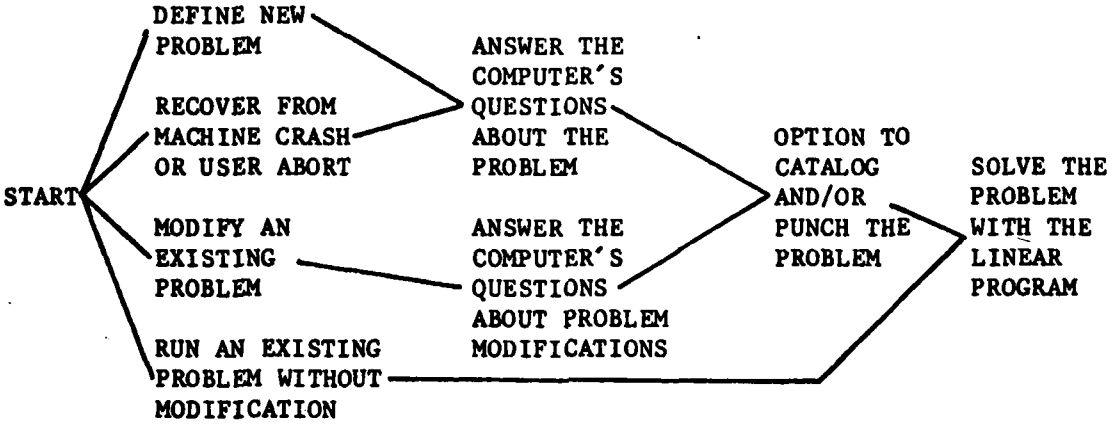
All file manipulations for batch and interactive users are controlled by procedures. This technique was used to make the programs easy to use. The following PROCFIL controls the linear programming package:

```
.PROC,LP.
ATTACH,NOSLIB,ID=LIBRARY,SN=ASD.
LIBRARY(NOSLIB)
SCREEN,80.
ATTACH,LPF,LPFRONT.
LPF.
RETURN,LPF,TAPE3.
ATTACH,LPS,LPSOLVE.
LPS.
RETURN,LPS,TAPE1,TAPE2,NOSLIB.
REVERT.
EXIT,S.
RETURN,LPF,LPS,TAPE1,TAPE2,TAPE3,NOSLIB.
REVERT.
*EOR
.PROC,LPBATCH.
ATTACH,NOSLIB,ID=LIBRARY,SN=ASD.
LIBRARY(NOSLIB)
ATTACH,LPS,LPSOLVE.
COPY,INPUT,TAPE1.
REWIND,TAPE1.
LPS.
RETURN,LPS,TAPE1,TAPE2,NOSLIB.
REVERT.
*EOR
```

LINEAR PROGRAM PREPROCESSOR(LPFRONT)

The functions of this program are to ask the interactive user a series of questions about the problem being solved, to check user responses for correctness whenever possible, and to convert the information provided by the user into formatted input for the linear program LPSOLVE. One of the constraints on the development of LPSOLVE was that it be able to process the data cards which were inputs for LPKODE, another linear program currently used at AFIT. Consequently, LPFRONT does a lot of data manipulation to transform user input into preordained formats.

There are four principal modes in which LPFRONT may be used. The following diagram illustrates how the user might progress through the program:



The rest of this section is devoted to selected topics which will illuminate the structure of LPFRONT.

Local Files. LPFRONT uses five local files. The following list discusses the purpose of each file:

FILE NAME	PURPOSE
TAPE1	Contains formatted data for input to LPSOLVE
TAPE3	Contains same data as TAPE1. Can be punched at user's option
TAPE4	Contains problem data in binary form. This file is rewritten as the user enters additional information. This file can be used to create a permanent file at the option of the user. This file is also used for program recovery if the user aborts or the machine crashes.
TAPE5	Input file (from the terminal)
TAPE6	Output file (at the terminal)

File Manipulations. LPFRONT and procedure LP do all file manipulations for the user. LPFRONT uses Battelle Disk File Manipulation routines to do to following for the user:

1. Create permanent files with user specified file names and passwords.
2. Attach permanent files requested by the user.
3. Purge any file attached by LPFRONT. This permits the program to write on the local file which still exists. A new permanent file can be created after alterations are made to the local file.
4. Route local files to the card punch at the request of the user.
5. Return local files which are no longer needed.

The Battelle manipulation routines are on the library NOSLIB.

F6. Constraint. LPSOLVE requires that all real-number inputs be less than 1E6 in magnitude. This requirement is related to the use of the "Big M" method of solving linear programming problems and will be fully explained later. Because of this limitation, the original LPKODE restricted all input to an F6. format. Therefore, all user-inputted real numbers must lie in the range (-99999.49,999999.49). Notice that negative numbers have one less decimal place available than positive numbers of the same magnitude. LPFRONT evaluates each user-inputted real numbers and converts it to an F6 format with as many decimal places as possible. Encoding is used to achieve this variable format capability. The following type of computer code is used to output the proper number of decimal places:

```
      SUBROUTINE TLPKODE
      .
      .
      IFMT(1)=10H(I2,I2,F6.
      JFMT   =10H(I2,I2,I6)
C
C     WORK IS AN ARRAY WHICH CONTAINS ALL USER-INPUT COEFFICIENTS
C     THIS SECTION LOOPS ON I AND J
      WT=WORK(I,J)
      IF(WT.EQ.0.)GO TO 60
C
C     GETK CALCULATES K,THE MAXIMUM NUMBER OF DECIMAL PLACES ALLOWED
      CALL GETK(WT,K)
      IF(K.EQ.-1)GO TO 55
C
C     THERE IS ROOM FOR AT LEAST THE DECIMAL POINT
C     WRITE WT AS AN F6 REAL. PUT WT AND INDICES INTO IOUT.
      ENCODE(10,54,IFMT(2))K
54    FORMAT(I1,")        ")
      ENCODE(10,IFMT(1),IOUT)J-1 I,WT
      GO TO 60
C
C     K=(-1). THEREFORE, NO ROOM FOR DECIMAL POINT OR ANY DECIMAL PLACE.
C     WRITE WT AS AN I6 INTEGER. ROUND WT TO THE NEAREST INTEGER.
55    IF(WT.GT.0)IWT=WT+.5
      IF(WT.LT.0)IWT=WT-.5
      ENCODE(10,JFMT,IOUT)J-1,I,IWT
60    CONTINUE
      .
      .
```

```
      SUBROUTINE GETK(WT,K)
C     THIS SUBROUTINE CALCULATES THE MAXIMUM NUMBER OF DECIMAL PLACES,K,
C     THAT WT CAN HAVE IN AN F6. FORMAT.
      IF(WT.LT.0.)GO TO 15
C
C     5 DECIMAL PLACES MAXIMUM
      FIRST=.99999499999999999
      DO 10 I=1,6
      IF(WT.GT.FIRST)GO TO 10
      K=6-I
      RETURN
10    FIRST=10.*FIRST
      K=-1
      RETURN
C
C     WT<0, 4 DECIMAL PLACES MAXIMUM
15    FIRST=-.99994999999999999
      DO 20 I=1,5
      IF(WT.LT.FIRST)GO TO 20
      K=5-I
      RETURN
20    FIRST=10.*FIRST
      K=-1
      RETURN
      END
```

If the user finds it necessary to increase the number of decimal places which may be input, the preceding code is the main area of the preprocessor which must be changed.

Program Generated Names For Constraints And Decision Variables.

The user has the option of inputting names for constraints and decision variables. When this option is not selected, the program generates a set of standard names used only in LPFRONT. The following type of code is used to create standard constraint names:

```
      SUBROUTINE THEDATA
      .
C     CREATE A NAME FOR EACH OF NCON CONSTRAINTS.
C     STORE THE NAMES IN ARRAY ILABEL.
      DO 270 I=1,NCON
270   ENCODE(10,274,ILABEL(I))I
274   FORMAT(7H  CNST ,I2,1H )
```

Constraint Relationships And Right Hand Sides. LPFRONT packs each constraint number, constraint relationship, and right hand side into a single word. It uses the following type of programming to do this:

```
      SUBROUTINE TLPKODE
      .
      .
      IFMT(1)=10H(I2,A2,F6.
C
C     LOOP ON I AND J
      WT=WORK(I,J)
      CALL GETK(WT,K)
      ENCODE(10,66,IFMT(2))K
66    FORMAT(I1,")        ")
C
C     J-1 IS THE CONSTRAINT NUMBER
C     ITEMP IS THE CONSTRAINT RELATIONSHIP
C     WT IS THE RIGHT HAND SIDE
      ENCODE(10,IFMT(1),IOUT)J-1,ITEMP,WT
      .
      .
```

This is almost exactly the same as the method used to convert coefficients to an F6 format. The only difference is that ITEMP is an A2 instead of an I2. ITEMP is output as one of the following: GT,LT, or ET. Notice that these have been changed from the user input GE,LE, or EQ. This change was made because the original LPKODE expected GT,LT, or ET. LPFRONT makes this switch with code similar to the following:

```
C     LOOP ON I
      ITEMP = ILABEL(I).AND.MASK(6).OR.(0024000000000000000B)
```

ILABEL(I) contains the original GE,LE, or EQ. The following example for ILABEL(I)=GE illustrates the technique (ILABEL(I) was read with an A2 format):

```
GE     octal-07055555555555555555

GE       bit-000111000101101101101101101101101101101101101101101101101101
MASK(6)  bit-111111000000000000000000000000000000000000000000000000000000

GE.AND.MASK(6)
         bit-000111000000000000000000000000000000000000000000000000000000

:T     octal-00240000000000000000
:T       bit-000000010100000000000000000000000000000000000000000000000000

GE.AND.MASK(6).OR.(00240000000000000000B)
         bit-000111010100000000000000000000000000000000000000000000000000
       octal-07240000000000000000

translation-GT
```

Reduced Core Requirements. The problem size which LPFRONT can process is limited by the amount of central memory which is made available to interactive users. To reduce core requirements, buffer lengths have been set equal to 0 on the PROGRAM card. This allows the system to use the smallest buffer size possible.

LINEAR PROGRAM (LPSOLVE)

LPSOLVE solves linear programming problems using the simplex method. Inputs to the program may be generated by the preprocessor or submitted through batch. LPSOLVE checks input for errors and then solves the problem. It also offers sensitivity analyses to the interactive user.

Local Files. LPSOLVE has four local files. The following list discusses the purpose of each file:

FILE NAME PURPOSE

TAPE1....Input file containing problem information

TAPE2....Output file which can be routed to the high-speed printer at the option of the interactive user

TAPE5....Interactive input file

TAPE6....Output file which is printed at the terminal of an interactive user and on the line printer for the batch user

File Manipulations. LPSOLVE does only one file manipulation. If an interactive user wants tableau iterative information printed on the high-speed printer, LPSOLVE uses the Battelle ROUTE routine to satisfy this option.

F6. Constraint. LPSOLVE uses the "Big M" simplex method to deal with EQ and GE constraints. This method requires the program to accurately add or subtract numbers from BIGM. This implies, for example, that BIGM cannot be as large as 1E100. This is true because the machine representation of 1E100 + 5.78 is 1E100. Thus, the 5.78 is lost by adding it to 1E100. Obviously, the magnitude of BIGM must depend on machine word length. The CDC6600 stores numbers with 14 significant figures in single precision. LPSOLVE currently allows up to 5 decimal places in input data. BIGM was therefore selected to be as large as possible and still allow resolution to the fifth decimal place. Thus, the following condition was required to be true:

(BIGM + .00001) - BIGM = .00001

Therefore, BIGM was selected to be 1E8=100,000,000.

Variable Dimensions. LPSOLVE has variable dimension capability. The program has one large array, WORK, which is split up according to

user inputs. The user can reduce core requirements to a minimum for a problem by properly dimensioning array WORK. The formula for calculating the required dimension of WORK is

1. (NVAR+NCON+NGT)*(3+NCON)+6*NCON+16

   if the user supplies names for constraints and decision variables

2. (NVAR+NCON+NGT)*(3+NCON)+5*NCON

   if the user does not supply names

The dimension of WORK in the version of LPSOLVE which resides on the system is 11000.

MODIFYING LPAFIT TO SOLVE PROBLEMS WITH MORE THAN 99 CONSTRAINTS OR DECISION VARIABLES

The first modification which must be made to allow the LPAFIT package to solve larger problems is to increase array dimensions. Such increases will likely disallow interactively running the package due to central memory limitations. Therefore, the only way to solve larger problems may be to run LPSOLVE through batch using the LPBATCH procedure.

LPSOLVE Changes. Because variable dimensioning is built into LPSOLVE, it is easy to modify the program to handle more than 99 constraints or decision variables. In addition to properly dimensioning array WORK, the only other changes which must be made are in SUBROUTINES INPUT1 and INPUT2. Each READ format on unit 1 needs to be checked. The output routines have been programmed to process up to 999 decision variables and constraints. The input stream to LPSOLVE will have to change to correspond to modified input formats. This implies that the preprocessor could not be used in its current form to create input for an altered LPSOLVE. A clever programmer could add some code to LPSOLVE

which would allow part of its input to be read in the format generated by LPFRONT. The remaining input for decision variables or constraints above 99 could be read with different formats.

LPFRONT Changes. It would be difficult to modify LPFRONT to process more than 99 constraints or decision variables. This is true because the program is currently using almost all of the central memory available to interactive users. Consequently, the changes required to enable LPFRONT to handle larger problems will not be addressed in detail. If modifications must be made, the following areas ought to be examined:

1. The dimensions of arrays WORK and ILABEL in each common block.
2. All WRITE formats in SUBROUTINE TLPKODE.
3. All ENCODE formats in SUBROUTINE TLPKODE.

MODIFYING LPAFIT TO ACHIEVE MORE THAN AN F6 FORMAT

No consideration will be given here as to how to achieve more than 5 decimal places in user-inputs. In order for the machine to track more than 5 decimal places accurately, BIGM would have to be reduced by an order of magnitude for each decimal place added. In turn, the maximum size of user inputs would also have to be reduced.

It might, however, be desirable to increase the number of significant figures which the user is permitted to input. This would require a relaxation of the F6 constraint. Changes must be made in LPSOLVE and LPFRONT to increase the number of significant figures which may be innput.

LPSOLVE Changes. The primary modifications in LPSOLVE are to the READ formats for unit 1. The number of coefficients read from each card may also have to be reduced.

LPFRONT Changes. Modifications to LPRONT are more extensive than those to LPSOLVE. Basically, all WRITE formats and ENCODE formats in SUBROUTINE TLPKODE must be checked. In addition, SUBROUTINE GETK, which calculates number of decimal places possible, must be carefully changed.

APPENDIX C

PROGRAM LISTINGS

TABLE OF CONTENTS

```
.PROC,LP.
ATTACH,NOSLIB,ID=LIBRARY,SN=ASD.
LIBRARY(NOSLIB)
SCREEN,80.
ATTACH,LPF,LPFRONT.
LPF.
RETURN,LPF,TAPE3.
ATTACH,LPS,LPSOLVE.
LPS.
RETURN,LPS,TAPE1,TAPE2,NOSLIB.
REVERT.
EXIT,S.
RETURN,LPF,LPS,TAPE1,TAPE2,TAPE3,NOSLIB.
REVERT.
*EOR
.PROC,LPBATCH.
ATTACH,NOSLIB,ID=LIBRARY,SN=ASD.
LIBRARY(NOSLIB)
ATTACH,LPS,LPSOLVE.
COPY,INPUT,TAPE1.
REWIND,TAPE1.
LPS.
RETURN,LPS,TAPE1,TAPE2,NOSLIB.
REVERT.
*EOR
```

```
      PROGRAM LPSOLVE(INPUT,OUTPUT,TAPE1=0,TAPE2=0,TAPE5=INPUT,
     1                TAPE6=OUTPUT)
C     ****************************************************************
C     *  LPSOLVE SOLVES LINEAR PROGRAMMING PROBLEMS.  IT WAS WRITTEN BY*
C     *  CAPT MICHAEL A SCHIEFER, GRADUATE OPERATIONS RESEARCH         *
C     *  CLASS 79D, AIR FORCE INSTITUTE OF TECHNOLOGY, WRIGHT-PATTERSON*
C     *  AIR FORCE BASE.  THE PRIMARY FUNCTION OF THE PROGRAM IS TO    *
C     *  SERVE AS A LEARNING AID FOR AFIT STUDENTS.  THE PROGRAM IS    *
C     *  DESIGNED TO ILLUSTRATE THE SIMPLEX METHOD.  CONSEQUENTLY IT   *
C     *  SOMETIMES SACRIFICES EFFICIENCY TO GAIN CLAIRITY.  FOR        *
C     *  EXAMPLE,THE PROGRAM DOES NOT USE THE MEMORY EFFICIENT         *
C     *  REVISED SIMPLEX METHOD.  THE USER NEED ONLY DETERMINE THE     *
C     *  OBJECTIVE FUNCTION AND ITS CONSTRAINTS.  THE PROGRAM  SUPPLIES*
C     *  SLACK, ARTIFICAL AND SURPLUS VARIABLES. LPSOLVE DOES NOT      *
C     *  PERMIT NEGATIVE RIGHT HAND SIDES OR NEGATIVE DECISION         *
C     *  VARIABLES.  TO MOST STUDENTS, LPSOLVE WILL BE A BLACK BOX.    *
C     *  HOWEVER, IT IS WRITTEN IN A MODULAR FASHION AND IS HEAVILY    *
C     *  COMMENTED.  CONSEQUENTLY, PROGRAM CHANGES TO FULFILL SPECIAL  *
C     *  NEEDS WILL BE STRAIGHT FOREWARD.  SURBOUTINES ARE LISTED IN   *
C     *  ALPHABETICAL ORDER, BUT THEY ARE MORE EASILY UNDERSTOOD IF    *
C     *  THEY ARE EXAMINED IN THE ORDER IN WHICH THEY ARE CALLED.      *
C     *  INPUT FORMATS LIMIT THE CODE TO 99 DECISION VARIABLES AND 99  *
C     *  CONSTRAINTS.  THIS CAN BE ALTERED BY CHANGING FORMATS IN      *
C     *  INPUT1 AND INPUT2.  MOST STORAGE IS IN ARRAY WORK WHICH       *
C     *  IS PARTITIONED BY THE PROGRAM TO MEET THE REQUIREMENTS OF THE *
C     *  PROBLEM BEING SOLVED.  THE DIMENSION AND STRUCTURE OF WORK    *
C     *  ARE FURTHER DESCRIBED IN SUBROUTINE INPUT1.  TO RUN LARGE     *
C     *  PROBLEMS, THE ONLY PROGRAM DIMENSION WHICH MUST BE CHANGED IS *
C     *  WORK'S.  LPSOLVE ALSO DOES SOME SENSITIVITY ANALYSIS.         *
C     *  A SOURCE DECK FOR THIS PROGRAM IS IN THE AFIT SYSTEMS         *
C     *  MANAGEMENT CARD FILE.                                         *
C     ****************************************************************
      COMMON/INDEX/IRHS,IORHS,IPRICE,ITEMP,IBASIS,IOBASIS,IT,
     1            ICNAME,IDVNAME,ISTATUS
      COMMON BIGM,COEF,CYCLE,DRIVE,IBANER,IBOT,INDEX,ITOP,IPRINT,MAX,ME,
     1NAMES,HEAD,NBASIS,NCOLS,SEN,TITLE(6),TCOLP1,TOTAL,TOTCOL,UNIT,ZERO
      DIMENSION WORK(11000)
      INTEGER UNIT
      CALL REQUEST(5HTAPE2,2H*Q)
C
C     SEE SUBROUTINE INPUT1 FOR THE DIMENSION FORMULAS FOR WORK.
C     IF THE DIMENSION OF WORK IS CHANGED, THE ONLY OTHER NECESSARY
C     CHANGE IS THE WORK INITIALIZATION LOOP LIMIT IMMEDIATELY BELOW
C
C     INITIALIZE WORK
      DO 5 I=1,11000
5     WORK(I)=0.
C
C     READ FIRST PART OF INPUT AND CALCULATE INDICES TO SPLIT WORK ARRAY
      CALL INPUT1(NROWS)
C
C     READ THE REST OF INPUT
      CALL INPUT2(WORK(IT),WORK(IRHS),WORK(IORHS),WORK(IPRICE),
     1            WORK(ITEMP),WORK(IBASIS),WORK(IOBASIS),WORK(ICNAME),
     1            WORK(IDVNAME),WORK(ISTATUS),NROWS)
```

```
C
C     USE SIMPLEX METHOD TO SOLVE LP PROBLEM
      CALL SOLVE(WORK(IT),WORK(IRHS),WORK(IPRICE),WORK(ITEMP),
     1           WORK(IBASIS),WORK(ISTATUS),NROWS)
C
C     ROUTE OUTPUT TO PRINTER IF REQUESTED BY INTERACIVE USER
C     THE FOLLOWING TWO SUBROUTINES ARE BATTELLE DISK FILE ROUTINES
      IF(UNIT.NE.2)GO TO 10
      REWIND 2
      CALL ROUTE(5HTAPE2,3HDC=,2HPR,4HTID=,2HBB,3HST=,3HCSB,4HFID=,
     1IBANER)
C     POST OPTIMALITY ANALYSIS (SENSITIVITY)
10    CALL POSTOP(WORK(IT),WORK(IRHS),WORK(IORHS),WORK(ISTATUS),
     1            WORK(IPRICE),WORK(ITEMP),WORK(IBASIS),WORK(IOBASIS),
     1            WORK(ICNAME),WORK(IDVNAME),NROWS)
C
C     INTERACTIVE SENSITIVITY ANALYSIS
      CALL SA(NROWS,WORK(IPRICE),WORK(ITEMP),WORK(IOBASIS),WORK(IT),
     1        WORK(IORHS),WORK(IRHS),WORK(IBASIS),WORK(ISTATUS))
      STOP
      END
```

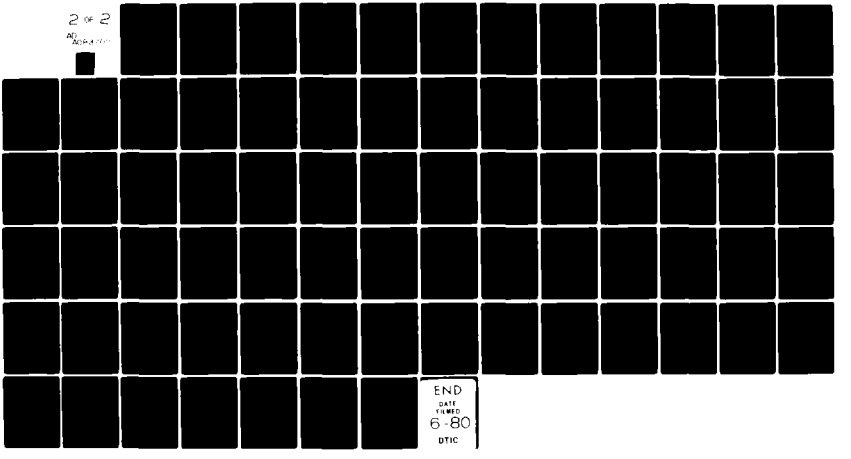
```
      SUBROUTINE ADDCOL(IROW,PCOEFF,TCOEFF,RSOURCE,RHS,ORHS,T,PRICE,
     1                  IBASIS,OBASIS,TEMCOL,NROWS)
C     ****************************************************************
C     *  SUBROUTINE ADDCOL IS CALLED BY INPUT2.  IT ADDS SLACK,      *
C     *  ARTIFICAL, AND SURPLUS VARIABLES TO THE TABLEAU.  IT ALSO   *
C     *  SAVES RIGHT HAND SIDE VALUES.                               *
C     *  LIST OF VARIABLES...                                        *
C     *  PCOEFF....COEFFICIENT(PRICE) OF AN ADDED VARIABLE IN THE    *
C     *            OBJECTIVE FUNCTION.  EQUAL TO 0. FOR SLACK AND    *
C     *            SURPLUS VARIABLES.  EQUAL TO BIG M FOR            *
C     *            ARTIFICAL VARIABLES.                              *
C     *  TCOEFF....COEFFICIENT OF A VARIABLE ADDED IN THE TABLEAU.   *
C     *            EQUAL TO 1. FOR SLACKS AND ARTIFICALS AND -1 FOR  *
C     *            SURPLUS VARIABLES.                                *
C     ****************************************************************
      DIMENSION RHS(1),ORHS(1),T(NROWS,1),IBASIS(1),OBASIS(1),PRICE(1)
      INTEGER OBASIS,TEMCOL
C
C     SAVE RIGHT HAND SIDE
      RHS(IROW)=RSOURCE
      ORHS(IROW)=RSOURCE
C
C     INCREMENT NUMBER OF COLUMNS
      TEMCOL=TEMCOL+1
C
C     SAVE COEFFICIENT OF SLACK, ARTIFICAL, OR SURPLUS VARIABLE
C     IT WILL BE -1 OR 1
      T(IROW,TEMCOL)=TCOEFF
C
C     SAVE COEFFICIENT OF SLACK ,ARTIFICAL, OR SURPLUS VARIABLE IN
C     OBJECTIVE FUNCTION. IT WILL BE 0 OR BIGM
      INDEX=NROWS+TEMCOL
      PRICE(INDEX)=PCOEFF
C
C     SAVE ORIGINAL BASIS ELEMENTS
C     SURPLUS VARIABLES ARE NOT IN BASIS-ONLY CORRESPONDING
C     ARTIFICAL VARIABLE
      IF(TCOEFF.LT.0.)RETURN
      OBASIS(IROW)=TEMCOL
      IBASIS(IROW)=TEMCOL
C
C     INITIALIZE PRICE OF BASIS ELEMENTS
      PRICE(IROW)=PCOEFF
      RETURN
      END
```

```
      SUBROUTINE ANSWER(NROWS,IBASIS,DVNAME,PRICE,TEMP,RHS,STATUS)
C     ***************************************************************
C     *  SUBROUTINE ANSWER OUTPUTS THE FINAL VALUES OF ALL DECISION  *
C     *  VARIABLES, THEIR SHADOW PRICES, AND THE FINAL VALUE OF THE  *
C     *  OBJECTIVE FUNCTION. ANSWER IS CALLED BY POSTOP.             *
C     *  LIST OF VARIABLES...COMMON BLOCK VARIABLES DEFINED IN INPUT2*
C     *  NROWPI....POINTER TO THE PRICE FOR DECISION VARIABLE I      *
C     *  ROWJ......INTEGER. IF DECISION VARIABLE I IS IN THE BASIS,  *
C     *            IT IS IN ROW J                                    *
C     *  TVALUE....CONTRIBUTION OF A DECISION VARIABLE TO THE VALUE  *
C     *            OF THE OBJECTIVE FUNCTION.                        *
C     ***************************************************************
C
      COMMON BIGM,COEF,CYCLE,DRIVE,IBANER,IBOT,INDEX,ITOP,IPRINT,MAX,ME,
     1NAMES,HEAD,NBASIS,NCOLS,SEN,TITLE(6),TCOLP1,TOTAL,TOTCOL,UNIT,ZERO
      DIMENSION IBASIS(1),DVNAME(1),PRICE(1),TEMP(1),RHS(1),STATUS(1)
      INTEGER ROWJ,STATUS
      LOGICAL NAMES,HEAD
C
      WRITE(6,1)NBASIS
1     FORMAT(1H1,57(1H*),/,2H *,23HTERMINAL SOLUTION AFTER,I5,
     111H ITERATIONS,16X,1H*)
      IF(HEAD)WRITE(6,2)
2     FORMAT(2H *,55X,1H*,/,
     129H * DESCRIPTION OF HEADINGS...,28X,1H*,/,
     158H * CURRENT PRICE...THE OBJECTIVE FUNCTION COEFFICIENT    *,/,
     12H *,17X,24HOF THE DECISION VARIABLE,14X,1H*,/,
     158H * ROW ZERO.......FINAL TABLEAU ENTRY IN ROW 0.  MUST   *,/,
     12H *,17X,39HBE GREATER THAN OR EQUAL TO ZERO.     *,/,
     158H * QUANTITY.......TERMINAL VALUE OF THE DECISION        *,/,
     12H *,17X,39HVARIABLE.  ONLY BASIC VARIABLES HAVE  *,/,
     12H *,17X,19HA NONZERO QUANTITY.,19X,1H*,/,
     158H * TOTAL PRICE.....(CURRENT PRICE)*QUANTITY=TOTAL PRICE  *)
      WRITE(6,3)
3     FORMAT(1X,57(1H*),//,
     12X,8HDECISION,7X,7HCURRENT,4X,6H ROW  ,17X,5HTOTAL,/,
     12X,8HVARIABLE,8X,5HPRICE,6X,5HZERO ,4X,8HQUANTITY,5X,5HPRICE,/,
     11X,12(1H-),4(2X,9(1H-)))
C
C     EXAMINE DECISION VARIABLES ONLY
      DO 50 I=1,NCOLS
      NROWPI=NROWS+I
C     IS DECISION VARIABLE I A BASIS ELEMENT?
      IF(STATUS(I).LT.0)GO TO 31
C     YES, I IS IN THE BASIS, FIND THE ROW
         DO 10 J=1,NROWS
         IF(IBASIS(J).NE.I)GO TO 10
         ROWJ=J
         GO TO 20
10       CONTINUE
C
```

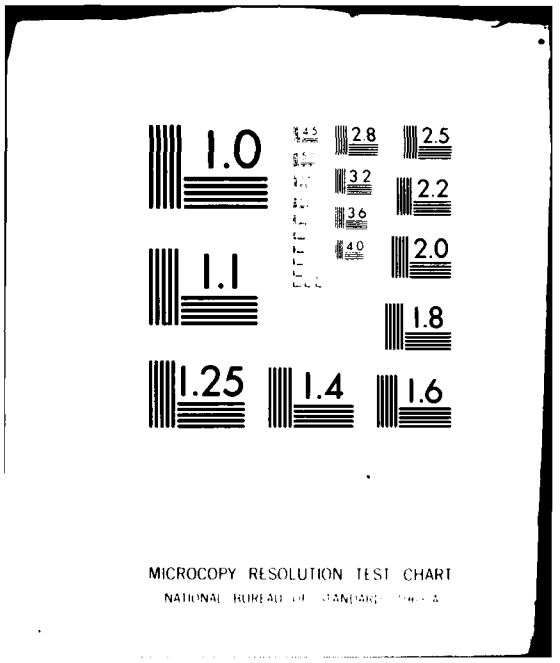
```
C     OUTPUT QUANTITY AND TOTAL PRICE
20    TVALUE=RHS(ROWJ)*PRICE(NROWPI)
      IF(.NOT.NAMES)GO TO 25
      WRITE(6,32)I,DVNAME(I),PRICE(NROWPI),TEMP(I),RHS(ROWJ),TVALUE
      GO TO 50
25    WRITE(6,38)I,PRICE(NROWPI),TEMP(I),RHS(ROWJ),TVALUE
      GO TO 50
C
C     I IS NOT IN THE BASIS, THEREFORE DO NOT OUTPUT QUANTITY
31    IF(.NOT.NAMES)GO TO 35
      WRITE(6,32)I,DVNAME(I),PRICE(NROWPI),TEMP(I)
32    FORMAT(1X,I3,1X,A8,4(1PE11.3))
      GO TO 50
35    WRITE(6,38)I,PRICE(NROWPI),TEMP(I)
38    FORMAT(1X,I3,9X,4(1PE11.3))
50    CONTINUE
      WRITE(6,60)TOTAL
60    FORMAT(/,47X,10(1H-),/,11X,33HTOTAL VALUE OF OBJECTIVE FUNCTION,
     11PE13.5)
      RETURN
      END
```

AD-A083 707 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOO--ETC F/G 9/2
LPAFIT, AN INTERACTIVE LINEAR PROGRAMMING PACKAGE DEVELOPED AT --ETC(U)
DEC 79 M A SCHIEFER
UNCLASSIFIED AFIT/GOR/SM/79D-7 NL

2 OF 2
AD A083707



END
DATE FILMED
6-80
DTIC


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS

```
      SUBROUTINE ARTOUT(NROWS,PRICE,TEMP,T,IBASIS,RHS,STATUS)
C     ****************************************************************
C     *  SUBROUTINE ARTOUT TRIES TO DRIVE ARTIFICAL VARIABLES OUT    *
C     *  OF THE BASIS.  CALLED BY SOLVE.                             *
C     *  LIST OF VARIABLES...                                        *
C     *  NEG.......LOGICAL VARIABLE.  NEG=.TRUE. IMPLIES THAT       *
C     *            NEGATIVES ARE IN ROW 0 AND THAT THE SOLUTION HAS  *
C     *            NOT BEEN FOUND.                                   *
C     *  LEAVE.....ROW OF LEAVING VARIABLE                           *
C     ****************************************************************
      COMMON BIGM,COEF,CYCLE,DRIVE,IBANER,IBOT,INDEX,ITOP,IPRINT,MAX,ME,
     1NAMES,HEAD,NBASIS,NCOLS,SEN,TITLE(6),TCOLP1,TOTAL,TOTCOL,UNIT,ZERO
      DIMENSION PRICE(1),TEMP(1),STATUS(1),IBASIS(1)
      INTEGER TOTCOL,STATUS
      LOGICAL CYCLE,NEG,DRIVE
C
      NEG=.FALSE.
C
C     EXAMINE BASIS FOR ARTIFICALS
      DO 5 I=1,NROWS
      IF(STATUS(IBASIS(I)).EQ.2)GO TO 8
5     CONTINUE
C
C     NONE FOUND, ALL ARTIFICALS DRIVEN OUT
      DRIVE=.FALSE.
      RETURN
C
C     ARTIFICALS DETECTED IN BASIS, TRY TO DRIVE OUT
8     DO 10 I=1,TOTCOL
C
C     IS I A POTENTIAL ENTERING VARIABLE? IF NOT, EXAMINE NEXT I
      IF(TEMP(I).GE.0.)GO TO 10
C
C     NEGATIVE FOUND IN ROW 0, COLUMN I. DETERMINE LEAVING VARIABLE.
      NEG=.TRUE.
      CALL ROW(LEAVE,I,CYCLE,T,RHS,NROWS,ZERO,DUMMY)
C
C     IF SOLUTION IS UNBOUNDED, RETURN
      IF(.NOT.CYCLE)RETURN
C     IF LEAVING VARIABLE IS NOT ARTIFICAL, EXAMINE NEXT I
      IF(STATUS(IBASIS(LEAVE)).NE.2)GO TO 10
C
C     LEAVING VARIABLE IS ARTIFICAL, DETERMINE NEW BASIS
      CALL SWITCH(LEAVE,I,TCOLP1,T,NROWS,IBASIS,PRICE,STATUS)
      RETURN
10    CONTINUE
C
```

```
C     ARTIFICALS IN BASIS BUT NOT DRIVEN OUT. HAS SOLUTION BEEN FOUND?
      IF(NEG)GO TO 20
C
C     NO NEGATIVES FOUND IN ROW 0, THEREFORE SOLUTION FOUND
      CYCLE=.FALSE.
      RETURN
C
C     NEGATIVES FOUND BUT UNABLE TO DRIVE ARTIFICALS OUT,DON'T TRY AGAIN
20    DRIVE=.FALSE.
      RETURN
      END
```

```
      SUBROUTINE COLUMN(IANS,NCOLS)
C     ************************************************************************
C     *  SUBROUTINE COLUMN IS USED IN SENSITIVITY ANALYSIS. IT READS   *
C     *  AND CHECKS THE NUMBER OF A DECISION VARIABLE WHICH IS TO BE   *
C     *  CHANGED. CALLED BY DELTA.                                     *
C     ************************************************************************
C
10    WRITE(6,11)
11    FORMAT(38H WHICH DECISION VARIABLE? (INTEGER)...)
      READ(5,*)IANS
      IF(IANS.LE.NCOLS.AND.IANS.GT.0)RETURN
12    WRITE(6,13)NCOLS,IANS
13    FORMAT(31H INCORRECT RESPONSE.  THERE ARE,I5,
     120H DECISION VARIABLES.,/,31H YOU WISH A CHANGE FOR VARIABLE,I5)
      GO TO 10
      END
```

```
      SUBROUTINE DELTA(NROWS,PRICE,TEMP,OBASIS,T,ORHS,RHS,STATUS)
C     ****************************************************************
C     *  SUBROUTINE DELTA INTERACTIVELY READS AND CHECKS ALL USER     *
C     *  INPUTS FOR SENSITIVITY ANALYSIS. CALLED BY SA.               *
C     ****************************************************************
      COMMON BIGM,COEF,CYCLE,DRIVE,IBANER,IBOT,INDEX,ITOP,IPRINT,MAX,ME,
     1NAMES,HEAD,NBASIS,NCOLS,SEN,TITLE(6),TCOLP1,TOTAL,TOTCOL,UNIT,ZERO
      DIMENSION PRICE(1),TEMP(1),OBASIS(1),T(NROWS,1),ORHS(1),RHS(1),
     1STATUS(1)
      INTEGER OBASIS,NROWS,STATUS
1     WRITE(6,2)
2     FORMAT(47H WHAT PRINT OPTION WOULD YOU LIKE? (INTEGER)...)
      READ(5,*)IPRINT
      IF(IABS(IPRINT).LE.2)GO TO 6
      WRITE(6,3)
3     FORMAT(42H INCORRECT RESPONSE,ABS(IPRINT) MUST BE <3)
      GO TO 1
4     FORMAT(A1)
17    FORMAT(57H MAXIMUM ALLOWABLE VALUE OF COEFFICIENT=999999. YOU INPU
     1T,1PE11.3)
C
C     CHECK FOR CHANGES IN THE OBJECTIVE FUNCTION
6     WRITE(6,7)
7     FORMAT(48H ANY CHANGES IN THE OBJECTIVE FUNCTION? (Y,N)...)
      CALL YESNO,RETURNS(10,30,6)
10    CALL COLUMN(IANS,NCOLS)
      POUT=COEF*PRICE(NROWS+IANS)
15    WRITE(6,16)POUT
16    FORMAT(17H OLD COEFFICIENT=,F12.5,28H   NEW COEFFICIENT=? (F6)...)
      CALL REPLY(ANS),RETURNS(15)
18    INDEX=NROWS+IANS
      TEMP(IANS)=TEMP(IANS)-(ANS-PRICE(INDEX))
      PRICE(INDEX)=ANS*COEF
19    WRITE(6,20)
20    FORMAT(53H ANY MORE CHANGES IN THE OBJECTIVE FUNCTION? (Y,N)...)
      CALL YESNO,RETURNS(10,30,19)
C
C     CHECK FOR CHANGES IN THE CONSTRAINTS
30    WRITE(6,31)
31    FORMAT(41H ANY CHANGES IN THE CONSTRAINTS? (Y,N)...)
      CALL YESNO,RETURNS(33,80,30)
33    WRITE(6,34)
34    FORMAT(31H WHICH CONSTRAINT? (INTEGER)...)
      READ(5,*)IANS
      IF(IANS.LE.NROWS)GO TO 38
35    WRITE(6,36)NROWS,IANS
36    FORMAT(31H INCORRECT RESPONSE.  THERE ARE,I5,12H CONSTRAINTS,/,
     133H YOU WISH A CHANGE FOR CONSTRAINT,I5)
      GO TO 33
```

```
38    IF(IANS.LE.0)GO TO 35
39    WRITE(6,40)
40    FORMAT(71H CHANGE IN DECISION VARIABLE COEFFICIENT OR RIGHT HAND S
     1IDE? (DV,RS)...)
      READ(5,4)ANS
      IF(ANS.EQ.1HD)GO TO 42
      IF(ANS.EQ.1HR)GO TO 60
      WRITE(6,41)
41    FORMAT(35H INCORRECT RESPONSE. REPLY DV OR RS)
      GO TO 39
42    CALL COLUMN(JANS,NCOLS)
45    WRITE(6,46)
46    FORMAT(32X,25HOLD COEFFICIENT=? (F6)...)
      CALL REPLY(OLD),RETURNS(45)
47    WRITE(6,48)
48    FORMAT(32X,25HNEW COEFFICIENT=? (F6)...)
      CALL REPLY(ANS),RETURNS(47)
49    DASUBIJ=ANS-OLD
      FACTOR=TEMP(OBASIS(IANS))
      IF(IABS(STATUS(OBASIS(IANS))).EQ.2)FACTOR=FACTOR-BIGM
      TEMP(JANS)=TEMP(JANS)+DASUBIJ*FACTOR
      DO 50 I=1,NROWS
50    T(I,JANS)=T(I,JANS)+DASUBIJ*T(I,OBASIS(IANS))
51    WRITE(6,52)
52    FORMAT(42H ANY MORE CHANGES IN CONSTRAINTS? (Y,N)...)
      CALL YESNO,RETURNS(33,80,51)
60    WRITE(6,61) ORHS(IANS)
61    FORMAT(9X,8HOLD RHS=,F12.5,11X,17HNEW RHS=? (F6)...)
      READ(5,*)ANS
      IF(ANS.LT.999999.5)GO TO 62
      WRITE(6,17)ANS
      GO TO 60
62    IF(ANS.GE.0)GO TO 64
      WRITE(6,63)ANS
63    FORMAT(49H NEGATIVE RIGHT HAND SIDES NOT ALLOWED. YOU INPUT,F12.5)
      GO TO 60
64    DRHS=ANS-ORHS(IANS)
      ORHS(IANS)=ANS
      DO 65 I=1,NROWS
65    RHS(I)=RHS(I)+DRHS*T(I,OBASIS(IANS))
      GO TO 51
80    CONTINUE
      RETURN
      END
```

```
      SUBROUTINE DUALSM(NROWS,RHS,T,TEMP,IBASIS,PRICE,STATUS)
C     ***********************************************************************
C     *  THIS SUBROUTINE CHECKS FOR NEGATIVE RIGHT HAND SIDES DURING   *
C     *  SENSITIVITY ANALYSIS. IF ANY ARE FOUND, THE DUAL SIMPLEX      *
C     *  METHOD IS USED TO SOLVE THE PROBLEM. CALLED BY SA.            *
C     ***********************************************************************
C
      COMMON BIGM,COEF,CYCLE,DRIVE,IBANER,IBOT,INDEX,ITOP,IPRINT,MAX,ME,
     1NAMES,HEAD,NBASIS,NCOLS,SEN,TITLE(6),TCOLP1,TOTAL,TOTCOL,UNIT,ZERO
      DIMENSION RHS(1),T(NROWS,1),TEMP(1)
      INTEGER ROW,COL,SAFE,TOTCOL
      LOGICAL CYCLE
      SAFE=0
5     SMALL=0
      ROW=0
      SAFE=SAFE+1
      IF(SAFE.GT.NROWS)RETURN
      DO 10 I=1,NROWS
      IF(RHS(I).GE.SMALL)GO TO 10
      SMALL=RHS(I)
      ROW=I
10    CONTINUE
      IF(ROW.EQ.0)RETURN
C
C     NEGATIVE RHS DETECTED, CHECK FOR NEGATIVE IN ROW 0
C     ROWZ IS AN ENTRY POINT IN PROFIT
      CALL ROWZ(PRICE,T,TEMP,RHS,STATUS,NROWS)
      IF(.NOT.CYCLE)GO TO 15
C
C     NEGATIVE RHS AND NEGATIVE IN ROW 0, LPAFIT CANNOT HANDLE
      WRITE(6,13)
13    FORMAT(/,70H YOU HAVE CREATED A SENSITIVITY SITUATION WHICH YIELDE
     1D A NEGATIVE RHS,/,     70H AND A NEGATIVE IN ROW 0. I CANNOT DIRE
     1CTLY HANDLE THIS SITUATION. YOU,/,     71H MUST BE TERMINATED. TO
     1SOLVE THIS PROBLEM INDIRECTLY, TYPE "BEGIN,LP.",/,    64H AND USE
     1THE OPTION TO RECOVER FROM A MACHINE CRASH. MODIFY YOUR,/,
     172H ORIGINAL PROBLEM INTO THE PROBLEM WHICH MADE ME SICK. I WILL T
     1HEN SOLVE,/,       26H THE NEW PROBLEM OUTRIGHT.,/)
      STOP  "STOP IN SUBROUTINE DUALSM"
```

```
C
C     NO NEGATIVES IN ROW 0, EVERYTHING O.K.
15    SMALL=1E100
      COL=0
      DO 20 I=1,TOTCOL
      IF(T(ROW,I).GE.0)GO TO 20
      RATIO=TEMP(I)/ABS(T(ROW,I))
      IF(RATIO.GE.SMALL)GO TO 20
      SMALL=RATIO
      COL=I
20    CONTINUE
C
      IF(COL.EQ.0)RETURN
      CALL SWITCH(ROW,COL,TCOLP1,T,NROWS,IBASIS,PRICE,STATUS)
C     ROWZ IS AN ENTRY POINT IN PROFIT
      CALL ROWZ(PRICE,T,TEMP,RHS,STATUS,NROWS)
      GO TO 5
      END
```

```
      SUBROUTINE DVCOEF(NROWS,TEMP,IBASIS,T,RHS,PRICE,DVNAME,STATUS)
C     ****************************************************************
C     *  SUBROUTINE DVCOEF DETERMINES HOW MUCH THE COEFFICIENT OF EACH *
C     *  DECISION VARIABLE IN THE OBJECTIVE FUNCTION CAN CHANGE BEFORE *
C     *  A VARIABLE LEAVES THE FINAL BASIS.  THE SUBROUTINE DETERMINES *
C     *  WHAT THE ENTERING AND LEAVING VARIABLES WILL BE. NO           *
C     *  SENSITIVITY IS DONE ON COEFFICIENTS IF THE CONSTRAINTS.       *
C     *  CALLED BY POSTOP.                                            *
C     *  LIST OF VARIABLES...COMMON VARIABLES DEFINED IN INPUT2        *
C     *  EVNEG.....INTEGER. ENTERING VARIABLE IF THE DECISION VARIABLE *
C     *            COEFFICIENT FALLS BELOW THE MINIMUM VALUE.          *
C     *  EVPOS.....INTEGER. ENTERING VARIABLE IF THE DECISION VARIABLE *
C     *            COEFFICIENT EXCEEDS THE MAXIMUM VALUE               *
C     *  FROMRS....LOGICAL VARIABLE.  CALLING ARGUMENT FOR SUBROUTINE  *
C     *            OPTION.  FROMRS=.FALSE. IMPLIES THAT THE CALL IS    *
C     *            NOT COMING FROM SUBROUTINE RSIDE.                   *
C     *  LVMAX.....LEAVING VARIABLE WHEN DECISION VARIABLE COEFFICIENT *
C     *            EXCEEDS MAXIMUM VALUE.                              *
C     *  LVMIN.....LEAVING VARIABLE WHEN DECISION VARIABLE COEFFICIENT *
C     *            FALLS BELOW THE MINIMUM VALUE.                      *
C     *  NEGMAX....AMOUNT WHICH IS SUBTRACTED FROM ORIGINAL VALUE OF   *
C     *            THE COEFFICIENT TO FIND ITS MINIMUM VALUE.          *
C     *  NROWPI....POINTER TO PRICE FOR DECISION VARIABLE I            *
C     *  PMAX......MAXIMUM VALUE OF COEFFICIENT(PRICE)                 *
C     *  PMIN......MINIMUM VALUE OF COEFFICIENT(PRICE)                 *
C     *  POSMIN....AMOUNT WHICH IS ADDED TO THE ORIGINAL VALUE OF THE  *
C     *            COEFFICIENT TO FIND ITS MAXIMUM VALUE.              *
C     ****************************************************************
      COMMON BIGM,COEF,CYCLE,DRIVE,IBANER,IBOT,INDEX,ITOP,IPRINT,MAX,ME,
     1NAMES,HEAD,NBASIS,NCOLS,SEN,TITLE(6),TCOLP1,TOTAL,TOTCOL,UNIT,ZERO
      DIMENSION TEMP(1),IBASIS(1),PRICE(1),T(NROWS,1),STATUS(1)
      INTEGER EVPOS,EVNEG,ROWJ,TOTCOL,STATUS
      REAL NEGMAX
      LOGICAL FROMRS,HEAD
C
      WRITE(6,1)
1     FORMAT(1H1,79(1H*),/,
     140H * THIS OUTPUT BLOCK IS A SENSITIVITY AN,
     140HALYSIS ON THE OBJECTIVE FUNCTION       *,/,
     140H * COEFFICIENT OF EACH DECISION VARIABLE,
     140H.  THE ANALYSIS IS ONLY FOR CHANGES    *,/,
     140H * IN OBJECTIVE FUNCTION COEFFICIENTS.  ,
     140HCONSTRAINT COEFFICIENTS ARE NOT        *,/,
     140H * PERMITTED TO VARY FROM THEIR INITIAL ,
     140HVALUES.  THE ANALYSIS ON AN INDIVIDUAL *,/,
     140H * DECISION VARIABLE ASSUMES THAT ONLY I,
     140HTS COEFFICIENT VARIES AND THAT ALL     *,/,
     140H * OTHER OBJECTIVE FUNCTION COEFFICIENTS,
     140H RETAIN THEIR INITIAL VALUES.          *)
```

```
      IF(HEAD)WRITE(6,2)
2     FORMAT(29H * DESCRIPTION OF HEADINGS...,50X,1H*,/,
     140H * EV.....VARIABLE WHICH E,
     140HNTERS THE FINAL BASIS AS A RESULT OF   *,/,
     12H *,22X,14HA PRICE CHANGE,41X,1H*,/,
     140H * LV.....VARIABLE WHICH L,
     140HEAVES THE FINAL BASIS AS A RESULT OF   *,/,
     12H *,22X,14HA PRICE CHANGE,41X,1H*)
      IF(HEAD)WRITE(6,3)
3     FORMAT(40H * CURRENT PRICE.....THE INITIAL OBJE,
     140HCTIVE FUNCTION COEFFICIENT OF THE      *,/,
     12H *,22X,17HDECISION VARIABLE,38X,1H*,/,
     140H * MINIMUM PRICE.....IF THE OBJECTIVE,
     140H FUNCTION COEFFICIENT OF THE DECISION  *,/,
     12H *,22X,16HVARIABLE FALLS B,
     140HELOW THIS PRICE, THEN VARIABLE EV WOULD*,/,
     12H *,22X,16HENTER THE FINAL ,
     140HBASIS AND VARIABLE LV WOULD LEAVE THE  *,/,
     12H *,22X,11HFINAL BASIS,44X,1H*,/,
     140H * MAXIMUM PRICE.....IF THE OBJECTIVE,
     140H FUNCTION COEFFICIENT OF THE DECISION  *,/,
     12H *,22X,16HVARIABLE INCREAS,
     140HES ABOVE THIS PRICE, THEN VARIABLE EV  *,/,
     12H *,22X,16HWOULD ENTER THE ,
     140HFINAL BASIS AND VARIABLE LV WOULD      *,/,
     12H *,22X,21HLEAVE THE FINAL BASIS,34X,1H*,/,
     140H * ENTERING QUANTITY....VALUE OF THE ENT,
     118HERING VARIABLE(EV),21X,1H*)
      IF(HEAD)WRITE(6,4)
4     FORMAT(1X,79(1H*),/,
     140H * THE FOLLOWING CONCLUSIONS MAY BE DRAW,
     140HN AS THE RESULT OF CHANGING A SINGLE   *,/,
     140H * OBJECTIVE FUNCTION PRICE COEFFICIENT-,39X,1H*,/,
     140H * 1.  IF THE MAGNITUDE OF THE CHANGE IS,
     140H SUFFICIENT, THEN A VARIABLE MAY LEAVE *,/,
     123H *     THE FINAL BASIS.,56X,1H*,/,
     140H * 2.  IF A VARIABLE LEAVES THE FINAL BA,
     140HSIS, THE NEW BASIS WILL STILL BE       *,/,
     115H *     OPTIMAL.,64X,1H*,/,
     140H * 3.  WHETHER OR NOT A VARIABLE LEAVES ,
     140HTHE FINAL BASIS, THE VALUES OF THE     *,/,
     162H *     BASIC VARIABLES AND THE OBJECTIVE FUNCTION WILL CHANGE.,
     117X,1H*)
      WRITE(6,5)
5     FORMAT(1X,79(1H*),//,
     12X,8HDECISION,7X,7HCURRENT,3X,7HMINIMUM,10X,8HENTERING,3X,
     17HMAXIMUM,10X,8HENTERING,/,
     12X,8HVARIABLE,8X,5HPRICE,5X,5HPRICE,3X,16HEV  LV  QUANTITY,4X,
     124HPRICE   EV  LV  QUANTITY,/,1X,10(1H-),3X,2(1X,9(1H-)),
     12(2X,2H--),2(1X,9(1H-)),2(2X,2H--), 1X,9(1H-))
C
```

```
      FROMRS=.FALSE.
C
C     EXAMINE EACH DECISION VARIABLE
      DO 100 I=1,NCOLS
C     INITIALIZE VARIABLES
      EVNEG=0
      EVPOS=0
      POSMIN=1E100
      NEGMAX=-1E100
      NROWPI=NROWS+I
C
C     IS DECISION VARIABLE IN BASIS?
      IF(STATUS(I).LT.0)GO TO 95
C
C     ********************************************************************
C     *  DECISION VARIABLE IN BASIS, ROW J, CONDUCT SENSITIVITY          *
C     *  ANALYSIS FOR CHANGE IN COEFFICIENT OF BASIS DECISION VARIABLE   *
C     *  (HILLIER AND LIEBERMAN, P190)                                   *
C     ********************************************************************
C
C     DECISION VARIABLE I IN BASIS, FIND THE ROW
      DO 10 J=1,NROWS
      IF(IBASIS(J).NE.I)GO TO 10
      ROWJ=J
      GO TO 50
10    CONTINUE
C
C        EXAMINE NONBASIC VARIABLES FOR POTENTIAL ENTERING VARIABLES
50       DO 80 J=1,TOTCOL
C
C        INSURE THAT J IS NOT IN BASIS, IF SO IT CANNOT ENTER
         IF(STATUS(J).GT.0)GO TO 80
C
C        J IS A NONBASIC DECISION VARIABLE, IS IT A
C        CANDIDATE FOR AN ENTERING VARIABLE?
         IF(ABS(T(ROWJ,J)).LT.ZERO)GO TO 80
C
C        J IS A CANDIDATE
         DEL=-TEMP(J)/T(ROWJ,J)
         IF(DEL.GT.0)GO TO 70
C
C        DEL<0, FIND THE LEAST NEGATIVE CASE
         IF(DEL.LT.NEGMAX)GO TO 80
         NEGMAX=DEL
         EVNEG=J
         GO TO 80
C
C        DEL>0, FIND THE LEAST POSITIVE CASE
70       IF(DEL.GT.POSMIN)GO TO 80
         POSMIN=DEL
         EVPOS=J
80       CONTINUE
C
```

```
C     MAXIMIZE OR MINIMIZE CASE?
      IF(MAX.EQ.3HMAX)GO TO 90
C
C     MINIMIZE PROBLEM
C     IF NO ENTERING VARIABLE FOR MINIMUM VALUE, DON'T TRY TO FIND
C     THE LEAVING VARIABLE
      IF(EVPOS.EQ.0)GO TO 81
      PMIN=PRICE(NROWPI )-POSMIN
      CALL ROW(LVMIN,EVPOS,CYCLE,T,RHS,NROWS,ZERO,EQMIN)
C     IF LVMIN IS RETURNED=0, THERE IS NO LEAVING VARIABLE
C
C     IF NO ENTERING VARIABLE FOR THE MAXIMUM VALUE, DON'T TRY TO
C     FIND THE LEAVING VARIABLE
81    IF(EVNEG.EQ.0)GO TO 82
      PMAX=PRICE(NROWPI)-NEGMAX
      CALL ROW(LVMAX,EVNEG,CYCLE,T,RHS,NROWS,ZERO,EQMAX)
C     IF LVMAX IS RETURNED=0, THERE IS NO LEAVING VARIABLE
C
C     OUTPUT ACCORDING TO PROGRAM CONDITIONS
82    CALL OPTION(NAMES,I,DVNAME,PRICE(NROWPI),PMIN,EVPOS,IBASIS(LVMIN),
     1            EQMIN,PMAX,EVNEG,IBASIS(LVMAX),EQMAX,FROMRS)
      GO TO 100
C
C     MAXIMIZE PROBLEM
C     IF NO ENTERING VARIABLE FOR THE MAXIMUM VALUE, DON'T TRY TO
C     FIND THE LEAVING VARIABLE
90    IF(EVPOS.EQ.0)GO TO 91
      PMAX=PRICE(NROWPI)+POSMIN
      CALL ROW(LVMAX,EVPOS,CYCLE,T,RHS,NROWS,ZERO,EQMAX)
C     IF LVMAX IS RETURNED=0, THERE IS NO LEAVING VARIABLE
C
C     IF NO ENTERING VARIABLE FOR MINIMUM VALUE, DON'T TRY TO FIND
C     THE LEAVING VARIABLE
91    IF(EVNEG.EQ.0)GO TO 92
      PMIN=PRICE(NROWPI)+NEGMAX
      CALL ROW(LVMIN,EVNEG,CYCLE,T,RHS,NROWS,ZERO,EQMIN)
C
C     OUTPUT ACCORDING TO PROGRAM CONDITIONS
92    CALL OPTION(NAMES,I,DVNAME,PRICE(NROWPI),PMIN,EVNEG,IBASIS(LVMIN),
     1            EQMIN,PMAX,EVPOS,IBASIS(LVMAX),EQMAX,FROMRS)
      GO TO 100
C
```

```
C     ************************************************************************
C     *  DECISION VARIABLE I NOT IN BASIS, CONDUCT SENSITIVITY               *
C     *  ANALYSIS FOR CHANGE IN COEFFICIENT OF A NONBASIC VARIABLE           *
C     *  (HILLIER AND LIEBERMAN, P188)                                       *
C     ************************************************************************
C     FIND LEAVING VARIABLE IF I IS ENTERING VARIABLE
95    CALL ROW(LEAVE,I,CYCLE,T,RHS,NROWS,ZERO,EQUANT)
C
C     MAXIMIZE OR MINIMIZE PROBLEM?
      IF(MAX.EQ.3HMAX)GO TO 97
C
C     MINIMIZE PROBLEM,CALCULATE HOW SMALL DECISION VARIABLE COEFFICIENT
C     MUST BE BEFORE IT CAN ENTER BASIS
      PMIN=PRICE(NROWPI)-TEMP(I)
C
C     OUTPUT ACCORDING TO PROGRAM CONDITIONS
      CALL OPTION(NAMES,I,DVNAME,PRICE(NROWPI),PMIN,I,IBASIS(LEAVE),
     1            EQUANT,0.,0,0,0.,FROMRS)
      GO TO 100
C
C     MAXIMIZE PROBLEM, CALCULATE HOW LARGE DECISION VARIABLE
C     COEFFICIENT MUST BE BEFORE IT CAN ENTER BASIS
97    PMAX=PRICE(NROWPI)+TEMP(I)
C
C     OUTPUT ACCORDING TO PROGRAM CONDITIONS
      CALL OPTION(NAMES,I,DVNAME,PRICE(NROWPI),0.,0,0,0.,PMAX,I,
     1            IBASIS(LEAVE),EQUANT,FROMRS)
100   CONTINUE
C
      RETURN
      END
```

```
      SUBROUTINE INPUT1(NROWS)
C     ************************************************************************
C     *  SUBROUTINE INPUT1 READS AND VERIFIES THE FIRST PART OF USER         *
C     *  INPUT.  IT ALSO CALCULATES POINTERS TO IDENTIFY VARIOUS AREAS       *
C     *  OF ARRAY WORK.  **NOTE**  THIS PROGRAM WAS WRITTEN TO REPLACE       *
C     *  AN EXISTING LP CODE AT AFIT.  TO MAKE THE TRANSITION BETWEEN        *
C     *  CODES AS EASY AS POSSIBLE,  LPSOLVE REQUIRES LITTLE                 *
C     *  MODIFICATION OF THE ORIGINAL DATA.  INPUT VARIABLES,NAME AND        *
C     *  IDRIVE,WERE ORIGINALLY INTEGERS.  I USED THEM TO SET LOGICALS       *
C     *  WHICH I PREFER TO INTEGER TYPE FLAGS. CALLED BY LPSOLVE.            *
C     ************************************************************************
C     *  INPUT VARIABLES...                                                  *
C     *  TITLE.....ALPHANUMERIC NAME TO BE ASSOCIATED WITH THE               *
C     *            PROGRAM.  UP TO 60 CHARACTERS LONG                        *
C     *  NROWS.....NUMBER OF PROGRAM CONSTRAINTS                             *
C     *  NCOLS.....NUMBER OF PROGRAM DECISION VARIABLES(EXCLUDING            *
C     *            SLACK, ARTIFICAL AND SURPLUS VARIABLES.                   *
C     *  IPRINT....OUTPUT PRINT OPTION                                       *
C     *            IPRINT=-2  PRINT FIRST TABLEAU, EACH BASIS, LAST          *
C     *                       TABLEAU                                        *
C     *            IPRINT=-1  PRINT COMPLETE TABLEAU AT EACH ITERATION       *
C     *            IPRINT= 0  PRINT FIRST AND LAST TABLEAU ONLY              *
C     *            IPRINT=+1  PRINT EACH BASIS                               *
C     *            IPRINT=+2  PRINT LAST BASIS ONLY                          *
C     *  MAX.......ALPHA. MAX=MAX IMPLIES A MAXIMIZE PROBLEM                 *
C     *            MAX=MIN IMPLIES A MINIMIZE PROBLEM                        *
C     *  NAME......NAME=0 IMPLIES DECISION VARIABLES AND CONSTRAINT          *
C     *            NAMES ARE NOT TO BE READ. NAME NE 0 IMPLIES READ          *
C     *            NAMES.                                                    *
C     *  IDRIVE....IDRIVE=0 IMPLIES ARTIFICAL VARIABLES ARE TO BE            *
C     *            DRIVEN FROM THE INITIAL BASIS FIRST.  IDRIVE NE 0         *
C     *            IMPLIES USUAL SIMPLEX RULES WILL DETERMINE LEAVING        *
C     *            AND ENTERING VARIABLES.                                   *
C     *  NGREAT....NUMBER OF GREATER THAN CONSTRAINTS                        *
C     *  UNIT......INTEGER.  UNIT TO WHICH OUTPUT SPECIFIED BY IPRINT        *
C     *            IS TO BE WRITTEN.                                         *
C     *            UNIT=2 IMPLIES OUTPUT WILL BE ON TAPE2.                   *
C     *            UNIT=6 IMPLIES OUTPUT WILL BE ON TAPE6.                   *
C     *  HD........ALPHA. HD=F IMPLIES THAT OUTPUT HEADINGS WILL BE          *
C     *            ABBREVIATED.                                              *
C     *  SN........ALPHA. SN=T IMPLIES THAT THE USER WANTS SENSITIVITY       *
C     *            ANALYSIS.                                                 *
C     *  IBANER....ALPHA. GENERATED BY PREPROCESSOR. OUTPUT WILL BE          *
C     *            ROUTED TO PRINTER UNDER THIS BANNER IF UNIT=2.            *
C     ************************************************************************
C     *  LIST OF VARIABLES...                                                *
C     *  DRIVE.....LOGICAL VARIABLE.  DRIVE=.TRUE. IMPLIES THAT              *
C     *            ARTIFICAL VARIABLES ARE TO BE DRIVEN OUT OF THE           *
C     *            BASIS FIRST.                                              *
C     *  NAMES.....LOGICAL VARIABLE.  NAMES=.FALSE. WHEN NAME=0              *
C     *  TCOLP1....TOTAL NUMBER OF COLUMNS PLUS 1                            *
C     *  TOTCOL....TOTAL NUMBER OF COLUMNS IN THE TABLEAU                    *
C     ************************************************************************
```

```
C     ************************************************************************
C     *  WORK IS SPLIT INTO THE FOLLOWING AREAS                              *
C     *  (IN THE ORDER THAT THEY OCCUR IN WORK)                              *
C     *  NAME     LENGTH          FUNCTION                                   *
C     *  ----     ------          --------                                   *
C     *  T        TOTCOL*NROWS    CONTAINS SIMPLEX TABLEAU EXCEPT ROW 0      *
C     *  RHS      NROWS           CONTAINS CURRENT RIGHT HAND SIDE VALUES    *
C     *  ORHS     NROWS           CONTAINS ORIGINAL RIGHT HAND SIDES         *
C     *  STATUS   TOTCOL          INTEGER ARRAY WHICH CONTAINS THE STATUS    *
C     *                           OF EACH VARIABLE.  NEGATIVE VALUES         *
C     *                           INDICATE THE VARIABLE IS NOT IN THE        *
C     *                           BASIS.  POSITIVE VARIABLES ARE BASIC.      *
C     *                           +-1 INDICATES A DECISION VARIABLE          *
C     *                           +-2 INDICATES AN ARTIFICAL VARIABLE        *
C     *                           +-3 INDICATES A SLACK VARIABLE             *
C     *                           +-4 INDICATES A SURPLUS VARIABLE           *
C     *  PRICE    NROWS+TOTCOL    CONTAINS PRICES OF BASIS ELEMENTS IN       *
C     *                           FIRST NROWS.     AND ORIGINAL PRICES IN    *
C     *                           NEXT TOTCOL ELEMENTS.                      *
C     *  TEMP     TOTCOL          CONTAINS ROW 0 (Z-C)                       *
C     *  IBASIS   NROWS           CONTAINS CURRENT BASIS ELEMENTS            *
C     *  OBASIS   NROWS           CONTAINS ORIGINAL BASIS(INTEGER ARRAY)     *
C     *  CNAME    NROWS+8         ALPHA ARRAY CONTAINING CONSTRAINT NAMES.   *
C     *                           LENGTH IS ACTUALLY LESS THAN THAT STATED   *
C     *                           BUT THE LENGTH GIVEN IS EASIER TO WORK     *
C     *                           WITH.                                      *
C     *  DVNAME   NCOLS+8         ALPHA ARRAY CONTAINING DECISION VARIABLE   *
C     *                           NAMES.  AGAIN LENGTH IS MISREPRESENTED.    *
C     *  WORK IS DIMENSIONED...                                              *
C     *   (NCOLS+NROWS+NGREAT)*(3+NROWS)+5*NROWS            IF NAME EQ 0    *
C     *   (NCOLS+NROWS+NGREAT)*(3+NROWS)+6*NROWS+NCOLS+16   IF NAME NE 0    *
C     ************************************************************************
      COMMON/INDEX/IRHS,IORHS,IPRICE,ITEMP,IBASIS,IOBASIS,IT,
     1          ICNAME,IDVNAME,ISTATUS
      COMMON BIGM,COEF,CYCLE,DRIVE,IBANER,IBOT,INDEX,ITOP,IPRINT,MAX,ME,
     1NAMES,HEAD,NBASIS,NCOLS,SEN,TITLE(6),TCOLP1,TOTAL,TOTCOL,UNIT,ZERO
      INTEGER TOTCOL,TCOLP1,UNIT
      LOGICAL DRIVE,NAMES,HEAD,SEN
C
      READ(1,5)TITLE
5     FORMAT(6A10)
C
C     DEFAULT INPUTS
      NAMES=.TRUE.
      DRIVE=.FALSE.
      NGREAT=0
      HEAD=.TRUE.
      SEN=.FALSE.
C
```

```
      READ(1,10)NROWS,NCOLS,IPRINT,MAX,NAME,IDRIVE,NGREAT,UNIT,HD,SN
     1,IBANER,ME
      IF(HD.EQ.1HF)HEAD=.FALSE.
      IF(SN.EQ.1HT)SEN=.TRUE.
      IF(UNIT.NE.2)UNIT=6
      COEF=1.
      IF(MAX.EQ.3HMAX)COEF=-1.
      IF(NAME.EQ.0)NAMES=.FALSE.
      IF(IDRIVE.EQ.0)DRIVE=.TRUE.
10    FORMAT(3I2,A3,1X,2I1,I2,I1,2A1,A3,A4)
      WRITE(UNIT,15)TITLE
15    FORMAT(1H1,6A10,//,17H BIGM = 1.000E+08,//)
C
C     CALCULATE POINTERS TO VARIOUS AREAS OF WORK ARRAY
C
C     DETERMINE TOTAL COLUMNS
      TOTCOL=NCOLS+NROWS+NGREAT
      TCOLP1=TOTCOL+1
C
C     TABLEAU INDEX
      IT=1
C
C     RIGHT HAND SIDE INDEX
C     NOTE THAT IRHS IS IMMEDIATELY AFTER IT.  THIS IS IMPORTANT IN
C     SUBROUTINE SOLVE WHICH REFERENCES T(IROW,TCOLP1) WHICH IS
C     THE SAME STORAGE LOCATION AS RHS(IROW)
      IRHS=TOTCOL*NROWS+IT
C
C     ORIGINAL RIGHT HAND SIDE
      IORHS=IRHS+NROWS
C
C     STATUS INDEX
      ISTATUS=IORHS+NROWS
C
C     INDEX FOR PRICE COEFFICIENTS
      IPRICE=ISTATUS+TOTCOL
C
C     INDEX FOR ROW 0
      ITEMP=IPRICE+NROWS+TOTCOL
C
C     INDEX FOR BASIS ELEMENTS
      IBASIS=ITEMP+TOTCOL
C
C     INDEX FOR ORIGINAL BASIS
      IOBASIS=IBASIS+NROWS
C
```

```
      IF(.NOT.NAMES)GO TO 20
C
C     INDEX FOR CONSTRAINT NAMES
      ICNAME=IOBASIS+NROWS
C
C     INDEX FOR DECISION VARIABLE NAMES
C     MAKE INDEX A MULTIPLE OF 8 TO FACILITATE READING IN SUB. INPUT2
      INDEX=(NROWS/8+1)*8
      IF(MOD(NROWS,8).EQ.0)INDEX=INDEX-8
      IDVNAME=ICNAME+INDEX
      RETURN
C
C     NO NAMES TO BE READ - SET NAME INDICES EQUAL TO 1 TO AVOID
C     MODE ERROR
20    ICNAME=1
      IDVNAME=1
      RETURN
      END
```

```
      SUBROUTINE INPUT2(T,RHS,ORHS,PRICE,TEMP,IBASIS,OBASIS,CNAME,
     1                  DVNAME,STATUS,NROWS)
C     ***************************************************************
C     *  SUBROUTINE INPUT2 READS AND VERIFIES REMAINING USER INPUT.    *
C     *  CALLED BY LPSOLVE.                                           *
C     *  COMMON BLOCK VARIABLES...                                    *
C     *  BIGM......THE PROGRAM SUPPLIED PRICE COEFFICIENT OF          *
C     *            ARTIFICAL VARIABLES WHICH IS SO LARGE THAT IT      *
C     *            DRIVES THEM OUT OF THE INITIAL BASIS.  AT THE SAME *
C     *            TIME, IT MUST BE SMALL ENOUGH THAT IT DOES NOT     *
C     *            OBSCURE SMALL NUMBERS WHICH ARE ADDED TO IT.       *
C     *            THEREFORE, THE MAGNITUDE OF BIGM IS DEPENDENT ON   *
C     *            MACHINE WORD LENGTH.  THE CDC 6600 WITH 60 BIT WORDS*
C     *            ALLOWS 14 SIGNIFICANT FIGURES.  INPUTS TO LPAFIT   *
C     *            ARE LIMITED TO THE RANGE(1E-5,1E6) SO A BIGM VALUE *
C     *            OF 1E8 WILL MEET NECESSARY REQUIREMENTS.           *
C     *  CYCLE.....LOGICAL VARIABLE.  CYCLE=.TRUE. IMPLIES THAT       *
C     *            ANOTHER PROGRAM ITERATION IS TO OCCUR.             *
C     *            CYCLE=.FALSE. IMPLIES THAT THE PROGRAM IS TO       *
C     *            TERMINATE BECAUSE THE OPTIMAL SOLUTION HAS BEEN    *
C     *            FOUND OR AN ERROR HAS BEEN DETECTED.               *
C     *  DRIVE.....LOGICAL VARIABLE.  IF DRIVE=.TRUE. ARTIFICAL       *
C     *            VARIABLES ARE TO BE DRIVEN FROM THE INITIAL BASIS  *
C     *            FIRST.                                             *
C     *  IBOT......CALCULATED LOWER INDEX FOR LOOPS                   *
C     *            FIRST.                                             *
C     *  IBANER....ALPHA. GENERATED BY PREPROCESSOR. OUTPUT WILL BE   *
C     *            ROUTED TO PRINTER UNDER THIS BANNER IF UNIT=2.     *
C     *  IBOT......CALCULATED LOWER INDEX FOR LOOPS                   *
C     *  INDEX.....CALCULATED ARRAY ADDRESS                           *
C     *  ITOP......CALCULATED UPPER INDEX FOR A LOOP                  *
C     *  IPRINT....OUTPUT OPTION-SEE INPUT1 FOR ADDITIONAL INFORMATION *
C     *  MAX.......IDENTIFIES WHETHER THE PROBLEM IS A MAXIMIZE OR    *
C     *            MINIMIZE                                           *
C     *  ME........ALPHA. GENERATED BY PREPROCESSOR. IF ME=HERE,      *
C     *            PROGRAM WILL KNOW THAT THE USER IS INTERACTIVE     *
C     *  NAMES.....LOGICAL VARIABLE.  NAMES=.TRUE. IMPLIES THAT       *
C     *            DECISION VARIABLES AND CONSTRAINTS HAVE NAMES.     *
C     *  NBASIS....BASIS NUMBER(ITERATION NUMBER)                     *
C     *  NCOLS.....NUMBER OF DECISION VARIABLES                       *
C     *  TITLE.....ALPHA ARRAY CONTAINING PROMLEM TITLE               *
C     *  TCOLP1....TOTCOL+1                                           *
C     *  TOTAL.....TOTAL VALUE OF OBJECTIVE FUNCTION BASED ON         *
C     *            CURRENT BASIS AND RIGHT HAND SIDE                  *
C     *  TOTCOL....TOTAL NUMBER OF VARIABLES=DECISION+SLACK+          *
C     *            ARTIFICAL+SURPLUS                                  *
C     *  ZERO......DUE TO IMPRECISE ARITHEMETIC, NUMBERS WHICH SOULD  *
C     *            BE EXACTLY 0 ARE NOT.  IF ABS(NUMBER)<0, THEN      *
C     *            NUMBER WILL BE ASSUMED TO BE 0. ZERO=1E-6 IN LPAFIT.*
C     ***************************************************************
```

```
C     *  INPUT VARIABLES...                                              *
C     *  IROW......ARRAY USED IN THE INPUT OF DECISION VARIABLE          *
C     *            COEFFICIENTS.  IDENTIFIES THE OBJECTIVE FUNCTION      *
C     *            OR CONSTRAINT NUMBER.                                 *
C     *  ICOL......ARRAY USED IN THE INPUT OF DECISION VARIABLE          *
C     *            COEFFICIENTS.  IDENTIFIES THE DECISION VARIABLE.      *
C     *  ASUBIJ....ARRAY USED IN THE INPUT OF DECISION VARIABLE          *
C     *            COEFFICIENTS.  CONTAINS THE COEFFICIENT IDENTIFIED    *
C     *            BY (IROW,ICOL)                                        *
C     *  IRELATE...ARRAY USED IN THE INPUT OF RIGHT HAND SIDES.          *
C     *            IDENTIFIES THE RELATIONSHIP BETWEEN CONSTRAINT        *
C     *            AND RIGHT HAND SIDE (LT,ET,GT)                        *
C     *  RSOURCE...ARRAY USED IN THE INPUT OF RIGHT HAND SIDES.          *
C     *            CONTAINS THE RIGHT HAND SIDE.                         *
C     *  CNAME.....ARRAY WHICH CONTAINS CONSRTAINT NAMES(OPTIONAL)       *
C     *  DVNAME....ARRAY WHICH CONTAINT DECISION VARIABLE NAMES(ALSO     *
C     *            OPTIONAL, HOWEVER, IF EITHER CNAME OR DVNAME IS       *
C     *            INPUT, SO MUST BE THE OTHER).                         *
C     ******************************************************************
C     *  LIST OF VARIABLES...                                            *
C     *  ERROR.....LOGICAL VARIABLE.  ERROR=.TRUE. IMPLIES THAT AN       *
C     *            INPUT ERROR HAS BEEN MADE.  PROGRAM WILL BE STOPPED.*
C     *  TEMCOL....INTEGER.  TEMPORARY COLUMN COUNTER.  INCREMENTED      *
C     *            EACH TIME A SLACK, ARTIFICAL OR SURPLUS VARIABLE      *
C     *            IS ADDED BY THE PROGRAM.                              *
C     ******************************************************************
      DIMENSION IROW(7),ICOL(7),ASUBIJ(7),RSOURCE(7),IRELATE(7)
      DIMENSION RHS(1),ORHS(1),T(NROWS,1),PRICE(1),TEMP(1),IBASIS(1),
     1          OBASIS(1),CNAME(1),DVNAME(1),STATUS(1)
      COMMON BIGM,COEF,CYCLE,DRIVE,IBANER,IBOT,INDEX,ITOP,IPRINT,MAX,ME,
     1NAMES,HEAD,NBASIS,NCOLS,SEN,TITLE(6),TCOLP1,TOTAL,TOTCOL,UNIT,ZERO
      INTEGER TOTCOL,TEMCOL,OBASIS,STATUS
      LOGICAL ERROR,NAMES
      EQUIVALENCE(ASUBIJ,RSOURCE),(ICOL,IRELATE)
C
      ERROR=.FALSE.
C
C     INITIALIZE STATUS FOR DECISION VARIABLES(NOT IN INITIAL BASIS)
      DO 2 I=1,NCOLS
2     STATUS(I)=-1
C
C     SET ZERO AND BIGM
      ZERO=1E-6
      BIGM=1E8
C
```

```
C     ***********************************************
C     *  READ CONSTRAINTS AND OBJECTIVE FUNCTION  *
C     ***********************************************
3     READ(1,5)(IROW(I),ICOL(I),ASUBIJ(I),I=1,7)
5     FORMAT(7(2I2,F6.0))
C
C     ERROR CHECK
      DO 25 I=1,7
C     CHECK COLUMN SUBSCRIPTS
      IF(ICOL(I))30,25,10
C
C     POSITIVE COLUMN #, MAKE SURE NOT TOO LARGE
10    IF(ICOL(I).LE.NCOLS)GO TO 20
C
C     ERROR
11    WRITE(6,12)NROWS,NCOLS,IROW(I),ICOL(I),ASUBIJ(I)
12    FORMAT(1X,19HNO. OF CONSTRAINTS=,I3,/,
     1       1X,26HNO. OF DECISION VARIABLES=,I3,/,
     1       1X,38HYOUR CONSTRAINT OR OBJECTIVE FUNCTION#,I4,/,
     1       1X,24HYOUR DECISION VARIABLE #,I4,/,
     1       1X,30HDECISION VARIABLE COEFFICIENT=,F7.0)
      ERROR=.TRUE.
      GO TO 25
C
C     CHECK ROW SUBSCRIPTS
20    IF(IROW(I))30,21,23
C
C     ROW=0, STORE PRICE COEFFICIENTS
21    INDEX=ICOL(I)+NROWS
      PRICE(INDEX)=ASUBIJ(I)
      GO TO 25
C
C     POSITIVE ROW #, MAKE SURE NOT TOO LARGE
23    IF(IROW(I).GT.NROWS) GO TO 11
C
C     ROW AND COLUMN SUBSCRIPTS O.K.
      T(IROW(I),ICOL(I))=ASUBIJ(I)
C
25    CONTINUE
C     READ ANOTHER CARD
      GO TO 3
C
```

```
C     ************************************************************************
C     *  NEGATIVE COLUMN NUMBER, ALL OBJECTIVE FUNCTION AND CONSTRAINT *
C     *  COEFFICIENT CARDS READ.  IF MAXIMIZE PROBLEM, CHANGE SIGNS OF *
C     *  PRICE COEFFICIENTS AND CHECK VALUE OF MAX.                    *
C     ************************************************************************
30    IF(MAX.EQ.3HMAX)GO TO 40
      IF(MAX.EQ.3HMIN)GO TO 50
C
C     ERROR
      WRITE(6,35)MAX
35    FORMAT(1X,38HERROR IN INPUT VALUE OF VARIABLE MAX= ,A3)
      ERROR=.TRUE.
      GO TO 50
C
C     MAXIMIZE PROBLEM
40    DO 45 I=1,TOTCOL
      INDEX=NROWS+I
45    PRICE(INDEX)=-PRICE(INDEX)
C
C     ************************************************************************
C     *  READ CONSTRAINT#(IROW), RELATIONSHIP TO RESOURCE(IRELATE),    *
C     *  AND RIGHT HAND SIDE(RSOURCE)                                  *
C     ************************************************************************
C     TEMCOL IS THE TEMPORARY # OF COLUMNS- IT IS INCREASED
C     WHENEVER A SLACK,ARTIFICAL, OR SURPLUS VARIABLE IS ADDED
50    TEMCOL=NCOLS
C
51    READ(1,55)(IROW(I),IRELATE(I),RSOURCE(I),I=1,7)
55    FORMAT(7(I2,A2,F6.0))
C     CHECK ROW NUMBERS
      DO 75 I=1,7
      IF(IROW(I))80,75,60
C
C     ROW # POSITIVE, CHECK RELATIONSHIP TO RESOURCE
60    IF(IROW(I).GT.NROWS)GO TO 70
C     MAKE SURE RIGHT HAND SIDE IS NOT NEGATIVE
      IF(RSOURCE(I).LT.0.)GO TO 70
      IF(IRELATE(I).EQ.2HLT)GO TO 61
      IF(IRELATE(I).EQ.2HGT)GO TO 62
      IF(IRELATE(I).EQ.2HET)GO TO 63
      GO TO 70
C
C     LESS THAN- ADD A SLACK VARIABLE
C     COEFFICIENT OF ADDED SLACK VARIABLE IN ROW 0 = 0.
61    CALL ADDCOL(IROW(I), 0.,1.,RSOURCE(I),RHS,ORHS,T,PRICE,IBASIS,
     1            OBASIS,TEMCOL,NROWS)
      STATUS(TEMCOL)=3
      GO TO 75
C
```

```
C     GREATER THAN- ADD AN ARTIFICAL, SUBTRACT A SURPLUS
62    CALL ADDCOL(IROW(I),BIGM,1.,RSOURCE(I),RHS,ORHS,T,PRICE,IBASIS,
     1            OBASIS,TEMCOL,NROWS)
      STATUS(TEMCOL)=2
      CALL ADDCOL(IROW(I),0.,-1.,RSOURCE(I),RHS,ORHS,T,PRICE,IBASIS,
     1            OBASIS,TEMCOL,NROWS)
      STATUS(TEMCOL)=-4
      GO TO 75
C
C     EQUAL TO - ADD AN ARTIFICAL
63    CALL ADDCOL(IROW(I),BIGM,1.,RSOURCE(I),RHS,ORHS,T,PRICE,IBASIS,
     1            OBASIS,TEMCOL,NROWS)
      STATUS(TEMCOL)=2
      GO TO 75
C
C     ERROR, BAD CONSTRAINT#, RELATIONSHIP, OR RIGHT HAND SIDE
70    WRITE(6,71) IROW(1),IRELATE(I),RSOURCE(I)
71    FORMAT(1X,18HERROR--CONSTRAINT=,I3,
     114H RELATIONSHIP=,A2,17H RIGHT HAND SIDE= ,F14.6)
      ERROR=.TRUE.
C
75    CONTINUE
C     READ ADDITIONAL CARDS
      GO TO 51
C
C     ********************************************************************
C     *  ROW NUMBER NEGATIVE, THEREFORE ALL RESOURCE CARDS READ.         *
C     *  CONTINUE IF NO ERRORS HAVE BEEN MADE IN INPUT.  READ NAMES      *
C     *  IF NECESSARY.                                                   *
C     ********************************************************************
80    IF(.NOT.ERROR) GO TO 85
      WRITE(6,81)
81    FORMAT(1X,39HPROGRAM TERMINATED DUE TO INPUT ERRORS.)
      STOP1
C
C     READ LABLES IF NECESSARY
85    IF(.NOT.NAMES)RETURN
C
C     CALCULATE # OF CONSTRAINT CARDS TO READ(8 LABLES PER CARD,8A8)
      NCARDS=NROWS/8.+.95
C
C     READ CONSTRAINT LABLES
      ITOP=8*NCARDS
      READ(1,90)(CNAME(I),I=1,ITOP)
90    FORMAT(8A8)
C
C     CALCULATE # OF DECISION VARIABLE CARDS TO READ(8 LABLES PER CARD)
      NCARDS=NCOLS/8.+.95
      ITOP=8*NCARDS
C     READ DECISION VARIABLE CARDS
      READ(1,90)(DVNAME(I),I=1,ITOP)
      RETURN
      END
```

```
      SUBROUTINE ITERATE(TEMP,T,RHS,NROWS,IBASIS,PRICE,STATUS)
C     ************************************************************************
C     *  SUBROUTINE ITERATE DOES THE ACTUAL SOLUTION OF THE LP PROBLEM.*
C     *  IT INITIATES THE SEARCH FOR ENTERING AND LEAVING VARIABLES    *
C     *  AND DETERMINES WHEN AN OPTIMAL SOLUTION HAS BEEN FOUND.       *
C     *  CALLED BY SOLVE.                                              *
C     *  LIST OF VARIABLES...                                          *
C     *  SMALL.....VALUE OF MOST NEGATIVE ELEMENT IN ROW 0             *
C     *  ENTER.....INTEGER POINTER TO COLUMN OF ENTERING VARIABLE.     *
C     *  LEAVE.....LEAVING ROW FOUND BY SUBROUTINE ROW                 *
C     ************************************************************************
      COMMON BIGM,COEF,CYCLE,DRIVE,IBANER,IBOT,INDEX,ITOP,IPRINT,MAX,ME,
     1NAMES,HEAD,NBASIS,NCOLS,SEN,TITLE(6),TCOLP1,TOTAL,TOTCOL,UNIT,ZERO
      DIMENSION TEMP(1),IBASIS(1),STATUS(1)
      LOGICAL CYCLE
      INTEGER TOTCOL,ENTER,STATUS
C
      ENTER=0
      SMALL=0.
C
C     FIND MOST NEGATIVE IN ROW 0
      DO 5 I=1,TOTCOL
C
C     DO NOT CHECK BASIS ELEMENTS
      IF(STATUS(I).GT.0)GO TO 5
      IF(TEMP(I).GE.SMALL)GO TO 5
C
C     NEGATIVE FOUND
      SMALL=TEMP(I)
      ENTER=I
5     CONTINUE
C
C     TERMINATE IF NO ENTERING VARIABLES
      IF(ENTER.EQ.0)GO TO 10
C
C     ENTERING VARIABLE FOUND, DETERMINE LEAVING VARIABLE ROW
      CALL ROW(LEAVE,ENTER,CYCLE,T,RHS,NROWS,ZERO,DUMMY)
      IF(.NOT.CYCLE)RETURN
C
C     CONSTRUCT NEW BASIS BY PLACING ENTERING VARIABLE IN BASIS
C     AND REMOVING LEAVING VARIABLE
      CALL SWITCH(LEAVE,ENTER,TCOLP1,T,NROWS,IBASIS,PRICE,STATUS)
      RETURN
C
C     SOLUTION FOUND, NOTHING NEGATIVE IN ROW 0
10    CYCLE=.FALSE.
      RETURN
      END
```

```
      SUBROUTINE OPTION(NAMES,DVNUM,DVNAME,PRICE,PMIN,EVMIN,LVMIN,EQMIN,
     1                  PMAX,EVMAX,LVMAX,EQMAX,FROMRS)
C     ********************************************************************
C     *  SUBROUTINE OPTION CONTROLS THE OUTPUT FOR RSIDE AND DVCOEF      *
C     *  LIST OF VARIABLES...                                            *
C     *  EVMAX.....INTEGER.  ENTERING VARIABLE FOR MAXIMUM VALUE         *
C     *  EVMIN.....INTEGER.  ENTERING VARIABLE FOR MINIMUM VALUE         *
C     *  FROMRS....LOGICAL VARIABLE.  FROMRS=.FALSE. IMPLIES THAT THE    *
C     *            CALL TO OPTION IS NOT COMING FROM RSIDE.              *
C     ********************************************************************
      DIMENSION DVNAME(1)
      INTEGER DVNUM,EVMIN,EVMAX
      LOGICAL FROMRS,NAMES
C
      IF(.NOT.NAMES)GO TO 22
C
C     ***********************************
C     *  DECISION VARIABLES HAVE NAMES  *
C     ***********************************
      IF(EVMIN.NE.0)GO TO 10
      IF(EVMAX.NE.0)GO TO 18
C
C     EVMIN=0, EVMAX=0
      IF(FROMRS)GO TO 6
      WRITE(6, 5)DVNUM,DVNAME(DVNUM),PRICE
5     FORMAT(1X,I3,1X,A8,1PE11.3,
     148H THIS VARIABLE CANNOT CHANGE STATUS AT ANY PRICE)
      RETURN
6     WRITE(6,7)DVNAME(DVNUM),PRICE
7     FORMAT(2X,A8,1PE12.3)
      RETURN
C
C     EVMIN NE 0
10    IF(EVMAX.EQ.0)GO TO 14
C
C     EVMIN NE 0,  EVMAX NE 0
      IF(FROMRS)GO TO 12
      WRITE(6,11)DVNUM,DVNAME(DVNUM),PRICE,PMIN,EVMIN,LVMIN,EQMIN,
     1           PMAX,EVMAX,LVMAX,EQMAX
11    FORMAT(1X,I3,1X,A8,1PE11.3,1PE10.3,2I4,2E10.3,2I4,E10.3)
      RETURN
12    WRITE(6,13)DVNAME(DVNUM),PRICE,PMIN,LVMIN,PMAX,LVMAX
13    FORMAT(2X,A8,2(1PE12.3),I4,E15.3,I4)
      RETURN
C
C     EVMIN NE 0,  EVMAX=0
14    IF(FROMRS)GO TO 16
      WRITE(6,15)DVNUM,DVNAME(DVNUM),PRICE,PMIN,EVMIN,LVMIN,EQMIN
15    FORMAT(1X,I3,1X,A8,1PE11.3,E10.3,2I4,E10.3)
      RETURN
16    WRITE(6,17)DVNAME(DVNUM),PRICE,PMIN,LVMIN
17    FORMAT(2X,A8,2(1PE12.3),I4)
      RETURN
C
```

```
C     EVMIN=0,  EVMAX NE 0
18    IF(FROMRS)GO TO 20
      WRITE(6,19)DVNUM,DVNAME(DVNUM),PRICE,PMAX,EVMAX,LVMAX,EQMAX
19    FORMAT(1X,I3,1X,A8,1PE11.3,28X,E10.3,2I4,E10.3)
      RETURN
20    WRITE(6,21)DVNAME(DVNUM),PRICE,PMAX,LVMAX
21    FORMAT(2X,A8,1PE12.3,16X,E15.3,I4)
      RETURN
C     ************************************
C     *  NO NAMES FOR DECISION VARIABLES  *
C     ************************************
22    IF(EVMIN.NE.0)GO TO 28
      IF(EVMAX.NE.0)GO TO 39
C     EVMIN=0, EVMAX=0
      IF(FROMRS)GO TO 26
      WRITE(6,25)DVNUM,PRICE
25    FORMAT(1X,I3,9X,1PE11.3,
     148H THIS VARIABLE CANNOT CHANGE STATUS AT ANY PRICE)
      RETURN
26    WRITE(6,27)DVNUM,PRICE
27    FORMAT(2X,I8,1PE12.3)
      RETURN
C
C     EVMIN NE 0
28    IF(EVMAX.EQ.0)GO TO 35
C
C     EVMIN NE 0,  EVMAX NE 0
      IF(FROMRS)GO TO 33
      WRITE(6,32)DVNUM,PRICE,PMIN,EVMIN,LVMIN,EQMIN,
     1           PMAX,EVMAX,LVMAX,EQMAX
32    FORMAT(1X,I3,9X, 1PE11.3,E10.3,2I4,2E10.3,2I4,E10.3)
      RETURN
33    WRITE(6,34)DVNUM,PRICE,PMIN,LVMIN,PMAX,LVMAX
34    FORMAT(2X,I8,2(1PE12.3),I4,E15.3,I4)
      RETURN
C
C     EVMIN NE 0,  EVMAX=0
35    IF(FROMRS)GO TO 37
      WRITE(6,36)DVNUM,PRICE,PMIN,EVMIN,LVMIN,EQMIN
36    FORMAT(1X,I3,9X, 1PE11.3,E10.3,2I4,E10.3)
      RETURN
37    WRITE(6,38)DVNUM,PRICE,PMIN,LVMIN
38    FORMAT(2X,I8,2(1PE12.3),I4)
      RETURN
C
C     EVMIN=0,  EVMAX NE 0
39    IF(FROMRS)GO TO 41
      WRITE(6,40)DVNUM,PRICE,PMAX,EVMAX,LVMAX,EQMAX
40    FORMAT(1X,I3,9X, 1PE11.3,28X,E10.3,2I4,E10.3)
      RETURN
41    WRITE(6,42)DVNUM,PRICE,PMAX,LVMAX
42    FORMAT(2X,I8,1PE12.3,16X,E15.3,I4)
      RETURN
      END
```

```
      SUBROUTINE OUTPUT(NROWS,PRICE,IBASIS,T,TEMP,RHS,STATUS)
C     ****************************************************************
C     *  THIS SUBROUTINE OUTPUTS IN ACCORDANCE WITH INPUT OPTION     *
C     *  IPRINT.  IT ALSO EXAMINES THE FINAL SOLUTION FOR            *
C     *  IRREGULARITIES.  CALLED BY SOLVE.                           *
C     ****************************************************************
C
      COMMON BIGM,COEF,CYCLE,DRIVE,IBANER,IBOT,INDEX,ITOP,IPRINT,MAX,ME,
     1NAMES,HEAD,NBASIS,NCOLS,SEN,TITLE(6),TCOLP1,TOTAL,TOTCOL,UNIT,ZERO
      DIMENSION RHS(1),IBASIS(1),TEMP(1),PRICE(1),STATUS(1)
      LOGICAL CYCLE
      INTEGER TOTCOL,STATUS
C
      NEWPRN=IPRINT+3
      GO TO (5,10,15,20,25)NEWPRN
C
C     IPRINT=-2 PRINT FIRST MATRIX, EACH BASIS, LAST MATRIX
5     IF(NBASIS.EQ.0)GO TO 8
      IF(.NOT.CYCLE)GO TO 8
      CALL PBASIS(NROWS,PRICE,IBASIS,RHS)
      RETURN
8     CALL PMATRIX(NROWS,PRICE,IBASIS,T,TEMP RHS)
      GO TO 30
C
C     IPRINT=-1 PRINT COMPLETE MATRIX AT EACH ITERATION
10    CALL PMATRIX(NROWS,PRICE,IBASIS,T,TEMP,RHS)
      GO TO 30
C     IPRINT=0  PRINT FIRST AND LAST MATRIX ONLY
15    IF(.NOT.CYCLE)GO TO 17
      IF(NBASIS.NE.0) RETURN
17    CALL PMATRIX(NROWS,PRICE,IBASIS,T,TEMP,RHS)
      GO TO 30
C
C     IPRINT=1  PRINT EACH BASIS
20    CALL PBASIS(NROWS,PRICE,IBASIS,RHS)
      GO TO 30
C
C     IPRINT=2  PRINT LAST BASIS ONLY
25    IF(CYCLE)RETURN
      CALL PBASIS(NROWS,PRICE,IBASIS,RHS)
30    IF(CYCLE)RETURN
C
```

```
C     ****************************************
C     *  FINAL SOLUTION FOUND, EXAMINE IT  *
C     ****************************************
C
C     EXAMINE RIGHT HAND SIDE
      DO 35 I=1,NROWS
      IF(RHS(I).GE.ZERO)GO TO 35
      IF(RHS(I).LT.-ZERO)GO TO 33
C
C     RHS(I)=0, DEGENERATE SOLUTION
      WRITE(6,32)I,IBASIS(I)
32    FORMAT(28H DEGENERATE SOLUTION IN ROW ,I3,15H BASIS VARIABLE,
     1I3,11H EQUAL TO 0)
      GO TO 35
C
C     RHS(I)<0,INFEASIBLE SOLUTION
33    WRITE(6,34)I,IBASIS(I)
34    FORMAT(27H INFEASIBLE SOLUTION IN ROW, I3,15H BASIS VARIABLE,I3,
     1 9H NEGATIVE)
35    CONTINUE
C
C     EXAMINE NON BASIC VARIABLE COEFFICIENTS IN ROW 0
      DO 50 I=1,TOTCOL
C     IF I IS A BASIC VARIABLE, EXAMINE NEXT I
      IF(STATUS(I).GT.0)GO TO 50
C
C     IF COEFFICIENT=0 FOR NONBASIC I, EXAMINE NEXT I
      IF(ABS(TEMP(I)).GE.ZERO)GO TO 50
      WRITE(6,45)
45    FORMAT(65H NONBASIC VARIABLE WITH 0 COEFFICIENT IN ROW 0. MULTIPLE
     1 SOLUTION)
50    CONTINUE
C
C     CHECK FOR ARTIFICALS IN BASIS
      DO 60 I=1,NROWS
      IF(STATUS(I).NE.2)GO TO 60
C     ARTIFICAL IN BASIS
      WRITE(6,55)I
55    FORMAT(24H ARTIFICAL IN BASIS, ROW,I3,52H THEREFORE, NO FEASIBLE S
     2OLUTION TO ORIGINAL PROBLEM)
60    CONTINUE
C
      RETURN
      END
```

```
      SUBROUTINE PBASIS(NROWS,PRICE,IBASIS,RHS)
C     ******************************************************************
C     *  SUBROUTINE PBASIS PRINTS THE CURRENT BASIS AND RIGHT HAND     *
C     *  SIDES. CALLED BY OUTPUT.                                      *
C     *  LIST OF VARIABLES...                                          *
C     *  TVALUE....CONTRIBUTION OF A PARTICULAR BASIS ELEMENT TO THE   *
C     *            VALUE OF THE OBJECTIVE FUNCTION.                    *
C     ******************************************************************
      COMMON BIGM,COEF,CYCLE,DRIVE,IBANER,IBOT,INDEX,ITOP,IPRINT,MAX,ME,
     1NAMES,HEAD,NBASIS,NCOLS,SEN,TITLE(6),TCOLP1,TOTAL,TOTCOL,UNIT,ZERO
      DIMENSION RHS(1),IBASIS(1),PRICE(1)
      INTEGER UNIT
C
      WRITE(UNIT,5)NBASIS
5     FORMAT(17H ITERATION NUMBER,I4,/,19H   BASIC    CURRENT,17X,
     15HTOTAL,/,41H VARIABLES   PRICE     QUANTITY     PRICE,/,3X,
     13(1H-),5X,9(1H-),3X,9(1H-),3X,9(1H-))
C
      DO 10 I=1,NROWS
      TVALUE=COEF*PRICE(I)*RHS(I)
      POUT=COEF*PRICE(I)
10    WRITE(UNIT,15)IBASIS(I),POUT     ,RHS(I),TVALUE
15    FORMAT(I6,1PE14.3,2(E12.3))
C
      TOUT=COEF*TOTAL
      WRITE(UNIT,20)TOUT
20    FORMAT(/,17X,15HTOTAL COST IS $,1PE12.3,///)
      RETURN
      END
```

```
      SUBROUTINE PMATRIX(NROWS,PRICE,IBASIS,T,TEMP,RHS)
C     ******************************************************************
C     *  SUBROUTINE PMATRIX PRINTS THE ENTIRE TABLEAU, ROW 0, AND RIGHT*
C     *  HAND SIDE VALUES.  USUALLY THE TABLEAU WILL BE TOO LARGE TO   *
C     *  PRINT ACROSS A SINGLE PAGE.  THIS SUBROUTINE SPLITS IT UP     *
C     *  INTO PARTS WHICH ARE SMALL ENOUGH TO FIT ON A PAGE.           *
C     *  CALLED BY OUTPUT.                                             *
C     *  LIST OF VARIABLES...                                          *
C     *  PCOL......INTEGER NUMBER OF TABLEAU COLUMNS, EACH 10          *
C     *            CHARACTERS WIDE, THAT ARE PRINTED ACROSS EACH PAGE. *
C     *  QUIT......LOGICAL VARIABLE. IF QUIT=.TRUE. THE LAST PART OF   *
C     *            THE TABLEAU IS BEING PRINTED.                       *
C     *  PART......INTEGER. CURRENT PART OF THE TABLEAU WHICH IS       *
C     *            BEING PRINTED.                                      *
C     *  NPARTS....TOTAL NUMBER OF PARTS INTO WHICH THE TABLEAU IS     *
C     *            SPLIT IN ORDER TO PRINT IT.                         *
C     *  IADD......NUMBER OF COLUMNS TO BE PRINTED IN THE CURRENT      *
C     *            PART MINUS 1.                                       *
C     *  IBOT......POINTER TO THE FIRST TABLEAU COLUMN TO PRINT IN THE *
C     *            CURRENT PART.                                       *
C     *  ITOP......POINTER TO THE LAST TABLEAU TO PRINT IN THE         *
C     *            CURRENT PART.                                       *
C     ******************************************************************
      DIMENSION T(NROWS,1),IBASIS(1),TEMP(1),PRICE(1),RHS(1)
      COMMON BIGM,COEF,CYCLE,DRIVE,IBANER,IBOT,INDEX,ITOP,IPRINT,MAX,ME,
     1NAMES,HEAD,NBASIS,NCOLS,SEN,TITLE(6),TCOLP1,TOTAL,TOTCOL,UNIT,ZERO
      LOGICAL QUIT
      INTEGER PCOL,PART,      TOTCOL,TCOLP1,PCOLM1,UNIT
      DATA PCOL/9/,PCOLM1/8/
C
      IF(UNIT.NE.6)GO TO 3
      PCOL=5
      PCOLM1=4
3     QUIT=.FALSE.
C     PRINT THE MATRIX FOR THIS ITERATION
      WRITE(UNIT,5)NBASIS
5     FORMAT(1H1,16HITERATION NUMBER,I4)
C
      PART=0
      IBOT=1
      NPARTS=TCOLP1/PCOL
      IADD=PCOLM1
C
```

```
      IF(NPARTS.EQ.0)GO TO 30
10    DO 25 I=1,NPARTS
      ITOP=IBOT+IADD
      WRITE(UNIT,13)(K,K=IBOT,ITOP)
13    FORMAT(10X,8HVARIABLE,I7,8I12)
      TOUT=COEF*TOTAL
      WRITE(UNIT,14)TOUT ,(TEMP(K),K=IBOT,ITOP)
14    FORMAT(/,1X,17HRIGHT SIDE  BASIS,/,1X,1PE10.3,4X,1HZ,E13.3,8E12.3)
      DO 20 J=1,NROWS
20    WRITE(UNIT,21)   RHS(J),IBASIS(J),(T(J,K),K=IBOT,ITOP)
21    FORMAT(1X,1PE10.3,I5,E13.3,8E12.3)
      IBOT=ITOP+1
      PART=PART+1
      IF(QUIT)GO TO 40
      IF(I.EQ.NPARTS)GO TO 30
      WRITE(UNIT,24)PART
24    FORMAT(//,24H TABLEAU CONTINUED PART ,I4)
25    CONTINUE
C
C     PRINT LAST PART
30    QUIT=.TRUE.
      NPARTS=1
      IADD=MOD(TOTCOL,PCOL)-1
      IF(IADD.LT.0)GO TO 40
      IF(IADD.LT.0)GO TO 40
      WRITE(UNIT,24)PART
      GO TO 10
40    CONTINUE
C
      RETURN
      END
```

```
      SUBROUTINE POSTOP(T,RHS,ORHS,STATUS,PRICE,TEMP,IBASIS,OBASIS,
     1                  CNAME,DVNAME,NROWS)
C     ***********************************************************************
C     *  SUBROUTINE POSTOP CONTROLS PRINTING FOR THE FINAL SOLUTION     *
C     *  AND FOR SENSITIVITY ANALYSES. CALLED BY LPSOLVE.               *
C     ***********************************************************************
      COMMON BIGM,COEF,CYCLE,DRIVE,IBANER,IBOT,INDEX,ITOP,IPRINT,MAX,ME,
     1NAMES,HEAD,NBASIS,NCOLS,SEN,TITLE(6),TCOLP1,TOTAL,TOTCOL,UNIT,ZERO
      DIMENSION PRICE(1)
      INTEGER TOTCOL,UNIT
C
C     CHANGE SIGN OF OBJECTIVE FUNCTION COEFFICIENTS IF NECESSARY
      IF(MAX.NE.3HMAX)GO TO 5
      DO 3 I=1,TOTCOL
      INDEX=NROWS+I
3     PRICE(INDEX)=-PRICE(INDEX)
5     TOTAL=COEF*TOTAL
C
C     OUTPUT SOLUTION GIVING VALUE OF EACH DECISION VARIABLE
C     AND TOTAL PRICE
      CALL ANSWER(NROWS,IBASIS,DVNAME,PRICE,TEMP,RHS,STATUS)
C
C     PRINT SURPLUS IN CONSTRAINTS AND SHADOW PRICES OF SLACK VARIABLES
      CALL SLACK(OBASIS,PRICE,TEMP,IBASIS,RHS,NROWS,CNAME,STATUS)
      IF(ME.NE.4HHERE)GO TO 10
6     WRITE(6,7)
7     FORMAT(/,53H DO YOU WANT AUTOMATIC SENSITIVITY ANALYSIS? (Y,N)...)
      CALL YESNO,RETURNS(10,20,6)
C
C     DO SENSITIVITY FOR DECISION VARIABLES
10    CALL DVCOEF(NROWS,TEMP,IBASIS,T,RHS,PRICE,DVNAME,STATUS)
C
C     DO SENSITIVITY FOR ORIGINAL RIGHT HAND SIDE VALUES
      CALL RSIDE(NROWS,PRICE,T,OBASIS,RHS,IBASIS,CNAME,ORHS)
C
20    RETURN
      END
```

```
      SUBROUTINE PROFIT(PRICE,T,TEMP,RHS,STATUS,NROWS)
C     ****************************************************************
C     *  SUBROUTINE PROFIT DETERMINES THE VALUE OF THE OBJECTIVE      *
C     *  FUNCTION BASED ON THE CURRENT BASIS.  IT ALSO PROTECTS AGAINST*
C     *  INFINITE LOOPS BY LIMITING THE NUMBER OF TIMES THE BASIS CAN  *
C     *  BE CHANGED WITHOUT A CORRESPONDING CHANGE IN THE OBJECTIVE    *
C     *  FUNCTION.  CALLED BY SOLVE.                                   *
C     *  LIST OF VARIABLES...                                          *
C     *  ICOUNT....COUNTER WHICH TRACKS THE NUMBER OF TIMES THE BASIS  *
C     *            CHANGES WITHOUT A CHANGE IN THE OBJECTIVE FUNCTION. *
C     *            WHEN ICOUNT=NROWS, THE PROGRAM TERMINATES.          *
C     *  OLDTOT....VALUE OF OBJECTIVE FUNCTION FOR LAST ITERATION.     *
C     *  TOTAL.....VALUE OF OBJECTIVE FUNCTION FOR THIS ITERATION      *
C     ****************************************************************
      COMMON BIGM,COEF,CYCLE,DRIVE,IBANER,IBOT,INDEX,ITOP,IPRINT,MAX,ME,
     1NAMES,HEAD,NBASIS,NCOLS,SEN,TITLE(6),TCOLP1,TOTAL,TOTCOL,UNIT,ZERO
      DIMENSION PRICE(1),T(NROWS,1),TEMP(1),STATUS(1),RHS(1)
      INTEGER TCOLP1,TOTCOL,STATUS
      LOGICAL CYCLE
      DATA ICOUNT/1/
C
      OLDTOT=TOTAL
      NBASIS=NBASIS+1
      TOTAL=0.
      DO 5 I=1,NROWS
5     TOTAL=TOTAL+RHS(I)*PRICE(I)
C
C     COMPARE OLD TOTAL TO THE NEW ONE
      IF(TOTAL-OLDTOT)20,15,10
C
C     ERROR, INCREASING OBJECTIVE FUNCTION
10    WRITE(6,12)NBASIS
12    FORMAT(1X,10(1H*),56HERROR IN SUBROUTINE PROFIT-INCREASING OBJEDTI
     1VE FUNCTION,/,11X,13HBASIS NUMBER=,I3)
      ICOUNT=1E50
      GO TO 21
C
C     ERROR, OBJECTIVE FUNCTION NOT DECREASING
15    ICOUNT=ICOUNT+1
      IF(ICOUNT.LT.NROWS)GO TO 21
      WRITE(6,16)NBASIS
16    FORMAT(1X,10(1H*),60HERROR IN SUBROUTINE PROFIT-OBJECTIVE FUNCTION
     1 NOT DECREASING,/,11X,13HBASIS NUMBER=,I3)
      GO TO 21
C
C     EVERYTHING O.K.-DECREASE IN OBJECTIVE FUNCTION-REINITIALIZE COUNT
20    ICOUNT=1
C
```

```
      ENTRY ROWZ
      CYCLE=.FALSE.
C     FIND (C-Z) FOR EACH COLUMN(CALCULATE ROW 0)
21    DO 30 I=1,TOTCOL
C     SET (C-Z)=0 FOR BASIC ELEMENT TO AVOID ACCUMULATION OF ROUNDOFF
      IF(STATUS(I).LT.0)GO TO 25
      TEMP(I)=0.
      GO TO 30
C
C     NONBASIC ELEMENT
25    TEMP(I)=PRICE(NROWS+I)
      DO 28 J=1,NROWS
28    TEMP(I)=TEMP(I)-PRICE(J)*T(J,I)
C     CYCLE AGAIN?
      IF(TEMP(I).LT.0)CYCLE=.TRUE.
30    CONTINUE
C
      IF(ICOUNT.GE.NROWS)CYCLE=.FALSE.
C
      RETURN
      END
```

```
      SUBROUTINE REPLY(ANS),RETURNS(NBAD)
C     ***********************************************************************
C     *  THIS SUBROUTINE CHECKS THE MAGNITUDE OF INTERACTIVELY              *
C     *  CHANGED COEFFICIENTS. CALLED BY DELTA.                             *
C     ***********************************************************************
      READ(5,*)ANS
      IF(ABS(ANS).LT.999999.5)RETURN
      WRITE(6,17)ANS
17    FORMAT(57H MAXIMUM ALLOWABLE VALUE OF COEFFICIENT=999999. YOU INPU
     1T,1PE11.3)
      RETURN NBAD
      END
```

```
      SUBROUTINE RSIDE(NROWS,PRICE,T,OBASIS,RHS,IBASIS,CNAME,ORHS)
C     ****************************************************************
C     *  SUBROUTINE RSIDE DOES SENSITIVITY FOR CHANGES TO THE        *
C     *  ORIGINAL RIGHT HAND SIDE VALUES.  IT DETERMINES HOW MUCH THE *
C     *  RIGHT SIDE COULD INCREASE OR DECREASE BEFORE A VARIABLE     *
C     *  LEAVES THE FINAL BASIS.  IT ALSO DETERMINES WHAT THE LEAVING *
C     *  VARIABLE WOULD BE. CALLED BT POSTOP                         *
C     *  LIST OF VARIABLES...                                        *
C     *  LVMIN.....THE LEAVING VARIABLE FOR THE MINIMUM VALUE        *
C     *  LVMAX.....THE LEAVING VARIABLE FOR THE MAXIMUM VALUE        *
C     *  NEGMAX....AMOUNT SUBTRACTED FROM THE ORIGINAL TO DETERMINE  *
C     *            THE MINIMUM RIGHT HAND SIDE VALUE                 *
C     *  PMAX......MAXIMUM VALUE OF COEFFICIENT(PRICE)               *
C     *  PMIN......MINIMUM VALUE OF COEFFICIENT(PRICE)               *
C     *  POSMIN....AMOUNT ADDED TO THE ORIGINAL TO DETERMINE THE     *
C     *            MAXIMUM RIGHT HAND SIDE VALUE                     *
C     ****************************************************************
      COMMON BIGM,COEF,CYCLE,DRIVE,IBANER,IBOT,INDEX,ITOP,IPRINT,MAX,ME,
     1NAMES,HEAD,NBASIS,NCOLS,SEN,TITLE(6),TCOLP1,TOTAL,TOTCOL,UNIT,ZERO
      DIMENSION PRICE(1),T(NROWS,1),OBASIS(1),RHS(1),IBASIS(1),ORHS(1)
      INTEGER OBASIS
      LOGICAL FROMRS
      REAL NEGMAX
C
      WRITE(6,1)
1     FORMAT(1H1,60(1H*),/,
     161H * THIS IS A SENSITIVITY ANALYSIS ON THE ORIGINAL RIGHT HAND*,/
     161H * SIDE(RHS) VALUES. WHILE ONE RHS CHANGES, THE OTHERS KEEP *,/
     124H * THEIR INITIAL VALUES.,36X,1H*,/,
     130H * DESCRIPTION OF VARIABLES...,30X,1H*,/,
     161H *      IF THE VALUE OF AN ORIGINAL RHS FALLS BELOW THE     *,/
     161H *      MINIMUM VALUE OR RISES ABOVE THE MAXIMUM VALUE,     *,/
     161H *      THEN VARIABLE LV WOULD LEAVE THE FINAL BASIS.       *,/
     11X,60(1H*),//,13X,8HORIGINAL,5X,
     17HMINIMUM,12X,7HMAXIMUM,/,1X,10HCONSTRAINT,4X,5HVALUE,7X,5HVALUE,
     14X,2HLV,8X,5HVALUE,4X,2HLV,/,
     12X,8(1H-),2(3X,9(1H-)),2X,2H--,6X,9(1H-),2X,2H--)
C
C     EXAMINE EACH COEFFICIENT
      DO 30 I=1,NROWS
      POSMIN=1E100
      NEGMAX=-1E100
      LVMIN=0
      LVMAX=0
```

```
          DO 20 J=1,NROWS
C
C         IS BASIC VARIABLE IN ROW J A POTENTIAL LEAVING VARIABLE?
          IF(ABS(T(J,OBASIS( I  ))).LT.ZERO)GO TO 20
C
C         YES, POTENTIAL LEAVING VARIABLE
          DEL=-RHS(J)/T(J,OBASIS(I))
          IF(DEL.GT.0.)GO TO 10
C
C         DEL<0, FIND MAXIMUM NEGATIVE
          IF(DEL.LT.NEGMAX)GO TO 20
          NEGMAX=DEL
          LVMIN=IBASIS(J)
          GO TO 20
C
C         DEL>0, FIND MINIMUM POSITIVE
10        IF(DEL.GT.POSMIN)GO TO 20
          POSMIN=DEL
          LVMAX=IBASIS(J)
20        CONTINUE
C
        PMIN=ORHS(I)+NEGMAX
        PMAX=ORHS(I)+POSMIN
        FROMRS=.TRUE.
        CALL OPTION(NAMES,I    ,CNAME,ORHS(I),       PMIN,LVMIN,LVMIN,0.,
       1            PMAX,LVMAX,LVMAX,0.,FROMRS)
30      CONTINUE
C
        RETURN
        END
```

```
      SUBROUTINE ROW(LEAVE,ENTER ,CYCLE,T,RHS,NROWS,ZERO,SMALL)
C     ****************************************************************
C     *  SUBROUTINE ROW DETERMINES THE ROW OF THE LEAVING VARIABLE     *
C     *            GIVEN THE ENTERING VARIABLE                         *
C     *  CALLED BY ARTOUT, DVCOEF, AND ITERATE.                        *
C     *  LIST OF VARIABLES...                                          *
C     *  ENTER.....INTEGER.  ENTERING VARIABLE                         *
C     *  LEAVE.....ROW NUMBER OF LEAVING VARIABLE                      *
C     *  SMALL.....SMALLEST POSITIVE RATIO OF A RIGHT HAND SIDE I      *
C     *            TO THE TABLEAU ELEMENT T(I,ENTER)                   *
C     ****************************************************************
      DIMENSION T(NROWS,1),RHS(1)
      LOGICAL CYCLE
      INTEGER ENTER
      SMALL=1E100
      LEAVE=0
      DO 20 I=1,NROWS
C     CHECK FOR POSITIVE ROW ELEMENT IN COLUMN ENTER
      IF(T(I,ENTER) .LT.ZERO)GO TO 20
C
C     POSITIVE FOUND
      QUOT=RHS(I)/T(I,ENTER)
C
C     THERE IS NO TIE BREAKER FOR DEGENERACY. DUE TO ROUNDOFF, TWO
C     ROWS WILL RARELY TIE FOR THE LEAVING VARIABLE.  IN THE EVENT THAT
C     THIS OCCURS, THE FIRST ROW FOUND WILL BE THE LEAVING ONE
      IF(QUOT.GE.SMALL)GO TO 20
C
C     CANDIDATE FOUND
      LEAVE=I
      SMALL=QUOT
20    CONTINUE
C     CHECK RESULTS
C     IF LEAVE=0 NO LEAVING VARIBALE FOUND
      IF(LEAVE.NE.0)GO TO 30
      CYCLE=.FALSE.
      WRITE(6,25)
25    FORMAT(55H NO LEAVING BASIC VARIABLE-UNBOUNDED OBJECTIVE FUNCTION)
C
C     IF SMALL<0 A RHS WAS NEGATIVE
30    IF(SMALL.GE.-ZERO)RETURN
      CYCLE=.FALSE.
      WRITE(6,32)
32    FORMAT(/,37H NEGATIVE RIGHT HAND SIDE ENCOUNTERED)
      RETURN
      END
```

```
      SUBROUTINE SA(NROWS,PRICE,TEMP,OBASIS,T,ORHS,RHS,IBASIS,STATUS)
C     ****************************************************************
C     *  THIS SUBROUTINE CONDUCTS INTERACTIVE SENSITIVITY ANALYSES.  *
C     *  CALLED BY LPSOLVE.                                          *
C     ****************************************************************
      DIMENSION PRICE(1),IBASIS(1)
      COMMON BIGM,COEF,CYCLE,DRIVE,IBANER,IBOT,INDEX,ITOP,IPRINT,MAX,ME,
     1NAMES,HEAD,NBASIS,NCOLS,SEN,TITLE(6),TCOLP1,TOTAL,TOTCOL,UNIT,ZERO
      INTEGER CASE,TOTCOL
      LOGICAL HEAD,SEN
C
      IF(.NOT.SEN)RETURN
      CASE=0
C
C     CHANGE SIGN OF OBJECTIVE FUNCTION IF NECESSARY
      IF(MAX.NE.3HMAX)GO TO 8
      DO 5 I=1,TOTCOL
      INDEX=NROWS+I
5     PRICE(INDEX)=-PRICE(INDEX)
C
8     IPRINT=-1
      WRITE(6,10)
10    FORMAT(1H1,59(1H*),/,
     160H * YOU MAY NOW CONDUCT AN INTERACTIVE SENSITIVITY ANALYSIS.*)
      IF(HEAD)WRITE(6,15)
15    FORMAT(
     160H * YOU MAY SIMULTANEOUSLY CHANGE OBJECTIVE FUNCTION        *,/,
     160H * COEFFICIENTS OF DECISION VARIABLES, CONSTRAINT          *,/,
     160H * COEFFICIENTS OF DECISION VARIABLES, AND RIGHT HAND      *,/,
     160H * SIDES.  YOU MAY CONSIDER AS MANY CASES AS YOU DESIRE.   *,/,
     160H * THE CHANGES FOR CASE 1 CARRY OVER TO CASE 2 AND SO ON.  *,/,
     160H * BASED ON YOUR CHANGES, THE NEW BASIS AND NEW VALUE OF   *,/,
     160H * THE OBJECTIVE FUNCTION WILL BE OUTPUT.  THE PROGRAM     *,/,
     160H * WILL PROMPT YOU FOR INFORMATION, GIVE YOU A CHOICE OF   *,/,
     160H * RESPONSES FOR ALPHA REPLYS, AND GIVE YOU NUMBER TYPE    *,/,
     160H * FOR NUMERICAL RESPONSES.  REMEMBER TO HIT THE RETURN    *,/,
     160H * BUTTON AFTER EACH RESPONSE.  LET'S GET STARTED.         *)
      WRITE(6,17)
17    FORMAT(1X,59(1H*))
20    WRITE(6,25)
25    FORMAT(42H DO YOU WISH TO MAKE ANY CHANGES? (Y,N)...)
30    CALL YESNO,RETURNS(60,80,20)
60    CASE=CASE+1
      CALL DELTA(NROWS,PRICE,TEMP,OBASIS,T,ORHS,RHS,STATUS)
```

```
      DO 62 I=1,NROWS
C     DUAL IS AN ENTRY POINT IN SWITCH
62    CALL DUAL(I,IBASIS(I),TCOLP1,T,NROWS,IBASIS,PRICE,STATUS)
      CALL DUALSM(NROWS,RHS,T,TEMP,IBASIS,PRICE,STATUS)
      WRITE(6,65)CASE
65    FORMAT(////,6H CASE ,I3)
      CALL SOLVE(T,RHS,PRICE,TEMP,IBASIS,STATUS,NROWS)
      WRITE(6,70)
70    FORMAT(47H DO YOU WISH TO MAKE ANY MORE CHANGES? (Y,N)...)
      GO TO 30
80    CONTINUE
      RETURN
      END
```

```
      SUBROUTINE SLACK(OBASIS,PRICE,TEMP,IBASIS,RHS,NROWS,CNAME,STATUS)
C     ****************************************************************
C     *  SUBROUTINE SLACK OUTPUTS ANY SURPLUS IN THE CONSTRAINTS AND  *
C     *  THE SHADOW PRICES OF THE SLACK VARIABLES. CALLED BY POSTOP.  *
C     *  LIST OF VARIABLES...                                         *
C     *  SHADOW....SHADOW PRICE                                       *
C     *  SURPLUS...AMOUNT OF SURPLUS                                  *
C     ****************************************************************
      COMMON BIGM,COEF,CYCLE,DRIVE,IBANER,IBOT,INDEX,ITOP,IPRINT,MAX,ME,
     1NAMES,HEAD,NBASIS,NCOLS,SEN,TITLE(6),TCOLP1,TOTAL,TOTCOL,UNIT,ZERO
      DIMENSION PRICE(1),TEMP(1),IBASIS(1),RHS(1),CNAME(1),OBASIS(1),
     1          STATUS(1)
      INTEGER OBASIS,STATUS,TOTCOL
      LOGICAL NAMES,HEAD
C
      WRITE(6,1)
1     FORMAT(1H1,70(1H*),/,
     171H * THIS OUTPUT BLOCK IDENTIFIES CONSTRAINT SHADOW PRICES AND TH
     1E      *,/,39H * AMOUNT OF SURPLUS IN THE CONSTRAINTS,31X,1H*)
      IF(HEAD)WRITE(6,2)
2     FORMAT(29H * DESCRIPTION OF HEADINGS...,41X,1H*,/,
     149H * SURPLUS........AMOUNT OF RESOURCE NOT UTILIZED,21X,1H*,/,
     171H * VARIABLE.......SLACK OR ARTIFICAL VARIABLE WHICH WAS INTRODU
     1CED    *,/,2H *,16X,19HWITH THE CONSTRAINT,33X,1H*,/,
     171H * SHADOW PRICE...PRICE ONE WOULD BE WILLING TO PAY FOR AN ADDI
     1TIONAL *,/,
     12H *,16X,53HUNIT OF RESOURCE. FOR GE CONSTRAINTS, THE PRICE ONE */
     12H *,16X,53HWOULD PAY TO RELAX(REDUCE) THE CONSTRAINT BY A UNIT.*/
     12H *,16X,53HEQ CONSTRAINTS MAY HAVE NEGATIVE SHADOW PRICES. IF  */
     12H *,16X,53HTHIS OCCURS, INCREASING THE RESOURCE DRIVES THE     */
     12H *,16X,42HOBJECTIVE FUNCTION IN THE WRONG DIRECTION.,10X,1H*)
      WRITE(6,4)
4     FORMAT(1X,70(1H*),//,17X,10HCONSTRAINT,17X,6HSHADOW,/,
     13X,10HCONSTRAINT,5X,7HSURPLUS,6X,8HVARIABLE,5X,5HPRICE,/,
     13X,10(1H-),4X,10(1H-),4X,8(1H-),4X,9(1H-))
C
C     EXAMINE EACH CONSTRAINT
      DO 40 I=1,NROWS
C
C     CALCULATE SHADOW PRICES, SEE P96 HILLIER AND LIEBERMAN FOR DETAILS
      IF(IABS(STATUS(OBASIS(I))).EQ.3)GO TO 10
C
C     EQ OR GE CONSTRAINT
      K=OBASIS(I)+1
      IF(K.GT.TOTCOL)GO TO 6
      IF(IABS(STATUS(K)).EQ.4)GO TO 8
C
C     EQ CONSTRAINT
6     SHADOW=TEMP(OBASIS(I))-BIGM
      GO TO 12
C
```

```
C     GE CONSTRAINT
8     SHADOW=TEMP(K)
      GO TO 12
C
C     LE CONSTRAINT
10    SHADOW=TEMP(OBASIS(I))
C
C     CALCULATE CONSTRAINT SURPLUS.  IF AN ORIGINAL BASIS ELEMENT
C     IS IN THE FINAL BASIS, THEN A CONSTRAINT MUST HAVE A SURPLUS
12    SURPLUS=0.
C     IS OBASIS(I) A BASIC ELEMENT?
      IF(STATUS(OBASIS(I)).LT.0)GO TO 20
C     YES A BASIS ELEMENT, WHICH ROW?
         DO 15 J=1,NROWS
         SURPLUS=RHS(J)
         IF(IBASIS(J).NE.OBASIS(I))GO TO 15
         GO TO 20
15       CONTINUE
C
20    IF(.NOT.NAMES)GO TO 30
C
C     CONSTRAINTS HAVE NAMES
      WRITE(6,25)I,CNAME(I),SURPLUS,OBASIS(I),SHADOW
25    FORMAT(1X,I3,1X,A8,1PE14.3,I9,E16.3)
      GO TO 40
C
C     CONSTRAINTS DON'T HAVE NAMES
30    WRITE(6,35)I,SURPLUS,OBASIS(I),SHADOW
35    FORMAT(1X,I9,1PE17.3,I9,E16.3)
40    CONTINUE
      RETURN
      END
```

```
      SUBROUTINE SOLVE(T,RHS,PRICE,TEMP,IBASIS,STATUS,NROWS)
C     ****************************************************************
C     *  SUBROUTINE SOLVE IS THE MAIN PROGRAM DRIVER.  IT CONTROLS   *
C     *  THE ACTUAL SOLUTION OF THE LP PROBLEM. CALLED BY LPSOLVE,SA. *
C     *  LIST OF VARIABLES...                                        *
C     *  CYCLE.....LOGICAL VARIABLE.  CYCLE=.TRUE. IMPLIES THE PROBLEM *
C     *            HAS NOT BEEN SOLVED AND ANOTHER ITERATION IS TO   *
C     *            TAKE PLACE.                                       *
C     *  TOTAL.....CALCULATED IN SUBROUTINE PROFIT.  IT IS THE VALUE  *
C     *            OF THE OBJECTIVE FUNCTION IMPLIED BY THE BASIS.   *
C     *  NBASIS....NUMBER OF THE CURRENT BASIS. INITIAL BASIS IS 0.   *
C     ****************************************************************
      COMMON BIGM,COEF,CYCLE,DRIVE,IBANER,IBOT,INDEX,ITOP,IPRINT,MAX,ME,
     1NAMES,HEAD,NBASIS,NCOLS,SEN,TITLE(6),TCOLP1,TOTAL,TOTCOL,UNIT,ZERO
      DIMENSION T(NROWS,1),RHS(1),PRICE(1),TEMP(1),IBASIS(1)
      INTEGER TOTCOL,TCOLP1
      LOGICAL DRIVE,CYCLE
C
      CYCLE=.TRUE.
      TOTAL=1E100
      NBASIS=-1
C
C     CALCULATE VALUE OF OBJECTIVE FUNCTION IMPLIED BY CURRENT BASIS.
C     ALSO CALCULATE ROW 0.
5     CALL PROFIT(PRICE,T,TEMP,RHS,STATUS,NROWS)
C
C     PRINT OUTPUT FOR THIS ITERATION BASED ON OPTION IPRINT
      CALL OUTPUT(NROWS,PRICE,IBASIS,T,TEMP,RHS,STATUS)
      IF(.NOT.CYCLE)GO TO 20
      IF(.NOT.DRIVE)GO TO 10
C
C     TRY TO DRIVE ARTIFICAL VARIABLES FROM THE BASIS
      CALL ARTOUT(NROWS,PRICE,TEMP,T,IBASIS,RHS,STATUS)
      IF(.NOT.DRIVE)GO TO 10
      GO TO 5
C
C     FIND THE NEXT BASIS
10    CALL ITERATE(TEMP,T,RHS,NROWS,IBASIS,PRICE,STATUS)
      GO TO 5
C     CYCLE=.FALSE., DON'T DO ANY MORE ITERATIONS
20    CONTINUE
      RETURN
      END
```

```
      SUBROUTINE SWITCH(LEAVE,JENTER,TCOLP1,T,NROWS,IBASIS,PRICE,STATUS)
C     ****************************************************************
C     *  SUBROUTINE SWITCH UPDATES THE BASIS BY PLACING IN IT THE     *
C     *  ENTERING VARIABLE AND REMOVING THE LEAVING VARIABLE.         *
C     *  CALLED BY ARTOUT, DUALSM, AND ITERATE.                       *
C     *  LIST OF VARIABLES...                                         *
C     *  PIVOT.....THE TABLEAU ELEMENT WHICH IS IN THE ENTERING COLUMN *
C     *            AND LEAVING ROW.                                   *
C     ****************************************************************
      DIMENSION T(NROWS,1),IBASIS(1),PRICE(1),STATUS(1)
      INTEGER TCOLP1,STATUS
C
C     UPDATE STATUS
      STATUS(IBASIS(LEAVE))=-STATUS(IBASIS(LEAVE))
      STATUS(JENTER)=-STATUS(JENTER)
C
      ENTRY DUAL
C
C
C     DIVIDE ROW LEAVE BY PIVOT
      PIVOT =T(LEAVE,JENTER)
      DO 10 I=1,TCOLP1
10    T(LEAVE,I)=T(LEAVE,I)/PIVOT
C     SET PIVOT=1. TO AVOID ACCUMULATION OF ROUNDOFF ERROR
      T(LEAVE,JENTER)=1.
C
C     ZERO OUT COLUMN JENTER EXCEPT FOR PIVOT
      DO 30 I=1,NROWS
      IF(I.EQ.LEAVE) GO TO 30
      FACTOR=T(I,JENTER)
         DO 20 J=1,TCOLP1
         IF(J.NE.JENTER)GO TO 15
C        SET=0 TO AVOID ROUNDOFF ERROR
         T(I,J)=0.
         GO TO 20
15       T(I,J)=T(I,J)-FACTOR     *T(LEAVE,J)
20       CONTINUE
30    CONTINUE
C
C     UPDATE BASIS AND BASIS PRICE
      IBASIS(LEAVE)=JENTER
      PRICE(LEAVE)=PRICE(NROWS+JENTER)
C
      RETURN
      END
```

```
      SUBROUTINE YESNO,RETURNS(Y,N,NOVER)
C     ********************************************************************
C     *  THIS SUBROUTINE CHECKS THE RESPONSE TO YES, NO QUESTIONS.       *
C     *  CALLED BY DELTA, POSTOP, AND SA.                                *
C     ********************************************************************
C
      READ(5,4)ANS
4     FORMAT(A1)
      IF(ANS.EQ.1HY)RETURN Y
      IF(ANS.EQ.1HN)RETURN N
      WRITE(6,5)
5     FORMAT(33H INCORRECT RESPONSE, REPLY Y OR N)
      RETURN NOVER
      END
```

```
      PROGRAM LPFRONT(INPUT=0/80,OUTPUT=0,TAPE1=0,TAPE4=0,
     1TAPE3=0,TAPE5=INPUT,TAPE6=OUTPUT)
C     ******************************************************************
C     *  LPFRONT IS A PREPROCESSOR FOR THE LINEAR PROGEAMMING          *
C     *  PROGRAM LPSOLVE.  THIS PROGRAM IS A MODIFICATION OF GORLPP    *
C     *  WHICH WAS WRITTEN BY  BY ROBERT M. SCHUMACHER.  THIS PROGRAM  *
C     *  WAS WRITTEN BY MICHAEL A. SCHIEFER, GOR79D.                   *
C     *  I WOULD HAVE PUT MORE COMMENTS IN IT BUT I RAN OUT OF TIME.   *
C     ******************************************************************
      COMMON/FLAGS/IEXPERT,NEWP,ICRASH,IMODFY,ISDATA,IINTEG
      COMMON/COUNTS/ICRSHCT,NVAR,NCON,NINT,NAME,IBOX,ILBL,ITYPE,IRCNT,
     X  IRGT,ISCALE,IPOPT,IDRIVE,NGREAT,UNIT,HD,SN,IBANER,II(2)
      COMMON//WORK(99,99),ILABEL(199),ITITLE(6)
      LOGICAL IQUIT
C
C     REQUEST,PERMFIL, AND ROUTE ARE BATTELLE DISK FILE ROUTINES
      CALL REQUEST(5HTAPE3,2H*Q)
      DO 1 I=1,9801
1     WORK(I)=0.
C     DETERMINE USER DESIRES
      CALL SAY(1)
5     N=1
      CALL INIT(5,N)
      GO TO (10,20,30,40,50)N
10    CALL SAY(25)
      GO TO 5
C
C     CREATE NEW PROBLEM
20    CALL RETURN(5HTAPE4)
      CALL REQUEST(5HTAPE4,3H*PF)
      ICRSHCT=1
      CALL THEDATA
      GO TO 2000
C
C     RECOVER FROM CRASH OR USER ABORT
30    CALL GETNUM
      CALL THEDATA
      GO TO 2000
C
C     MODIFY EXISTING PROBLEM
40    CALL RETURN(5HTAPE4)
      CALL ATCH(5HTAPE4,N),RETURNS(5)
      CALL THEDATA
      GO TO 2000
C
C     RUN EXISTING PROBLEM WITHOUT MODIFICATION
50    CALL ATCH(5HTAPE4,N),RETURNS(5)
```

```
C     FORMAT DATA FOR INPUT TO LPSOLVE
2000  CALL TLPKODE,RETURNS(2105)
      CALL THEDATA
      GO TO 2000
C
C     SAVE DATA THEN TERMINATE
2105  IF(N.EQ.5)GO TO 2106
      IQUIT=.FALSE.
      CALL SAVE(IQUIT)
2106  CALL SAY(2)
      REWIND 1
      STOP " USER STOP AFTER LPFAIT SETUP"
      END
```

```
      SUBROUTINE ASK(I,IANS),RETURNS(M,NY,NN,NH)
C     ******************************************************************
C     *  THIS SUBROUTINE ASKS QUESTIONS WHICH HAVE A YES/NO ANSWER     *
C     ******************************************************************
      IF (I.LE.0.OR.I.GT.20) STOP "BAD CALL TO ASK"
  998 GO TO (1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20)I
1     WRITE(6,100)
100   FORMAT(77H DO YOU WANT TO SUPPLY NAMES FOR CONSTRAINTS AND DECISIO
     1N VARIABLES? (Y,N)...)
      GO TO 997
2     PRINT*,"ARE YOU GOING TO DEFINE A NEW PROBLEM? (Y,N)..."
      GO TO 997
3     PRINT*,"ARE YOU RECOVERING FROM A CRASH OR USER ABOUT? (Y,N)..."
      GO TO 997
4     WRITE(6,400)
400   FORMAT(75H ARE YOU GOING TO MODIFY AN EXISTING PROBLEM? NOTE...THI
     1S DOES MEAN MODIFY.,/,49H THE ORIGINAL PROBLEM WILL BE DESTROYED.
     1(Y,N)...)
      GO TO 997
5     PRINT*,"DO YOU WISH TO RUN LPAFIT WITH AN EXISTING "
      PRINT*,"DATA SET WITHOUT ANY CHANGES? (Y,N)..."
      GO TO 997
6     PRINT*,"IS THIS EQUATION OK? (Y,N)..."
      GO TO 997
7     PRINT*,"DO YOU REALLY WANT TO ADD A NEW DECISION VARIABLE?(Y,N).."
      GO TO 997
8     PRINT*,"DO YOU REALLY WANT TO DELETE A DECISION VARIABLE?(Y,N)..."
      GO TO 997
9     PRINT*,"DO YOU REALLY WANT TO ADD A CONSTRAINT? (Y,N)..."
      GO TO 997
10    PRINT*,"DO YOU REALLY WANT TO DELETE A CONSTRAINT? (Y,N)..."
      GO TO 997
11    WRITE(6,1100)
1100  FORMAT(69H DO YOU KNOW HOW TO RESPOND TO PROGRAM REQUESTS FOR REAL
     1*NUMBER DATA?,/,61H (I.E. DO YOU KNOW HOW TO ENTER LIST DIRECTED D
     1ATA?) (Y,N)...)
      GO TO 997
12    WRITE(6,1200)
1200  FORMAT(75H DO YOU WANT THE MOST RECENT VERSION OF THE DATA(LARGEST
     1 CYCLE #)? (Y,N)...)
      GO TO 997
13    WRITE(6,1300)
1300  FORMAT(54H DO YOU HAVE A PASSWORD TO PROTECT YOUR FILE? (Y,N)...)
      GO TO 997
14    WRITE(6,1400)
1400  FORMAT(52H IF YOU HAVE JUST CREATED A DATA SET, DO YOU WANT TO,/,
     159H SAVE IT (PERMENANT FILE) FOR USE AT A LATER TIME? (Y,N)...)
      GO TO 997
15    RETURN
```

```
16    WRITE(6,1600)
1600  FORMAT(63H DO YOU WANT TO PUNCH YOUR DATA FOR LATER BATCH INPUT? (
     1Y,N)...)
      GO TO 997
17    WRITE(6,1700)
1700  FORMAT(76H YOU HAVE NOT SAVED YOUR DATA, DO YOU WANT ANOTHER CHANC
     2E TO DO SO? (Y,N)...)
      GO TO 997
18    WRITE(6,1800)
1800  FORMAT(52H EXTENSIVE HEADINGS?(Y,N, H FOR MORE INFORMATION)...)
      GO TO 997
19    WRITE(6,1900)
1900  FORMAT(66H INTERACTIVE SENSITIVITY ANALYSIS?(Y,N, H FOR MORE INFOR
     1MATION)...)
      GO TO 997
20    WRITE(6,2000)
2000  FORMAT(74H DO YOU WANT TO DRIVE ARTIFICAL VARIABLES OUT OF THE BAS
     1IS FIRST? (Y,N)...)
997   READ 150,IANS
150   FORMAT(A10)
      CALL CKANS(IANS),RETURNS(1000,1010,1020,1030,998)
 1000 RETURN NY
1010  RETURN NN
 1020 RETURN NH
 1030 RETURN M
      END
```

```
      SUBROUTINE ATCH(LFN,N),RETURNS(M)
C     ***********************************************************************
C     *  THIS SUBROUTINE ATTACHES PERMANENT FILES SPECIFIED BY THE USER*
C     ***********************************************************************
      COMMON/NAMEIN/NAME(41),NAMEOUT(4)
      LOGICAL CY,PW
      CY=.TRUE.
      PW=.FALSE.
1     FORMAT(A10)
5     WRITE(6,6)
6     FORMAT(36H UNDER WHAT NAME IS YOUR DATA SAVED?,/,
     131H (UP TO 40 ALPHA CHARACTERS)...)
      CALL FMTNAME(LAST),RETURNS(50)
7     CALL ASK(12,IANS),RETURNS(50,20,12,10)
10    CALL HELP(7),RETURNS(7)
12    CALL INTNUM(21,ICY,999),RETURNS(50,25,13)
13    CALL HELP(10),RETURNS(12)
20    CY=.FALSE.
25    CALL ASK(13,IANS),RETURNS(50,30,40,28)
28    CALL HELP(7),RETURNS(25)
30    WRITE(6,32)
32    FORMAT(44H WHAT IS THE PASSWORD(UP TO 7 CHARACTERS)...)
      READ(5,1)IPW
      PW=.TRUE.
40    IF(PW.AND.CY)GO TO 44
      IF(.NOT.PW.AND..NOT.CY)GO TO 45
      IF(PW.AND..NOT.CY)GO TO 46
      CALL PERMFIL(ERR,6HATTACH,LFN,NAMEOUT,2HCY,ICY)
      IF(ERR.NE.0)CALL ERROR(ERR),RETURNS(50)
      CALL GETNUM
      IF(N.NE.5)CALL PERMFIL(ERR,5HPURGE,LFN,NAMEOUT,2HCY,ICY)
      GO TO 47
44    CALL PERMFIL(ERR,6HATTACH,LFN,NAMEOUT,2HPW,IPW,2HCY,ICY)
      IF(ERR.NE.0)CALL ERROR(ERR),RETURNS(50)
      CALL GETNUM
      IF(N.NE.5)CALL PERMFIL(ERR,5HPURGE,LFN,NAMEOUT,2HPW,IPW,2HCY,ICY)
      GO TO 47
45    CALL PERMFIL(ERR,6HATTACH,LFN,NAMEOUT)
      IF(ERR.NE.0)CALL ERROR(ERR),RETURNS(50)
      CALL GETNUM
      IF(N.NE.5)CALL PERMFIL(ERR,5HPURGE,LFN,NAMEOUT)
      GO TO 47
46    CALL PERMFIL(ERR,6HATTACH,LFN,NAMEOUT,2HPW,IPW)
      IF(ERR.NE.0)CALL ERROR(ERR),RETURNS(50)
      CALL GETNUM
      IF(N.NE.5)CALL PERMFIL(ERR,5HPURGE,LFN,NAMEOUT,2HPW,IPW)
47    IF(ERR.NE.0)CALL ERROR(ERR),RETURNS(50)
      RETURN
50    RETURN M
      END
```

```
      SUBROUTINE CKANS(I),RETURNS(NY,NN,NH,NQ,NA)
C     ********************************************************************
C     *  THIS SUBROUTINE CHECKS ANSWERS TO YES/NO QUESTIONS              *
C     ********************************************************************
      IF(I.EQ.1HY)RETURN NY
      IF(I.EQ.1HN)RETURN NN
      IF(I.EQ.1HH) RETURN NH
      IF(I.EQ.1HQ) RETURN NQ
      RETURN NA
      END
```

```
      SUBROUTINE CKINT(I,ITOP),RETURNS(NA,NH,NQ)
C     ***************************************************************
C     *  THIS SUBROUTINE CHECKS ANSWERS TO QUESTIONS WHICH WANTS     *
C     *  INTEGER ANSWERS.                                            *
C     ***************************************************************
      IF(I.EQ.-999)RETURN NQ
      IF(I.LE.0) RETURN NH
      IF(I.GT.ITOP)RETURN NH
      RETURN NA
      END
```

```
      SUBROUTINE ECHO(N)
C     ******************************************************************
C     *  THIS SUBROUTINE ECHOS THE CURRENT PROBLEM                     *
C     ******************************************************************
      COMMON//WORK(99,99),ILABEL(199),ITITLE(6)
      COMMON/COUNTS/ICRSHCT,NVAR,NCON,NINT,NAME,IBOX,ILBL,ITYPE,IRCNT,
     X  IRGT,ISCALE,IPOPT,IDRIVE,NGREAT,UNIT,HD,SN,IBANER,II(2)
      COMMON/FLAGS/IEXPERT,NEWP,ICRASH,IMODFY,ISDATA,IINTEG
10    FORMAT(1X,A3,1X,R8,18H WHICH IS EQUAL TO)
11    FORMAT(1X,R8)
12    FORMAT(4(1X,G9.3,1X,R8))
13    FORMAT(10X,A2,5X,G9.3)
      IF(N.EQ.0) WRITE(6,10) ITYPE,ILABEL(1)
      IF(N.NE.0) WRITE(6,11) ILABEL(1+NVAR+N)
      WRITE(6,12) ((WORK(J,N+1),ILABEL(J+1)),J=1,NVAR)
      IF(N.NE.0) WRITE(6,13) ILABEL(NVAR+1+N),WORK(NVAR+1,N+1)
      RETURN
      END
```

```
      SUBROUTINE ERROR(ERR),RETURNS(M)
C     ***************************************************************
C     *  THIS SUBROUTINE FLAGS ERRORS.                              *
C     ***************************************************************
      WRITE(6,10)ERR
10    FORMAT(51H ERROR IN READING OR WRITING FILE NAME. ERROR CODE=,
     1F4.0,/,63H SEE BATTELLE, DISKFILE MANIPULATION ROUTINES, USER'S GU
     1IDE,P14,/,41H FOR FURTHER DETAILS. TRY SOMETHING ELSE.)
      RETURN M
      END
```

```
      SUBROUTINE FMTNAME(LAST),RETURNS(N)
C     ***********************************************************************
C     *  THIS SUBROUTINE CREATES THE PERMANENT FILE NAME UNDER WHICH        *
C     *  A FILE IS CATALOGED.                                               *
C     ***********************************************************************
      COMMON/NAMEIN/NAME(41),NAMEOUT(4)
      DIMENSION TWO(2)
      ICNT=0
5     READ(5,8)(NAME(I),I=1,40)
8     FORMAT(40A1)
      DO 10 I=1,41
      IF((NAME(I).NE.1H ).AND.(NAME(I).NE.1H,))GO TO 10
      LAST=I-1
      GO TO 30
10    CONTINUE
      ICNT=ICNT+1
      IF(ICNT.NE.1)GO TO 15
      WRITE(6,12)
12    FORMAT(56H UNRECOGONIZABLE INPUT--TRY AGAIN. THIS TIME FOLLOW YOUR
     1,/,25H DATA SET NAME WITH A    ,)
      WRITE(6,13)
13    FORMAT(26H YOU HAVE TWO MORE CHANCES)
      GO TO 5
15    IF(ICNT.NE.2)GO TO 20
      WRITE(6,12)
      WRITE(6,16)
16    FORMAT(25H THIS IS YOUR LAST CHANCE)
      GO TO 5
20    WRITE(6,21)
21    FORMAT(66H SORRY I CANNOT READ DATA NAME. I WILL LET YOU TRY SOMET
     1HING ELSE.)
      RETURN N
C
C     ENCODE THE NAME INTO NAMEOUT
30    ENCODE(14,35,TWO)LAST
35    FORMAT(5H(1H*,,I2,7HA1,1H*))
      NEW=LAST+2
      ENCODE(NEW,TWO,NAMEOUT)(NAME(I),I=1,LAST)
      RETURN
      END
```

```
      SUBROUTINE GETK(WT,K)
C     ************************************************************
C     *  THIS SUBROUTINE CALCULATES THE MAXIMUM NUMBER OF DECIMAL  *
C     *  PLACES, K, THAT WT CAN HAVE IN AN F6 FORMAT.              *
C     ************************************************************

C
C     5 DECIMAL PLACES MAXIMUM
      FIRST=.99999499999999
      DO 10 I=1,6
      IF(WT.GT.FIRST)GO TO 10
      K=6-I
      RETURN
10    FIRST=10.*FIRST
      K= -1
      RETURN
C
C     WT<0, 4 DECIMAL PLACES MAXIMUM
15    FIRST=-.99994999999999
      DO 20 I=1,5
      IF(WT.LT.FIRST)GO TO 20
      K=5-I
      RETURN
20    FIRST=10.*FIRST
      K=-1
      RETURN
      END
```

```
      SUBROUTINE GETNUM
C     ******************************************************************
C     *  THIS SUBROUTINE READS DATA PREVIOUSLY WRITTEN BY LPFRONT.     *
C     ******************************************************************
      COMMON//WORK(99,99),ILABEL(199),ITITLE(6)
      COMMON/FLAGS/IEXPERT,NEWP,ICRASH,IMODFY,ISDATA,IINTEG
      COMMON/COUNTS/II(20)
      REWIND 4
      REWIND 4
      READ(4)(II(J),J=1,20)
      NVARP1=II(19)
      NCONP1=II(20)
      DO 5 I=1,NVARP1
5     READ(4)(WORK(I,J),J=1,NCONP1)
      ITOP=NCONP1+NVARP1-1
      READ(4)(ILABEL(J),J=1,ITOP)
      READ(4)(ITITLE(J),J=1,6)
      RETURN
      END
```

```
      SUBROUTINE HELP(I),RETURNS(M)
C     ********************************************************************
C     *  THIS SUBROUTINE OFFERS ADDITIONAL INFORMATION WHEN NEEDED.      *
C     ********************************************************************
      IF(I.LE.0.OR.I.GT.12) STOP"BAD CALL TO HELP"
      GO TO (1,2,3,4,5,6,7,8,9,10,11,12)I
1     PRINT*,"INCORRECT RESPONSE, REPLY MUST BE A POSITIVE INTEGER<100"
      RETURN M
2     PRINT*,"INCORRECT RESPONSE, REPLY MUST BE A POSITIVE INTEGER"
      PRINT*,"LESS THAN OR EQUAL TO THE NUMBER OF DECISION VARIABLES"
      RETURN M
3     PRINT*,"INCORRECT RESPONSE, REPLY MUST BE A POSITIVE INTEGER"
      PRINT*,"LESS THAN OR EQUAL TO THE NUMBER OF CONSTRAINTS"
      RETURN M
4     PRINT*,"INCORRECT RESPONSE, REPLY MUST BE A POSITIVE INTEGER"
      PRINT*,"LESS THAN OR EQUAL TO (# DEC VAR)*(# CONSTRAINTS)"
      RETURN M
5     PRINT*,"INCORRECT RESPONSE, REPLY MUST BE MIN OR MAX"
      RETURN M
6     PRINT*,"INCORRECT RESPONSE, REPLY MUST BE GE  OR  LE  OR  EQ"
      RETURN M
7     PRINT*,"INCORRECT RESPONSE, REPLY MUST BE Y OR N"
      RETURN M
8     WRITE(6,800)
800   FORMAT(/,68H EXPLANATION OF HOW TO ENTER REAL NUMBERS (LIST-DIRECT
     1ED INPUT)----,/,40H YOU ARE GOING TO INPUT THE REAL VALUED ,
     128HCOEFFICIENTS OF THE DECISION,/,
     140H VARIABLES IN THE OBJECTIVE FUNCTION AND,
     138H IN THE CONSTRAINTS. EVERY COEFFICIENT,/,
     140H MUST BE ENTERED, EVEN IF IT IS 0.      ,/,
     140H    *COEFFICIENTS MUST BE SEPARATED BY E,
     133HITHER A BLANK, A COMMA OR A SLASH,/,
     140H    *THE DECIMAL POINT CAN BE OMITTED AN,
     128HD IS ASSUMED TO BE THE RIGHT,/,
     140H     OF THE NUMBER ENTERED. E.G. 34.,6. ,
     122HMAY BE ENTERED AS 34,6,/,
     140H    *TO REPEAT A VALUE, AN INTEGER REPEA,
     137HT CONSTANT IS FOLLOWED BY AN ASTERISK,/,
     140H     AND THE CONSTANT TO BE REPEATED(DO ,
     117HNOT EMBED BLANKS),/,
     140H     E.G.  1.5,0,0,0,3.  MAY BE ENTERED ,
     112HAS 1.5,3*0,3)
```

```
      WRITE(6,850)
850   FORMAT(40H    *IF YOU HAVE TOO MUCH DATA FOR ONE L,
     128HINE ON THE TERMINAL, HIT THE,/,
     140H     CARRIAGE RETURN AND CONTINUE ON THE,
     110H NEXT LINE,/,
     140H    *IF AFTER YOU ENTER LIST DIRECTED DA,
     133HTA, THE TERMINAL DOES NOT RESPOND,/,
     140H     FAIRLY QUICKLY, RECOUNT THE # OF DA,
     137HTA POINTS YOU ENTERED. IF YOU SKIPPED,/,
     140H     A POINT, ENTER ENOUGH GARBAGE TO RE,
     133HACH THE REQUIRED POINT TOTAL. YOU,/,
     140H     WILL THEN BE GIVEN A CHANCE TO CHAN,
     115HGE YOUR ANSWER./)
      RETURN M
9     PRINT*,"INCORRECT RESPONSE, REPLY MUST BE A POSITIVE INTEGER<20"
      RETURN M
10    PRINT*,"INCORRECT RESPONSE,REPLY MUST BE POSITIVE INTEGER<1000"
      RETURN M
11    WRITE(6,1100)
1100  FORMAT(75H THE OUTPUT FROM THE LINEAR PROGRAM CONTAINS DESCRIPTIVE
     1 HEADINGS WHICH ARE,/,72H FAIRLY EXTENSIVE. YOU SHOULD SEE THESE H
     1EADINGS AT LEAST ONCE TO INSURE,/,72H PROPER UNDERSTANDING OF THE
     1OUTPUT. REPLY Y TO SEE EXTENSIVE HEADINGS.,/,70H IF YOU HAVE SEEN
     1THESE HEADINGS, YOU MAY NOT WANT TO WAIT TO SEE THEM,/,41H PRINT.
     1REPLY N FOR ABBREVIATED HEADINGS.,/)
      RETURN M
12    WRITE(6,1200)
1200  FORMAT(67H AFTER LPAFIT SOLVES YOUR LP PROBLEM, YOU MAY INTERACTIV
     1ELY PERFORM,/,22H SENSITIVITY ANALYSES.,/,
     132H REPLY Y IF YOU WANT THIS OPTION,/,31H REPLY N IF YOU DO NOT WA
     1NT IT.)
      RETURN M
      END
```

```
      SUBROUTINE INIT(IU,N)
C     ***********************************************************************
C     *  THIS SUBROUTINE DETERMINES WHAT THE USER WANTS TO DO.              *
C     ***********************************************************************
      COMMON/FLAGS/IANS(6)
10    DO 500 I= 2,IU
   50 CALL ASK(I,IANS(I)),RETURNS(350,250,300,101)
      STOP "BAD ASK RETURN IN INIT"
250   N=I
      RETURN
  300 GO TO (500,500,500,500,101)I
350   CALL KEEPNUM
      CALL SAY(3)
      STOP "USER ABORT  IN INIT"
101   PRINT*,"CHOOSE SOMETHING THAT I CAN DO--TRY AGAIN"
      GO TO 10
  500 CONTINUE
      END
```

```
      SUBROUTINE INTNUM(I,IA,ITOP),RETURNS(MQUIT,MANS,MHELP)
C     ********************************************************************
C     *  THIS SUBROUTINE ASKS QUESTIONS WHICH EXPECT AN INTEGER ANSWER *
C     ********************************************************************
      IF(I.LE.0.OR.I.GT.22) STOP "BAD CALL TO INTNUM"
  998 GO TO (1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22)I
1     WRITE(6,100)
100   FORMAT(71H HOW MANY DECISION VARIABLES(EXCLUDE ARTIFICAL AND SLACK
     1)? (INTEGER)...)
      GO TO 997
2     PRINT*,"TOTAL NUMBER OF CONSTRAINTS? (INTEGER)..."
      GO TO 997
3     WRITE(6,300)
300   FORMAT(60H HOW MANY VARIABLE NAMES DO YOU WANT TO CHANGE? (INTEGER
     1)...)
      GO TO 997
4     PRINT*,"WHICH VARIABLE NUMBER TO CHANGE NEXT? (INTEGER)..."
      GO TO 997
5     WRITE(6,500)
500   FORMAT(62H HOW MANY CONSTRAINT NAMES DO YOU WANT TO CHANGE? (INTEG
     1ER)...)
      GO TO 997
6     PRINT*,"WHICH CONSTRAINT NUMBER TO CHANGE NEXT? (INTEGER)..."
      GO TO 997
7     PRINT*,"HOW MANY OBJECTIVE COEFFICIENTS TO CHANGE? (INTEGER)..."
      GO TO 997
8     PRINT*,"WHICH OBJECTIVE COEFFICIENT TO CHANGE NEXT? (INTEGER)..."
      GO TO 997
9     PRINT*,"HOW MANY RELATIONS (GE,LE,EQ) TO CHANGE? (INTEGER)..."
      GO TO 997
10    PRINT*,"WHICH CONSTRAINT RELATIONSHIP TO CHANGE? (INTEGER)..."
      GO TO 997
11    PRINT*,"HOW MANY RIGHT HAND SIDES TO CHANGE? (INTEGER)..."
      GO TO 997
12    PRINT*,"NUMBER OF THE RIGHT HAND SIDE TO CHANGE NOW? (INTEGER)..."
      GO TO 997
13    PRINT*,"HOW MANY CONSTRAINT COEFFICIENTS TO CHANGE? (INTEGER)..."
      GO TO 997
14    CONTINUE
15    PRINT*,"WHICH VARIABLE TO DELETE? (INTEGER)..."
      GO TO 997
16    CONTINUE
      GO TO 997
17    PRINT*,"WHICH CONSTRAINT TO DELETE? (INTEGER)..."
      GO TO 997
18    PRINT*,"WHICH VARIABLE TO RESCALE? (INTEGER)..."
      GO TO 997
19    PRINT*,"WHICH CONSTRAINT TO RESCALE? (INTEGER)..."
      GO TO 997
```

```
20    WRITE(6,2000)
2000  FORMAT(70H OPTION? (E.G. 1 FOR NO CHANGES, 3 TO REPEAT OPTION LIST
     1) (INTEGER)...)
      GO TO 997
21    WRITE(6,2100)
2100  FORMAT(32H WHAT CYCLE NUMBER? (INTEGER)...)
      GO TO 997
22    RETURN
997   READ*,IA
      CALL CKINT(IA,ITOP),RETURNS(1000,1010,1020)
 1000 RETURN MANS
 1010 RETURN MHELP
 1020 RETURN MQUIT
      END
```

```
      SUBROUTINE KEEPNUM
C     ******************************************************************
C     *  THIS SUBROUTINE STORE USER DATA.                              *
C     ******************************************************************
      COMMON/COUNTS/II(20)
      COMMON/FLAGS/IEXPERT,NEWP,ICRASH,IMODFY,ISDATA,IINTEG
      COMMON//WORK(99,99),ILABEL(199),ITITLE(6)
      REWIND 4
      REWIND 4
      NVARP1=II(19)
      NCONP1=II(20)
      WRITE(4)(II(J),J=1,20)
      DO 5 I=1,NVARP1
5     WRITE(4)(WORK(I,J),J=1,NCONP1)
      ITOP=NCONP1+NVARP1-1
      WRITE(4)(ILABEL(J),J=1,ITOP)
      WRITE(4)(ITITLE(J),J=1,6)
      RETURN
      END
```

```
      SUBROUTINE SAVE(IQUIT)
C     ***********************************************************************
C     *  THIS SUBROUTINE CATALOGS PERMANENT FILES.                          *
C     ***********************************************************************
      COMMON/NAMEIN/NAME(41),NAMEOUT(4)
      LOGICAL PW,KEEP,IQUIT
      PW=.TRUE.
      KEEP=.FALSE.
1     FORMAT(A10)
3     CALL ASK(14,IANS),RETURNS(50,5,20,4)
4     CALL HELP(7),RETURNS(3)
5     WRITE(6,6)
6     FORMAT(54H UNDER WHAT NAME DO YOU WANT THE DATA SET TO BE SAVED?,/
     1,29H UP TO 40 ALPHA CHARACTERS...)
      CALL FMTNAME(LAST),RETURNS(50)
      WRITE(6,7)
7     FORMAT(52H UNDER WHAT PASSWORD DO YOU WANT THE DATA SET SAVED?,/,
     153H UP TO 7 ALPHA CHARACTERS(ENTER * FOR NO PASSWORD)...)
      READ(5,1)IPW
      IF(IPW.EQ.1H*)PW=.FALSE.
      REWIND 4
      IF(PW)CALL PERMFIL(ERR,7HCATALOG,5HTAPE4,NAMEOUT,2HRP,999,
     12HPW,IPW)
      IF(.NOT.PW)CALL PERMFIL(ERR,7HCATALOG,5HTAPE4,NAMEOUT,2HRP,999)
      IF(ERR.NE.0.)CALL ERROR(ERR),RETURNS(3)
      KEEP=.TRUE.
      CALL SAY(24)
20    IF(IQUIT)RETURN
      CALL ASK(16,IANS),RETURNS(50,25,40,23)
23    CALL HELP(7),RETURNS(20)
25    REWIND 3
      WRITE(6,27)
27    FORMAT(70H WHAT THREE LETTERS DO YOU WANT TO IDENTIFY YOUR PUNCHED
     1 DATA DECK?...)
      READ(5,1)IANS
      CALL ROUTE(5HTAPE3,3HDC=,2HPU,4HTID=,2HBB,3HST=,3HCSB,4HFID=,IANS)
      KEEP=.TRUE.
      WRITE(6,29)IANS
29    FORMAT(51H YOUR DECK WILL BE PUNCHED AT AFIT WITH IDENTIFIER ,A3)
40    IF(KEEP)RETURN
44    CALL ASK(17,IANS),RETURNS(50,3,50,45)
45    CALL HELP(7),RETURNS(44)
50    RETURN
      END
```

```
      SUBROUTINE SAY(I)
C     ******************************************************************
C     *  THIS SUBROUTINE TRANSMITS INFORMATION TO THE USER.            *
C     ******************************************************************
      IF(I.LE.0.OR.I.GT.27)STOP "BAD CALL TO SAY"
      GO TO (1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,
     1       22,23,24,25,26,27)I
1     WRITE(6,100)
100   FORMAT(//,61H WELCOME. I AM L.P.AFIT. MY CREATOR WAS MIKE SCHIEFER
     1,GOR79D.,//,
     146H USE THE FOLLOWING METHODS TO ANSWER QUESTIONS,/,
     147H IF THE QUESTION IS A YES/NO TYPE QUESTION, USE,/,
     114H    Y  FOR YES,/,13H    N  FOR NO,/,63H    Q  TO LEAVE THE PREP
     1ROCESSOR IMMEDIATELY AFTER STORING DATA,/,
     150H IF THE QUESTION WANTS A SINGLE INTEGER REPLY, USE,/,
     137H    AN INTEGER FOR THE DESIRED ANSWER,/,65H    *999 TO LEAVE TH
     1E PREPROCESSOR IMMEDIATELY AFTER STORING DATA,/,
     178H PERIODICALLY, GARBAGE WILL BE PRINTED(E.G. PF CYCLE NO.=999).
     1JUST IGNORE IT.)
      RETURN
2     PRINT*,"LPFAIT PREPROCESSOR SUCESSFULLY TERMINATED"
      RETURN
3     WRITE(6,300)
300   FORMAT(23H YOU HAVE USER ABORTED.,/,72H SORRY TO HAVE YOU QUIT. AL
     1L DATA SO FAR IS TEMPORARY.YOU MUST TAKE SOME,/,72H ACTION IN ORDE
     1R TO SAVE IT.  IN A MOMENT, YOU WILL GET A READ EOF ERROR,/,
     139H ON TAPE1. TO RECOVER, TYPE   BEGIN,LP.)
      RETURN
4     WRITE(6,400)
400   FORMAT(80H INPUT OBJECTIVE FUNCTION COEFFICIENTS. START WITH FIRST
     1 DEC. VAR.,END WITH LAST,/)
      RETURN
5     PRINT*,"ECHO OF THE CURRENT PROBLEM BEING SET UP FOR SOLUTION"
      PRINT*," "
      RETURN
6     PRINT*,"SUBJECT TO THESE CONSTRAINTS"
      PRINT*," "
      RETURN
7     WRITE(6,700)
700   FORMAT(67H SELECT AN OPTION FOR CHANGES. YOU ARE ALLOWED MORE THAN
     1 ONE CHANGE,/,36H BUT THEY MUST BE MADE ONE AT A TIME)
      RETURN
```

```
8     WRITE(6,800)
800   FORMAT(15H OPTION  ACTION,15X,14HOPTION  ACTION,/,1X,6(1H-),2X,
     16(1H-),15X,6(1H-),2X,6(1H-),/,
     130H     1    NO MORE CHANGES       ,
     128H10    CHANGE A VARIABLE NAME,/,
     130H     2    REECHO THE PROBLEM    ,
     130H11    CHANGE A CONSTRAINT NAME,/,
     130H     3    REPEAT OPTION LIST    ,
     127H12    CHANGE OBJECTIVE NAME,/,
     130H     4    ADD A VARIABLE        ,
     126H13    EXCHANGE MAX AND MIN,/,
     130H     5    DELETE A VARIABLE     ,
     144H14    CHANGE OBJECTIVE FUNCTION COEFFICIENTS,/,
     130H     6    ADD A CONSTRAINT      ,
     127H15    CHANGE THE (GE,LE,EQ),/,
     130H     7    DELETE A CONSTRAINT   ,
     135H16    CHANGE RIGHT HAND SIDE VALUES,/,
     130H     8    RESCALE A VARIABLE    ,
     136H17    CHANGE CONSTRAINT COEFFICIENTS,/,
     130H     9    RESCALE A CONSTRAINT  ,
     144H18    QUIT NOW--DO NOT RUN LPAFIT--SAVE DATA)
      WRITE(6,850)
850   FORMAT(30X,42H19    CHANGE OUTPUT OR SENSITIVITY OPTIONS)
      RETURN
9     PRINT*,"IS THIS A MAX OR A MIN PROBLEM? (MAX,MIN)..."
      RETURN
10    WRITE(6,1000)
1000  FORMAT(75H INPUT COEFFICIENTS FOR THE NEW DECISION VARIABLE. START
     1 WITH THE OBJECTIVE,/,71H FUNCTION AND THEN ENTER THE COEFFICIENT
     1FOR EACH SUCESSIVE CONSTRAINT.,/)
      RETURN
11    PRINT*,"INPUT THE NEW VARIABLE NAME(UP TO 8 CHARACTERS)..."
      RETURN
12    PRINT*,"THE DELETE IS COMPLETE"
      RETURN
13    WRITE(6,1300)
1300  FORMAT(52H INPUT CONSTRAINT COEFFICIENTS. START WITH THE FIRST,/,
     137H DECISION VARIABLE, END WITH THE LAST,/)
      RETURN
14    PRINT*,"INPUT THE NEW RIGHT HAND SIDE VALUE (REAL)..."
      RETURN
15    PRINT*,"INPUT THE NEW GE,LE,OR EQ? (GE,LE,EQ)..."
      RETURN
16    PRINT*,"INPUT THE NEW CONSTRAINT NAME(UP TO 8 CHARACTERS)..."
      RETURN
17    PRINT*,"ENTER SCALE FACTOR. ENTER 1. FOR NO CHANGE (REAL)..."
      RETURN
18    WRITE(6,1800)
1800  FORMAT(60H LPAFIT PRINT OPTION(OPTION=3 TO DISPLAY OPTIONS) OPTION
     1?...)
      RETURN
19    RETURN
```

```
20    PRINT*,"NUMBER OF GREATER THAN (GE) CONSTRAINTS? (INTEGER)..."
      RETURN
21    WRITE(6,2100)
2100  FORMAT(66H ITERATIVE OUTPUT ON UNIT(2 OR 6, -1 FOR MORE INFORMATIO
     2N) UNIT...)
      RETURN
22    WRITE(6,2200)
2200  FORMAT(71H IF YOU WANT THE INFORMATION GENERATED BY THE LPAFIT PRI
     2NT OPTION TO BE,/,70H PRINTED AT YOUR TERMINAL, UNIT=6. IF THE INF
     2ORMATION IS TOO EXTENSIVE,/,73H TO PRINT AT THE TERMINAL, UNIT=2 W
     2ILL CAUSE IT TO PRINT AT THE AFIT LINE,/,9H PRINTER.,/)
      RETURN
23    WRITE(6,2300)
2300  FORMAT(/,17H OPTION  TO PRINT,/,1X,6(1H-),2X,8(1H-),/,
     14H  -2,5X,39HFIRST TABLEAU, LAST TABLEAU, EACH BASIS,/,
     14H  -1,5X,26HTABLEAU FOR EACH ITERATION,/,
     14H   0,5X,27HFIRST AND LAST TABLEAU ONLY,/,
     14H   1,5X,10HEACH BASIS,/,4H   2,5X,15HLAST BASIS ONLY,/,
     111H OPTION?...)
      RETURN
24    WRITE(6,2400)
2400  FORMAT(68H REMEMBER YOUR DATA SET NAME AND PASSWORD. YOU MUST USE
     1THE DATA SET,/,42H AT LEAST EVERY 7 DAYS OR IT WILL BE LOST.)
      RETURN
25    PRINT*,"PROGRAM RESTARTS BECAUSE THERE IS NOTHING TO DO"
      RETURN
26    WRITE(6,2600)
2600  FORMAT(72H IS THIS AN EQUAL TO,GREATER THAN,OR LESS THAN CONSTRAIN
     1T? (EQ,GE,LE)...)
      RETURN
27    WRITE(6,2700)
2700  FORMAT(73H THIS IS A NORMAL TERMINATION. IGNORE THE EOF ERROR YOU
     1ARE ABOUT TO GET.,/,
     158H IF YOU CONTINUE YOU ARE NOT RECOVERING FROM A USER ABORT.)
      END
```

```
      SUBROUTINE THEDATA
C     ********************************************************************
C     *  THIS SUBROUTINE GATHERS DATA RELATIVE TO THE PROBLEM.           *
C     ********************************************************************
      COMMON//WORK(99,99),ILABEL(199),ITITLE(6)
      COMMON/COUNTS/ICRSHCT,NVAR,NCON,NINT,NAME,IBOX,ILBL,ITYPE,IRCNT,
     XIRGT,ISCALE,IPOPT,IDRIVE,NGREAT,UNIT,HD,SN,IBANER,NVARP1,NCONP1
      COMMON/FLAGS/IEXPERT,NEWP,ICRASH,IMODFY,ISDATA,IINTEG
      INTEGER OPTION,UNIT
      LOGICAL CHANGE,IQUIT
      CHANGE=.FALSE.
12    FORMAT(R8)
13    FORMAT(6A10)
      GO TO (100,200,300,400,520)ICRSHCT
100   ICRSHCT=1
110   CALL INTNUM(1,NVAR,NVAR),RETURNS(9999,120,118)
  118 CALL HELP(1),RETURNS(110)
120   CALL INTNUM(2,NCON,NCON),RETURNS(9999,130,128)
  128 CALL HELP(1),RETURNS(120)
130   PRINT*,"TITLE OF THIS PROBLEM IN 60 SPACES OR LESS?"
      PRINT*,">"
      READ(5,13) (ITITLE(I),I=1,6)
      NVARP1=NVAR+1
      NCONP1=NCON+1
132   CALL SAY(18)
133   READ*,IPOPT
      IF(IPOPT.GE.-2.AND.IPOPT.LE.2) GO TO 135
      CALL SAY(23)
      GO TO 133
135   IDRIVE=0
136   CALL ASK(20,IANS),RETURNS(9999,140,138,137)
137   CALL HELP(7),RETURNS(136)
138   IDRIVE=1
140   CALL SAY(21)
141   READ(5,*)UNIT
      IBANER=3HZZZ
      IF(UNIT.EQ.6)GO TO 142
      IF(UNIT.EQ.2)GO TO 1405
      CALL SAY(22)
      GO TO 140
1405  WRITE(6,1410)
1410  FORMAT(67H WHAT THREE LETTERS DO YOU WANT TO IDENTIFY YOUR PRINTED
     1 OUTPUT?...)
      READ(5,*)IBANER
142   HD=1HT
      CALL ASK(18,IANS),RETURNS(9999,145,144,143)
143   CALL HELP(11),RETURNS(142)
144   HD=1HF
145   SN=1HT
      CALL ASK(19,IANS),RETURNS(9999,149,147,146)
146   CALL HELP(12),RETURNS(145)
147   SN=1HF
149   CONTINUE
      IF(CHANGE)GO TO 531
```

```
150   CALL SAY(20)
      READ(5,*)NGREAT
      IF(NGREAT.LT.0.OR.NGREAT.GT.NCON)CALL HELP(3),RETURNS(150)
      CALL KEEPNUM
200   ICRSHCT=2
  201 CALL ASK( 1,ILBL),RETURNS(9999,205,250,202)
  202 CALL HELP( 7),RETURNS(201)
205   PRINT*,"OBJECTIVE NAME(UP TO 8 CHARACTERS)..."
      READ(5,12)  ILABEL(1)
      PRINT*,"INPUT  VARIABLE NAMES(UP TO 8 CHARACTERS EACH)"
      DO 220 I=1,NVAR
      PRINT*,"X",I,":"
  220 READ(5,12)  ILABEL(I+1)
      PRINT*,"INPUT  CONSTRAINT NAMES(UP TO 8 CHARACTERS EACH)"
      DO 230 I=1,NCON
      PRINT*,"CONST ",I,":"
  230 READ(5,12)  ILABEL(I+NVAR+1)
      GO TO 290
250   DO 260  I= 1,NVAR
  260 ENCODE(10,263,ILABEL(I+1)) I
  263 FORMAT(4H  X(,I2,4H)   )
      ILABEL(1)=10H  Z
      DO 270 I=1,NCON
  270 ENCODE(10,274,ILABEL(NVAR+I+1)) I
  274 FORMAT(7H  CNST#,I2,1H )
290   CALL KEEPNUM
300   ICRSHCT=3
  301 CALL SAY(9)
      READ(5,13)  ITYPE
      IF(ITYPE.NE.3HMIN.AND.ITYPE.NE.3HMAX) CALL HELP( 5),RETURNS(301)
  302 CALL ASK(11,IDUM),RETURNS(9999,305,303,303)
  303 CALL HELP( 8),RETURNS(302)
  305 CALL SAY(4)
      READ*,(WORK(I,1),I=1,NVAR)
      CALL ECHO(0)
      IRCNT=0
      CALL ASK( 6,IDUM),RETURNS(9999,320,302,302)
  320 CALL KEEPNUM
400   ICRSHCT=4
      IRCNT=IRCNT+1
  401 WRITE(6,402) IRCNT
  402 FORMAT(" INPUT COEFFICENTS FOR CONSTRAINT # ",I2,4H NOW,/)
      READ*,(WORK(I,1+IRCNT),I=1,NVAR)
403   CALL SAY(26)
      READ(5,13)  IRGT
      IF(IRGT.NE.2HGE.AND.IRGT.NE.2HLE.AND.IRGT.NE.2HEQ)CALL HELP( 6),
     X         RETURNS(403)
      ILABEL(NVAR+1+IRCNT)=(ILABEL(NVAR+1+IRCNT).AND..NOT.MASK(12)).OR.
     X                     (IRGT.AND.MASK(12))
      PRINT*,"INPUT RIGHT HAND SIDE NOW. B(",IRCNT,")="
      READ*,WORK(NVAR+1,1+IRCNT)
      CALL KEEPNUM
      IF(IRCNT.EQ.NCON)GO TO 520
      GO TO 400
```

```
520   ICRSHCT=5
      GO TO 700
525   CALL SAY(7)
      CHANGE=.TRUE.
531   CALL INTNUM(20,OPTION,19),RETURNS(9999,533,532)
532   CALL HELP(9),RETURNS(531)
533   CALL KEEPNUM
      GO TO (600,700,630,1900,2000,2100,2200,2800,2900,
     X       1100,1200,1300,1400,1500,1600,1700,1800,3000,132)OPTION
600   RETURN
630   CALL SAY(8)
      GO TO 531
C     * * * * * * * * *  ECHO THE PROBLEM * * * * * * * * *
700   CALL SAY(5)
      CALL ECHO(0)
      CALL SAY(6)
      DO 710 I= 1,NCON
      CALL ECHO(I)
710   CONTINUE
      IF(CHANGE)GO TO 531
      GO TO 525
C * * * * * * * * * * * * * *  CHANGE THE VARIABLES NAMES * * * * * *
1100  ILBL=1HY
 1101 CALL INTNUM( 3,INMOD,NVAR),RETURNS(9999,1105,1103)
 1103 CALL HELP( 2),RETURNS(1101)
1105  DO 1150 I=1,INMOD
 1106 CALL INTNUM( 4,J,NVAR),RETURNS(9999,1110,1108)
 1108 CALL HELP( 2),RETURNS(1106)
1110  PRINT*,"NEW NAME(UP TO 8 CHARACTERS)... "
      READ(5,12)   ILABEL(J+1)
 1150 CONTINUE
      GO TO 531
C * * * * * * * * * * CHANGE THE CONSTRAINT NAMES * * * * * * * * * *
1200  ILBL=1HY
 1201 CALL INTNUM( 5,INMOD,NCON),RETURNS(9999,1205,1203)
 1203 CALL HELP( 3),RETURNS(1201)
1205  DO 1250 I=1,INMOD
 1206 CALL INTNUM( 6,J,NCON),RETURNS(9999,1210,1208)
 1208 CALL HELP( 3),RETURNS(1206)
1210  PRINT*,"NEW NAME(UP TO 8 CHARACTERS)... "
      READ(5,12)   ILHOLD
      ILABEL(NVAR+J+1)=(ILABEL(NVAR+J+1).AND.MASK(12))
     X                 .OR.(ILHOLD.AND..NOT.MASK(12))
 1250 CONTINUE
      GO TO 531
C * * * * * * * * * * *  CHANGE THE OBJECTIVE NAME * * * * * * * * * *
1300  PRINT*,"NEW OBJECTIVE NAME..."
      READ(5,12)      ILABEL(1)
      GO TO 531
C * * * * * * * * * * * CHANGE MIN TO MAX OR VICE VS * * *
1400  PRINT*,"INPUT NEW OPTIMUM (MIN,MAX)..."
      READ(5,13)   ITYPE
      IF(ITYPE.NE.3HMIN.AND.ITYPE.NE.3HMAX)CALL HELP( 5),RETURNS(1400)
      GO TO 531
```

```
C * * * * * *  CHANGE AN OBJECTIVE COEFFICIENT * * * * * **
1500  CALL INTNUM( 7,INMOD,NVAR),RETURNS(9999,1505,1503)
 1503 CALL HELP( 2),RETURNS(1500)
1505  DO 1550 I=1,INMOD
 1507 CALL INTNUM( 8,J,NVAR),RETURNS(9999,1510,1508)
 1508 CALL HELP( 2),RETURNS(1507)
 1510 PRINT*,"NEW VALUE..."
 1550 READ*,WORK(J,1)
      GO TO 531
C * * * * * * * * CHANGE THE GE LE OR EQ RELATIONS * * * * * * * * * * *
1600  CALL INTNUM( 9,INMOD,NCON),RETURNS(9999,1605,1603)
 1603 CALL HELP( 3),RETURNS(1600)
1605  DO 1650 I=1,INMOD
 1607 CALL INTNUM(10,J,NCON),RETURNS(9999,1610,1608)
 1608 CALL HELP( 3),RETURNS(1607)
1610  PRINT*,"INPUT (GE,LE,EQ) ..."
      READ(5,13)   ILHOLD
      ILABEL(NVAR+J+1)=(ILABEL(NVAR+J+1).AND..NOT.MASK(12))
     X                 .OR.(ILHOLD.AND.MASK(12))
 1650 CONTINUE
1660  CALL SAY(20)
      READ(5,*)NGREAT
      IF(NGREAT.LT.0.OR.NGREAT.GT.NCON)CALL HELP(3),RETURNS(1660)
      GO TO 531
C * * * * * * * * *  CHANGE A RIGHT HAND SIDE VALUE
1700  CALL INTNUM(11,INMOD,NCON),RETURNS(9999,1705,1703)
 1703 CALL HELP( 3),RETURNS(1700)
1705  DO 1750  I=1,INMOD
 1707 CALL INTNUM(12,J,NCON),RETURNS(9999,1710,1708)
 1708 CALL HELP( 3),RETURNS(1707)
1710  PRINT*,"NEW RIGHT HAND SIDE VALUE="
 1750 READ*,WORK(NVAR+1,J+1)
      GO TO 531
C * * * * * * * * * *  CHANGE A CONSTRAINT COEFFICIENT * * * * * * *
1800  NCONVAR=NCON*NVAR
      CALL INTNUM(13,INMOD,NCONVAR),RETURNS(9999,1805,1803)
 1803 CALL HELP( 4),RETURNS(1800)
1805  DO 1850 I=1,INMOD
1806  PRINT*,"WHICH CONSTRAINT? (INTEGER)..."
      READ*,J
      PRINT*,"WHICH DECISION VARIABLE? (INTEGER)..."
      READ*,JI
      CALL CKINT(J,NCON),RETURNS(1807,1808,9999)
1807  CALL CKINT(JI,NVAR),RETURNS(1811,1809,9999)
1808  CALL HELP( 3),RETURNS(1806)
1809  CALL HELP( 2),RETURNS(1806)
1811  PRINT*,"NEW A(",J,",",JI,")="
1850  READ*,WORK(JI,J+1)
      GO TO 531
```

```
C * * * * * * * * * * * ADD A VARIABLE * * * * * * * * * * * * * * *
1900  CALL ASK( 7,IDUM),RETURNS(9999,1910,531,1907)
1907  CALL HELP(7),RETURNS(1900)
1910  NVAR=NVAR+1
      NVARP1=NVAR+1
      DO 1912 J=2,NCONP1
1912  WORK(NVAR+1,J)=WORK(NVAR,J)
      CALL SAY(10)
      READ*,(WORK(NVAR,J),J=1,NCONP1)
      DO 1920 I=1,NCON
      J=NVAR+NCON+1-I
1920  ILABEL(J+1)=ILABEL(J)
      IF(ILBL.EQ.1HN)GO TO 1940
      CALL SAY(11)
      READ(5,12)ILABEL(NVAR+1)
      GO TO 1950
1940  ENCODE(10,1981,ILABEL(NVAR+1))NVAR
1950  WRITE(6,1960)NVAR
1960  FORMAT(19H DECISION VARIABLE ,I2,6H ADDED)
      GO TO 531
 1981 FORMAT(4H  X(,I2,4H)   )
C  * * * * * * * * * * * * * * * * DELETE A VARIABLE
2000  CALL ASK(8,IDUM),RETURNS(9999,2005,531,2003)
2003  CALL HELP(7),RETURNS(2000)
2005  CALL INTNUM(15,INMOD,NVAR),RETURNS(9999,2010,2008)
2008  CALL HELP(2),RETURNS(2005)
2010  IUPR=NVAR+NCON-1
      NCONP1=NCON+1
      DO 2030 I= INMOD,IUPR
2030  ILABEL(I+1)=ILABEL(I+2)
2032  JUPR=NCON+1
      DO 2040 I= INMOD,NVAR
      DO 2040 J= 1,NCONP1
 2040 WORK(I,J)=WORK(I+1,J)
      NVAR=NVAR-1
      NVARP1=NVAR+1
      IF(ILBL.EQ.1HY)GO TO 2060
      DO 2050 I=INMOD,NVAR
      ENCODE(10,2043,ILABEL(I+1))I
 2043 FORMAT(4H  X(,I2,4H)   )
 2050 CONTINUE
2060  CALL SAY(12)
      GO TO 531
```

```
C * * * * * * * * * *  ADD A CONSTRAINT * * * * * * * * * * * * * * *
2100  CALL ASK(9,IDUM),RETURNS(9999,2131,531,2107)
2107  CALL HELP(7),RETURNS(2100)
2131  NCON=NCON+1
      NCONP1=NCON+1
      CALL SAY(13)
      READ*,(WORK(I,NCONP1),I=1,NVAR)
      CALL SAY(15)
      READ(5,13) ILABEL(NVAR+NCONP1)
      CALL SAY(14)
      READ*,WORK(NVAR+1,NCONP1)
      IF(ILBL.EQ.1HY) GO TO 2150
      ITEMP=ILABEL(NVAR+NCONP1).AND.MASK(12)
      ENCODE(10,2136,ILABEL(NVAR+NCONP1)) ITEMP,NCON
2136  FORMAT(A2,"CNST#",I2)
      GO TO 2160
2150  CALL SAY(16)
      READ(5,  12) ITEMP
      ILABEL(NVAR+NCONP1)=(ILABEL(NVAR+NCONP1).AND.MASK(12))
     X             .OR.(ITEMP.AND..NOT.MASK(12))
2160  WRITE(6,2170)NCON
2170  FORMAT(12H CONSTRAINT ,I2,6H ADDED)
      GO TO 1660
C * * * * * * * * * DELETE A CONSTRAINT   * * * * * * * * * * * *
2200  CALL ASK(10,IDUM),RETURNS(9999,2205,531,2203)
2203  CALL HELP(7),RETURNS(2200)
2205  CALL INTNUM(17,INMOD,NCON),RETURNS(9999,2210,2207)
2207  CALL HELP(3),RETURNS(2205)
2210  IUPR=NVAR+1
      JUPR=NCON-1
      IF(INMOD.EQ.NCON) GO TO 2240
      DO 2220 J= INMOD, JUPR
      ILABEL(J+NVAR+1)=ILABEL(J+NVAR+2)
      DO 2220 I= 1,IUPR
2220  WORK(I,J+1)= WORK(I,J+2)
2240  NCON=NCON-1
      NCONP1=NCON+1
      IF(ILBL.EQ.1HY) GO TO 2260
      DO 2250 J=1,NCON
      ITEMP=ILABEL(J+1+NVAR).AND.MASK(12)
2250  ENCODE(10,2136,ILABEL(J+1+NVAR)) ITEMP,J
2260  CALL SAY(12)
      GO TO 1660
C * * * * * * * * *  RESCALE A VARAIBLE * * * * * * * * * * * * * *
2800  CALL INTNUM(18,INMOD,NVAR),RETURNS(9999,2810,2805)
2805  CALL HELP( 2),RETURNS(2800)
2810  CALL SAY(17)
      READ*,SVAR
      JU=NCON+1
      DO 2820 J=1,JU
2820  WORK(INMOD,J)= WORK(INMOD,J)*SVAR
      GO TO 531
```

```
C * * * * * * * * * * * *  RESCALE A CONSTRAINT * * * * * * * * * * * *
2900  CALL INTNUM(19,INMOD,NCON),RETURNS(9999,2910,2905)
2905  CALL HELP( 3),RETURNS(2900)
2910  CALL SAY(17)
      READ*,SCON
      IU=NVAR+1
      INMOD=INMOD+1
      DO 2920 I=1,IU
2920  WORK(I,INMOD)= WORK(I,INMOD)*SCON
      GO TO 531
3000  CALL KEEPNUM
      IQUIT=.TRUE.
      CALL SAVE(IQUIT)
      CALL SAY(27)
      STOP
9999  CONTINUE
      CALL SAY(3)
      CALL KEEPNUM
      STOP " USER ABORT IN THEDATA"
      END
```

```
      SUBROUTINE TLPKODE,RETURNS(N)
C     ***********************************************************************
C     *  THIS SUBROUTINE REFORMATS DATA FOR INPUT TO LPSOLVE                *
C     ***********************************************************************
      COMMON//WORK(99,99),ILABEL(199),ITITLE(6)
      COMMON/FLAGS/IEXPERT,NEWP,ICRASH,IMODFY,ISDATA,IINTEG
      COMMON/COUNTS/ICRSHCT,NVAR,NCON,NINT,NAME,IBOX,ILBL,ITYPE,IRCNT,
     X  IRGT,ISCALE,IPOPT,IDRIVE,NGREAT,UNIT,HD,SN,IBANER,II(2)
      DIMENSION IFMT(2),IOUT(7)
      INTEGER UNIT
      CALL GETNUM
      REWIND 1
      REWIND 3
      IU=NVAR+1
      JU=NCON+1
12    DO 15 I=1,IU
      DO 15 J=1,JU
      IF((WORK(I,J).GE.999999.5).OR.(WORK(I,J).LE.-99999.5))GO TO 20
15    CONTINUE
      GO TO 30
20    PRINT*,"YOU HAVE A SCALE PROBLEM IN COEF(",I,J-1,")=",WORK(I,J)
      PRINT*,"FIX IT WITH OPTION  8 OR  9 "
      RETURN
30    WRITE(1,35) (ITITLE(I),I=1,6)
      WRITE(3,35) (ITITLE(I),I=1,6)
35    FORMAT(7A10)
      IFLBL=0
      IF(ILBL.EQ.1HY) IFLBL=1
      ME=4HHERE
50    FORMAT(3I2,A3,1X,2I1,I2,I1,2A1,A3,A4)
      WRITE(1,50)NCON,NVAR,IPOPT,ITYPE,IFLBL,IDRIVE,NGREAT,UNIT,HD,SN
     1,IBANER,ME
      WRITE(3,50)NCON,NVAR,IPOPT,ITYPE,IFLBL,IDRIVE,NGREAT,UNIT,HD,SN
     1,IBANER
      IFMT(1)=10H(I2,I2,F6.
      JFMT=10H(I2,I2,I6)
      KFMT=10H(I2,A2,I6)
      M=1
      DO 60 J=1,JU
      DO 60 I=1,NVAR
      IF(M.LT.8) GO TO 53
      M=1
      WRITE(1,35) (IOUT(NN),NN=1,7)
      WRITE(3,35) (IOUT(NN),NN=1,7)
53    WT=WORK(I,J)
      IF(WT.EQ.0)GO TO 60
      CALL GETK(WT,K)
      IF(K.EQ.-1)GO TO 55
      ENCODE(10,54,IFMT(2)) K
54    FORMAT(I1,")        ")
      ENCODE(10,IFMT(1),IOUT(M)) J-1,I,WT
      GO TO 59
```

```
55    IF(WT.GT.0)IWT=WT+.5
      IF(WT.LT.0)IWT=WT-.5
      ENCODE(10,JFMT,IOUT(M))J-1,I,IWT
59    M=M+1
60    CONTINUE
      K=M-1
      IF(M.NE.1) WRITE(1,35) (IOUT(NN),NN=1,K)
      IF(M.NE.1) WRITE(3,35) (IOUT(NN),NN=1,K)
      WRITE(1,63)
      WRITE(3,63)
63    FORMAT(4H-1-1)
      M=1
      DO 70 J=2,JU
      WT= WORK(NVAR+1,J)
      IFMT(1) = 10H(I2,A2,F6.
      ITEMP=ILABEL(NVAR+J).AND.MASK(6).OR.(0024000000000000000B)
      CALL GETK(WT,K)
      IF(M.LT.8) GO TO 65
      M=1
      WRITE(1,35) (IOUT(NN),NN=1,7)
      WRITE(3,35) (IOUT(NN),NN=1,7)
65    IF(K.EQ.-1)GO TO 68
      ENCODE(10,66,IFMT(2))K
66    FORMAT(I1,")          ")
      ENCODE(10,IFMT(1),IOUT(M)) J-1,ITEMP,WT
      GO TO 69
68    IF(WT.GT.0)IWT=WT+.5
      IF(WT.LT.0)IWT=WT-.5
      ENCODE(10,KFMT,IOUT(M))J-1,ITEMP,IWT
69    M=M+1
70    CONTINUE
      K=M-1
      IF(M.NE.1) WRITE(1,72) (IOUT(NN),NN=1,K)
      IF(M.NE.1) WRITE(3,72) (IOUT(NN),NN=1,K)
72    FORMAT(7A10)
      WRITE(1,73)
      WRITE(3,73)
73    FORMAT(2H-1)
      IF(ILBL.EQ.1HN) RETURN N
      WRITE(1,80) (ILABEL(NVAR+1+I),I=1,NCON)
      WRITE(3,80) (ILABEL(NVAR+1+I),I=1,NCON)
      WRITE(1,80) (ILABEL(I+1),I=1,NVAR)
      WRITE(3,80) (ILABEL(I+1),I=1,NVAR)
80    FORMAT(8R8)
      RETURN N
      END
```

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFIT/GOR/SM/79D-7	2. GOVT ACCESSION NO. AD-A083707	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) LPAFIT AN INTERACTIVE LINEAR PROGRAMMING PACKAGE DEVELOPED AT THE AIR FORCE INSTITUTE OF TECHNOLOGY		5. TYPE OF REPORT & PERIOD COVERED MS THESIS 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Michael A. Schiefer Captain USAF		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Air Force Institute of Technology (AFIT/EN) Wright-Patterson AFB, Ohio 45433		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Department of Systems Management AFIT/EN		12. REPORT DATE December 1979 13. NUMBER OF PAGES 169
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

APPROVED FOR PUBLIC RELEASE AFR 190-17.

Joseph P. Hipps, Major, USAF
Director of Information

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Computer Programs
Operations Research Computer Programs
Interactive Computer Programs

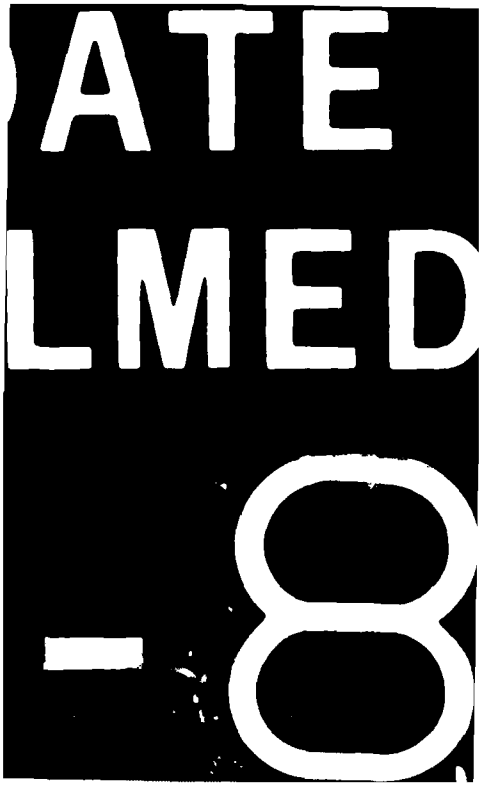
20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This thesis effort produced a linear programming package, called LPAFIT. Two FORTRAN programs were written. The first, LPSOLVE, solves linear programming problems using the simplex method. The second program, LPFRONT, is a preprocessor for LPSOLVE. LPFRONT assists the user in creating the input for LPSOLVE. A procedure controls all file manipulations. The only thing a user must do is express a problem in terms of a constrained objective function. The routines do not solve integer or mixed integer problems. The package runs on a Control Data Corporation 6600.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)


ATE
LMED
8