

INFORMATION SYSTEMS LABORATORY

STANFORD ELECTRONICS LABORATORIES  
DEPARTMENT OF ELECTRICAL ENGINEERING  
STANFORD UNIVERSITY · STANFORD, CA 94305



SEL-79-019

79-0557

LEVEL

COMPUTATIONAL ISSUES IN LINEAR LEAST-SQUARES

ESTIMATION AND CONTROL

By

John A. Newkirk

Technical Report No. M355-7

June 6, 1979

DTIC  
ELECTE  
SEP 22 1980  
S A D

Research Sponsored by  
Defense Advanced Research Project Agency

Contract Number: MDA 903-78-C-0179

and

National Science Foundation

Contract Number: ENG 78-10003

DDC FILE COPY

AD A089350

DISTRIBUTION STATEMENT A

Approved for public release;  
Distribution Unlimited

80 9 19 057

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM	
1. REPORT NUMBER 14 SU-SEL-79-019	2. GOVT ACCESSION NO. M355-7 (A) A089350	3. RECIPIENT'S CATALOG NUMBER (9) Technical report	
4. TITLE (and Subtitle) 6 COMPUTATIONAL ISSUES IN LINEAR LEAST-SQUARES ESTIMATION AND CONTROL		5. TYPE OF REPORT & PERIOD COVERED Tech. Report	
7. AUTHOR(s) 10 John A. Newkirk		6. PERFORMING ORG. REPORT NUMBER M355-7	
9. PERFORMING ORGANIZATION NAME AND ADDRESS Stanford University Stanford Electronics Labs Stanford, CA 94305		8. CONTRACT OR GRANT NUMBER(s) MDA 903-78-C-0179 ✓ ARPA Order-3506	
11. CONTROLLING OFFICE NAME AND ADDRESS DARPA 1400 Wilson Blvd. Arlington, VA 22209		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS AO 3506	
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) 12 250		12. REPORT DATE 11 6 June 1979	
		13. NUMBER OF PAGES 248	
		15. SECURITY CLASS. (of this report) Unclas.	
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE	
16. DISTRIBUTION STATEMENT (of this Report) This document is approved for public release and sale; distribution is unlimited. This document may be reproduced for any purpose of the United States Government.			
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report) NA			
18. SUPPLEMENTARY NOTES NA			
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Scattering theory Computers Ricatti Equation Communications and control			
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) (U) The dissertation focuses on the steady-state solution to the linear estimation and control problems. Following a pertinent review of linear algebra and computation fundamentals, various approaches to solving the matrix Riccati equation are examined. After reviewing eigenvector decomposition and various iterative algorithms, square root doubling algorithms are motivated. Scattering theory is used to initiate an algebraic derivation of these algorithms, and is then extended to provide a pure derivation of several square root algorithms.			

A set of criteria for choosing among these algorithms is presented and examined with empirical evaluations. Differentiating criteria include sensitivity to repeated closed-loop eigenvalues, the impact of singular model parameters, computational accuracy, and rate of convergence.

Sensitivity to parameter uncertainty in discrete-time systems is considered using a quadratic minimization of a generalized cost function. This same algorithm is used to design arbitrary-order compensation using complete system information.

Algorithm implementation is then considered in terms of both present and future hardware.

In conclusion, for discrete-time systems, the square root doubling algorithms are found to be comparable to the eigenvector decomposition algorithms in terms of memory requirements, elapsed computation time, and performance. The doubling algorithms are also found to solve a wider class of problems.



COMPUATIONAL ISSUES IN LINEAR LEAST-SQUARES  
ESTIMATION AND CONTROL

By

John A. Newkirk

Technical Report No. M355-7

June 6, 1979

Research Sponsored by  
Defense Advanced Research Project Agency

Contract Number: MDA 903-78-C-0179

and

National Science Foundation  
Contract Number: ENG 78-10003

DISTRIBUTION STATEMENT A  
Approved for public release;  
Distribution Unlimited

Accession For	
NTIS GRANT	<input checked="checked" type="checkbox"/>
DDC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
<i>also letter on</i>	
By <i>file</i>	
Distribution/	
Availability Codes	
Dist.	Avail and/or special
<i>A</i>	



© Copyright 1978

by

JOHN ANTHONY NEWKIRK




## ABSTRACT

The dissertation focuses on the steady-state solution to the linear estimation and control problems. Following a pertinent review of linear algebra and computation fundamentals, various approaches to solving the matrix Riccati equation are examined. After reviewing eigenvector decomposition and various iterative algorithms, square root doubling algorithms are motivated. Scattering theory is used to initiate an algebraic derivation of these algorithms, and is then extended to provide a pure derivation of several square root algorithms.

A set of criteria for choosing among these algorithms is presented and examined with empirical evaluations. Differentiating criteria include sensitivity to repeated closed-loop eigenvalues, the impact of singular model parameters, computational accuracy, and rate of convergence.

Sensitivity to parameter uncertainty in discrete-time systems is considered using a quadratic minimization of a generalized cost function. This same algorithm is used to design arbitrary-order compensation using complete system information.

Algorithm implementation is then considered in terms of both present and future hardware.

In conclusion,  for discrete-time systems, the square root doubling algorithms are found to be comparable to the eigenvector decomposition algorithms in terms of memory requirements, elapsed computation time, and performance. The doubling algorithms are also found to solve a wider class of problems.

## ACKNOWLEDGEMENTS

My highest accolades are for my advisor, Professor Martin Morf. His patience and understanding, combined with an inspiring ability to see problems in a new light or from a fresh perspective, was of indescribable value. I am also grateful for the many hours he devoted to carefully reviewing this manuscript.

I must also specially acknowledge Professor Thomas Kailath for his perceptive feedback as second reader; his suggestions and comments were extremely valuable.

Of nearly equal importance to me has been the support of my close friends and family. I am especially indebted to my brother, Christopher Newkirk, for his immense effort in drafting the drawings found herein, and to my very good friend Abbas El Gamal for being of good humor and stout heart. Support was also gratefully found in Richard Kriesel, a friend of long standing, and in my parents; it is to my parents, each individually -- John and Audrey Newkirk -- that this thesis is dedicated.

Many more at Stanford deserve generous acknowledgement including :

Dr. Jon Claerbout, for years of support and for serving as third reader;

Dr. David Powell, for introducing me to digital control and thus launching this effort;

Dr. Robert Mathews, for a lifetime of stimulating conversations and intellectual adventures.

I am grateful for the financial support received from the Defense Advanced Research Projects Agency (DARPA) under contract MDA 903-78-C-0179. Support was also received from the Stanford Exploration Project (SEP) through Professor J. Claerbout.

Finally, I would like to thank Mrs. Dorothy Gundy for her expert typing of this dissertation.



# TABLE OF CONTENTS

	<u>Page</u>
ABSTRACT.....	iv
ACKNOWLEDGMENTS.....	v
TABLE OF CONTENTS.....	vi
LIST OF ILLUSTRATIONS.....	viii
LIST OF TABLES.....	xi
I. INTRODUCTION TO LINEAR ESTIMATION.....	1
I.1 General Introduction.....	1
I.2 Dissertation Outline.....	4
II. MATRIX FUNDAMENTALS.....	6
II.1 Introduction.....	6
II.2 The Fundamentals.....	7
III. THE STEADY-STATE SOLUTION TO THE DISCRETE RICCATI EQUATION.....	26
III.1 Introduction.....	26
III.2 Iteration.....	31
III.3 Eigenvector Decomposition.....	33
III.4 Square Roots and Orthogonal Transformations.....	37
III.5 Square Root Algorithms.....	42
III.6 Scattering Theory.....	48
III.7 Square Root Doubling.....	67
III.8 Scattering Revisited.....	78
III.9 Algorithm Revisions.....	94
IV. THE STEADY-STATE SOLUTION TO THE CONTINUOUS RICCATI EQUATION.....	98
IV.1 Introduction.....	98
IV.2 Eigenvector Decomposition.....	100
IV.3 Square Root Doubling.....	101
IV.4 Conclusions.....	105
V. DIFFERENTIATING APPLICATIONS.....	106
V.1 Introduction.....	106
V.2 Reduced-Order Observers.....	108
V.3 Repeated Closed-Loop Eigenvalues.....	124
VI. EMPIRICAL RESULTS.....	127
VI.1 Introduction.....	127
VI.2 Memory Requirements.....	128
VI.3 Computation Time.....	130
VI.4 Performance.....	139
VI.5 Convergence Sensitivity.....	139
VI.6 Conclusions.....	143
VII. PARAMETER SENSITIVITY IN DISCRETE SYSTEMS.....	145
VII.1 Introduction.....	145
VII.2 Chapter Outline.....	146
VII.3 Sensitivity Equations for Discrete Systems.....	147
VII.4 Augmented Observability and Controllability.....	155
VII.5 Application to Parameter Desensitization.....	157
VII.6 Applications.....	161

# TABLE OF CONTENTS (Cont)

	<u>Page</u>
VIII. REDUCED-ORDER COMPENSATORS.....	171
VIII.1 Introduction.....	171
VIII.2 Design of Reduced-Order Compensation.....	172
VIII.3 Comparison -- Gradient Search and "Simplification" Design.....	185
VIII.4 An Example.....	191
IX. IMPLEMENTATION ISSUES.....	193
IX.1 Introduction.....	193
IX.2 Algorithm Implementation.....	194
IX.3 Parallel Implementation.....	195
IX.4 Special Hardware.....	196
X. CONCLUSIONS.....	199
Appendix A: SOFTWARE.....	202
A1 Software for Solving the Steady-State Riccati Equation.....	202
Appendix B: TEXT ELABORATIONS.....	207
B1 Design of a Discrete Estimator for the Space Lab Infrared Telescope.....	207
Appendix C: EXAMPLE PROBLEM FORMULATIONS.....	212
C1 F-H Aircraft.....	213
C2 Star Tracking Telescope.....	223
C3 Paper Mill Machine.....	227
C4 Relativity Satellite.....	231
REFERENCES.....	234

# LIST OF ILLUSTRATIONS

<u>Figure</u>		<u>Page</u>
2.1	A Householder Reflection in Three Dimensions . . . . .	19
2.2	A Householder Reflection in Two Dimensions . . . . .	19
2.3	Schematic Application of a Householder Transformation .	21
3.6.1	Canonical Scattering Layer . . . . .	51
3.6.2	Concatenation of Layers . . . . .	51
3.6.3	Domain Translation . . . . .	51
3.6.4	Hamiltonian Scattering Layer . . . . .	61
3.6.5	Hamiltonian Transition Matrix . . . . .	61
3.6.6	Eigenlayers . . . . .	63
3.6.7	Accumulated Covariance Layer . . . . .	66
3.6.8	Update Layer . . . . .	66
3.7.1	Initial Interval . . . . .	68
3.7.2	Interval Segmentation . . . . .	68
3.7.3	Interval Description . . . . .	70
3.7.4	Doubling Layers . . . . .	72
3.8.1	Scattering Hamiltonian for the Estimation Problem . . .	79
3.8.2	Transition Domain Block Diagram . . . . .	80
3.8.3	Square Root Block Diagram . . . . .	80
3.8.4	Square Root Scattering Hamiltonian . . . . .	81
3.8.5	Orthogonal Mappings . . . . .	81
3.8.6	J-Unitary Mappings . . . . .	82
3.8.7	Square Root Star Product Definition . . . . .	83
3.8.8	Square Root Star Product Definition . . . . .	83
3.8.9	Detail of Combined Layer . . . . .	84



<u>Figure</u>		<u>Page</u>
3.8.10	Simplified Boundary Mapping . . . . .	85
3.8.11	Combined Layer in Canonical Form . . . . .	85
3.8.12	Definition of Loop Unwinding . . . . .	86
3.8.13	Transformation to Canonical Form . . . . .	87
3.8.14	Elimination of Feedback Term . . . . .	87
3.8.15	Final Result . . . . .	89
3.8.16-3.8.20	Application to Square Root Doubling . . . . .	89-91
3.8.21-3.8.22	Application to Normal Square Root . . . . .	92-93
5.1	Alternative Realizations for System with Pure Delay on Output . . . . .	112
5.2	Alternative Estimators for System with Pure Delay on Output . . . . .	114
5.3	Alternative Closed-loop Realizations for System with Pure Delay . . . . .	115
5.4	Locus of Estimator Roots vs. Disturbance Covariance . . . . .	117
5.5	Estimator Error Covariance . . . . .	117
5.6	Comparison of Covariance Terms . . . . .	120
5.7	Locus of Estimator Roots vs. Disturbance Covariance, from Powell [1978] . . . . .	121
5.8	Estimator Error Covariance, from Powell [1978] . . . . .	122
5.9	Conservative System Estimator Root Locus . . . . .	125
6.1	Relative Elapsed Computation Time, Trial I . . . . .	132
6.2	Relative Elapsed Computation Time, Trial II . . . . .	132
6.3	Elapsed Computation Time for Square Root Doubling . . . . .	134
6.4	Elapsed Computation Time for Eigenvector Decomposition . . . . .	135
6.5	Relative Elapsed Computation Time, Trial III . . . . .	136
6.6	Square Root Doubling Convergence . . . . .	138

<u>Figure</u>		<u>Page</u>
6.7	Convergence of Bilinear Square Root Doubling . . . . .	141
6.8	Bilinear Square Root Doubling Convergence . . . . .	142
7.1	Desensitization Locus . . . . .	164
7.2	Aircraft Parameter Uncertainty Desensitization . . . . .	169
8.1	Sample Problem Bode Plot . . . . .	175
8.2	Classical Designs . . . . .	177
8.3	'Optimal' Design Comparison, Trial I Bode Plots . . . . .	187
8.4	'Optimal' Design Comparison, Trial II Bode Plots . . . . .	188
8.5	'Optimal' Design Comparison, Trial II Nyquist Plots . . . . .	189
8.6	Star-Tracking Telescope Compensation Optimization . . . . .	192
9.1	Array Processor Performance . . . . .	197
A1.1	Sample Program Input File . . . . .	206
C2.1	Optimal Drift Estimation and Gyro Noise Model . . . . .	224
C4.1	Simplified Model of the Stanford Relativity Satellite . . . . .	232

## LIST OF TABLES

<u>Table</u>		<u>Page</u>
3.1	Comparative Computational Overhead . . . . .	96
3.2	Basic Algorithm Overhead . . . . .	97
6.1	Main Memory Storage Requirements . . . . .	129
7.1	Results of Program Verification . . . . .	163
8.1	Summary of Design Trials . . . . .	183
8.2	Simplification Equivalence . . . . .	186
A1.1	Implemented Software . . . . .	205
C1.1	Coefficient References . . . . .	218



## Chapter I

## INTRODUCTION

Consideration of how to interpret measurements and of how to make predictions based on these measurements has a long, colored, and oft obscure history ("Hamlet's Mill," Santillana [1969]). It has been a primary concern of orderly enquiry since at least the earliest megalithic cultures. But only much later, with the dawning of modern science in the late middle ages, did measurement and considered prediction become a central paradigm in rational enquiry ("The Waning of the Middle Ages," Huizinga [1924]), ("The Civilization of the Renaissance in Italy," Burckhardt [1860]).

Refinements have often come about in unexpected ways, many of them fascinating. Sir Isaac Newton considered the verification of the law of universal gravitation fundamental to his theory of motion-- so fundamental he strove for unparalleled precision. Beginning with data accurate to five percent, Newton proceeded to derive a correlation between the acceleration of gravity and the earth's attraction for the moon accurate to one part in 3000 [Newton, 1726] -- a persuasive correlation! Unfortunately, his editor (Professor Roger Cotes) found a grievous inconsistency in the calculation, and replied [Edleston, 1850]

I have considered how to make [it] appear to the best advantage as to  $y^e$  numbers, and I propose to alter it thus.

With unbridled faith in his mentor's theory, Cotes proceeded to alter the latitude of Paris. Sir Isaac declined the offer (choosing instead to alter the orbit of the moon).

Measurement and estimation continued to advance in the early eighteenth century, but in 1795 Gauss introduced linear least-squares estimation, and progress began in earnest. The past forty years has seen a burgeoning of the field, beginning with work by Kolmogorov, Krein, and Wiener. For a comprehensive history, see Kailath [1974].

Estimation theory, in its modern context, addresses the issue of making predictions from observations. We begin by assuming a stochastic process,  $x(\cdot)$ , and then try to prognosticate inferences about  $x$  based upon measurements of a related process  $y(\cdot)$ . In linear estimation theory, these inferences are linear functionals of the measurements. If the objective is to estimate  $x$ , and if the performance criterion is to minimize the mean square error between  $x$  and its estimate,  $\hat{x}$ , then the solution simplifies. The theory becomes quite tractable, depending solely on the first and second moments of the original and of the observed process.

In the discrete-time case, the solution follows immediately with the inversion of the observation process covariance matrix. However, since  $n$  observations require on the order of  $n^3$  operations for the inversion, this can become a Herculean task for even the largest computer.

An alternate approach uses recursive filters, which update state information as new observations become available. There are several formulations, but Kalman filters are the best known of these formulae. Using the state space model, Kalman [1960] [1963] formulated the process as a linear dynamical system driven by white noise. The observation process,  $y$ , is a function of the state process,  $x$ , and is similarly corrupted by white noise.

Arising in the solution of this recursive problem is a nonlinear, matrix difference equation, an example of the Riccati type equations. The solution to this equation is of primary interest in the sequel.

This dissertation will focus on discrete-time problems; the signal importance digital computers have attained in our current technology often obviates continuous alternatives. Further, the benefits of digital implementations are persuasive. Costs can be reduced, flexibility is increased, greatly increased accuracy is possible, and model complexity can often be incorporated readily. There are also considerable disadvantages, arising because of the interface with a continuous world.

Two primary factors in determining costs, in both the design and the implementation phase, are speed and complexity; if a system is slow and simple, then it is usually less expensive. In the implementation context, reducing cost implies reducing the order of the compensation, the accuracy of the computation, and reducing the sample rate (usually at the expense of the system bandwidth and sensitivity). In the design context, reducing cost implies faster, simpler algorithms and synergetic hardware.

In the sequel, various design algorithms are considered from several different analytical perspectives. Both theoretical and empirical comparisons are made. Sensitivity and compensation complexity are also considered. The relationship between algorithms and hardware is presented as the denouement.



### Dissertation Outline

Chapter 2 is a brief and fast-paced review of linear algebra, numerical analysis, and fundamental algorithms. Definitions are presented, concepts are outlined, and references are given for additional background, if required.

Chapter 3 provides the gist of the dissertation. Half of the chapter is a review of earlier work. It begins by developing initial criteria for gauging the algorithms of the sequel; standard iteration of the matrix Riccati equation is prescribed as a benchmark. Next, a standard algorithm for solving the steady-state Riccati equation is reviewed (eigenvector decomposition).

Before presenting another class of algorithms, orthogonal transformations are considered in pertinent detail; these mappings are then used in an algebraic derivation of square root algorithms. Several questions arise, and are deferred.

Scattering theory is then introduced and applied to the iteration of the Riccati equation and to the eigenvector decomposition developed earlier. Scattering theory provides the impetus for developing the next set of algorithms, the square root doubling algorithms. They are found to be fast, widely applicable, and numerically well behaved, but their derivation again raises questions that are deferred.

To answer these questions, and to gain insight into the underlying structure of square root algorithms, the concept of the square root of a scattering medium is developed. The ensuing physical insight clarifies many of the unresolved issues, and provides for rapid derivation of several of the square root algorithms.

The chapter ends after proposing possible remedies for several of the algorithm's weaknesses.

Chapter 4 briefly examines the connection between the continuous and the discrete solution to the matrix Riccati equation. Various doubling formulae are examined, with varying degrees of disfavor.

In Chapter 5 the problem of a singular state transition matrix is examined from several perspectives. Gever's [1972] work on order reduction is reviewed, as are Katz's [1974] and Powell's [1978] conjectures. Pure prediction, singular covariance matrices, and repeated closed-loop eigenvalues are also considered.

Chapter 6 presents empirical results comparing the algorithms of Chapters three and four. Memory requirements, elapsed execution time, speed of convergence, and accuracy are the primary criteria for evaluation of the results.

Chapter 7 considers a procedure for reducing the sensitivity of compensation to the effects of uncertainties in the parameters of the model of the system. This procedure is compared to an ad hoc approach. Limitations are delineated.

Chapter 8 uses the algorithm of the previous chapter to design compensation of arbitrary order. The results are compared to frequency domain design, and are applied to a practical problem.

Chapter 9 discusses the problems associated with algorithm implementation. The focus is on the weaknesses and false promises of present hardware, the need for considerable research into algorithm-hardware interaction, and for the development of new, specialized computers.

Chapter II  
MATRIX FUNDAMENTALS  
A Focused Review

Fundamental matrix properties and the problems inherent in actually computing with arrays lies at the heart of this treatise. For that reason a brief discussion of matrices, computation, and algorithms will follow.

The need for specific algorithms to solve specific problems dictated the contents of this chapter. We assume a working knowledge of matrices, review our definitions, and then concentrate on the details and their implications for computation. The objective is to review the quintessential algorithms, examining why they are relevant, and what we can expect from them. Lesser algorithms-- for example, Gram-Schmidt orthogonalization and Gaussian Elimination-- will be mentioned along the way, often in a quite familiar fashion. If these references-- to pivoting, poorly bounded errors, etc.--are not well understood, these secondary references are not crucial to an understanding of the remainder of this treatise.

This is a field rich with fine work and excellent texts. Many must be consulted, however, to garner the full flavor of the field. James Hardy Wilkinson [Wilkinson, 1965] is preeminent in the field, as is Gene Golub [1965]. Also consider Forsythe [1967], Moler [1974], Acton [1970], Lawson [1974], and Strang [1976].

## II.2 Fundamentals

### Computation-- the Theory and the Practice

The primary constraint computers place on purely theoretical algorithms is the unavailability of real real numbers. Instead, there are integers and there are ersatz real numbers, termed floating point numbers. A variety of ersatz schemes exist, but conceptually the real number is represented in scientific notation, with a truncated fractional part and a bounded exponent.

Thus, for example, we might require the fractional part to be greater than one-half, but less than one, and limited to seven digits of precision. The exponent part might be limited to three digits of precision. Thus, we might have

$$\pm f_1 f_2 \dots f_7 \times \beta^e \quad (2.1)$$

where  $\beta$  is the base (2,8,10, or 16 are common),  $f$  is the fractional part, and  $e$  the exponent.

First, we note that we have specified a finite set on the real line of far from equidistant points; small numbers are densely represented; large numbers are comparatively sparsely represented. Second, most real numbers must be represented only approximately, especially if the number lies outside the range of the exponent.

The fundamental theorem in floating-point rounding-error analysis defines the relative error; following Moler [1967]

if we replace a real number  $x$  by the closest floating point approximation,  $x_f$ , then

$$x_f = x(1+\delta)$$

where

$$|\delta| \leq \frac{1}{2}\beta^{1-t}, \quad t: \text{ the number of digits of precision.} \quad (2.2)$$

This result can be extended to bound various floating point operations, such as addition or multiplication, and therefore motivates our later introduction of matrix conditioning.

### Mappings

The square matrix  $A$  represents a linear mapping, or transformation, of each vector  $x$  of one  $n$ -dimensional space  $X$  into the vector  $y = Ax$  of a second  $n$ -dimensional space  $Y$ . Mappings-- especially orthogonal mappings-- are very important in what follows.

### Singular values, Eigenvalues, and Eigenvectors

The fundamental algebraic eigenvalue problem is the determination of those scalars  $\lambda$  for which the equation

$$Ax = \lambda x \quad (2.3)$$

has a non-trivial solution. The general theory of simultaneous linear algebraic equations shows that there is a non-trivial solution if, and only if, the matrix  $(\lambda I - A)$  is singular. This may be restated as requiring that

$$\det (\lambda I - A) = 0 \quad (2.4)$$

This determinant may be expanded into the polynomial

$$\lambda^n + a_1 \lambda^{n-1} + \dots + a_n = 0 \quad (2.5)$$

where  $n$  is the order of  $A$ . This equation always has  $n$  roots, called the eigenvalues of the matrix  $A$ .

Corresponding to any eigenvalue  $\lambda$ , the set of equations (2.3) has at least one non-trivial solution  $x$ ; this solution is called an eigenvector corresponding to that eigenvalue. If the rank of  $(\lambda I - A)$  is less than  $(n-1)$ , then there will be more than one independent vector satisfying (2.3). Also, if  $x$  is a solution to (2.3), then  $cx$  is also a solution, where  $c$  is a scalar.

Not all systems have a complete set of eigenvectors. For example, the matrix  $A_r$ , where

$$A_r = \begin{bmatrix} r & 1 & & \\ & r & 1 & \\ & & \ddots & 1 \\ 0 & & & r \end{bmatrix} \quad (2.6)$$

has the repeated root  $r$  times, but has only one eigenvector

$$x = [1 \ 0 \ \dots \ 0]^T \quad (2.7)$$

We will return to this issue later.

A matrix is positive definite,  $A > 0$ , if all eigenvalues are greater than zero; positive semidefinite or non-negative definite,  $A \geq 0$ , if none of the eigenvalues are negative.

Singular values are the non-negative square roots of the eigenvalues of the symmetric matrix  $AA^T$ , and are denoted  $\mu_i$ . Practically all of the important properties concerning the solution of a system of equations with the matrix  $A$  are determined by the nature of the matrix's singular values (Forsythe [67]); convergence and accuracy are often two major properties.

For non-negative definite, symmetric matrices, the eigenvalues and singular values correspond.

#### Matrix Transformations-- the Definitions

Practical strategies for computing eigenvalues break the problem into two parts: reducing the original matrix to a specialized matrix having many zero elements but the same eigenvalues, followed by the finding of the eigenvalues for the specialized matrix. The square root algorithms presented in the sequel also begin by reducing the system to specialized forms as an initial step.

Reductions of general matrices may be phrased in terms of similarity transformations. The matrix A is said to "suffer" a similarity transformation if it is pre- and post- multiplied by any other matrix and its inverse; the only restriction is that the inverse must exist:

$$B = TAT^{-1}$$

$$C = T^{-1}AT$$

Similarity transformations have the property that they preserve eigenvalues.

An important class of similarity transformations are orthogonal transformations. The transforming matrix T is orthogonal if its transpose is also its inverse

$$T^T = T^{-1}$$

This property is enough to preserve symmetry through the similarity transformation. More importantly, orthogonal transformations are extremely stable (see below).

Orthogonal transformations consist of a possible reflection in some hyperplane, followed by a rigid rotation of n-dimensional space onto itself. The key facts to remember concerning orthogonal transformations are:

- 1) They are simple to generate,
- 2) They are useful for creating blocks of zeroes,
- 3) They are useful for creating triangular forms,
- 4) They preserve eigenvalues and symmetry,
- 5) Their errors are bounded and well-behaved.

They also generally require more computation than their competition.

#### Norms

Norms are measures of length and distortion. A common norm is Euclidean distance, denoted by  $\|x\|$ , which we define as

$$\|x\| \triangleq \sqrt{|x_1|^2 + |x_2|^2 + \dots + |x_n|^2} = \sqrt{x^T x} \quad (2.8)$$

This norm obeys our common sense feeling for determining distance, but it is often difficult to compute; computers are notoriously slow at performing square roots.

The Euclidean norm obeys the following properties in two and three dimensions:

$$\|cx\| = |c| \cdot \|x\| \quad \forall \text{ real } c, \text{ and all vectors } x \quad (2.9)$$

$$\|0\| = 0 \quad \text{and} \quad \|x\| > 0 \quad \text{iff } x \neq 0, \text{ where } 0 \text{ is the null vector} \quad (2.10)$$

$$\|x+y\| \leq \|x\| + \|y\| \quad \text{for all vectors } x \text{ and } y \quad (2.11)$$

If we take these properties as our definition of a norm, we can generate several measures which are more readily computable. For example



$$\|x\|_1 = \sum_{i=1}^n |x_i| \quad , \quad \text{the } L_1 \text{ norm} \quad (2.12)$$

$$\|x\|_\infty = \max_{1 \leq i \leq n} |x_i| \quad , \quad \text{the } L_\infty \text{ norm} \quad (2.13)$$

Both of these norms satisfy our formal requirements for a measure, although they don't allow us to use our geometric (Euclidean) intuition. More importantly, they are both readily computable, and in the appropriate context each can be used to answer the question, "which is smaller."

We now define the norm of a square matrix  $A$  in the obvious fashion-- as a direct extension of the vector definition:

$$\|A\| = \max_{x \neq 0} \frac{\|Ax\|}{\|x\|} = \max_{\|x\|=1} \|Ax\| \quad (2.14)$$

As a direct consequence

$$\|cA\| = |c| \cdot \|A\| \quad \forall \text{ real } c \text{ and all } A \quad (2.15)$$

$$\|\Theta\| = 0 \quad \text{and} \quad \|A\| > 0 \quad \text{if } A \neq \Theta, \text{ where } \Theta \text{ is the null matrix} \quad (2.16)$$

$$\|A+B\| \leq \|A\| + \|B\| \quad \forall \text{ matrices } A \text{ and } B \quad (2.17)$$

Further, it follows immediately that

$$\|Ax\| \leq \|A\| \cdot \|x\| \quad \forall A, x \quad (2.18)$$

$$\|AB\| \leq \|A\| \cdot \|B\| \quad \forall A, B \quad (2.19)$$

It is these last two properties that make norms so useful in analyzing linear mappings and linear systems of equations.

As with the case of vectors, we can propose more readily computable norms, such as

$$\|A\|_1 = \max_{1 \leq j \leq n} \sum_{i=1}^n |a_{ij}| \quad (2.20)$$

$$\|A\|_\infty = \max_{1 \leq i \leq n} \sum_{j=1}^n |a_{ij}| \quad (2.21)$$

### Distortion

Combining many of the notions we have defined above, we can consider the norm of a matrix  $A$ , for example

$$\|A\| = \max_{\|x\|=1} \|Ax\| \quad (2.22)$$

to be a measure of the distortion under the transformation  $x \rightarrow Ax$ .

For  $\|A\|$  is the length of the longest vector in the image set  $\{Ax\}$  of the unit sphere  $\{x: \|x\|=1\}$  under the linear transformation.

Quoting results to be found in any treatment of singular values, if we have a non-singular square matrix  $A$  with singular values arranged such that

$$\mu_1 \geq \mu_2 \geq \dots \geq \mu_n > 0$$

then there is one vector,  $x_1$ , in  $X$  such that  $A$  stretches  $x_1$  by  $\mu_1$ , as  $x_1$  is mapped into  $Ax_1$  of  $Y$ . There is a second vector,  $x_n$ , that stretches by  $\mu_n$ .  $x_1$  and  $x_n$  are orthogonal (assuming  $\mu_1 \neq \mu_n$ ), as are  $Ax_1$  and  $Ax_n$  in  $Y$ . A unit circle in the plane of  $x_1$  and  $x_n$  is mapped into an ellipse with semiaxes  $\mu_1$  and  $\mu_n$ . This is the greatest distortion which can occur to any circle in  $X$ .

Further, the singular values give the Euclidean norm directly, for

$$\|A\| = \mu_1 \quad (2.23)$$

$$\|A^{-1}\| = \mu_n^{-1}$$

Orthogonal matrices are useful because

$$\|Tx\| = \|x\| \quad (2.24)$$

for all orthogonal matrices  $T$ ; orthogonal matrices preserve length and do not distort.

### Conditioning of Matrices

An especially important problem to consider when programming an algorithm is the impact of numeric truncation on the behavior of the computation. Two questions need to be asked:

- 1) Will the algorithm remain stable?
- 2) Will the final answer be correct?

Again we consider the square,  $n \times n$ , nonsingular matrix  $A$ , this time in the context of solving the linear equation  $Ax=b$  for the vector  $x$ . We begin by assuming  $A$  is known exactly, but we assume  $b$  is perturbed from the correct value by an amount  $\Delta b$ . How large, then, can  $\Delta x$  be? We have

$$\|A\| \cdot \|x\| \geq \|b\| \quad (2.25)$$

$$\Delta x = A^{-1} \Delta b \quad (2.26)$$

$$\|\Delta x\| \leq \|A^{-1}\| \cdot \|\Delta b\| \quad (2.27)$$

We begin by noting that for the proper choice of  $\Delta b$ , the equality holds in (2.27). Then, reversing (2.25), multiplying by (2.27), and assuming  $b$  is nonzero, we have:

$$\|\Delta x\| \cdot \|b\| \leq \|A\| \cdot \|A^{-1}\| \cdot \|x\| \cdot \|\Delta b\| \quad (2.28)$$

$$\left\| \frac{\Delta x}{x} \right\| \leq \|A\| \cdot \|A^{-1}\| \cdot \left\| \frac{\Delta b}{b} \right\| \quad (2.29)$$

We next define the spectral condition number,  $\kappa(A)$ , to be the quantity  $\|A\| \cdot \|A^{-1}\|$ . For the Euclidean norm, we have

$$\kappa(A) = \mu_1 / \mu_n \geq 1 \quad (2.30)$$

Therefore, the condition number is a measure of the maximum distortion the transformation  $A$  makes on the unit sphere. Expression (2.29) is an expression of the relative uncertainty in the solution vector  $x$  due to relative uncertainty in the data vector  $b$ . It can never be less than 1.

Similarly, if there is uncertainty in  $A$ , we have

$$\left\| \frac{\Delta x}{x} \right\| \leq \kappa(A) \cdot \left\| \frac{\Delta A}{A} \right\| \quad (2.31)$$

#### Caveats:

The bound is precise in that for proper choice of  $b$  and  $\Delta b$

$$\left\| \frac{\Delta x}{x} \right\| = \kappa(A) \left\| \frac{\Delta b}{b} \right\| \quad (2.32)$$

Since  $\kappa(A)$  is never less than unity, uncertainty can never decrease.

$\kappa(A)$  is a good measure of the quality of the mapping. Conditioning is far more important to determining performance than the determinant or the order of the system [Moler, 1967]. Again, we see that orthogonal mappings are likely to be well behaved, because their condition number is unity.

It is important to note that even small systems can be extremely ill conditioned. For example, the condition number of the matrix

$$A_2 = \begin{bmatrix} 99 & 100 \\ 98 & 99 \end{bmatrix} \quad (2.33)$$

is nearly 40,000; this matrix will introduce serious distortion.

#### Orthogonal Reductions

A popular reduction, and one that we shall consider often in the sequel, takes a general matrix into a triangular form, with all zeroes either above or below the diagonal, for example

$$\begin{bmatrix} x & \dots & x \\ \vdots & & \vdots \\ x & \dots & x \end{bmatrix} \Rightarrow \begin{bmatrix} x & \dots & x \\ & \ddots & \vdots \\ 0 & & x \end{bmatrix} \approx \begin{array}{|c|} \hline \triangle \\ \hline \end{array}$$

One of the better behaved triangularizations-- the Givens approach-- proceeds by making an orderly series of plane rotations. For example, in the  $k$ th step, we would pre-multiply by rotations in the  $(k,k+1)$ ,  $(k,k+2)$ ,  $\dots$ ,  $(k,n)$  planes. Zeroes in the  $k$ th column are then produced one at a time, with each rotation; zeroes in preceding columns are unaffected.

The Givens transformations are easy to construct. To rotate an angle  $\theta$ , we have

$$T = \begin{bmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{bmatrix}$$

If we embed this rotation in a higher dimension space, we can rotate in the plane  $(k, \ell)$  through pre-multiplying by

$$T = \begin{matrix} & & k & & \ell & & \\ & & \begin{matrix} 1 & & & & 0 \\ & \ddots & & & \\ & & 1 & & \\ & & & \cos \theta & \sin \theta \\ & 0 & & 1 & \\ & & & & \ddots & \\ & & & & & 1 \end{matrix} & & \\ k & & & & & & \\ \ell & & & & \begin{matrix} 0 & & & & \cos \theta \\ & & & & 1 \\ & & & & & \ddots \\ & & & & & & 1 \end{matrix} & & 0 \end{matrix} \quad , \text{ for } k < \ell \leq n$$

$$\theta = \cos^{-1} \left[ \frac{a_{kk}}{(a_{kk}^2 + a_{\ell k}^2)^{1/2}} \right]$$

Compared with triangularization by Gaussian elimination, plane rotations are expensive: Gauss, with complete pivoting, requires  $\frac{1}{3}n^3$  multiplications, compared to  $\frac{4}{3}n^3$  multiplications and  $\frac{1}{2}n^2$  square roots for Givens. However, with Gaussian elimination the perturbation error bound always contains a factor of  $2^{n-1}$ , whereas the Givens reduction never displays such unwarranted growth.

We next consider the prima reductions, by A.S. Householder: his famous Householder reflection. These transformations seem to have the essential advantages: they are orthogonal, they have better error properties than even plane rotations, they are fast-- though they still require

$\frac{2}{3}n^3$  multiplications and  $n$  square roots-- and they are easy to compute. As a consequence, they are often recommended sans reservation.

With the Householder reflection, we map the chosen vector, "a", onto the first coordinate of the orthogonal space, but reflected from its initial orientation in that space. Schematically (see Figure 2.1) we could look upon this reduction as a series of Givens rotations, terminated with a reflection, lumped into a single mapping. A better interpretation can be seen from Figure 2.2, a second order example. We begin by constructing the vector "t" with the same coordinates as "a", except for the first coordinate of "t", which is  $a_1 + \|a\|$ . We then construct the plane,  $P_t$ , perpendicular to "t" and passing through the origin. The mapping we desire will reflect all vectors through this plane. A moment's reflection shows that, by construction,  $Ta$  produces the mirror image of "a" lying along " $e_1$ ", the first coordinate of the space.

Construction of  $T$  is especially simple. We choose  $T$  to be  $I - \frac{tt'}{t'a}$ . Then, we have that

$$\begin{aligned} Ta &= \left(I - \frac{tt'}{t'a}\right)a = a - \frac{tt'a}{t'a} \\ &= a - t \end{aligned}$$

as desired. Furthermore,  $T$  is orthogonal:

$$T' = \left(I - \frac{tt'}{t'a}\right)' = I - \frac{tt'}{t'a}, \text{ since } t'a \text{ is a scalar}$$

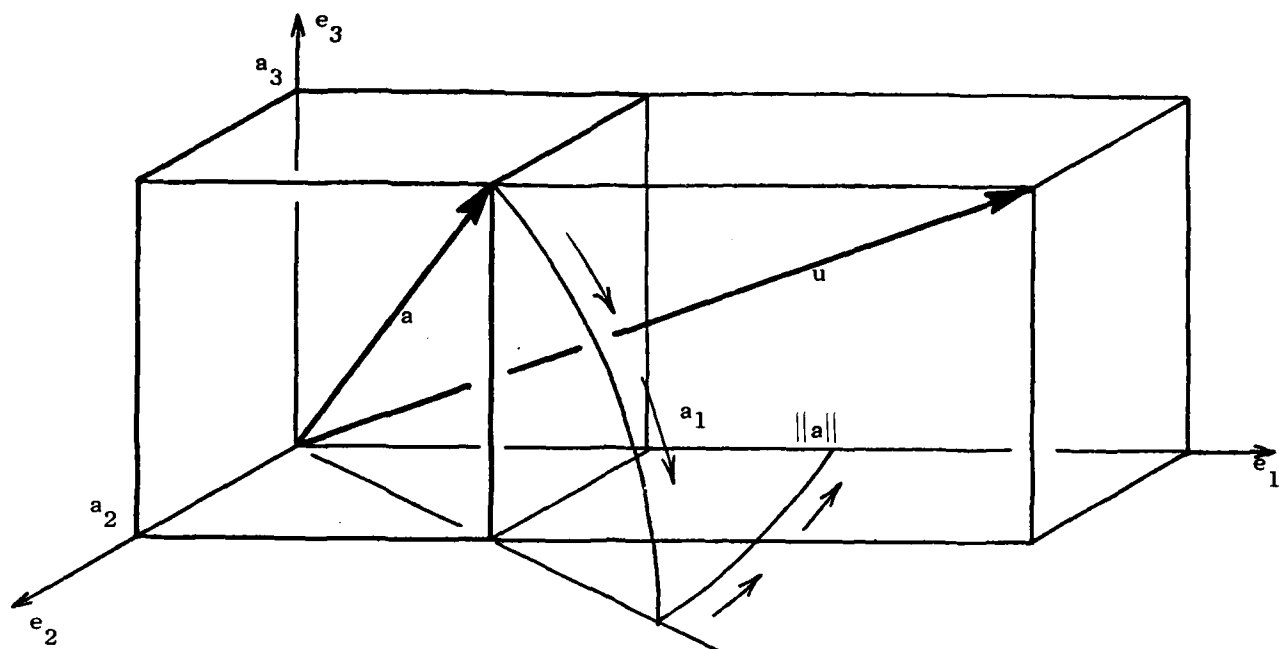


Figure 2.1

A HOUSEHOLDER REFLECTION IN THREE DIMENSIONS

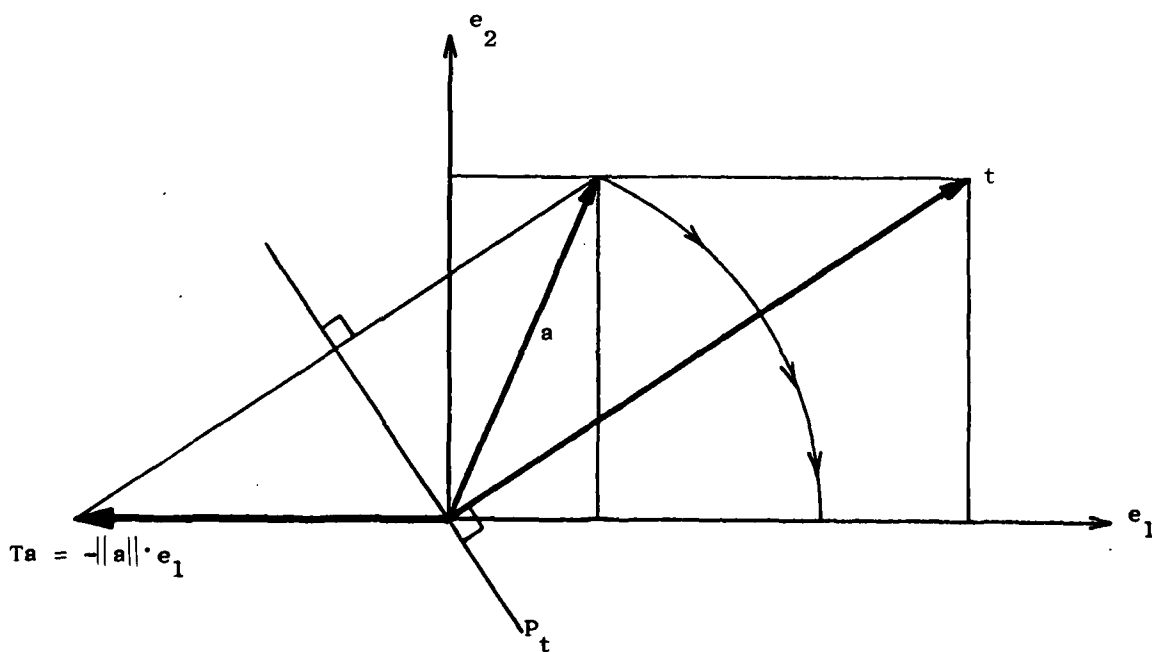


Figure 2.2

A HOUSEHOLDER REFLECTION IN TWO DIMENSIONS



$$\begin{aligned}
TT' &= \left(I - \frac{tt'}{t'a}\right)^2 \\
&= I - 2\frac{tt'}{t'a} + \frac{tt'tt'}{t'at'a} = I - \frac{2}{t'at'a} (t'a - \frac{tt'}{2})tt' \\
&= I - \frac{2}{t'at'a} (t'a - \frac{t't}{2})tt' \\
&= I \quad , \quad \text{since } t't = 2t'a
\end{aligned}$$

As may already be clear, to reduce a column of A, we need not multiply A explicitly by the matrix T. Rather (see Figure 2.3), we calculate  $\|a\|$ ,

$$1) \quad \alpha = \text{sign}(a_1) \|a\|$$

Then we calculate "t" by augmenting "a" by  $\alpha$ ,

$$2) \quad t = a + \alpha e_1$$

which only requires one addition. Next, we need the scale factor  $t'a$ , which is again a single operation

$$3) \quad \beta = t'a = (a + \alpha e_1)^T a = \alpha \cdot t_1$$

Then, for any vector, "b", we must calculate an inner product

$$4) \quad t^T \cdot b$$

normalize

$$5) \quad \gamma = t^T b / \beta$$

and reflect the vector "b"



$$6) \quad T_b = b - \gamma_t$$

Finally, we can replace "a" (which in this explication was temporarily replaced by t)

$$7) \quad a_i = 0, \quad i \neq 1$$

$$a_1 = -\alpha$$

The last point to note is that once we have dealt with blocks of the matrix A, these blocks will remain unchanged by future reflections; further, they need not be considered in calculating the norm of the "pivot" vector. The order of the problem effectively reduces by one after each reflection.

Orthogonal transformations can be applied to non-square matrices. Triangularization, for example, has the effect of compressing a rectangular matrix into a triangular partition of maximal rank and a partition of zeroes.

#### Matrix Square Roots

Any positive-definite, symmetric matrix A has a unique decomposition

$$A = BB^T$$

where B is a lower triangular matrix with positive diagonal elements.

We write the square root of A as  $A^{1/2}$ , where

$$A = A^{1/2} A^{1/2}$$

similarly, for  $A^{-1}$  we have

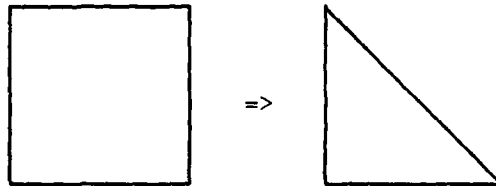
$$A^{-1} = A^{-1/2} A^{-1/2}$$

If  $A$  is symmetric, but not positive definite, then there will be no real non-singular  $B$ . If there is a triangular decomposition, then  $B$  will consist of either pure real or pure imaginary columns. Thus, we may write  $B$  as

$$LS, \quad \text{where } A = L\Lambda L^T, \quad \Lambda = S^2$$

where  $L$  is a pure real lower triangular matrix and  $\Lambda$ , the signature matrix, is a diagonal matrix whose diagonal elements are either 1 or -1.

For the symmetric, positive definite case, the best approach to calculating the square-root is normally Cholesky decomposition, a simple and well behaved algorithm. We define the vector  $\bar{v}_{ij}$  to be the elements of row  $i$  up to, but not including, element  $j$ . Cholesky decomposition (in place) begins by first zeroing all entries above the diagonal.

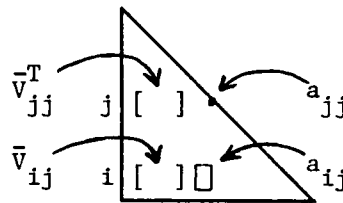


Next, we step down the diagonal, performing each of two operations at each stage. First, the diagonal element,  $a_{jj}$ , is replaced by

$$a_{jj} \leftarrow (a_{jj} - \bar{v}_{jj}^T \bar{v}_{jj})^{1/2}$$

Second, we step down the  $j$ th column, beginning with row  $i=j+1$ . Each element  $a_{i,j}$ , is replaced by

$$a_{ij} \leftarrow \frac{a_{ij} - \bar{v}_{ij}^T \bar{v}_{jj}}{a_{jj}}$$



The Cholesky is extremely stable, never needs pivoting, and is quick-- requiring only  $\frac{n^3}{6}$  multiplications and  $n$  square roots. But yield positive definiteness, and you lose the numerical stability of symmetric decomposition.

Kaminski [1971] claims Cholesky decomposition can be applied to the singular case; the algorithm he presents in Appendix A, however, fails for some singular matrices. In contrast, Wilkinson [1965] states "it cannot be emphasized too strongly that the symmetric decomposition of a matrix which is not positive definite enjoys none of the numerical stability of the positive definite case" [p. 231].

For the singular case, we use Singular Value Decomposition.

#### Singular Value Decomposition (SVD)

Singular value decomposition uses the QR algorithm to decompose a matrix  $A$  into

$$A_{m \times n} = Q_{m \times m} \begin{bmatrix} S_{n \times n} \\ 0 \end{bmatrix} R_{n \times n}^T, \quad m \geq n \text{ (w.o.l.g.)}$$

where  $Q$  and  $R$  are orthogonal and  $S$  is diagonal and non-negative.

Assuming  $A$  is square, symmetric, and non-negative, then  $Q = R$ .

Then the square root of  $A$  is

$$A^{\frac{1}{2}} = Q \cdot S^{\frac{1}{2}}$$

Singular value decomposition is accurate; its deficiency is its complexity. According to Lawson [1974] Cholesky decomposition is more than twenty-five times faster than SVD. Unless  $A$  may be singular, Cholesky decomposition is to be preferred.

#### Finding Eigenvalues and Eigenvectors

The currently best accepted algorithm for finding eigenvalues is generally a two step procedure: reduce the matrix to a standard form, using orthogonal transformations (for example, Householder reflections), and then apply an iterative eigenvalue decomposition routine such as the QR algorithm to generate the eigenvalues.

For a general matrix, we begin by reducing the matrix to Hessenberg form -- all zeroes below the subdiagonal -- while still preserving eigenvalues; this requires  $\frac{5}{3}n^3$  multiplications. Then we begin iterating, using an algorithm that steadily reduces the magnitude of the off-diagonal elements. This requires  $4n^2$  multiplications per iteration. Convergence on each iteration is very fast; for the case of symmetric matrices, it is at least cubic! Total calculations, including all iterations, is normally about  $4n^3$  (Lawson [1974]).

Multiple eigenvalues do not pose any special problems. Their associated eigenvectors, however, introduce a perversity we cannot avoid. For example, with symmetric matrices, multiple eigenvalues correspond to a circular cross section in the corresponding subspace, and therefore any direction is intrinsically an axis. Concomitantly, almost equal eigenvalues correspond to almost circular subspaces. The computational problem remains difficult [Acton, 1970].

## Chapter III

## THE STEADY-STATE SOLUTION TO THE DISCRETE RICCATI EQUATION

As we briefly outlined in Chapter 1, the solution to the linear recursive state estimation problem and to the quadratic control problem are strongly connected to the solution of certain Riccati differential and difference equations. In this chapter we will examine a variety of algorithms for solving the appropriate discrete Riccati equation. Part of this analysis will build upon the connection between estimation theory and scattering theory elucidated by Ljung, Kailath, and Friedlander [1976]. The introduction of a physical paradigm leads to new insights into the strengths, weaknesses, and intrinsic structure of several algorithms.

Before considering the solutions, we more precisely define the problem. Two applications of linear least-squares theory are central to the development of this chapter:

## PROBLEM I-- State Estimation

Let  $x_i$  be an  $n$ -dimensional discrete vector process generated by white noise driving a linear dynamical system:

$$x_{i+1} = \phi_i x_i + \Gamma_i w_i \quad (1.1)$$

and let  $y_i$  be a linear function of  $x_i$ , observed through white noise:

$$y_i = H_i x_i + v_i \quad (1.2)$$

$w_i$  and  $v_i$  are referred to as process noise and measurement noise, respectively. We further assume

$$E x_0 = E w_i = E v_i = 0$$

$$E x_0 x_0^T = 0 \quad E v_i w_j^T = 0$$

$$E w_i w_j^T = Q_i \delta_{ij} \quad E v_i v_j^T = R_i \delta_{ij} \quad , R_i > 0$$

$$E x_0 w_j^T = 0 \quad E \delta_{ij} = \begin{cases} 0 & i \neq j \\ 1 & i = j \end{cases}$$

$$E x_0 v_j^T = 0$$

#### Objective:

To find a linear estimate of  $x_i$  given measurements  $y_j$  up to, but not including, measurement  $i$ . This estimate,  $\hat{x}_{i|i-1}$ , is chosen to minimize the expected error covariance,

$$M_{i|i-1} = E \{ (\hat{x}_{i|i-1} - x_i) (\hat{x}_{i|i-1} - x_i)^T \} , \forall i \quad (1.4)$$

#### Solution:

Assuming  $R_i > 0$ , then the optimal linear estimate,  $\hat{x}_{i|i-1}$ , in the sense of minimizing equation (1.4), is given by

$$\hat{x}_{i+1|i} = (I - K_i H_i) \phi_i \hat{x}_{i|i-1} + K_i y_i$$

where

$$K_i = P_i H_i^T R_i^{-1}$$



and the error covariance in (1.4),  $M_{i|i-1}$ , is given by the following pairs of equations:

Measurement Update

$$\begin{aligned} P_{i|i} &= M_{i|i-1} - M_{i|i-1} H_i^T [R_i - H_i M_{i|i-1} H_i^T]^{-1} H_i M_{i|i-1} \\ &= E\{(\hat{x}_{i|i} - x_i)(\hat{x}_{i|i} - x_i)^T\} = P_i \end{aligned} \quad (1.7)$$

Time Update

$$\begin{aligned} M_{i+1|i} &= \phi_i P_{i|i} \phi_i^T + \Gamma_i Q_i \Gamma_i^T \\ &= E\{(\hat{x}_{i+1|i} - x_i)(\hat{x}_{i+1|i} - x_i)^T\} \end{aligned}$$

Equations (1.5) through (1.7) comprise the Kalman filter for discrete processes.

#### PROBLEM II-- Closed Loop Control

Let  $x_i$  be an  $n$ -dimensional discrete vector process, driven by an external input  $u_i$ , where the governing linear dynamical system is again

$$x_{i+1} = \phi_i x_i + \Gamma_i u_i, \quad \forall i, 0 \leq i \leq N \quad (1.8)$$

$x_0$  given

where  $u_i$  is a deterministic input sequence.

Objective:

To find the input sequence  $u_i$  which minimizes a specified scalar cost function of the form

$$J = \frac{1}{2} \sum_{i=0}^N [x_i^T A_i x_i + u_i^T B_i u_i]$$

where  $A_i$  and  $B_i$  are symmetric matrices;  $A_i$  is non-negative definite and  $B_i$  is positive definite.

Solution:

Assuming  $B_i > 0$ , then the optimal input  $u_i$  in the sense of minimizing equation (1.9) is given by

$$u_i = -C_i x_i \quad (1.10)$$

where

$$C_i = B_i^{-1} \Gamma_i \phi_i^{-1} (S_i - A_i) \quad (1.11)$$

and where  $S_i$  can be found by solving

$$S_i = \phi_i^T (S_{i+1}^{-1} + \Gamma_i B_i^{-1} \Gamma_i^T)^{-1} \phi_i + A_i, \quad S_N = A_N \quad (1.12)$$

Caveats:

Both problems are discrete-time applications, often informally referred to subsequently as discrete systems. Amplitude will always be treated as a continuous dimension in the domain of the model. Actual computations will never involve real-valued quantities, but will be "good" discrete approximations, depending upon the precision available for the calculation.

Most problems will involve the steady-state solution to time invariant problems. In this eventuality, the subscripts will be dropped from the model parameters to simplify the notation.

Another simplification accrues from recognizing the similarity between the two applications. With the introduction of an appropriate mapping between variables, known as duality, a solution to one problem can be claimed as the solution to its dual problem. This duality is contained in the following substitutions:

CONTROL  $\Longleftrightarrow$  ESTIMATION

$\Gamma^T$	H
B	R
A	$\Gamma Q \Gamma^T$
$S_N$	$P_0$
$\phi$	$\phi^T$

In the sequel, most algorithms will be developed for the estimation problem. For some algorithms, however, the derivation is clearer in the control context. Duality, ensures that a solution for one application can always be adapted to the other application using the above mapping; the choice of problem will be based primarily on considerations of physical insight.

The previous chapter reviewed matrix fundamentals and basic algorithms. In this chapter these fundamentals will be used to develop algorithms for solving the two fundamental applications. Section 2 defines criteria for analysis, briefly touching on numerical problems.

Section 3 describes an algorithm, Eigenvector Decomposition (EVD), which exploits modal decomposition along eigenvectors to solve the Hamiltonian (the model of the system and its adjoints). Section 4 elaborates properties of orthogonal transformations, which are then used in Section 5 to develop a class of iterative algorithms; these algorithms are valuable because of their numerical properties. Section 6 introduces another perspective on estimation and control Riccati equations that evolved from scattering theory.

In Section 7 scattering theory is then used to motivate another class of algorithms, the square-root doubling (SQD) algorithms. In Section 8 square-root algorithms are developed purely within the scattering theory framework. In the last substantive section, Section 9, various weaknesses of these algorithms are reviewed, and proposed revisions are considered.

### III.2 Iteration

The benchmark, by which all methods should be compared, is straight iteration of the Riccati error covariance equations:

$$M_{i+1} = \phi_i P_i \phi_i^T + \Gamma_i Q \Gamma_i^T, \quad \text{where } P_0 = 0 \quad (2.1)$$

$$P_i = M_i - M_i H_i^T (R_i + H_i M_i H_i^T)^{-1} H_i M_i \quad (2.2)$$

and where the filter gain is calculated as a function of this covariance

$$K_i = P_i H_i^T R_i^{-1} \quad (2.3)$$

Several problems can arise while trying to iterate equations (2.1) and (2.2) to steady-state. These equations may converge inordinately slowly, making the calculation expensive. The calculation may also be inaccurate. The error covariances may become indefinite due to truncation errors; this has an unfortunate effect on the feedback gains,  $K_i$ . In a related problem, ill-conditioning can lead to inaccurate convergence, or even divergence.

We will address these latter questions when we discuss the square-root algorithms in Section III.5. The first issue, cost of computation, we will discuss only in relative terms. In Table III.1 we present operations counts for several of the algorithms. The first major row accumulates initialization overhead, the second major row accumulates iteration overhead, and the third row, termination overhead. After a relative accounting of the number of iterations required to terminate, a final column presents cost in cycles assuming computations for a single-input/single output system, and for a system with as many inputs and outputs as states.

For costs we have assumed the following

- additions take two cycles
- multiplies take three cycles
- divisions take five cycles
- square roots take forty cycles.

These ratios should be fairly typical for modern computers-- from small to large. We have assumed square roots will be done in software, using hardware addition, multiplication, and division.

We note immediately that the computation time for square roots has a negligible impact on the cycle cost precisely because the algorithms always require  $O(n)$  square-root computations; for all algorithms considered, at most  $4n$  square-roots need to be computed.

Table III.2 presents the computation costs for the fundamental operations we assumed (matrix multiply, Cholesky decomposition, etc.)

### III.3 Eigenvector Decomposition

We begin by solving the dual to the optimal filter problem--the optimal quadratic regulator problem. Following the approach of Bryson [1975], the minimal cost function is determined from a two-point boundary value problem. From the previous section, given the model

$$x_{i+1} = \phi x_i + \Gamma u_i \quad (3.1)$$

and a quadratic cost function

$$J = \frac{1}{2} \sum_{i=0}^N x_i^T A x_i + u_i^T B u_i \quad (3.2)$$

where  $A$  and  $B$  are symmetric;  $A$  is positive definite and  $B$  is positive semi-definite. We wish to find a linear feedback specification

$$u_i = -C_i x_i \quad (3.3)$$

such that  $J$  is minimized. Then  $N$  is allowed to go to infinity; if a steady-state is reached then  $C_N$  will be a constant,  $C$ .

The derivation begins by noting that  $u_N$  will affect only states outside the range of interest. We can therefore rewrite (3.2) as

$$J = \frac{1}{2} x_N^T A x_N + \frac{1}{2} \sum_{i=0}^N x_i^T A x_i + u_i^T B u_i \quad (3.4)$$

We then minimize  $J$  via the calculus of variations, by augmenting  $J$  with an undetermined multiplier,  $\lambda_i$ . Then

$$\bar{J} = \frac{1}{2} x_N^T A x_N - \lambda_N^T x_N + \sum_{i=1}^{N-1} [H_i - \lambda_i^T x_i] + H_0,$$

where  $H_i$  is given by

$$H_i = \frac{1}{2} x_i^T A x_i + \frac{1}{2} u_i^T B u_i + \lambda_{i+1}^T (\phi x_i + \Gamma u_i) \quad \forall i, 0 \leq i \leq N \quad (3.6)$$

We now consider differential changes in  $\bar{J}$  due to differential changes in  $u_i$ . The appropriate choice for  $\lambda_i$  is then

$$\frac{\partial H_i}{\partial x_i} - \lambda_i^T = 0 \quad \forall i, 0 \leq i \leq N \quad (3.7)$$

$$x_N^T A - \lambda_N^T = 0$$

This gives

$$d\bar{J} = \sum_{i=0}^{N-1} \frac{\partial H_i}{\partial u_i} du(i) + \lambda_0^T dx_0 \quad (3.8)$$

For an extremum,  $d\bar{J}$  must be zero for arbitrary  $du_i$ . Therefore,

$$\frac{\partial H_i}{\partial u_i} = 0 \quad \forall i, 0 \leq i \leq N \quad (3.9)$$

Installing (3.6) in (3.9), we have

$$u_i^T B + \lambda_{i+1}^T \Gamma = 0 \quad (3.10)$$

$$u_i = -B^{-1} \Gamma^T \lambda_{i+1}$$

Using (3.10) in (3.1), and rewriting (3.7), gives

$$x_{i+1} = \phi x_i - \Gamma B^{-1} \Gamma^T \lambda_{i+1} \quad x_0 = x(0) \quad (3.11)$$

$$\lambda_i^T = \lambda_{i+1}^T \phi + x_i^T A \quad \lambda_N^T = x_N^T A$$

the Euler-Lagrange difference equations. Formulated in state space notation, we have

$$\begin{bmatrix} x \\ \lambda \end{bmatrix}_{i+1} = \begin{bmatrix} \phi + \Gamma B^{-1} \Gamma^T \phi^{-T} A & -\Gamma B^{-1} \Gamma^T \phi^{-T} \\ -\phi^{-T} A & \phi^{-T} \end{bmatrix} \begin{bmatrix} x \\ \lambda \end{bmatrix}_i = H \begin{bmatrix} x \\ \lambda \end{bmatrix}_i \quad (3.12)$$

assuming  $\phi$  is invertible. The square matrix, called the Hamiltonian of the system, is symplectic. That is, for each eigenvalue  $v_i$  there exists an eigenvalue  $v_j$  with the same multiplicity, where  $v_j$  equals  $1/v_i$ .

The vector  $\lambda_i$  can also be expressed as a linear combination of the  $x_i$ . Calling this mapping  $P_i$ , (3.12) becomes

$$P_i = \phi^T (P_{i+1}^{-1} + \Gamma B^{-1} \Gamma^T)^{-1} \phi + A \quad (3.13)$$

where

$$\lambda_i = P_i x_i$$



Vaughn [1970] used equation (3.13) and the fact that  $H$  is symplectic to extend Potter's original solution for the equivalent continuous system [1966]. He considered the array of column vectors  $[T_S; T_U]$  which are the eigenvectors of (3.12) associated respectively with the stable and unstable eigenvalues. Then, defining

$$T_U = \begin{bmatrix} \bar{X}_U \\ \Lambda_U \end{bmatrix} \quad (3.14)$$

to be a partition of  $T$  based on the equations for  $x_i$  and  $\lambda_i$ , the steady-state solution for  $P_i$  can be written as:

$$P = \Lambda_U \bar{X}_U^{-1} \quad (3.15)$$

The algorithm for calculating the steady-state controller gains,  $C$ , is therefore straightforward. After computing the matrix (3.12), the matrix of eigenvectors is determined, using, for example, the QR algorithm. Given  $T_U$ , (assuming a complete set of eigenvectors exist), then (3.15) follows directly.

We note in passing that the closed loop system is stable if

- 1) The system is stabilizable (i.e., if the unstable modes are controllable) through the control gain  $\Gamma$ , and
- 2) The system is detectable (i.e., if the unstable modes are observable) in the cost function  $J$ .

Similarly, for the Kalman filter, the closed loop system will be stable if

- 1) The system is detectable (i.e., if the unstable modes are observable), and
- 2) The system is stabilizable (i.e., if the unstable modes are controllable).

### Comments on the Strengths and Weaknesses of Eigenvector Decomposition

The implementation of this algorithm is predicated upon a numerically well-conditioned eigenvalue decomposition procedure. As outlined in Chapter II, by choosing the QR algorithm we would expect excellent performance. This has been our experience (see Chapter VI).

On the negative side, several assumptions made during the derivation of this method preclude the solution of interesting and important problems. For example, the state transition matrix,  $\phi$ , was assumed invertible. In systems incorporating pure delay, invertibility may not obtain. In addition, the control weighting matrix,  $B$  (or the measurement noise covariance matrix,  $R$ ), was assumed non-singular; some observations, however, may be uncontaminated by noise leading to a singular weighting matrix (see Chapter V).

At another level, the existence of a complete set of eigenvectors was assumed. This may not be true if the system has repeated closed-loop eigenvalues.

In each of these cases the Riccati equation may still have a unique, positive, steady-state solution. These issues will be reconsidered in Section III.9 (which proposes algorithm revisions) and again in Chapter Five, under differentiating applications.

### III.4 Square Roots and Orthogonal Transformations

Several facts about orthogonal transformations will clarify the results of the subsequent sections. These deal with the implicit nature

of the orthogonal mapping and with an indirect mechanism for calculating inverse square roots via triangularization.

The previous chapter extolled the many virtues of these mappings. They provide an additional benefit when working with the square roots of matrices. If two arrays are equal, for example, if

$$A = B$$

$$A^{\frac{1}{2}} A^{T/2} = B^{\frac{1}{2}} B^{T/2}$$

then any orthogonal transform can be inserted between  $A^{\frac{1}{2}}$  and  $A^{T/2}$ :

$$A^{\frac{1}{2}} \sigma^T A^{T/2} = B^{\frac{1}{2}} B^{T/2}, \quad \sigma \sigma^T = I$$

The matrix square root,  $A^{\frac{1}{2}}$  or  $B^{T/2}$ , is only unique to within an orthogonal transformation.

If  $A^{\frac{1}{2}}$  and  $B^{\frac{1}{2}}$  have different structures, then there is an orthogonal transformation such that

$$A^{\frac{1}{2}} \sigma = B^{\frac{1}{2}}$$

Specifically, if  $B^{\frac{1}{2}}$  is lower triangular, then any orthogonal triangularization algorithm can be used to map  $A^{\frac{1}{2}}$ . We will always use Householder reflections, for the reasons outlined earlier.

To calculate  $[I + AA^T]^{-1}$  or  $[I + A^T A]^{-1}$  we consider applying an orthogonal triangularization to the array

$$[I; A^T] \cdot \sigma = [B; 0], \quad B = \begin{bmatrix} \times & \times \\ 0 & \times \end{bmatrix} \quad (4.1)$$

where

$$\sigma \cdot \sigma^T = I$$

Then, if

$$\sigma = \begin{bmatrix} T_1 & T_{12} \\ T_{21} & T_2 \end{bmatrix} \quad (4.2)$$

we will demonstrate that

$$B = T_1^{-T} \quad (4.3)$$

$$A = T_{21} \cdot T_1^{-1} \quad (4.4)$$

$$[I + A^T A]^{-1} = T_1 \cdot T_1^T \quad (4.5)$$

$$[I + A A^T]^{-1} = T_2 \cdot T_2^T \quad (4.6)$$

To see this, multiply (4.1) on the right by  $\sigma^T$ . Then we have

$$[I; A^T] = [B; 0] \cdot \begin{bmatrix} T_1^T & T_{21}^T \\ T_{12}^T & T_2^T \end{bmatrix} \quad (4.7)$$

$$I = B \cdot T_1^T \quad (4.8)$$

$$A^T = B \cdot T_{21}^T \quad (4.9)$$

Squaring both sides of (4.1) gives

$$[I + A^T A] = B \cdot B^T = (T_1 \cdot T_1^T)^{-1}$$

Similarly, using (4.9)

$$\begin{aligned} [I + AA^T] &= [I + T_{21} T_1^{-1} T_1^{-T} T_{21}^T] \\ &= T_2^{-T} [T_2^T T_2 + T_2^T T_{21} T_1^{-1} T_1^{-T} T_{21}^T T_2] T_2^{-1} \end{aligned}$$

Examining

$$\Theta^T \cdot \Theta = \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix} = \begin{bmatrix} \text{---} & \text{---} \\ T_{12}^T T_1 + T_2^T T_{21} & T_{12}^T T_{12} + T_2^T T_2 \end{bmatrix} = \begin{bmatrix} - & - \\ C & D \end{bmatrix}$$

We have, from element C of  $\Theta^T \cdot \Theta$ ,

$$\begin{aligned} I + AA^T &= T_2^{-T} [T_2^T T_2 + T_{12}^T T_1 T_1^{-1} T_1^{-T} T_{12}^T] T_2^{-1} \\ &= T_2^{-T} [T_2^T T_2 + T_{12}^T T_{12}] T_2^{-1} \\ &= (T_2^T T_2)^{-1}, \text{ from element D of } \Theta^T \cdot \Theta \end{aligned}$$

(For an alternate proof of these results, see Bierman [1976].)

We can generalize these results further. For the case of  $[R + AA^T]^{-1}$  and  $[R + A^T A]^{-1}$ , we have

$$[R^{\frac{1}{2}}; A] \Theta = [B; 0] \quad B = \triangle$$

(note: we have constructed the left array with A, not  $A^T$ )

$$\begin{aligned} B &= R^{\frac{1}{2}} T_1^{-T} \\ A &= R^{\frac{1}{2}} T_1^{-T} T_{21}^T & T_{21} &= A^T R^{-T/2} T_1 \\ [R + AA^T]^{-1} &= R^{-T/2} T_1^T T_1^{-1} R^{-1/2} \\ [R + A^T A]^{-1} &= R^{-1/2} T_2^T T_2^{-1} R^{-T/2} \end{aligned}$$

We typically need to work with the elements of  $\sigma$ , although we do not want to calculate the elements of the matrix explicitly. This is easily done. For example, to calculate  $T_2 = [I + AA^T]^{-T/2}$ , construct

$$\begin{bmatrix} I & A^T \\ 0 & I \end{bmatrix} \cdot \sigma = \begin{bmatrix} T_1^{-T} & 0 \\ T_{21} & T_2 \end{bmatrix}$$

The calculation of the inverse square root is therefore straightforward, and implicit.

We will also need to concatenate orthogonal transformations. For this, if

$$AA^T = I \quad \text{and} \quad BB^T = I$$

then

$$(AB) \cdot (AB)^T = I$$

and defining  $C$  and  $D$  such that

$$C = \begin{bmatrix} A & 0 \\ 0 & I \end{bmatrix} \quad \text{and} \quad D = \begin{bmatrix} I & 0 \\ 0 & B \end{bmatrix}$$

then

$$CC^T = DD^T = (CD) \cdot (CD)^T = I$$

### III.5 Square Root Algorithms

The earliest class of square-root algorithms recursively computed the square root of the Riccati difference equation. Many have dealt with this subject, including recently Golub [1965], Schmidt [1970], Kaminski and Bryson [1971,1972], Morf and Kailath [1975], and Bierman [1977].

The properties of orthogonal transformations presented in the last section can be used to derive these algorithms in a purely algebraic fashion. In so doing, interesting questions arise, and the stage will be set for the scattering derivation to follow.

Beginning again with the Riccati equation for the estimation problem

$$M_{i+1} = \phi_i P_i \phi_i^T + \Gamma_i Q_i \Gamma_i^T \quad (5.1)$$

$$P_i = M_i - M_i H_i^T (R_i + H_i M_i H_i^T)^{-1} H_i M_i \quad (5.2)$$

(5.2) can be rewritten to eliminate the subtraction

$$P_i + M_i H_i^T (R_i + H_i M_i H_i^T)^{-1} H_i M_i = M_i \quad (5.3)$$

The quantities of interest are still  $M_{i+1}$  and  $P_i$ . We also note that (5.1), (5.2), and (5.3) are all symmetric equations, in the sense that each term can be written as  $AA^T$ .

It is clear how to separate (5.1):

$$[\phi_i P_i^{1/2} \Gamma_i Q_i^{1/2}] \begin{bmatrix} 1 \\ 0 \end{bmatrix} = [M_{i+1}^{1/2} \begin{bmatrix} 1 \\ 0 \end{bmatrix}] , \quad M_{i+1}^{1/2} , P_i^{1/2} = \Delta \quad (5.4)$$

Taking transposes, and squaring both sides produces equation (5.1).

To calculate (5.3) as a square root, we need

$$\begin{aligned} &M^{\frac{1}{2}} \text{ on the left} \\ &P^{\frac{1}{2}} \text{ and } MH^T(R+HMH^T)^{-T/2} \text{ on the right.} \end{aligned}$$

We begin by computing the inverse,  $(R+HMH^T)^{-1}$ . From the previous section, we have

$$\begin{bmatrix} R^{\frac{1}{2}} & HM^{\frac{1}{2}} \end{bmatrix} \sigma_2 = \begin{bmatrix} B & 0 \end{bmatrix} \quad B = R^{\frac{1}{2}} T_1^{-T} = \triangle \quad (5.5)$$

where  $\sigma_2$  implicitly computes the inverse of  $R + H M H^T$ . Examining the elements of  $\sigma_2$ :

$$\begin{aligned} T_1 &= R^{T/2} [R + H M H^T]^{-T/2} \\ T_{21} &= M^{\frac{1}{2}} H (R+HMH^T)^{-T/2} \\ T_2 &= R^{\frac{1}{2}} [R + H^T M H]^{-T/2} \end{aligned} \quad (5.6)$$

We note that  $T_{21}$  is precisely the quantity we need, for

$$\begin{bmatrix} R^{\frac{1}{2}} & HM^{\frac{1}{2}} \\ 0 & M^{\frac{1}{2}} \end{bmatrix} \sigma_2 = \begin{bmatrix} B & 0 \\ MH^T (R+HMH^T)^{-T/2} & C \end{bmatrix} \quad (5.7)$$

and  $C$  must be  $P^{\frac{1}{2}}$ , as desired.

We now have two update equations, which we would like to combine into one (with a single orthogonal mapping).

In the next section we will show how to concatenate mappings. In this section we will, instead, quickly rederive the formulas; but this time for a single update. Beginning by combining equations (5.1) and (5.2):



$$M_{i+1} = \phi M_i \phi^T + \Gamma Q \Gamma^T - \phi M_i H^T (R + H M_i H^T)^{-1} H M_i \phi^T \quad (5.8)$$

remove the minus sign

$$M_{i+1} + \phi M_i H^T (R + H M_i H^T)^{-1} = \phi M_i \phi^T + \Gamma Q \Gamma^T \quad (5.9)$$

construct the inverse transformation

$$[R^{\frac{1}{2}}; H M_i^{\frac{1}{2}}] \sigma = [B; 0] \quad (5.10)$$

and note the result of this mapping

$$T_{21} = M_i^{\frac{1}{2}} H^T (R + H M_i H^T)^{-T/2}$$

$$\begin{bmatrix} R^{\frac{1}{2}} & H M_i^{\frac{1}{2}} \\ 0 & \phi M_i^{\frac{1}{2}} \end{bmatrix} \sigma = \begin{bmatrix} B & 0 \\ \phi M_i H^T (R + H M_i H^T)^{-T/2} & C \end{bmatrix} \quad (5.11)$$

The two sides are nearly complete. Adding the final term, we have

$$\begin{bmatrix} R^{\frac{1}{2}} & H M_i^{\frac{1}{2}} & 0 \\ 0 & \phi M_i^{\frac{1}{2}} & \Gamma Q^{\frac{1}{2}} \end{bmatrix} \sigma = \begin{bmatrix} B & 0 & 0 \\ \phi M_i H^T (R + H M_i H^T)^{-T/2} & C & 0 \end{bmatrix} \quad (5.12)$$

where again, C must equal  $M_{i+1}^{\frac{1}{2}}$  (which follows immediately from squaring both sides).

It may seem surprising that the three step procedure outlined initially coalesces so neatly into a uniform, single step procedure. This interrelationship of the individual component equations is a product

of the intrinsic structure of the problem. We shall examine this structure, in terms of scattering layers, shortly.

#### Comments on the Strengths and Weaknesses of Standard Square Root Algorithms

As a positive note, the square root algorithms provide improved accuracy in the final result. This improvement arises for two reasons. First, since the iterated variable is the square root of the error covariance, this variable, when squared, must be positive semidefinite; the calculated error covariance can never evolve negative eigenvalues. In contrast, no such assurances can be made for normal iteration.

The conditioning can be examined analytically by introducing the spectral condition number,  $\kappa$ , derived in the previous chapter. Given the condition number for a matrix  $P$ , it follows directly that the condition number for the square root of  $P$  is the square root of the condition number for  $P$ :

$$P = LL^T \quad (5.13)$$

$$\kappa(L) = \kappa(P)^{\frac{1}{2}}, \text{ since the singular values of } L \text{ are the square root of the singular values of } P.$$

Recall that spectral conditioning was defined from the expression

$$\frac{\|\Delta x\|}{\|x\|} \leq \kappa \frac{\|\Delta b\|}{\|b\|} \quad (5.14)$$

If the computer calculates in floating point with  $2n$  bits of accuracy, then

$$\frac{\|\Delta b\|}{\|b\|} \leq \frac{1}{2} 2^{1-2n} \quad (5.15)$$

Thus, for square root algorithms, approximately half the word length ( $n$  bits of accuracy), should be sufficient to provide the same error,  $\frac{\|\Delta x\|}{\|x\|}$ .

As Kaminski [1971] points out, this analysis applies to the linear case, but the Riccati equation is non-linear. For example, in the case of two equations updated separately (e.g., (5.4) and (5.7)), Kaminski found that the appropriate error equations were

$$\frac{\|\Delta x\|}{\|x\|} \leq \kappa(L) \left[ \frac{\|\Delta L\|}{\|L\|} + \frac{\|\Delta y\|}{\|y\|} \right] + \kappa^2(L) \cdot \frac{\|r\|}{\|y-r\|} \cdot \frac{\|\Delta L\|}{\|L\|}.$$

As long as the residual is small compared to the update, the term associated with the squared spectral condition number can be ignored; this is usually the case when the number of measurements are much fewer than the order of the system. However, even if this is not the case and the  $\kappa^2$  term dominates, the square root filter is never more ill-conditioned than the normal iterative method (Kaminski [1971]).

For a final note, we turn to Table 3.1 to point out that the square-root implementation typically requires more computation. Variation within the classes of algorithms is large, depending upon the particular problem being solved and the strategy adopted (for example, choosing Givens, Householder, or square root free implementations; see, for example, Bierman [1977]). The square root algorithms, however, often requires up to 40% more computation than the normal iterative algorithms (see Table 3.1).

### Faster Algorithms

If the model parameters are time invariant, Morf et al. [1975] have shown that a considerable computational savings can be realized by working with the square roots of  $\Delta P_i$ , instead of  $P_i$ . Assuming known initial conditions, or if we assume they are zero (if the steady-state error covariance is the objective) then,

$$\Delta P_i = P_{i+1} - P_i$$

is non-negative definite, and of non-increasing rank. Therefore, as explained earlier, we can factor  $\Delta P_i$  as

$$\Delta P_i = L_i L_i^T$$

where  $L_i$  is of size  $n \times \alpha$ , where  $\alpha$  is the rank of  $\Delta P_i$ . Since

$$\Delta P_0 = P_1 - \pi_0 = \Gamma Q \Gamma^T$$

$\alpha$  must be at most the number of inputs to the system.

Using the fact that

$$P_{i+1} = P_i + L_i L_i^T$$

we can derive the square root forms as

$$\begin{matrix} p & \alpha \\ P & \left[ \begin{array}{cc} (R_{i-1}^E)^{1/2} & H L_{i-1} \\ \tilde{K}_{i-1} & \phi L_{i-1} \end{array} \right] \end{matrix} \sigma = \begin{matrix} p & \alpha \\ \left[ \begin{array}{cc} (R_i^E)^{1/2} & 0 \\ \tilde{K}_i & L_i \end{array} \right] \end{matrix}$$

where

$$R_1^E = R$$

$$\tilde{K}_0 = 0$$

$$L_0^{\frac{1}{2}} = \Gamma Q^{\frac{1}{2}}$$

#### Comments on Strengths and Weaknesses

The fast algorithms are expected to have error properties similar to those outlined for the original square root algorithms; this conjecture has yet to be demonstrated. They are, however, several times faster than other straight iteration algorithms, as can be seen from Table 3.1.

#### III.6 Scattering Theory

Scattering theory describes the reflection and transmission characteristics of a layered medium as particles, waves, etc., pass through that medium. Linear least squares estimation and control describes the propagation of states and error covariances, etc., as they pass through time. Both give rise to Riccati and related equations. As with the estimation-control duality introduced in Section III.1, a correspondence can be made between Riccati equations arising from different contexts. Identifying similar terms found in different equations provides a mathematical isomorphism that can then lead to new insights.

In this dissertation we will be working with a particular theory of scattering originally developed by Redheffer in 1950 ([1950], ... , [1962]). Redheffer set out the solution to the scattering problem in

equations directly related to the Riccati-type formulae. When Ljung, Kailath, and Friedlander [Ljung, 1976], [Friedlander, 1976] explored the connection with linear estimation, several key concepts were introduced. Redheffer's work could be interpreted as associating time with layers in the estimation context, and it therefore included time-varying parameter models.

Another approach is taken in the study of geophysics, which considers the scattering of waves in layered media. This scattering corresponds to constant parameter, time invariant models, and leads to models where layers correspond to different system orders or state components. Early published work was done by Wiggins and Robinson [1965]. See also Claerbout [1976].

The scattering of waves provides a conceptual structure for visualizing updates of estimates, states, adjoint variables, or covariances via the Riccati equation. Updates in the scattering context are effectively generalized to include the combination of any two adjacent layers into an equivalent single layer. In the estimation context, these layers can correspond to two arbitrary, adjacent blocks of time. Alternatively, they can correspond to different components of a state-vector, or to operations arising from [feedback] loop removal.

The operator which describes this union, the star-product, leads to a powerful set of matrix manipulations. These manipulations are often more convenient to use because of their immediate connection with the estimation or control problem being considered.

## BASIC OPERATIONS

We define a canonical scattering layer as in Figure 6.1. A signal from the left will be transmitted (with amplitude  $T_L$ ) and reflected (with amplitude  $R_L$ ). Similarly, for the signal entering from the right. The signal exiting from the left is composed of a weighted sum of both entering signals. Schematically, we represent this layer as a matrix, with the transmission coefficients on the diagonal:

$$\begin{bmatrix} T_L & R_R \\ R_L & T_R \end{bmatrix}$$

Looking at Figure 6.2, it is clear how to concatenate two layers; the concatenation would be straightforward, except for the loop,  $A$ . Using a simple application of Mason's Rule for networks, the result of a star-product can be easily "read out" by following the arrows along each path. This procedure gives

$$\begin{bmatrix} a_1 & \rho_1 \\ r_1 & \alpha_1 \end{bmatrix} * \begin{bmatrix} a_2 & \rho_2 \\ r_2 & \alpha_2 \end{bmatrix} = \begin{bmatrix} a_2(I-\rho_1 r_2)^{-1} a_1 & \rho_2 + a_2 \rho_1 (I-r_2 \rho_1)^{-1} \alpha_2 \\ r_1 + \alpha_1 r_2 (I-\rho_1 r_2)^{-1} a_1 & \alpha_1 (I-r_2 \rho_1)^{-1} \alpha_2 \end{bmatrix} \quad (6.1)$$

as the definition of the star-product. Note that the terms  $(I-\rho_1 r_2)^{-1}$  and  $(I-r_2 \rho_1)^{-1}$  arise because of the loop  $A$ .

In Figure 6.3, we examine the relationship between the scattering layer and the state transition matrix. Schematically, to convert from one to the other, we need to reverse the flow along one of the transmission

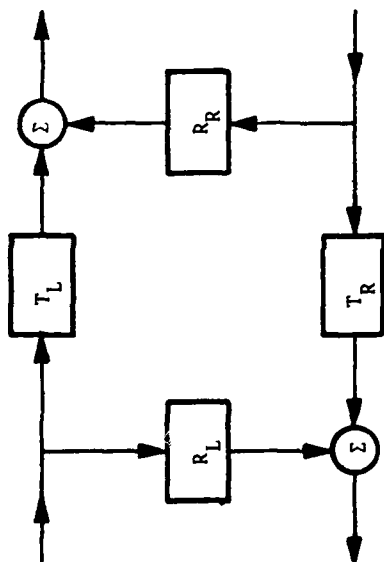


Figure 6.1 Canonical Scattering Layer

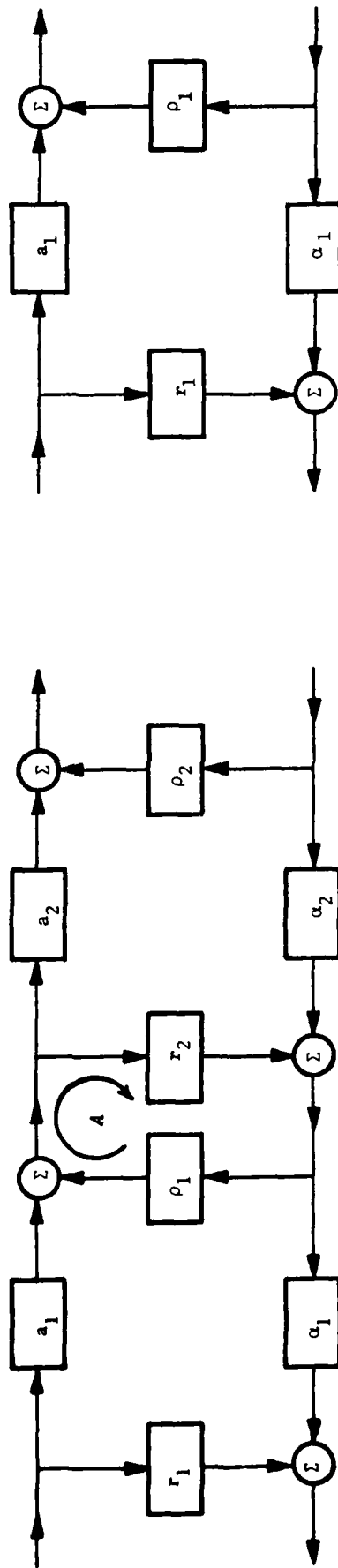


Figure 6.2 Concatenation of Layers

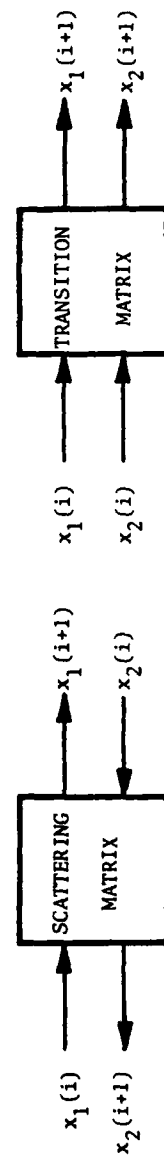


Figure 6.3 Domain Translation



paths. As with the definition of star-product, the relationship is algebraic:

$$\begin{bmatrix} x_{i+1}^{(1)} \\ x_{i+1}^{(2)} \end{bmatrix} = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} x_i^{(1)} \\ x_i^{(2)} \end{bmatrix}, \text{ where } A \text{ and } D \text{ are transmission coefficients, and } B \text{ and } C \text{ are reflection coefficients} \quad (6.2)$$

Then, assuming the determinant of  $D$  does not equal zero, we have

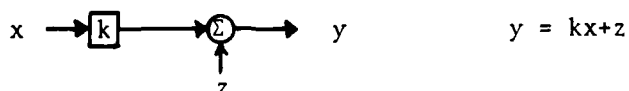
$$\begin{bmatrix} x_{i+1}^{(1)} \\ x_i^{(2)} \end{bmatrix} = \begin{bmatrix} A-BD^{-1}C & BD^{-1} \\ -D^{-1}C & D^{-1} \end{bmatrix} \begin{bmatrix} x_i^{(1)} \\ x_{i+1}^{(2)} \end{bmatrix} \quad (6.3)$$

where  $D$  is denoted as the pivot element. We could also pivot about  $A$ , assuming  $A$  is non-singular, so that

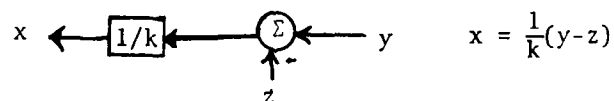
$$\begin{bmatrix} x_i^{(1)} \\ x_{i+1}^{(2)} \end{bmatrix} = \begin{bmatrix} A^{-1} & -A^{-1}B \\ CA^{-1} & D-CA^{-1}B \end{bmatrix} \begin{bmatrix} x_{i+1}^{(1)} \\ x_i^{(2)} \end{bmatrix} \quad (6.4)$$

This mapping, called an exchange step, converts a transition matrix to a scattering layer. The same mapping converts a scattering layer back to the appropriate transition matrix.

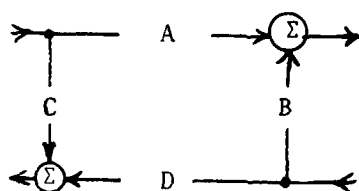
A simpler interpretation obtains by pictorially examining the exchange step (see Verghese [1978]). Beginning with a network of the form



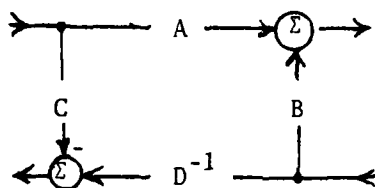
the transmission flow can be reversed by inverting the gain, and by changing the sign of flows into the path (see, for example, Mason and Zimmerman [1960]),



Applying this procedure to the scattering layer converts



into the transition matrix



from which we can "read out" the entries directly as

$$\begin{bmatrix} A-BD^{-1}C & BD^{-1} \\ -D^{-1}C & D^{-1} \end{bmatrix}$$

# PROPERTIES OF SCATTERING LAYERS

It will be useful in the sequel to choose a notation that differentiates the scattering layers and transition matrices more explicitly. We define  $A^*$  to be a scattering layer,  $A$  (unadorned) to be a transition matrix, and  $A^E$  to be the mapped equivalent of  $A^*$ , i.e.,  $A^{*E} = A$ . This matrix identity,  $I$ , remains the same in both domains:

$$P1) \quad I = I^*$$

We will also have need for a modified identity, the  $J$  matrix, which we define as

$$P2) \quad J = \begin{bmatrix} -I & 0 \\ 0 & I \end{bmatrix} = J^* \quad J * A = J \cdot A$$

$$J^{-1} = J$$

$J$  can be considered as a particular choice of signature matrix (see Chapter II).

From Figure 6.3, we can see an additional implication of the exchange step mapping, for

$$P3) \quad A \cdot B = C \quad \text{then} \quad A^* * B^* = C^*, \text{ assuming}$$

we can pivot each of the arrays.

Similarly

$$P4) \quad \text{If } A^* * B^* = C^* \quad \text{then} \quad A \cdot B = C$$

Inverse transition matrices have a particularly useful property (see Friedlander [August, 1976]).

$$P5) \text{ If } A \cdot A^{-1} = I \quad \text{then } A * A^{-1} = I$$

From these properties, we can also derive many of the important orthogonal properties

$$P6) \text{ If } \sigma \cdot \sigma^T = I \text{ then } \sigma^* (\sigma^T)^* = I \text{ (from P3)}$$

$$P7) \text{ If } \sigma \cdot \sigma^T = I \text{ then } \sigma^* \sigma^T = I \text{ (from P5)}$$

A slightly more elaborate definition of orthogonality is required in the scattering domain (for example, see Vieira [1977]). Because the exchange step complements one of the coupling terms during the mapping, we find that  $\sigma^*$  is J-orthogonal

$$P8) \text{ If } \sigma \cdot \sigma^T = I \text{ then } \sigma^* J \sigma^{*T} = J$$

which we can generalize by repeated application of property (P2):

$$P9) \text{ If } \sigma \cdot \sigma^T = I \text{ then } \sigma^* J \sigma^{*T} = J$$

By application of property (P6), we also have

$$P10) \text{ If } \sigma \cdot \sigma^T = I \text{ then } \sigma^{T*} = J \cdot \sigma^{*T} \cdot J$$

### Triangular Layers

We define a lower triangular layer to be a layer of the form

$$L^* = \begin{bmatrix} 0 & & 0 \\ A & & 0 \\ C & D & 0 \end{bmatrix}$$

where  $A$  and  $D$  are lower triangular, and  $C$  is full. We define an upper triangular layer similarly:

$$U^* = \begin{bmatrix} \diagup & A & B \\ 0 & & \\ \hline & & \diagup \\ 0 & 0 & D \end{bmatrix}$$

Property P11)

The exchanged version of a triangular layer is a triangular matrix. This follows from an exchange step (pivoting about  $A$ ) applied to  $L^*$  or  $U^*$

$$L = \begin{bmatrix} \diagup & 0 & \\ A^{-1} & & 0 \\ \hline CA^{-1} & & D \end{bmatrix} \quad U = \begin{bmatrix} \diagup & A^{-1} & \\ 0 & & -A^{-1}B \\ \hline 0 & & D \end{bmatrix}$$

$L$  and  $U$  are triangular because the inverse of an upper (lower) triangular matrix is upper (lower) triangular, and because  $D$  remains unchanged.

Property P12)

There exists a J-orthogonal layer,  $\sigma_J^*$ , that maps any layer into an upper or lower triangular layer (i.e., removes reflections from one side). To show the existence of an  $\sigma_J^*$ , such that

$$A^* \sigma_J^* = T^*$$

we choose  $\sigma$  such that

$$A \cdot \sigma = T$$

$$\text{then } \sigma^* = \sigma^E$$

Property P13)

If we lower triangularize a layer in the scattering domain

$$A^* \sigma^* = L^*$$

then the equivalent transformation,  $\sigma^{*E}$ , applied in the transition matrix domain also yields a lower triangular matrix.

Eigenlayers - Property P14)

From Wilkinson [1965], if the eigenvalues of  $A$  are distinct, then there exists a similarity transform such that

$$A = X \cdot \text{diag}(\lambda_i) \cdot X^{-1} \quad (6.5)$$

where the  $\lambda_i$  are the eigenvalues of  $A$ , and the matrix  $X$  consists of the right eigenvectors of  $A$ ; the  $i^{\text{th}}$  column of  $X$  corresponds to the  $i^{\text{th}}$  eigenvalue.

It then follows that in the scattering domain, any layer can be decomposed into an eigenlayer, a diagonal layer, and an inverse eigenlayer

$$A^* = X^* \cdot \begin{bmatrix} \lambda & & 0 \\ & \ddots & \\ 0 & & \lambda \end{bmatrix} \cdot X^{-1*}$$

which has the important effect of introducing a layer without any reflection coefficients. (Note that zero eigenvalues can be interpreted as "transmission zeroes.")

In the case of repeated eigenvalues property (P14) can be generalized using Jordan (block) form, thus

$$A = X \cdot \begin{bmatrix} J_1 & & 0 \\ & J_2 & \\ 0 & & \ddots \\ & & & J_m \end{bmatrix} \cdot X^{-1}, \quad \text{where } J_i = \begin{bmatrix} \lambda_i & 1 & & 0 \\ & \ddots & \ddots & \\ & & \ddots & 1 \\ 0 & & & \lambda_i \end{bmatrix}$$

and, in the scattering domain (assuming appropriate partitioning)

$$A^* = X^* \cdot \begin{bmatrix} J_1 & | & 0 \\ \hline & | & \\ 0 & | & J_m \end{bmatrix} \cdot X^{-1*}$$

where the reflection coefficients of the eigenvalue layer are zero.

#### Summary of Scattering Operation Properties

P1)  $I = I^*$

P2)  $J = J^*$

$J^{-1} = J$

$J \cdot A = J * A$

P3)  $A \cdot B = C \Rightarrow A^* \cdot B^* = C^*$

P4)  $A^* \cdot B^* = C^* \Rightarrow A \cdot B = C$

P5)  $A \cdot A^{-1} = I \Rightarrow A^* \cdot A^{-1} = I$

If  $\theta \cdot \theta^T = I$ , then

$$P6) \theta^* \cdot (\theta^T)^* = I$$

$$P7) \theta^* \theta^T = I$$

$$P8) \theta^* \cdot J \cdot \theta^{*T} = J$$

$$P9) \theta^* \cdot J \cdot \theta^{*T} = J$$

$$P10) \theta^{T*} = J \cdot \theta^{*T} \cdot J$$

### Triangular Layers

P11) If  $T^*$  is an upper (lower) triangular layer, then  $T$  is an upper (lower) triangular matrix.

P12)  $\exists \theta_J^* \rightarrow A^* \cdot \theta_J^* = T^*$ ,  $A^*$  full,  $T^*$  triangular, upper or lower.

P13)  $A^* \cdot \theta_J^* = T^* \Rightarrow A \cdot \theta = T$ ,  $T$  triangular.

### Eigenlayers

P14) If  $A = X \text{diag}(\lambda_i) X^{-1}$ , where  $X$  are the right eigenvectors of  $A$ , then

$$A^* = X^* \cdot \begin{bmatrix} \lambda & & 0 \\ & \ddots & \\ 0 & & \lambda \end{bmatrix} \cdot X^{-1*}$$



## EXAMPLES

To illustrate the utility of the scattering context, several examples will be developed in scattering terms. The first example, the Hamiltonian system, was presented in Section III.3. The basic equations (3.1) and (3.7) (already in the scattering domain)

$$\begin{aligned} x_{i+1} &= \phi x_i - \Gamma B^{-1} \Gamma^T \lambda_{i+1}, \quad x_0 \text{ given} \\ \lambda_i &= \phi^T \lambda_{i+1} + A x_i, \quad \lambda_n = x_n^T A \end{aligned} \quad (6.5)$$

are presented in Figure 6.4 as a scattering layer. (Since this is a control problem, time through the layer is reversed.) We can now convert from the scattering domain back to the transition domain, pivoting about  $\phi^T$ , producing Figure 6.5, and yielding equations (3.12).

$$M^* = \begin{bmatrix} \phi & -\Gamma B^{-1} \Gamma^T \\ A & \phi^T \end{bmatrix}, \quad M^{*E} = \begin{bmatrix} \phi + \Gamma B^{-1} \Gamma^T \phi^{-T} A & -\Gamma B^{-1} \Gamma^T \phi^{-T} \\ -\phi^{-T} A & \phi^{-T} \end{bmatrix} = M \quad (6.6)$$

This is the matrix we needed to decompose along eigenvector coordinates. Again we note that if  $\phi$  is singular, we will not be able to pivot as required.

Next, invoke property (P14) to reformulate  $M^*$  to include an eigenlayer. Recalling the definitions from Section III.3 for the eigenvectors of  $M$

$$\begin{aligned} M &= E \cdot D \cdot E^{-1} \quad (\text{property P14}) \\ &= \begin{bmatrix} X_u & X_s \\ \Lambda_u & \Lambda_s \end{bmatrix} \cdot \begin{bmatrix} \lambda_u & 0 \\ 0 & \lambda_s \end{bmatrix} \cdot \begin{bmatrix} X_u & X_s \\ \Lambda_u & \Lambda_s \end{bmatrix}^{-1} \end{aligned} \quad (6.7)$$

where  $X_u$ ,  $\Lambda_u$ , and  $\lambda_u$  are the unstable eigen-elements. Applying an exchange step gives  $M^*$

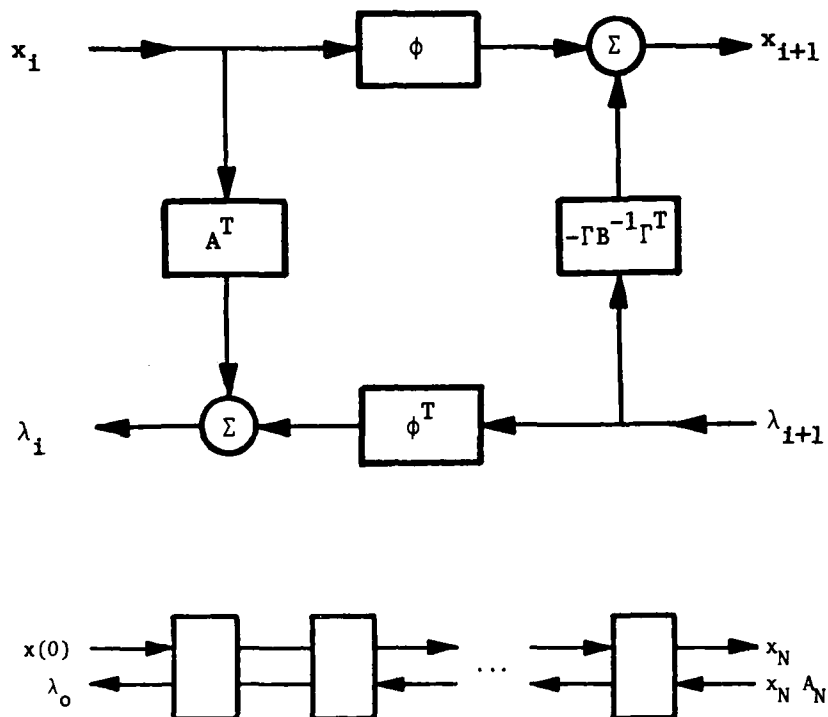


Figure 6.4 Hamiltonian Scattering Matrix

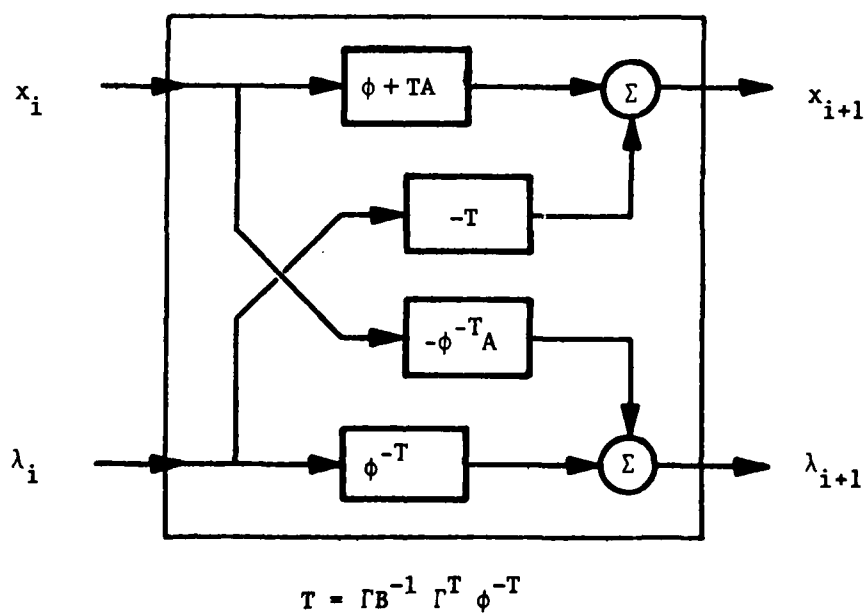


Figure 6.5 Hamiltonian Transition Matrix

$$M^* = \begin{bmatrix} X_u^{-1} & -X_u^{-1}X_s \\ \Lambda_u X_u^{-1} & \Lambda_s - \Lambda_u X_u^{-1}X_s \end{bmatrix} * \begin{bmatrix} \lambda_u^{-1} & 0 \\ 0 & \lambda_s \end{bmatrix} * E^{-1*} \quad (6.8)$$

The sequence of layers  $M_0^* M_1^* M_2^* \dots M_n^*$  appearing in Figure 6.6a can therefore be reconfigured as in Figure 6.6b, i.e.,  $(E_1^* D_1^* E_1^{-1*}) * (E_2^* D_2^* E_2^{-1*}) * \dots * (E_n^* D_n^* E_n^{-1*})$ .

Two simplifications immediately obtain. First, since the Hamiltonian is symplectic,  $\lambda_u^{-1} = \lambda_s$ . Second, since  $E^{-1*} * E^* = I$  from property (P4), many adjacent layers cancel. This leads to the very simple structure of Figure 6.6c, i.e.,  $E^* D^{*n} E^{-1*}$ .

Since  $\lambda_s$  are the stable roots of the system and are therefore less than unity in magnitude, as  $n$  goes to infinity  $(\lambda_s)^n = (\lambda_u)^{-n}$  goes to zero; if we consider an infinite number of layers, there is no transmission through the network (i.e., the two boundary eigenlayers are decoupled). The reflection coefficient relating the state  $x_i$  to the adjoint variable  $\lambda_i$  is therefore

$$\lambda_{i+1} = \Lambda_u X_u^{-1} x_{i+1} = P x_{i+1}$$

as expected.

This approach introduces an interesting alternate solution to the normal eigenvector approach. Note that constructing the Hamiltonian in the scattering domain does not require inverting the state transition matrix. Therefore, if an eigenvalue decomposition routine could be written to calculate the scattering domain eigenvectors directly, the pivot step (6.6) could be avoided. This has not yet been fully pursued.

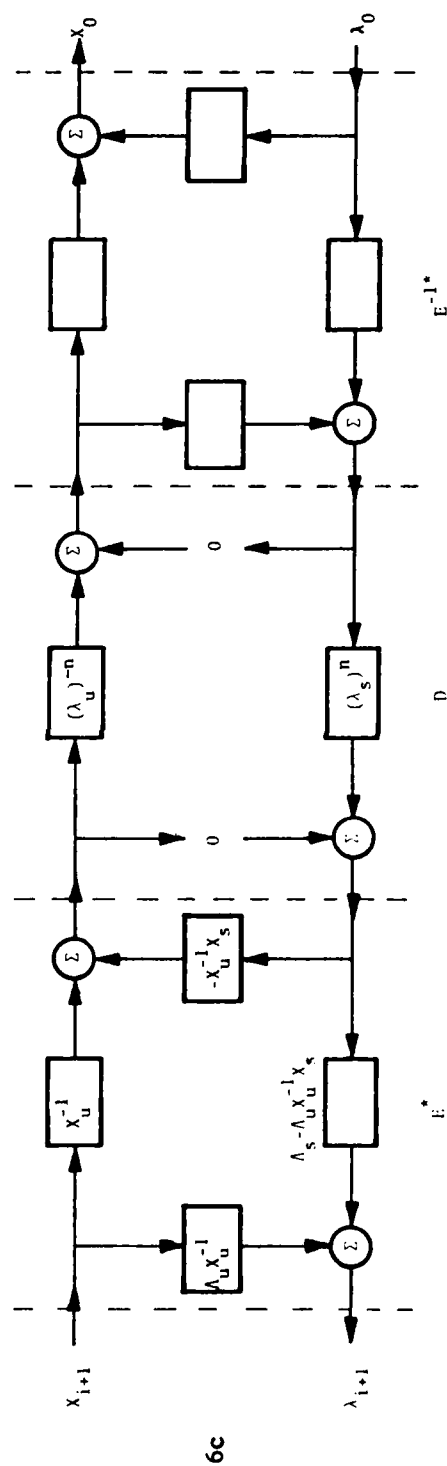
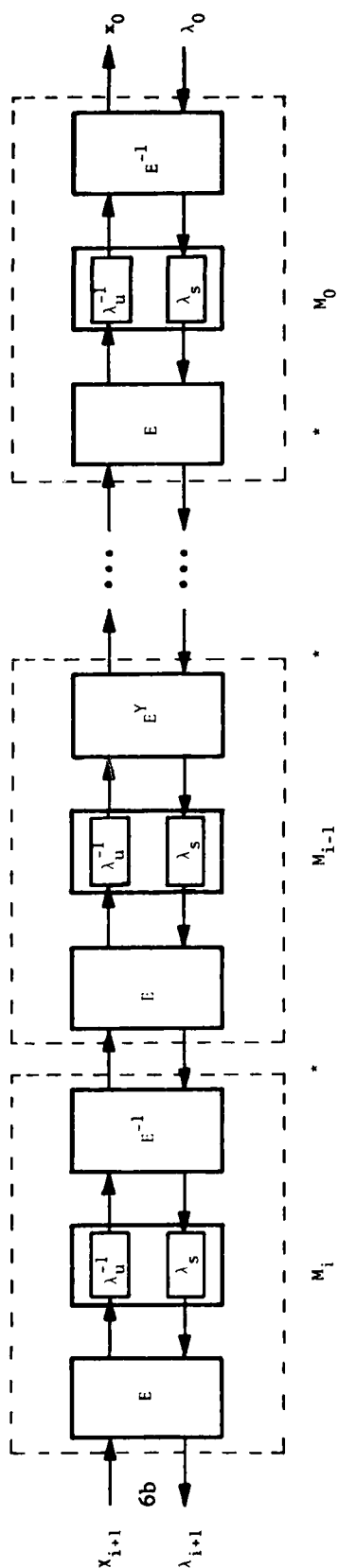
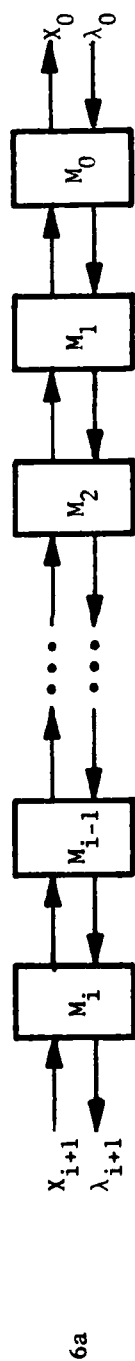


Figure 6.6abc Eigenlayers

Another estimation example, which is important in the sequel, introduces the propagation of error covariance matrices in the scattering framework. From Friedlander [1976], we note that we can update the forward error covariance, and the inverse smoothed error covariance, by constructing a layer as shown in Figure 6.7 and cascading it with the layer shown in Figure 6.8. The relationship between the scattering matrices-- the accumulated covariance of Figure 6.7 and the update of Figure 6.8 -- and the transition matrix domain can be clarified by examining the normal update equations for the requisite quantities. For example, for the estimate error covariance, we have:

$$P(t+1|t) = \phi P(t|t-1) \phi' + \Gamma Q \Gamma^T - \phi P(t|t-1) H^T (R + H P(t|t-1) H^T)^{-1} H P(t|t-1) \phi^T \quad (6.9)$$

From the star product definition, Equation (6.1), we see this must be converted to the form

$$\rho_p = \rho_2 + a_2 \rho_1 [I - r_2 \rho_1]^{-1} \alpha_2 \quad (6.10)$$

which follows immediately, after applying the matrix inversion lemma

$$(A + BCD)^{-1} = A^{-1} - A^{-1} B (DA^{-1} B + C^{-1})^{-1} DA^{-1} \quad (6.11)$$

to give

$$P(t+1|t) = \Gamma Q \Gamma^T + \phi P(t|t-1) [I + H^T R^{-1} H P(t|t-1)]^{-1} \phi^T \quad (6.12)$$

Similarly for  $\phi_0$ , the closed loop Kalman-filter transition matrix, we can write

$$\phi_0(t+1,0) = \phi [I + P(t|t-1) H^T R^{-1} H] \phi_0(t,0) \quad (6.13)$$

which is exactly analogous to the star product quantity of (6.1)

$$a_0 = a_2 [I - \rho_1 r_2]^{-1} a_1 \quad (6.14)$$

And finally, we can also present the equation for  $W$ . We shall be discussing the significance of  $W$  in the next section.  $W$  is basically the smoothed error covariance  $P_j^{-1} | [j, k]$ , which we can write as

$$W(t+1|t) = W(t|t-1) + \phi_0^T(t,0) H^T R^{-1} H [I + P(t|t-1) H^T R^{-1} H]^{-1} \phi_0(t,0) \quad (6.15)$$

again, analogous to the star product quantity

$$r_0 = r_1 + \alpha_1 r_2 (I - \rho_1 r_2)^{-1} a_1 \quad (6.16)$$

The connection between the equations (6.9) through (6.16) and the scattering diagrams, Figures 6.7 and 6.8, follow immediately.

We shall return to examine this example in close detail because of one striking observation. As Friedlander et al., noted [1976], if we combine two "accumulation" layers, instead of an "accumulation" and an "update" layer, we generate a very interesting sequence. Beginning with covariances at time  $t=0$  we generate the covariances for  $t=1$ ,  $t=2$ ,  $t=4$ ,  $t=8$ , ...; at each iteration,  $i$ , we generate

$$P(2^i), W(2^i), \text{ and } \phi_0(2^i)$$

Given a constant parameter system and the objective of calculating the steady-state error covariances, this approach would then converge exponentially compared to the previous iterative algorithms. This is the subject of our next few sections.

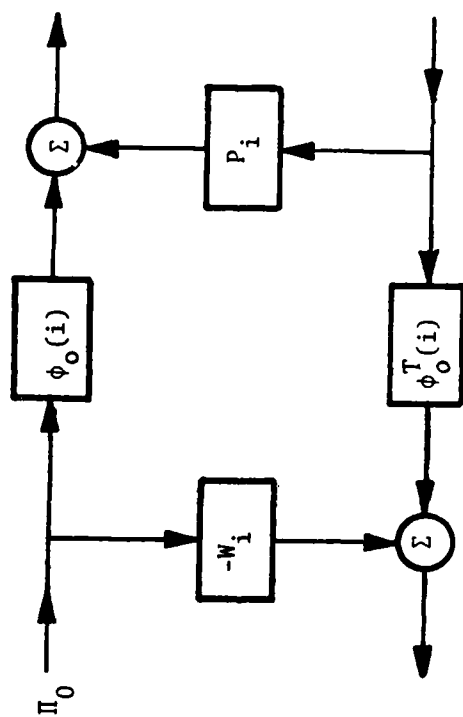


Figure 6.7 Accumulated Covariance Layer

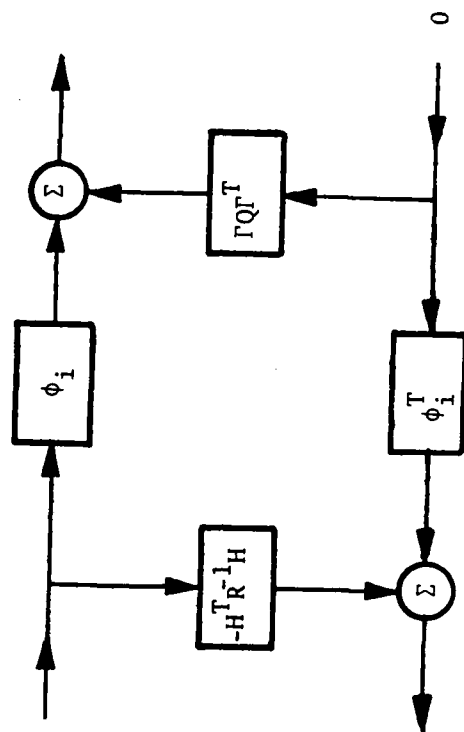


Figure 6.8 Update Layer

### III.7 Square Root Doubling

We return again to another class of algorithms for solving the steady state Riccati equation. In this case, we partition an interval of data into subintervals. We then make estimates of the states within each interval based only on data from the subinterval. Then, global calculations are used to combine information from subintervals to obtain optimal estimates at the subinterval endpoints; in this fashion subintervals can be "recombined" to yield the answer over the entire interval.

Besides the work by Morf, Dobbins, Friedlander, and Kailath [1978], various others have expanded on this idea. Following Womble and Potter [1975] and Lainiotis [1976], Bierman and Sidhu [1977] focused on doubling formulas for the continuous case. We shall give an algebraic derivation for the discrete case, using the scattering theory of the last section, and then develop a pure scattering framework for rapid derivation of equations.

Schematically, we can picture the interval of data in Fig. 7.1 being partitioned as in Fig. 7.2. Focusing on the interval from  $j$  to  $k$ , Morf et. al. [1978] noted that the Markov property of the  $\{x\}$  process implies that for the purpose of state estimation outside the observation interval, the interval can be summarized in terms of four quantities (two "boundary" state estimates and their covariances):



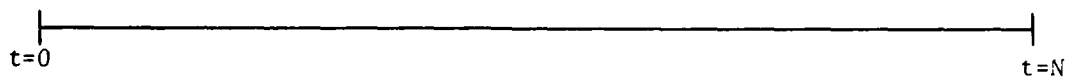
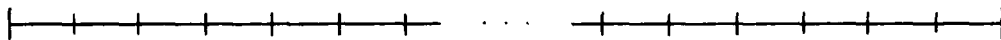
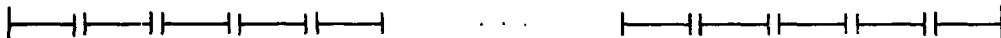


Figure 3.7.1 ORIGINAL INTERVAL

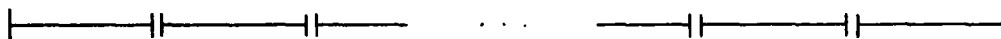
Beginning with the complete interval,



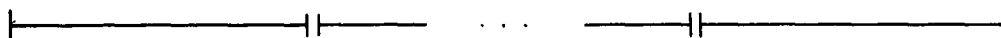
Solve over each interval for the endpoint state information describing that interval,



Combine state information at every other adjacent endpoint to form doubled intervals twice as long,



Continue combining intervals, until



the salient characteristics for the entire interval, from  $t = 0$  to  $t = N$ , are determined

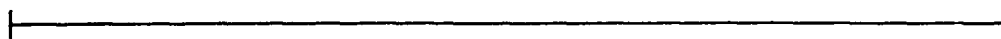


Figure 3.7.2 INTERVAL SEGMENTATION

- 1)  $\hat{x}_{k|[j,k-1]}$  , the predicted estimate\*
- 2)  $P_{k|[j,k-1]}$  , the predicted error covariance
- 3)  $\hat{x}_j|[j,k-1]$  , the smoothed estimate
- 4)  $P_j|[j,k-1]$  , the smoothed error covariance

and one relation, describing the closed-loop state transition matrix  $\phi_0(j,k-1)$ ; this interval transition matrix relates the state information at one endpoint to the state information at the other endpoint (see Figure 7.3).

The scattering domain will be used to quickly derive the algorithm, but considerable insight can be gained by examining the theoretical basis for the algorithm. If we begin by assuming the value of the state at the initial endpoint,  $x_j$  , is known, then we have

$$\hat{x}_{k|x_j|[j,k-1]} = \phi_0(j,k-1)x_j + \hat{x}_{k|x_j=0,[j,k-1]} \quad (7.1)$$

where  $\phi_0$  is the closed-loop state transition matrix for the Kalman filter, assuming the initial error covariance is zero. That is, the best estimate of  $x$  at the endpoint is the sum of the best estimate assuming zero initial conditions plus the known initial condition, propagated (via  $\phi_0$ ) over the interval.

Assuming we don't know  $x_j$  , the best we can do is to use the best estimate of  $x_j$  ,  $\hat{x}_j|[j,k-1]$  , to give

$$\hat{x}_{k|[j,k-1]} = \phi_0(j,k-1)\hat{x}_j|[j,k-1] + \hat{x}_{k|x_j=0,[j,k-1]} \quad (7.2)$$

---

\* Recall that by  $\hat{x}_{k|[j,k-1]}$  we mean the estimate  $\hat{x}$  at time  $k$  given observations at times  $j$  through  $k-1$ .

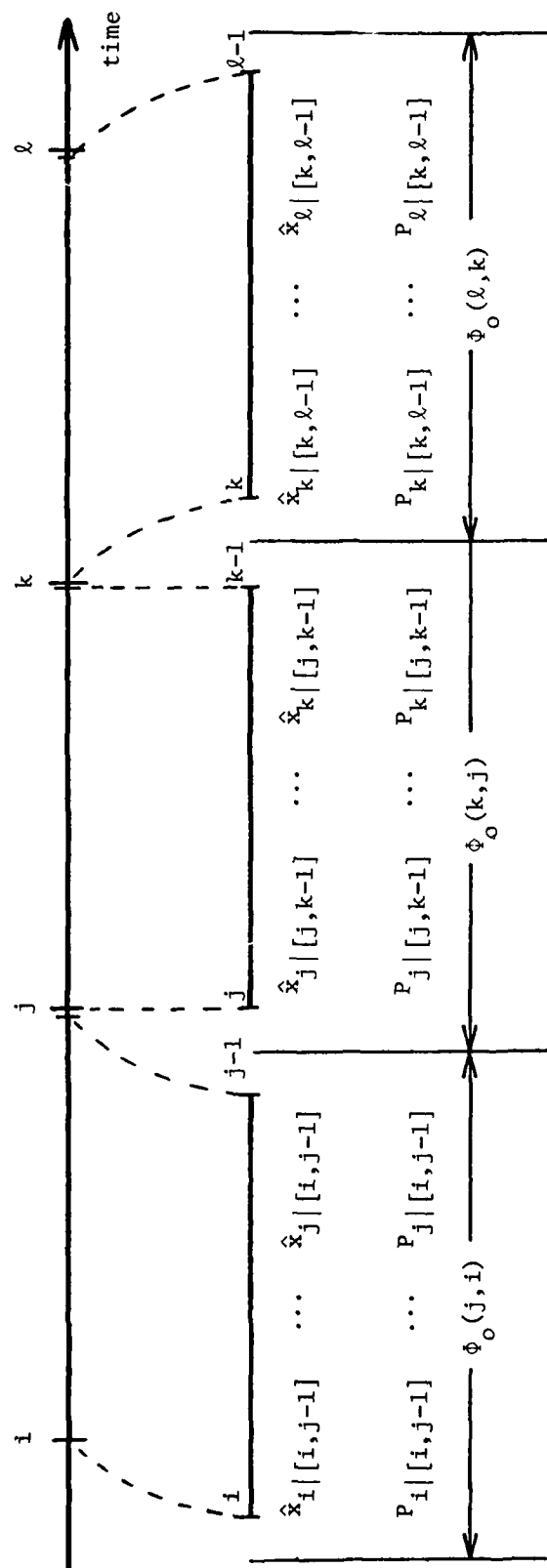


Figure 3.7.3  
INTERVAL DESCRIPTION

Similarly, we can write the associated error covariance as

$$P_{k|[j,k-1]} = \phi_o(j,k-1)P_{k|[j,k-1]}\phi_o^T(j,k-1) + P_{k|x_j,[j,k-1]} \quad (7.3)$$

If we write the other interval equations-- for  $\hat{x}_j|[j,k-1]$  and  $P_{j|[j,k-1]}$ -- then we would be ready to consider combining  $\hat{x}_k|[j,k-1]$  with  $\hat{x}_j|[k,\ell-1]$  to produce  $\hat{x}_\ell|[k,\ell-1]$ . Morf, et al., develops this formulation.

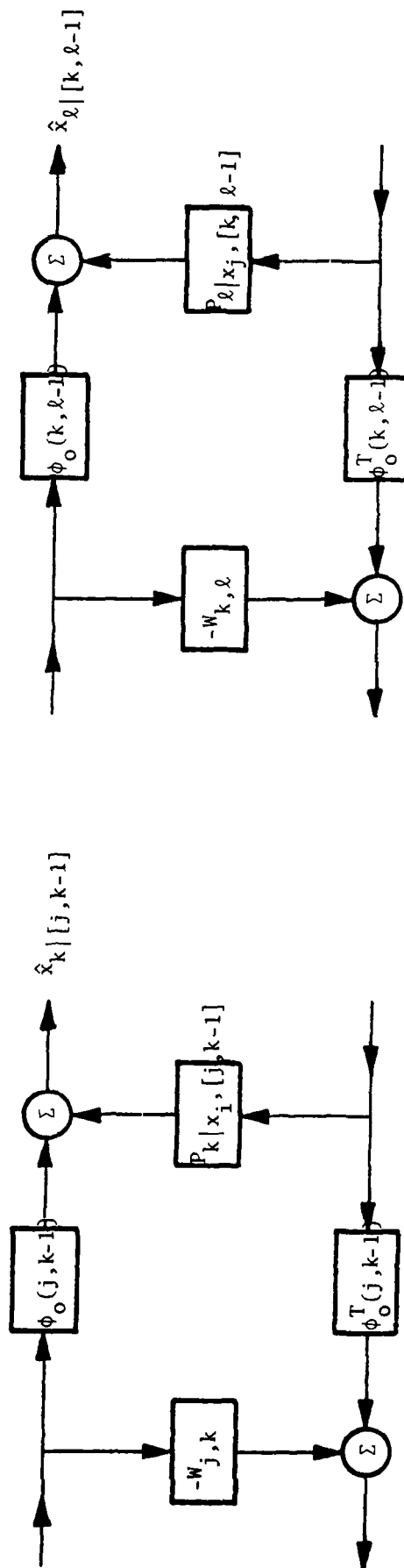
Another approach is to recognize that these equations and quantities are precisely those terms we were treating in the last example of the previous section. Making the association with Figure 6.8, we can construct Figure 7.4 and write down the equations immediately. Assuming time invariant coefficients, we have:

$$\begin{bmatrix} \phi_o(i) & P(i) \\ \bar{W}(i) & \phi_o^T(i) \end{bmatrix} * \begin{bmatrix} \phi_o(i) & P(i) \\ \bar{W}(i) & \phi_o^T(i) \end{bmatrix} = \begin{bmatrix} \phi_o[I-P\bar{W}]^{-1}\phi_o & P+\phi_o P[I-\bar{W}P]^{-1}\phi_o^T \\ \bar{W}+\phi_o^T \bar{W}[I-P\bar{W}]^{-1}\phi_o & \phi_o^T[I-\bar{W}P]^{-1}\phi_o^T \end{bmatrix} \quad (7.4)$$

Making a change of variables, of  $W = -\bar{W}$ , we get

$$\begin{aligned} (a) \quad P(2i) &= P(i) + \phi_o(i)P(i)[I+W(i)P(i)]^{-1}\phi_o^T(i) \\ (b) \quad W(2i) &= W(i) + \phi_o^T(i)W(i)[I+P(i)W(i)]^{-1}\phi_o(i) \\ (c) \quad \phi_o(2i) &= \phi_o(i)[I+P(i)W(i)]^{-1}\phi_o(i) . \end{aligned} \quad (7.5)$$

Recall that the objective is to derive a square root algorithm for propagating these equations. Initially, we will guess that (7.5c) cannot be reduced to a square-root form unless  $\phi_o$  is symmetric. [In



$$W_{a,b} = P_a^{-1}[a, b-1]$$

Figure 7.4 Doubling Layers

the next section we will gain some insight into why this is the case.]

The other two equations, (7.5a) and (7.5b) are close to the form we could readily recognize and exploit; we need

$$A(2i) = A(i) + BT_1 T_1^T B^T \quad (7.7)$$

$$\begin{bmatrix} A_{2i}^{1/2} & 0 \end{bmatrix} = \begin{bmatrix} A_i^{1/2} & BT_1 \end{bmatrix} \cdot \sigma$$

where  $A^{1/2}$  is lower triangular, and  $\sigma$  is an orthogonal matrix, constructed so as to triangularize the matrix  $\begin{bmatrix} A_i^{1/2} & BT_1 \end{bmatrix}$ .

In the derivation that follows, if the subscript is missing on a time-varying quantity, "i" is assumed. Beginning with the equation for  $P(2i)$  from (7.5),

$$\begin{aligned} P(2i) &= P + \phi_o P[I + WP]^{-1} \phi_o^T \\ &= P + \phi_o P^{1/2} P^{T/2} [I + WP_i]^{-1} P^{-T/2} P^{T/2} \phi_o^T, \text{ where } P^{1/2} P^{T/2} = P \\ &= P + \phi_o P^{1/2} [I + P^{T/2} W^{1/2} W^{T/2} P^{1/2}]^{-1} P^{T/2} \phi_o^T \\ &= P + \phi_o P^{1/2} [I + XX^T]^{-1} P^{T/2} \phi_o^T, \text{ where } X = P^{T/2} W^{1/2} \end{aligned} \quad (7.8)$$

and similarly, for  $W(2i)$

$$\begin{aligned} W(2i) &= W + \phi_o^T W^{1/2} W^{T/2} [I + PW^{1/2} W^{T/2}]^{-1} \phi_o \\ &= W + \phi_o^T W^{1/2} [I + W^{T/2} P^{1/2} P^{T/2} W^{1/2}]^{-1} W^{T/2} \phi_o \\ &= W + \phi_o^T W^{1/2} [I + X^T X]^{-1} W^{T/2} \phi_o \end{aligned} \quad (7.9)$$

From Section III.4, we compute  $[I + XX^T]^{-1}$  by choosing  $\sigma_1$  so that  $\sigma_1$  is orthogonal,

$$\sigma_1 \cdot \sigma_1^T = I \quad , \quad \sigma_1 = \begin{bmatrix} T_1 & T_{12} \\ T_{21} & T_2 \end{bmatrix}$$

and so that it triangularizes  $[I; X^T]$

$$[I; X^T] \sigma_1 = [R; 0] \quad R = \begin{array}{c} \triangle \end{array}$$

Then

$$\begin{aligned} [I + X^T X] &= T_1^T T_1^T & [I + X X^T] &= T_2 T_2^T & (7.10) \\ R &= T_1^{-T} & X &= T_{21} T_1^{-1} \end{aligned}$$

Rewriting (7.8), we have

$$P(2i) = P + \phi_0 P^{\frac{1}{2}} T_2^T T_2^T P^{T/2} \phi_0^T \quad (7.11)$$

We now choose a second orthogonal transform,  $\sigma_2$ , so that

$$[\phi_0 P^{\frac{1}{2}} T_2^T; P^{\frac{1}{2}}] \cdot \sigma_2 = [P_{2i}^{\frac{1}{2}}; 0] \quad (7.12)$$

As long as  $P^{\frac{1}{2}}$  has at most  $n$  columns, any orthogonal mapping,  $\sigma_2$ , will suffice. (Triangularizing mappings, however, reduce later computations.)

Recalling the properties of orthogonal transformations we outlined in Section III.4, we can now write

$$\begin{bmatrix} I & X^T & 0 \\ 0 & \phi_O P_i^{1/2} & P_i^{1/2} \end{bmatrix} \cdot \begin{bmatrix} \sigma_1 & 0 \\ 0 & I_n \end{bmatrix} \cdot \begin{bmatrix} I_n & 0 \\ 0 & \sigma_2 \end{bmatrix} \\
= \begin{bmatrix} T_1^{-T} & 0 & 0 \\ \phi_O P_i^{1/2} T_{21} & \phi_O P_i^{1/2} T_2 & P_i^{1/2} \end{bmatrix} \cdot \begin{bmatrix} I_n & 0 \\ 0 & \sigma_2 \end{bmatrix} \quad (7.13)$$

$$= \begin{bmatrix} T_1^{-T} & 0 & 0 \\ \phi_O P_i^{1/2} T_{21} & P_{2i}^{1/2} & 0 \end{bmatrix} = M_1 \quad (7.14)$$

Since  $\sigma_1$  and  $\sigma_2$  were arbitrary, except for their orthogonal nature, we note that we can replace them with any orthogonal mapping that yields (7.14)

$$M_1 = \begin{bmatrix} T_1^{-T} & 0 & 0 \\ \phi_O P_i^{1/2} T_{21} & P_{2i}^{1/2} & 0 \end{bmatrix} \Leftrightarrow \begin{bmatrix} \diagdown & 0 & 0 & 0 \\ X & \diagdown & \diagdown & \diagdown \\ X & \diagdown & 0 & 0 \\ \diagdown & X & \diagdown & \diagdown \end{bmatrix} \quad (7.15)$$

We will soon see that these quantities,  $T_1^{-T}$  and  $\phi_O P_i^{1/2} T_{21}$ , are important for calculating  $W(2i)$  and  $\phi_O(2i)$ ; in the next section we shall examine why this might be anticipated.

To calculate  $W(2i)$  of equation (7.9), we already have most of the quantities we need:



$$W(2i) = W + \phi_0^T W^{\frac{1}{2}} [I + X^T X]^{-1} W^{T/2} \phi_0 \quad (7.16)$$

$$\begin{aligned} W_{2i}^{\frac{1}{2}} W_{2i}^{T/2} &= W^{\frac{1}{2}} W^{T/2} + \phi_0^T W^{\frac{1}{2}} [I + X^T X]^{-1} W^{T/2} \phi_0 \\ &= W^{\frac{1}{2}} W^{T/2} + \phi_0^T W^{\frac{1}{2}} T_1^T T_1 W^{T/2} \phi_0 \end{aligned}$$

To calculate  $P_{2i}$ , we needed  $W^{T/2}$  (for calculating  $W^{T/2} P^{\frac{1}{2}}$ ). We have  $\phi_0$  and  $T_1^T$  available, so consider calculating the right square root of  $W$ ,  $W^{T/2}$ , instead of  $W^{\frac{1}{2}}$ . Selecting our third orthogonal transformation,  $\sigma_3$ , we have

$$\sigma_3 \cdot \begin{bmatrix} T_1^T W_i^{T/2} \phi_0 \\ W_i^{T/2} \end{bmatrix} = \begin{bmatrix} 0 \\ W_{2i}^{T/2} \end{bmatrix} \iff \begin{bmatrix} 0 \\ X \backslash 0 \end{bmatrix} \quad (7.17)$$

which is the solution we required.

To calculate  $\phi(2i)$  we begin by noting the lack of symmetry in equation (7.5c) and the need for calculating  $[I + PW]^{-1}$ , which isn't available from previous calculations. Instead, we have only the two quantities  $T_1^T$  and  $\phi_0^T P^{\frac{1}{2}} T_{21}$ :

$$\begin{aligned} T_1^T &= [I + X^T X]^{-\frac{1}{2}} \\ &= [I + W^{T/2} P W^{\frac{1}{2}}]^{-\frac{1}{2}} \end{aligned}$$

and

$$\begin{aligned} \phi_0^T P^{\frac{1}{2}} T_{21} &= \phi_0^T P^{\frac{1}{2}} X X^{-1} T_{21} = \phi_0^T P^{\frac{1}{2}} P^{T/2} W T_1 \\ &= \phi_0^T P W^{\frac{1}{2}} [I + W^{T/2} P W^{\frac{1}{2}}]^{-T/2} \end{aligned}$$

We need an expression for  $\phi_0(2i)$  that requires  $[I + X^T X]^{-1}$  :

$$\begin{aligned}\phi_0(2i) &= \phi_0 [I + PW]^{-1} \phi_0 \\ &= \phi_0 W^{-T/2} [I + W^{T/2} P W^{1/2}]^{-1} W^{T/2} \phi_0\end{aligned}$$

which requires computing  $W^{-T/2}$ . However,

$$\begin{aligned}W^{-T/2} &= W^{-T/2} + P W^{1/2} - P W^{1/2} \\ &= W^{-T/2} [I + W^{T/2} P W^{1/2}] - P W^{1/2}\end{aligned}$$

so

$$\begin{aligned}\phi_0(2i) &= \phi_0 (W^{-T/2} [I + W^{T/2} P W^{1/2}] - P W^{1/2}) ([I + W^{T/2} P W^{1/2}]^{-1} W^{T/2} \phi_0) \\ &= \phi_0 \phi_0 - \phi_0 P W^{1/2} [I + W^{T/2} P W^{1/2}]^{-1} W^{T/2} \phi_0 \\ &= \phi_0 \phi_0 - \phi_0 P^{1/2} T_{21}^T T_1^T W^{T/2} \phi_0\end{aligned}$$

as we desired.

For initial conditions, we look to the initial layer for the scattering problem. From Friedlander [1976] we have:

$$\begin{aligned}P_0^{1/2} &= \Gamma Q^{1/2} \\ W_0^{T/2} &= R^{-T/2} H \\ \phi_0 &= \phi\end{aligned}$$

#### Comments on the Strengths and Weaknesses of Square-Root Doubling

Notice that this derivation leads to updating equations which require no explicit inverses. If we now introduce the convention used

by Morf [1978], whereby  $R^{*1/2} = T_1^{-T}$  and  $K^* = \phi_0 P_i^{1/2} T_{21}$ , we have

$$\begin{bmatrix} I & W^{T/2} P^{1/2} & 0 \\ 0 & \phi_0 P^{1/2} & P^{1/2} \end{bmatrix} \cdot \sigma = \begin{bmatrix} R^{*1/2} & 0 & 0 \\ K^* & P_{2i}^{1/2} & 0 \end{bmatrix}$$

$$\sigma_3 \cdot \begin{bmatrix} (R^*)^{-1/2} W^{T/2} \phi_0 \\ W^{T/2} \end{bmatrix} = \begin{bmatrix} 0 \\ W_{2i}^{T/2} \end{bmatrix}$$

$$\phi_0(2i) = \phi_0 (I - K^* (R^*)^{-1/2} W^{T/2}) \phi_0$$

The first equation provides  $(R^*)^{1/2}$ ; the second and third equations require  $(R^*)^{-1/2}$ .

As a second observation, note that repeated eigenvalues pose no problems, and that state transition matrices can be singular. However, the measurement covariance matrix,  $R$ , must be inverted to determine the initial conditions.

Morf et al. also note the ready adaptability of this algorithm to parallel processing. Each processor can independently compute estimates for a distinct interval. Later, the intervals can be joined, again in parallel (see Chapter IX). This procedure can be generalized to the time-varying problem.

### III.8 Scattering Revisited

In the previous section we showed how to begin with results from the scattering domain, and develop them algebraically into a square-root algorithm. In this section we will consider deriving algorithms

solely in the scattering domain, developing the concept of the "square-root" of a medium, orthogonal scattering layers, and the triangularization of a network. The results are salutary; we can derive the square-root algorithms quickly, and with new insight into the intrinsic constitution of this class of solutions.

We switch now from the controller application, presenting the Hamiltonian for the estimation problem in the scattering format:

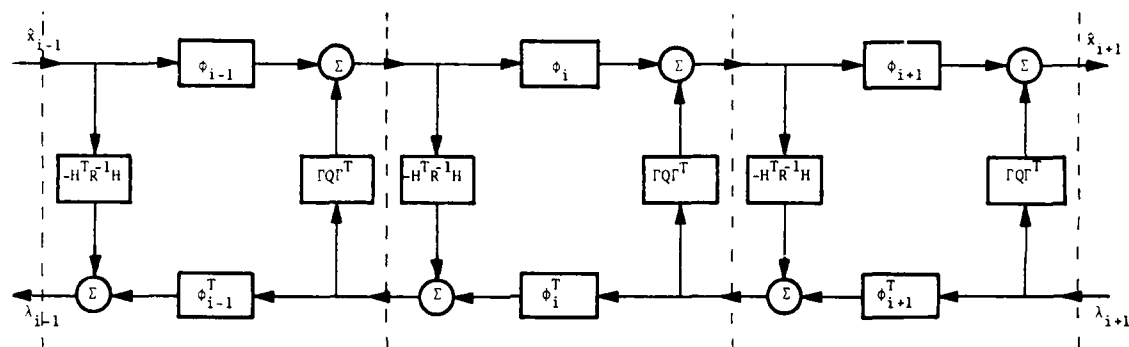
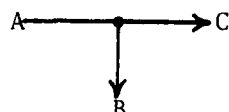
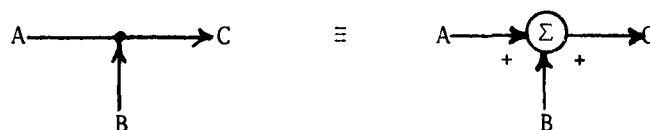


Figure 8.1 Scattering Hamiltonian for the Estimation Problem

In this diagram, a node of the form



indicates a passive branch, where  $A = B = C$ . A node of the form



indicates summation. We read this block schematic exactly as we would read a transmission matrix block diagram, e.g., for

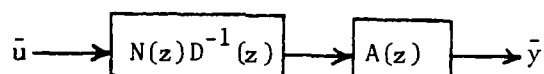


Figure 8.2

we would lift the equation  $\bar{y} = A \cdot N \cdot D^{-1} \bar{u}$ , preserving the order of the matrix calculations. Expanding on this concept, we note that if we take the square-root of a matrix  $A$

$$A = A^{\frac{1}{2}} A^{T/2} \quad (8.1)$$

this translates into the schematic

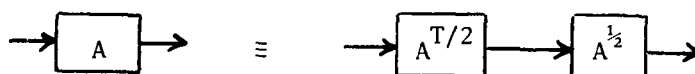


Figure 8.3

With this background we are able to exploit the strong symmetry of Figure 8.1.

We begin by defining the transpose of a network to be the network reflected about the defined line of symmetry, where all quantities (path gains) are transposed, and all arrows are reversed; this last condition exchanges all branch and summation nodes.

We then define the square root of a network to be a network which has a line of symmetry such that when divided along the line of symmetry, one partition is the transpose of the other. Looking at Figure 8.1, we note that one square root of the scattering network will be as shown in Figure 8.4 .

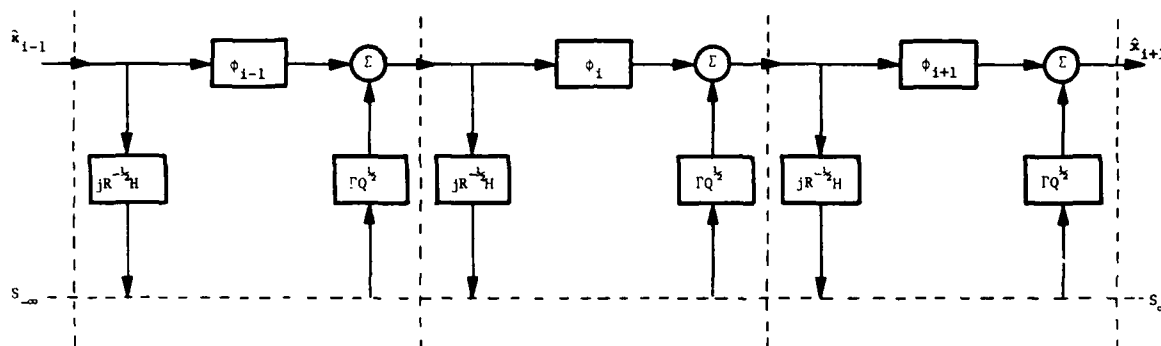


Figure 8.4 Square Root Scattering Hamiltonian

The square-root operation introduces complex arithmetic, which appears to be an added complication compared to the algebraic approach presented earlier. The complication is ephemeral, however, arising because of the minus sign injected by the exchange step (see Section III.6).

Examining the connections at the line of symmetry,  $S_\infty \dots S_\infty$ , notice that any orthogonal mapping could be attached without altering the terminal characteristics of the network. For example, if we contemplate a simple orthogonal network such as a permutation layer,

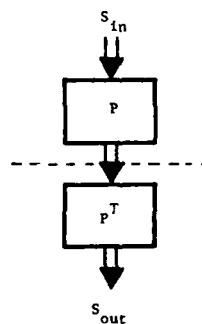


Figure 8.5

it is clear that  $S_{in}$  and  $S_{out}$  are identical-- the signals are "crossed" at the border, and then uncrossed. Any mapping with this characteristic would suffice. Recalling from Section (III.6), property (P7),

$$\sigma \cdot \sigma^T = I \Rightarrow \sigma * \sigma^T = I$$

any orthogonal mapping is a candidate. However, we are interested in orthogonal mappings which correspond via an exchange step to transmission domain mappings. From property (P9), these were

$$\sigma \cdot \sigma^T = I \Rightarrow \sigma^* * J * (\sigma^*)^T = J$$

The J-unitary property, in this case, is an imposed boundary condition.

It also has the welcome effect of eliminating the complex arithmetic.

Thus, Figure 8.4 becomes

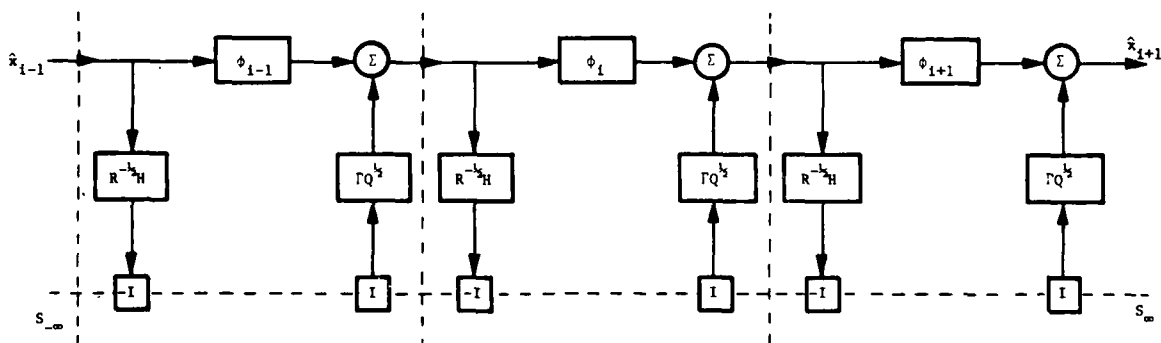


Figure 8.6 J-Unitary Mapping

with the border we require. Henceforth, the J-layer boundary will be maintained as a boundary condition.

When using the star-product, we combine two layers into a single layer. We have a similar objective when working with scattering square-roots: to transform a two-layer network

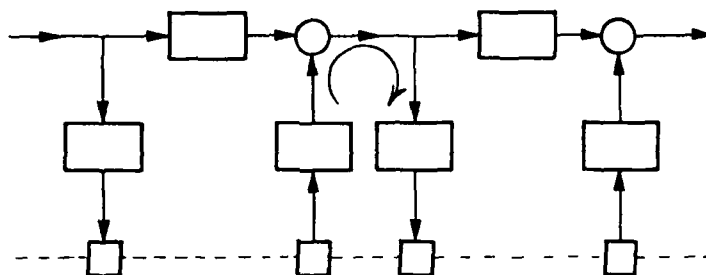


Figure 8.7 Square-root Star-product Definition

into an equivalent single layer

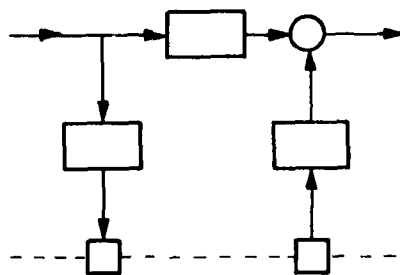


Figure 8.8 Square-root Star-product Definition

If we consider only terminal behavior (in the network theory sense), then by introducing a J-orthogonal layer  $\theta^*$  to Figure 8.7 we get Figure 8.9 .



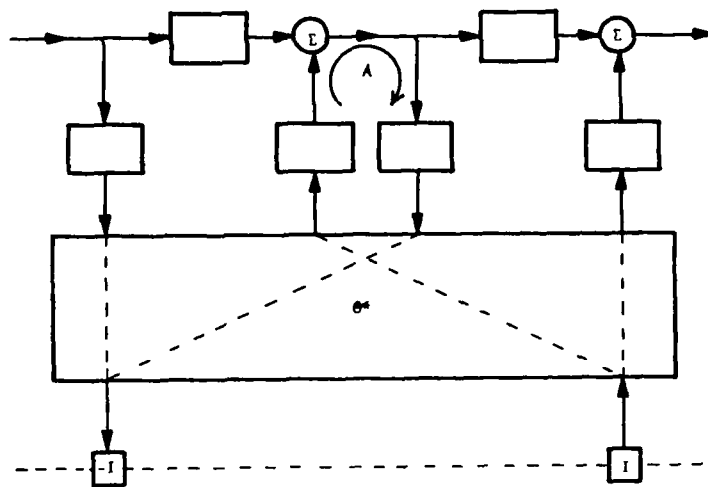


Figure 8.9 Detail of Combined Layer

At the terminals, the mapping  $\sigma^*$  has the desired effect of producing a single layer.

To understand the structure of  $\sigma^*$ , we consider decomposing it into two concatenated mappings

$$\sigma^* = \sigma_1^* * \sigma_2^* \quad (8.2)$$

$\sigma_1^*$  will be used to compress two unidirectional paths (for example, two downward paths) into a single path;  $\sigma_2^*$  will be used to uncross the paths we find crossed in  $\sigma^*$  of Figure 8.9.

The first step is to demonstrate that unidirectional scattering layers (where one direction neither transmits nor reflects) can be mapped into a triangular layer. We begin by considering a structure of the form:

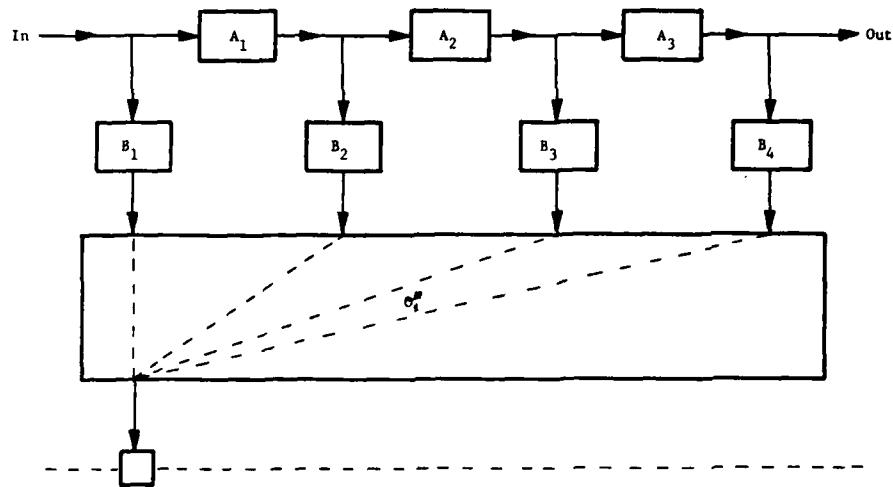


Figure 8.10 Simplified Boundary Mapping

Reorganizing and rotating this schematic into canonical form (as a standard scattering diagram), we have

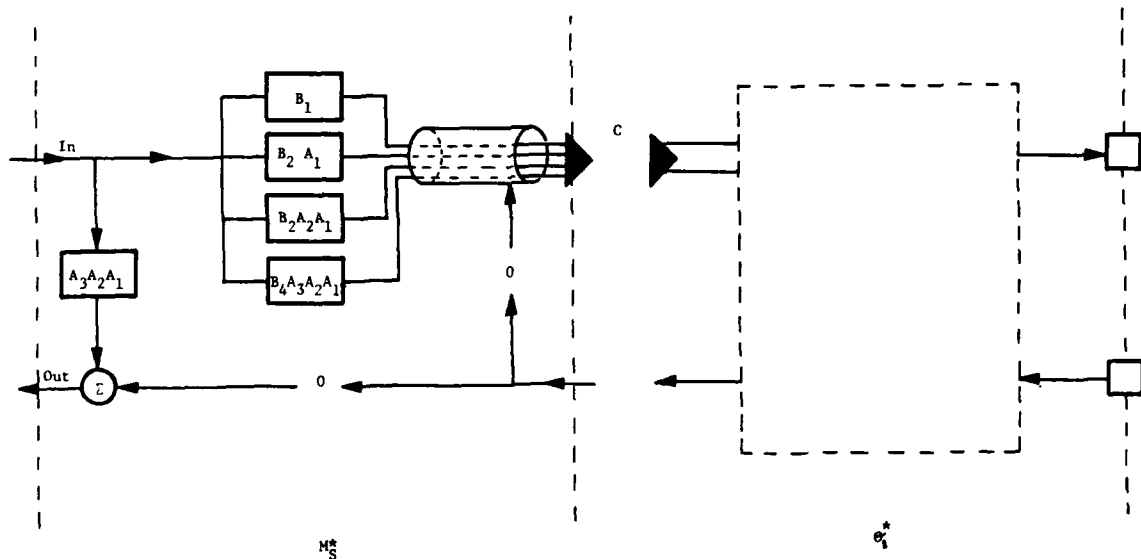


Figure 8.11 Combined Layer in Canonical Form

The zero transmission and reflection gains of  $M_S^*$  make this an especially simple scattering layer to work with.

Conceptually, we can say that the rank of the signal at  $C$  is, at most equal to the rank of the signal at  $In$ . We saw earlier that orthogonal

matrices can be used to consolidate rank by triangularizing and producing columns (or rows) of zeroes; any orthogonal triangularization,  $\bar{\theta}$ , can be used to reduce  $M_S^*$  to consolidated (triangular) form.

Aggregating the non-zero transmission gains produces

$$\begin{bmatrix} \bar{\theta} & 0 \\ 0 & 0 \end{bmatrix} * \begin{bmatrix} B_1 \\ B_2 A_1 \\ B_3 A_2 A_1 \\ B_4 A_3 A_2 A_1 \\ A_3 A_2 A_1 \end{bmatrix} 0 = \begin{bmatrix} X \\ 0 \\ 0 \\ 0 \\ A_3 A_2 A_1 \end{bmatrix} 0 \quad (8.3)$$

$\theta_1^* \quad * \quad M_S^*$

The need for  $\theta_2^*$  of (8.2) arises when we notice that the scattering problem would segment into two unidirectional layers (flowing in opposite directions) with the elimination of the loop,  $A$ , in Figure 8.9. This gives rise to the second application of orthogonal triangularization, the application discussed in Section III.7. We need to unwind the following loop:

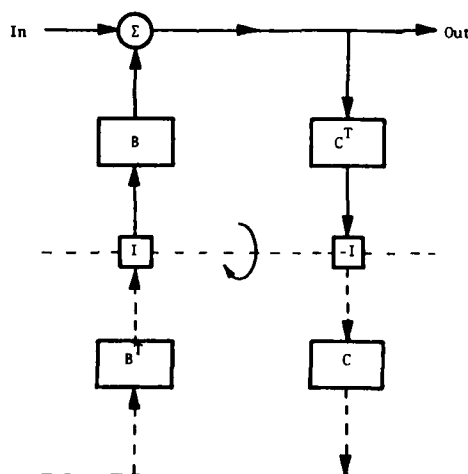


Figure 8.12 Definition of Loop Unwinding

(Note that the lower square-root layer was introduced for clarity.)

If we consider In and Out to be one side of a layer, this layer can be transformed into canonical form by re-ordering the summation computation and determining the transfer functions between the terminals:

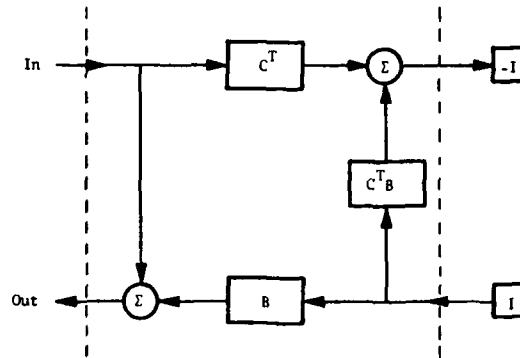
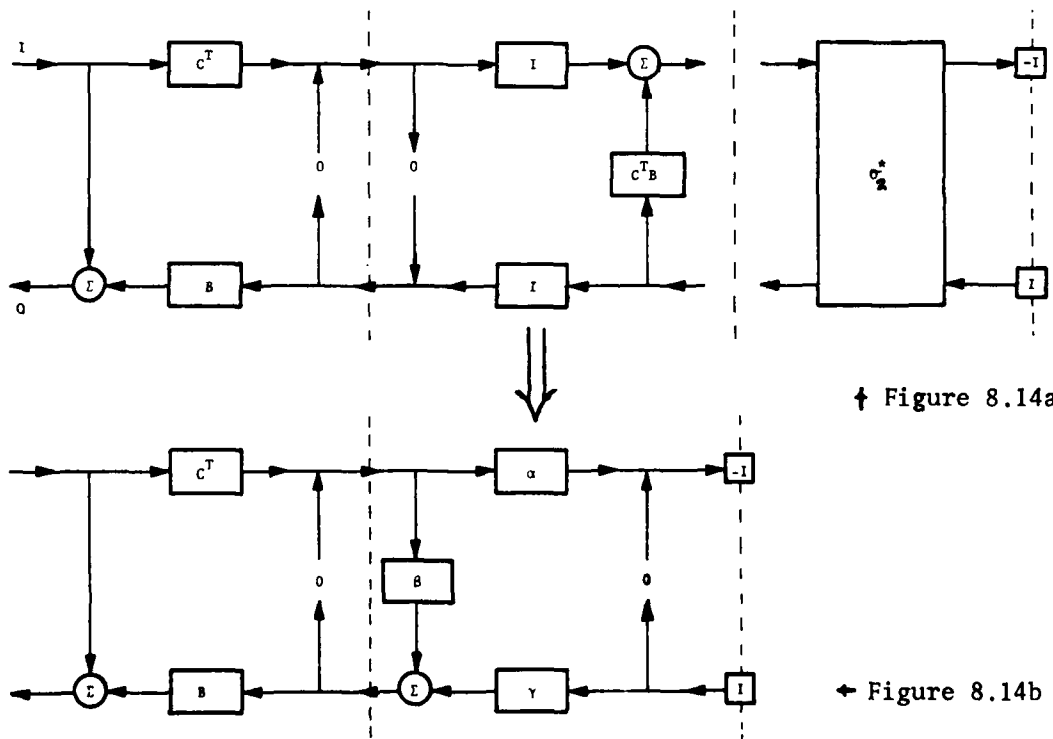


Figure 8.13 Transformation to Canonical Form

The next step is to eliminate the feedback term  $C^T_B$  from this structure by arranging for all of the cross-terms to flow in one direction only. Mapping Figure 8.14a into Figure 8.14b would have the desired result.



Finding this mapping is equivalent to asking whether an orthogonal transformation can always be applied to an upper triangular matrix  $U^*$  to map it into a lower triangular matrix  $L^*$ . The mapping exists, as was demonstrated in Section III.6

$$P12) \exists \sigma^* A^* \sigma^* = T^*, T^* \text{ triangular, upper or lower.}$$

In the first application of the orthogonal layer  $\sigma_1^*$  (Figure 8.10), the structure of the layer was unimportant. The side-effect of producing a desired terminal behavior made the layer useful. In the present context, the structure of the mapping and the values resulting are important; it is necessary to determine the parameters  $\alpha$ ,  $\beta$  and  $\gamma$  of Figure 8.14b.

These parameters can be determined in terms of  $\sigma_2^*$  and  $U^*$ , but it is easier to work in the transition domain. The parameters could then be determined from the results of Section III.4, or from a comparison of  $L$  and  $L^*$  as derived in Section III.6, where:

$$U^* \sigma_2^* = L^*$$

$$U \cdot \sigma_2 = \begin{bmatrix} I & C^T B \\ 0 & I \end{bmatrix} \cdot \begin{bmatrix} T_1 & T_{12} \\ T_{21} & T_2 \end{bmatrix} = \begin{bmatrix} T_1^{-T} & 0 \\ T_{21} & T_2 \end{bmatrix} = L$$

Therefore, applying an exchange step gives the parameters in the desired format:

$$L^* = \begin{bmatrix} \alpha & 0 \\ \beta & \gamma \end{bmatrix} = \begin{bmatrix} -T_1^T & 0 \\ -T_{21} T_1^T & T_2 \end{bmatrix}$$

The result of applying the orthogonal transformation  $\sigma_2^*$  to Figure 8.12 to obtain Figure 8.14b can be drawn as in Figure 8.15.

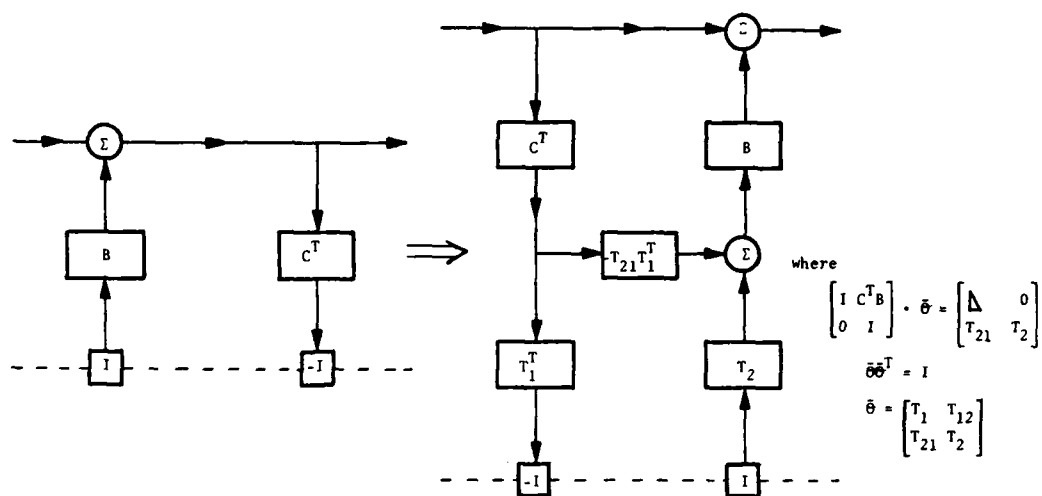


Figure 8.15

We have now characterized  $\sigma_1^*$  and  $\sigma_2^*$  of (8.2), and thus, we have characterized  $\sigma^*$ . We will now examine the application of these transformations to scattering layers which we can then use to "read out" square-root formulae.

#### Application I -- Square Root Doubling

We want to convert the scattering layers given by Friedlander [1976]

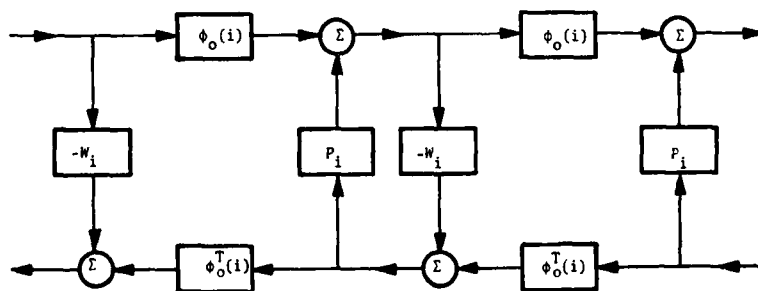


Figure 8.16 Application to Square Root Doubling

into a single layer, which is advanced in time by a factor of two.

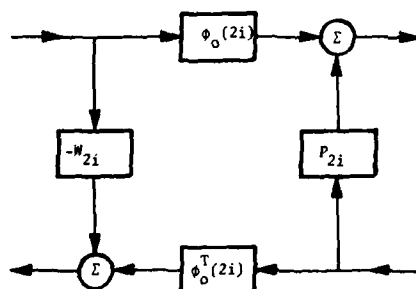


Figure 8.17 Application to Square Root Doubling

We begin with the square root of Figure 8.16

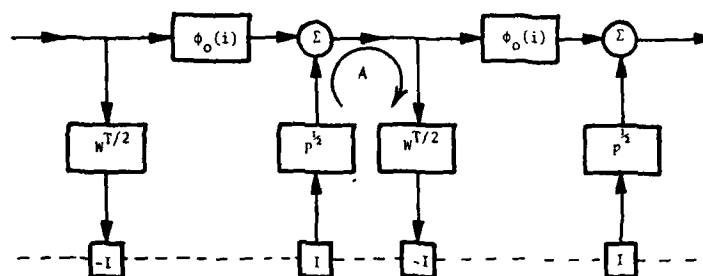


Figure 8.18 Application to Square Root Doubling

proceed first by unwinding the loop at A

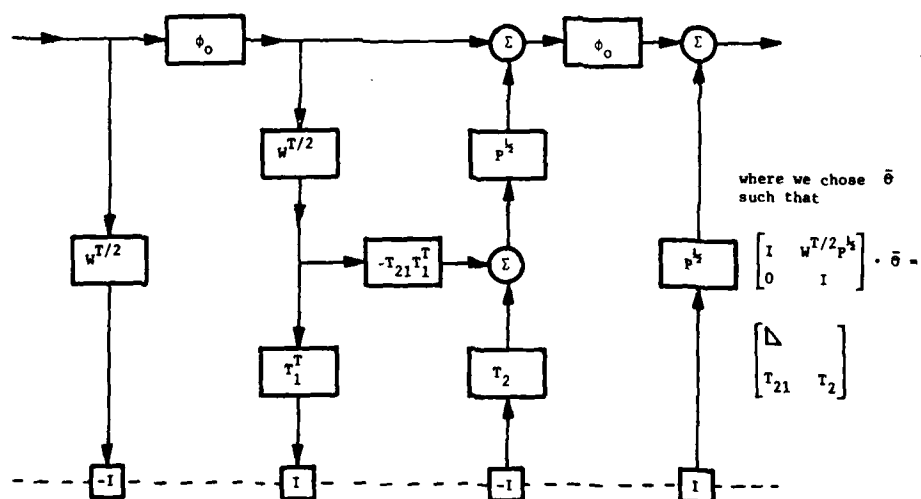


Figure 8.19 Application to Square Root Doubling

and finally collapse the boundary connections using orthogonal transforms

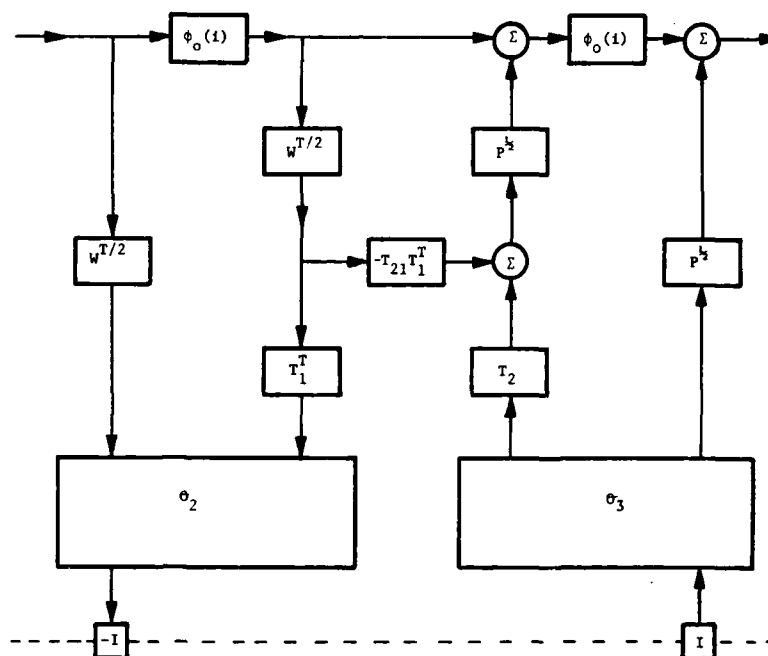


Figure 8.20 Application to Square Root Doubling

We can now read out the requisite equations required to calculate the entries of Figure 8.17. These equations follow directly from our previous rules:

$$W^{T/2}(2i): \sigma_2 \cdot \begin{bmatrix} W^{T/2} \\ T_1^T W^{T/2} \phi_o \end{bmatrix} = \begin{bmatrix} W_{2i}^{T/2} \\ 0 \end{bmatrix} \quad (8.6)$$

$$P^{1/2}(2i): [P^{1/2} \phi_o P^{1/2} T_2] \cdot \sigma_3 = [P_{2i}^{1/2} 0] \quad (8.7)$$

$$\phi(2i): \phi_o(2i) = \phi_o [I - P^{1/2} T_{21} T_1^T W^{T/2}] \phi_o \quad (8.8)$$



Comment:

This approach clarifies several issues. The calculation of  $P$  and  $W$ , which cut across lines of symmetry, is fundamentally different from the calculation of  $\phi_0$ , which lies along a transmission path. The transmission path variables are not subject to calculation via orthogonal transformations, although terms arising from the loop removal at  $A$  are useful.

There are two distinct applications of orthogonal transformations: an explicit application for loop removal, and an implicit application to compress the structure. The latter application will involve transformations both from the left (for an output border junction), and the right (for an input border junction).

#### Application II-- Normal Square Root

We begin with another of Friedlander's [1976] layer ensembles, this one specified for propagating the Riccati equation by one time step. (See Figures 5.7 and 5.8.) We have

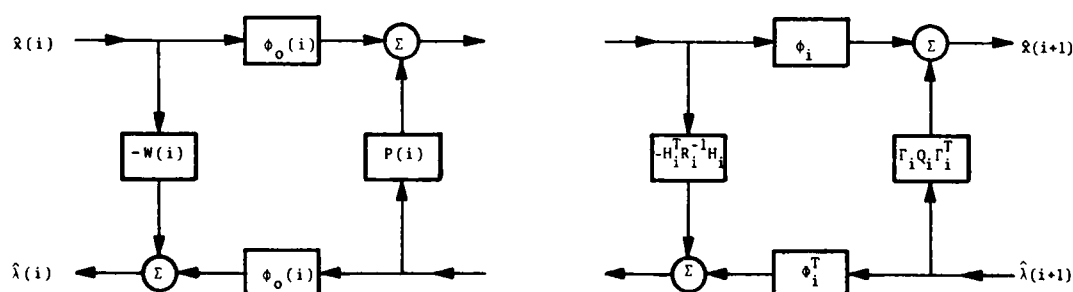


Figure 8.21 Application to Normal Square Root

We take the square root of the network, and apply the appropriate loop removal

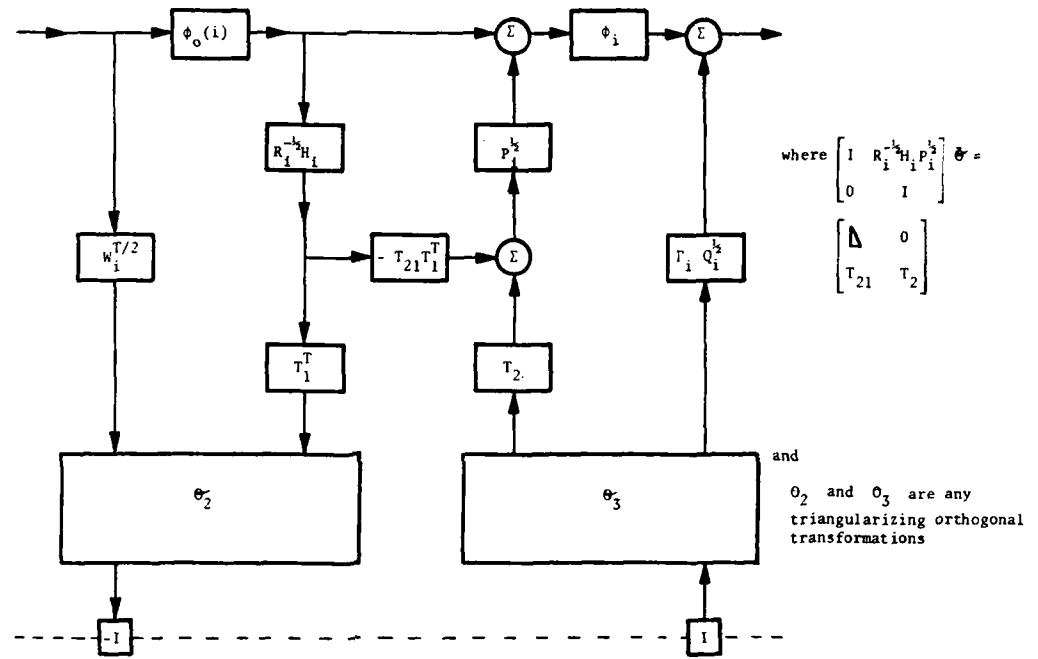


Figure 8.22 Application to Normal Square Root

Again, we can read the equations from Figure 8.22 directly:

$$P^{1/2}(i+1): [\phi_i P_i^{1/2} T_2^T \Gamma_i Q_i^{1/2}] \cdot \sigma_3 \quad (8.9)$$

$$W^{T/2}(i+1): \sigma_2 \cdot \begin{bmatrix} W_i^{T/2} \\ T_1^T R_i^{-1/2} H_i \phi_o(i) \end{bmatrix} \quad (8.10)$$

$$\phi_o(i+1): \phi_i [I - P_i^{1/2} T_{21}^T T_1^T R_i^{-1/2} H_i] \phi_o(i) \quad (8.11)$$

As we found earlier (Section III.5),  $P^{1/2}$  can be computed independently of  $W^{T/2}$  and  $\phi_o$ ; the first equation can be propagated, and the second two ignored.

### III.9 Algorithm Revisions

#### Eigenvector Decomposition

Several problems were noted with eigenvector decomposition:

- 1) The Hamiltonian must have a full set of eigenvectors, which does not always occur when there are repeated closed loop eigenvalues,
- 2) The measurement noise matrix cannot be singular;  $R$  (or, in the controller case,  $B$ ) must be invertible,
- 3) The state transition matrix cannot be singular.

Several extensions have been suggested to solve the first problem. Paul Van Dooren [1978] has suggested using Stewart's [1973] results on repeated eigenvalues. Stewart notes that the subspace spanned by the eigenvectors corresponding to a cluster of eigenvalues of a Hermitian matrix is relatively insensitive to perturbations in the matrix. Using these results, Van Dooren suggested looking for a basis of the subspace directly, without computing eigenvectors (see, for example, Laub [1979]).

The second problem, arising because of a singular measurement covariance matrix, can be solved using the technique of Bryson and Henrikson [1968] and Gevers and Kailath [1973]. A singular matrix implies that after a finite time, several of the states can be determined precisely. We can therefore remove these states from the estimation process, reduce the order of the system, and derive a system with a non-singular  $R$ . In Chapter V we will discuss Gever and Kailath's work which gives conditions for determining when this reduction is possible.

Square Root and Square Root Doubling

These algorithms do not suffer from the repeated eigenvalue problem or the transition matrix singularity problem. They do, however, require an invertible measurement covariance matrix,  $R$ , or control weighting matrix,  $B$ . This can be seen quite clearly from the scattering picture or from the square root equations; both approaches require inverting this quantity. As with eigenvector decomposition, the predictable directions need to be removed from the system.

OVERHEAD <sup>1</sup>		ITERATION		SQUARE ROOT ITERATION		SQUARE ROOT ITERATION (FAST)		SQUARE ROOT DOUBLING	
INITIALIZATION		$n(n+m)(m-1)$ $n(n+m)m$		$0(n)$ $0(n)$ $p+m-2$ $p+m$		$0(n)$ $0(n)$ $p+m-2$ $p+m$		$0(n)$ $0(n)$ $p+m-2$ $p+m$	
ITERATION		$2n^3 + 0(n^2)$ $2n^3 + 0(n^2)$ $2p-1$ $p$		$7/6 n^3 + 0(n^2)$ $7/6 n^3 + 0(n^2)$ $n^2/2 + 0(n)$ $n+p$		$0(n^2)$ $0(n^2)$ $n^2/2 + 0(n)$ $n+p$		$17n^3 + 0(n^2)$ $17n^3 + 0(n^2)$ $n^2/2 + 0(n)$ $3n$	
TERMINATION		$(4p - \frac{1}{2})n^2 + 0(n)$ $4pn^2 + 0(n)$ $4p-2$ $2p$		$n^3/6 + 0(n^2)$ $n^3/6 + 0(n^2)$ $4p-2$ $2p$		$n^3/6 + 0(n^2)$ $n^3/6 + 0(n^2)$ $4p-2$ $2p$		$n^3/6 + 0(n^2)$ $n^3/6 + 0(n^2)$ $4p-2$ $2p$	
ITERATIONS <sup>2</sup>		$\alpha$		$\alpha$		$\alpha$		$\beta = \log_2 \alpha$	
COST <sup>3</sup>		$[10n^3 + 0(n^2)] \cdot \alpha$ $[37.5n^3 + 0(n^2)] \cdot \alpha + 45n^3$		$[35n^3 + 0(n^2)] \cdot \alpha + \frac{5n^3}{6}$ $[50n^3 + 0(n^2)] \cdot \alpha + 48n^3$		$[8n^2 + 0(n)] \cdot \alpha$ $[37n^3 + 0(n^2)] \cdot \alpha$		$[85n^3 + 0(n^2)] \cdot \beta + \frac{5n^3}{6}$ $[85n^3 + 0(n^2)] \cdot \beta + 50n^3$	

## Notes:

- 1 For algorithms as implemented, assuming  $p, m \ll n$
- 2 Relative to ITERATION, valued at  $\alpha$
- 3 Number of cycles required
- 4 With inversion of  $T^{-1}$

## Relative Cost

- + - 2 cycles
- X 3 cycles
- ÷ 5 cycles
- √ 40 cycles

$n$  - number of states  
 $m$  - number of disturbances  
 or inputs  
 $p$  - number of measurements

TABLE 3.1 COMPARATIVE COMPUTATIONAL OVERHEAD

Table 3.2 BASIC ALGORITHM OVERHEAD

OPERATION   Operators	+ -	x	÷	$\sqrt{\quad}$
$n \begin{array}{ c } \hline m \\ \hline \end{array} \cdot \begin{array}{ c } \hline p \\ \hline \end{array}$	$n(m-1)p$	$nmp$		
$n \begin{array}{ c } \hline m \\ \hline \end{array} \cdot \begin{array}{ c } \hline m \\ \hline \end{array}$	$\frac{n}{2} m(m-1)$	$\frac{n}{2} m(m+1)$		
$n \begin{array}{ c } \hline m \\ \hline A \\ \hline \end{array} \cdot \begin{array}{ c } \hline n \\ \hline B \\ \hline \end{array}$ $B = A^T$	$\frac{n}{2} (n+1)(m-1)$	$\frac{n}{2} m(m+1)$		
$n \begin{array}{ c } \hline n \\ \hline \end{array} \cdot \begin{array}{ c } \hline n \\ \hline \end{array}$	$\frac{n}{6} (n+1)(n-1)$	$\frac{n}{6} (n+2)(n+1)$		
$n \begin{array}{ c } \hline m \\ \hline \end{array} + \begin{array}{ c } \hline m \\ \hline \end{array}$	$nm$			
$n \begin{array}{ c } \hline n \\ \hline \end{array} + \begin{array}{ c } \hline n \\ \hline \end{array}$ symmetric	$\frac{n}{2} (n+1)$			
$n \begin{array}{ c } \hline n \\ \hline \end{array} + \begin{array}{ c } \hline n \\ \hline \end{array}$	$\frac{n}{2} (n+1)$			
$n \begin{array}{ c } \hline n \\ \hline \end{array}^{-1}$ (Gaussian Elimination)	$n(n-1)^2$	$n^3 - 1$	$n$	
$n \begin{array}{ c } \hline n \\ \hline \end{array}^{-1}$ (Triangular Decomposition) symmetric	$\frac{n^2}{2} (n-1)$	$\frac{n}{2} (n^2 + 3n - 2)$	$2n-1$	$n$
$n \begin{array}{ c } \hline n \\ \hline \end{array}^{-1}$	$\frac{n}{6} (n-2)(n-1)$	$\frac{n}{6} (n+4)(n-1)$	$n$	
$m \begin{array}{ c } \hline n \\ \hline \end{array} \Rightarrow \begin{array}{ c } \hline 0 \\ \hline \end{array}$ $m \leq n$	$-\frac{m}{3} (m^2 - 3nm + 2)$	$-\frac{m}{6} (2m^2 - 3(2n+1)m + 1)$	$\frac{m}{2} (m-1)$	$m$
$n \begin{array}{ c c c } \hline n & n & n \\ \hline \end{array} \Rightarrow \begin{array}{ c c c } \hline 0 & 0 & 0 \\ \hline \end{array}$	$\frac{n}{3} (19n^2 + 4)$	$\frac{n}{3} (19n^2 + 6n + 5)$	$n(2n-1)$	$2n$
$n \begin{array}{ c } \hline n \\ \hline \end{array}^{\frac{1}{2}}$ (Cholesky) symmetric, $> 0$	$\frac{n}{6} (n+1)(n-1)$	$\frac{n}{6} (n+4)(n-1)$	$n-1$	$n$

## Chapter IV

## THE STEADY-STATE SOLUTION TO THE CONTINUOUS RICCATI EQUATION

In the previous chapter we examined several diverse approaches to solving the discrete-time Riccati equation; in this chapter we will consider the continuous equivalent. The focus of this dissertation is on discrete solutions. This chapter will therefore be brief, principally discussing iterative solutions to the continuous problem.

We again consider two applications: estimation and control. In exactly analogous fashion with the discrete-time case, the solution of these two problems requires the solution of one of two related differential equations. The continuous problem begins with the definition of the equivalent state-space model:

$$\begin{aligned}\dot{x}(t) &= F(t)x(t) + G(t)u(t) \\ y(t) &= H(t)x(t) + v(t)\end{aligned}\tag{1.1}$$

For the estimation problem, we assume  $u(t)$  and  $v(t)$  are white noise disturbances

$$\mathbb{E}\begin{bmatrix} u(t) \\ v(t) \end{bmatrix} \begin{bmatrix} u^T(s) & v^T(s) \end{bmatrix} = \begin{bmatrix} Q & 0 \\ 0 & R \end{bmatrix} \delta(t-s)\tag{1.2}$$

Then,  $\hat{x}(t)$ , the linear least squares estimate of  $x(t)$  given observations up to time  $t$ , can be obtained via the Kalman filter equation:

$$\dot{\hat{x}}(t) = [F(t) - P(t)H^T(t)]\hat{x}(t) + P(t)H^T(t)y(t), \quad \hat{x}(0) = 0\tag{1.3}$$

where  $P(t)$  is the covariance of the state error estimate

$$\begin{aligned}\dot{P}(t) &= F(t)P(t) + P(t)F^T(t) + G(t)Q(t)G^T(t) \\ &\quad - P(t)H^T(t)R^{-1}(t)H(t)P(t) \\ &= E\{(\hat{x}(t) - x(t))(\hat{x}(t) - x(t))^T\}\end{aligned}\tag{1.4}$$

$$P(0) = \Pi_0$$

This is the estimation Riccati equation.

The control problem begins with the same state space model, except that perfect knowledge of the states,  $x$ , are assumed (the output,  $y$ , is ignored) and the input,  $u$ , is assumed deterministic and available for control.

The problem solution begins by defining a performance index,  $J$ , to be minimized:

$$J = \int_{t_0}^{t_f} [x^T(t)A(t)x(t) + u(t)B(t)u(t)]dt\tag{1.5}$$

where  $A(t)$  is non-negative definite and  $B(t)$  is positive definite. The optimal solution, in the sense of a control input,  $u(t)$ , which minimizes  $J$ , is given by

$$u(t) = -B(t)G^T(t)S(t)\tag{1.6}$$

where  $S(t)$  is the solution of the second Riccati equation:

$$\dot{S}(t) = S(t)F(t) + F^T(t)S(t) + A(t) - S(t)G(t)B^{-1}(t)G^T(t)S(t)\tag{1.7}$$



Whereas in the discrete case an exact solution to the matrix Riccati equation was readily computable, in the continuous case this is often more difficult. For the continuous solution, it is often insufficient to rely upon matrix fundamentals and linear least squares theory; a detailed understanding of numeric integration is required. This takes us beyond the purview of this thesis; perforce, we shall focus on those issues directly related to the presentation of the previous chapter.

However, we will be interested in steady state solutions which means we are interested in solutions where the derivatives  $\dot{P}$  and  $\dot{S}$  are identically zero. This constraint converts differential equations into algebraic matrix equations. Therefore, in the sequel, when an algorithm requires integration, we should suspect that excessive complication is being introduced.

This chapter begins by briefly noting the similarities between the discrete and the continuous eigenvector decomposition routines. Treatment of continuous square roots is deferred to the paper by Morf, Lévy, and Kailath [1978], and is not discussed. Square root doubling is treated in terms of sundry proposed solutions, with emphasis given to their strengths and weaknesses. One approach, the bilinear transform, is discussed more deeply.

#### IV.2 Eigenvector Decomposition

As in the discrete case, eigenvector decomposition can be used to solve for the steady-state error covariance. The solution was described by Potter [1966], and later implemented by Bryson and Hall [1971] using the QR algorithm.

Unlike the discrete-time algorithm, the continuous version of the algorithm does not require the inversion of the dynamics matrix. Repeated closed loop eigenvalues, implying possibly non-existent eigenvectors, and singular measurement covariance matrices still cause problems. An extension of the algorithm is required, as outlined in Section Nine of the previous Chapter.

#### IV.3 Square Root Doubling

Implementing doubling in the continuous domain is conceptually identical to the discrete problem. The difference is found in the determination of the initial interval. In the discrete case, we begin with solutions for one time step, assuming zero initial conditions -- a straightforward procedure. In the continuous case, we again need to solve for that initial interval. In the latter case, the obvious approach proves more difficult.

A typical solution to this problem, as outlined by Bierman and Sidhu [1976], and reviewed by B.D.O. Anderson [1978] involves a two-stage procedure: the first stage requires translating the continuous problem into an analogous discrete-time problem, and the second stage is the solution of this analogous problem. One approach would be to explicitly solve the Riccati equation over the initial interval. The question arises as to how short to make the interval; as Anderson notes, if the interval is too short, the recursive part of the algorithm may not converge properly. There is also the added burden of computing the transition matrix for the interval, independent of whether integration or power series approximation is used for the computation.

As an alternative, which avoids the computational burden, Anderson suggests an algorithm by Roberts [1971] and Denman [1976]. They begin with the continuous Hamiltonian

$$H = \begin{bmatrix} -F & GR^{-1}G^T \\ Q & F^T \end{bmatrix}$$

which they iterate by computing

$$H_{i+1} = \frac{1}{2}(H_i + H_i^{-1}) \quad .$$

This leads both to the steady-state error covariance and to a doubling algorithm. This approach was not very promising because the doubling algorithms we derived required the inversion of the dynamics matrix,  $F$ ; in the continuous case, singular  $F$  matrices are quite common. Possibly by using pseudo-inverses, a more complex square-root expansion, or a scattering-domain derivation, this problem could be circumvented.

Another approach is to recognize that a bilinear mapping exists which yields a discrete equivalent to the continuous Riccati equation; it is equivalent in the sense that both equations have the same steady state solution.

Hitz and Anderson [1972] present a mapping from the continuous to the discrete-time domain. They prove that the steady-state solutions to both the differential and the difference equation are identical, and that as long as a positive definite solution exists and is unique, then the discrete-time equivalent converges for any nonnegative definite initial condition.

Noting that the bilinear transform,  $S \leq \frac{z-\alpha}{z+\alpha}$ , considered with the Laplace transform and the inverse  $z$  transformation, establishes an explicit isomorphism between the spaces of square integrable functions and square summable sequences, Hitz et al develops the mapping for the control form of the Riccati equation. We will briefly present the estimator equivalent.

Given an algebraic matrix equation for the steady-state estimator error covariance of the form

$$FP + PF^T + GQG^T - PH^T R^{-1} HP = 0 \quad (3.1)$$

we want to transform this equation into the discrete equivalent

$$\Phi M \Phi^T + \Gamma Q_D \Gamma^T - \Phi M H_D^T (R_D + H_D M H_D^T)^{-1} H_D M \Phi^T = M \quad (3.2)$$

where  $M$  and  $P$  are exactly equal. The coefficients of these two equations can be related by either full or square root formulas:

<u>Full</u>	<u>Square Root</u>
$H_D = \sqrt{2} \alpha H(\alpha I - F)^{-1}$	$H_D = \sqrt{2} \alpha H(\alpha I - F)^{-1} \quad (3.3)$
$Q_1 = 2\alpha(\alpha I - F)^{-1} G Q G^T (\alpha I - F)^{-T}$	$Q_1^{\frac{1}{2}} = \sqrt{2\alpha} (\alpha I - F)^{-1} G Q^{\frac{1}{2}}$
$B = \frac{\sqrt{2}}{2} H Q_1$	$\tilde{B} = \frac{\sqrt{2}}{2} H Q^{\frac{1}{2}}$
$R_D = \alpha R + \frac{1}{2} H Q_1 H^T$	$R_D^{\frac{1}{2}} = [\alpha R^{\frac{1}{2}} \mid \tilde{B}]$
$\Gamma Q_D \Gamma^T = Q_1 - B^T R_D^{-1} B$	$\Gamma Q_D^{\frac{1}{2}} = Q^{\frac{1}{2}} [I \mid j \tilde{B}^T R_D^{-T/2}]$
$\phi = (\alpha I + F)(\alpha I - F)^{-1} - B^T R_D^{-1} H_D$	$\phi = (\alpha I + F)(\alpha I - F)^{-1} - Q^{\frac{1}{2}} \tilde{B}^T R_D^{-T/2} R_D^{-1/2} H_D$

where  $\alpha$  is real, greater than zero, and not equal to any eigenvalue of  $F$ .

The proof that equations (3.3) do in fact transform (3.1) into (3.2) follow from a direct substitution and simplification. For details of such a proof, see Hitz [1972]. They also show that any solution for an equation similar to (3.1) is also a solution for an equation similar to equation (3.2).

We chose to implement this version of the doubling because it could easily and quickly be incorporated into existing discrete doubling algorithms. The overall algorithm is still a two step procedure, but the initial step is relatively fast compared to the doubling stage. There are also fewer constraints on the class of problems which can be solved. " $\alpha$ " can always be chosen so that  $(\alpha I - F)$  is invertible. We may have problems if  $R_D$  is not invertible, but in the bilinear algorithm  $R_D$  is comprised of a combination of all noise covariances. It will usually be invertible.

The three main problems with the bilinear mapping are other aspects of accuracy, speed, and convergence. We have introduced another level of computation, including an inversion; in some cases this will exacerbate ill-conditioning. For speed, we already noted the handicap of added overhead. Rate of convergence is also determined by the choice of  $\alpha$ . These factors will be explored in Chapter VI.

Finally, we note that intermediate values in this scheme have no significance; only the final value has a continuous analog. (In the purely discrete case, the error covariance at the " $i$ th" iteration corresponded to  $P(2^i)$ .) In some contexts, this abstraction might be a disadvantage compared to doubling techniques that iterated continuous-time variables.

#### IV.4 Conclusions

For solving the continuous, steady-state Riccati equation, eigenvector decomposition is the favored approach. It is numerically well behaved, fast, and it does not suffer from some of the limitations that make eigenvector techniques the secondary choice for the discrete problem; repeated eigenvalues and singular  $R$  matrix can still be a problem, however.

The square-root doubling algorithms were all limited by one of three weaknesses. The first set of algorithms required an initial solution to the Riccati differential equation over a fixed interval. The second set of algorithms was limited in the classes of practical problems they could solve. The third set of algorithms was subject to burdensome computational overhead, incurred while converting a continuous problem into an equivalent discrete-time problem; the result was to sacrifice speed and accuracy. The importance of this overhead will be considered later.

In the next chapter we will consider the class of problems which differentiate among these algorithms, focusing on discrete-time systems. In Chapter VI, we return to these algorithms, presenting an empirical comparison.

## Chapter V

### DIFFERENTIATING APPLICATIONS

Most algorithms handle most practical problems most of the time. For this set of problems, important questions involve criteria such as speed of computation, storage space requirements, and computational precision required. These issues are examined in the next chapter.

Another class of problems have salient characteristics which preclude the use of certain algorithms. In this chapter we will focus on two such classes: problems having singular state transition matrices, and problems having repeated closed-loop eigenvalues.

An interesting and important class of problems arises when we consider modelling discrete systems with pure delays. These delays arise for a variety of reasons-- sampling delays, computation overhead, transport lag-- and often lead to models with singular dynamics matrices. As we mentioned in Chapter III, singular dynamics matrices exacerbate weaknesses in several algorithms, so we shall examine this issue in detail.

The singular transition matrix collection of examples is related to the problem of designing reduced-order observers because of a conjecture that there always exists a reduced order system equivalent to the original singular system [Katz, 1972]. We therefore begin by considering Bryson and Henrikson's work [Bryson, 1968] on reduced order estimators. We then consider Gevers' extension [Gevers, 1972]. After examining Brammer [Brammer, 1968], and Tse and Athan's [Tse, 1970] work on singular systems, we consider a practical example, and then evaluate Katz's conjecture and

Powell and Parson's [Powell, 1978] result in light of our results.

In the second section we consider the case of repeated closed-loop eigenvalue problems as they arise in satellite design and perfect measurement examples.



## V.2 Reduced-Order Observers

### The Discrete Case

System order, ' $n$ ', is a significant criterion in both filter design and filter implementation. Computation time is typically proportional to  $n^3$ , at best (except for the fast square root algorithms); data storage requirements are often proportional to  $n^2$ . Order reduction, when cost effective and while preserving other performance properties, is therefore desirable.

In continuous time systems, if the output of the system,  $y(t)$ , is sufficiently smooth, then a uniquely identifiable number,  $\alpha$ , of different projections of the state are calculable without error as some linear combination of the output and its first  $\alpha-1$  derivatives. This achieves a maximal reduction in the order of the Kalman filter Riccati equation [Bryson, 1965] [Geesey, 1973] [Gevers, 1972].

Bryson, et al., noted similar computation-reducing effects from differencing observations in the discrete-time case [Bryson, 1968]. Bucy, Rappaport, and Silverman, noted that in certain discrete-time cases, differencing did not have the same computation-reducing consequences noted in the analogous continuous time case [Bucy, 1970], [Rappaport, 1970]. Gevers and Kailath went on to elucidate and explain this difference in terms of properties of the associated covariance matrices [Gevers, 1972].

Two differences between continuous and discrete systems lead to these disparate results. First, that differencing operations introduce a time-delay between  $y(i)$  and  $y(i-1)$ , whereas the signal  $y(t)$  and its derivative  $\dot{y}(t)$  have the same time argument. Secondly, any transfer function of the form

$$H(z) = z^{-k} H_1(z), \quad 1 \leq k \leq \alpha-1, \quad \alpha > 2$$

produces the same power spectral density and can be realized by a causal system. In contrast, in the continuous case, the difference between the degree of the numerator and denominator is unique for all rational factorizations.

Gevers went on to consider a special form of the discrete-time, one-step prediction filter

$$\begin{aligned} x(i+1) &= \phi x(i) + \Gamma w(i+1) & \mathbb{E} [w(i)w(j)^T] &= Q \delta(i-j) \\ y(i) &= H x(i) & \Pi(i) &= \mathbb{E} (x(i)x(i)^T) \end{aligned}$$

This model often arises in systems that have correlated noise terms in their output [Bryson, 1968]; the measurement noise term is eliminated by state-augmentation.

As Bryson points out, the associated Riccati equation may then be ill-conditioned, because its dynamic rank is lower than its order. This can be alleviated by proper partitioning of the augmented state equations, but is a potential hazard for blind application with all strategies, including the square root schemes.

The direct approach to the conditioning problem, however, is to reduce the order of the Riccati equation. Gevers showed that this could be done maximally only when

$$\begin{aligned} H(i-k)\phi(i-k,i)\Gamma(i) &= 0 & k &= 1, \dots, \alpha-1 \\ H(i-k)\phi(i-k,i)x(i) &= y(i-k) & k &= 0, \dots, \alpha-1 \\ H(i-k)\phi(i-k,i)\Pi(i) &= N^T(i-k)\phi^T(i,i-k) & k &= 0, \dots, \alpha-1 \\ & & k &\leq i \leq N \end{aligned}$$

where

$$\Pi(i+1) = \phi(i+1,i)\Pi(i)\phi^T(i+1,i) + \Gamma(i+1)Q\Gamma^T(i+1), \quad N(i) = \Pi(i)H^T(i)$$

Here,  $\alpha$  is the difference between the order of the numerator and denominator polynomials of the spectral density function:

$$S_y(z) = H(z)H(z^{-1}) = \frac{b_\alpha z^{n-\alpha} + b_{\alpha+1} z^{n-\alpha-1} + \dots + b_{2n-\alpha} z^{-n+\alpha}}{z^n + a_1 z^{n-1} + \dots + a_{2n} z^{-n}}$$

For the time invariant, steady-state case, we have

$$H \phi^{i-k} \Gamma = 0, \quad k = 1, \dots, \alpha-1$$

$$H \phi^{i-k} x(i) = y(i-k), \quad k = 0, \dots, \alpha-1$$

$$H \phi^{i-k} \Pi = H \Pi^T [\phi^{k-i}]^T, \quad k = 0, \dots, \alpha-1$$

$$\Pi = \Phi \Pi \phi^T + \Gamma Q \Gamma^T$$

It is therefore necessary to identify the definite relative order of the system,  $\alpha$ , determine if the above constraints are matched, and if not, transform the model to a suitably constrained model. Gevers demonstrated this could be done for the single-input, single-output case.

Another important result, derived for the case of no measurement noise and single output, shows that for singular  $\phi$  with completely observable states, the system has no predictable directions. The system order can, however, be reduced by one; for the system with white measurement noise, this reduces the system to the normal, unaugmented realization (that is still singular).

## Design with Delays

### Singular $\phi$ Matrix

A system with a pure delay -- where the output of at least one state is not fed back into that state-- modelled in state space format, yields a system with a singular  $\phi$  matrix. Assume  $\{\phi, H\}$  is observable. Then, as we saw from Gevers' results, we would not expect to be able to reduce the order of the estimator by taking differences of the system outputs.

An interesting alternative structure for the filter arises if we consider a special case. If each state suffers the same pure delay before output, then the  $\phi$  matrix of the filter can be simplified by removing states corresponding to pure delays (see Figure 5.1).

### Prediction

The best estimate of a state  $x$  at time  $i+m$ ,  $\hat{x}(i+m)$ , given data up to time  $i$  is simply (see, for example, Sage [1971])

$$\hat{x}(i+m|i) = \phi^m \hat{x}(i|i)$$

where  $\hat{x}(i)$  is the filtered estimate of  $x(i)$ . Further, the state error covariance is

$$\begin{aligned} P(i+m|i) &= E[\tilde{x}(i+m|i)\tilde{x}(i+m|i)^T] \\ &= (\phi^m)^T P(i|i) \phi^m + \sum_{i=1}^m (\phi^{m-i})^T \Gamma Q \Gamma^T (\phi^{m-i}) \end{aligned}$$

where

$$\tilde{x} \triangleq (x - \hat{x})$$

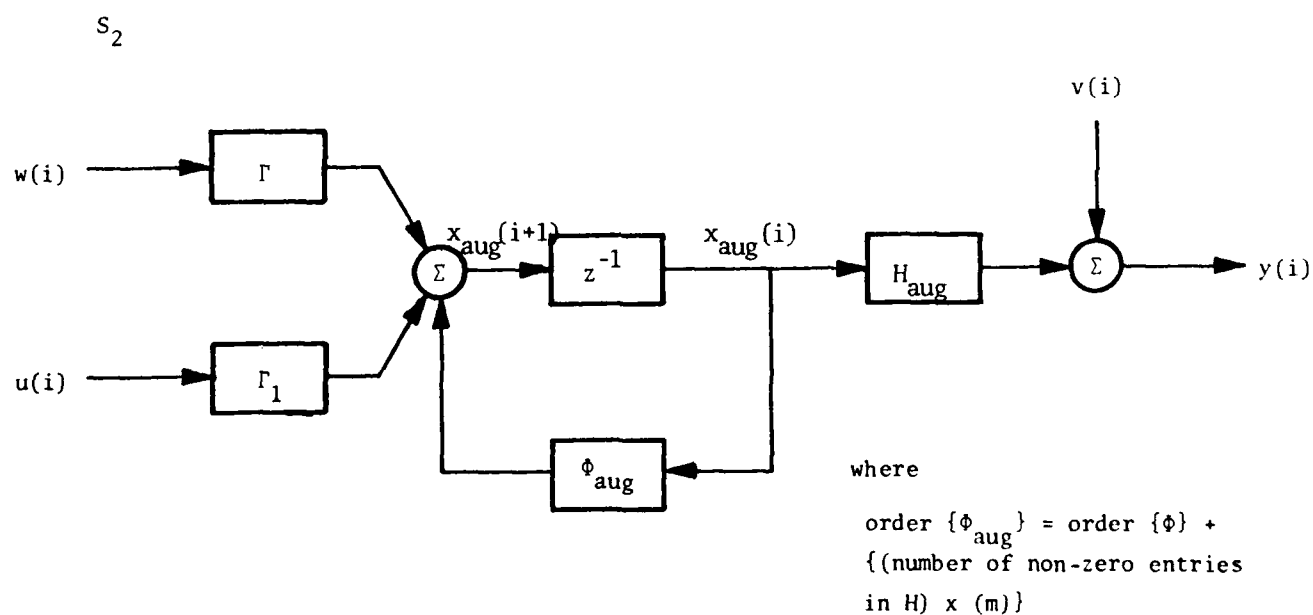
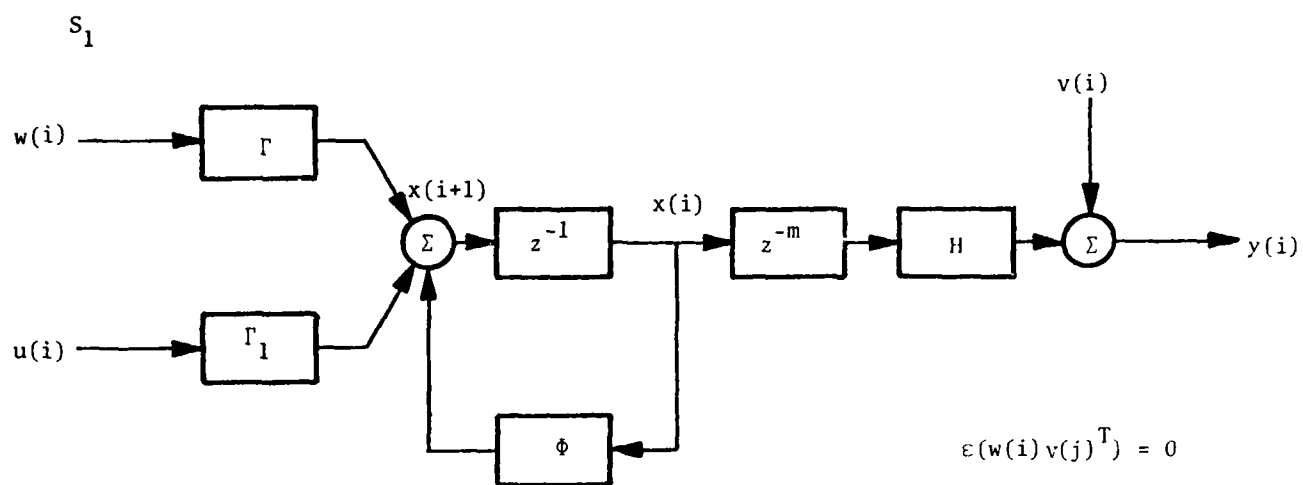


Figure 5.1

ALTERNATIVE REALIZATIONS FOR SYSTEM WITH PURE DELAY ON OUTPUT.

The estimator for Figure 5.1 can therefore be constructed without the  $m$ -order delay incorporated in the state transition matrix. Outside the feedback loop, the states can be predicted from the past back to the present, for the estimator will now be running  $m$  cycles behind the modelled system (see Figure 5.2).

This reorganization may provide a significant computational savings, especially if delayed estimates are satisfactory. It may, however, introduce an increase in complexity, especially if state estimation will be used for closed-loop control. The additional complexity arises because now delayed versions of the predicted states must be fed back into the system (see Figure 5.3).

If all states must be predicted, and later delayed, the augmented approach may be more computationally efficient. In most cases, however, the computational tradeoff is between increased multiplications (for the augmented realization) versus storage (for the predicted realization), with storage being comparatively inexpensive.

These issues only arise because we have imposed a structure on our estimator. The output,  $\hat{x}$ , is identical in both cases. The computations, in contrast, are performed in different coordinate systems. One realization can be derived from the other, and the minimal realization (in operation count) may be neither of those proposed.

#### An Example

As an example, we consider the problem detailed in Appendix C2. As presented, this is a second order system with a pure, second-order delay:

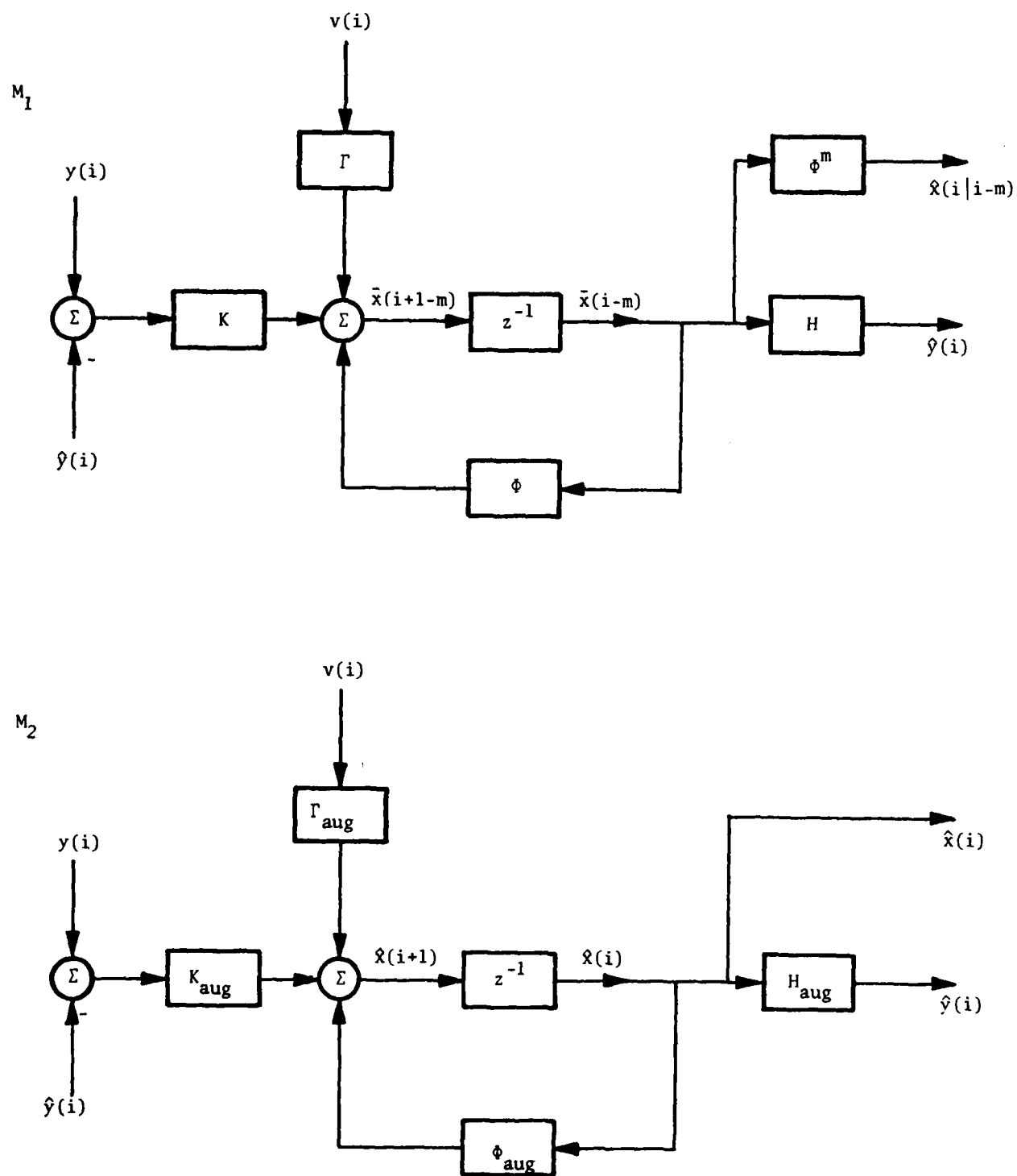


Figure 5.2

ALTERNATIVE ESTIMATORS FOR SYSTEM WITH PURE DELAY ON OUTPUT.

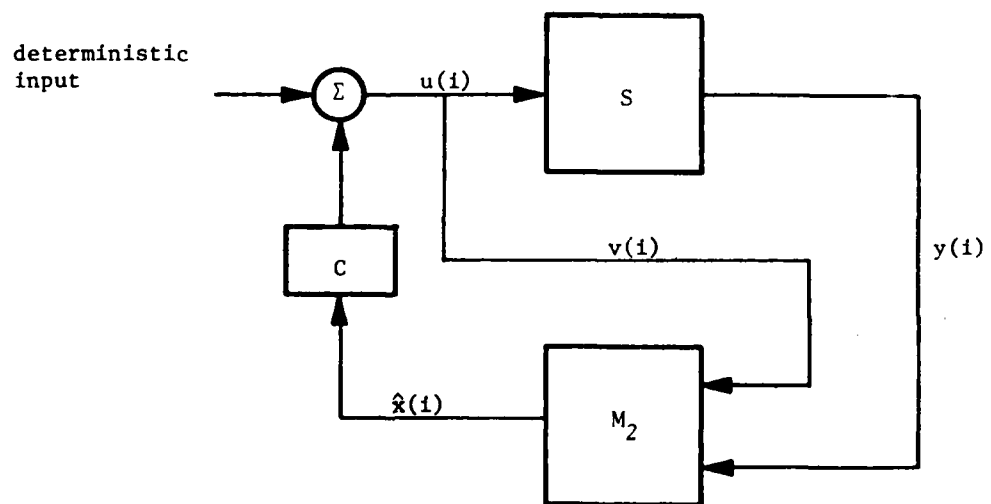
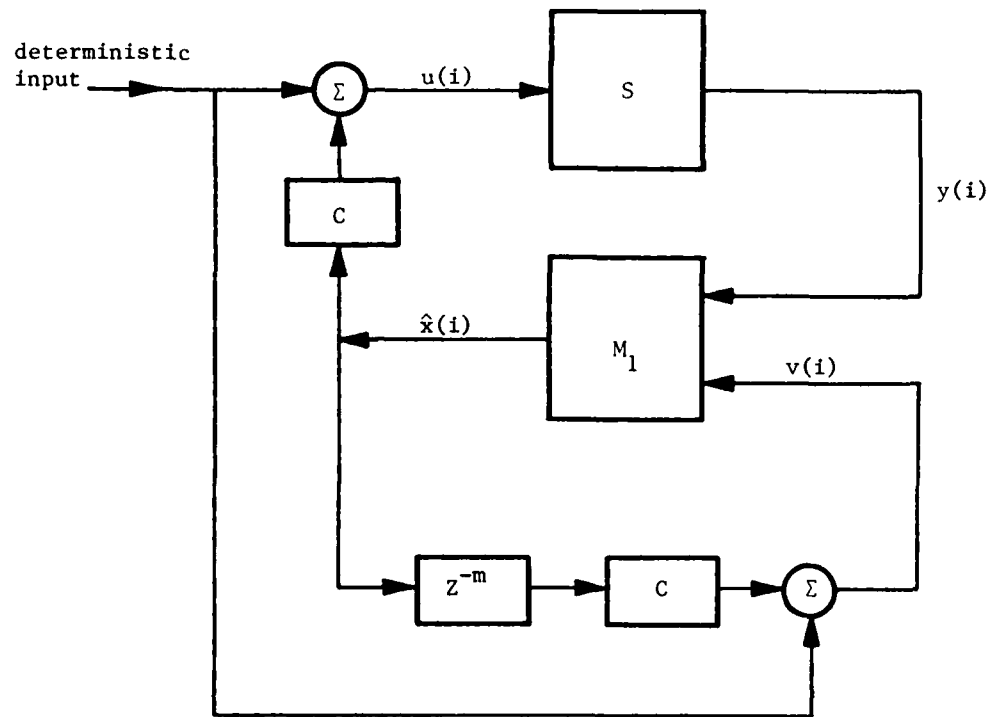


Figure 5.3

ALTERNATIVE CLOSED-LOOP REALIZATIONS FOR SYSTEM WITH PURE DELAY



$$x(i+1) = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} x(i) + \begin{bmatrix} T^2/2 \\ T \end{bmatrix} w(i)$$

$$y(i) = [1 \quad 0]x(i-2) + v(i)$$

$$\epsilon(w(i)w(j)^T) = q \delta(i-j)$$

$$\epsilon(v(i)v(j)^T) = r \delta(i-j)$$

$$\epsilon(w(i)v(j)^T) = 0$$

(For numerical results,  $r$  is assumed to be unity and  $T$  to be 1 Hertz.)

We begin by considering the augmented system incorporating the delay:

$$\Phi = \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad \Gamma = \begin{bmatrix} T^2/2 \\ T \\ 0 \\ 0 \end{bmatrix} \quad H = [0 \quad 0 \quad 0 \quad 1]$$

This problem is in the standard state-space format; in addition,  $\Phi$  is singular. As detailed in the last section, we can approach this problem in two ways.

Working with the full-order augmented system, we cannot solve the Riccati equation using the proposed eigenvector decomposition algorithm. Square Root Doubling, in contrast, is perfectly satisfactory.

Working with the segmented system-- a second order filter and a second order predictor-- we can use any of the algorithms presented in Chapter III to solve the Riccati equation.

The root locus of estimator roots plotted as a function of disturbance covariance appears in Figure 5.4. As expected, the loci for both designs coincide, except for the presence of an additional pair of poles corresponding to the two delays in the augmented system.

## STAR-TRACKING TELESCOPE OPTIMAL ESTIMATOR

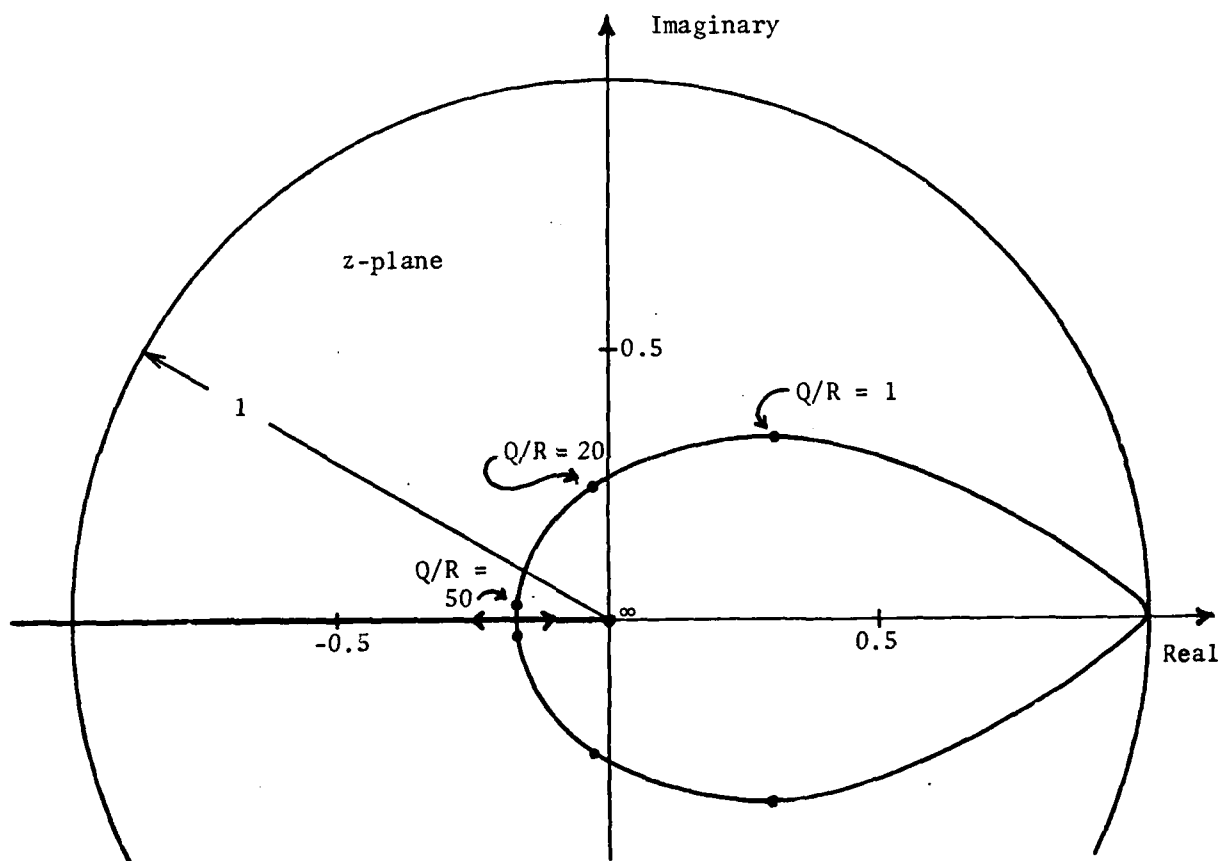
Eigenvalues vs.  $\frac{\text{Process noise}}{\text{Measurement noise}}$ Assumes  $m$  cycles of delay,  $m \geq 0$ 

Figure 5.4 LOCUS OF ESTIMATOR ROOTS VERSUS DISTURBANCE COVARIANCE

In Figure 5.5 we present a plot of the error covariance of the first state (the measured state) as a function of the disturbance covariance. Again, the two plots coincide, as do the other covariance terms (Figure 5.6).

Also presented in Figure 5.5 is the first-state error covariance assuming the delay in the estimator is ignored ( $m \approx 0$ ). In addition, we have plotted the covariance assuming  $m=0$  in both the system and the estimator. We will comment on these results after examining another approach to solving this problem.

#### Comments on Previous Work

Katz, in his treatment of the eigenvector decomposition problem with singular  $\phi$  matrix [Katz, 1972] conjectured, in effect, that an equivalent reduced-order estimator with non-singular transition matrix could always be found. An estimator for this non-singular equivalent could then be developed by direct application of the eigenvector decomposition algorithm. This conjecture, however, was not formulated as an algorithm; it was simply stated as an existence hypothesis. No counter examples have yet been found, and without a mathematical formulation, the conjecture cannot be proved correct.

We consider an example proffered by Powell and Parsons [Powell, 1978]. The example is again the star-tracker estimator presented in Appendix C2. using linear transformations, Powell, et al, reduced the fourth-order system to a system of second order. Their results, compared to those derived above, are presented in Figures 5.7 and 5.8. We note that for the case where measurement noise dominates process disturbances (small  $Q/R$  ratio), the design is insensitive to the

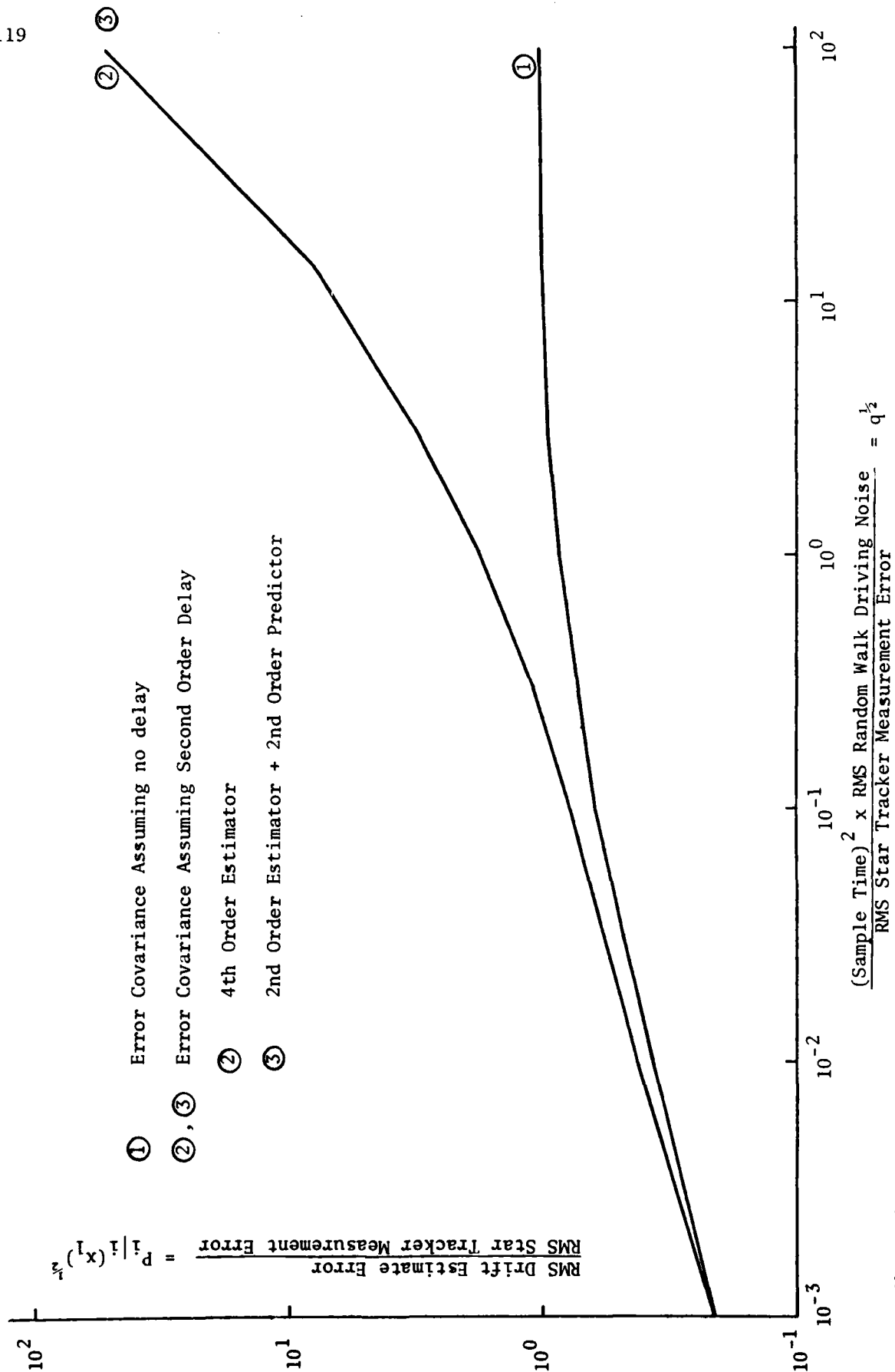


Figure 5.5 ESTIMATOR ERROR COVARIANCE

Figure 5.6

Comparison of covariance terms for augmented  
versus predicted systems

Augmented System:

$$P_{aug} = E\{\tilde{x}(i|i-1)\tilde{x}(i|i-1)^T\} = \begin{bmatrix} 1.337439e+02 & 5.914705e+01 & 7.709689e+01 & 3.044984e+01 \\ 5.914705e+01 & 3.439391e+01 & 2.975314e+01 & 1.035923e+01 \\ 7.709689e+01 & 2.975314e+01 & 4.734375e+01 & 2.009060e+01 \\ 3.044984e+01 & 1.035923e+01 & 2.009060e+01 & 9.731371e+00 \end{bmatrix}$$

Second Order System

$$P_2 = \begin{bmatrix} 4.597871e+01 & 6.854102e+01 \\ 6.854102e+01 & 1.170820e+02 \end{bmatrix}$$

Predicted Second Order System

$$P_{pred} = \begin{bmatrix} 133.34 & 58.945 \\ 58.945 & 34.2929 \end{bmatrix}$$

$$\text{Root Loci vs } \frac{T^2 \eta_{\text{RMS}}}{V_{\text{RMS}}} = \frac{(\text{sample time})^2 \times \text{RMS Random Walk Drift Rate Noise}}{\text{RMS Star Tracker Measurement Noise}}$$

For Drift Estimator

- ① Optimal Estimator Poles for no delay in measurement
- ② Optimal Estimator Poles for two sample interval delay in measurement
- Estimator Poles when gains designed assuming no delays are used with delayed measurement

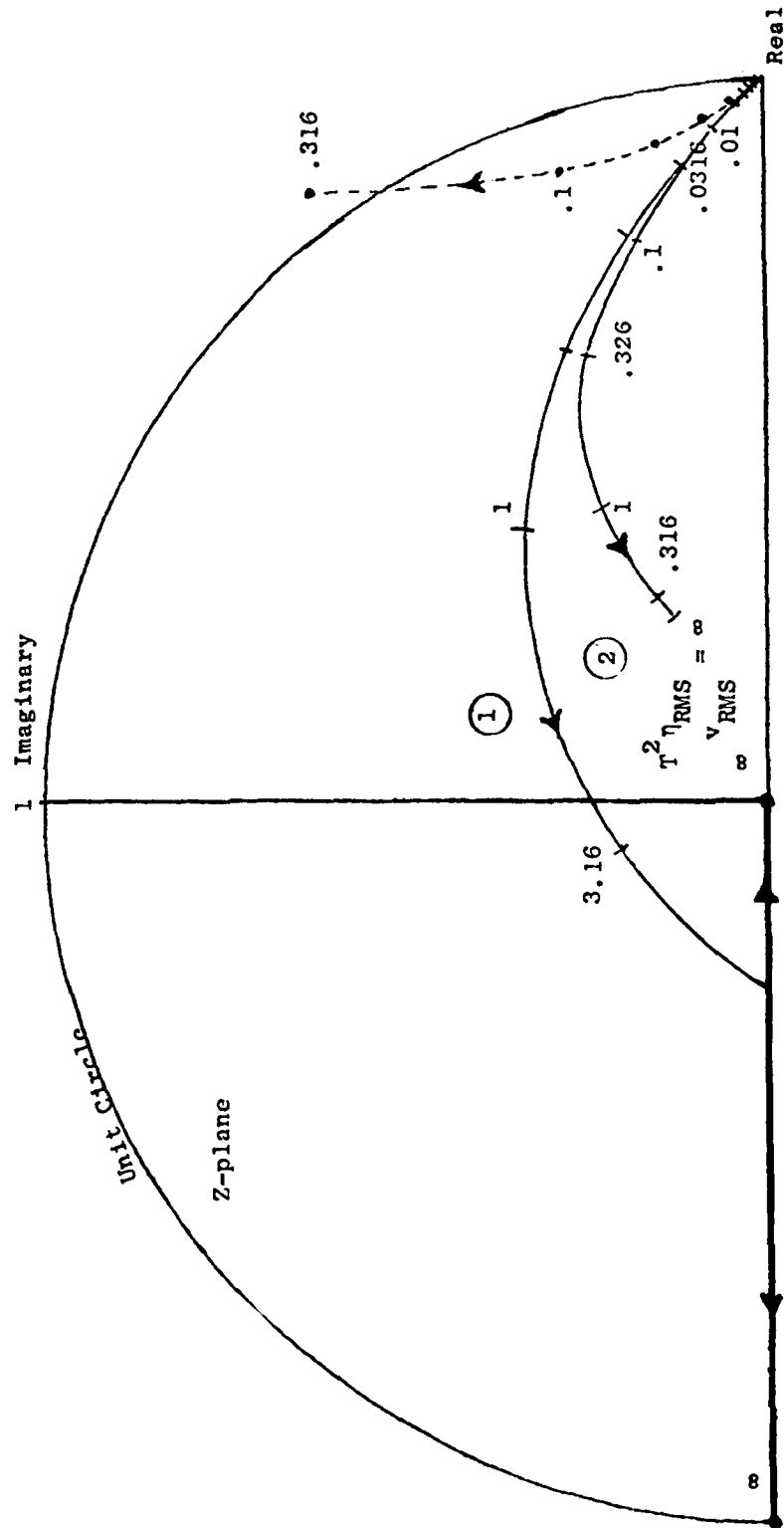


Figure 5.7 LOCUS OF ESTIMATOR ROOTS VERSUS DISTURBANCE COVARIANCE  
(from Powell [1978])

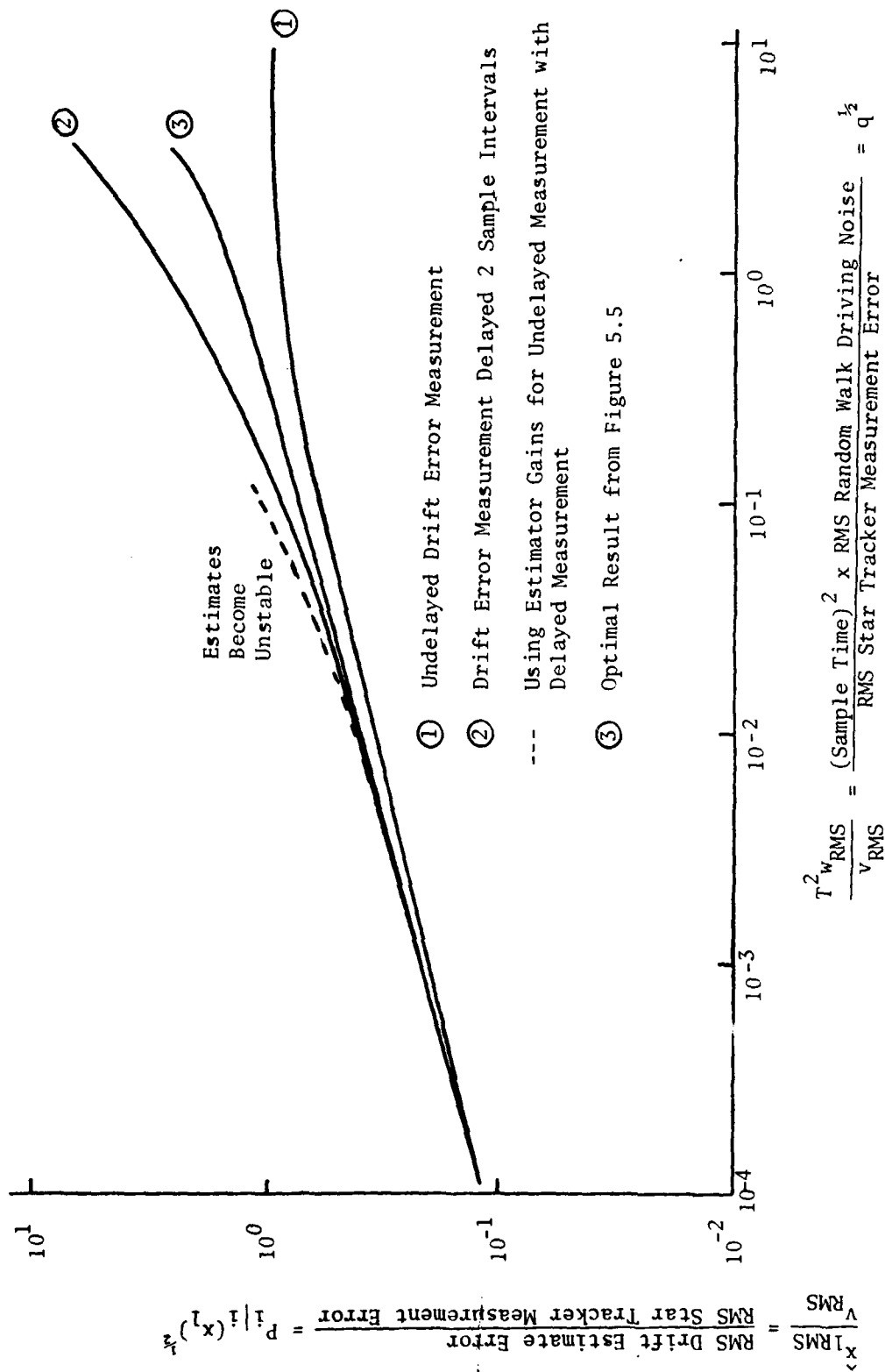


Figure 5.8 ESTIMATOR ERROR COVARIANCE, from Powell [1978].

estimator pole placements considered; all solutions have comparable performance, measured in terms of first-state error covariance.

We would suspect from our earlier work that this design only approximates the best linear estimator for this case. This question is examined in Appendix B1, where we show that the proffered "equivalent" second-order system has sequentially correlated noise, both disturbance and measurement. For small  $Q/R$  ratios, this correlation can be neglected, as shown.

#### Engineering Judgment

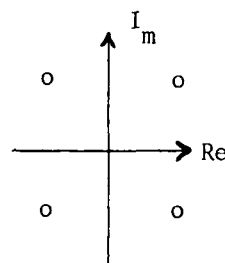
It is clearly important to identify whether the problem under consideration is sensitive to variations in the parameters, as is the case in Figure 5.8 when  $Q/R > 1$ , or insensitive and computationally robust. In the latter case, the increased complexity often isn't justified in terms of the expected improvement.



### V.3 Repeated Closed-Loop Eigenvalues

System designs which lead to repeated closed loop poles also serve to differentiate algorithms. As we noted earlier, repeated eigenvalues lead to difficulties in computing eigenvectors.

These problems arise, for example, in satellite control contexts [Bryson, Feb. 1978]. A conservative system experiencing rotation can lead to a zero configuration of the form:



When we consider the design of a controller for this system, using a quadratic performance index, we recognize that the non-minimum phase zeroes will effectively be mapped onto their real-conjugate counterparts.

If we now weight the states in the quadratic performance index

$$J = \frac{1}{N} \sum_{i=1}^N x_i^T A x_i + u_i^T B u_i$$

much more heavily than the control, we find we have repeated eigenvalues (for a sample root-locus, see Figure 5.9). This would happen if, for example, the control,  $u$ , was unbounded ( $B=0$ ) but we desired little variation in the states.

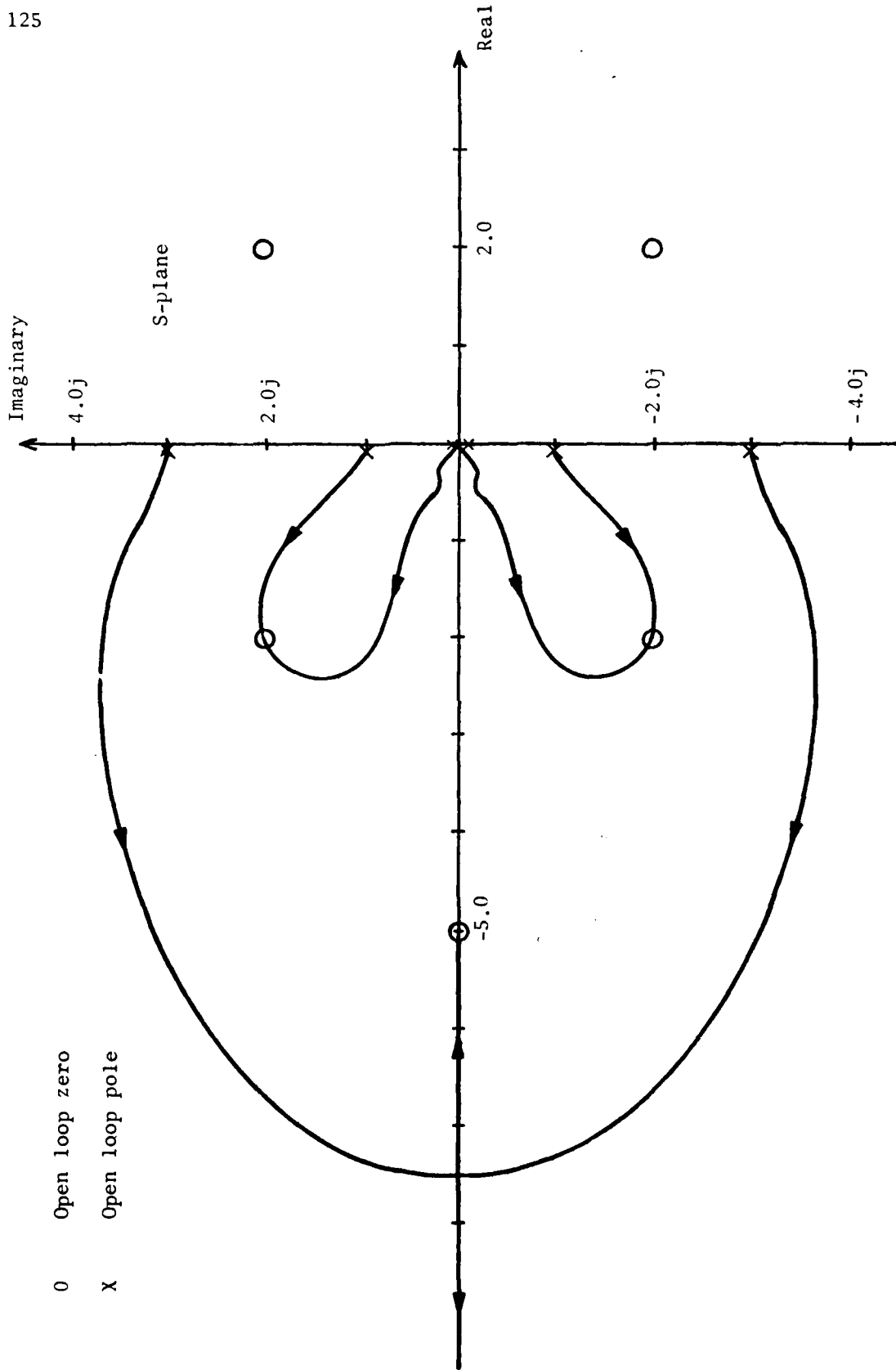


Figure 5.9 CONSERVATIVE SYSTEM ESTIMATOR ROOT LOCUS (as a function of  $Q/R$ ).

We note that for the example of Figure 5.9, a ratio of  $A/B$  of greater than  $10^{10}$  yields nearly repeated eigenvalues. For a ratio of  $A/B$  of greater than  $10^{11}$ , the version of eigenvalue decomposition we have implemented fails.

Square root doubling also fails in approximately the same region. This happens because  $B$  going to zero in the control context is equivalent to the measurement noise covariance,  $R$ , going to zero in the estimation context. To handle the case of  $B$  or  $R$  equal to zero, both algorithms must be reformulated.

The repeated-root example also arises in less extreme examples. In the case of repeated roots undisturbed by process noise, the roots are not moved by the Kalman gains; they remain repeated. Also, the roots of the closed-loop system may be moved so that they happen to coincide.

In these cases, with non-zero measurement noise covariance, the square root algorithms work; the eigenvalue decomposition algorithm would have to be reformulated. Both cases seem to be pathological. In the case of repeated roots undisturbed by process noise, the states are not controllable through  $\Gamma$ , and the Kalman gains are zero. In the latter case, the conjunction of closed-loop roots resulting in a Hamiltonian without a full set of eigenvectors seems rare. Although we can conjecture possible problems which would have these characteristics, we have yet to find an actual industrial example.

## EMPIRICAL RESULTS

In the last chapter we discussed sets of problems which could be solved by some algorithms, but not by others. In this chapter, we shall examine problems that can be solved by all algorithms, but at varying cost and with varying performance.

Cost is determined primarily by two factors: the computer resources allocated to the task, and the amount of time these resources are required. For the algorithms presented in Chapter III, the primary computer resource will be main processor data memory. For processors with separated instruction and data space, but with very expensive instruction memory (e.g., Floating Points Systems AP-120B), program size will be an issue. For processors with combined memories (e.g., Digital Equipment Corporation PDP-11/34), program size will also be important for small problems (when the program's overhead is relatively large).

For solving large problems, those which will not fit in central memory, auxiliary memory becomes an important resource. Also resource demands on the central processor could theoretically be a primary resource. However, no current computer has a significant capacity for allocating part of its processing unit to one problem, while simultaneously applying other computational resources to other problems. The questions of large problems and special processors will be relegated to Chapter IX.

Cost, for the purposes of this study, will be measured in terms of memory required and elapsed execution time.

Performance will be measured in terms of solution accuracy. For a given problem and algorithm, how much accuracy is attained given a maximum machine precision.

## VI.2 Memory Requirements

Table 6.1 gives the memory requirements for the algorithms considered. The full input data set, consisting of  $\{\phi, \Gamma, H, Q, R\}$ , is assumed to be external and not available to the algorithm. Similarly, the output arrays consisting of the error covariance matrix,  $P$ , and the filter gains,  $K$ , are considered to be separate from the algorithm work space.

For the work space estimates, the table presents both theoretical minima and requirements for algorithms as currently implemented. The minima are not necessarily practically attainable. Issues such as data shuffling overhead and addressing complexity were ignored. These considerations may make this limit unreachable.

The minima for eigenvector decomposition assume the QR algorithm operates on the Hamiltonian in place. The square root doubling minima assumes four triangular matrices need to be stored ( $W^{T/2}, P^{1/2}, T_1$  and  $T_2$ ), two full matrices are required (for  $T_{21}$  and for  $\phi_0$ ) and that one full-order temporary matrix is required, to hold partial results.

Most of the algorithms were implemented optimized for speed. The square-root doubling algorithms were written to minimize data access time. The eigenvector decomposition routine, in contrast, was originally coded for an IBM 370. Presumably, memory was not considered a valuable resource.

All algorithms were coded using double-precision arithmetic (sixteen decimal digits of precision). If the square root algorithms perform adequately with reduced precision, then the doubling algorithms would

Table 6.1  
MAIN MEMORY STORAGE REQUIREMENTS

	ASSUMING	
	$m, p \ll n$	$m, p \approx n$
Input and Output Storage	$2n^2 + O(n)$	$7n^2$
Full iteration	$3n^2 + O(n)$	$8n^2$
Square root iteration <sup>1</sup>	$n^2 + O(n)$	$6n^2$
Eigenvector Decomposition <sup>2</sup>	$4n^2 + O(n)$	$4n^2 + O(n)$
Square Root Doubling <sup>1</sup>	$5n^2$	$5n^2$
Bilinear Square Root Doubling <sup>1</sup>	$5n^2$	$5n^2$

Notes:

1: All data assumed stored at full precision

2: Assuming the standard, IMSL QR algorithm

$n$  = number of states

$m$  = number of input disturbances

$p$  = number of measurements

only require half the storage. Similarly, if double precision computations are overly conservative for eigenvector decomposition, the memory requirement could be lowered for them also (see performance, Section VI.4).

For problems with many inputs and many outputs, all algorithms require approximately the same data memory. For the single-input single-output problems, square root iteration is preferable. An important observation is that these results are strongly effected by the storage required for the input and the output of data. In many practical cases, the variations among algorithms would be unimportant. Also, if the input data can be overwritten, then the square root and eigenvector decomposition algorithms may require no additional storage.

### VI.3 Computation Time

The two fastest algorithms, eigenvector decomposition (EVD) and square-root doubling (SQD), were compared to determine relative speed. The eigenvector decomposition routine was taken from a standard package designed to handle many facets of the design problem-- designing discrete controllers and estimators for a continuous plant, determining observability, etc. For the purposes of this comparison, this EVD program was liberally ravaged to minimize excess computations. However, it was inexpertly coded (see the previous section) and mere pruning failed to optimize the code. The fundamental algorithms-- the Householder and QR algorithms-- were part of an industry standard package. Performance is therefore assumed to be near optimum.

In contrast, square root doubling was implemented for optimal performance on a small machine. Separate facets of the design problem are solved by distinct programs (see Appendix A1). Each implementation is within a factor of two of optimal performance for the algorithm as specified, not including data entry.

The target machine was a PDP-11 minicomputer with a thirty-two thousand word address space. Since the EVD algorithm was originally targeted for an IBM 370, with a relatively unlimited address space, the transplant severely limited the size of problem that could be solved. For this study, the SQD program was limited to twelfth order problems, and the EVD program was limited to sixth order problems.

Figure 6.1 presents preliminary time trials. System order is plotted against relative elapsed execution time, given by

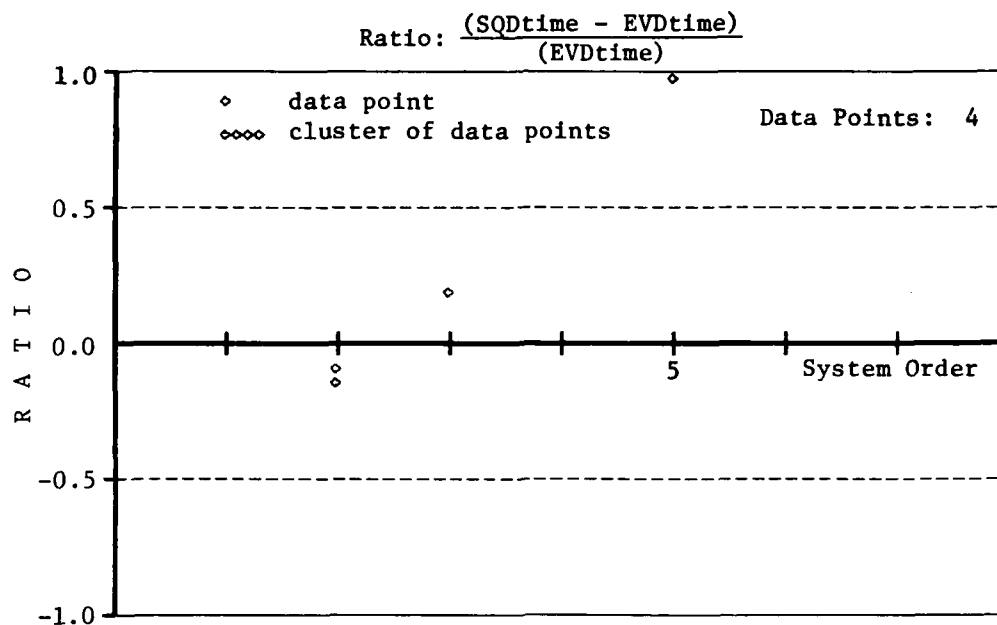
$$\frac{(\text{SQD time}) - (\text{EVD time})}{(\text{EVD time})}$$

Thus, positive increments result when eigenvector decomposition is the faster algorithm. The EVD algorithm appears faster in these results.

This data was collected on a PDP-11/34 using a relatively fast hardware floating point processor and a threaded-code Fortran compiler. (The compiler is basically interpretive.) This minicomputer is medium speed, and uses a memory well matched to the speed of the central processor.

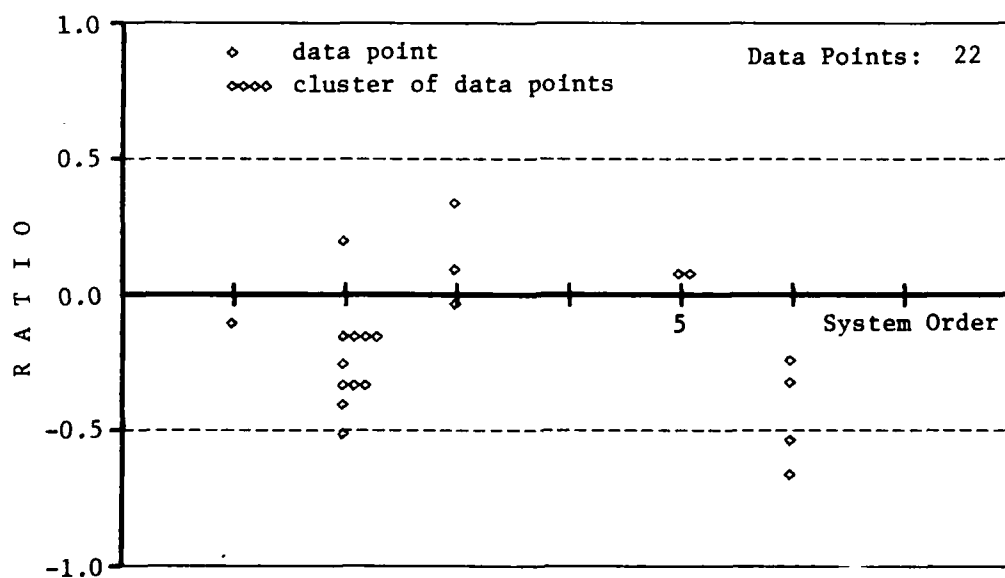
Figure 6.2 presents the next time trials. It consists of twenty-two problems, and includes the results from the data presented in Figure 6.1. These results were collected on a PDP-11/70 using a Fortran compiler that generated pure executable code. Notice the different trend.





System: PDP 11/34 with Princeton Fortran

Figure 6.1 RELATIVE ELAPSED COMPUTATION TIME - TRIAL I



System: PDP 11/70 with Optimized Fortran Four Plus Compiler

Figure 6.2 RELATIVE ELAPSED COMPUTATION TIME - TRIAL II

In this second trial the square-root doubling algorithm is consistently faster. Although the algorithms are identical in both cases, the environments have changed, albeit subtly. The change in Fortran compilers has altered the relative execution times of the program steps. In the central processor, the computation-time mix between data addressing and floating point instructions has changed. Also, the second computer uses a buffered (cached) memory system that can significantly alter memory dynamics.

Figures 6.3, 6.4, and 6.5 present the final results for time trials, run on an IBM 370/168 using the Fortran H optimizing compiler; both programs were extended to handle problems of up to twentieth order. The square root doubling algorithm was also rewritten in pure Fortran, increasing its execution time by approximately 80%.

Both algorithms give results qualitatively comparable to the PDP11 for systems of order less than six, with the square root doubling algorithm typically slightly better. Above tenth-order, however, the Eigenvector decomposition algorithm is markedly superior, outperforming the square root algorithm by a factor of two to six.

It is interesting to include a curve of  $O(n^3)$  on Figures 6.3 and 6.4, normalized to the same value on each graph for  $n = 5$ . We note that the square root doubling trials are predominantly above the  $O(n^3)$  curve; this result is plausible since the doubling algorithm requires  $O(n^3)$  computations per iteration, with the total number of iterations typically increasing with system order. In contrast, the time trials for the EVD algorithm are below the  $O(n^3)$  curve, as we anticipated in Chapter II.

An interesting anomaly can be found in the one eighth-order system where the square root doubling algorithm was faster than the EVD trial.

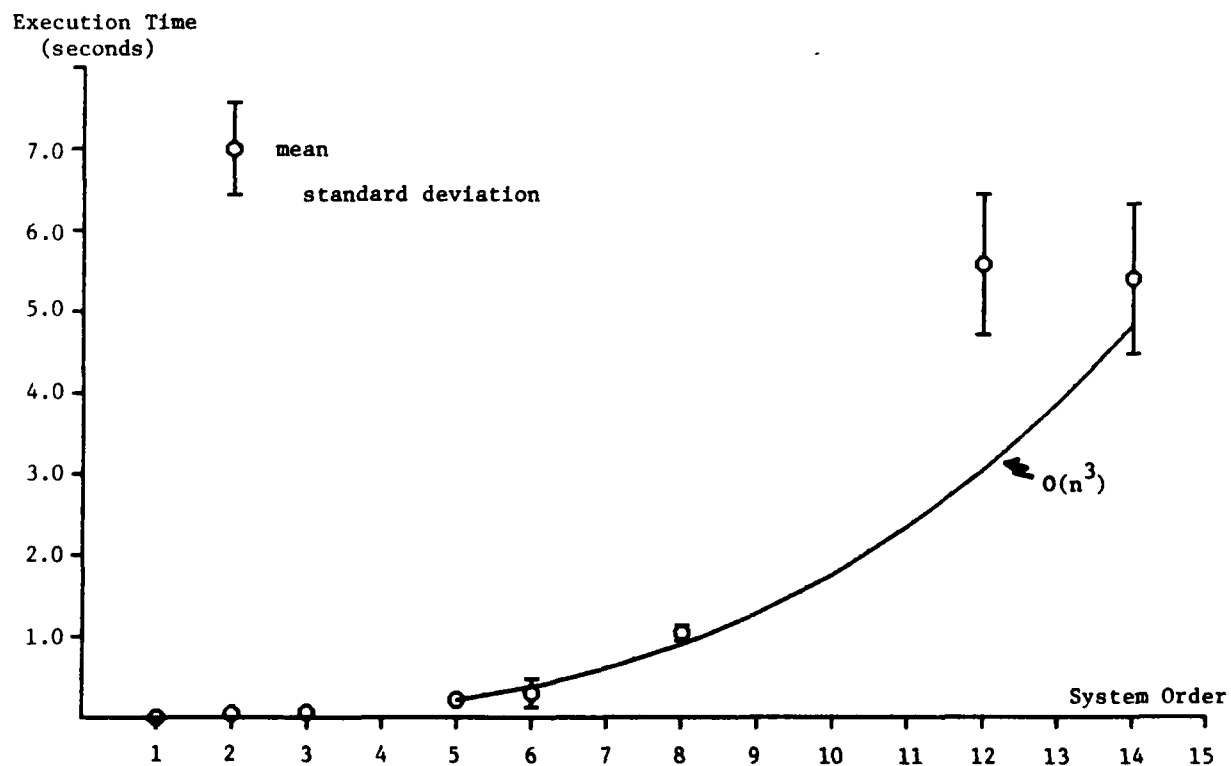
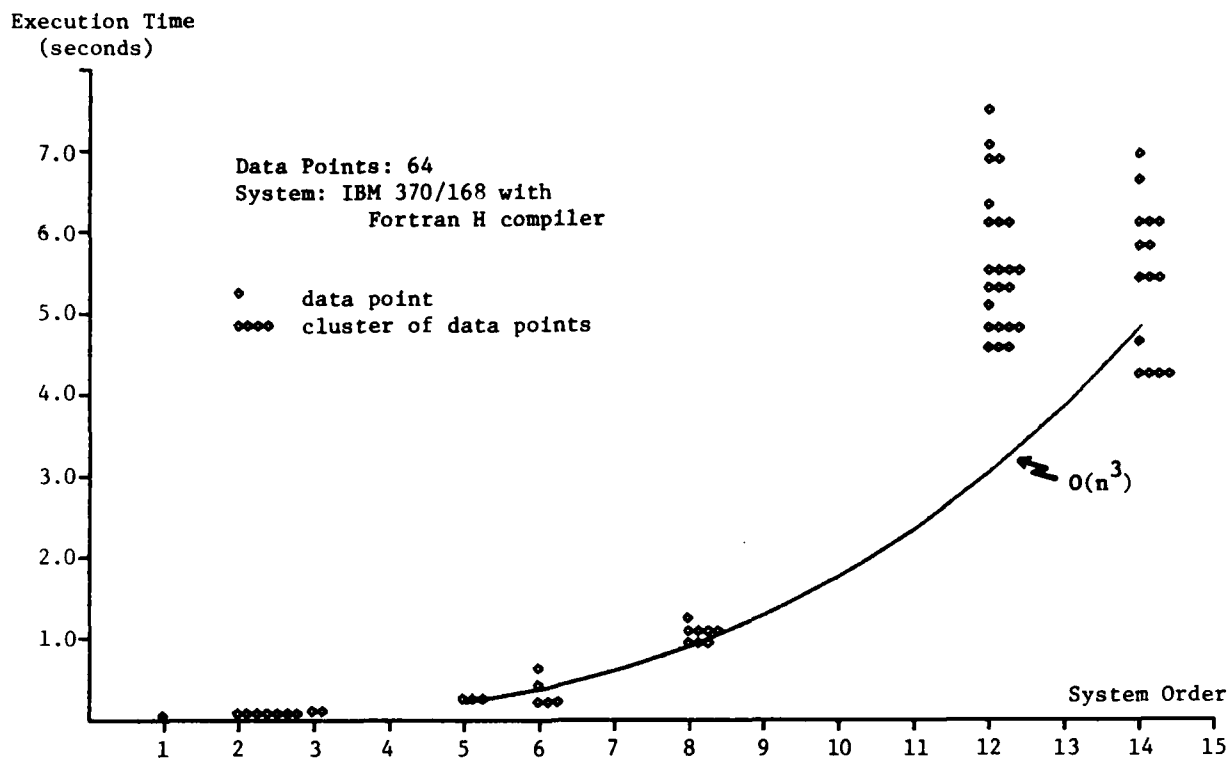


Figure 6.3 ELAPSED COMPUTATION TIME FOR SQUARE ROOT DOUBLING

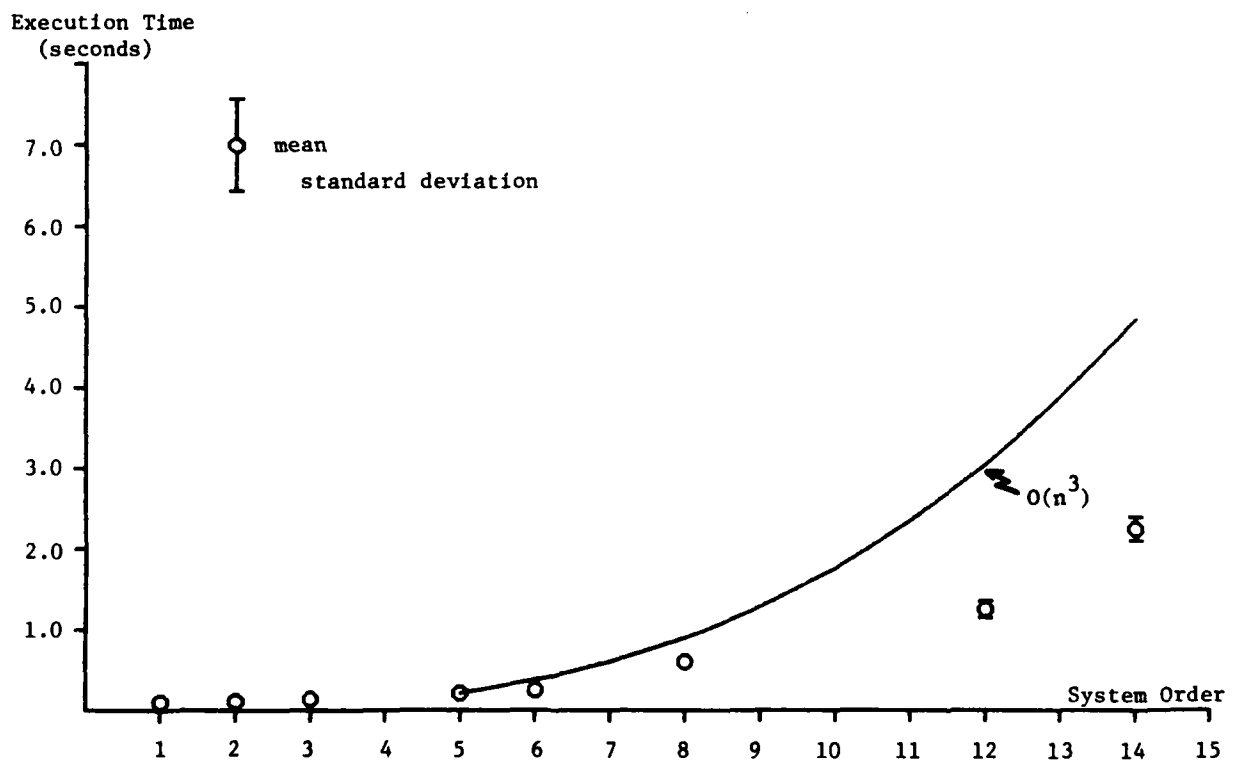
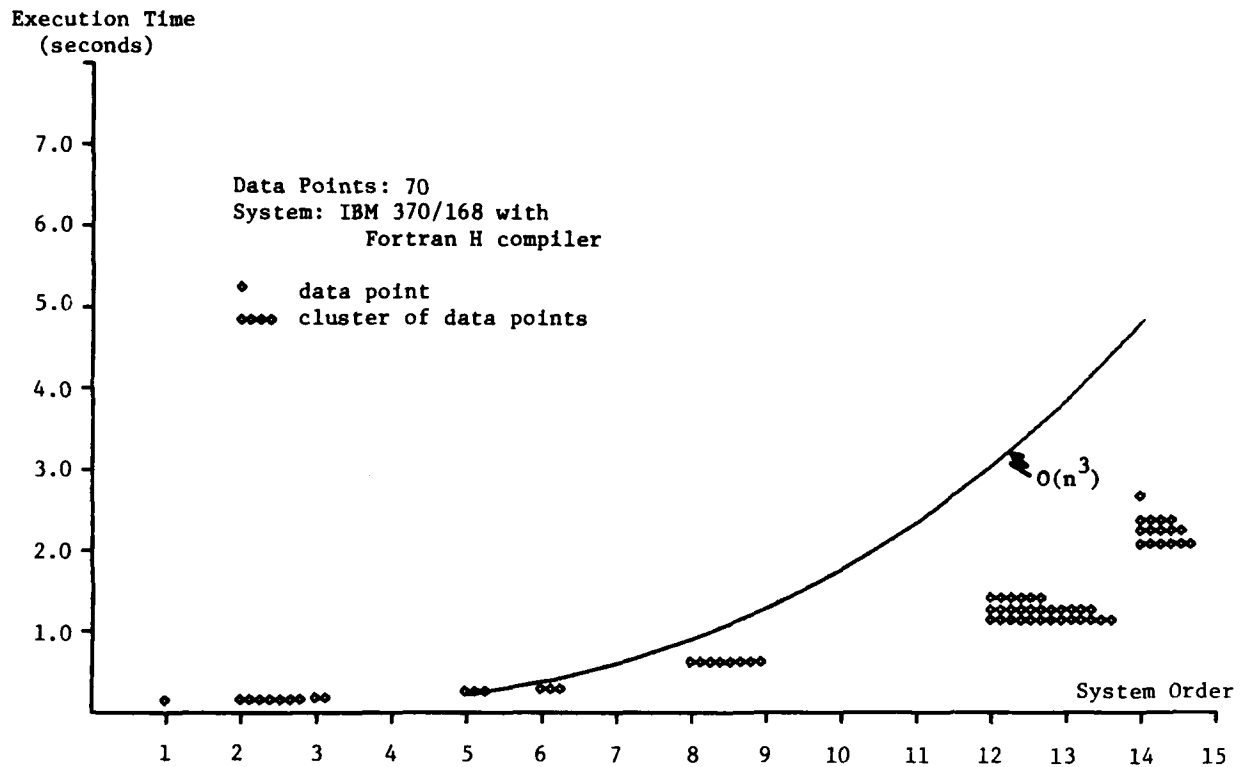


Figure 6.4 ELAPSED COMPUTATION TIME FOR EIGENVECTOR DECOMPOSITION

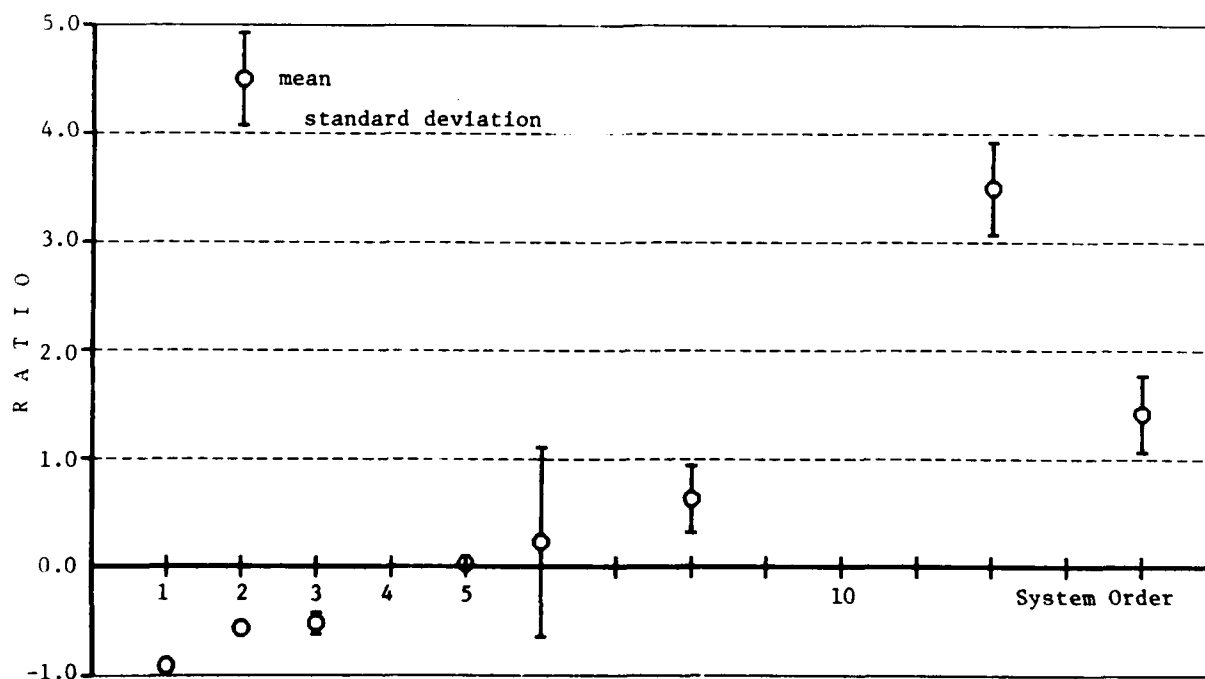
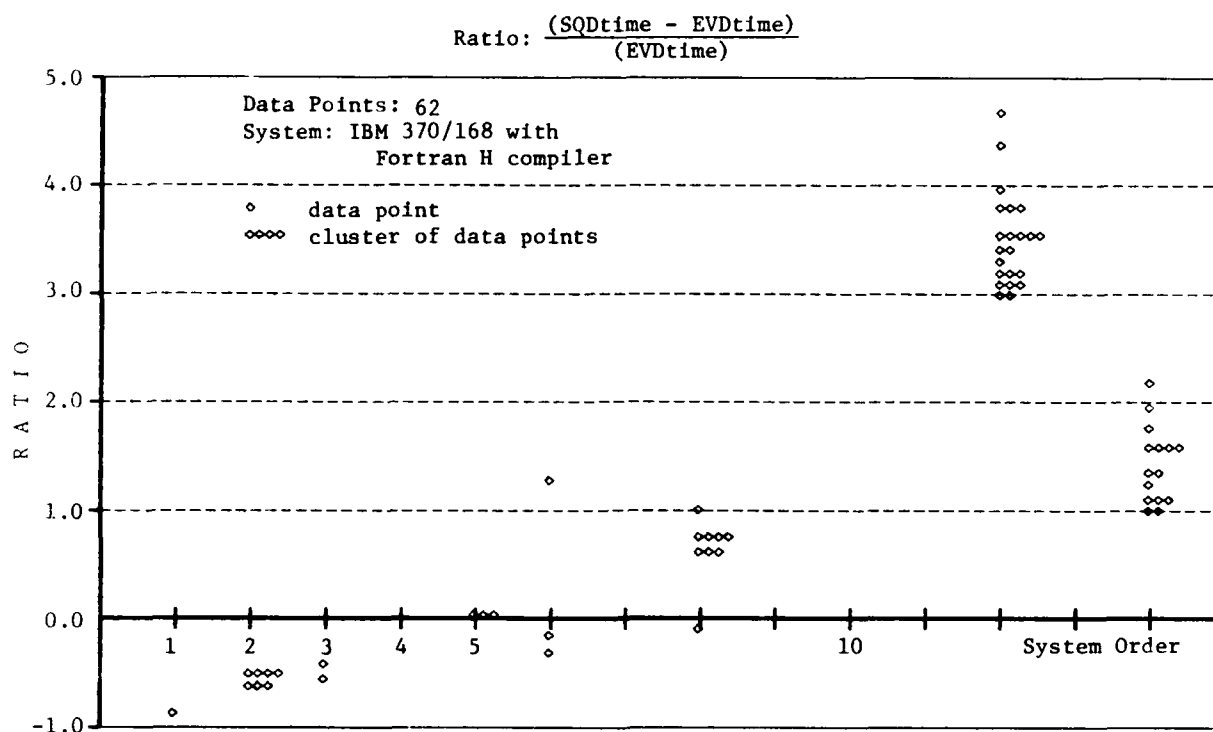


Figure 6.5 RELATIVE ELAPSED COMPUTATION TIME - TRIAL III

This was the only data point for which the closed-loop poles of the system were all of magnitude less than or equal to 0.1 in the Z-plane.

Figure 6.6 shows examples of the convergence behavior for the doubling algorithm, which behave as expected; convergence occurred when the difference between the trace of the covariance matrix on two successive iterations was less than the machine's epsilon. The error is fairly constant for most of the computation, and then drops off very rapidly.

These results are partially conclusive. For small problems, the difference between algorithms is slight, and for large problems the EVD algorithm is consistently faster for almost all problems. Alternate formulations of the SQD need to be considered (for example, calculating  $T_1$  explicitly instead of inverting  $T_1^{-T}$ ), although these are not likely to change the overall conclusion.

Another valuable lesson has been learned: that two very similar computers can give qualitatively different results. The interaction between algorithms and their host computers, especially the memory dynamics, has not been properly appreciated.

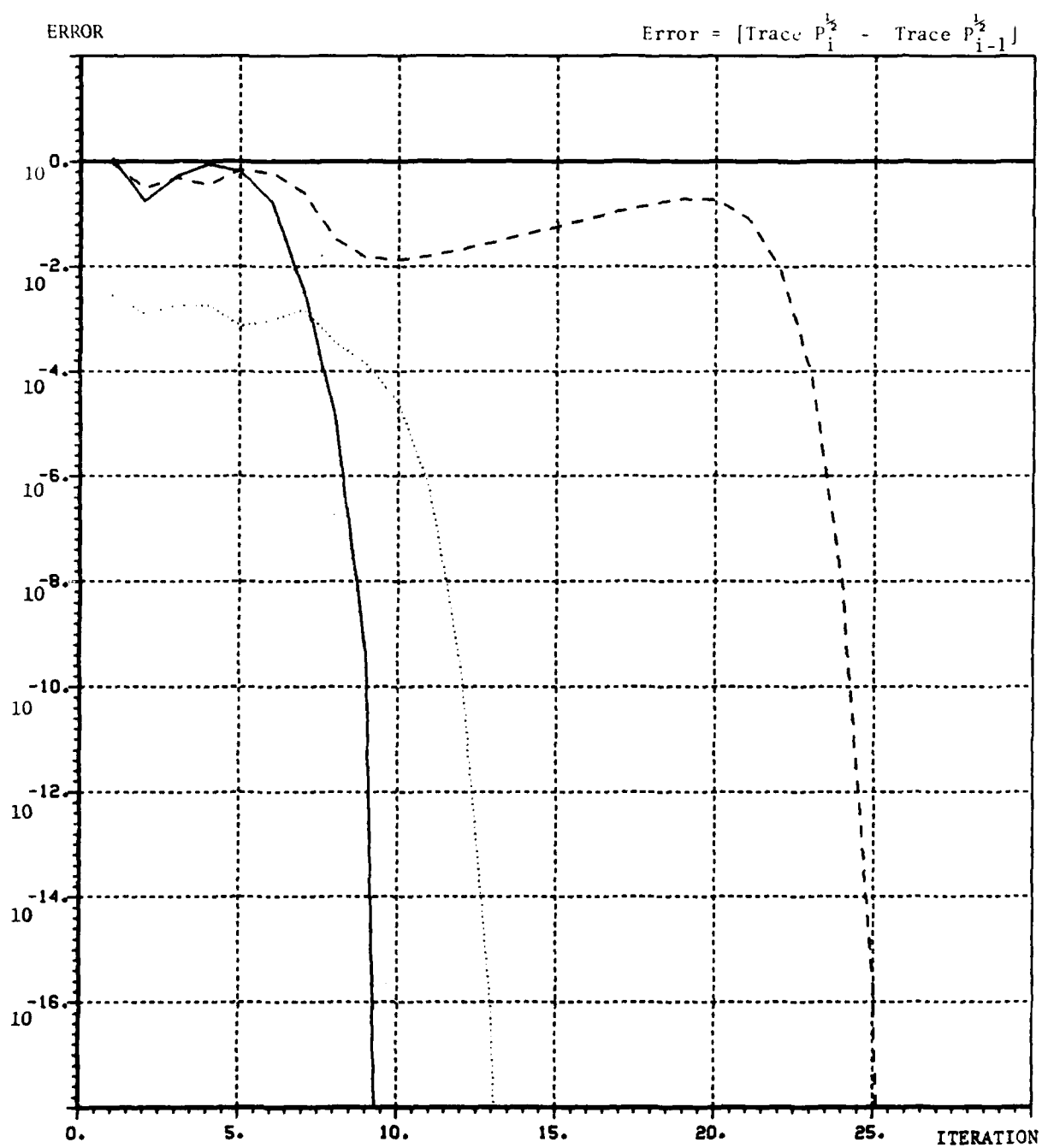


Figure 6.6 EXAMPLES OF SQUARE ROOT DOUBLING CONVERGENCE

#### VI.4 Performance

For all of the test problems considered in the previous section, square-root doubling and eigenvector decomposition gave identical results to at least one part in  $10^6$ . All computations were in double precision (sixteen decimal digits of accuracy).

A question still to be answered concerns the precision required to accurately and stably determine the steady-state Riccati solution. As demonstrated in Chapter III, square root algorithms often require only half the precision required for full iteration, but these SQD algorithms may require full precision in some cases. Do these full-precision problems occur in practice, or is this upper bound irrelevant and overly loose?

Second, how does the required minimum floating-point precision compare for various algorithms? This question is especially important because all algorithms are implemented using double-precision computations. Double-precision arithmetic requires twice as much storage and up to four times more execution time than single-precision calculations. A reduction in precision is significant both in time and in memory.

The efficient answer to these problems required modifying the Fortran compiler. This has not yet been completed.

#### VI.5 Convergence Sensitivity

Chapter Four considered solving the continuous steady-state Riccati equation using square-root doubling. One approach required converting the continuous problem to a discrete-time problem by an application of the bilinear transform to the original Riccati equation. The important quantity in the mapping was the approximation to the exact discrete transition matrix, given by:



$$(\alpha I + F)(\alpha I - F)^{-1} \quad (5.1)$$

where  $F$  was the continuous state dynamics matrix. We noted that  $\alpha$  should not coincide with an eigenvalue of  $F$  and should be positive. No other criterion was specified.

Figures 6.7 and 6.8 show the dependence of elapsed time on the choice of  $\alpha$  for a representative sample of problems. Figure 6.7 plots elapsed time versus  $\alpha$ ; Figure 6.8 plots relative elapsed time versus  $\alpha$  (centered about the smallest  $\alpha$  giving minimum elapsed time). These results for square root doubling are consistent with the results found by Hitz [1972] for iteration of the mapped, discrete-time Riccati equation.

The computation converges quickest for choices of  $\alpha$  near unity. If  $\alpha$  becomes too large or too small, the mapping (5.1) tends to the identity matrix, modulo the sign. When this happens, the difference between the discrete-time quantity  $\phi_D^T P \phi_D$  and  $P$  becomes very small. Similar problems arise for other mapped terms in the equation (see Chapter IV, equations (3.1) through (3.3)).

This problem can be severe. In example (5) of Figure 6.8, any choice of  $\alpha$  less than 0.014 caused a divergent solution.

Another complication arises if  $F$  has a positive eigenvalue near  $\alpha$ . For example, result (3) has an eigenvalue at .004 and a complex pair at  $.025 \pm 0.64j$ . If  $\alpha$  is near an eigenvalue, then some entries of  $(\alpha I - F)^{-1}$  will be very large. The iteration calculation leads to taking the difference of two very large and almost equal numbers, with concomitant failures in convergence.

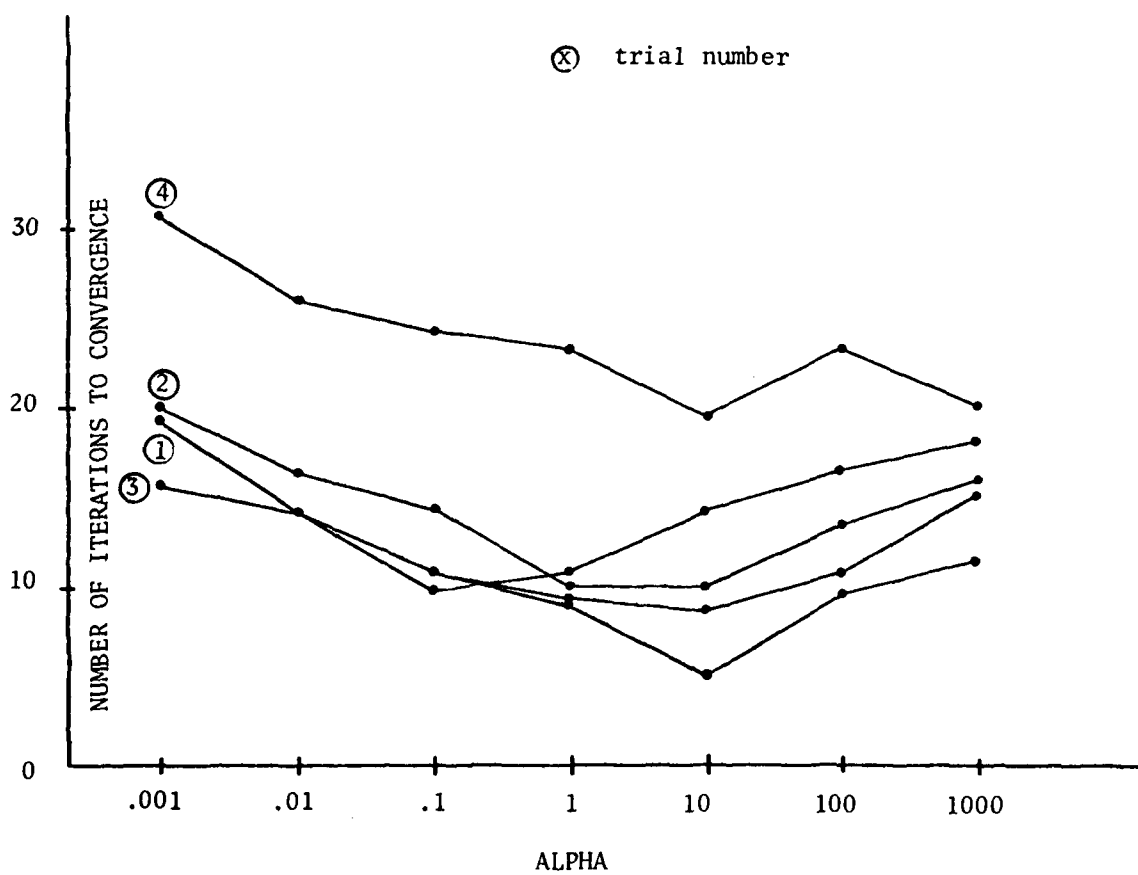


Figure 6.7 CONVERGENCE OF BILINEAR SQUARE ROOT DOUBLING

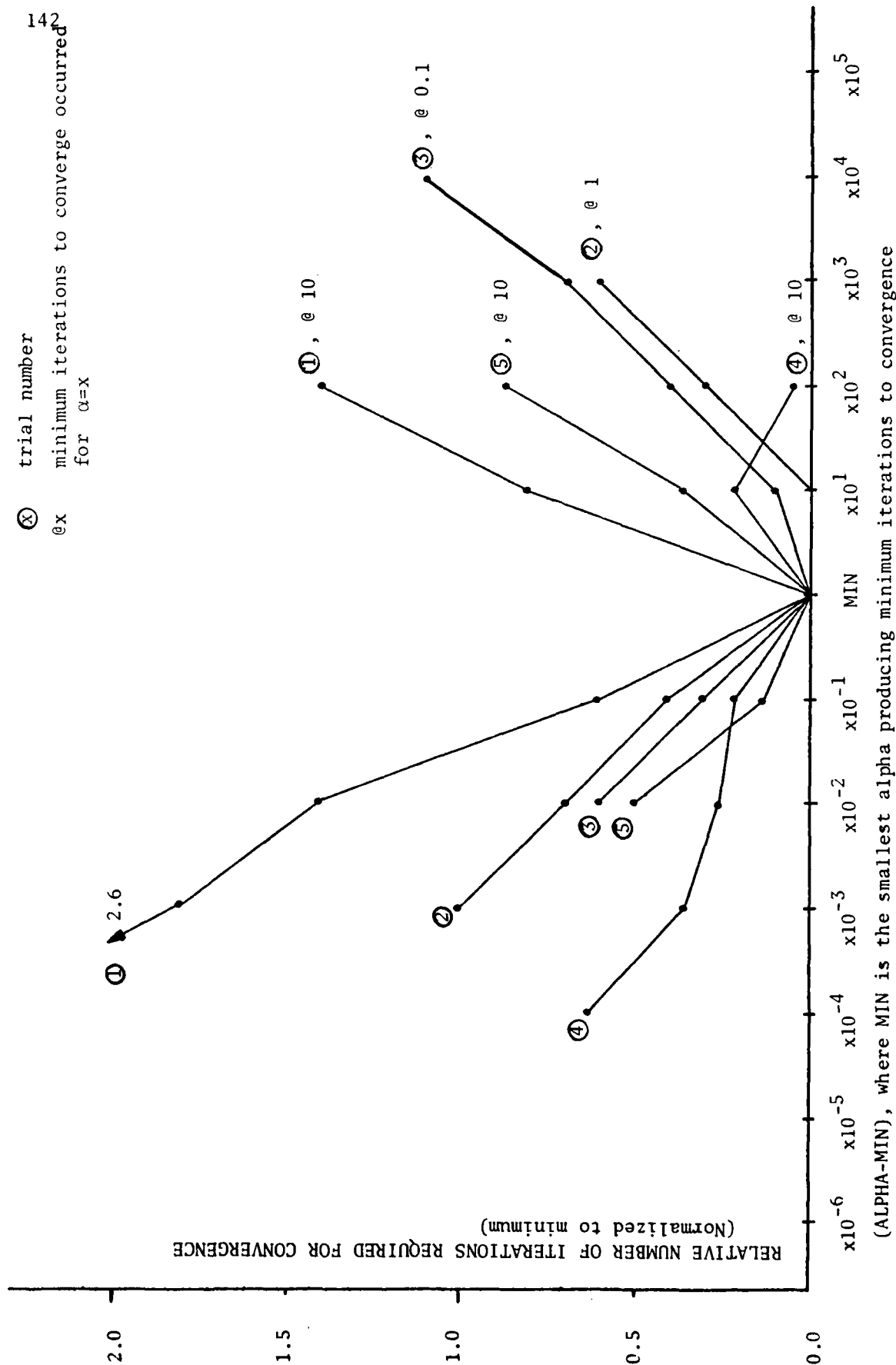


Figure 6.8 BILINEAR SQUARE ROOT DOUBLING CONVERGENCE

In Chapter IV we noted that the bilinear approach was basically a two step procedure; first the transform was taken, then the iterations were performed. A question was raised concerning the significance of the added overhead of this initial step. However, for systems above fifth-order, the time required to perform the bilinear transform does not appear to be a significant part of the elapsed time for the computation.

#### VI.6 Conclusions

Eigenvector decomposition and square root doubling require comparable elapsed computation time for problems smaller than sixth-order. For larger systems, the QR algorithm was faster.

At least for the smaller problems, speed is not an important decision criterion. This is true both because the two algorithms are comparable in speed, and because the total elapsed time was negligible for these applications; the sixth-order problems always required less than two seconds to solve. Memory requirements are also of little importance, judging from conjectured minimum memory specifications; this is certainly true if the input data can be overwritten. Other criteria emerge as being more important, including

- 1) Accuracy,
- 2) General applicability,
- 3) Adaptability to and synergy with the computing hardware.

The bilinear transform was quite sensitive to the mapping constant,  $\alpha$ . Minimum iterations occurred for  $\alpha$  near unity. No direct

relationship between  $\alpha$  and the dynamics matrix,  $F$ , was established. Hitz [1972] suggested that  $\alpha$  should equal the average of the moduli of the eigenvalues of  $F$ . We have insufficient examples to substantiate this conjecture. However, departing from the optimal choice for  $\alpha$  by several orders of magnitude commonly leads to the divergence of the doubling solution.

The computational overhead of the bilinear transform was negligible for the sixth-order systems examined.

In the next chapter the topic changes. The discussion turns to considering the design of optimal compensation given qualified uncertainty in the parameters of the system model.

## Chapter VII

## PARAMETER SENSITIVITY IN DISCRETE SYSTEMS

In many optimal control applications, where state-estimators rather than directly measured states are used for feedback, the closed loop system is excessively sensitive to variations in parameters [Berger, 1973]. Early work focused on dynamic desensitization; Bar-Shalom and Sivan [Bar-Shalom, 1969] augmented the estimator state to include the unknown parameters, and developed a suboptimal scheme for solving this problem. Berger [Berger, 1973] assumed constant but unknown plant parameters, with a known parameter probability distribution; he solved the continuous controller problem, assuming rectangular probability distributions. (His solution was optimal.)

Palsson and Whittacker [Palsson, 1972] solved the continuous single input-single output case assuming a Gaussian distribution for the system parameters. Hadass [Hadass, 1974] solved the same problem for multi-input, multi-output systems.

In this chapter we present the optimal solution for the discrete multi-input, multi-output system assuming uncertain constant parameters whose uncertainty can be described by a Gaussian distribution. We will also comment upon the efficacy of this approach in terms of Bryson's "oblivious" filter [Bryson, 1977].

## VII.2 Chapter Outline

We begin by considering the uncertain parameters of the system as random variables. A performance index will then be defined as the weighted sum of the steady-state control and state covariances, which in turn are a function of random system disturbances and the variances of the uncertain parameters. This performance-index will then be minimized by adjusting a set of free-parameters. These gains are normally the Kalman estimator gains and the controller feedback gains. The appropriate equations for desensitizing discrete systems are then developed.

With the theory established, an algorithm was developed and implemented. Several practical examples were studied, including the problem of desensitizing a marginally stable high performance aircraft.

### VII.3 Sensitivity Equations for Discrete Systems

We begin by assuming we can describe the physical system as a set of states,  $x_k$ , governed by linear vector difference equations:

$$\begin{aligned} x_{k+1} &= \phi_1 x_k + \Gamma_{11} u_k + \Gamma_{21} w_k \\ y_k &= H_1 x_k + v_k \end{aligned} \quad (7.1)$$

These equations have two sets of inputs-- a deterministic input,  $u_k$ , and stochastic inputs,  $w_k$  and  $v_k$ . These stochastic inputs are assumed to be Gaussian random variables, determined completely by their mean and variance. We shall assume:

$$\begin{aligned} \mathbb{E} \begin{bmatrix} w_i \\ v_i \end{bmatrix} \begin{bmatrix} w_j^T & v_j^T \end{bmatrix} &= \begin{bmatrix} Q & 0 \\ 0 & R \end{bmatrix} \delta_{ij} \\ \mathbb{E}(v_i) &= \mathbb{E}(w_i) = \mathbb{E}(x_0) = \mathbb{E}(x_0 v_i^T) = \mathbb{E}(x_0 w_i^T) = 0 \\ \mathbb{E}(x_0 x_0^T) &= P_0 \end{aligned} \quad (7.2)$$

Bryson [Bryson, 1975] has shown that an interesting class of solutions arises from minimizing the quadratic performance index:

$$J = \lim_{N \rightarrow \infty} \frac{1}{N} \mathbb{E} \sum_{i=0}^{N-1} (x_i^T A x_i + u_i^T B u_i) \quad , \quad (7.3)$$

where

$$\begin{aligned} u_i &= -C \hat{x}_i \\ \hat{x}_i &= \bar{x}_i + K(y_i - H \bar{x}_i) \\ \bar{x}_{i+1} &= \phi \hat{x}_i + \Gamma u_i \end{aligned} \quad (7.4)$$



and we choose  $C$  and  $K$  to minimize  $J$ . (We note in passing that if we select  $\phi = \phi_1$ ,  $H = H_1$ , and  $\Gamma = \Gamma_{11}$  then  $\hat{x}_i$  is the maximum likelihood estimate of  $x_i$  as originally outlined by Kalman.) The design procedure, then, is to choose  $K$  to give the maximum likelihood estimate,  $\hat{x}_i$ , and to choose  $C$  to minimize  $J$ . The closed loop performance of the system can then be varied parametrically through the weighting matrices  $A$  and  $B$ .

To facilitate handling this system, we begin by writing the augmented state equations:

$$\begin{bmatrix} x_{k+1} \\ \hat{x}_{k+1} \end{bmatrix} = \begin{bmatrix} \phi_1 & -\Gamma_{11}C \\ KH_1\phi_1 & (I-KH)(\phi-\Gamma_{11}C) - KH_1\Gamma_{11}C \end{bmatrix} \begin{bmatrix} x_k \\ \hat{x}_k \end{bmatrix} + \begin{bmatrix} \Gamma_{21} & 0 \\ KH_1\Gamma_{21} & K \end{bmatrix} \begin{bmatrix} w_k \\ v_{k+1} \end{bmatrix} \quad (7.5)$$

$$J = \lim_{N \rightarrow \infty} \frac{1}{N} \sum_{i=0}^{N-1} \epsilon [x_i^T (A + C^T B C) x_i]$$

which we shall henceforth write as

$$x_{k+1} = \phi x_k + \Gamma w_k, \quad x_k = \begin{bmatrix} x_k \\ \hat{x}_k \end{bmatrix}, \quad w_k = \begin{bmatrix} w_k \\ v_{k+1} \end{bmatrix} \quad (7.6)$$

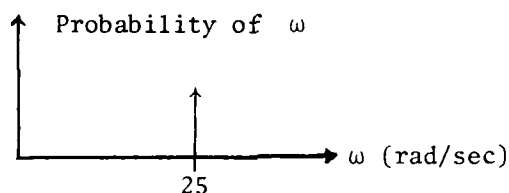
$$J = \lim_{N \rightarrow \infty} \frac{1}{N} \sum_{i=0}^{N-1} \epsilon [x_i^T (A + C^T B C) x_i]$$

$$A = \begin{bmatrix} A & 0 \\ 0 & 0 \end{bmatrix}, \quad C = [0 \mid -C]$$

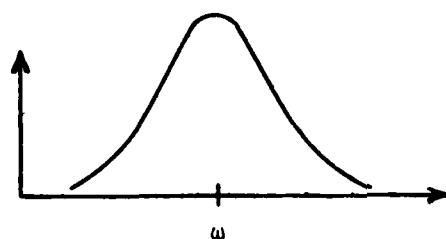
### Introduction of Uncertain Parameters

We assume  $\Phi$  and  $\Gamma$  are constant coefficients, but that they are derived from physical parameters of uncertain value. We propose to model these physical parameters as Gaussian random variables of known mean and variance.

We usually assume that all physical constants are known exactly -- the frequency,  $\omega$ , is precisely 25 radians/sec.



An alternate assumption would arise if we expect the frequency,  $\omega$ , to be about 25 rad/sec, but we are 95% certain  $\omega$  lies within  $\pm 5$  radians of the expected value. Modelling  $\omega$  as a Gaussian distribution, then 95% certainty encompasses about 2 standard deviations, so the variance of  $\omega$  would be about 6, and we would have



$$\bar{\omega} = 25 \text{ rad/sec}$$

$$\sigma^2(\omega - \bar{\omega}) = 6(\text{rad/sec})^2$$

$\omega(\text{rad/sec})$

Note again that we assume  $\omega$  remains constant over time, but that we are uncertain as to its exact value.

We shall henceforth describe these uncertain parameters,  $\psi$ , with a prescribed and constant normal distribution:

$$\begin{aligned}
\psi &= N(\bar{\psi}, \Psi) \\
\chi_{k+1} &= \Phi(\psi)\chi_k + \Gamma(\psi)N(k) \\
J &= \lim_{N \rightarrow \infty} \frac{1}{N} \sum_{i=0}^{N-1} \varepsilon[\chi_i^T (A + C^T B C) \chi_i] \quad .
\end{aligned} \tag{7.7}$$

We now will be taking the expected value of  $\chi_i$  over the random processes  $w_i$  and  $v_{i+1}$ , and over the random vector  $\psi$ .

Since, for any vector  $v, w$ , and matrix  $A$  we can write:

$$\begin{aligned}
v^T w &= \text{trace}(wv^T) \\
v^T (Aw) &= \text{trace}[(Aw)v^T]
\end{aligned}$$

we have

$$J = \text{trace} \{ [A + C^T B C] \lim_{N \rightarrow \infty} \frac{1}{N} \sum_{i=0}^{N-1} \varepsilon(\chi_i \chi_i^T) \} \tag{7.8}$$

If we can find an expression for  $\varepsilon(\chi_i \chi_i^T)$ , then we can minimize  $J$  by appropriate choice of  $C, K, \phi, H$ , and  $\Gamma$ .

The uncertain parameter vector,  $\psi$ , entered in a non-linear fashion, so we begin by linearizing the equation for  $\chi_k$ , (7.6), about the nominal value for  $\psi$ .

$$\chi(\eta, \bar{\psi}) = \chi^{\text{nom}}(\eta, \bar{\psi}) + \delta\chi(\eta, \delta\psi)$$

where  $\eta$  is the random process composed of  $w$  and  $v$ ,  $\chi^{\text{nom}}$  is the state vector obtained when the parameters have their nominal values, and  $\delta\chi$  is the perturbation in the state vector due to a perturbation,  $\delta\psi$ , in the uncertain parameters. Our goal is to write state equations in terms of  $\chi^{\text{nom}}$  and  $\delta\chi$ . Assuming small perturbations in  $\psi$ , so

that a first order expansion is satisfactory, we have

$$\chi_{k+1}^{\text{nom}}(\eta, \bar{\psi}) + \delta\chi_{k+1}(\eta, \delta\psi) = \Phi(\bar{\psi} + \delta\psi) [\chi_k^{\text{nom}}(\eta, \bar{\psi}) + \delta\chi_k(\eta, \delta\psi)] + \Gamma(\bar{\psi} + \delta\psi) N_k$$

Linearizing  $\Phi$  by taking a Taylor Series expansion, we get

$$\Phi(\bar{\psi} + \delta\psi) = \Phi(\bar{\psi}) + \frac{\partial\Phi}{\partial\psi} (\bar{\psi} + \delta\psi - \bar{\psi}) + R_2(\delta\psi)$$

Neglecting the higher order terms, i.e., assuming  $R_2 \sim 0$ , we have

$$\Phi(\bar{\psi} + \delta\psi) = \Phi(\bar{\psi}) + \delta\Phi(\delta\psi), \quad \delta\Phi(\delta\psi) = \sum \left[ \frac{\partial\Phi}{\partial\psi_i} \right]_{\psi=\bar{\psi}} \delta\psi_i$$

$$\Gamma(\bar{\psi} + \delta\psi) = \Gamma(\bar{\psi}) + \delta\Gamma(\delta\psi)$$

$$\chi_{k+1}^{\text{nom}} + \delta\chi_{k+1} = (\Phi + \delta\Phi)(\chi_k^{\text{nom}} + \delta\chi_k) + (\Gamma + \delta\Gamma)N_k \quad (7.9)$$

$$= (\Phi\chi_k^{\text{nom}} + \Gamma N_k) + (\delta\Phi\chi_k^{\text{nom}} + \Phi\delta\chi_k + \delta\Gamma N_k) + (\delta\Phi\delta\chi_k).$$

The last term, to first order, is again neglected. Separating by independent variables, we get

$$\chi_{k+1}^{\text{nom}}(\eta, \bar{\psi}) = \Phi(\bar{\psi})\chi_k^{\text{nom}}(\eta, \bar{\psi}) + \Gamma(\bar{\psi})N_k, \quad \chi_0^{\text{nom}} = 0 \quad (7.10a)$$

$$\delta\chi_{k+1}(\eta, \delta\psi) = \delta\Phi(\delta\psi)\chi_k^{\text{nom}}(\eta, \bar{\psi}) + \Phi(\bar{\psi})\delta\chi_k(\eta, \delta\psi) + \delta\Gamma(\delta\psi)N_k, \quad \delta\chi_0 = 0 \quad (7.10b)$$

With  $\chi^{\text{nom}}$  and  $\delta\chi$  expressed at linear functions of  $\eta$  and  $\psi$ , we now consider evaluating  $\delta\chi_i \chi_i^T$

$$\delta\chi_i \chi_i^T = \delta\chi_{\eta, \psi}^{\text{nom}} \chi^{\text{nom}T} + \delta\chi^{\text{nom}} \delta\chi^T + \delta\chi \chi^{\text{nom}T} + \delta\chi \delta\chi^T$$

Examining the second term, we find

$$\varepsilon_{\eta, \psi} [\chi(\eta, \bar{\psi}) \delta \chi^T(\eta, \delta \psi)] = \varepsilon_{\eta} [\chi(\eta)] \varepsilon_{\psi} \delta \chi^T(\eta, \delta \psi)$$

$$\varepsilon_{\psi} \delta \chi_{k+1} = \varepsilon (\delta \phi \chi_k^{\text{nom}} + \phi \delta \chi_k + \delta \Gamma N_k)$$

$$= \frac{\partial \phi}{\partial \psi} \varepsilon (\delta \psi \chi^{\text{nom}}) + \phi \varepsilon \delta \chi_k + \frac{\partial \Gamma}{\partial \psi} \varepsilon (\delta \psi \eta_k)$$

$$\varepsilon (\delta \psi) = \varepsilon (\delta \psi \chi) = \varepsilon (\delta \psi \eta) = 0, \text{ since } \eta \text{ and } \psi \text{ are independent.}$$

$$\text{Thus, } \varepsilon_{\eta, \psi} \delta \chi(\eta, \delta \psi) = 0, \quad \varepsilon_{\eta, \psi} [\chi^{\text{nom}} \delta \chi^T] = 0 = \varepsilon [\delta \chi \chi^{\text{nom}^T}]. \text{ Further}$$

since

$$\varepsilon_{\eta, \psi} \chi_{k+1}^{\text{nom}} = \varepsilon (\phi \chi_k^{\text{nom}} + \Gamma N_k) = 0$$

We have that  $\varepsilon \chi_i \chi_i^T = \varepsilon \{[\chi_i - \bar{\chi}_i] [\chi_i - \bar{\chi}_i]^T\}$  so we can define two covariance matrices--

$$\varepsilon_{\eta, \psi} \chi_i \chi_i^T = \varepsilon \chi_i^{\text{nom}} \chi_i^{\text{nom}^T} + \varepsilon \delta \chi_i \delta \chi_i^T = \bar{\chi}_i + \delta \bar{\chi}_i$$

where  $\bar{\chi}_i$  is the state covariance due to random disturbances, and  $\delta \bar{\chi}_i$  is the additional state covariance arising from the uncertainty in the parameters.

If we assume a stable system, then the state covariance converges to a steady-state value, and we can re-write the performance index as

$$J = \text{trace} [(A + C^T B C) [\bar{\chi} + \delta \bar{\chi}]]$$

where  $\underline{\bar{X}}$  and  $\underline{\delta\bar{X}}$  are the appropriate steady state covariance matrices.

$\underline{\bar{X}}$  follows directly from multiplying the governing equation for  $\chi^{\text{nom}}$  by its transpose and taking the expected value; the result is the well-known Lyapunov Equation:

$$\Phi \underline{\bar{X}} \Phi^T - \underline{\bar{X}} = - \Gamma \Theta \Gamma^T, \quad \Theta = \underset{\eta, \psi}{\mathbb{E}} \mathbf{NN}^T = \begin{bmatrix} Q & 0 \\ 0 & R \end{bmatrix}$$

Similarly, we have

$$\begin{aligned} \delta\chi_{k+1} \delta\chi_{k+1}^T &= (\delta\phi\chi + \phi\delta\chi + \delta\Gamma N) (\chi^T \delta\phi^T + \delta\chi^T \phi^T + N^T \delta\Gamma^T) \\ &= \delta\phi\chi\chi^T \delta\phi^T + \phi\delta\chi\delta\chi^T \phi^T + \delta\Gamma N N^T \delta\Gamma^T \\ &\quad + \delta\phi\chi\delta\chi^T \phi^T + \phi\delta\chi N^T \delta\Gamma^T \\ &\quad + \phi\delta\chi\chi^T \delta\phi^T + \phi\delta\chi N^T \delta\Gamma^T \\ &\quad + \delta\Gamma N \chi^T \delta\phi^T + \delta\Gamma N \delta\chi^T \phi^T \end{aligned}$$

Taking expected values of both sides, many terms are zero yielding

$$\begin{aligned} \underline{\delta\bar{X}}_{k+1} &= \Phi \underline{\delta\bar{X}}_k \Phi^T + \underset{\psi}{\mathbb{E}} [\delta\phi \underline{\bar{X}}_k \delta\phi^T + \delta\Gamma \Theta \delta\Gamma^T] \\ &\quad + \underset{\eta, \psi}{\mathbb{E}} [\delta\phi \chi_k \delta\chi_k^T \phi^T + \phi \delta\chi_k \chi_k^T \delta\phi^T] \end{aligned}$$

to solve this equation, we introduce the intermediate variable

$$Y_k = \underset{\eta}{\mathbb{E}} \chi^{\text{nom}} \delta\chi_k^T$$

so that

$$\phi \frac{\delta \bar{X}}{\delta \psi} \phi^T - \frac{\delta \bar{X}}{\delta \psi} = -\epsilon [\delta \phi \bar{X} \delta \phi^T + \delta \Gamma \Theta \delta \Gamma^T + \delta \phi Y \phi^T + \phi Y^T \delta \phi^T]$$

To solve for  $Y$ , we again write the equations for  $\chi$  and  $\delta \chi$  and take expected values of both sides w.r.t.  $\eta$  to get

$$\phi Y \phi^T - Y = -\phi \bar{X} \delta \phi^T - \Gamma \Theta \delta \Gamma^T$$

Assuming  $\bar{\Psi} = \epsilon (\psi - \bar{\psi})(\psi - \bar{\psi})^T$  is a diagonal matrix, we can now write the solution as:

Choose  $K, C, \phi, \Gamma$ , and  $H$  to minimize  $J$ , where

$$\phi \bar{X} \phi^T - \bar{X} = -\Gamma \Theta \Gamma^T$$

$$\phi Y \phi^T - Y = -\phi \bar{X} \delta \phi^T - \Gamma \Theta \delta \Gamma^T$$

$$\phi \frac{\delta \bar{X}}{\delta \psi} \phi^T - \frac{\delta \bar{X}}{\delta \psi} = -\sum_i \left[ \frac{\partial \phi}{\partial \psi_i} \bar{X} \left( \frac{\partial \phi}{\partial \psi_i} \right)^T + \left( \frac{\partial \Gamma}{\partial \psi_i} \right) \Theta \left( \frac{\partial \Gamma}{\partial \psi_i} \right)^T + \left( \frac{\partial \phi}{\partial \psi_i} \right) Y \phi^T + \phi Y^T \left( \frac{\partial \phi}{\partial \psi_i} \right)^T \right] \bar{\Psi}_{ii}$$

$$J = \text{trace}[(A + C^T B C) (\bar{X} + \delta \bar{X})]$$

These three equations can now be solved sequentially to yield  $J$ . We can then iterate, using a gradient search procedure, to find the minimal value of  $J$ .

The program implementing these equations will be described in a separate technical report.

#### VII.4 Augmented Observability and Controllability

##### Choice of A, B, and $\Gamma_{21}$

Claims appear in the literature [Katz, 1974] to the effect that the result of optimizing a quadratic performance index, for example

$$J = \sum_{i=0}^{N-1} x_i^T A x_i + u_i^T B u_i \quad (7.11)$$

always yields a stable closed-loop system. That this is not true for a large class of systems will be shown in the examples which follow.

If A of equation 7.11 is a symmetric, non-negative definite matrix, then we can decompose A into an  $m \times n$  vector d satisfying

$$A = d'd \quad (7.12)$$

where m is the rank of A. If we have a system of the form

$$\dot{x} = Ax + bu$$

A sufficient condition for ensuring stability of the closed loop system is for the related system

$$\begin{aligned} \dot{x} &= Fx + bu \\ y &= dx \end{aligned} \quad (7.13)$$

to be detectable. This is directly related to a further result-- that the performance index will be finite as long as the unstable modes of (7.13) are not observable--so a finite performance index does not ensure a stable optimal solution.

The dual result is that the Kalman filter will be stable if the system



$$\{[\Gamma_{21} \ Q \ \Gamma_{21}^T]^{\frac{1}{2}}, F\}$$

is detectable.

### VII.5 Application to Parameter Desensitization

Using exactly analogous arguments, we can see immediately that failing to meet the augmented observability and controllability conditions will have a profound effect on the performance of the desensitization algorithms. Given

$$J = \sum x^T A x + u^T B u$$

We redefine  $x^T A x = x^T d^T d x$

$$z = d x$$

If a state,  $x$ , is not observable through the observations  $z$ , then the uncertainty of that state will not be observed in  $J$ . So, the effect of varying parameters-- such as the gains  $K$  and  $C$ -- will not be seen in the performance index,  $J$ . We must also consider the effect of uncertain states on the estimator. It can be shown that minimizing the aforementioned performance index, without introducing sensitivity considerations, will yield the expected Kalman filter as an estimator. We therefore expect the efficacy of the desensitization to be effected by the controllability of the system  $\{F, (FQF^T)^{1/2}\}$ .

For example, consider a simple, first-order system with no process noise

$$\begin{aligned} x_{i+1} &= \alpha x_i + u_i + w_i & \& w_i w_j^T = 0 \\ y_i &= h x_i + v_i & \& v_i v_j^T = r \delta(i-j) \\ J &= \lim_{N \rightarrow \infty} \frac{1}{N} \sum_{i=0}^{N-1} x_i^T (A + C^T B C) x_i \end{aligned}$$

Clearly, this system is not controllable (via  $w$ ) in the augmented sense. If we now synthesize the optimal estimator

$$\hat{x}_{i+1} = (1-kh)\alpha\hat{x}_i + ky_{i+1} \quad .$$

We can write the augmented state equations

$$\begin{bmatrix} x_{i+1} \\ \hat{x}_{i+1} \end{bmatrix} = \begin{bmatrix} \alpha & -c \\ kh\alpha & (1-kh)(\alpha-c)-khc \end{bmatrix} \begin{bmatrix} x_i \\ \hat{x}_i \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & k \end{bmatrix} \begin{bmatrix} w_i \\ v_{i+1} \end{bmatrix}$$

$$x_{i+1} = \Phi x_i + \Gamma N_i$$

Consider the sensitivity equations,

$$\Phi \underline{X} \Phi^T = -\Gamma \Theta \Gamma^T$$

$$\Phi Y \Phi^T - Y = -\Phi \bar{X} \delta \Phi^T - \Gamma \Theta \delta \Gamma^T$$

$$\Phi \underline{\delta X} \Phi^T - \underline{\delta X} = -\Sigma \begin{bmatrix} \cdot & \cdot & \cdot \end{bmatrix} I \underline{\psi}_{1i}$$

$$J = \text{trace}[(A + C^T B C)(\bar{X} + \underline{\delta X})]$$

If we assume  $k$  equals zero, then  $\Gamma \Theta \Gamma^T$  is zero. It follows that as long as  $\Phi$  and  $A$  are positive definite,  $\bar{X} = 0$  solves the first Lyapunov equation. Similarly,  $\underline{Y} = 0$  solves the second Lyapunov equation, and  $\underline{\delta X} = 0$  solves the third! The conclusion, of course, is that  $J=0$ , for all choices of  $c$  yielding a stable system matrix  $F$ . And, since  $J$  must be non-negative, we have found an optimal choice for  $k$ .

Clearly, the limited control over the closed loop poles (via choosing the feedback gain  $c$ ) is intolerable. The important point for the moment, however, is that our uncertainty in the value of the system parameter  $\alpha$  has no effect on our choice of feedback gains--  $k$  or  $c$  -- as long as the system is not controllable in the broad sense.

Turning once again to the original dynamic equations, after linearizing, we had (7.10)

$$\begin{aligned} x_{i+1}^{\text{nom}} &= \phi x_i^{\text{nom}} + \Gamma N \\ \delta x_{i+1} &= \phi \delta x_i + \delta \phi x_i + \delta \Gamma N \end{aligned}$$

Assuming  $k=0$ , we have

$$\begin{aligned} x_{i+1}^{\text{nom}} &= \phi x_i^{\text{nom}} \\ \delta x_{i+1} &= \phi \delta x_i + \delta \phi x_i \end{aligned}$$

With stable dynamics matrices,  $x^{\text{nom}}$  is undriven, and will decay to zero. Therefore,  $\delta x$  will be undriven, and will also decay to zero. As long as  $x = x^{\text{nom}} + \delta x$  is undriven by stochastic disturbances, the state will decay to zero in steady state, and be unaffected by (small) errors in the dynamics matrix (assuming a stable dynamics matrix).

This is the same type of result we see with "oblivious" filters [Bryson, 1978]. With oblivious filters, if a mode is unexcited by disturbance noise, then the Kalman gain associated with that mode decays to zero. In our case, we find that if a mode is undisturbed, then it is ignored in the desensitization procedure!

Further, in the steady state, the impact of parameter uncertainty is determined both by the uncertainty and by the strength of the noise driving the mode. Lightly driven modes are lightly weighted in the performance index.

This explains the necessity for the scale factor  $H_{\text{dass}}$  introduced to boost the impact of the parameter uncertainty, and the necessity for using unrealistically large variances for parameters in some practical

examples.

We can gain additional insight into the nature of this problem by examining equation (7.10b)

$$\delta x_{k+1} = \delta \phi x_k^{\text{nom}} + \phi \delta x_k + \delta \Gamma_k^N + \delta \phi \delta x_k, \quad \delta x_0 = 0$$

We assumed in the derivation that the last term, to first order, can be neglected. This term provides the direct coupling from parameter uncertainties (in  $\delta \phi$ ) into state uncertainty (in  $\delta x_k$ ), and hence into the performance index,  $J$ . Therefore, neglecting this term has an unfortunate consequence, especially if the state is undisturbed as outlined above.

Unfortunately, with the elaboration of equation (7.10b), the expected value of  $\delta x$  with respect to  $\psi$

$$\frac{\partial}{\partial \psi} \delta x_k$$

is no longer zero. The formulation no longer follows directly as outlined, and an alternative formulation must be sought.

## VII.6 Applications

A computer program was written to solve the Lyapunov equations for minimizing sensitivity to parameter uncertainties. This program can handle both continuous and discrete systems.

### Program Verification:

We would expect discrete systems with relatively fast sample rates to display parameter sensitivities analogous to the corresponding continuous system. We therefore chose a continuous system originally investigated by Hadass [Hadass, 1974]. This choice allows us to verify both the continuous solution and the discrete solution generated by the computer program.

A description of the model can be found in Appendix C4. It was basically a fourth order system, with the governing equations:

$$I_1 \ddot{\Theta} = k\phi + w_1$$

$$I_2 (\ddot{\phi} + \ddot{\Theta}) = -k\phi + \dot{u} - w_1 + w_2$$

where the state vector is  $x^T = [\theta, \dot{\theta}, \phi, \dot{\phi}]$ , the input is  $u$ , and  $w_1$  and  $w_2$  are process noise sources. The output of the system is  $y = \Theta + v$ , where  $v$  is the measurement noise source. We use the data presented by Hadass on pages 55 and 56, except we need to use different noise power spectral densities (to match his results). Specifically:

$$Q_w = \begin{bmatrix} 6.6 \times 10^{-2} & 0 \\ 0 & 2.0 \times 10^{-2} \end{bmatrix} [\text{rad}^2 \text{sec}^{-3}]$$

$$r_v = 3.4 \times 10^{-6} \text{ rad}^2 \text{sec}$$

We assume all of the model parameters are known exactly, except for the spring stiffness,  $k$ . If we assume  $k$  has a nominal value of 25.0 and a covariance of 49, we would expect the program would give the results presented by Hadass on pages 99 through 105. This was the case.

Next, we derived the discrete equivalents of the continuous system, using the transformation:

$$\dot{x} = Fx + Gu + \Gamma w \quad (\text{continuous})$$

$$x_{n+1} = \phi x_n + \Gamma_1 u + \Gamma_2 w \quad (\text{discrete})$$

$$y = Hx + v$$

where we assumed

$$\phi = e^{FT}, \quad T \text{ is the sample interval}$$

$$\Gamma_1 = \int_0^T \phi(\tau) G d\tau, \quad \text{where } \phi(\tau) = e^{F\tau}$$

$$\Gamma_2 \approx \Gamma$$

The fastest roots of the physical system lie at 5 radians, so if we choose a sample interval of 25 milliseconds, or 40 samples per second, we expect the behavior of the continuous and discrete systems to be nearly identical.

The results of an optimization run are presented in Table 7.1 and Figure 7.1. The results can be compared along several dimensions.

First, over what range can we vary the spring stiffness and still retain stability? This is given by the terms  $\Delta K_+$  and  $\Delta K_-$ , where

Table 7.1  
RESULTS OF PROGRAM VERIFICATION

iteration	PARAM	CONTINUOUS	DISCRETE
0	$\Delta K$ -	0.344	0.344
	+	0.297	0.297
	J n	$1.02 \times 10^4$	$1.02 \times 10^4$
	$\Delta$	$1.19 \times 10^4$	$1.19 \times 10^4$
	T	$2.22 \times 10^4$	$2.22 \times 10^4$
	$\omega$	25.0	25.0
3	$\Delta K$ -	-	0.625
	+	-	0.828
	J n	$1.12 \times 10^4$	$1.15 \times 10^4$
	$\Delta$	$5.85 \times 10^3$	$4.07 \times 10^3$
	T	$1.71 \times 10^4$	$1.55 \times 10^4$
	$\omega$	30.2	25.0
15	$\Delta K$ - *	0.539	-
	+	1.132	-
	J n	$1.21 \times 10^4$	$1.16 \times 10^4$
	$\Delta$	$3.19 \times 10^3$	$3.42 \times 10^3$
	T	$1.53 \times 10^4$	$1.51 \times 10^4$
	$\omega$	26.4	26.7

$\Delta K$  - relative change of spring stiffness causing instability

- relative decrease

+ relative increase

J - performance index

n - nominal, no uncertainty

$\Delta$  - addition due to uncertainty

T - sum of nominal and  $\Delta$

$\omega$  - nominal spring stiffness coefficient used in estimator

\* - values of  $\Delta K$  quoted for 44 iterations



## Closed-Loop Pole Gravitation

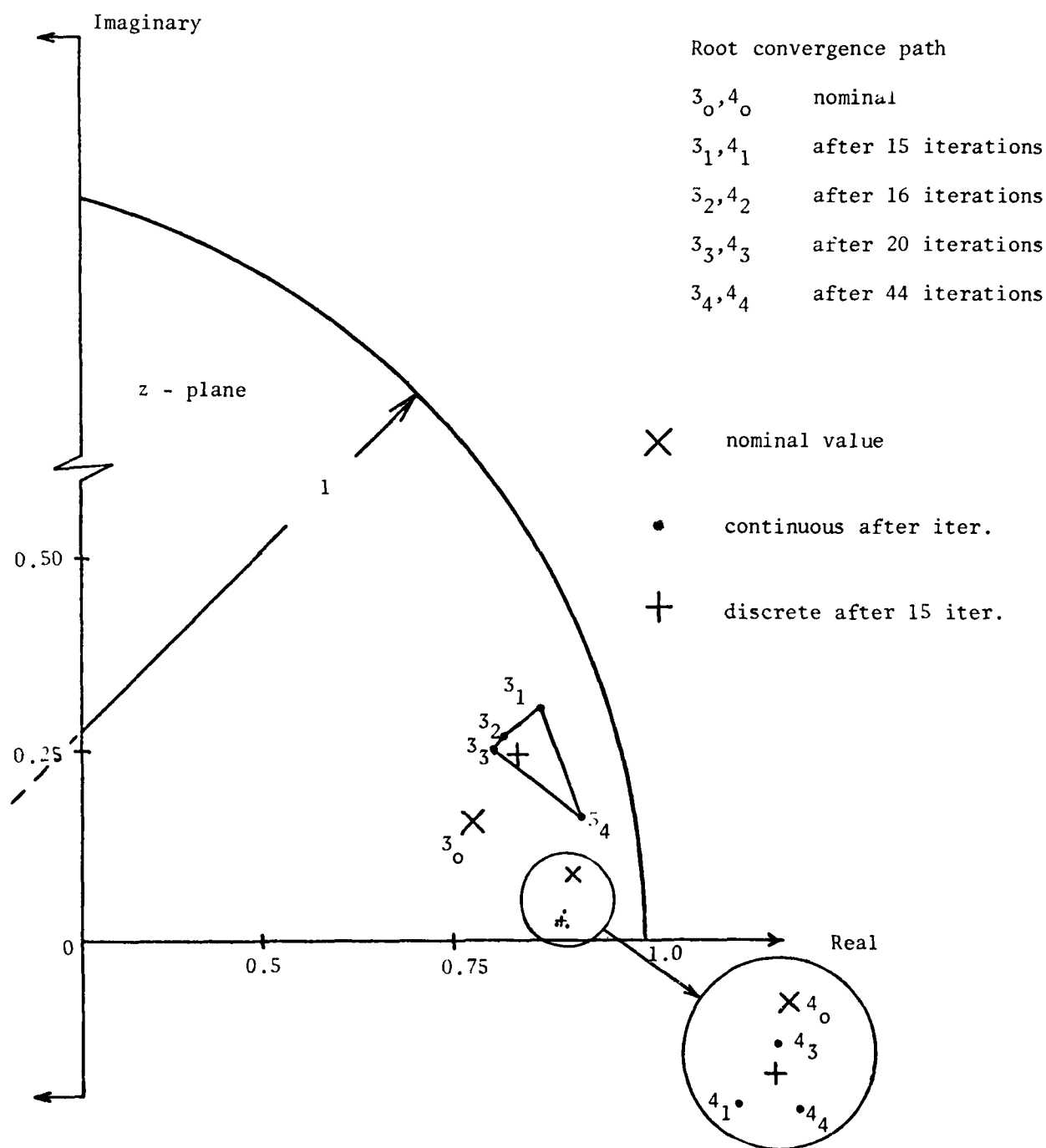


Figure 7.1 DESENSITIZATION LOCUS

$$\Delta k_+ = \frac{k_1 - k_n}{k_n}, \quad \Delta k_- = \frac{k_n - k_2}{k_n}$$

where  $k_1$  and  $k_2$  are the coefficients yielding marginal stability if we increase or decrease  $k$ , respectively. From Figure 7.1 the results are comparable.

Second, we would expect the performance index to closely match-- this is an excellent check of the two sets of Lyapunov equations. We have a nominal performance index,  $J_n$ , which is generated without considering parameter uncertainty; next, we have the added cost due to uncertainty,  $J_\Delta$ ; and finally, we have the total uncertainty,  $J_T$ . We note a good correspondence initially, before iteration began; there is also a good correspondence between the continuous and discrete results after fifteen iterations-- when improvement in the index had effectively stopped.

The final analytic criterion for comparison involves the coefficient  $w$ ; since  $k$  was an uncertain parameter, we chose to optimize the estimator by varying the Kalman gains and by varying the assumed value of the spring stiffness in the estimator. The result of the freedom was an increase in the assumed stiffness-- from 25.0 to over 26. The correspondence is again good between the two cases.

Next we look at the pole map in Figure 7.1. This diagram plots the relevant closed loop pole locations as a function of program iterations. The plane we plotted is the discrete or  $z$ -plane; stable roots have a magnitude of less than 1, lying within the unit circle. We have plotted the result of the discrete solution directly, and the results of the continuous solution using the mapping

$$z = e^{sT}$$

where  $z$  is the discrete pole location,  $s$  is the continuous pole location, and  $T$  is the sample interval.

We begin by noting that even when the performance index has converged to its terminal value-- which occurs at approximately fifteen iterations-- the pole locations for this value of the index are only loosely fixed. In other words, there are a wide choice of filter gains and closed loop pole locations which will yield the same value for the total performance index. This behavior is expected when using a gradient search procedure; the final value depends upon the starting conditions and the number of iterations performed. The results, however, are all qualitatively identical.

We note that the discrete solution is bracketed by the continuous solutions. We therefore conclude that all tests comparing the discrete and continuous cases have yielded the expected results.

#### An Example

As a practical example of reducing sensitivity, we investigated the test case presented in Appendix C1. This example is a hypothetical high-performance aircraft, henceforth termed the FH, modelling the longitudinal short period mode and the first bending mode. In addition, wind gusts were modelled as a first order Gauss Markov process. A discrete compensator was then derived.

Relevant specifications are that we have a fifth order system with the marginally stable modes (the bending modes) and the short period modes. The details of

the model derivation are somewhat involved; they are given in the appendix. The short period poles are located at  $-2.5 \pm 13.6j$ ; the bending mode poles are located at  $-0.5 \pm 25.0j$ .

Katz [1974], investigated the sensitivity of an optimal discrete compensator for this example, and offered ad hoc design modifications for reducing sensitivity. We intended to begin with his initial design, and to then use the desensitization procedures to systematically reduce the sensitivity. Presumably we could improve his results, and gain insight into his ad hoc desensitization.

The latter objective-- of gaining insight-- proved to be our most valuable contribution, and partially obviated our other goals. Specifically, Katz's initial design is intrinsically an invalid design for achieving reasonable parameter sensitivity.

Katz initially chose to work with a physical model where the bending modes were uncoupled from the short period modes, and he further chose a cost function weighting matrix which ignored these higher frequency modes. Further, these bending modes were only lightly driven by the process noise. As a result, we were synthesizing compensation for a system which was unobservable and only marginally controllable (in the augmented sense defined above).

Since we started with a highly sensitive system (bending mode poles at  $-0.5 \pm 25j$ , with a locus quickly crossing the axis), we would not expect good compensation to result from ignoring these modes-- as Katz found. His solution was to weight the bending mode states in his cost function and to inject process noise into the dynamics for these states (pages 103-109). This apparently ad hoc approach, however, can be

clearly understood as soon as it is interpreted in terms of augmented controllability and observability.

The added process noise introduces a significant covariance, deriving from the bending mode states, and since they are weighted, they significantly affect the performance index. The resulting design is significantly less sensitive, as Katz demonstrated and as theory predicts.

From our earlier theoretical work, it is now clear why Katz's initial design is inadequate for our program-- our algorithms will not affect unobservable or uncontrollable modes (exactly those modes which in this case introduce the sensitivity problem). Therefore, we abandoned our goal of beginning with Katz's original design. Instead, we set a lesser goal. Beginning with the final results for the FH aircraft, we attempted to improve upon them.

The example we chose was the FH aircraft flying at Mach 1.2 at ground level. In the model we assumed light coupling from the bending mode back to the short period mode, a heavy injection of process noise into the bending mode states, and a light weighting of the bending mode states. Selecting a sampling rate of 10 Hertz, we began from Katz's least sensitive design.

Figure 7.2 shows a root locus of the closed loop bending mode poles versus the uncertain parameter-- the bending mode frequency. Before optimization, the stability range as a function of  $\omega_B$  is limited to a 16% decrease. After optimization, the permissible change in  $\omega_B$  is a 24% decrease. Optimization has increased the stability range by 50%.

FH Aircraft-- Mach 1.2 at Altitude = 0  
 Locus of Bending Mode Poles versus  $W_B$  Given  
 Compensation Designed for  $W_B = 25$  radians

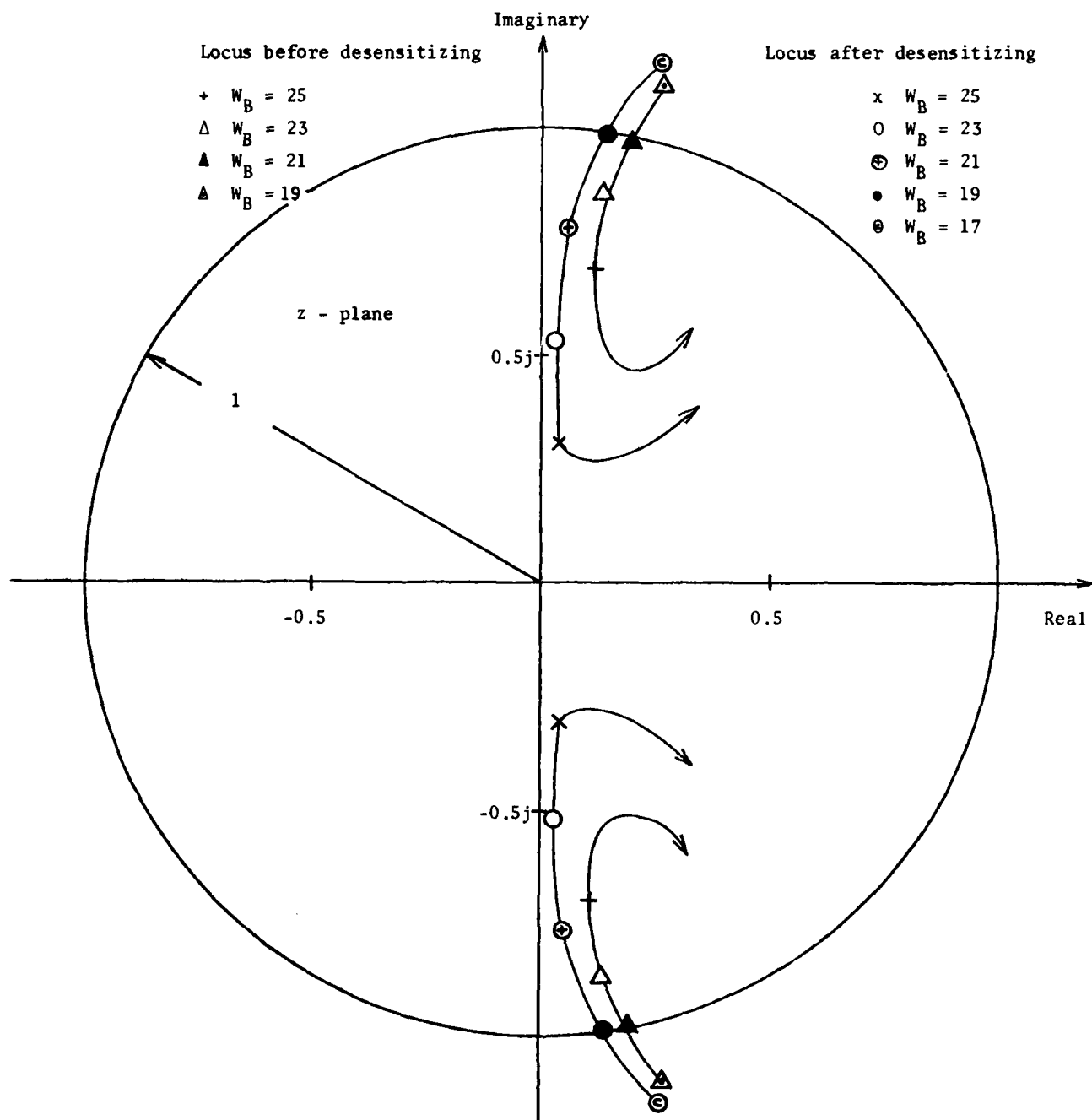


Figure 7.2 AIRCRAFT PARAMETER UNCERTAINTY DESENSITIZATION

Cost

This program is slow, complex, large, and therefore expensive. Each "run" on a tenth order problem (fifth order model and fifth order compensator) cost ten to twenty dollars on an IBM 370. The program runs in  $O(n^4)$  time.

Conclusions and Comments

An algorithm for desensitizing discrete closed loop systems given parameter uncertainty was presented. The importance of augmented controllability and observability in this procedure was noted. The algorithm's sensitivity to changes in magnitude of disturbance noise was noted and several other anomalies were explained; in particular, we demonstrated that neglecting second-order terms is not always possible.

The observations on augmented controllability were extended to explain other ad hoc desensitization procedures.

Stability margin improvement was demonstrated for a practical problem.

The procedure is comparatively expensive.

## Chapter VIII

### REDUCED-ORDER COMPENSATORS

In many applications-- including the design of compensators intended for implementation in digital processors-- the complexity of the compensator is a significant issue. Typical state-space design algorithms yield compensators whose complexity is comparable to the complexity of the original model for the physical system. If a reduced-order compensator is desired, the designer usually begins by trying to simplify the physical model.

We will consider an alternate approach, suggested by J.D. Powell [1976], which does not require simplifying the original model. Using the program mentioned in Chapter VII, we will consider designing compensators of arbitrary order. The design criterion will again be to minimize the expected variance of the states of the physical system.

First, we will gain insight into the problem by working with a simple two-body problem. Then we will consider the result of designing reduced-order compensation for a seventh-order star-tracking telescope.



### VIII.2 Design of Reduced-Order Compensation

We wish to consider techniques for designing reduced-order feedback using modern control procedures. We begin with a simple two-body problem, and use frequency-domain analysis to evaluate performance.

Three design techniques will be compared.

#### 1) Classical Technique

A classical design of comparable order provides insight and can be used to qualitatively evaluate the modern design technique.

#### 2) Engineering Judgment Technique

The system being modelled is simplified to the desired complexity using engineering judgment. A compensating network is designed for this simplified system, and then the compensator is analyzed for performance in the original system. The technique is iterated until satisfactory performance is obtained.

#### 3) Optimal Technique

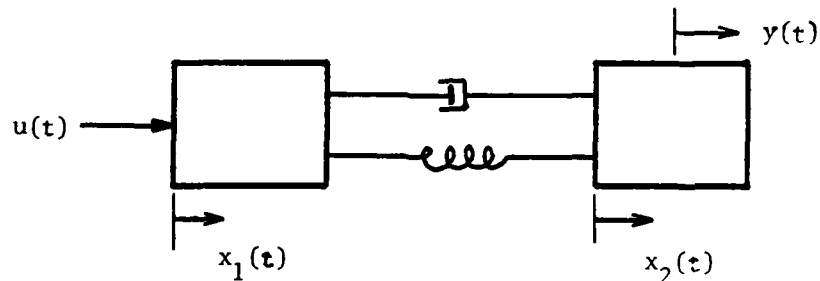
The compensating network is designed to optimize the performance of the original system, within the constraints imposed by limiting the order. As in Chapter VII, the original system will be augmented with a compensating network, and then a gradient search procedure will be used to choose the parameters of the compensator to optimize performance.

For the example considered, the three procedures are equivalent for low bandwidth systems. For high bandwidth systems, the third method yields designs which are stable, with maximal bandwidth and margin. The second

method, in general, yields unstable results, and the selection of some design parameters (e.g., weighting functions) becomes counter intuitive.

Problem:

The example system considered was the fourth-order two-body problem:



which we write as

$$\dot{X} = FX + Gu + \Gamma w$$

$$y = Hx$$

$$\frac{d}{dt} \begin{bmatrix} x_1 \\ \dot{x}_1 \\ x_2 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -K^2 & -b & K^2 & b \\ 0 & 0 & 0 & 1 \\ K^2 & b & -K^2 & -b \end{bmatrix} \begin{bmatrix} x_1 \\ \dot{x}_1 \\ x_2 \\ \dot{x}_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} u + \Gamma w$$

$$y = [0 \quad 0 \quad 1 \quad 0]x + v$$

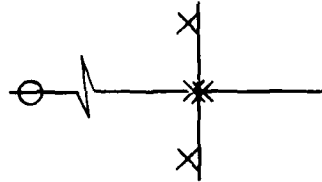
choosing  $M_1 = M_2 = 1$

$$K = 5$$

$$b = .01$$

we have the transfer function

$$G(s) = \frac{0.01(s+2500)}{s^2(s+.01*7.07j)}$$



a system with two poles at the origin, two lightly damped poles at 7 radian/sec, and a zero near infinity.

Figure 8.1 presents the open loop Bode plot for the full fourth order system. We note that for appropriately slow systems we can approximate the system transfer function as:

$$G'(s) = \frac{0.51}{s^2}$$

Our objective will be to design a second order compensator for  $G(s)$  which maximizes the system bandwidth while maintaining satisfactory phase and gain margins.

#### Classical Design-- pole placement

Assuming a second order system,  $G'(s)$ , an optimal design using any of the algorithms mentioned in Chapter IV will yield a second-order compensator with a single zero:

$$H(s) = \frac{K(s+\alpha)}{(s+\beta)(s+\gamma)}$$

Since the objective in executing a classical design is to gain insight into the problem, we simplify the design by limiting ourselves to real poles; we must therefore determine four real numbers--  $K$ ,  $\alpha$ ,  $\beta$ ,  $\gamma$  -- to yield  $H(s)$ .

Considering the Bode plot for  $G(s)$  [Figure 8.1] note that the phase of  $G$  is always less than  $-180^\circ$ . Therefore  $H(s)$  must provide

|G| Uncompensated, dB

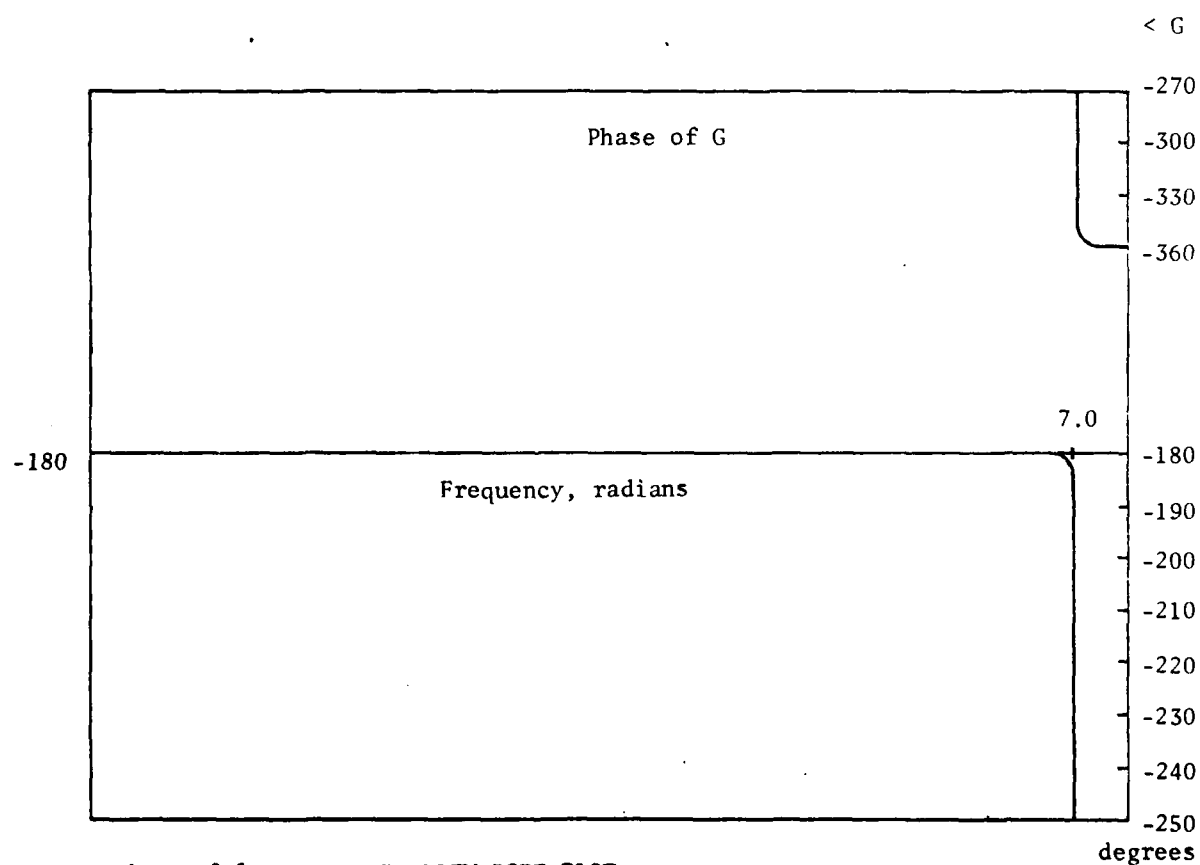
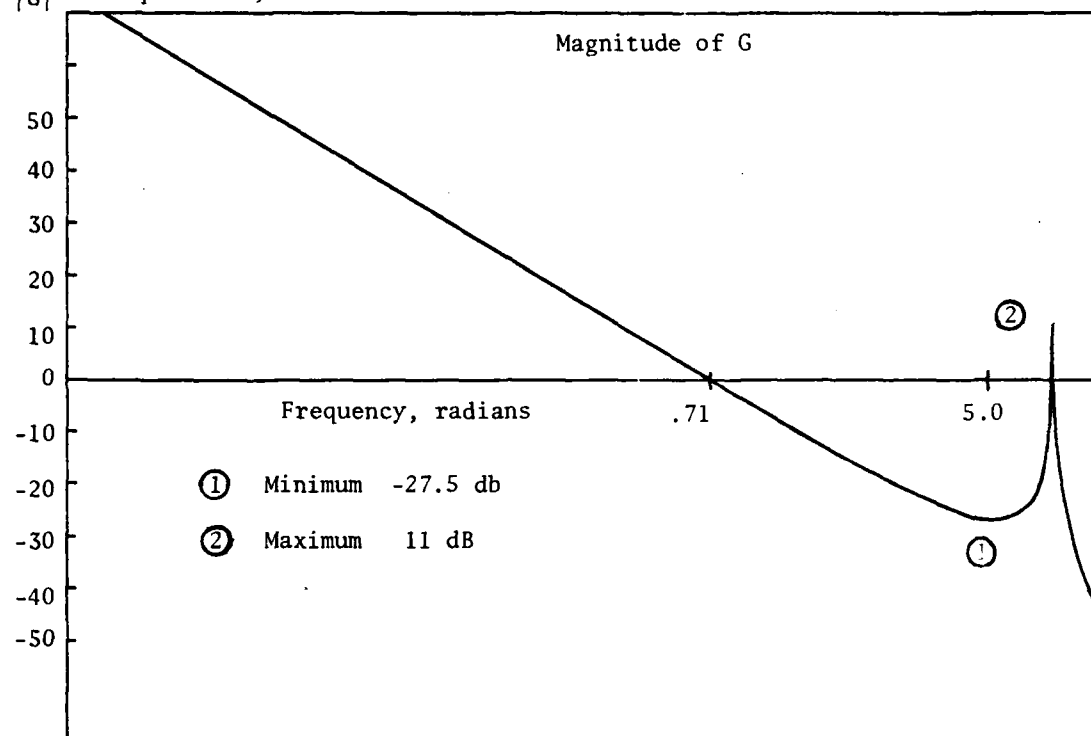


Figure 8.1 SAMPLE PROBLEM BODE PLOT

phase lead.  $H(s)$  should also provide adequate margins and a maximum bandwidth. Using the open and closed-loop Bode plots and the Nyquist plot for  $G(s) \cdot H(s)$  [Figure 8.2] we can determine reasonable bounds on these margins and on the bandwidth.

### Stability

Either the crossover through  $-180^\circ$  degrees must occur before "C" or after the magnitude peak at 7 radians. Since there is a phase change of  $-180^\circ$  at 7.07 radians, only the first alternative is viable; the phase must be less than  $-180^\circ$  at approximately 6 radians/second.

### Phase Margin

The phase margin will be the minimum of the phase angles at "A" and "C" radians. Thus, we want to maximize the height of the phase peak and roll off the phase curve as quickly as possible beyond "B" radians/second. The minimum of "A" and "C" is approximately  $30^\circ$ .

### Bandwidth

The bandwidth is the frequency range over which the output of the closed loop system follows the input with less than a 3db magnitude deviation. Looking at the closed loop bode plot, this deviation can occur at either "a," "b," or "c" radians. If the magnitude of  $G \cdot H$  is greater than -7 db at point "B," then "c" determines the bandwidth. If the magnitude of  $G \cdot H$  is less than -7 db, then the magnitude at "b" radians will be less than -3db. This gives two bandwidth ranges:

- 1) in the range of 5 to 7 radians (point "c"),
- 2) in the range of less than 4 radians (point "b").

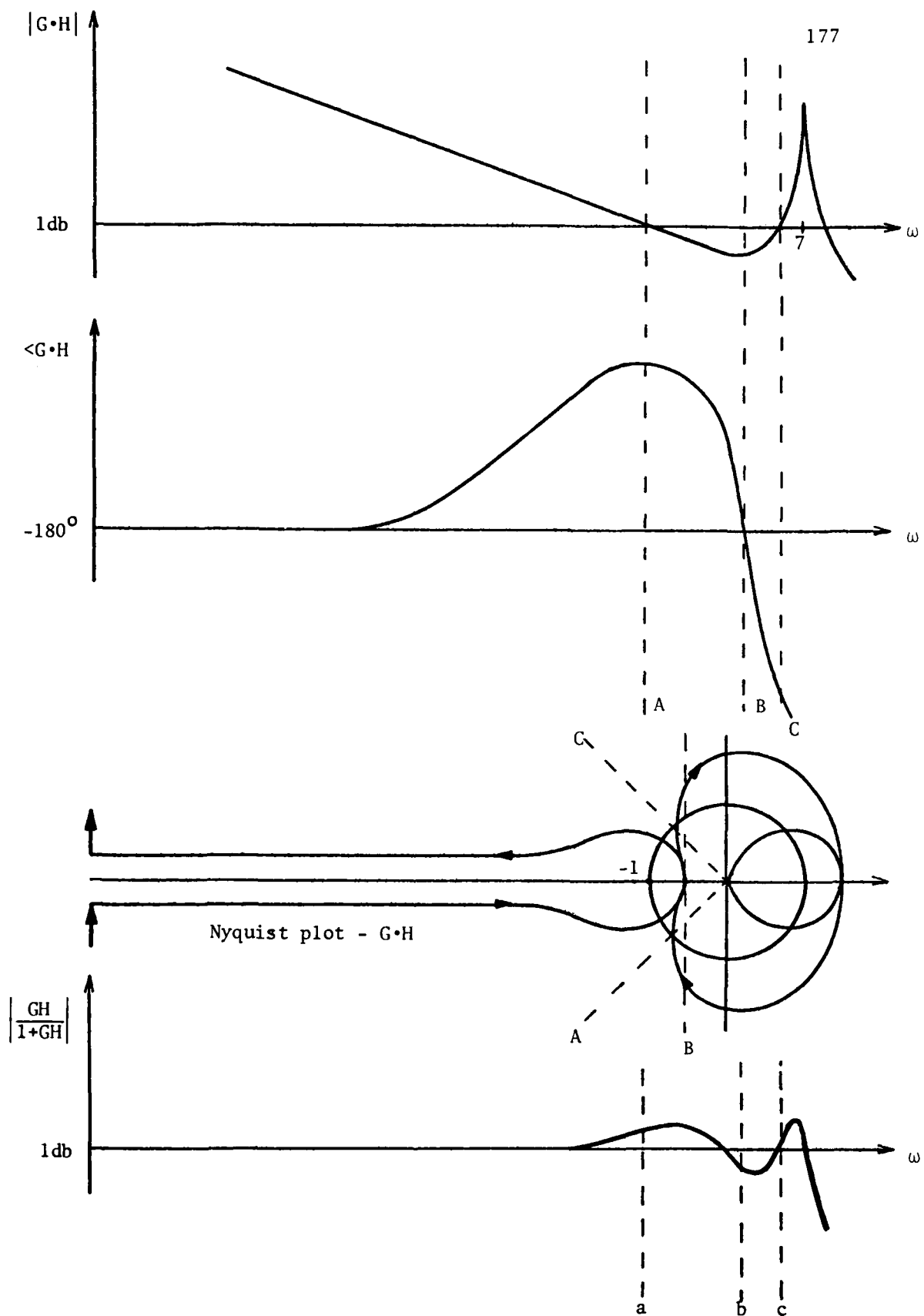


Figure 8.2 CLASSICAL DESIGN PLOTS

Gain Margin

The gain margin is the reciprocal of the magnitude at point "B" ; decreasing the magnitude at "B" increases the gain margin. Thus, gain margin and bandwidth must be traded against one another. For a bandwidth less than 1 radian any gain margin can be obtained. For a bandwidth of more than 5 radians, the gain margin must be less than 2.3.

Conclusion

Our objective is to design a high-bandwidth system. We therefore expect to attain a bandwidth of about 6 radians, a gain margin of about 2, and a phase margin of about  $30^\circ$ .

Examples

$$\text{Choosing } H_1(s) = \frac{100(s+.2)}{(s+5)^2}$$

Phase margin =  $18^\circ$  (determined by "C")

Gain margin = 2.2

Band width = 6.3 radians (determined by "c")

Trying to improve the phase margin, we can increase the phase rolloff, but we lose bandwidth

$$H_2(s) = \frac{50(s+.1)}{(s+4)^2}$$

Phase margin =  $30^\circ$  (Phase at "A" = Phase at "C")

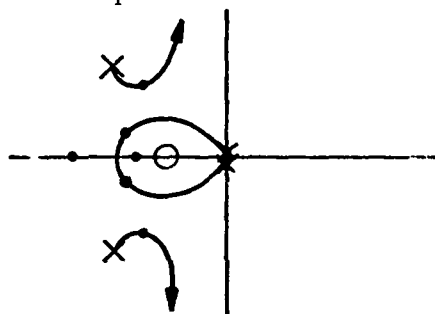
Gain margin = 3.2

Band width = 3 radians (Determined by "b")

### Optimal Design-- with System Simplification

If we ignore the higher order dynamics of the original system, we can design a full-order compensator for  $G'(s) = 0.5/s^2$  using optimal methods. We can then insert this compensator into the full fourth-order system and examine the results.

The resulting second-order compensators are characteristically similar to the previous optimal designs; the closed-loop second-order system and second-order compensator have a similar root locus:



When this compensator is then introduced into the full-order system, we get a reasonable design providing the system bandwidth is less than 10% of the resonant frequency. For larger bandwidths, however, the high frequency poles move into the right half plane, and the system is unstable. (Similar results can be seen by evaluating the appropriate Nyquist diagram; the magnitude of  $G \cdot H$  is greater than 1 db when the phase passes through  $-180^\circ$ .)

Clearly, this approach fails because relevant information is not retained in the design synthesis. This information can be included if we begin our system simplification by assuming we have perfect measurements, which in our example means we can measure state  $x_3$  directly, so a full state optimal compensator would have a third order numerator and denominator. Again, it is unclear in general how to then go beyond the first-order reduction unless:



- 1) Poles and zeroes happen to cancel in the compensator, or
- 2) we throw information away (by ignoring higher order dynamics, etc.).

### Optimal Design-- Gradient Search

Given

$$\dot{x} = Fx + Gu + \Gamma w \quad (8.1)$$

$$y = Hx + v \quad (8.2)$$

where  $F$  is of order  $\underline{n}$ , we want to choose  $f$ ,  $g$ ,  $h$ ,  $k$  and  $C$  to minimize the performance index

$$J = \lim_{t_f \rightarrow \infty} \frac{1}{t_f - t_0} E \int_{t_1}^{t_f} (x^T A x + u^T B u) dt \quad (8.3)$$

where we have

$$u = -C\hat{x} \quad (8.4)$$

$$\dot{\hat{x}} = f \hat{x} + gu + K(y - h\hat{x}) \quad (8.5)$$

The normal solution to this problem would assume  $f = F$ ,  $g = G$ , and  $h = H$ , then choose  $K$  and  $C$  to minimize  $J$ .

If we now write  $\bar{X} = \begin{bmatrix} x \\ \hat{x} \end{bmatrix}$  equations (1), (2), (4), and (5) can be combined to give the full-order, combined state-estimator and compensator:

$$\dot{\bar{X}}_1 = \begin{bmatrix} F & -GC \\ KH & F-KH-GC \end{bmatrix} \bar{X}_1 + \begin{bmatrix} \Gamma & 0 \\ 0 & K \end{bmatrix} \begin{bmatrix} w \\ v \end{bmatrix}$$

$$u = [0 \quad -C] \bar{X}_1$$

and we can use expression (3) to solve iteratively for  $K$  and  $C$ . Similarly,  $\hat{x}$  is assumed to be of order  $m$  less than  $n$ . This yields the augmented system of order  $(n+m)$ :

$$\dot{\bar{X}}_2 = \begin{bmatrix} F & -GC \\ KH & f-gC-Kh \end{bmatrix} \bar{X}_2 + \begin{bmatrix} \Gamma & 0 \\ 0 & K \end{bmatrix} \begin{bmatrix} w \\ v \end{bmatrix}$$

Again, the quadratic performance index (3) and a gradient search procedure are used to choose optimal  $K$  and  $C$  for the given  $\bar{X}_2$ .

#### Range of Results

Note first that all four states of the original system can be independently weighted (using the  $A$  matrix) and that noise can be injected into each state. Therefore, there are many more independent design parameters (weighting functions, etc.) than dependent parameters; in this example, there are only four degrees of freedom in  $H(s)$ , but none of the entries in  $K$ ,  $C$ ,  $f$ ,  $g$  or  $h$  have been specified.

By proper selection of  $f$ ,  $K$ , and  $C$ , we can realize any transfer function  $H(s)$ . Specifically, the choice of  $f$  determines the zeroes of  $H(s)$ ; the choice of  $K$  and  $C$  determine the poles of  $H(s)$ . Therefore, the gradient search technique can explore the entire range of desired transfer functions.

### Rate of Convergence

The technique converges to a solution in twenty to thirty iterations. The entire run, for this example, required approximately four times as much computer time as one run of an Eigenvector Decomposition routine for the same example.

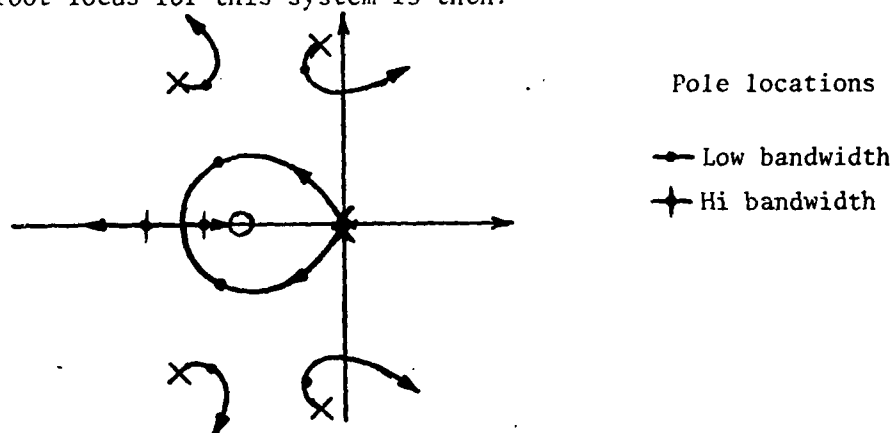
Convergence is always faster if the search has more parameters than degrees of freedom. For example, convergence is faster when  $K$ ,  $C$ , and  $f$  are allowed to vary. The same transfer function for  $H(s)$  results.

### Examples

The results shown in Table 8.1 represent a large range of possible designs, with bandwidths ranging from 1.5 radians/sec. to 5.9 radians/sec. All systems are stable, with a phase margin of  $22^\circ$  and a gain margin of about 2. The resulting compensator is approximately--

$$H(s) \approx \frac{100 (s + .5)}{(s+4+4j)(s+4-4j)}$$

The root locus for this system is then:



The optimal design is conceptually a refinement of the classical design, and reasonably attains the performance limits for bandwidth and margin.

Table 8.1  
SUMMARY OF DESIGN TRIALS

Explanatory Notes

PARAMS--

$\Theta(a,b)$  - Steady-state Design

$a = Q_2/R_1$  , noise ratio , ( $Q_1 = 0$ )

$b = A/B$  , weighting ratio, ( $A_1 = A_2$ )

$P(a,b)$  = Gradient Search Design

T     --     Run Time (fraction of a minute)

(XX) - number of degrees of freedom equals number  
of parameters

XX - excess number of free parameters

BW     --     Band width (radians)

PM     --     Phase Margin

GM     --     Gain Margin

H(s)   --     Compensator

PARAMS	T	BW	PM	GM	H(s)	Closed-Loop Poles	
						Second Order	Fourth Order
$\Theta(1,1)$		-0.85	35°	2.2	$\frac{3.13(s+.32)}{(s+1.57*1.4j)}$	-0.87 * 0.5j -0.707 * 0.707j	-0.03 * 6.8j -0.35 * 0.522j -4.46 * 4.77j
$P(1,1)$	(.71) .55	-1.45	32°	2	$\frac{3.87(s+.29)}{(s+1.144*1.37j)}$		-0.28 * 7.0j -0.45 * 0.43j -0.677 * 1.05j
$\Theta(10^3,1)$		-1.3	5°	2	$\frac{62.4(s+.515)}{(s+4.84*479j)}$	-0.86 * 0.5j -3.98 * 3.98j	-0.03 * 6.8j -0.35 * 0.5j -4.5 * 4.8j
$P(10^3,1)$	(.4) .3	-2.6	~22°	~3	$\frac{66.3(s+.435)}{(s+4.05*3.66j)}$		-0.13 * 6.8j -0.71 * 0.45j -3.2 * 3.4j
$\Theta(10^3,10)$	.1	Unstable	-10°	.193	$\frac{153(s+.654)}{(s+6*5.56)}$	-1.06 -2.98 -3.98 * 3.98j	+0.11 * 6.57j -0.64 * 0.73j -5.48 * 5.63j
$P(10^3,10)$	(.38) .39	+5.9	22°	2	$\frac{97(s+.569)}{(s+3.88*3.6j)}$		-0.26 * 6.6j -2.17 * 3.37j -1.83 , -1.08
$\Theta(10^3,10^2)$		Unstable	-30°	.032	$\frac{432(s+.75)}{(s+9.5*6.3j)}$	-1.005 -0.995 -3.98 * 3.98j	0.45 * 6.45j -1.0 * 0.811j -8.9 * 6.2j
$P(10^3,10^2)$	.39	+6.9	22°	1.6	$\frac{114.3(s+.628)}{(s+3.78*3.58j)}$		-0.3 * 6.4j -1.5 * 3.6j -2.7 -1.0

Table 8.1 (cont) SUMMARY OF DESIGN TRIALS

### VIII.3 Comparison -- Gradient Search and "Simplification" Design Choice of Independent Design Parameters

A reasonable question to ask is whether an appropriate choice of weighting functions for the "Simplification" design would have yielded the transfer functions found using the gradient search. The answer is no (see Table 8.2). Depending upon the compensator being matched, we would need to specify either a negative weighting matrix,  $A$ , or a negative covariance. Thus, using simplified models and steady-state Riccati analysis, we cannot realize all possible compensators. However, if we use this same model,  $G'(s)$ , as the basis for the estimator in the gradient search procedure:

$$G'(s) \Leftrightarrow \dot{\hat{x}} = f \hat{x} + gu + K(y - h\hat{x})$$

then all reduced-order transfer functions are realizable.

#### Evaluation in terms of Frequency Domain Criteria

Comparing the two 'optimal' procedures in the frequency domain (Figure 8.3 through Figure 8.5) we see that without introducing the added information about the resonant peak at 7 radians, the phase of the compensator is inadequately constrained to guarantee stability. The "Simplification" design is inadequate.

#### Conclusions

- 1) Both techniques are comparable for low bandwidth systems.
- 2) The gradient search procedure insures stable systems.
- 3) The gradient search procedure requires more computer time.

We are comparing a sixth order calculation (a fourth order

Table 8.2

## SIMPLIFICATION EQUIVALENCE

A compensator equivalent to the gradient design,  $P(a,b)$ , could be realized using simplification if the design parameters are chosen correctly. For the parameters shown,  $\Theta(a,b)$ , the appropriate design parameters are given.

PARAMS	$Q_1/R$	$Q_2/R$	$A_1/B$	$A_2/B$
$\Theta(10^3, 10)$	0	$10^3$	10	10
$\bar{\Theta}(1, 1)$	2.5	0.42	3.5	-3.6
$\bar{\Theta}(10^3, 1)$	-13.7	140	6	20.1
$\bar{\Theta}(10^3, 10)$	-27.1	216	14	31.2

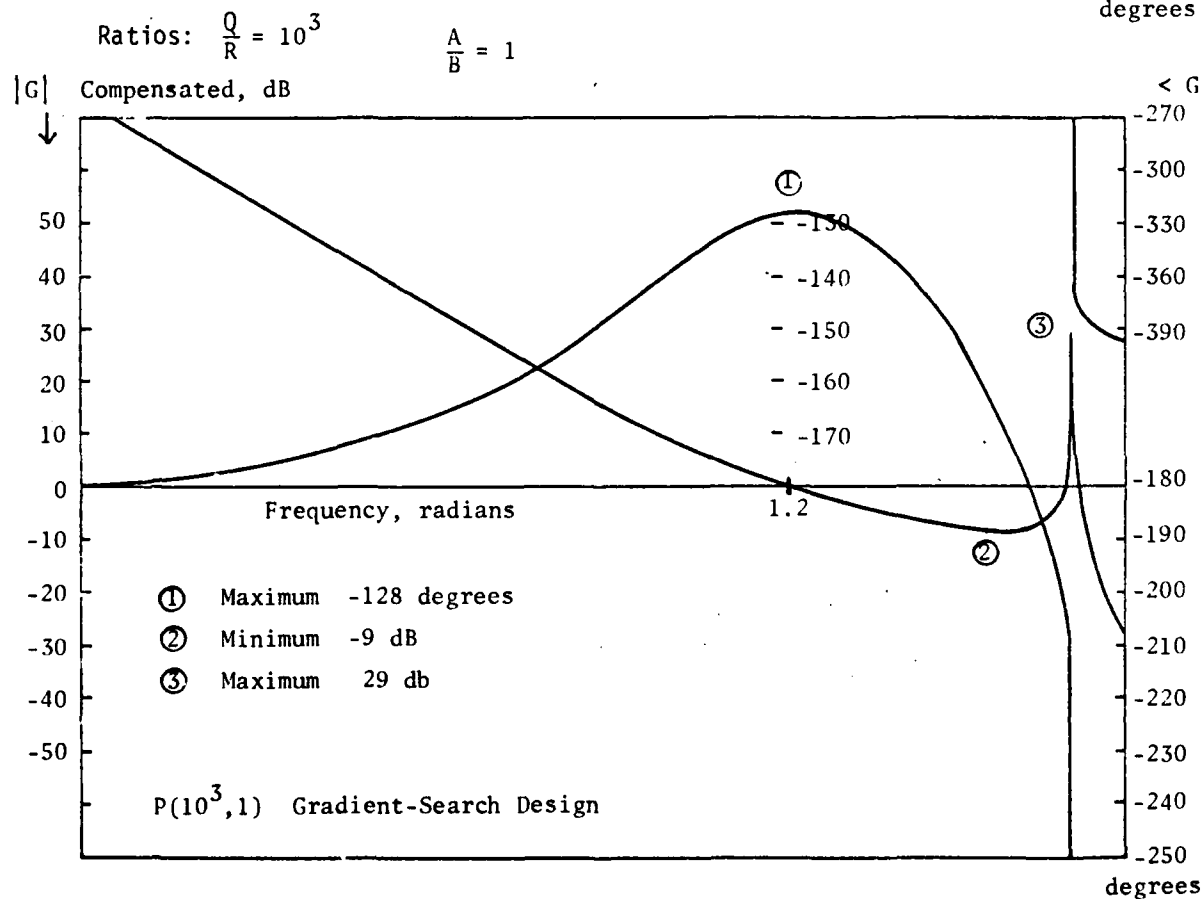
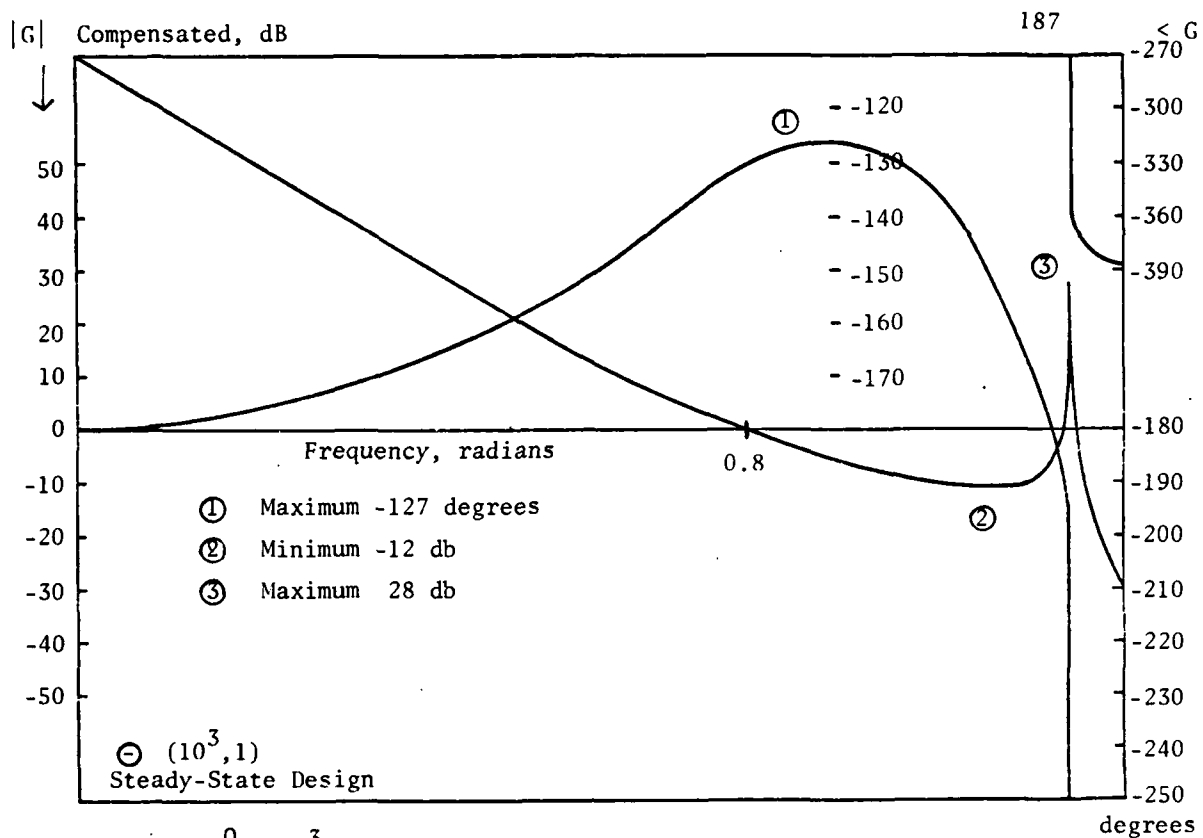
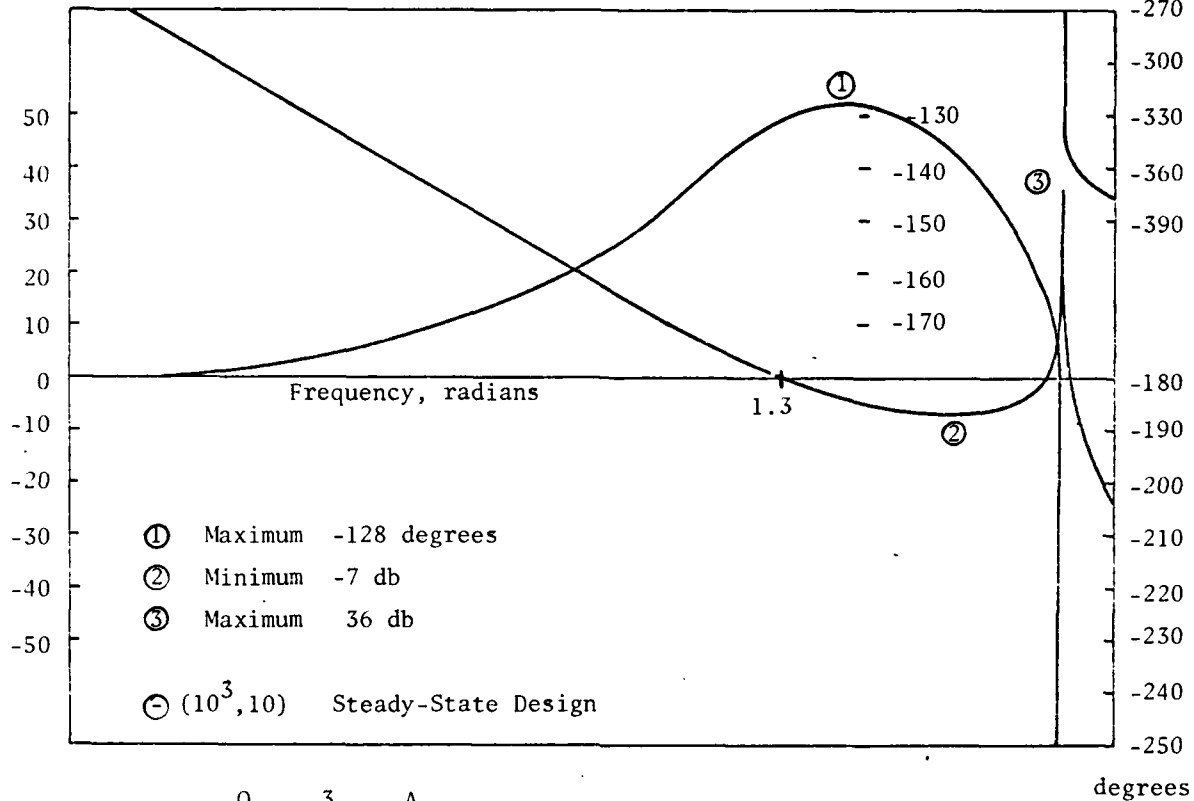


Figure 8.3 'OPTIMAL' DESIGN COMPARISON, TRIAL I BODE PLOTS



$|G|$  Compensated, dB

Ratios:  $\frac{Q}{R} = 10^3$   $\frac{A}{B} = 10$

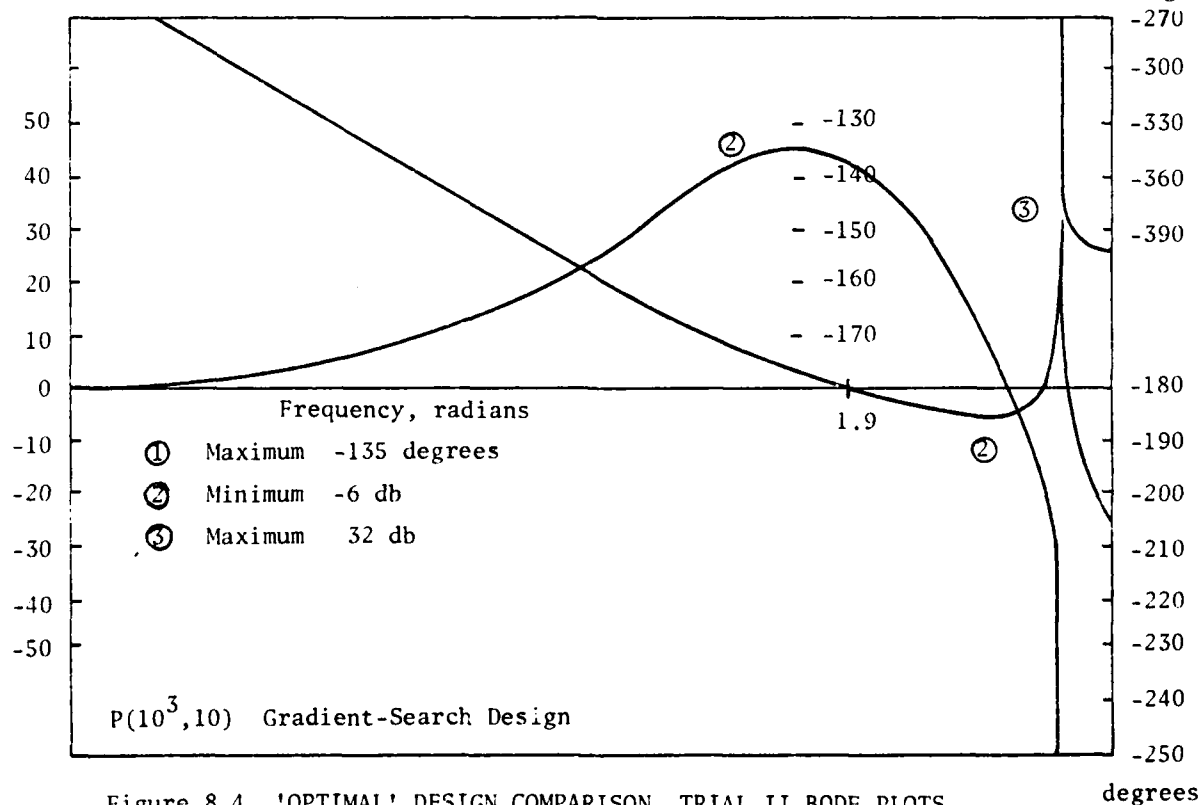
 $|G|$  Compensated, dB

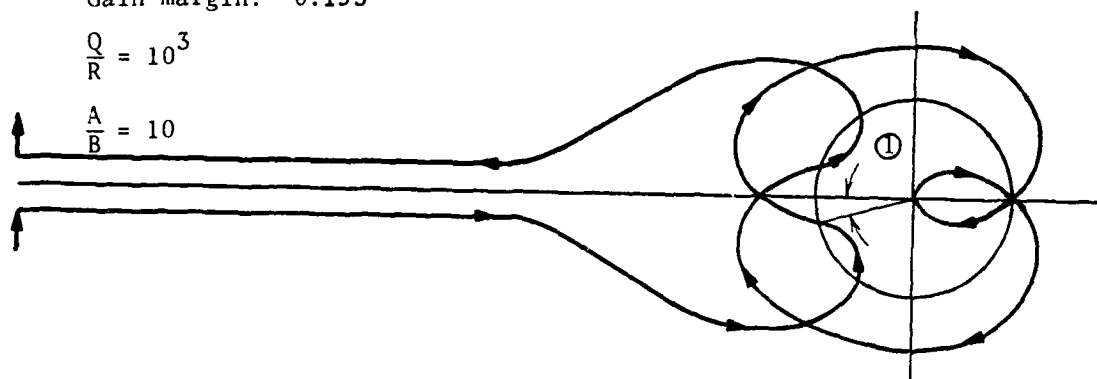
Figure 8.4 'OPTIMAL' DESIGN COMPARISON, TRIAL II BODE PLOTS

① Phase margin:  $-10^\circ$

Gain margin: 0.193

$$\frac{Q}{R} = 10^3$$

$$\frac{A}{B} = 10$$



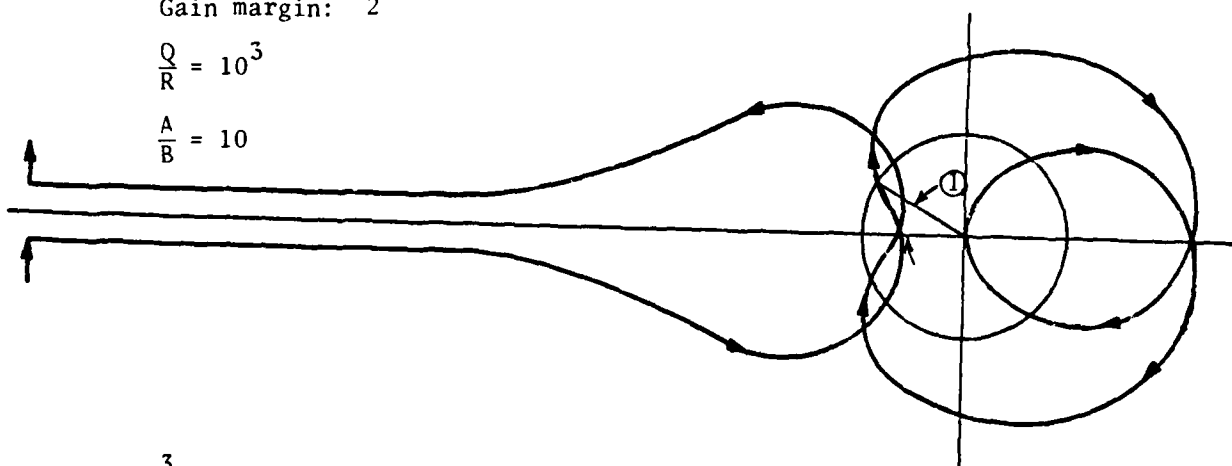
⊖  $(10^3, 10)$  Steady-State Design

① Phase margin:  $22^\circ$

Gain margin: 2

$$\frac{Q}{R} = 10^3$$

$$\frac{A}{B} = 10$$



P  $(10^3, 10)$  Gradient-Search Design

Figure 8.5 'OPTIMAL' DESIGN COMPARISON, TRIAL II NYQUIST PLOTS

system augmented by a second order compensator) to a fourth order design using steady-state Riccati analysis. The computation ratio was 4 to 1.

- 4) The "Simplification" design cycle must be iterated.
- 5) When designing with "Simplification," the space of all desirable compensators is not available.
- 6) The gradient search procedure allows the designer to include all known information, to weight all known states, and to minimize the overall system sensitivity to parameter variations. Since the compensator is not of full-order, some of this information is lost. It is not yet known how this design compares to the full state compensation, or what information is lost in the simplification.

We therefore have a viable algorithmic procedure for designing reduced-order compensators. This technique uses the same independent parameters as full-state compensator design-- the weighting of states and the minimization of state covariance. System stability is guaranteed, and an optimal design of any order can be realized directly.

#### VIII.4 An Example

Having demonstrated the efficacy of the gradient search procedure when applied to a hypothetical problem, we shall now consider an engineering application.

The problem is to design a minimum-order compensator for a star-tracking telescope (see Appendix C2). Using engineering judgment, the model for the telescope was reduced from twelfth order to fifth order. With much effort, a third order compensator was then derived. The loss of response, however, was significant.

Figure 8.6 shows the results of optimizing the third order compensator using the gradient procedure; the important criterion in this application-- the speed of response-- is approximately proportional to the radial distance from the origin. Therefore, optimization has improved the speed of response by 30%.

## COMPENSATED STAR-TRACKING TELESCOPE

Physical System - 5th order

Compensation - 3rd order

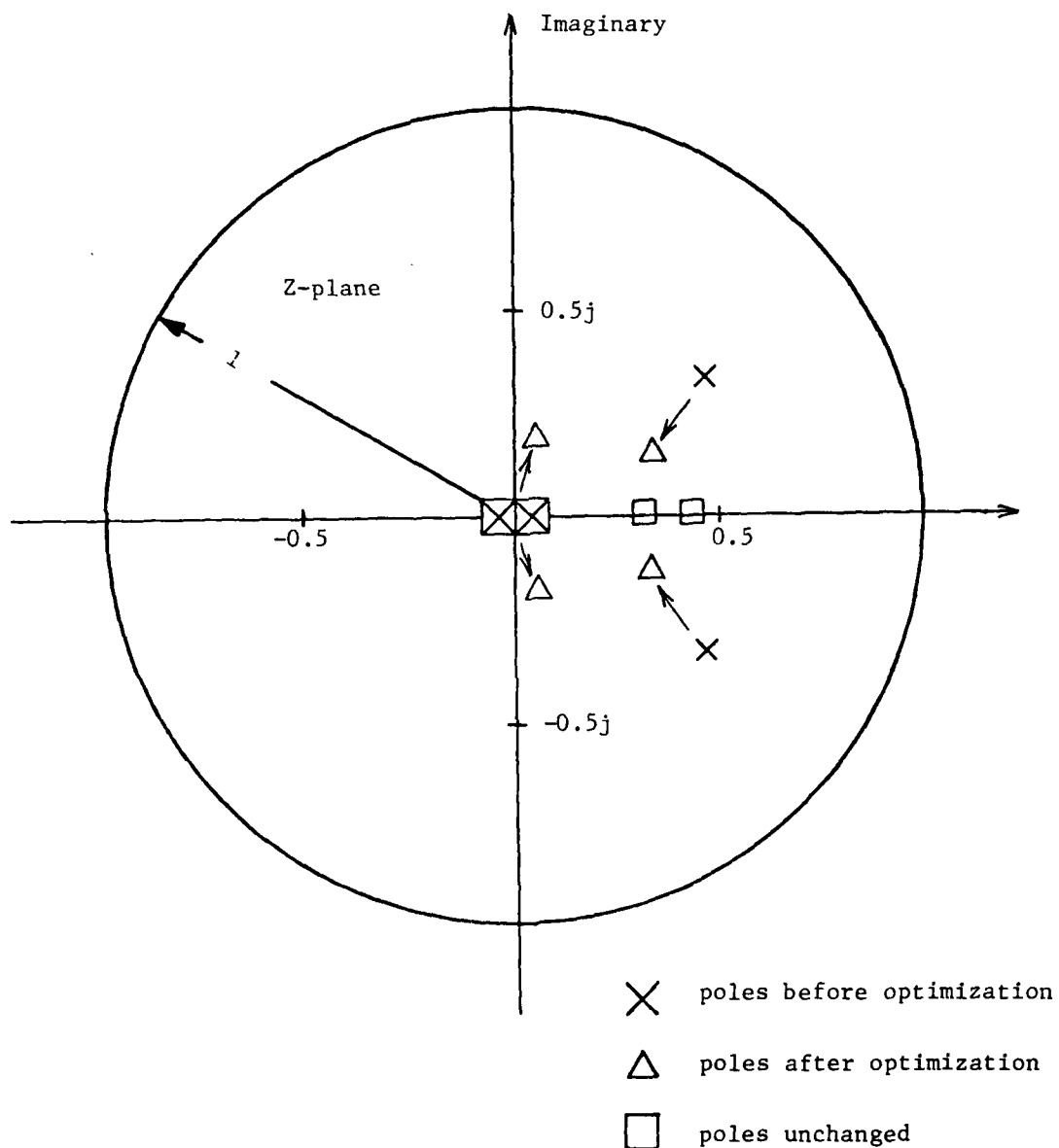


Figure 8.6

LOCUS OF COMPENSATION OPTIMIZATION

## Chapter IX

### IMPLEMENTATION ISSUES

Implementation of the algorithms presented in earlier chapters is generally straightforward, but some interesting computational issues do arise. These issues will be the focus of this chapter; specific details, such as program descriptions and source code, will follow in a separate technical report.

The algorithms described in chapters III and IV were implemented in a hybrid mixture of Fortran and C.<sup>‡</sup> Since many numerical analysis programs -- the QR algorithm, Singular Value Decomposition, etc. -- already exist in Fortran, we chose to use these programs as written. New programs, were written in C, a language often better suited to this type of procedural programming. (In addition, our system provides much better support for developing C programs.)

In retrospect, the concomitant advantages intrinsic in using two languages may not have outweighed the disadvantages of increased complexity and of pioneering the interface between the resulting bifurcated program. Two Fortran compilers and two language interfaces later (and after a tremendous amount of work) the language processors were ultimately debugged, and a powerful signal-processing library evolved. The diversion of effort required, however, was considerable.

The decision to develop a hybrid implementation was made too lightly. In research, as in industry, decisions which involve significant commitments of programming effort must be made after carefully considering the consequences. The requisite pioneering required was uniformly underestimated.

---

<sup>‡</sup>C is a language similar to Algol supplied by Bell Laboratories with their Unix operating system.

### IX.2 Algorithm Implementation

By exploiting the structure of the arrays, many speed advantages can be accrued. For example, to multiply two general matrices requires  $n^3$  multiplications (using traditional sequential algorithms). To multiply two lower triangular requires  $n \cdot (n+1)(n+2)/6$  multiplies; for a sixteenth order system, this is an 80% reduction.

In square root algorithms, we can exploit this characteristic quite often. If the square root of a quantity  $C$  is  $L$ , where  $C = LL^T$ , and we normally compute with  $L$ , then constrain  $L$  to be lower left triangular. If we normally compute with  $L^T$ , then constrain  $L^T$  to be lower left triangular. In both cases,  $C = LL^T$ .

Gaussian Elimination can also be used to simplify multiplication. If we have four block matrices--  $A$ ,  $B$ ,  $C$ ,  $D$ -- and need to calculate  $D - CA^{-1}B$ , this is equivalent to arranging the blocks in an array

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix}$$

and performing Gaussian Elimination on  $B$  and  $D$  to bring this array to the form

$$\begin{bmatrix} A & 0 \\ C & E \end{bmatrix}$$

$E$  then equals  $D - CA^{-1}B$ .

### IX.3 Parallel Implementation

Several papers have been written on using special hardware configurations to assist in computing these algorithms. Sameh and Kuck [Sameh, 1977] present a QR algorithm requiring  $O(n)$  processors which results in a speedup of  $O(n/\log_2 n)$ .

Morf, Dobbins, Friedlander, and Kailath [Morf, 1978] made a similar observation in noting the applicability of parallel processing to square root algorithms. Recalling Section III.7, they proposed partitioning an interval of data into subintervals within which state estimates are calculated based only on data within the subinterval; a separate processor then calculates the estimate for each subinterval. Only then do the processors need to communicate, to "connect" the subinterval endpoints.

These algorithms look extremely promising, but other alternatives are also attractive. For example, in the doubling algorithms, we could perform Householder reflections in parallel, with each processor operating on a different column or a different row. As we found in Chapter VI after looking at results, it is not necessarily obvious which solution is preferable. Memory dynamics and intra-processor interactions can profoundly effect the efficacy of an algorithm.

The interaction between algorithms and their hosts, and the ensuing synergy, is a largely unexplored area, especially for parallel processing. More research is needed to identify the underlying structure of linear system theory, to understand the communication of information required during the processing, and to bound the error propagation as parallelism is exploited.



#### IX.4 Special Hardware

The parameter optimization program of Chapter VII, in solving a fifth order problem, cost twenty dollars to run on the IBM 370. The algorithm is order  $n^4$ . For a sixteenth order problem, the cost soars to \$2,000 per design cycle. The question is whether to focus on improving the algorithm or minimizing the hardware costs.

The IBM 370 rents for approximately \$1300 per hour. Recently, very fast, sequential processors have been introduced which are approximately one hundred times faster, and rent for \$75 per hour. If one could be adapted to the task, the 16th order system could be solved for \$1.15 instead of \$2,000.00.

The transition, and consequent savings, is not straight forward because the fast sequential processor (henceforth referred to as an Array Processor, or AP) could not be programmed in FORTRAN. Programming them in their language is so primitive, by current standards, that it can only be justified for processing very large volume problems. (For example, a Fast Fourier Transform program was estimated to take 200 hours of programming effort.) Industry experience indicates we could program and debug one machine language instruction per hour; the parameter optimization algorithm is over 2,000 FORTRAN statements in length.

As an alternative, the AP manufacturer supplied FORTRAN callable subroutines for nearly all vector operations, and for such standardized computations as matrix inversion and Fourier transforms. Performance data for our system using this software appear in Figure 9.1; execution times are shown for vector multiplication (VMUL), vector initialization

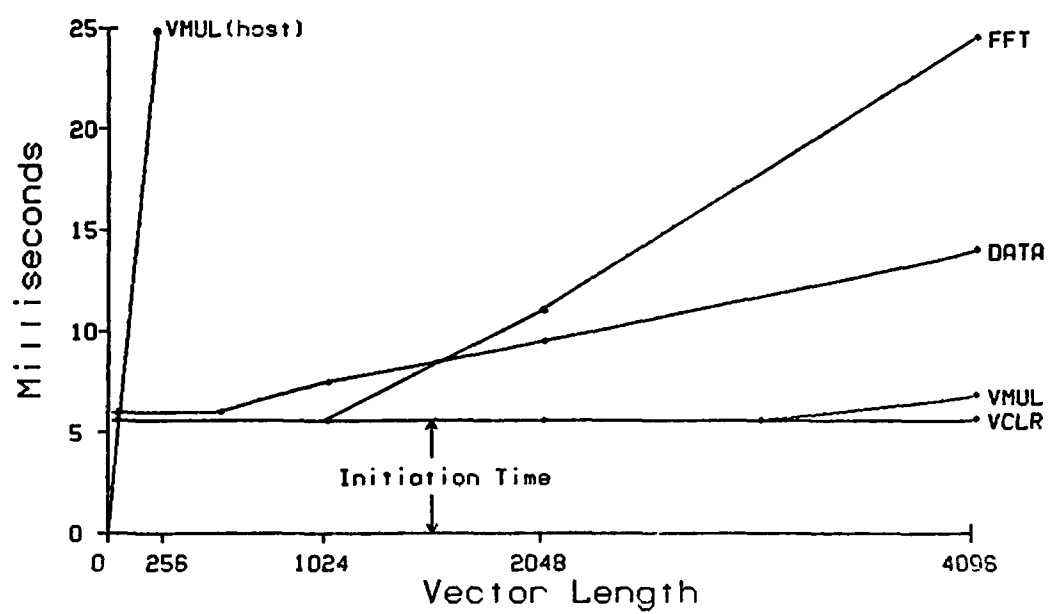


Figure 9.1 ARRAY PROCESSOR PERFORMANCE

(VCLR), Fast Fourier transforms (FFT), and data transfer (DATA) as a function of vector length  $N$ . Note that the crossover between host and AP vector multiply occurs near  $N = 60$ . Our AP can hold a  $90 \times 90$  matrix. Thus, manipulating columns of a matrix in the AP is only marginally faster than in our host (and additional AP memory would provide little improvement).

Conventional wisdom has held that array processors tend to be input-output limited; the array processor, as a peripheral device attached to the host, must transfer all of its data to and from the host memory. For most matrix problems and many vector problems, however, the real limitation lies in the time required for the host to initiate the next AP program. This limit makes few tasks besides Fourier transforms look attractive. As a consequence, despite diligent efforts at applying this system to projects in our laboratory, we have yet to exceed ninety seconds of AP useage per day (as of March, 1978).

These results are consistent with measurements made at other installations. Therefore, if the research potential of array processors is to be realized either the host computer must be dedicated to serving the AP, or the AP's software and interface hardware must be improved beyond what is currently available.

The array processor, although capable of resolving the speed-cost problem, could not be used because of these weaknesses.

## Chapter X

### CONCLUSIONS

The important conclusion from the first chapters is that square-root doubling is an attractive alternative for solving the discrete-time Riccati equations arising in estimation and control. Numerically, the algorithm compares favorably with the alternatives: the speed is comparable up to tenth order, the memory requirements are comparable, and the accuracy is comparable. The primary advantage of square root doubling is the broad class of problems it will solve. This expanded class includes systems with singular dynamics matrices (not handled by eigenvector decomposition), and the SQD algorithm handles problems with repeated closed-loop eigenvalues sans difficulty. As noted, eigenvector decomposition can presumably be extended to also handle these cases.

From a theoretical perspective, a direct, algebraic derivation of the square root doubling algorithm was presented. The role of orthogonal transformations was elucidated; this included rank compression and the implicit calculation of matrix inverses. In the process, several questions concerning the structure of the algorithm were raised.

A pure scattering theory derivation was given. Defining the square-root of a scattering matrix-- actually one "rail" of a symmetric network-- these questions were readily solved. The J-orthogonal transformations appeared both to unwind loops and to "compress" data paths. The  $\Phi$  matrix update arising in the scattering framework was now clearly different from the error covariances being updated with the square root algorithm. Transmission path variables were not amenable to square roots, whereas variables cutting the line of symmetry were.

In the continuous case, eigenvector decomposition still appears to be the preferred choice for solving the Riccati equation. This result follows because other alternatives either require choosing a discretionary constant or inordinately constrain the class of solvable problems. This conclusion may change either from the introduction of new algorithms or by specifying a procedure for selecting the discretionary constant.

In examining systems with singular transition matrices, a published example was considered, and shown to be a suboptimal solution which performed well, within the problem's context.

A discrete-time parameter-uncertainty algorithm was evolved from previous continuous algorithms. Limitations on the algorithm were found:

- 1) uncertain modes had to be observable through the cost function, and
- 2) uncertain modes had to be excited by external disturbances.

This latter constraint arises, at least in part, from simplifying assumptions made during the derivation; these assumptions led to the introduction of an unphysical scale factor introduced in an earlier study [Hadass, 1974].

In applications, this approach elucidated the ad hoc desensitization of an earlier work [Katz, 1974]. When applied to a realistic example, it improved the sensitivity range by 50%. The procedure was straightforward, but required  $O(n^4)$  computer time.

This algorithm was applied to the design of arbitrary-order compensation using full-system information. Excellent results were obtained, both for created problems in the frequency domain and for an actual telescope example. Again, the computation required was high, but the

total man hours required was significantly reduced compared to a classical design.

The adaptability of system theory algorithms to specialized hardware was examined and found very promising. The adaptability of present hardware to present algorithms was found lacking. Software research, including new languages and fundamental numerical algorithms, is required. The area promising the most return for invested research, however, is probably in hardware, including special processors, innovative memory systems, and external interfaces.

## APPENDIX A1

SOFTWARE FOR SOLVING THE STEADY-STATE RICCATI EQUATION  
Implemented in 1978 on a DEC PDP11

Part of this software implements many of the algorithms developed in Chapters III and IV. Other programs provide the support required to properly interpret the computations. These routines cannot be considered design tools; they were written with research as the prime objective. We feel, however, their design philosophy will lead to effective design tools as our repertoire of software expands.

We chose to implement each separable portion of each algorithm as an individual program, thereby minimizing memory requirements and cleanly modularizing the software. For example, there are separate programs for calculating the requisite controller gains, the desired filter gains, the loop eigenvalues, for generating data, and for plotting the results.

Our operating system, UNIX, provides facilities for easily connecting the output of one program to the input of another. For example, to generate the Kalman gains for a given model ( stored in file "model")-- where we wish to vary the noise parameter, q-- we would feed the input data into a generator program ("gen"), which feeds into the square root doubling algorithm ("dbl. dk"), then into the eigenvalue routine ("eigen"), and then into an editing routine which extracts the Kalman gains ("strip"). On our system, this is entered as the command:

```
gen model iq q|dbl.blk|eigen|strip K
```

Ten seconds later the results appear at the terminal-- and only the results requested (the gains K).

The currently available software is listed in Table A.1.1.

### Program Modules

#### Mainline

The mainline code is written in FORTRAN, and does little more than call subroutines.

#### Numerical Analysis

The numerical analysis routines closely follow the IMSL<sup>†</sup> package, and are written in FORTRAN. The routines used include the QR algorithm, Householder transforms, Cholesky decomposition (for taking square roots of positive definite matrices), and a singular-value-decomposition routine, also for taking matrix square roots.

#### Input-output

A centralized module provides input and output capabilities.

#### Matrix support

A final module provides all requisite matrix operations-- such as matrix multiply and Gaussian elimination-- on block matrices. Matrices are defined at compile time to have a prescribed structure, e.g., to be of the form:

	P	m	n	
P	1	3	5	
n	2	4	6	matrix 3

---

<sup>†</sup> IMSL - International Mathematics and Statistical Libraries.



The programmer could then multiply block 2 of matrix 3 by block 5, leaving the result in block 6 [an (n by p) times (p by n) calculation] by writing

```
call mul(3,2, 3,5, 3,6)
```

Arguments come in pairs; the first element specifies the matrix, the second specifies the block. To facilitate the programming process, this code was originally written in C (an ALGOL-like language), and was later translated into FORTRAN.

#### Input Format

The input format is nearly completely unrestricted. The user merely specifies an array name, followed by data values. The input routine checks for consistent dimensions, and for a complete input set.

A nearly self-explanatory input file appears in Figure A.1.1. Note that comments are any string of characters appearing between '/' and '/'; comments are completely ignored by all programs.

## Implemented Software

Table A.1.1

<u>NAME</u>	<u>FUNCTION</u>
gen	Take an existing model and modify it repeatedly, according to specifications supplied from a table
eigen	Find all appropriate eigenvalues (given $\phi$ , $K$ , and $H$ ); eigen finds both the open and closed loop values
strip	Delete unwanted output
dbl.dk	square root doubling, discrete, Kalman gains
sqr.dk	square root, discrete, Kalman gains
ric.dk	Riccati iteration, discrete, Kalman gains
dbl.blk	bilinear square root doubling, Kalman gains
dbl.dc	square root doubling, discrete, controller gains
disc	Eigenvector decomposition, discrete
optsys	Eigenvector decomposition, continuous

All additional software descriptions, listings, etc., will appear in a separate technical report.

/\*

## SOLUTION OF THE STEADY STATE RICATTI EQUATION

Tue May 23 11:00:45 1978

Square-Root Doubling

Revision I

star. 1

Number of

states-- 4

disturbances-- 1

measurements-- 1

\*/ ①

/\*

\* Test of star tracking telescope model

\*

\* Two-fold problem:

\* 1) Has repeated roots, marginally stable

\* 2) Incorporates two delays.

\*

\*/

print\_flag = 50

iteration\_count = 30

timing\_cycles = 10

phi

1.0 1.0 0.0 0.0

0.0 1.0 0.0 0.0

1.0 0.0 0.0 0.0

0.0 0.0 1.0 0.0

gamma 0.5

1.0

0.0

0.0

q

100.0

h

0.0 0.0 0.0 1.0

r

1.0

Note:

① First comment supplied by pre-processor

Figure A.1.1 SAMPLE PROGRAM INPUT FILE

APPENDIX B1  
DESIGN OF A DISCRETE ESTIMATOR FOR THE  
SPACE LAB INFRARED TELESCOPE

From the paper by J. David Powell and Eric Parsons, [Powell, 1978] we have the following model for the gyro star tracker (see also Appendix C2):

$$x_n = \begin{Bmatrix} \Theta_{DN} \\ D \end{Bmatrix}_n = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} x_{n-1} + \begin{bmatrix} T^2/2 \\ T \end{bmatrix} \eta_{n-1} = \phi x_{n-1} + \Gamma \eta_{n-1} \quad (1)$$

$$y_n = [1 \quad 0] x_{n-1} + w_n - v_n = H x_{n-1} + w_n - v_n \quad (2)$$

We now follow their development, elaborating where necessary to clarify issues.

We assume the following disturbance characteristics:

$$\varepsilon(\eta_i \eta_j^T) = N \delta_{ij} \quad \varepsilon(\eta_i) = 0 \quad (3)$$

$$\varepsilon(v_i v_j^T) = R \delta_{ij} \quad \varepsilon(v_i) = 0 \quad (4)$$

$$\varepsilon(w_i w_j^T) = Q \delta_{ij} \quad \varepsilon(w_i) = 0 \quad (5)$$

$$\varepsilon(\eta_i v_j^T) = 0 \quad (6)$$

$$\varepsilon(\eta_i w_j^T) = 0 \quad (7)$$

$$\varepsilon(v_i w_j^T) = 0 \quad (8)$$

Replacement by Equivalent System with no Delays

Applying (1) to (2) recursively, we get

$$\begin{aligned} x_n &= \phi x_{n-1} + \Gamma \eta_{n-1} \\ &= \phi^2 x_{n-2} + \phi \Gamma \eta_{n-2} + \Gamma \eta_{n-1} \end{aligned}$$

Assuming  $\phi$  is invertible, this gives

$$x_{n-2} = \phi^{-2} x_n - \phi^{-2} \Gamma \eta_{n-1} - \phi^{-1} \Gamma \eta_{n-2}$$

$$y_n = H(\phi^{-2} x_n - \phi^{-2} \Gamma \eta_{n-1} - \phi^{-1} \Gamma \eta_{n-2}) - v_n + w_n$$

Defining  $\bar{H} = H\phi^{-2}$

$$\bar{v} = w_n - v_n - \bar{H}\Gamma\eta_{n-1} - H\phi^{-1}\Gamma\eta_{n-2}$$

$$y_n = \bar{H}x_n + \bar{v}_n \quad (9)$$

However, it is no longer true that

$$\varepsilon(\eta_i \bar{v}_j) = 0$$

$$\varepsilon(\bar{v}_i \bar{v}_j) = C \delta_{ij}, \text{ for some constant } C.$$

We begin uncorrelating the noises by modifying the state equation.

Since

$$y_n - \bar{H}\phi x_{n-1} - \bar{H}\Gamma\eta_{n-1} - \bar{v}_n = 0$$

we can introduce a weighting matrix,  $L$ , such that

$$x_n = (I - L\bar{H})\phi x_{n-1} + Ly_n + (I - L\bar{H})\Gamma\eta_{n-1} - L\bar{v}_n \quad (10)$$

Since  $Ly_n$  is a deterministic input, we are free to choose  $L$  so that the new process noise is "uncorrelated" with the measurement noise  $\bar{v}$ , thus:

$$\varepsilon\{(I-L\bar{H})\Gamma\eta_{n-1} - L\bar{v}_n\} \cdot \bar{v}_n^T = \varepsilon\{\eta_{n-1} \cdot v_n^T\} \quad (11)$$

$$\begin{aligned} \varepsilon\{(I-L\bar{H})\Gamma\eta_{n-1} - L(w_n - v_n - \bar{H}\Gamma\eta_{n-1} - H\phi^{-1}\Gamma\eta_{n-2}) \cdot (w_n - v_n - \eta_{n-1}\Gamma^T\bar{H}^T - \eta_{n-2}\Gamma^T\phi^{-T}H^T)\} \\ = -(I-L\bar{H})\Gamma\Gamma^T\bar{H}^T - LQ - LR - L\bar{H}\Gamma\Gamma^T\bar{H}^T \\ - LH\phi^{-1}\Gamma\Gamma^T\phi^{-T}H^T \\ = -\Gamma\Gamma^T\bar{H}^T - L(Q+R+H\phi^{-1}\Gamma\Gamma^T\phi^{-T}H^T) \end{aligned}$$

or, defining  $\bar{R}$  such that

$$\begin{aligned} \bar{R} = \varepsilon(\bar{v}_i \bar{v}_i^T) &= Q + R + \bar{H}\Gamma\Gamma^T\bar{H}^T + H\phi^{-1}\Gamma\Gamma^T\phi^{-T}H^T \\ &= Q + R + \bar{H}(\Gamma\Gamma^T + \phi\Gamma\Gamma^T\phi^T)\bar{H}^T \end{aligned} \quad (12)$$

Substituting (12) into (11), and setting (11) to zero yields

$$\begin{aligned} 0 &= -(I-L\bar{H})\Gamma\Gamma^T\bar{H}^T - L\bar{R} \\ i &= -\Gamma\Gamma^T\bar{H}^T(\bar{H}\Gamma\Gamma^T\bar{H}^T - \bar{R})^{-1} \\ &= -\Gamma\Gamma^T\bar{H}^T(Q+R+H\phi^{-1}\Gamma\Gamma^T\phi^{-T}H^T)^{-1} \end{aligned} \quad (13)$$

Next, we examine the resulting measurement and process noise correlation functions:

$$\begin{aligned} \varepsilon(\bar{v}_n \bar{v}_{n-1}^T) &= \varepsilon(w_n - v_n - \bar{H}\Gamma\eta_{n-1} - H\phi^{-1}\Gamma\eta_{n-2}) (w_{n-1} - v_{n-1} - \eta_{n-2}\Gamma^T\bar{H}^T - \eta_{n-3}\Gamma^T\phi^{-T}H^T) \\ &= H\phi^{-1}\Gamma\Gamma^T\bar{H}^T = \bar{R}_1 \quad (\text{from (9)}) \\ \varepsilon(\bar{v}_i \bar{v}_j^T) &= \begin{cases} \bar{R}_0 & , \text{ if } i=j \\ \bar{R}_1 & \text{ if } |i-j| = 1 \\ 0 & \text{ else} \end{cases} \end{aligned}$$

$$\begin{aligned}\varepsilon(\bar{\eta}_{n-1} \bar{\eta}_{n-2}^T) &= \varepsilon((I - L\bar{H})\Gamma\eta_{n-1} - L(w_n - v_n - \bar{H}\Gamma\eta_{n-1} - H\phi^{-1}\Gamma\eta_{n-2}))^*(\quad)^T \\ &= LH\phi^{-1}\Gamma N\Gamma^T = \bar{N}_1\end{aligned}$$

$$\varepsilon(\bar{\eta}_i \bar{\eta}_j^T) = \begin{cases} \bar{N}_0 & \text{if } i=j \\ \bar{N}_1 & \text{if } |i-j| = 1 \\ 0 & \text{else} \end{cases}$$

$$\varepsilon(\bar{v}_n \bar{\eta}_{n-1}^T) = 0, \quad \text{by construction}$$

$$\begin{aligned}\varepsilon(\bar{v}_n \bar{\eta}_n^T) &= \{(w_n - v_n - \bar{H}\Gamma\eta_{n-1} - H\phi^{-1}\Gamma\eta_{n-2})((I - L\bar{H})\Gamma\eta_n - L(w_{n+1} - v_{n+1} - \bar{H}\Gamma\eta_n - \\ &\quad H\phi^{-1}\Gamma\eta_{n-1}))^T\} \\ &= -\bar{H}\Gamma N\Gamma^T \phi^{-T} H^T L^T = \bar{C}_{01}\end{aligned}$$

$$\varepsilon(\bar{v}_n \bar{\eta}_{n-2}^T) = -H\phi^{-1}\Gamma N\Gamma^T = \bar{C}_{10}$$

$$\varepsilon(\bar{v}_i \bar{\eta}_j^T) = \begin{cases} \bar{C}_{01} & , \quad \text{if } i-j = 0 \\ \bar{C}_{10} & , \quad \text{if } i-j = 2 \\ 0 & \text{else} \end{cases}$$

### Numerical Results

$$\phi^{-1} = \begin{bmatrix} 1 & -T \\ 0 & 1 \end{bmatrix} \quad \phi^{-2} = \begin{bmatrix} 1 & -2T \\ 0 & 1 \end{bmatrix} \quad \bar{H} = H\phi^{-2} = \begin{bmatrix} 1 & -2T \end{bmatrix}$$

$$\begin{aligned}\bar{R}_0 &= Q + R + \frac{5}{2} T^4 N & L &= \frac{3}{4} N \begin{bmatrix} T^4 \\ 2T^3 \end{bmatrix} (Q + R + \frac{T^4 N}{4})^{-1} \\ \bar{R}_1 &= \frac{3}{4} T^4 N\end{aligned}$$

$$\bar{N}_0 = (I - L\bar{H}) \begin{bmatrix} T^{4/2} & T^{3/2} \\ T^{3/2} & T^2 \end{bmatrix} N$$

$$\bar{N}_1 = L \begin{bmatrix} -\frac{T^4}{4} & -\frac{T^3}{2} \end{bmatrix} N$$

$$\bar{C}_{01} = -\frac{3}{4} T^4 L^T N$$

$$\bar{C}_{10} = \begin{bmatrix} \frac{T^4}{4} N & \frac{T^3}{2} N \end{bmatrix}$$

Assuming  $\frac{T^4 N}{4} \gg (Q+R)$

$$L = \begin{bmatrix} 3 \\ 6/T \end{bmatrix}$$

$$\bar{R}_0 = \frac{5}{2} T^4 N$$

$$\bar{R}_1 = \frac{3}{4} T^4 N$$

$$, \quad \bar{R}_1 = \frac{3}{10} \bar{R}_0$$

$$\bar{N}_0 = \begin{bmatrix} \frac{5}{2} T^4 & 5T^3 \\ 5T^3 & 10T^2 \end{bmatrix} N$$

$$\bar{N}_1 = - \begin{bmatrix} \frac{3}{4} T^4 & \frac{3}{2} T^3 \\ \frac{3}{2} T^3 & 3T^2 \end{bmatrix} N$$

$$, \quad \bar{N}_1 = -\frac{3}{10} \bar{N}_0$$

$$\bar{C}_{01} = -\frac{9}{4} [T^4 \quad 2T^3] N$$

$$\bar{C}_{10} = \frac{1}{4} [T^4 \quad 2T^3] N$$

Note that

$$\|\mathcal{E}(\bar{v}_i \bar{v}_{i+1}^T)\| \sim 0.3 \times \|\mathcal{E}(\bar{v}_i \bar{v}_i^T)\| \quad (14)$$

$$\|\mathcal{E}(\bar{n}_i \bar{n}_{i+1}^T)\| \sim 0.3 \times \|\mathcal{E}(\bar{n}_i \bar{n}_i^T)\|$$

$$\|\mathcal{E}(\bar{v}_i \bar{n}_i^T)\| \sim \|\mathcal{E}(\bar{n}_i \bar{n}_i^T)\|$$

for many cases.

In conclusion, for this example, when process noise dominates measurement noise, the assumption that these disturbances are white and uncorrelated breaks down significantly. We saw in Chapter V that this leads to suboptimal results.



## Appendix C

## EXAMPLE PROBLEM FORMULATIONS

The following appendices detail the problem formulations used as examples in the body of the text. They are included to facilitate replication of the results presented earlier.

## APPENDIX C1

DERIVATION OF F-H AIRCRAFT EQUATIONS  
A Synthesis of Paul Katz's Example Problem [Katz, 1974]Admonitions:

The references used to synthesize the FH example have several significant typographic errors; in some figures terms are unquestionably missing, the derivative terms are of the wrong order, or the subscripts are incorrect. There is insufficient background information presented in the cited references to reconstruct these equations; corrections were normally made by appealing to consistency in those cases where the correction was unclear.

Derivation:

If the objective is to accurately model the FH example aircraft, then the best set of equations and coefficients appear on pages 256-259 of reference 6. These equations incorporate three bending modes, a set of flight conditions which don't match those of Katz, and no wind gust model. The wind gust can easily be introduced by using Katz's first order Gaussian model, and the substitution

$$\alpha = \alpha_T + \frac{1}{U_0} w_g$$

The sequel assumes the objective is to match Katz's results, using his proffered equations, reference 1, pages 45-50. This is difficult, since the equations and coefficients are inconsistent even within the section just cited; these equations were actually drawn from several diverse sections of references 2 through 6, they were not copied accurately, and the equations which were chosen as an original basis had

ambiguities which were not resolved satisfactorily. These problems will be pointed out in the sequel, but no attempt will be made to analyze the impact.

We begin by reproducing Katz's initial equations (pp. 45-50, Ref. 1)

$$[1] \quad \dot{q} = (M_q)q + (M_\alpha)\alpha_T + (M_\alpha^*)\dot{\alpha}_T + (M_{\delta_e})\delta_e + (k_b M_{\delta_e} \frac{z'_e}{\omega_b^2}) x_3$$

$$[2] \quad \dot{\alpha}_T = q + \left(\frac{Z_\alpha}{U_0}\right)\alpha_T + \left(\frac{1}{U_0 \tau_w}\right) w_g + \left(\frac{Z_{\delta_e}}{U_0}\right)\delta_e - \left(\frac{\Gamma}{U_0}\right)\eta_g$$

$$[3] \quad \dot{w}_g = \left(-\frac{1}{\tau_w}\right) w_g + \Gamma \eta_g$$

$$[4] \quad \dot{x}_3 = \omega_b x_4 + \Gamma_4 \eta_g$$

$$[5] \quad \dot{x}_4 = (-\omega_b) x_3 - (2f_b \omega_b) x_4 + (\omega_b k_1 Z_{\delta_e}) \delta_e + (k_2 \omega_b Z_\alpha) \dot{\alpha}$$

output

$$[6] \quad q_T = q + \left(\frac{Z'_1}{\omega_b}\right) x_4 + v_q$$

$$[7] \quad n_T = U_0 (q - \dot{\alpha}) + (\ell_a) \dot{q} + \left(\frac{\bar{Z}_1}{\omega_b}\right) \dot{x}_4 + v_n$$

where

$$\alpha_T = \alpha - \frac{1}{U_0} w_g$$

$q$  - pitch rate,  $\text{sec}^{-1}$

$\alpha$  - angle of attack, rad

$w$  - gust velocity, ft/sec

$x_4$  - bending mode velocity, rad/sec

$$\Gamma = \frac{\sqrt{2\tau_w}}{\tau_w} \sigma_w$$

From notations in reference 2, Katz apparently began with the FH simulation appearing in Appendix M, pgs. 382-383. This is a 7th order simulation, incorporating two bending modes, presented with no explanation. The higher order bending mode is simply dropped in the derivation of the fifth order model.

We have, removing obvious typographic errors--

$$[8] \quad \dot{q} = (M_q)q + (M_\alpha)\alpha + (M_\alpha^*)\dot{\alpha}_i + (M_{\delta_e})\delta_e + (M_{\delta_s})\delta_s$$

$$[9] \quad \dot{\alpha} = q + \left(\frac{Z_\alpha}{U_0}\right)\alpha_T + \left(\frac{Z_{\delta_e}}{U_0}\right)\delta_e + \left(\frac{Z_{\delta_s}}{U_0}\right)\delta_s$$

$$\dot{x}_4 = (\omega_1)x_5$$

$$\dot{x}_5 = (\omega_1)\dot{e}_5$$

$$\dot{e}_5 = -x_4 - (2f)x_5 + (4Z_{\delta_e})\delta_e + (0.6 Z_\alpha)\dot{\alpha}_i$$

$$\dot{w}_g = \frac{1}{T_w}(-w_g + \sqrt{2T_w} \sigma)\eta_g$$

$$\alpha_T = \alpha_i - \frac{1}{U_0} w_g$$

$$Z_{1a} = .070$$

$$[10] \quad q_T = Z'_{1g} e_5 + Z'_{2g} e_6 + q$$

$$Z'_{1g} = -0.005$$

$$f = 0.01$$

$$n_{cg} = U_0(q - \alpha_i)$$

$$n_{ar} = n_{cg} + (\ell_a)\dot{q}$$

$$n_{aT} = n_{ar} + (\bar{Z}_{1a})\dot{e}_5 + (\bar{Z}_{2a})\dot{e}_6$$

$$\dot{e}_6 = -x_6 - \dots \quad \text{2nd bending mode}$$

dropping the 2nd bending mode and spoiler input, and making the following change of notation,

$$\begin{array}{lll}
 x_4 \rightarrow x_3 & \sigma \rightarrow \sigma_w & \bar{z}_{1a} \rightarrow \bar{z}_1 \\
 x_5 \rightarrow x_4 & \alpha_i \rightarrow \alpha & \omega_1 \rightarrow \omega_b \\
 T_w \rightarrow \tau_w & z_{1g} \rightarrow z_1 &
 \end{array}$$

The remaining question involves determining whether  $\alpha$ , in equations [8-9], should be  $\alpha_i$  or  $\alpha_T$ . We shall investigate this point in a moment.

$$[11] \quad \dot{q} = (M_q)q + (M_\alpha)\alpha_T + (M_\alpha^*)\dot{\alpha} + (M_{\delta_e})\delta_e$$

$$\dot{\alpha}_T = q + \left(\frac{z_\alpha}{U_0}\right)\alpha_T + \left(\frac{z_{\delta_e}}{U_0}\right)\delta_e$$

$$\dot{w}_g = \frac{w_g}{\tau_w} + \Gamma\eta_g$$

$$\dot{x}_3 = \omega_b x_4$$

$$\dot{x}_4 = \omega_b (-x_3 - 2f_1 x_4 + 4z_{\delta_e} \delta_e + 0.6 z_\alpha \dot{\alpha})$$

$$[12] \quad q_T = z_1' e_5 + q = q + \frac{z_1' x_4}{\omega_b} + K \quad \{\text{we assume } K=0\}$$

$$\eta_T = U_0(q - \alpha) + (\ell_a)\dot{q} + \left(\frac{z_1}{\omega_b}\right)\dot{x}_4$$

We begin by noting an apparent discrepancy between equations [1] and [11]. Working now from the equations given on page 257 of Ref. 6 and pages 18-22 of Ref. 4, we get

$$[13] \quad \dot{\alpha}_T = q + \left(\frac{z_\alpha}{U_0} + \frac{1}{\tau_w U_0}\right)w_g - \frac{\Gamma}{U_0}\eta_g + \left(\frac{z_\alpha}{U_0}\right)\alpha_T + \left(\frac{z_{\delta_e}}{U_0}\right)\delta_e$$

$$[14] \quad \dot{q} = (M_q)q + (M_\alpha)\alpha_T + (M_\alpha^*)\dot{\alpha}_T + (M_{\delta_e})\delta_e + \left[ \frac{M_\alpha^*}{U_0}\dot{w}_g + \frac{M_\alpha}{U_0}w_g \right]$$



Table C.1.1

## COEFFICIENT REFERENCES

Coefficients:

	Found in Katz	Used in Eq. [2-7]	From Ref. 2 pp. 190-191	Ref. 2 p. 382	Ref. 2 192
$U_o$	x	x	x		
$M_{ql}$	x	x	x		
$M_{\omega}$	x		x		
$M_{\delta_e}$	x	x	x		
$M_{\omega}^{\bullet}$	x		x		
$Z_{\omega}$	x		x		
$Z_{\delta_e}$	x	x	x		
$\tau_{\omega}$	x	x	x		
$\sigma_{\omega}$	x	x	x		
$f$	x	x		x	
$\omega_b$	x	x		x	
$\delta_{e_{max}}$	x				
$\bar{Z}$	x	x		x	
$Z'$	x	x		x	
$k_1$	x	x		x	
$k_2$	x	x		②	
$\ell_a$	x	x	①		③
$k_b$	x	x			
$Z_{\alpha}$	④	x			
$M_{\alpha}$	④	x			
$M_{\alpha}^{\bullet}$	④	x			
$Z'_e$	⑤	x			
$\eta_g$	x	x			
$v_q$	x	x			
$v_n$	x	x			

(con't)

Table C.1.1 (con't)

- ① Katz used 14 feet, which is the value given for ③. Calculated, however, the value would be

$$l_{cg} - i/12 = \frac{320.4 - 77.0}{12} = 20.3 \text{ feet}$$

- ② The simulation gives a value of 0.6 for  $k_2$ . Katz used 0.06.

$$\textcircled{4} \quad Z_{\alpha} = Z_{\omega} \cdot U_0 \quad M_{\alpha} = M_{\omega} \cdot U_0 \quad M_{\alpha}^* = M_{\omega}^* \cdot U_0$$

- ⑤ By empirical induction on Katz's data,  $Z_e' = -7.23 \times 10^{-2}$



Design parameters:Weighting matrices--

Katz originally used  
{Ref. 1, page 78}

$$A = \begin{bmatrix} 0.12 & & & \\ & 11 & & 0 \\ & & 0 & \\ & 0 & & 0 \\ & & & & 0 \end{bmatrix}$$

$$B = 1$$

This choice yields a non-observable cost function; to produce adequate sensitivity margins he later adopted [Ref. 1, p. 105]

$$A = \begin{bmatrix} 0.12 & & & & \\ & 11 & & & \\ & & 0 & & \\ & & & 0 & \\ & & & & 10^{-5} \end{bmatrix}$$

$$B = 1$$

Noise statistics--

Katz originally assumed a zero mean normal distribution for the wind gust noise (unit variance). This choice yields an oblivious filter, and forced the introduction of  $\Gamma_4$ , whose value appears on page 109, Ref. 1.

Measurement noise statistics are given on page 85, Ref. 1.

Results:

Pole locations are given on pages 50, 58, 78, 96, 105, 109, 132, and 144, Ref. 1.

In interpreting these results, carefully determine

- 1)  $k_b$
- 2)  $T$ , the sample rate
- 3)  $A$
- 4)  $\Gamma_4$

With the proper choice of these parameters, equations [1-7], and Katz's coefficients yield his results to within a few percent.

Addendum:

Reference 7 became available after this writeup was completed. Longitudinal equations for the FH example occur in two places, pages 2-4 to 2-5, and page 2-61. Two points need to be made:

- 1) Katz assumed  $Z_\alpha = U_0 Z_w$ , etc. I have not studied this derivation, but it should be noted that the authors of the basis equations-- Sutton, et al.-- apparently used the transformation

$$Z_\alpha = U_0 Z_w / 57.3$$

- 2) The conjectures concerning the inconsistency between equation [1], [11], and [14] is reinforced if we look at equation (1) on page 2-61, where

$$\begin{aligned} \dot{q} &= M_q q + M_\alpha \dot{\alpha} + M(\alpha - \alpha_g) + M_{\delta_e} \delta_e \\ \dot{\alpha} &= (1 + Z_q)q + Z_\alpha(\alpha - \alpha_g) + M_{\delta_e} \delta_e \quad Z_q \rightarrow 0 \end{aligned}$$

where  $\alpha_g$  is the angle of attack due to still air. Katz used, instead,

$$\begin{aligned} \dot{q} &= M_q q + M_\alpha(\dot{\alpha} - \dot{\alpha}_g) + M(\alpha - \alpha_g) + M_{\delta_e} \delta_e, \text{ etc.} \\ \text{but } \dot{\alpha}_g &\neq 0 \end{aligned}$$

References

- [1] Selection of Sampling Rate for Digital Control of Aircraft;  
Katz and Powell, Sudaar 486.
- [2] Navy Digital Flight Control System Development, Borow et al.,  
Honeywell Document #21857-FR.
- [3] Feasibility Study for DIGIFLIC, Vol. II Sutton, et al., LSI  
Technical Report #ADR-773.
- [4] DIGIFLIC Systems for Tactical Fighters, VI, Honeywell, Technical  
Report AFFDL-TR-73-119.
- [5] \_\_\_\_\_, Vol. 2.
- [6] \_\_\_\_\_, Final Report, Technical  
Report AFFDL-TR-74-69.
- [7] Feasibility Study for DIGIFLIC, Vol. I, Sutton et al., LSI Technical  
Report #ADR-773.

## APPENDIX C2

## STAR TRACKING TELESCOPE

(Original Source: reference [Powell, 1978])

The Spacelab Infrared Telescope facility [Ref. 1] is designed to allow astronomers to see more than a factor of ten deeper into the universe than is currently possible with ground based telescopes. The video-inertial pointing technology for this system has been under development at Ames Research Center for several years [2,3,4]. The design combines gyro and video information to arrive at an attitude estimate which possesses the best characteristics of each sensor [5].

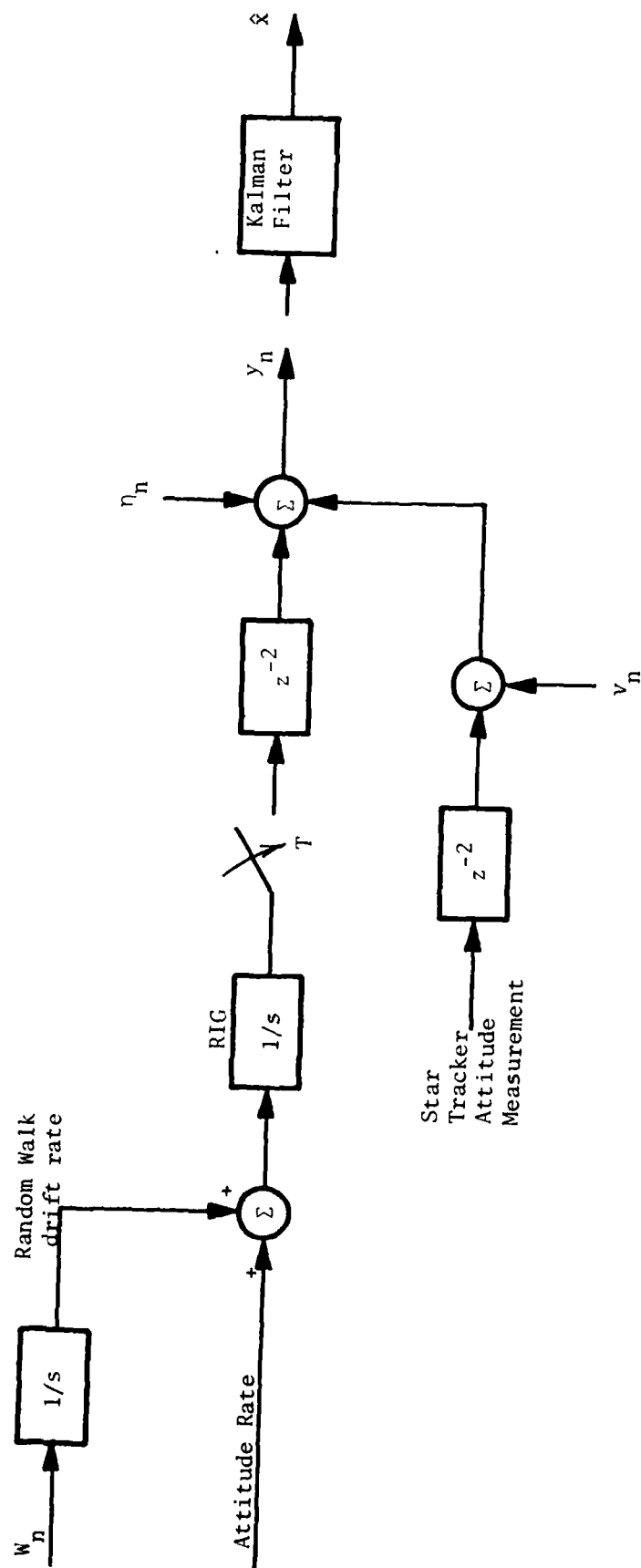
The example in this appendix focuses on the estimation of the gyro drift using delayed measurements from a video star-tracker. We will assume the video sensor and the processor implementing the filter are synchronous in operation, and have comparable delays. The analysis is for steady-state, single-axis fine pointing of the telescope; slew and other induced disturbances are not considered.

Although the gyro noise is not actually white in practice, after sampling the disturbance noise is apparently uncorrelated [5].

The filtering problem and gyro noise model appear in Figure C2.1, and can be modeled as a two state system:

$$\begin{bmatrix} \Theta_D \\ D \end{bmatrix}_{n+1} = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \Theta_D \\ D \end{bmatrix}_n + \begin{bmatrix} T^2/2 \\ T \end{bmatrix} w_n$$

$$y_n = [1 \quad 0] \begin{bmatrix} \Theta_D \\ D \end{bmatrix}_{n-2} - v_n + \eta_n$$



$w_n, \eta_n, v_n$  - White noise

RIG - rate integrating gyro

$y_n$  - random walk drift error measurement

$\hat{x}$  - estimate of drift error

Figure C.2.1 OPTIMAL DRIFT ESTIMATION AND GYRO NOISE MODEL

where

$T$  = the star tracker integration time and sample time in seconds;

$\theta_{Dn}$  = the random walk drift error of the rate integrating gyro output at time  $nT$  seconds;

$D_n$  = the random walk drift rate at  $nT$  seconds;

$w_n$  = a discrete white random sequence driving the random walk drift rate (units are arcseconds/second);

$z_n$  = the delayed drift error measurement in arcseconds;

$v_n$  = the discrete random noise in the star tracker measurement in arcseconds;

$\eta_n$  = the discrete random noise in the RIG measurement resulting from high frequency rate noise (units are arcseconds).

The covariances of the discrete random sequences are defined as:

$$N = \langle w_n^2 \rangle, \quad \text{related to the rms gyro drift}$$

$$R = \langle v_n^2 \rangle, \quad \text{mean square star tracker error}$$

$$Q = \langle \eta_n^2 \rangle, \quad \text{mean square high frequency RIG (Rate Integrating Gyro) output noise.}$$

where  $\langle \rangle$  denotes an ensemble average. We further assume that all noise sources are uncorrelated, that is

$$\langle w_n v_n \rangle = 0$$

$$\langle w_n \eta_n \rangle = 0$$

$$\langle v_n \eta_n \rangle = 0$$

and we further assume the variables are zero mean.

For the example used in Chapter V, the following coefficients were used:

$$T = 1 \text{ second}$$

$$N = 0.5 \text{ arcsecond/second}, \quad \text{or varied}$$

$$(R+Q) = 1.0 \text{ arcseconds}$$

References:

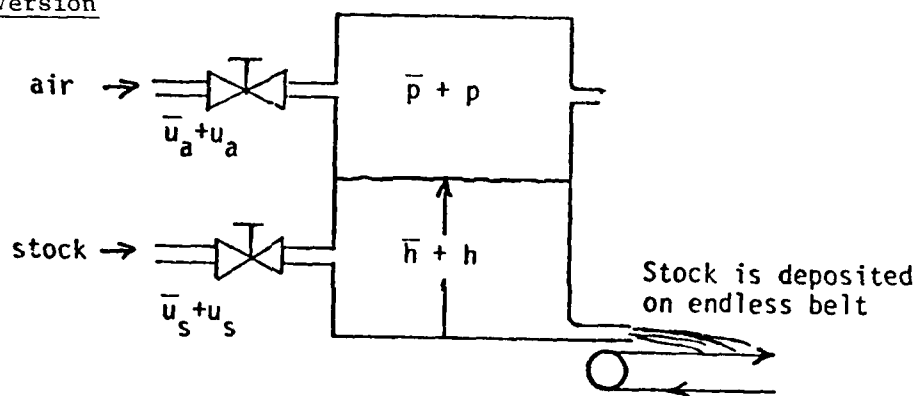
- [1] F.C. Witteborn and L.S. Young, "A Cooled Infrared Telescope for the Space Shuttle-- The Spacelab Infrared Telescope Facility (SIRTF)," AIAA Paper No. 76-174, January 1976.
- [2] K.R. Lorell, J.P. Murphy, C.D. Swift, "A Computer Aided Telescope Pointing System Utilizing a Video Star Tracker," 7th IFAC Symposium on Automatic Control in Space, Rottach-Egern, Germany, May 17, 1976.
- [3] J.P. Murphy, "Multi Star Processing and Gyro Filtering for the Video Inertial Pointing System," NASA TMX-73, 130.
- [4] J.D. Powell and A.J. Throckmorton, "A Study of Filter Mechanizations for the Video Inertial Pointing System Microprocessor," Final Report to NASA-Ames University Consortium, October 1975.
- [5] J.D. Powell and E. Parsons, "Control System Concepts for the Spacelab Infrared Telescope Facility," Final Report to NASA-Ames Research Center, January 1978.

## Appendix C3

## PAPER MACHINE MODEL

Control of a pressurized flow-box for a paper machine.

This example was originally presented by MacFarlane [MacFarlane, 1972], and subsequently was written up by A.E. Bryson [personal notes, 11/72]. Later, J.D. Powell modified the parameters, as an academic exercise, to couple the states.

Bryson's Version

$H = h + \frac{p}{\rho_s g}$  = total head at slice (perturbation in),

$h$  = perturbation in stock level,

$u_a$  = perturbation in air valve opening,

$u_s$  = perturbation in stock valve opening,

$(\bar{\phantom{x}})$  denotes mean values

Plant Model

$$\begin{bmatrix} \dot{H} \\ \dot{h} \end{bmatrix} = \begin{bmatrix} -a_1 & a_2 \\ -a_3 & 0 \end{bmatrix} \begin{bmatrix} H \\ h \end{bmatrix} + \begin{bmatrix} b_1 & b_2 \\ 0 & b_3 \end{bmatrix} \begin{bmatrix} u_a \\ u_s \end{bmatrix} \quad \leftarrow a_i \text{'s and } b_i \text{'s constants}$$



Control Objectives

- (a) To keep  $H$  and  $h$  near zero in presence of disturbances using a small range of available  $u_a, u_s$ .
- (b) To be able to command small changes in  $H$  and  $h$  separately with quick response and good steady-state accuracy.

Numerical Example

For  $a_1 = .395 \text{ sec}^{-1}$ ,  $a_2 = .0115 \text{ sec}^{-1}$ ,  $a_3 = .011 \text{ sec}^{-1}$

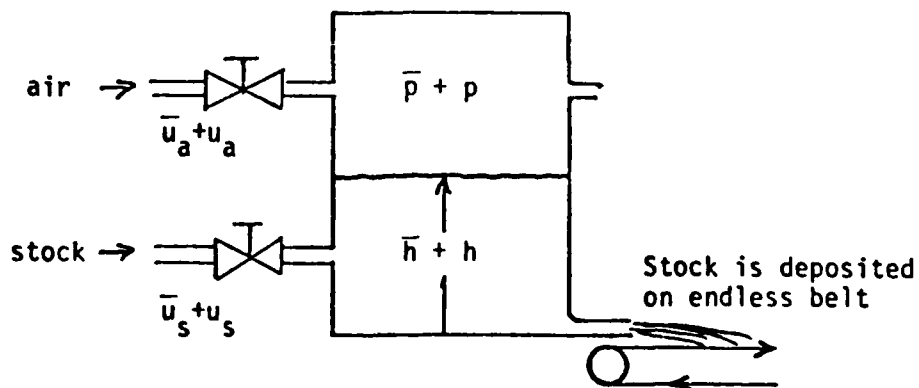
$b_1 = 1.038 \text{ inches of } H_2O \text{ per sec per unit of } u_a$

$b_2 = .0336 \text{ inches of } H_2O \text{ per sec per unit of } u_s$

$b_3 = .000966 \text{ inches of } H_2O \text{ per sec per unit of } u_s$

and the choice that we are willing to commit 100 units of  $u_a$  for  $H = 1 \text{ inch } H_2O$  and 1000 units of  $u_s$  for  $h = 1 \text{ inch } H_2O$ , and the choice that we want the system to "settle" for an  $H_c$  within 2 sec. and for an  $h_c$  command in 5 sec., and the choice that levels of  $H$  and  $h$  be about the same, a good choice of weighting parameters for the quadratic performance is:

$$A_h = 1, A_{c_H} = \frac{1}{4} \text{ sec}^2, A_{e_n} = \frac{1}{25} \text{ sec}^2, B_a = \left(\frac{1}{100}\right)^2, B_s = \frac{1}{(1000)^2}$$

Powell's modified version:

$$\begin{bmatrix} \dot{H} \\ \dot{h} \\ \dot{u}_a \end{bmatrix} = \begin{bmatrix} -0.2 & +0.1 & +1 \\ -0.05 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} H \\ h \\ u_a \end{bmatrix} + \begin{bmatrix} 0 & 1 \\ 0 & 0.7 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} u_c \\ u_s \end{bmatrix}$$

where  $H = h + \frac{P}{P_s g}$  = total head perturbation, in meters

$h$  = stock level perturbation, in meters

$u_a$  = perturbation in air valve opening

$u_c$  = command value to air valve, in /sec

$u_s$  = perturbation in stock valve opening, in kg/sec

$(\bar{\phantom{x}})$  = denotes mean value

$$y = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} H \\ h \\ u_a \end{bmatrix}$$

This system was discretized using a sample rate of five Hertz to produce the following model (from Abbas Emami):

$$x_{i+1} = \phi x_i + \Gamma_1 u_i + \Gamma_2 w_i \quad \mathbb{E}(w_i w_j^T) = Q$$

$$y_i = H x_i + v_i \quad \mathbb{E}(v_i v_j^T) = R$$

$$J = \Sigma (x_i^T A x_i + u_i^T B u_i)$$

where  $u_i$  is the deterministic input. The parameters are

$$\phi = \begin{bmatrix} 0.96069 & 0.019605 & 0.17757 \\ -.0098023 & 0.9999 & -.00092396 \\ 0 & 0 & 0.81873 \end{bmatrix}$$

$$\Gamma_1 = \begin{bmatrix} .018479 & .19743 \\ -6.2 \times 10^{-5} & .139 \\ .18127 & 0 \end{bmatrix} \quad \Gamma_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$H = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 1 \end{bmatrix} \times 10^3$$

$$B = \begin{bmatrix} 100 & 0 \\ 0 & 1 \end{bmatrix}$$

$$Q = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \times 10^{-3}$$

$$R = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \times 10^{-4}$$

### References

MacFarlane, A.G.J. "Notes on the Vector Frequency Response Approach to Analysis and Design of Multivariable Feedback Systems," August, 1972.

## Appendix C4

## STANFORD RELATIVITY SATELLITE [Hadass, 1974]

The plant used in this example was originally investigated by Bull [Ref. 1]. The governing equations for an approximate model are:

$$\begin{aligned} I_1 \ddot{\theta} &= k\phi + w_1 \\ I_2(\ddot{\phi} + \ddot{\theta}) &= -k\phi + U - w_1 + w_2 \end{aligned}$$

and a pictorial representation can be found in Figure C.4.1.

From Hadass, we get the following state space description:

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} \theta \\ \dot{\theta} \\ \alpha \\ \dot{\alpha} \end{bmatrix}$$

$$\dot{x} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & \alpha\omega_o^2 & b\alpha/I_2 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -\omega_o^2 & -b\alpha/I_\alpha \end{bmatrix} x + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1/I_2 \end{bmatrix} u + \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} w$$

$$y = [1 \quad 0 \quad 0 \quad 0]w + v$$

where  $\omega_o^2 = k/I_3$ ,  $\alpha\omega_o^2 = k/I_1$ , and  $I_3 = (I_1 I_2)/(I_1 + I_2)$ . Values chosen included

$$\begin{aligned} \alpha\omega_o^2 &= 19.375 \text{ sec}^{-2} & b\alpha/I_2 &= 1.3175 \times 10^{-2} \\ \omega_o^2 &= 25 \text{ sec}^{-2} & -b\alpha/I_\alpha &= -1.7 \times 10^{-2} \\ I_2 &= 250 \text{ kg m}^2 \end{aligned}$$

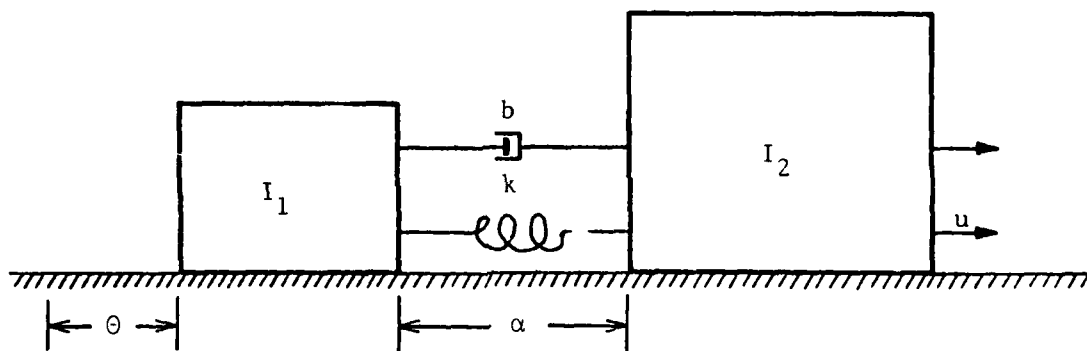


Figure C.4.1      SIMPLIFIED MODEL OF THE STANFORD RELATIVITY SATELLITE

$$\mathbb{E}(ww^T) = Q = \begin{bmatrix} 6.6 & 0 \\ 0 & 2.0 \end{bmatrix} \times 10^{-2} [\text{rad}^2 \text{ sec}^{-3}]$$

$$\mathbb{E}(vv^T) = R = 3.4 \times 10^{-6} \text{ rad}^2 \text{ sec}$$

The cost function weighting matrices were

$$A = \begin{bmatrix} 1.2 \times 10^4 & 0 & 0 & 0 \\ 0 & 5 & 0 & 0 \\ 0 & 0 & 6.5 \times 10^5 & 0 \\ 0 & 0 & 0 & 6.5 \times 10^3 \end{bmatrix}$$

Note that this data was chosen to match Hadass's results, and differ in details from the models he describes.

## REFERENCES

- Acton, F., [1970], Numerical Methods that Work, Harper and Row.
- Anderson, B.D.O., [1977], "Second Order Convergent Algorithms for the Steady-State Riccati Equations," IEEE Conference on Decision and Control, pp. 948-953.
- Bar-Shalom, Y. and Sivan, R., [1969], "On Linear Systems with Random Parameters," IEEE Transactions on Automatic Control, AC-14, 1, February 1969.
- Berger, C.S., [1973], "Numerical Method for the Design of Insensitive Control Systems," IEEE Proceedings, vol. 120, no. 10, pp. 1283-1292, October 1973.
- Bierman, G.J., [1977], Factorization Methods for Discrete Sequential Estimation, Academic Press.
- Bierman, G.J. and Thornton, C.L., [1977], "Numerical Comparison of Kalman Filter Algorithms: Orbit Determination Case Study," Automatica, vol. 13, no. 1, pp. 23-36, January 1977.
- Bierman, G.J. and Sidhu, G.S., [1977], "Integration Free Interval Doubling for Riccati Equation Solutions," IEEE Transactions on Automatic Control, vol. AC-20, no. 5, pp. 831-834, October 1977.
- Brammer, K., [1968], "Lower Order Optimal Linear Filtering of Non-stationary Random Sequences," IEEE Transactions on Automatic Control, vol. AC-13, no. 4, pp. 198-199, April 1968.
- Bryson, A.E. and Johansen, D.E., [1965], "Linear Filtering for Time-Varying Systems Using Measurements Containing Colored Noise," IEEE Transactions on Automatic Control, vol. AC-10, pp. 4-10, January 1965.
- Bryson, A.E. and Henrikson, L.J., [1968], "Estimation Using Sampled Data Containing Sequentially Correlated Noise," J. Spacecraft and Rockets, vol. 5, pp. 662-665, June 1968.
- Bryson, A.E. and Hall, W.E., [1971], "Optimal Control and Filtering Synthesis by Eigenvector Decomposition," Sudaar Report no. 436, Stanford University, Department of Aero. & Astro., Stanford, California, November 1971.
- Bryson, A.E. and Yu-Chi Ho, [1975], Applied Optimal Control, John Wiley and Sons.
- Bryson, A.E., [1977], "Kalman Filter Divergence and Aircraft Motion Estimators," Sudaar 505, Department of Aero. & Astro., Stanford University, July 1977.

- Bryson, A.E., [1978], Personal communication, February, 1978.
- Bucy, R.S., Rappaport, D. and Silverman, L.M., [1970], "Correlated Noise Filtering and Invariant Directions for the Riccati Equation," IEEE Transactions on Automatic Control, vol. AC-15, pp. 535-540, October, 1970.
- Burckhardt, J., [1860], Die Cultur der Renaissance in Italien: Ein Versuch, Leipzig, Germany.
- \_\_\_\_\_, [1929], The Civilization of the Renaissance in Italy, Harper Torch Books.
- Claerbout, J.F., [1976], Fundamentals of Geophysical Data Processing with Applications to Petroleum Prospecting, McGraw-Hill.
- Davison, E.J. and Wang, S.H., [1973], "Properties of Linear Time-Invariant Multivariable Systems Subject to Arbitrary Output and State Feedback," IEEE Transactions, vol. AC-18, no. 1, pp. 24-32, February, 1973.
- Denman, E.D. and Beavers, A.N., [1976], "The Matrix Sign Functions and Computations in Systems," Applied Mathematics and Computers, vol. 2, pp. 63-94.
- Edleston, J., Ed., [1850], Correspondence of Sir Isaac Newton and Professor Cotes, John W. Parker, London.
- Forsythe, G. and Moler, C., [1967], Computer Solutions of Linear Algebraic Systems, Prentice Hall.
- Friedlander, B., Kailath, T. and Ljung, L., [1976], "Scattering Theory and Linear Least Squares Estimation II, Discrete Time Problems," Journal of the Franklin Institute, vol. 301, no. 1 & 2, pp. 71-82, January, 1976.
- Friedlander, B., [1976], "Scattering Theory and Linear Least Squares Estimation," Ph.D. Thesis, Department of Electrical Engineering, Stanford University, Stanford, California, August, 1976.
- Geesey, R. and Kailath, T.K., [1973], "An Innovations Approach to Least Squares Estimation-- Part V: Innovations and Recursive Estimation in Colored Noise," IEEE Transactions Automatic Control, vol. AC-18, pp. 435-453, October, 1973.
- Gevers, M.R., [1972], "Structural Properties of Realizations of Discrete Time Markovian Process," Ph.D. Dissertation, Department of Electrical Engineering, Stanford University, California; also SEL Technical Report, no. 7050-19, May, 1972.
- Gevers, M.R., and Kailath, T., [1973], "Constant, Predictable, and Degenerate Directions of the Discrete-time Riccati Equation," Automatica, vol 9, pp. 699-712, November, 1973.
- Golub, G.H., [1965], "Numerical Methods for Solving Linear Least Squares Problems," Numer. Math., vol. 7, pp. 206-216.



- Hadass, Z., [1974], "Design of State-Feedback Controllers Including Sensitivity Pointing," Ph.D. Dissertation, Department of Aero. and Astro., Stanford University, California; also Sudaar Technical Report no. 482.
- Hitz, K.L. and Anderson, B.D.O., [1972], "An Iterative Method of Computing the Limiting Solution of the Matrix Riccati Differential Equation," IEEE Proceedings, vol. 119, no. 9, pp. 1402-1403, Sept., 1972.
- Huizinga, J., [1924], The Waning of the Middle Ages, Doubleday Anchor Books.
- Kailath, T., [1974], "A View of Three Decades of Linear Filtering Theory," IEEE Transactions on Information Theory, IT-20, pp. 145-181.
- Kalman, R.E., [1961], "A New Approach to Linear Filtering and Prediction Problems," J. Basic Eng., vol. 83, pp. 34-45.
- \_\_\_\_\_, [1963], "New Methods of Wiener Filtering Theory," Proceedings of 1st Symp. Engineering Applications of Random Function Theory and Probability, J.L. Bogdanoff and F. Kozin Eds., New York: Wiley, pp. 270-388; also RIAS, Baltimore, Md., Technical Report 61-1.
- Kaminski, P., [1971], "Square Root Filtering and Smoothing for Discrete Processes," Ph.D. Thesis, Department of Aero. and Astro., Stanford University, California, September, 1971.
- Kaminski, P., Bryson, A. and Schmidt, S., [1971], "Discrete Square Root Filtering: A Survey of Current Techniques," IEEE Transactions on Automatic Control, vol. AC-16, no. 6, pp. 727-735, December, 1971.
- Katz, P., [1974], "Selection of Sample Rate for Digital Control of Aircraft," Ph.D. Dissertation, Department of Aero. and Astro., Stanford University, California; also Sudaar Technical Report no. 486, September, 1974.
- Kwakernaak, H. and Sivan, R., [1972], Linear Optimal Control Systems, Wiley-Interscience.
- Lainiotis, D.G., [1971], "Optimal Nonlinear Estimation," Int. J. Control, vol. 14, no. 6, pp. 1137-1148.
- \_\_\_\_\_, [1976], "A Unifying Framework for Linear Estimation: Generalized Partitioned Algorithms," J. Information Sciences, vol. 10, pp. 243-276, April, 1976.
- Laub, A.J., [1979], "A Schur Method for Solving Algebraic Riccati Equations," IEEE Conference on Decision and Control, pp. 60-65.
- Lawson, C.L. and Hanson, R.J., [1974], Solving Least Squares Problems, Prentice Hall.

- Ljung, L., Kailath, T. and Friedlander, B., [1976], "Scattering Theory and Linear Least Squares Estimation, Part I: Continuous-Time Problems," IEEE Proceedings, vol. 64, no. 1, pp. 131-139, January, 1976.
- Mason, S.J. and Zimmermann, H.J., [1960], Electronic Circuits, Signals, and Systems, J. Wiley and Sons.
- Moler, C., [1974], Matrix Eigenvalue and Least Squares Computations, Computer Science Department, Stanford University, California, March, 1974.
- Morf, M. and Kailath, T., [1975], "Square Root Algorithms for Least Squares Estimation," IEEE Transactions on Automatic Control, vol. AC-20, no. 4, August, 1975.
- Morf, M., Dobbins, J.R., Friedlander, B., and Kailath, T., [1978], "Square Root Algorithms for Parallel Processing in Optimal Estimation," to appear in Automatica, May, 1979.
- Newton, Issac, [1926], Philosophe Naturalis Principia Mathematica, H. Pemberton, Ed. (G. & J. Innys, London, ed. 3).
- \_\_\_\_\_, [1934], Mathematical Principles of Natural Philosophy, A. Motte, Translation, F. Cajori, Ed., University of California, Berkeley.
- Falsson, T. and Whitaker, [1972], "Parameter Uncertainties in Control Systems Design," Proceedings JACC, pp. 248-254.
- Potter, J.E., [1966], "Matrix Quadratic Solutions," SIAM Journal of Applied Mathematics, vol. 14, no. 3, May, 1966.
- Powell, J.D., [1976], Personal communication.
- Fowell, J.D. and Parsons, E., [1978], Final Report Submitted to NASA Ames Research Center, Moffet Field, California, by Guidance and Control Laboratory, Department of Aero. and Astro., Stanford University, California, January, 1978; also Sudaar No. 510.
- Rappaport, D., [1969], "Constant Directions of the Riccati Equations," Ph.D. Thesis, Department of Electrical Engineering, University of Southern California, August, 1969.
- Redheffer, R.M., [1950], "Remarks on the Basis of Network Theory," J. Mathematics and Physics., 28, pp. 237-258.
- \_\_\_\_\_, [1954], "Novel Uses of Functional Equations," J. Rational Mech. Anal., 3, pp. 271-279.
- \_\_\_\_\_, [1956], "On Solutions of Riccati's equation as Functions of Initial Values," J. Rational Mech. Anal., pp. 835-848.

- Redheffer, R.M., [1957], "The Ricatti Equation: Initial Values and Inequalities for a Matrix," Math. Ann., 133, pp. 235-250.
- \_\_\_\_\_, [1959], "Inequalities for a Matrix Riccati Equation," J. Math. Mech., 8, pp. 349-367.
- \_\_\_\_\_, [1960], "On a Certain Linear Fractional Transformation," J. Math. & Phys., 39, pp. 269-286.
- \_\_\_\_\_, [1960], "Supplementary Note on Matrix Riccati Equations," J. Math. Mech., 9, pp. 745-748.
- \_\_\_\_\_, [1960], "The Mycielsky-Paszkowski Diffusion Problem," J. Math. Mech., 9, pp. 607-621.
- \_\_\_\_\_, [1961], "Difference Equations in Transmission-Line Theory," Modern Mathematics for the Engineer, Second Series, E.F. Beckenbach (Ed.), pp. 282-354, McGraw-Hill, New York.
- \_\_\_\_\_, [1962], "On the Relation of Transmission-Line Theory to Scattering and Transfer," J. Math. Phys., 41, pp. 1-41.
- Redheffer, R.M. and Wang, A.P., [1970], "Formal Properties of Time-Dependent Scattering Processes," J. Math. and Mech., 19, pp. 765-781.
- Roberts, J.D., [1971], "Linear Model of Algebraic Riccati Equations by Use of the Sign Function," CUED/B-Control/TR-15 Rept., Cambridge University.
- Sage, A.P. and Melsa, J.L., [1971], Estimation Theory with Applications to Communications and Control, McGraw-Hill.
- Sameh, A.H. and Kuck, D.J., [1977], "A Parallel QR Algorithm for Symmetric Tridiagonal Matrices," IEEE Transactions on Computers, vol. C-26, no. 2, pp. 147-152.
- Santillana, G. and von Dechend, H., [1969], Hamlet's Mill, Gambit, Ipswich.
- Schmidt, S.F., [1970], "Computational Techniques in Kalman Filtering," Nato-AGARD-139, February, 1970.
- Smith, E.W. and Davison, E.J., [1972], "Design of Industrial Regulators. Integral Feedback and Feedforward Control," IEEE Proceedings, vol. 119, no. 8, pp. 1210-1216, August, 1972.
- Smith, E.W., [1977], Personal communication with Abbas Emami.
- Stewart, G.W., [1973], "Error and Perturbation Bounds for Subspaces Associated with Certain Eigenvalue Problems," Siam Review, vol. 15, no. 4, pp. 727-764, October, 1973.

- Strang, G., [1976], Linear Algebra and Its Applications, Academic Press.
- Tse, E. and Athans, M., [1970], "Optimal Minimal-Order Observer-Estimators for Discrete Linear Time-Varying Systems," IEEE Transactions on Automatic Control, vol. AC-15, no. 4, August, 1970, pp. 416-426.
- Van Dooren, P., [1978], Personal communication.
- Vaughn, D.R., [1970], "A Non-Recursive Solution of the Discrete Matrix Riccati Equation," IEEE Trans. on Automatic Control, October, 1970.
- Vergheze, G., Friedlander, B., and Kailath, T., [1978], "Scattering Theory and Linear Least-Square Estimation, Part III: The Estimates," to appear in IEEE Trans. on Automatic Control.
- Vieira, A., [1977], "Matrix Orthogonal Polynomials, with Applications to Auto Regressive Modeling and Ladder Forms," Ph.D. Thesis, Department of Electrical Engineering, Stanford University, Stanford, California, December, 1977.
- Wiggins, R.A. and Robinson, E.A., [1965], "Recursive Solution to the Multichannel Filtering Problem," J. Geophysical Res., vol. 70, no. 8, pp. 1885-1891.
- Wilkinson, J.H., [1965], The Algebraic Eigenvalue Problem, Clarendon Press, Oxford.
- Womble, M. and Potter, J., [1975], "A Prefiltering Version of the Kalman Filter and New Numerical Integration Formulas for Riccati Equations," IEEE Transactions on Automatic Control, vol. AC-20, no. 3, pp. 378-380.