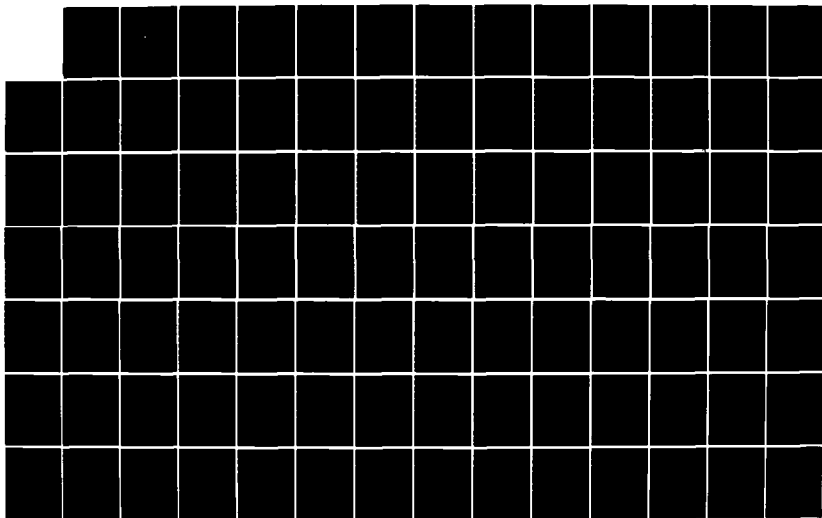AD-A148 289   UTILIZATION OF NUMERICAL OPTIMIZATION TECHNIQUES IN THE   1/2
                DESIGN OF ROBUST..(U) NAVAL POSTGRADUATE SCHOOL
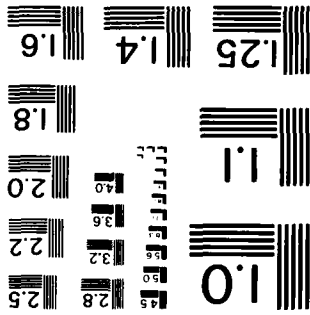   MONTEREY CA   Y C GORDON SEP 84

UNCLASSIFIED   F/G 12/1   NL

MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

1.25   1.4   1.6

1.8

2.0   4.0

1.1

2.2   3.6   3.2

2.5   2.8   1.0

NAVAL POSTGRADUATE SCHOOL
Monterey, California

DEC 4 1984

A

THESIS

UTILIZATION OF NUMERICAL OPTIMIZATION
TECHNIQUES IN THE DESIGN OF ROBUST
MUTLI-INPUT MULTI-OUTPUT CONTROL SYSTEMS

by

Vernon Curtis Gordon

September 1984

Thesis Advisor: D. J. Collins

84   12 03 020

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO. AD-A148 789	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Utilization of Numerical Optimization Techniques in the Design of Robust Multi-Input, Multi-Output Control Systems		5. TYPE OF REPORT & PERIOD COVERED Doctor of Philosophy September 1984
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Vernon Curtis Gordon		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93943		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, California 93943		12. REPORT DATE September 1984
		13. NUMBER OF PAGES 175
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)	-	15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

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

18. SUPPLEMENTARY NOTES

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

Multi-input Multi-output Control, Robustness, Modern Control, Multivariable Systems.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)
A direct design method for solving the problem of robustness to cross-coupling perturbations in multivariable control systems is presented. The method uses numerical optimization procedures to manipulate the system feedback gains as direct design variables. The manipulation is accomplished in a manner that produces desired performance by pole placement and robustness by modification of the minimum singular values of the system return difference matrix.

Channels affected by cross-coupling perturbation may be recognized by the character of their transfer function plot. The mechanism used by the pole placement and robustness routine in obtaining a robust design is evident from the gain changes associated with the transfer function diagram and the zero shifts shown on pole-zero plots. The pole placement and robustness routine uses gain equalization and zero assignment to modify the characteristics of the system in the areas of low singular values producing a robust design.

A modification of the pole placement and robustness routine that may be applied to the design of robust observers is also presented. Using feedback and filter gains as direct design variables a practical design procedure for robustness recovery in observer based systems is obtained.

Utilization of Numerical Optimization Techniques
in the Design of
Robust Multi-input Multi-output Control Systems

by

Vernon C. Gordon
Commander, United States Navy
B.A.E., Auburn University, 1968
M.S.A.E., Naval Postgraduate School, 1970

Submitted in partial fulfillment of the
requirements for the degree of

DOCTOR OF PHILOSOPHY

from the

NAVAL POSTGRADUATE SCHOOL
· September 1984

Author: _____

APPROVED BY:

_____   _____
M. F. Platzer   T. H. Gawain
Professor of Aeronautics   Professor of Aeronautics

_____   _____
R. D. Strum   G. E. Latta
Professor of Electrical   Professor of Mathematics
Engineering

_____
D. J. Collins
Professor of Aeronautics
Thesis Supervisor

Approved by: _____
              Chairman, Department of Aeronautics

Approved by: _____
              David A. Schrady, Academic Dean

3

ABSTRACT

A direct design method for solving the problem of robustness to cross-coupling perturbations in multivariable control systems is presented. The method uses numerical optimization procedures to manipulate the system feedback gains as direct design variables. The manipulation is accomplished in a manner that produces desired performance by pole placement and robustness by modification of the minimum singular values of the system return difference matrix.

Channels affected by cross-coupling perturbation may be recognized by the character of their transfer function plot. The mechanism used by the pole placement and robustness routine in obtaining a robust design is evident from the gain changes associated with the transfer function diagram and the zero shifts shown on pole-zero plots. The pole placement and robustness routine uses gain equalization and zero assignment to modify the characteristics of the system in the areas of low singular values, producing a robust design.

A modification of the pole placement and robustness routine that may be applied to the design of robust observers is also presented. Using feedback and filter gains as direct design variables a practical design procedure for robustness recovery in observer based systems is obtained.

4

TABLE OF CONTENTS

5

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

For over fifteen years the United States Navy has allowed me to make practical application of the science of flight as a Naval Flight Officer and as a flight test project officer. I wish to express my appreciation to the Navy and the citizens of the United States for the opportunity to further expand my knowledge of aviation by pursuing this degree.

I would also like to express a sincere thank you to the Doctoral Committee of professors Platzer, Gawain, Latta, Strum and Collins for their support in these efforts. Of course special thanks must go to Professor D.J. Collins for heading the committee and having worked so long and hard with me on two theses. Without his support this effort would never have been completed.

Thanks are also expressed to Professor G.N. Vanderplaats for allowing me to use the Automated Design Synthesis code . His help in integrating the code into the pole placement and robustness design program and the insight into the power of numerical optimization that he provided were extremely beneficial to the outcome of this thesis.

The Naval Postgraduate School Library staff and Roger Martin are appreciated tremendously for providing excellent service. No request for reference material was ever to difficult for them to handle. The W.R. Church Computer Center is acknowledged for their support in the computational processing for this thesis. I would also like to acknowledge the work of CDR Al Diel and LTs Cliff Cooksey, John Boden and Mike Iaptas for the work done to provide an excellent control system analysis package at the Naval Postgraduate School. Their efforts made the analysis of data from the pole placement and robustness routine much easier.

10

I would also like to thank my wife, Janice, and children, Bradley and Melissa. As anyone knows who has undertaken an effort of this magnitude without their love and support this would have been impossible. They have "gone it alone" quite often so that this paper could be completed.

Above all thanks to Him who started it all.

I. INTRODUCTION

With the rising interest in multivariable control theory brought on by increasingly complex systems the need has arisen to develop design methods that will allow the designer to specify system performance while at the same time ensuring relatively high stability margins or robustness. In the single-input single-output (SISO) case the designer has had the tools to do these tradeoffs in the form of Nyquist, Bode and root locus plots. In the multi-input multi-output (MIMO) case the classical methods are not totally appropriate.

With the increased interest in MIMO systems numerous methods of design have been employed to obtain suitable system performance and robustness with varying degrees of success. One primary method of design is to keep the plant as decoupled as possible throughout the design so that each individual element may be controlled · independently and designed essentially as a single loop system. Rosenbrock [Ref. 1] has developed a procedure where the multiloop system is modified into a system that has diagonal elements that are much larger than any off-diagonal elements. This diagonally dominant system is then in a form where conventional Nyquist type techniques can be employed in the analysis. A third common MIMO design method is that of the Linear Quadratic (LQ) method. This method uses a quadratic cost functional and optimization principles to allow the designer to design for various performance levels by adjusting the matrix weighting terms used in the cost function. The major difficulty with all of the above methods is that they are not necessarily robust. This is especially true for cross-coupling terms between loops.

12

The primary achievement of this thesis has been the incorporation of the time domain pole placement design procedure with a method of using return difference matrix singular values to improve the robustness of the design. The technique, which utilizes a modern optimization routine, can significantly assist the designer in obtaining robustness in the face of cross-coupling perturbations. It has also been shown that the cross-coupling perturbation problem can be detected by using classical open-loop node diagrams as well as modern control analysis. The pole placement and robustness design routine developed for this thesis has been used on several problems discussed in recent literature. In these studies the pole placement and robustness design code has proven capable of meeting the desired goals of pole placement and robustness and also brought to light some interesting aspects of the cross-coupling perturbation problem. A slightly modified pole placement and robustness routine has proven effective in the design of robust observers.

The remainder of the thesis will present background material on SISO systems in Chapter Two and on MIMO systems in Chapter Three. Optimization will be discussed in Chapter Four along with a discussion of the Automated Design Synthesis (ADS) program used as the optimizer routine for the pole placement and robustness design procedure developed in this thesis. The thesis methodology will be discussed and outlined in Chapter Five. This will be followed by chapters discussing results from selected problems. Conclusions will be presented in the final chapter.

II. <u>SINGLE-INPUT SINGLE-OUTPUT SYSTEMS</u>

The purpose of this thesis research has been to develop a method of obtaining a robust multivariable control system design. A brief review of the concept of robustness and stability in the framework of a conventional SISO system will be done before pursuing the concepts in a more complicated fashion in the following chapters. A simple interpretation of robustness is the ability of the system to tolerate design perturbations. These perturbations could be in the form of actuator failures, plant parameter uncertainty, unmodeled dynamics or nonlinear terms, or any one of many other perturbations to the nominal design of the system.

The primary reason for feedback systems·is the control of uncertainty within the system. By appropriate use of feedback, properties that would lead to an unstable system may be controlled. When stability and robustness aspects are considered for a SISO system, frequency domain design concepts, using either Nyquist or Bode plots, are normally used. Robustness in SISO systems is formulated naturally by the concept of gain and phase margins, both of which are readily available on the Nyquist or Bode diagram.

In figure 2.1 a nominal feedback system can be seen with a perturbation element α (s) placed in series with the nominal system. When α =1 the system is nominal and stable. To determine the positive phase margin of the system the value of $\alpha(s) = \alpha(j\omega) = e^{j\psi}$ will be changed by varying ψ until the system just becomes unstable. This value of ψ will then be the system phase margin. The negative phase margin can be computed in the same way. To find the gain margin the magnitude value of α is increased until the system just
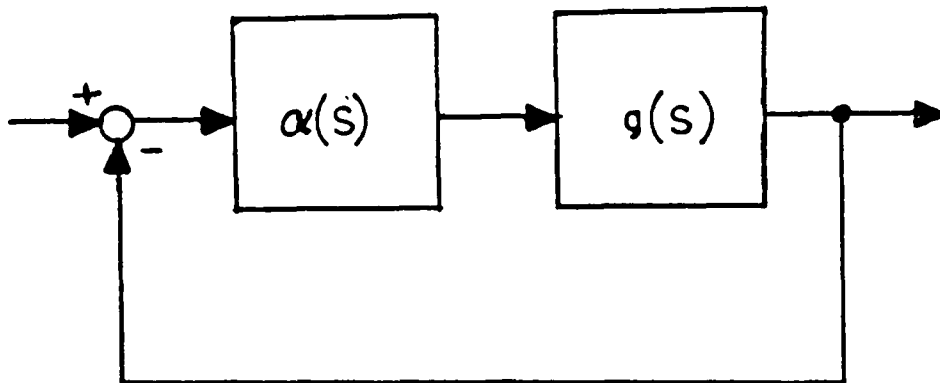
Figure 2.1   Basic Gain and Phase Margin Determination Model.

becomes unstable. This is the upward gain margin. A similar
lower margin is also defined.

These gain and phase margins may not be adequate meas-
ures of robustness [Ref. 2] because they do not account for
simultaneous variation in both gain and phase. Therefore,
while large individual gain or phase changes may not desta-
bilize the system, small simultaneous changes in gain and
phase may destabilize the system. This is not a major diffi-
culty in classical SISO techniques because the effect can be
easily detected.

Gain and phase margin can be defined in terms of the
open-loop frequency domain plots in either the Bode or
Nyquist format. Figure 2.2 depicts a classical Bode plot
showing gain and phase margin determination from the plot.
The Nyquist plot may also be used to obtain this informa-
tion. Nyquist criterion states that if the open-loop
transfer function $G(s)H(s)$ has n poles in the right half
plane and the limit of $G(s)H(s)$=constant as $s \to \infty$ then for a

15
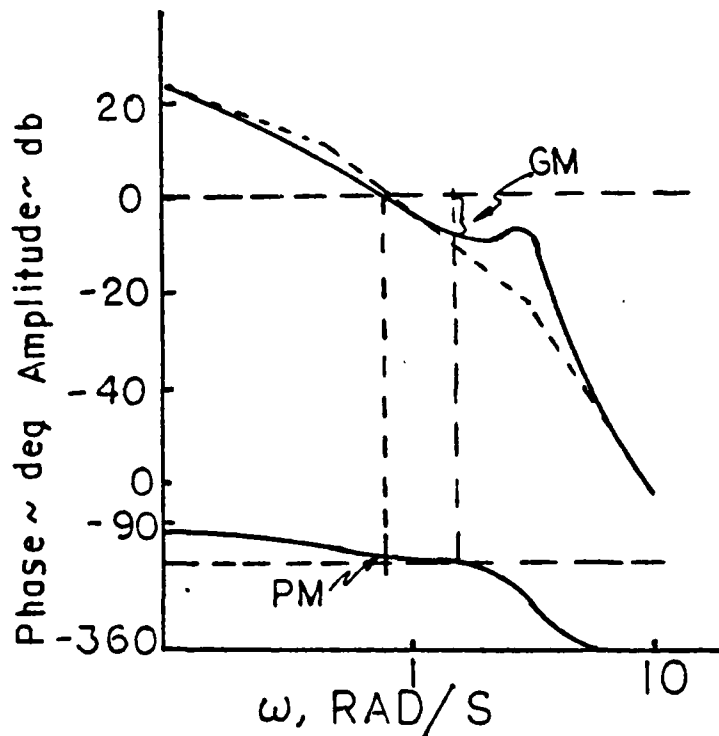
Figure 2.2   Classical Bode Plot.

stable system the locus of G(s)H(s)   will encircle the -1+j0
point n times in the counterclockwise direction as s varies
along the Nyquist ccntour.   If there are no poles in the
right half s plane then the locus will not encircle the -1 +
j0 point. The diagram in figure 2.3 illustrates a nominally
stable system.   The gain and phase margin may be determined
directly from the diagram.

     Any change in the loop transfer function, provided the
order of G(s)H(s) does not change, that changes the number
of times the locus of G(s)H(s) encircles the (-1,0) point in
the Nyquist plot causes the system to become unstable. This
leads to the conclusion that the minimum distance of the
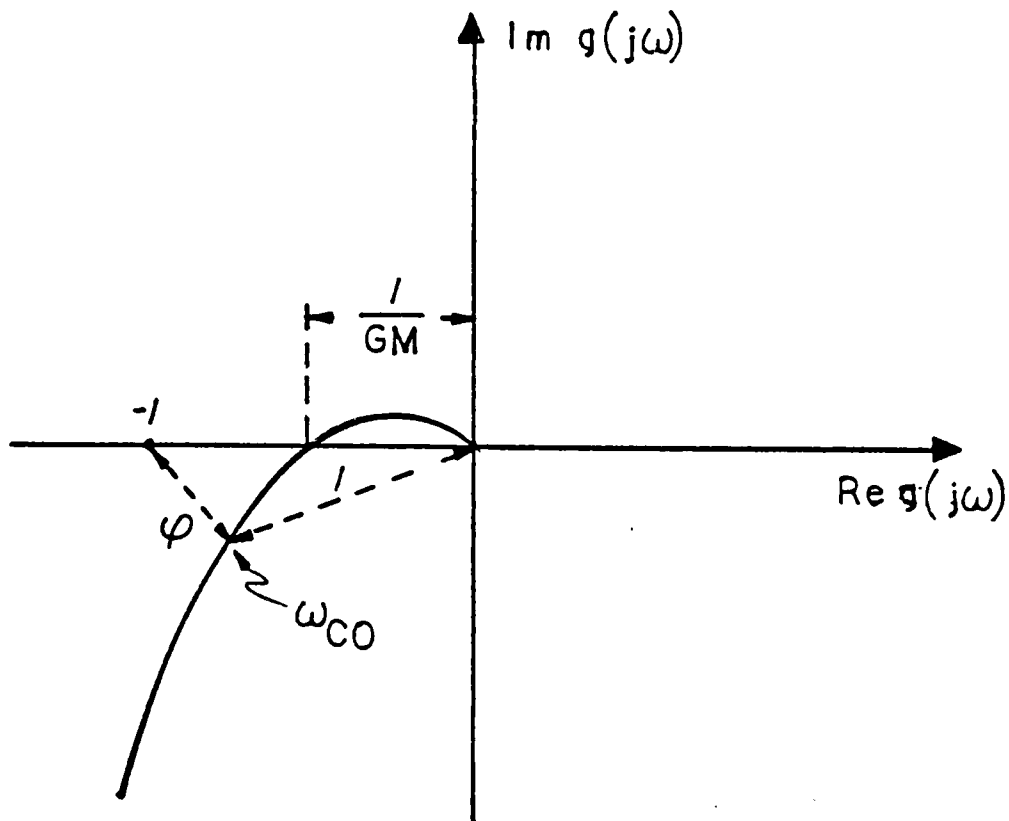
Figure 2.3   Nyquist Plot of Stable System.

locus of G(s)H(s) to the (-1,0)   point is a measure of the
system stability.   This distance   concept carries over
directly to the MIMC system as will be shown in the next
chapter.   Examples cf a multiplicative perturbaticn and an
additive perturbation illustrate this idea. Figure 2.4 is an
additively perturbed system.   Figure   2.5 shows the Nyquist
plot for this system.   Assuming that the plant is itself
stable and the perturtations are also stable the diagram may
then ke used to determine how near the system is to insta-
bility for the given perturbation.

17
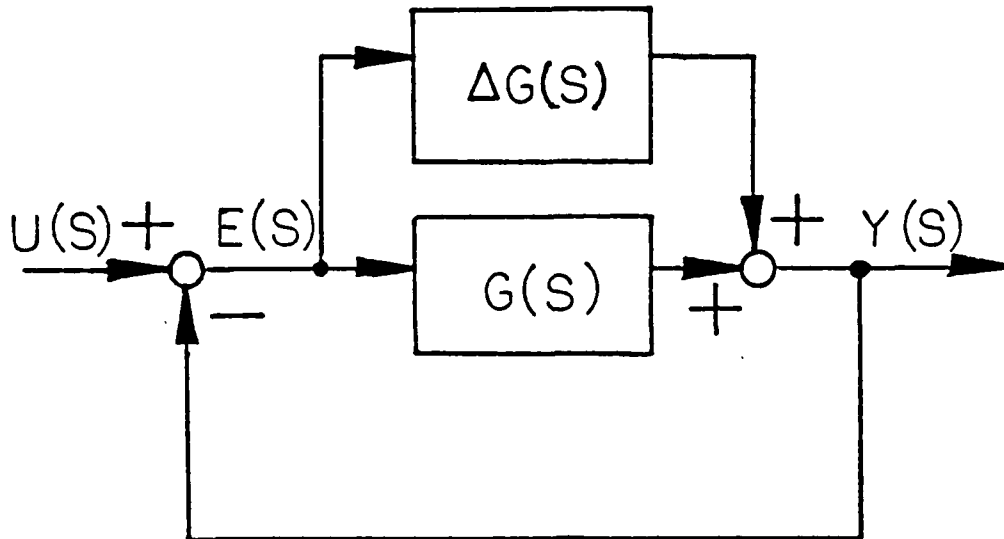
Figure 2.4   Additively Perturbed System.

Since the system is stable the (-1,0) point is encircled
the correct number of times by the nominal plant.   If the
locus of g(jω) in the diagram is warped until it passes
beyond the (-1,0) point then clearly the number of encircle-
ments of this point will change and the system will beccme
unstable, assuming the order of the plant is not changed by
the perturbation. To keep the locus of points from moving
beyond the (-1,0) point equation 2.1 must hold.
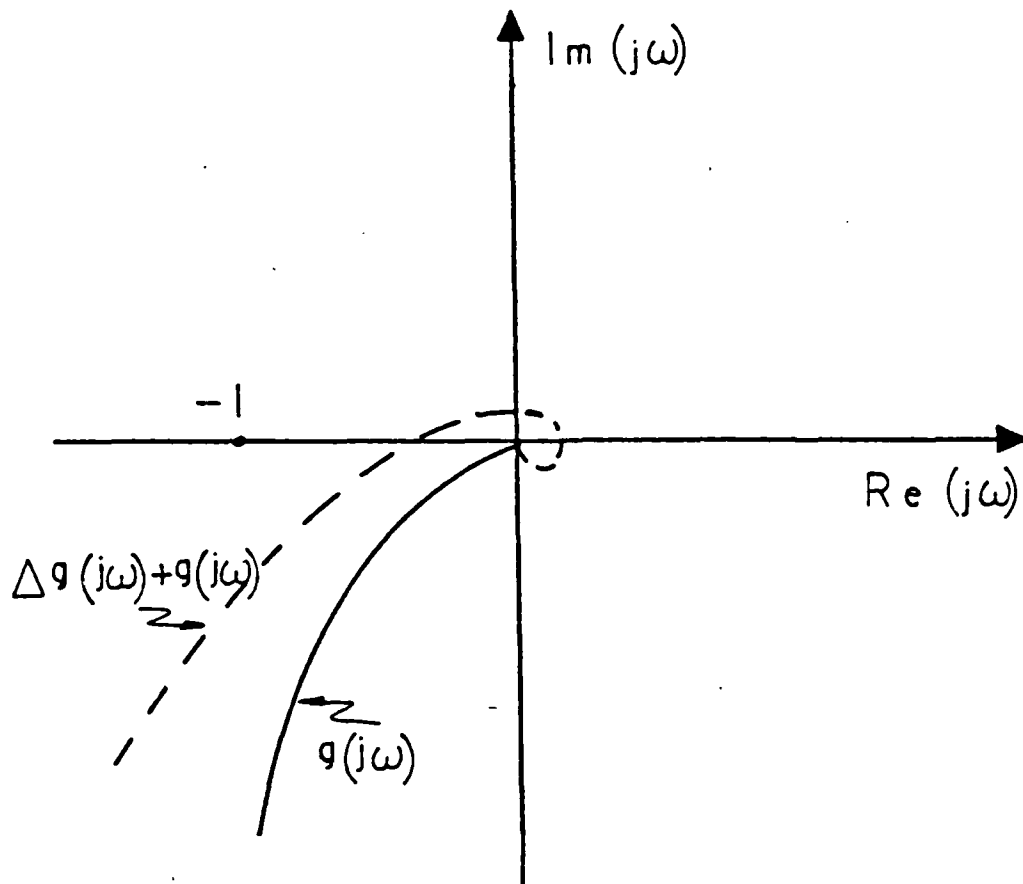
Figure 2.5   Additive Nyquist Plot.

$$| \triangle g(j\omega)| < | 1 + g(j\omega)| \qquad (2.1)$$

This condition is illustrated in figure 2.6. The right-hand side of equation 2.1 is just the magnitude of the return difference transfer function of the nominal system.   The multiplicative case is depicted in figure 2.7 with its associated Nyquist plot in figure 2.8.   The requirement for stability is similar to the additive case and may be stated in equation 2.2

$$| \triangle g(j\omega)| < | 1 + (g(j\omega)^{-1})| \qquad (2.2)$$

19
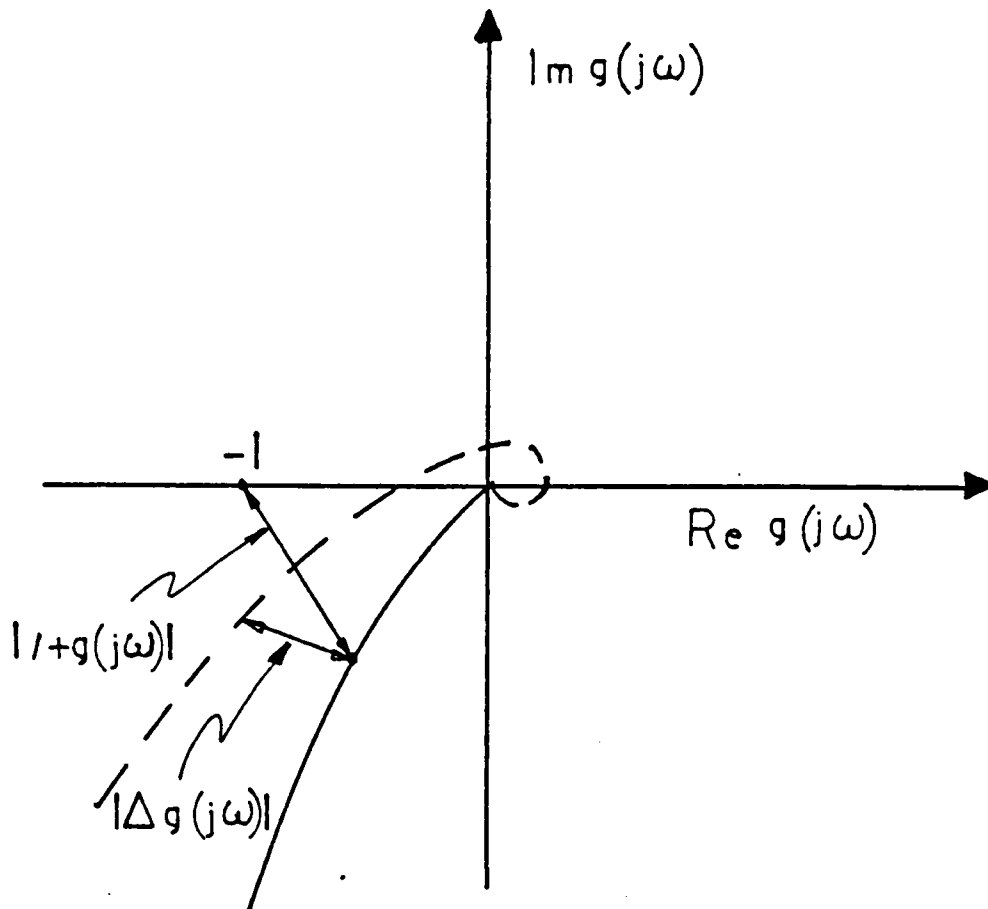
Figure 2.6   Nyquist for Inequality Additive Condition.

The above arguments will be applied again in Chapter 3 to
develop multivariable stability and robustness properties.

The linear quadratic design has been the primary method
employed in modern control design practice.   In this method
an optimal state feedback control law is developed to find a
set of feedback gains that optimizes a chosen performance
index. The performance index for the steady-state case is
given in equation 2.3

20
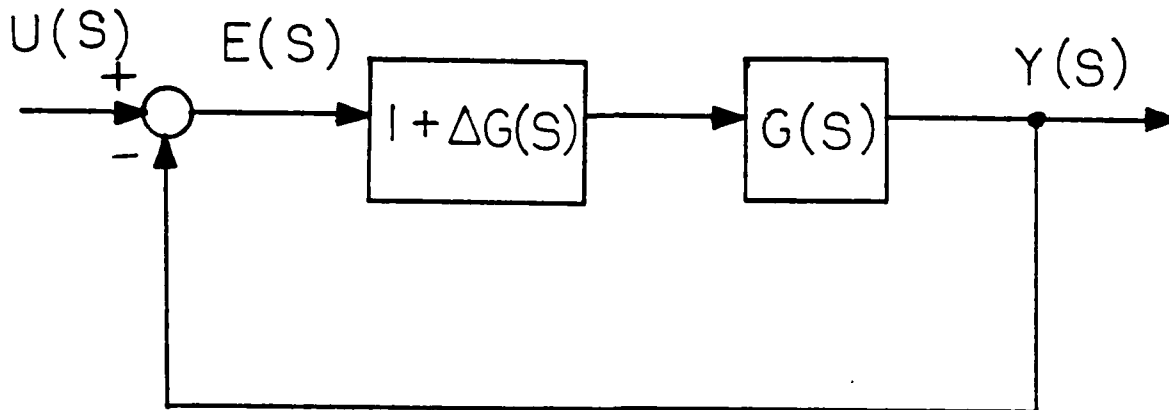
Figure 2.7   Multiplicative System.

$$PI = \int (\underline{x}'\underline{Q}\ \underline{x} + \underline{u}'\underline{R}\ \underline{u})dt \qquad (2.3)$$

where the $\underline{x}'\underline{x}$ term and the $\underline{u}'\underline{u}$ term form quadratics. The $\underline{Q}$ and $\underline{R}$ matrices are chosen by the designer to provide the best compromise between the minimum error of the system and the minimum energy needed to control the system. The LQ method is based on the use of closed-loop state variable feedback for the control of the system. In the MIMO problem LQ methods have been used extensively because of their guaranteed stability margins with diagonal weighting matrices. For diagonal weighting matrices the LQ method yields a guaranteed phase margin of 60 degrees and -6 db to infinite gain margin.

21
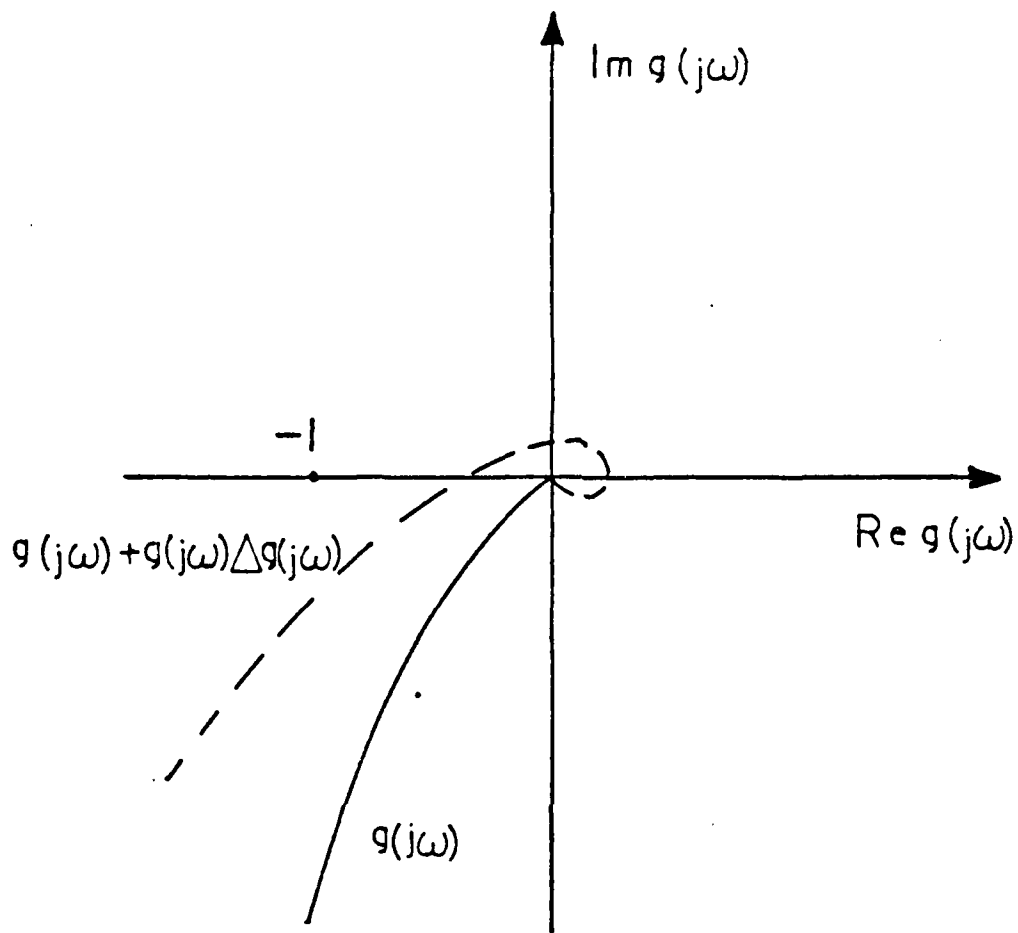
Figure 2.8   Nyquist Plot for Multiplicative System.

For SISO linear quadratic optimal regulators these
stability margins can be developed from inequality 2.4
[Ref. 3]

$$|1 + f'(j\omega I - a)^{-1}t| \geq 1 \qquad (2.4)$$

Writing the Kalman equation as 2.5

$$|1 + f'(sI-a)^{-1}b|^2 = 1 + (1/\rho)|G_K(s)|^2 \qquad (2.5)$$

cne has further

$$|1 + G(s)H(s)|^2 = 1 + (1/\rho)|G_K(s)|^2 \qquad (2.6)$$

and for all $s = j\omega$ and $0 \leq \omega < \infty$ the function $(1/\rho)G_K(s)$ is greater than zero, therefore the Kalman inequality is shown to be 2.7.

$$|1 + G(s)H(s)| > 1 \qquad (2.7)$$
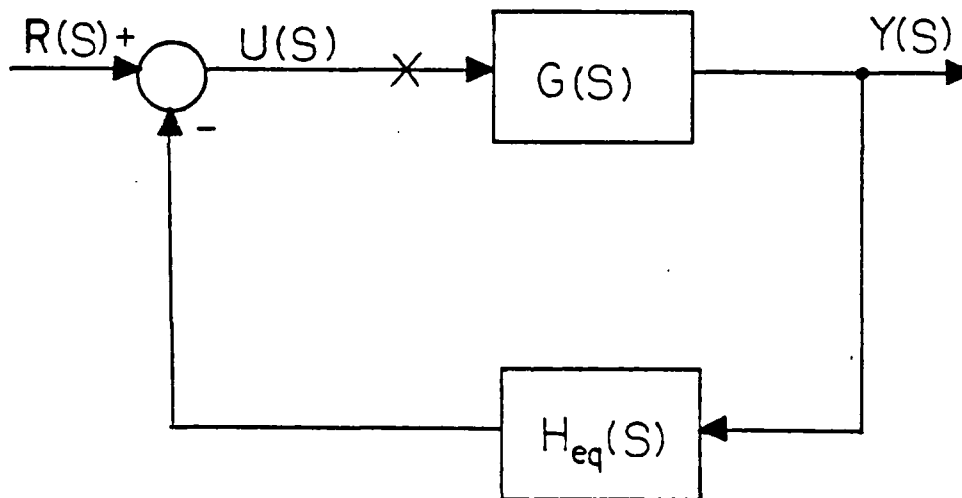
Figure 2.9   Typical Equivalent Feedback System.

23

The graphical interpretation of this result is that the polar plot of G(s)H(s) must remain outside the unit circle centered at the -1 + j0 point for all frequencies. Figure
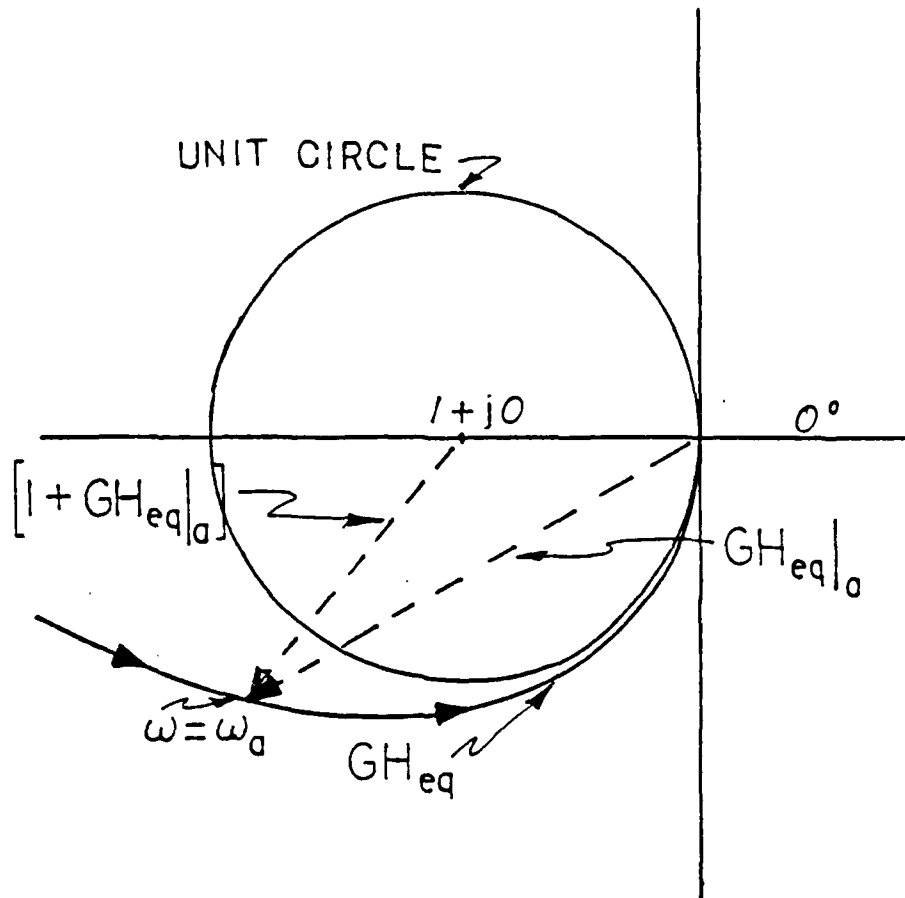
Figure 2.10   Polar Plot of an Optimal System.

2.10 shows such a polar plot. Since the optimal regulator with the loop broken at the input to the plant (denoted by the x in figure 2.9) does not penetrate the unit disk about -1 + j0 this means that the single input regulator will have

24

a phase margin of at least 60 degrees and a gain margin tolerance of fifty percent gain reduction and infinite upward margin. Further discussion of this property may be found in Anderson and Moore [Ref. 4]. With this basic review of the concepts of stability and robustness in the classical SISO system complete, the next chapter will extend some of these basic concepts to the MIMO system.

III. MULTIVARIABLE SYSTEMS

Linear quadratic design has performed relatively well in aircraft control concepts because of the ability to formulate the system state equation and to quantify acceptable performance indices for the system. Industrial applications for LQ theory have been less successful and have led British researchers to look into forms of decoupled design methods. One of the primary methods used in multivariable design is to make the system totally decoupled. This method allows each loop to be designed as a separate entity by classical means. One of the primary difficulties with this method is the problem of finding a compensator which will totally decouple the system. The method also suffers from the effects of cross-coupled perturbation terms. A method that does not totally decouple the system but makes the design problem simplier by designing a compensator that only causes the diagonal terms of the transfer matrix to be dominant over all off-diagonal terms has also been developed, [Ref. 5]. Classical frequency domain techniques are then used to design each loop of the system. The major difficulty with this and other single loop design techniques is their failure to account for cross-coupling perturbation terms that may interact between the loops. The individual loops may be highly robust in these designs but the overall system robustness may be low because of the loop interaction not accounted for in the design. This is the precise area that the singular value analysis has proven so beneficial in design. Singular value concepts may be applied to conventionally designed systems to assess their robustness. For instance, if a system is designed by LQ methods the designer may then formulate the transfer function of the system and

26

assess the singular values of the return difference matrix of the system. If the minimum singular value is found to be low at some critical frequencies the designer can then modify the $\underline{Q}$ and/or $\underline{R}$ matrices chosen in the LQ performance index and recalculate the design. In this iterative fashion a robust design would be developed.

A generalization of the SISO Nyquist theory discussed in the previous chapter has been made for the MIMO problem. This generalization leads directly to the singular value concept. The generalization is expressed in the form of the multivariable Nyquist theorem which requires that a closed loop stable system have the same number of counterclockwise encirclements of the origin by the locus of the $\det(\underline{I}+\underline{G}(j\omega))$ as the number of open loop poles that are unstable. This theorem is formally stated as;

Let $N[f(s)]$ denote the number of clockwise encirclements of $(-1,0)$ by the locus of $f(s)$ as s traverses the contour D of figure 3.1 in a clockwise sense. The closed-loop system will be stable if and only if for all R sufficiently large

$$N[f(s)] = -P$$

where

$$f(s) = -1 + \det[\underline{I}+\underline{G}(s)] = \varphi_{ol}(s)/\varphi_{cl}(s) -1 \text{ and}$$

$P = $ the number of closed right-half plane zeros of $\varphi_{ol}(s)$.

The application of the Nyquist theorem comes through the fact that a multivariable system will not be robust to modelling errors if the return difference matrix, $\underline{I} + \underline{G}$, is nearly singular for some frequency. If $\underline{I} + \underline{G}$ is nearly singular a small change in $\underline{G}$ may make $\underline{I} + \underline{G}$ exactly singular. This causes the $\det(\underline{I} + \underline{G})$ to become zero and the Nyquist encirclement count to change indicating an unstable system. It is possible for very small changes in $\underline{I}+\underline{G}$ to produce large changes in the determinant of $\underline{I}+\underline{G}$. The matrix $\underline{I}+\underline{G}$

27
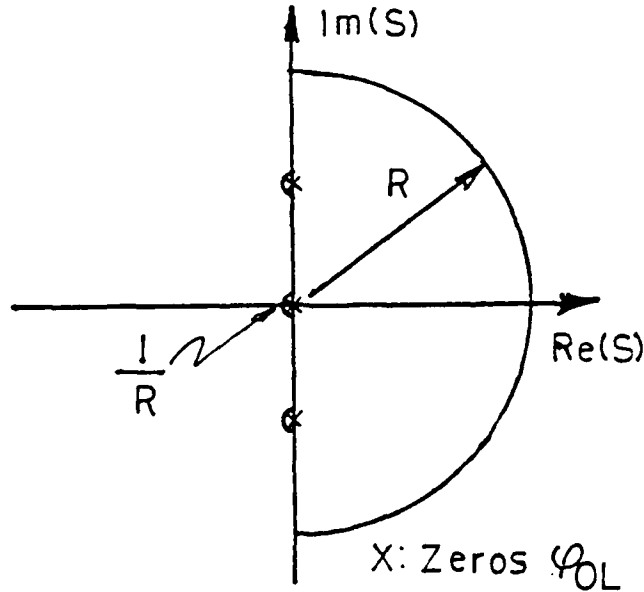
Figure 3.1   Nyquist D Contour.

$$\begin{bmatrix} 10/s+a & 9.99/s+a \\ 10/s+a & 10/s+a \end{bmatrix}$$

has determinant $0.1/(s+a)^2$. If the element p_{21} is changed by only one percent to $9.9/s+a$ the determinant becomes $1.1/(s+a)^2$ which is a significant change in the determinant value. Therefore, it is evident that $\det(\underline{I} + \underline{G})$ is not an accurate measure of how near the return difference is to singularity. Researchers, in the field of controls [Ref. 6], [Ref. 7], [Ref. 8], [Ref. 9] have used singular value analysis to determine how near the return difference matrix is to singularity.

Since the number of encirclements of the Nyquist diagram changes as f(s) passes through the -1 point or when $\det(\underline{I}+\underline{G})$ is zero it is important to find how near the return difference matrix $\underline{I}+\underline{G}$ is to being singular. This nearness to singularity can be interpreted as closeness of the matrix G

28

to the critical point, -1. A quantity which can be used to express the nearness to singularity of the matrix is the minimum matrix singular value denoted by $\underline{\sigma}$. Given a matrix A the singular value may be expressed by equation 3.1

$$\underline{\sigma}(A) = \min_{\lambda} (\ \lambda_i\ (\underline{A}^H\underline{A}))^{1/2} \tag{3.1}$$

where $\lambda_i\ (\underline{A}^H\underline{A})$ is the eigenvalue of the complex conjugate transpose of $\underline{A}$ times $\underline{A}$. A basic MIMO linear system is
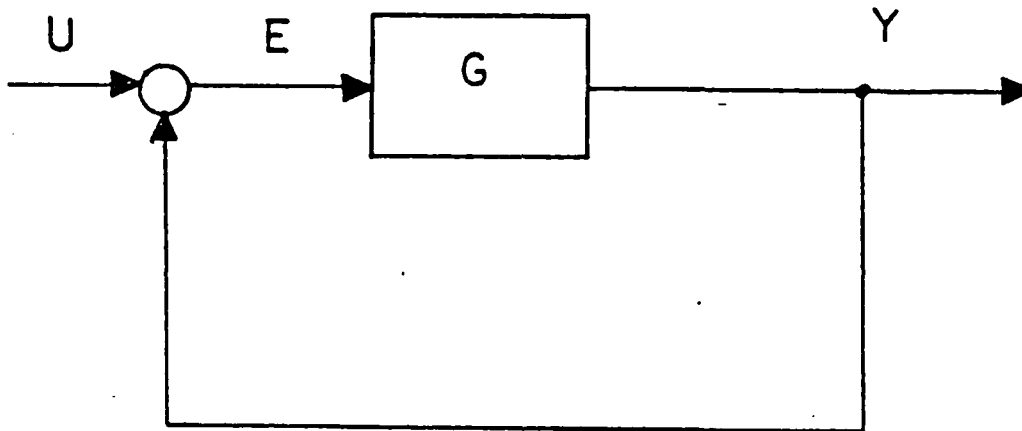
Figure 3.2 Basic Multi-input, Multi-output System.

depicted in figure 3.2. An additive perturbation to the plant is shown in figure 3.3. If the plant is stable before the perturbation is added to the system the Nyquist theorem will be satisfied and the locus of GH will not encircle the -1,0 critical point. When the perturbation is added to the system as long as the Nyquist locus is not forced to

29
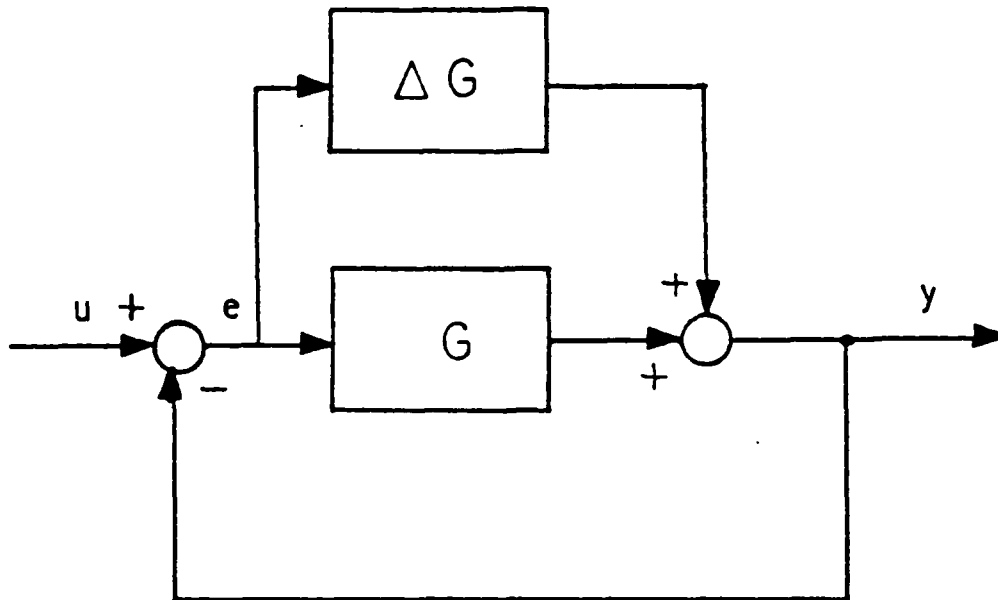
Figure 3.3   Additive Perturbation.

encircle the -1 point the system will remain stable.   A
sufficient condition, recalling the SISO discussion in
chapter 2,   for the perturbed Nyquist plot not to change
encirclements is that the norm of the perturbation $\Delta \underline{G}$
remain less than the norm of the return difference matrix as
expressed in equation 3.2.

$$|| \Delta \underline{G}(j\omega)|| < 1/||(\underline{I}+\underline{G})^{-1})|| \qquad (3.2)$$

$\omega \geq 0$ This condition will guarantee that the locus of the
$\det(\underline{I}+\underline{G})$ does not pass through the -1 point. If the l_2 or
Euclidean norm is assumed for this condition the equation

30

3.2 may be expressed in terms of singular values as equation
3.3.

$$\bar{\sigma}(\Delta \underset{\sim}{G}) \leq \underline{\sigma}(\underset{\sim}{I} + \underset{\sim}{G}) \qquad (3.3)$$

This result states that as long as the maximum singular
value of the perturbation matrix $\Delta \underset{\sim}{G}$ is below the minimum
norm value of the return difference matrix the system will
remain stable. The problem of guaranteeing robustness
becomes that of finding the largest norm of the perturbation
quantity, the largest singular value, for which the smallest
norm or singular value of the return difference matrix will
remain non singular.

The multiplicative form for a system such as figure 3.4
gives the similar norm equation in 3.4

$$||\Delta \underset{\sim}{G}(j\omega)|| < 1/||(\underset{\sim}{I} + (\underset{\sim}{G})^{-1})^{-1}|| \qquad (3.4)$$
$\omega \geq 0.$ which may be expressed as;

$$\bar{\sigma}(\Delta \underset{\sim}{G}) \leq \underline{\sigma}(\underset{\sim}{I} + \underset{\sim}{G}^{-1}) \qquad (3.5)$$

Singular value decomposition software is readily available
to determine how near the matrix $\underset{\sim}{I} + \underset{\sim}{G}$ or $\underset{\sim}{I} + (\underset{\sim}{G})^{-1}$ is to
singularity.
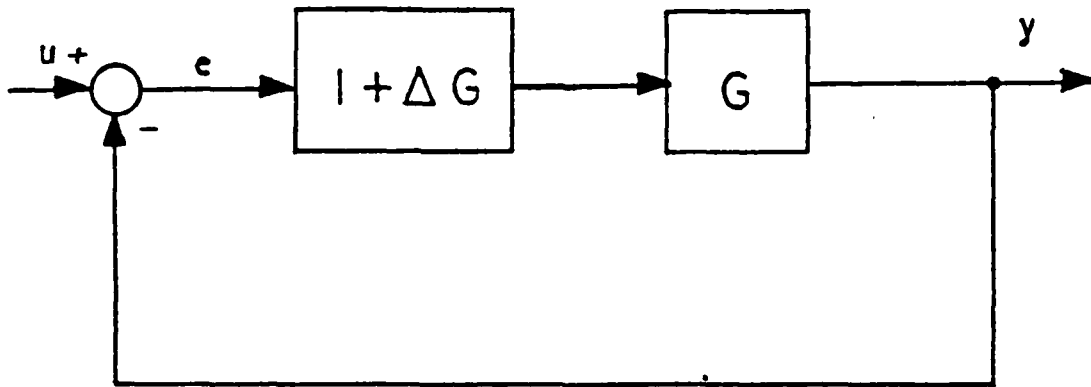
Figure 3.4   Multiplicative Perturbation.

IV. OPTIMIZATION

The purpose of this chapter will be to briefly describe several of the currently employed optimization techniques and the Automated Design Synthesis (ADS) program, [Ref. 10], which employs these techniques. In general, optimization implies finding the "best" possible solution to a problem. In actuality the best solution found by an optimization technique might really only be a "better" solution to the problem. The purpose of ADS and other optimization routines is to allow a rational search to be conducted to find the best possible design. The techniques of numerical optimization are used to logically vary the various parameters that affect the design until a good solution is found.

As an example of an unconstrained optimization problem consider the following problem developed in [Ref. 11]. The problem is to minimize the function

$$F(\underline{x}) = 10x_1{}^4 - 20x_1{}^2x_2 + 10x_2{}^2 + x_1{}^2 - 2x_1 + 5 \qquad (4.1)$$

$F(x)$ is often called the objective function, the cost function or the penalty function. Since there are no conditions imposed on the design variables, x_1 and x_2, and no additional limits imposed on the overall design, the problem is considered to be one of unconstrained minimization. Figure 4.1 represents this problem in the design space. From the figure it appears the optimum is near the point 1,1. Calculus may be applied to determine the optimum exactly. Taking the derivatives
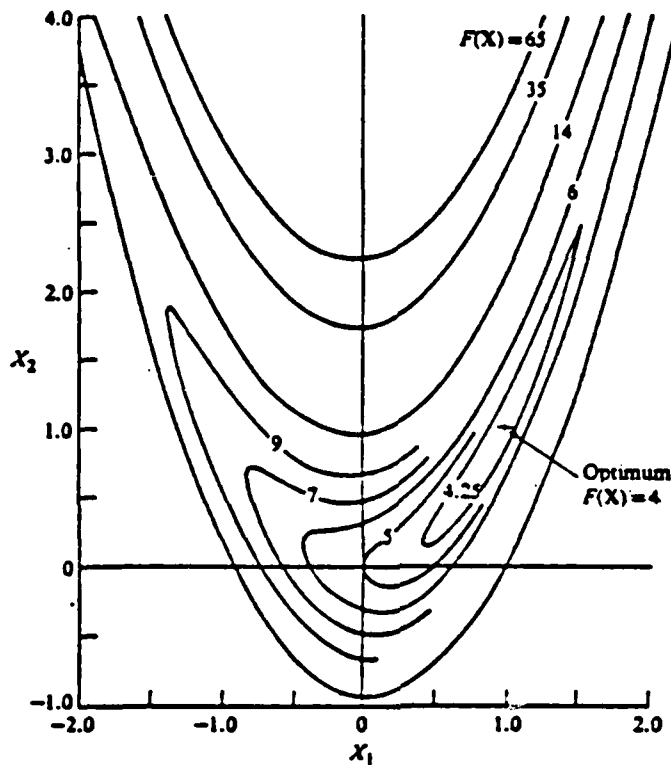
Figure 4.1   Design Space for Example Problem.

$$\partial F (\underline{x}) / \partial x_1 = 40 x_1{}^3 - 40 x_1 x_2 + 2x_1 - 2 = 0 \qquad (4.2)$$

$$\partial F (\underline{x}) / \partial x_2 = -20 x_1{}^2 + 20 x_2 = 0 \qquad (4.3)$$

and then solving the set of equations it is found that x_1 =1.0 and x_2 =1.0.

If design conditions are imposed on the problem then the optimization becomes one of constrained function minimization. In other words, while the minimum of a function is still sought, this minimum must exist within the limits

34

imposed by the design conditions or constraints. The minimum of an unconstrained function will not necessarily be the same minimum for a constrained function.

As another example , also from [Ref. 11], a design is sought which gives the minimum weight of a particular column. This weight is expressed as:

$$W = \rho Ah = \rho \pi Dh \qquad (4.4)$$

where ρ is the unit weight of the material and A the cross sectional area. The stress in the column is given by:

$$\sigma = P/A = P/\pi Dt \qquad (4.5)$$

without going into detail the design will be constrained by the allowable stress on the structure. Other constraints , Euler buckling and shell buckling, are also of interest to the designer of this column. The design problem is then stated as:

$$\text{minimize} \quad W = \rho \pi Dth \qquad (4.6)$$

for the constraints

$$g(1) = \sigma / \bar{\sigma} - 1 \leq 0 \qquad (4.7)$$

$$g(2) = \sigma / \sigma_b - 1 \leq 0 \qquad (4.8)$$

$$g(3) = \sigma / \sigma_s - 1 \leq 0 \qquad (4.9)$$

with $D \geq 10e-06$ and $t \geq 10e-06$.

$$g(4) = t - D \leq 0 \qquad\qquad (4.10)$$

$\bar{\sigma}$, σ_6 , σ_s refer to the allowable design stress, Euler buckling stress and the shell buckling stress respectively.
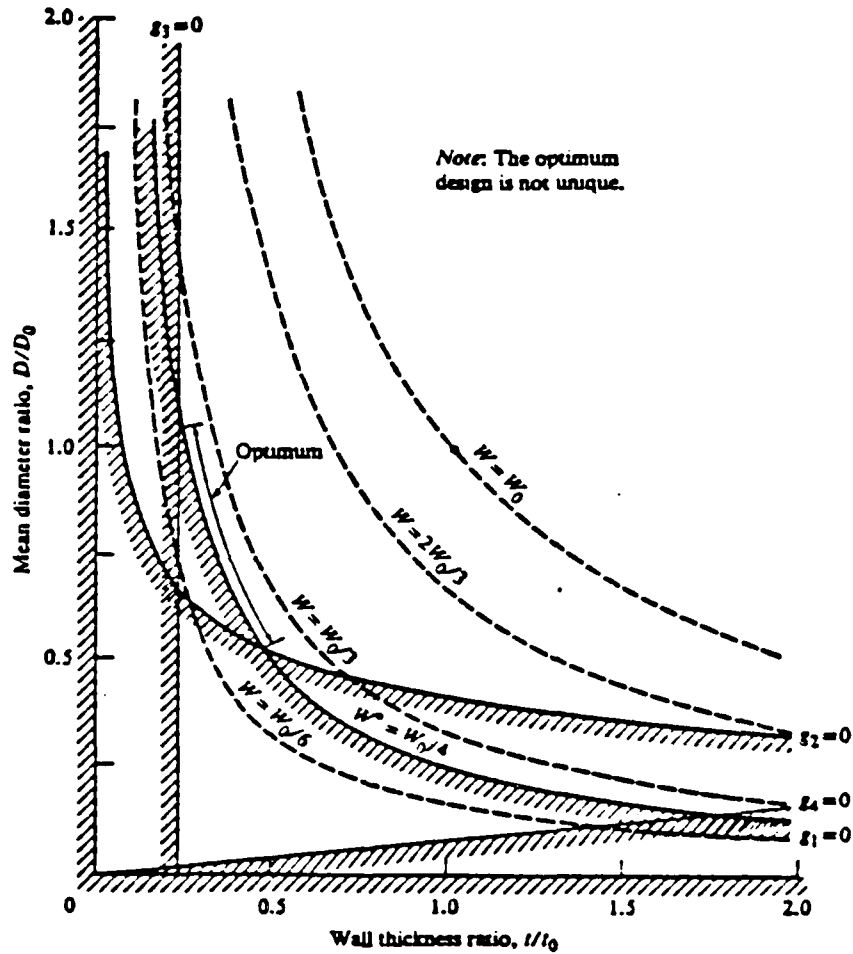
Figure 4.2   Design Space for Column Problem.

Figure 4.2 is a design space diagram for this problem.   It is noted from the figure 4.2 that the optimum is not unique and can be any value along the arc noted as optimum in the figure.

36

The general optimization problem is then written as:

minimize $F(\underline{x})$ $\qquad$ objective $\qquad$ (4.11)

subject to $g_j(x) \leq 0$ $j=1,m$ inequality constraint $\quad$ (4.12)

$h_K(\underline{x}) = 0$ $k=1,1$ equality constraint $\quad$ (4.13)

$x_i^l \leq x_i \leq x_i^u$ $i=1,n$ side constraint $\qquad$ (4.14)

where $\underline{X} = \text{col}(x1,x2,\ldots,xn)$ is the design variable.

The methods used to solve this problem are usually iterative. After the establishment of an initial set of variables the optimizer will update this initial value until the optimum values are found. Again, borrowing from [Ref. 11], the iterative technique may be demonstrated by a simple example. Figure 4.3 is used to illustrate this problem. Given the initial data set $\underline{X}^0$ the formula

$$\underline{X}^1 = \underline{X}^0 + \alpha^* \underline{S}^1 \qquad (4.15)$$

can be used to upgrade this estimate of $\underline{X}$. The vector S is the search direction for the iteration and the scalar quantity α is the distance of the move in the $\underline{S}$ direction. Beginning at $\underline{X}^0$ it is desired to reduce the objective function. The search for values of $\underline{X}$ that reduce the objective function is made in the $\underline{S}$ direction which in this example is the opposite of the gradient of the function at point $\underline{X}^0$. The choice of $\underline{S}$ could be arbitrary as long as it reduces the

Figure 4.3   Illustrative Example for Iteration.

value of the objective. If $\underline{S}$ is chosen to be opposite the gradient of the objective then the search would be a steepest descent search.   Once the direction is chosen then the scalar   must be found to minimize the objective along this direction   vector without   crossing a   constraint boundary. The technique for finding α is to evaluate $\underline{X}$, the objective and the constraint functions for values of α using computer code and numerical interpolation to estimate α. This α value then gives the minimum value for $F(\underline{X})$ along this search vector. Now that α for this direction is known the new equation becomes:

$$\underline{X}^2 = \underline{X}^1 + \alpha^* \underline{S}^2 \tag{4.16}$$

and now $\underline{X}^1$ is used to start a new search in direction $\underline{S}^2$ and compute a new α^* that continues to reduce the objective

38

function. These one dimensional searches are continued until
no more reduction in the objective can be found. At this
point no further design improvement is possible.

Optimization techniques do not always lead to the abso-
lute optimum when applied to problems of practical interest.
The reasons for this could be numerical ill-conditioning of
the problem formulation or simply that there are multiple
solutions to the problem. Because of these difficulties it
may be adviseable to choose several different starting
points for the optimization process and use engineering
judgement as to the design most applicable to the problem
under analysis.

Considering the unconstrained case first where the
desire is to minimize the function $F(\underline{x})$, it is well known
that $F(\underline{x})$ will have a minimum where the gradient of $F(\underline{x})$ is
zero. That is:

$$grad(F(\underline{x})) = 0 \qquad\qquad (4.17)$$

with the $\nabla F(\underline{x})$ defined as:

$$grad(F(\underline{x})) = (\partial F(\underline{x})/\partial x_1, \ldots, \partial F(x)/\partial x_n)' \qquad (4.18)$$

Figure 4.4 shows why this is a necessary condition but does
not guarantee a global minimum. The gradient of $F(\underline{x})$ is
zero at all four points A,B,C,and D. However, only A and D
are minima. A would be the global minimum for the function
as defined here. D would be only a relative minimum. To
check that the zero gradient corresponds to a minimum the
Hessian matrix,i.e the matrix of second partial derivatives,
can be examined for positive definiteness. A positive defi-
nite Hessian ensures a relative minimum. The only way to
prove a global minimum for the function is to show that the
Hessian matrix is positive definite for all design variables
X. A test that is seldom possible to perform.

39

F(X)
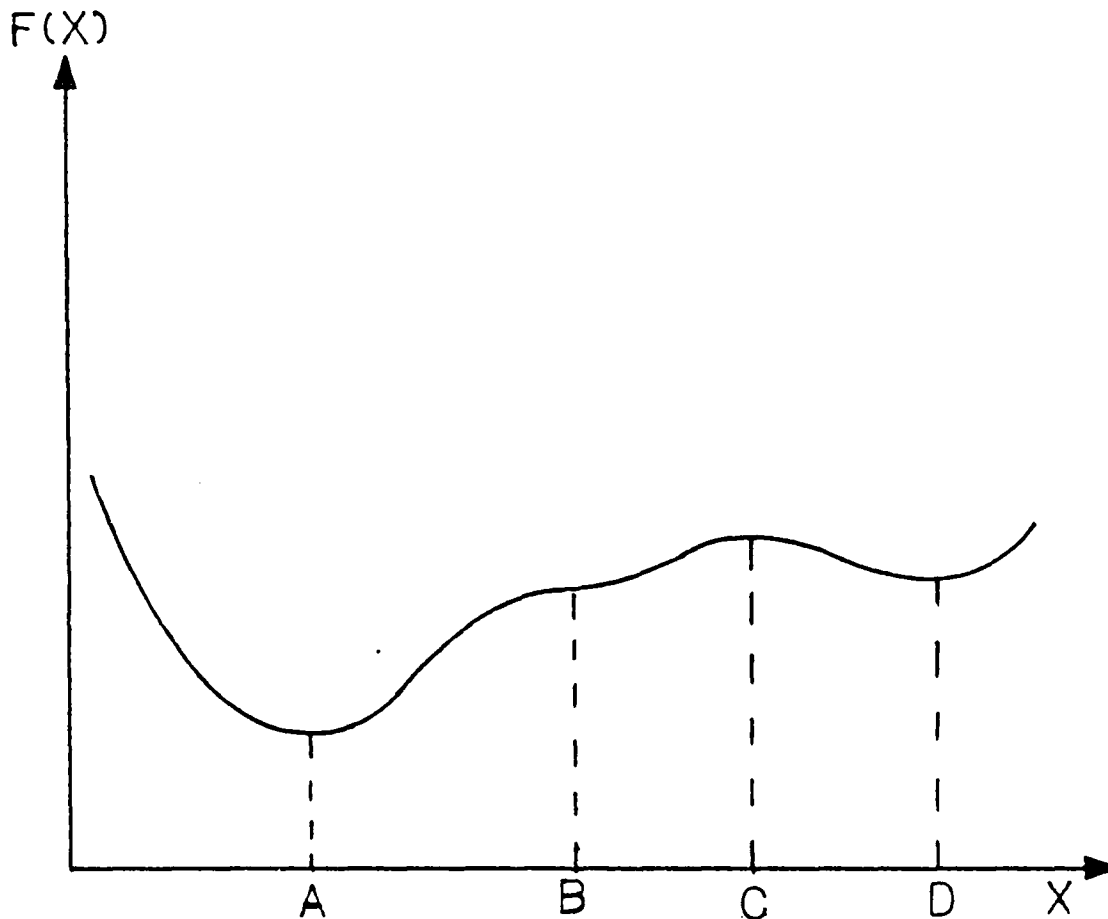
Figure 4.4   Plot of Various Points With Zero Gradient.

If the problem of minimization is a constrained problem
the situation is different from that discussed above. The
objective function gradient does not have to be zero at the
optimum. Figure 4.5 illustrates this case. Using figure 4.5
and assuming a start at point A it is necessary to choose a
search direction S which will reduce the objective function
while not violating the constraint functions. Any direction
that will reduce the objective is said to be useable. This
is seen to be the half plane sector below the tangent to
F(x) at point A.   If a half plane to the right of the
tangent to the active constraint at point A is considered
then the feasible sector is formed. A combination of the two

40

conditions gives the useable, feasible sector where the
search direction S must be chosen. In mathematical nomencla-
ture the above argument may be stated as:

useable direction $\text{grad}(F(\underline{X})) \cdot \underline{S} \leq 0$ (4.19)

feasible direction $\text{grad}(g_j(\underline{x})) \cdot \underline{S} \leq 0$ for all j   (4.20)

for which $g_j(x) = 0$.

Point B in figure 4.5 shows a point where the gradient
of the objective and constraint point in exactly the oppo-
site direction. At a point such as B the only search vector
S that meets requirements for useability and feasibility is
tangent to the constraint boundary and to a line of constant
objective function. This condition is stated as:

$$\text{grad}(F(\underline{x})) + \mathcal{Z}\lambda_j \text{grad}(g_j(\underline{x})) + \mathcal{Z}\lambda_{m+k} \text{grad}(h_k(\underline{x})) = 0 \qquad (4.21)$$

where $\lambda_j \geq 0$ and $\lambda_{\ell+m}$ -unrestricted.

With this brief background in optimization complete the
Automated Design Synthesis (ADS) program will be briefly
discussed. More detail on the ADS routine can be found in
[Ref. 10]. This code was developed as a follow-on to the
successful CONMIN code [Ref. 12] developed by Vanderplaats.
It is designed as a black box optimizer which allows the
user to choose combinations of one dimensional search, opti-
mization algorithm and optimization strategy. These will be
discussed later. For the user with specific requirements the
code may be tailored by parameter modification to meet
specific requirements. For the work done in this thesis it
was assumed the user of the code has no detailed knowledge
of optimization and will want to use the code in the

41
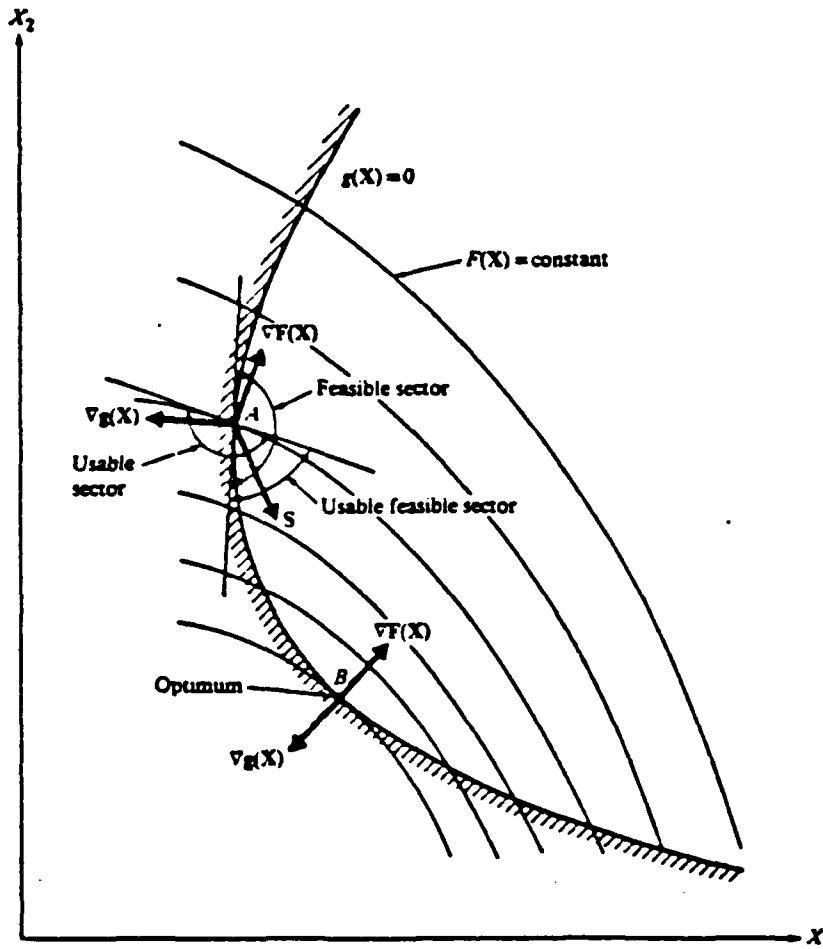
Figure 4.5 Constrained Optimization Example.

simplest form. As such, the ADS calls made from the main
program use only default parameters and first forward finite
difference gradients. Should analytical gradients be avail-
able the user could use them within the code if desired with
no difficulty. The calls made by the user to the ADS
routines specify several important aspects of the problem
solution. The user may call for an optimization strategy to
be used in the routine. This is not required and its use

42

depends on the problem. These strategies are discussed in [Ref. 10]. Two strategies used most often in the analysis done for this thesis have been sequential unconstrained minimization using quadratic exterior penalty function and the augmented Lagrange multiplier (ALM) method. Others available include sequential linear programming and sequential quadratic programming.

The basic optimizer is also chosen by the user from two unconstrained and three constrained optimization algorithms. The unconstrained algorithms are Fletcher-Reeves conjugate directions, Davidon-Fletcher-Powell (DFP) variable metric method and the Broyden-Fletcher-Goldfarb-Shanno (BFGS) variable metric method. The method of feasible directions and robust feasible directions are available for constrained minimization.

The user has available several types of one dimensional searches using Golden Section or polynomial approximation techniques. The ADS code has tailored these one dimensional search algorithms for the unconstrained and constrained cases, allowing the user to make appropriate choices for the type of problem to be solved. Figure 4.6 shows the basic organization of the ADS program.

Since it is not the purpose of this chapter to cover the subject of optimization in detail all the possible routines in ADS will not be discussed. A few of the routines found to work well for the work completed in this thesis will be briefly described, however. There are several methods of optimizing functions of one variable or one dimensional searches. For instance a large number of points, n, could be chosen and the function $F(x)$ evaluated at each point. The point corresponding to the smallest value of $F(x)$ could then be considered the optimum value of the one dimensional search. This method is hit and miss and better methods of

43

USER-SUPPLIED CALLING PROGRAM

DIMENSION ARRAYS

INITIALIZE PARAMETERS

INFO=0

CALL ADS(INFO,ISTRAT,...) ──────────> ADS CONTROL

NO / INFO=0 \ YES

YES / INFO=1 \ NO   EXIT SOLUTION COMPLETE

STRATEGY

OPTIMIZER

EVALUATE OBJECTIVE AND CONSTRAINTS

EVALUATE REQUIRED GRADIENTS

ONE-DIMENSIONAL SEARCH
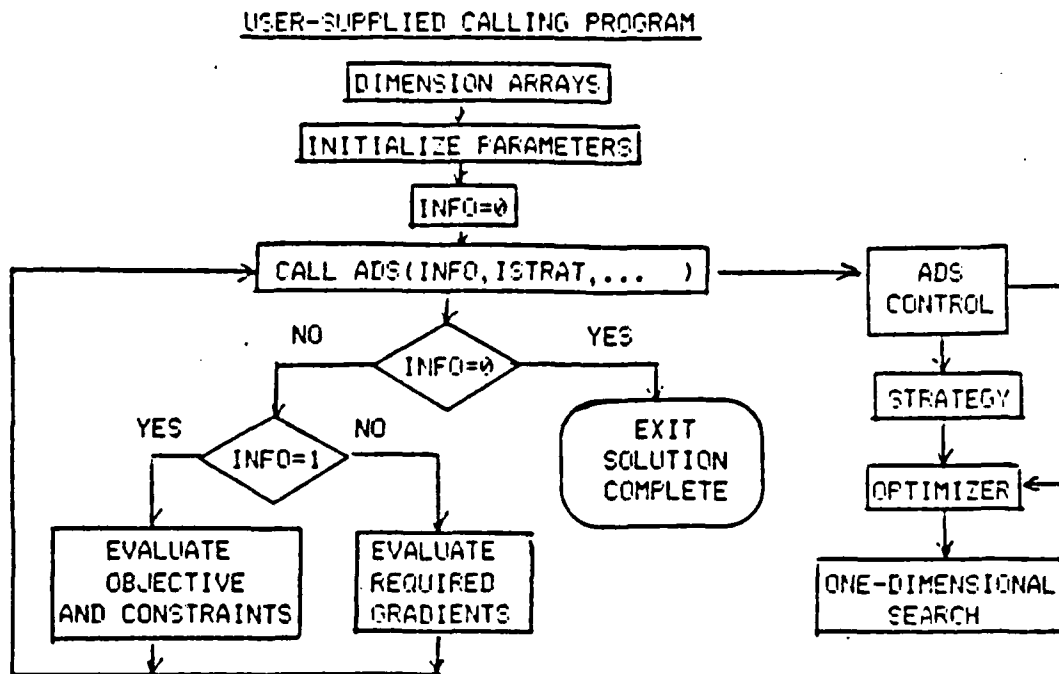
Figure 4.6   Organization of ADS Program.

locating optimum points can be used. The methods used in ADS are Golden Section and polynomial interpolation.

Golden Section search methods are easy to program on the digital computer and do not require continuous derivatives. They have a known convergence rate and are reliable for poorly conditioned problems. The major drawback of the Golden Section routines is the large number of function evaluations require. The Golden Section algorithm is simply illustrated through the use of figure 4.7 Assume that X^0 and X^1 are known to be bounds on the curve's minimum value. Also, the function values of $F(x)$, F^0 and F^1 , are evaluated and known at these points. By picking two intermediate points X^2 and X^3 where $X^2 < X^3$ and evaluating the function
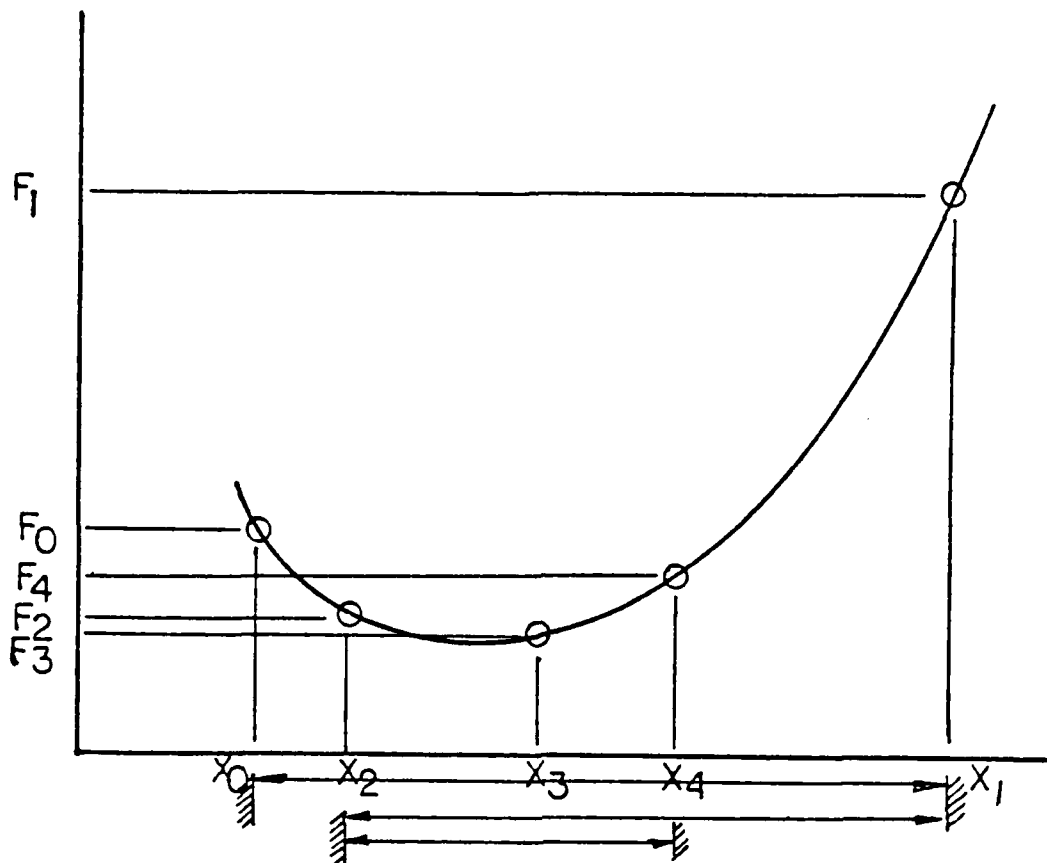
Figure 4.7   Golden Section Diagram.

at these points a bound on the minimum may be modified. Since F^2 for this figure is larger than F^3 the point X^2 forms the new lower bound. The minimum is now between X^2 and X^1 . If the function F^4 at X^4 is determined and shown to be larger than F^3 then X^4 becomes the new upper bound on the minimum. By repeating this procedure the bounds may be narrowed to any desired tolerance. The Golden Section rule is applied to this problem to reduce the bounds in the quickest possible time. By appropriately picking the values of X's at which each function evaluation is made an efficient algorithm that uses the ratio proportion of the Golden

45

Section, i.e. $X^2/X^1 = 1.62803$ is developed. By choosing a value τ based on this Golden Section rule where $\tau = 0.38197$ estimates for interior points X^2 and X^3 can be made as:

$$X^2 = (1 - \tau) X^0 + \tau X^1 \qquad (4.22)$$

$$X^3 = \tau X^0 + (1 - \tau) X^1 \qquad (4.23)$$

As each new bound is found the process repeats until the accepted level of convergence is reached.

The polynomial interpolation method is accomplished by first fitting a polynomial curve to the points about where the minimum is desired and then finding the minimum of the polynomial function. For example if the function $F(X)$ is approximated by a quadratic as:

$$F = a_0 + a_1 X + a_2 X^2 \qquad (4.24)$$

Then the value of X, X^*, where F' is zero can be shown to be :

$$X^* = -a_1 / 2a_2 \qquad (4.25)$$

If a_2 is positive then F will be minimum. Other degrees of polynomials may be used in similar fashion.

Now that the basic one dimensional search methods have been reviewed the next step is to examine the basic optimization routines. First, the unconstrained case will be reviewed. The optimum X^* is at the point where

$$\mathrm{grad}(\ F(\underline{X}^*))= 0 \qquad\qquad\qquad (4.26)$$

Several zero order methods exist for the purpose of finding the minimum value. These include random search, Powell's method and Box's method. Since ADS does not use these methods they will not be discussed.

Automated Design Synthesis makes use of first order methods which will now be discussed. The steepest descent method is best known, but poor in performance. Just as the name implies a search direction is chosen opposite the gradient of the objective function. At iteration 0

$$\underline{S}^0 = - \mathrm{grad}(\ F(\underline{X}^0)) \qquad\qquad\qquad (4.27)$$

Figure 4.8 shows this algorithm geometrically. Note that the method simply stair steps its way down the "hill" to the valley or minimum. ADS uses the Fletcher-Reeves modification to steepest descent. In this routine a conjugate direction is chosen to improve the speed of the search. Figure 4.9 shows how this method tracks to the solution.

The variable metric methods listed earlier are usually more powerful than Fletcher-Reeves because they store information that allows the algorithm to approximate the inverse of the Hessian matrix or second derivative. For further discussion of these methods see [Ref. 11] or other similar optimization texts.

ADS employs two direct methods for constrained minimization. One method is that of feasible direction and the other is the method of robust feasible direction. Since these two methods were seldom employed within the work presented in this paper they will not be discussed.

The methods chosen to handle the constrained minimization problems formulated in this thesis are referred to as Sequential Unconstrained Minimization Techniques (SUMT).

47
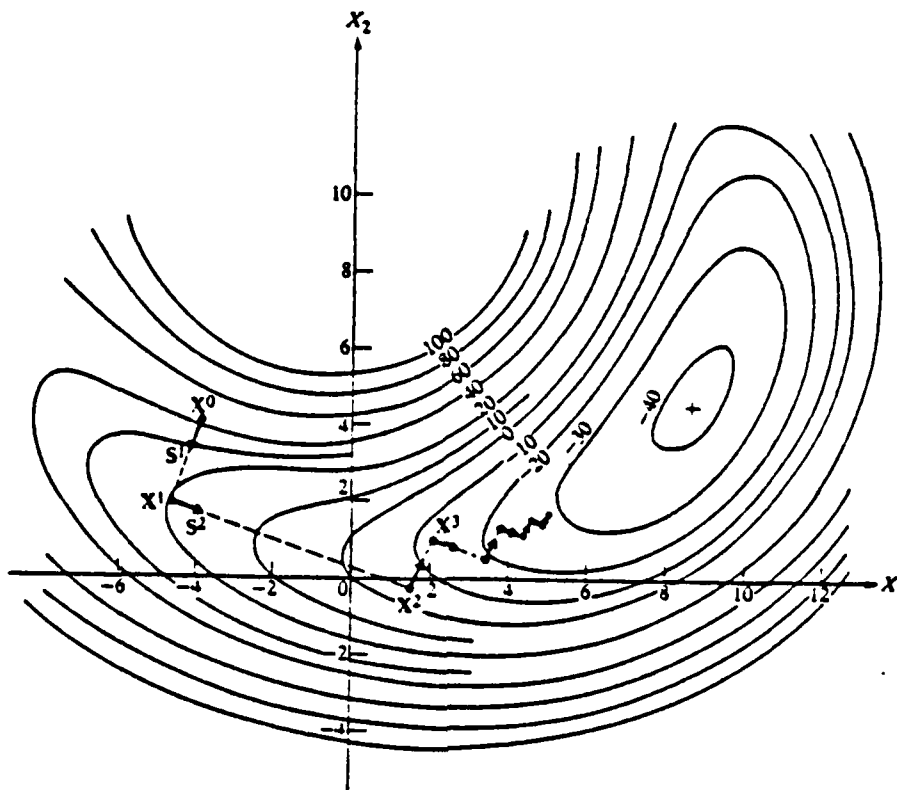
Figure 4.8   Steepest Descent Algorithm.

SUMT methods are methods which formulate the objective
function and the constraint functions into an augmented
objective function and then solving the problem as if it
were an unconstrained optimization task. ADS employs
interior and exterior penalty function techniques as well as
an Augmented Lagrange Multiplier (ALM) technique.

The exterior penalty function method is incorporated by
forming a penalty from the constraint equations. This
penalty is of the form :

$$P(\underline{x}) = \sum_{j=1}^{m} \max(0, g_j(\underline{x}))^2 + \sum_{k=1}^{l} (h_k(\underline{x}))^2 \qquad (4.28)$$

48
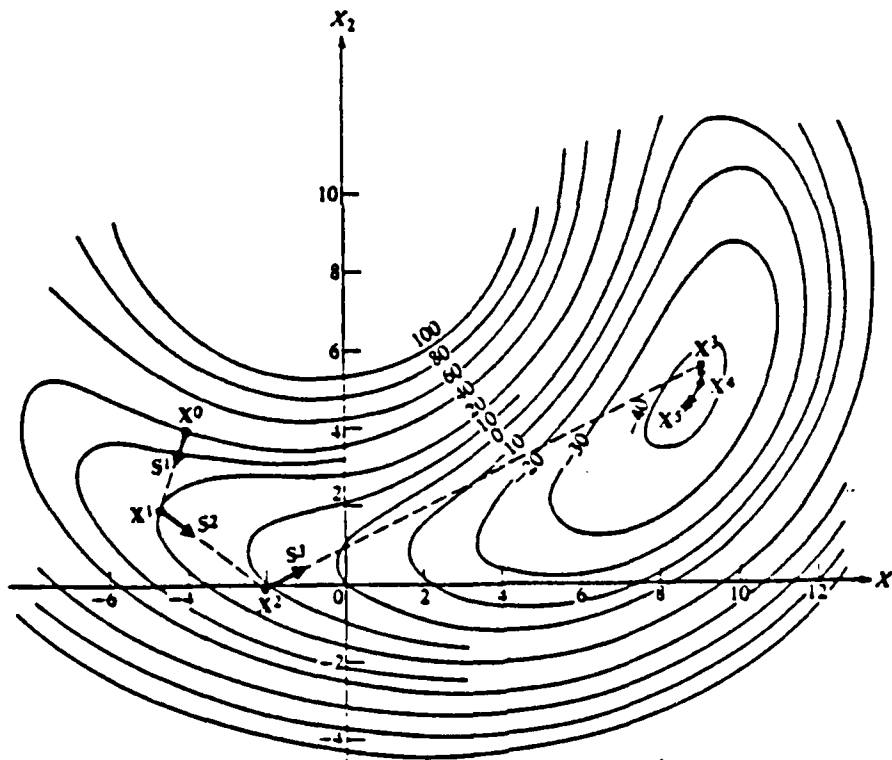
Figure 4.9   Fletcher-Reeves Conjugate Direction Algorithm.

From this equation it is obvious that $P(\underline{x})$ is zero if all
the constraints are satisfied. That is, if $g_j(\underline{x}) \leq 0$ and h_k
$(\underline{x}) = 0,$ then all conditions of the penalty function are
satisfied. If an element of the penalty function is violated
then the penalty increases as the square of the violated
constraint.   A pseudo or augmented objective function is
formulated where :

$$\phi(\underline{x}, r_\rho) = P(\underline{x}) + r_\rho P(\underline{x}) \tag{4.29}$$

The constant r_ρ is a weighting factor for the penalty. It is
adjusted with ADS as the optimization proceeds to allow the
program to systematically converge to an optimum solution.

49

One disadvantage of this method is that if the optimizer is stopped short of the optimum the design will be in the infeasible region and probably not useful.

Originally developed as a method to solve equality constrained problems the version of ALM in ADS has been modified to work with both equality and inequality constraints. The statement is as follows:

minimize $F(\underline{x})$ (4.30)

subject to $h_k(\underline{x}) = 0$ k=1,l   (4.31)

Next a Lagrangian is created such that

$$L(\underline{x},\lambda) = F(\underline{x}) + \sum_{k=1}^{l} \lambda_k\, h_k(x) \qquad (4.32)$$

Then the equation is augmented with an exterior penalty function such that

$$A(\underline{x},\lambda,r_p) = F(\underline{x}) + \sum_{k=1}^{l}(\lambda_k h_k(\underline{x}) + r_p(h_k(\underline{x}))^2) \qquad (4.33)$$

The power of this method is that in theory precise constraint matching is possible whereas in the exterior penalty function method it is not. The full details of the method will not be covered here, however, the final form of the objective function will be included:

$$A(\underline{x},\lambda,r_p) = F(\underline{x}) + \sum_{j=1}^{m}(\lambda_j\psi_j + r_p\psi_j^2) + \sum_{k=1}^{l}(\lambda h_k(\underline{x}) + r_p(h_k(\underline{x}))^2) \qquad (4.34)$$

where $\psi_j = \max(g_j(x), -\lambda_j/2r_p)$, and the update formulas for the Lagrange multipliers are:

$$\lambda_j^{p+1} = \lambda_j^{P} + 2r_p(\max(g_j(x), -\lambda_j/2r_p)) \quad j=1, m \tag{4.35}$$

$$\lambda_{k+m}^{p+1} = \lambda_{k+m}^{P} + 2r_p h_k(x) \quad k=1, l \tag{4.36}$$

With this brief review of optimization and the ADS program completed the next chapter will discuss the program development for this thesis research.

V. OPTIMIZATION DESIGN PROCEDURE

To accomplish the task of designing a control system with acceptable time domain performance and robustness characteristics in a straight forward manner a numerical optimization technique was chosen as the method of implementation of the design algorithm. Using numerical procedures to adjust selected design variables, in this case the feedback and/or filter gains, a desired level of performance can be achieved. This level of performance is actually a combination of time domain performance and robustness or frequency domain performance. By establishing the criteria for the system performance in terms of an optimization objective and constraint functions a versatile procedure can be developed to set the system feedback gains and affect an acceptable design in terms of performance and robustness. The pole placement and robustness (POPLAR) design procedure uses pole placement to establish a designer selected performance level and then a minimum singular value level to establish the robustness.

The pole placement portion of the procedure will be discussed first. The pole placement technique was chosen because it was relatively easy to implement through a numerical optimization routine. By using this numerical procedure it is also simple to incorporate robustness into the procedure along with the performance requirements. A numerical technique similar to one posed in [Ref. 13] was chosen for the pole placement algorithm. An unconstrained optimization routine from the IBM IMSL library was used for this program. The routine, a Newton method, uses adjustments to the output feedback gains to reduce the size of an objective function. This objective function was expressed as a function of the

52

pole location of the system and as the objective was reduced the poles were moved toward the desired locations. To provide more versatility in the pole placement algorithm a method that can use constraint functions as well as unconstrained optimization was chosen for this program. The designer may use either objective, constraint or a combination of functions to secure the desired pole locations. As currently implemented in the program the cost or objective portion of the pole placement procedure is constructed as equation 5.1

$$OBJ = \sum_{i=1}^{j} (\lambda_{R_{D_i}} - \lambda_{R_i})^2 + (\lambda_{I_{D_i}} - \lambda_{I_i})^2 \qquad (5.1)$$

where λ_R = real eigenvalue

λ_I = imaginary eigenvalue

λ_{R_D} = desired eigenvalue location

λ_{I_D} = desired eigenvalue location

The constraint formulation is a function that must be kept negative or the constraint is violated. It is written as equation 5.2

$$g(j) = \sqrt{(\lambda_{R_{D_j}} - \lambda_{R_j})^2 + (\lambda_{I_{D_j}} - \lambda_{I_j})^2} - r \qquad (5.2)$$

where r is a tolerance circle established as a function of pole placement position. Since the aim of the optimizer is to keep g negative any time the λ function of the constraint is greater than r the constraint will become active, i.e. violated. The optimizer will then attempt to move the constraint to the inactive status by adjusting the design parameters of the system.

Consideration of implementation of the frequency domain or robustness portion of the design procedure begins with the concept of MIMO phase and gain margins. Several useful theorems on singular value analysis of multiloop systems are

53

presented in [Ref. 8]. One of these theorems relates the matrix singular value of the return difference function to a parameter, α, and further shows that as long as the maximum singular value of the perturbation function $(\underset{\sim}{L}^{-1} - \underset{\sim}{I})$ remains less than this α, the system remains stable. The value of α is then related to gain and phase margins of the MIMO system. The relationship developed is given in equations 5.3 and 5.4:

$$\text{gain margin} = GM = 1/(1+\alpha_o) \qquad\qquad (5.3)$$

$$\text{phase margin} = PM = \pm\cos^{-1}(1-\alpha_o^2/2) \qquad\qquad (5.4)$$

provided that equation 5.5 holds.

$$\underline{\sigma}(\underset{\sim}{I}+\underset{\sim}{G}) \geq \alpha_o \qquad\qquad (5.5)$$

for some $\alpha_o \leq 1$

These phase and gain margins are guaranteed in every loop simultaneously.

Universal gain and phase margin curves, [Ref. 14], based on the minimum singular values of the return difference matrix are developed from equation 5.6.

$$(\underset{\sim}{L}^{-1}-\underset{\sim}{I}) = \max\sqrt{(1-1/k_n)^2 + 2/k_n(1-\cos\varphi_n)} \qquad\qquad (5.6)$$

for all n with $k_n > 0$. These curves shown in figure 5.1 allow the designer to pick a singular value that corresponds to a specific gain and phase margin for a given system. In addition to the universal gain and phase plot [Ref. 15] developes an optimizer solution for formulating a robust controller using the CONMIN algorithm [Ref. 12].
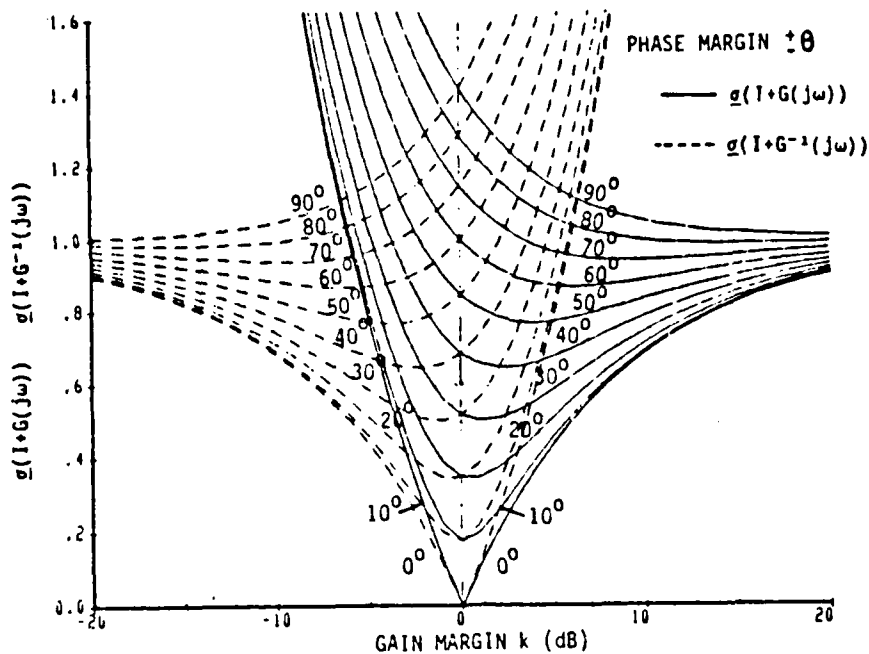
Figure 5.1   Universal Gain and Phase Singular Value Plot.

Since the universal curve in figure 5.1 provides a convenient method of specifying gain and phase margins in terms of singular values the robustness portion of the pole placement and robustness design procedure uses the minimum singular value level of the return difference matrix to determine the robustness. The minimum singular value level is formulated as an objective or constraint function in equation 5.7

$$J = \sum_{J} (\max (0, (\sigma_0 - \underline{\sigma} (j\omega, p))))^2 \qquad (5.7)$$

The optimization procedure may be used to change feedback gains until the minimum singular value is raised above this desired design level. Although the same formu__tion can be used as a negative constraint function it has not been implemented as such within this program. There are numerous

55

ways the singular value formulation could be implemented within the program by changes of the code if design requirements forced such changes.

The pole placement and robustness design program is based on the ADS code discussed in Chapter 4 to implement the design variable selection procedures. The pole placement and robustness program consists of two separate programs. The first program is used to provide designs for state or output feedback problems while the second program is used for observer or filter designs.

For the state or output feedback design program the user must input the plant matrices $\underset{\sim}{A}, \underset{\sim}{B}, \underset{\sim}{C}$ and initial starting values for the feedback matrix F. The matrices correspond to the following linear differential system:

$$\underset{\sim}{\dot{x}} = \underset{\sim}{A}\underset{\sim}{x} + \underset{\sim}{B}\underset{\sim}{u} \qquad (5.8)$$

$$\underset{\sim}{y} = \underset{\sim}{C}\underset{\sim}{x} \qquad (5.9)$$

$$\underset{\sim}{u} = -\underset{\sim}{F}\underset{\sim}{x} \qquad (5.10)$$

A feed-forward matrix has not been considered in the current program. A feed forward matrix could be added to the procedure if required for specific design cases.

As the design program is currently coded the user may run output feedback or state feedback design by specifying the $\underset{\sim}{C}$ matrix as the diagonal($\underset{\sim}{I}$) matrix for state feedback. The program relies on initial starting values of the feedback gains, $\underset{\sim}{F}$. As discussed in Chapter 4 there is no guarantee that the optimum found by the procedure each time is the global optimum or that the procedure will always

56

converge to an acceptable solution. The ability to select acceptable starting values for the feedback gains will make the procedure more efficient in operation. As currently employed, the program is used to obtain pole placement and robustness for a given set of starting gains and a selected optimization routine from the ADS program. If the optimizer is not able to meet the desired design goals on this program run two options are available. First, change to a different optimization routine from the list of available ADS routines and rerun the problem. This was usually successful in improving the design. Second, the designer uses a new set of starting values for the feedback gains and repeats the design procedure. Both options might be used on particularly difficult cases.

The pole placement and robustness design procedure has consistently been able to find improved designs; however the program does not always yield acceptable designs. Certain problems require changes in the optimizer routine and modification in the initial feedback gain starting values in order to obtain acceptable designs. Using the IBM 3033 time share system the pole placement and robustness routine requires about ten CPU seconds to work a second order problem and on the order of 15 to 60 CPU seconds to run a forth order problem. The actual amount of time varies with optimization requirements and time share utilization.

The observer robustness design program requires two passes of the ADS program. In the first pass the feedback gains, F, of the controller are computed to obtain the desired pole locations. The second pass of the ADS routine is used to adjust the observer gains to recover the system robustness. The two pass procedure was chosen because it allows a smaller number of design variables at each stage of the optimization and much more efficient computer usage. Figure 5.2 shows how the observer is implemented. This diagram is algebraicly stated as;

57

$$\dot{\underline{x}} = \begin{bmatrix} A & -BK \\ FC & A_c-BK \end{bmatrix} \begin{bmatrix} x \\ \hat{x} \end{bmatrix} + \begin{bmatrix} B \\ B \end{bmatrix} r \qquad (5.11)$$

$$A_c = A - FC$$

where $\underline{x}$ is the state, $\hat{\underline{x}}$ is the estimator variable, $\underline{F}$ is the feedback gain and $\underline{K}$ is the observer gain. The design cf the feedback gains and the observer gains are accomplished as separate quantities in keeping with the separation prin-
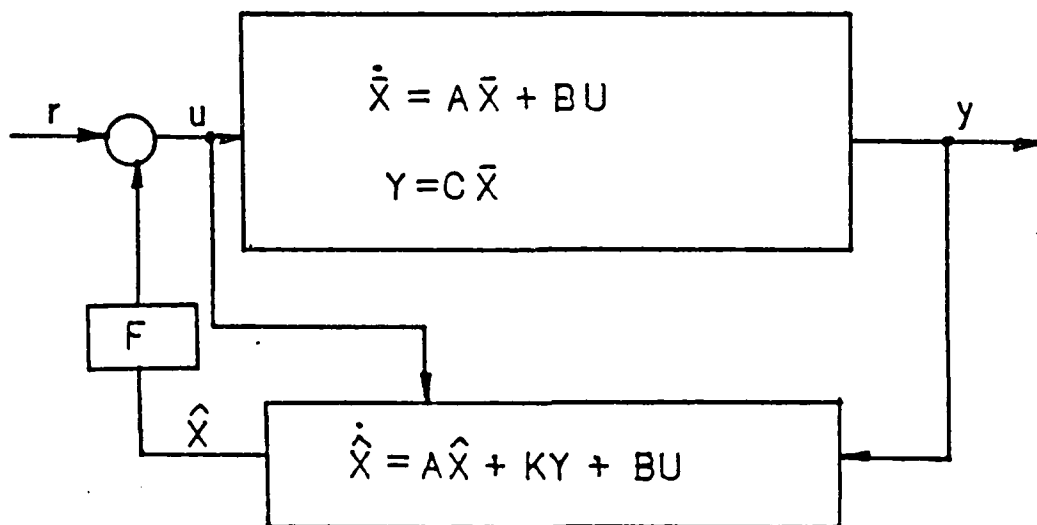
Figure 5.2   Observer Implementation.

ciple.   In using the pole placement and robustness design procedure for the observer system,   initial values of the F and K matrices must be input. The same or different optimization techniques from ADS may be employed.

The routines contained in this thesis have been based on the input additive singular value level. The pole placement and robustness design algorithm computes input additive,

58

output additive, input multiplicative, and output multiplicative singular values. Any of these singular values can be incorporated into the objective or constraint formulations but have not been for this version of the program.

In summary, the pole placement and robustness design procedure is a straight forward numerical optimization procedure for the practical application modern MIMO system analysis. The new aspects of the procedure are the implementation of both pole placement and robustness criteria within the same design program. The versatility of the pole placement and robustness design is obtained by incorporating a state of the art optimizer routine ADS, with currently available singular value computation routines. Using the optimizer format for the pole placement and robustness design gives the designer the ability to modify variables directly that affect both the time domain or performance of the system and the frequency domain or robustness of the system. The numerical optimization incorporated into the pole placement and robustness design is flexible enough to incorporate the double pass design technique using the separation of the feedback and filter gains to obtain robustness recovery for observer based controllers.

VI. INTRODUCTORY PROBLEM

The purpose of this introductory problem is to review some results of modern multivariable robustness theory using a simple problem. This same simple system is then used as a test problem for the optimization technique developed for this thesis. The problem provides excellent insight into the cross-coupling problem and demonstrates how effectively the pole placement and robustness design procedure can be in dealing with the cross perturbation terms.

The problem chosen for this introductory analysis comes from [Ref. 9]. Figure 6.1 is a diagram of this basic system. In this problem a simple plant is specified by the following linear system:

$$\begin{bmatrix} \dot{x}1 \\ \dot{x}2 \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x1 \\ x2 \end{bmatrix} + \begin{bmatrix} 1 & b_{12} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u1 \\ u2 \end{bmatrix} \tag{6.1}$$

where

$$y1 = x1 \tag{6.2}$$
$$y2 = x2$$

A feedback compensation of the form of equation 6.3 was assumed.

$$\begin{bmatrix} u1 \\ u2 \end{bmatrix} = -\begin{bmatrix} x1 \\ x2 \end{bmatrix} + \begin{bmatrix} uc1 \\ uc2 \end{bmatrix} \tag{6.3}$$

Which gives a closed-loop system, equation 6.4

$$\begin{bmatrix} \dot{x}1 \\ \dot{x}2 \end{bmatrix} = \begin{bmatrix} -2 & b_{12} \\ 0 & -2 \end{bmatrix} \begin{bmatrix} x1 \\ x2 \end{bmatrix} + \begin{bmatrix} 1 & b_{12} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} uc1 \\ uc2 \end{bmatrix} \tag{6.4}$$

This system has eigenvalues at -2 ,-2 and is therefore stable. Using equation 6.5

$$\underline{G} = \underline{C}(s\underline{I}-\underline{A})^{-1}\underline{B} \qquad (6.5)$$

the transfer matrix may be written as equation 6.6.

$$\underline{G} = \begin{bmatrix} 1/s+1 & b_{12}/s+1 \\ 0 & 1/s+1 \end{bmatrix} \qquad (6.6)$$

which gives the return difference matrix, equation 6.7.

$$\underline{I}+\underline{G}(s) = \begin{bmatrix} s+2/s+1 & b_{12}/s+1 \\ 0 & s+2/s+1 \end{bmatrix} \qquad (6.7)$$
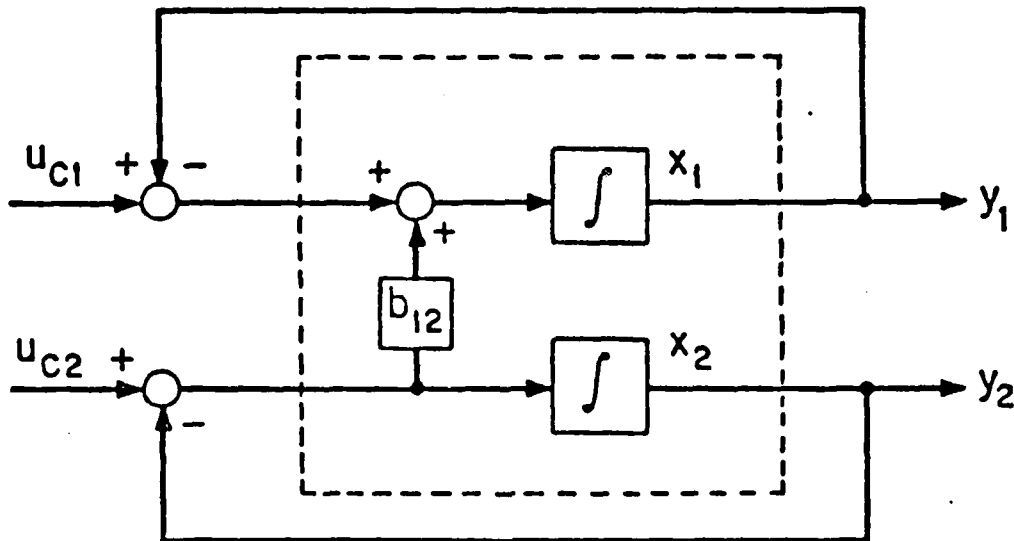
Figure 6.1   Basic Multi-input Multi-output System.

The problem shows the inadequacies of classical methods in establishing the robustness of the system. A brief review

of the results will be presented. Using the return differ-
ence matrix $(\underset{\sim}{I}+\underset{\sim}{G})$, the determinant may be written as
equation 6.8.

$$\det(\underset{\sim}{I}+\underset{\sim}{G})-1=2s+3/(s+1)^2 \tag{6.8}$$

Therefore, the multivariable Nyquist diagram will be as
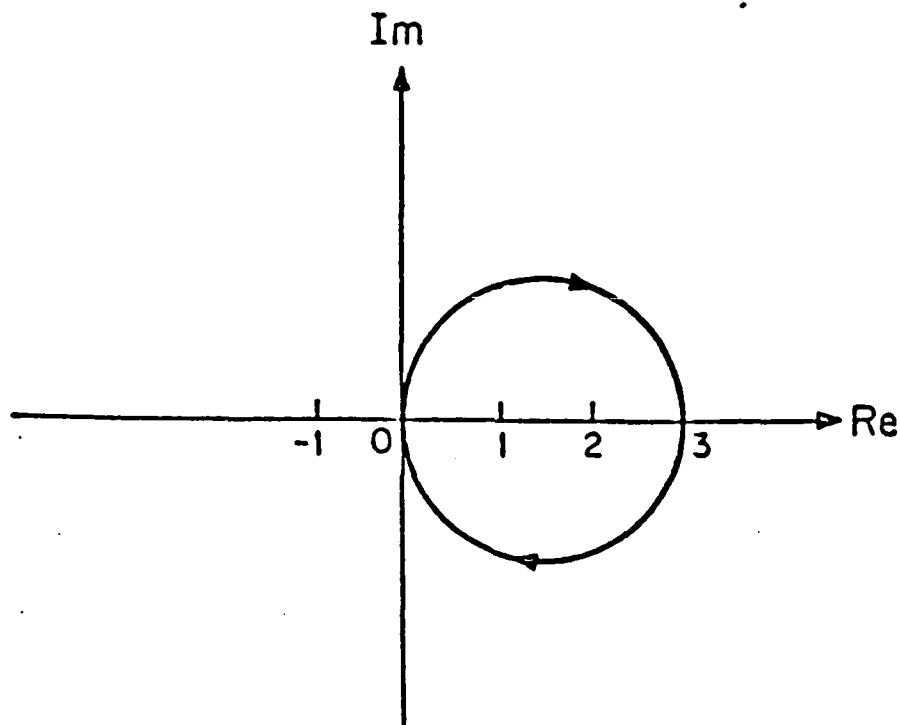
Figure 6.2   Multivariable Nyquist for $2s+3/(s+1)^2$.

shown in figure 6.2.   The diagram does not encircle the
(-1,0)   point and is indicative of a closed-loop stable
system. Considered as a SISO system one has a gain margin of
-1/3 to ∞ and a phase margin of ±106 degrees.   Under this
criteria one can conclude that this is a good design.   This
will be shown later not to be the case.

Multi-input systems are often designed a loop at a time.
Applying loop design to the system the transfer function for
either loop taken separately becomes equation 6.9

$$G = 1/s + 1 \qquad\qquad\qquad (6.9)$$

Figure 6.4 shows the Nyquist diagram of this problem. Using this Nyquist diagram the system is indicated to be stable and have phase and gain margins of $GM = (-1, \infty)$ and $PM = (\pm 180)$. This analysis does not show the true nature of the robustness of the system. Since the factor b is not a parameter in either of the Nyquist curves it plays no part in the stability determination using these diagrams.

Using the criteria of singular values discussed in Chapter 3 a measure of the nearness to instability for this problem may be obtained by plotting the minimum singular value of $(\underset{\sim}{I} + \underset{\sim}{G})$, σ. (For numerical calculations a value of $b_{/2} = 50$ is assumed). Figure 6.3 shows the plot of this value vs. frequency. This gives a minimum singular value of about -23 db or 0.071 which corresponds to a gain margin of about 0.93 to 1.08 and a phase margin of ± 4.1 degrees. These phase and gain margins are quite small and are evident in the cross-feed perturbation problem.

A perturbed system as shown in figure 6.5 can be produced which will lead to stability problems with very small values of perturbation. The closed-loop system if perturbed by a small perturbation, $5/b_{12}$, where b_{12} is a large number, will have as a characteristic equation, equation 6.10

$$(s\underset{\sim}{I} - \underset{\sim}{A}) = s^2 + 4s + 9 \qquad\qquad (6.10)$$

with the eigenvalues of $s = -2 \pm \sqrt{5}$. There is one positive root in this solution and the system is unstable.

To determine the nature of the robustness of this system the return difference matrix must be considered. If the return difference matrix of the transfer function $(\underset{\sim}{I} + \underset{\sim}{G}(j\omega))$

63
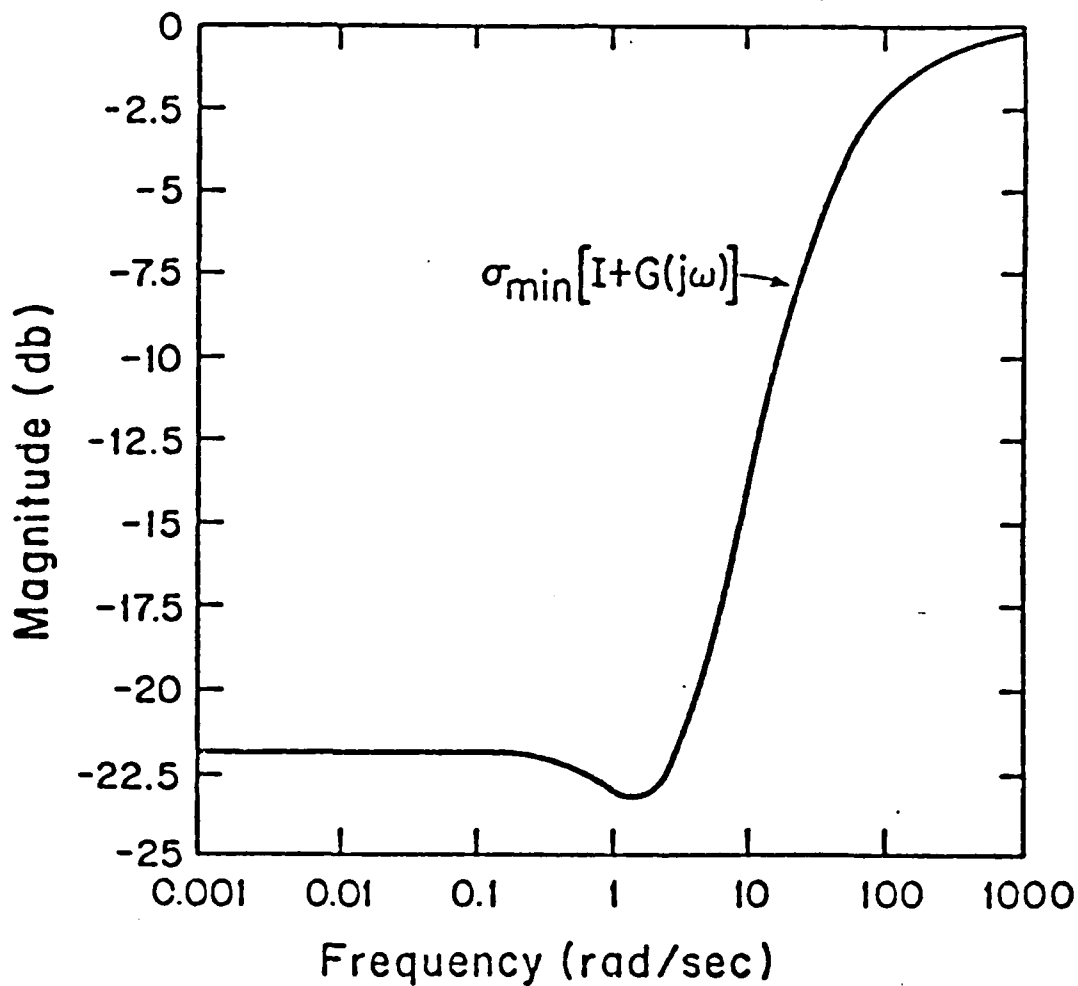
Figure 6.3   Minimum Singular Value Plot, Example Problem.

is nearly singular at some frequency $\omega_\bullet$, then the multivari-
able system will not be robust with regard to any modelling
errors within the system.   This is because any small change
in $\underline{G}(j\omega)$ can then make $\underline{I}+\underline{G}(j\omega)$ singular and the $\det(\underline{I}+\underline{G})$
becomes zero, thus changing the encirclements of the Nyquist
stability point and indicating a system instability.   Using
equation 6.1 the pole placement and robustness design method
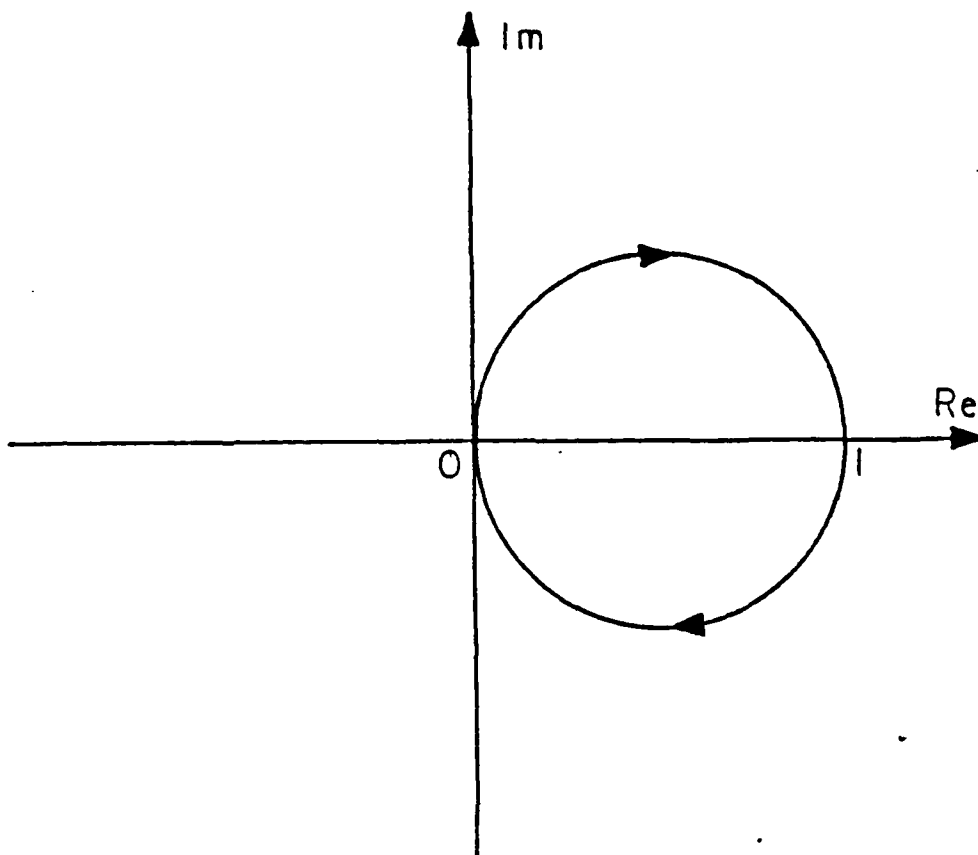
Figure 6.4   Nyquist Diagram of 1/s+1.

can be demonstrated . The first step is to establish a base-
line for the design. The state unity feedback model in this
simple problem gives pole locations of -2 and -2.   This set
of eigenvalues -2,   -2 were chosen as the baseline for the
system. Since a second order system requires only two feed-
back gains to place the poles, the diagonal feedback gains
were chosen for pole placement purposes. The pole placement
and robustness design program was then used to place the
poles and in the process obtained the gains required to do
this (1,1).   A plot of some of the singular value criteria
obtained is shown in figure 6.6.   Pole-zero plots of the

65
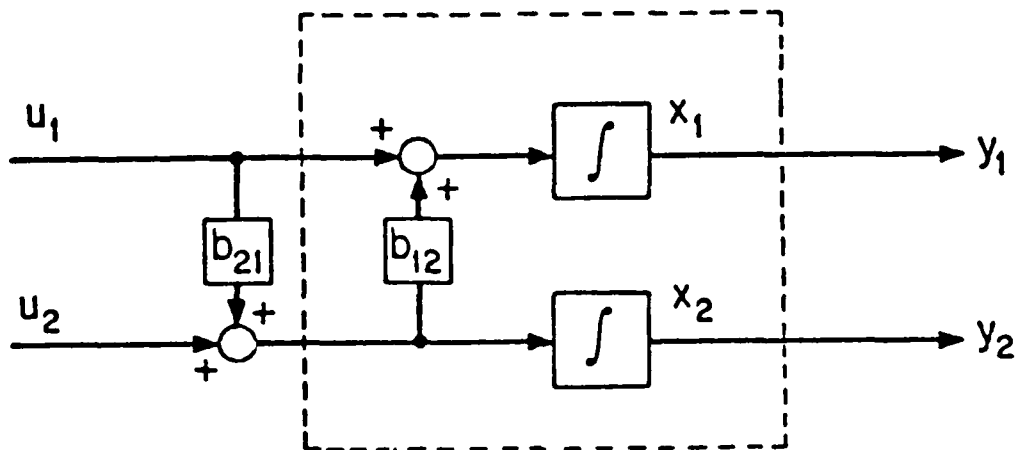
Figure 6.5   Perturbed System.

closed- loop transfer functions of the closed-loop transfer
matrix are shown in figure 6.7.   In this case the poles and
a zero are clustered about the -2 point and in the input two
to output one channel a zero is located at the -1 point on
the pole-zero diagram. This point closely corresponds to the
minimum singular value frequency.   The only significant
observations are the ability of the pole placement and
robustness routine to place the poles and the relatively
poor singular values indicative of low robustness.

Since there is a requirement for two feedback parameters
to set the poles of a second order system there should be no
additional freedom in design to account for robustness.
Case two was a run to demonstrate this fact. Again allowing
only two design variables for the problem the pole placement
and robustness design program was run but with an objective
function formulated to adjust the singular value above a
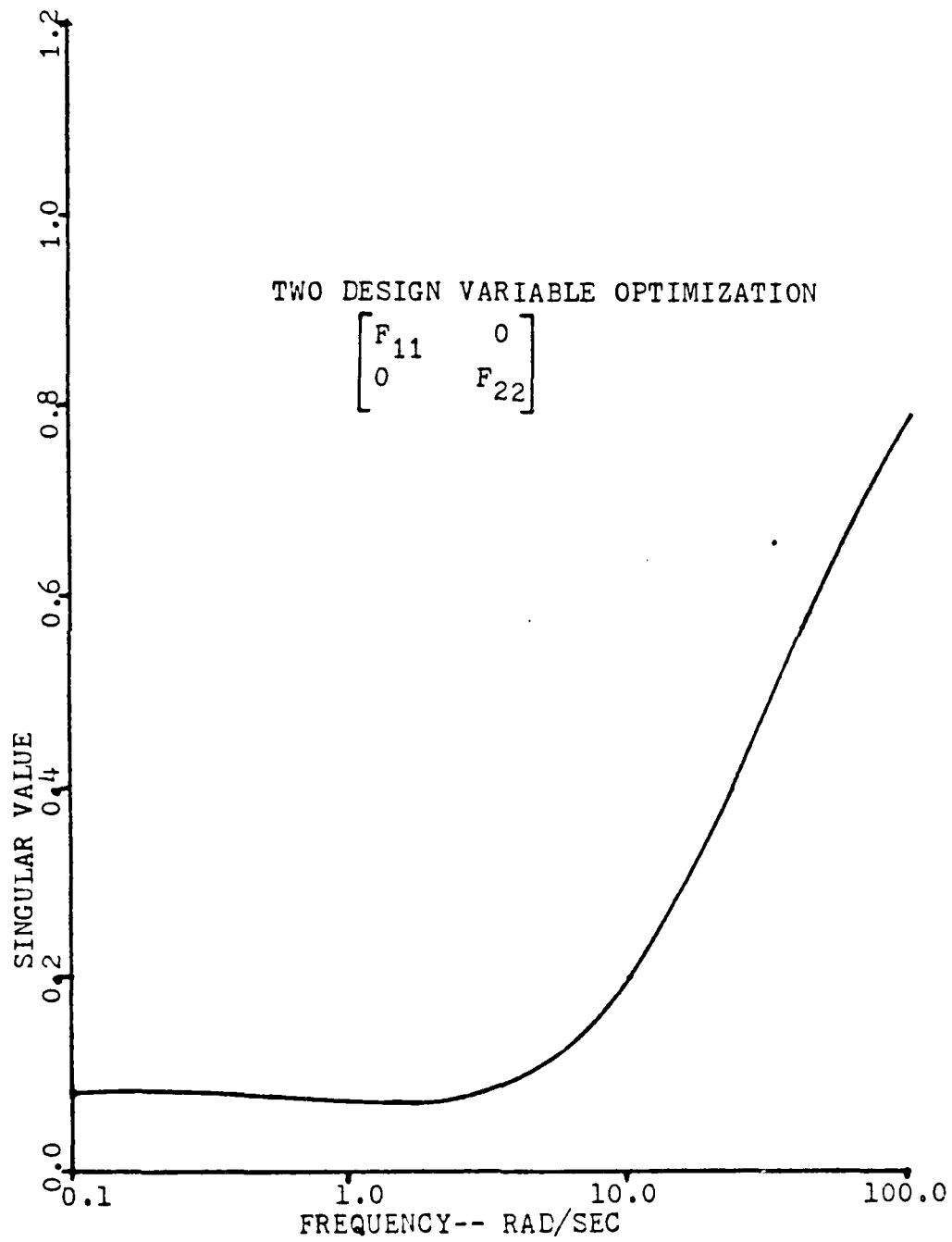
Figure 6.6   Singular Value Plot for Simple Problem.

67

1 TO 1   2 TO 1

2 TO 2
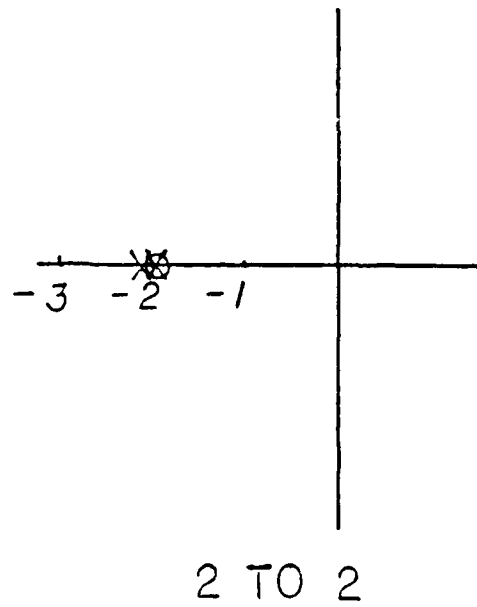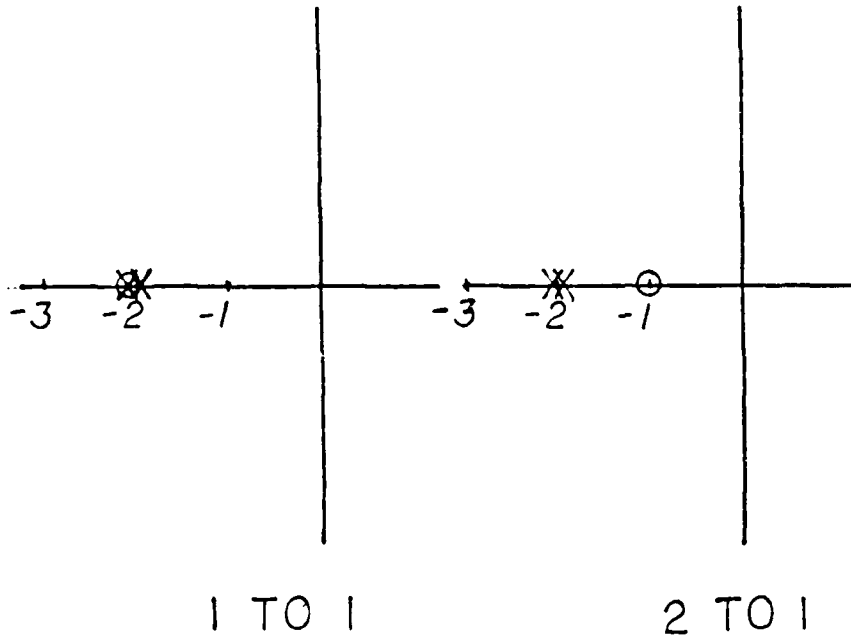
Figure 6.7   Closed-loop Poles for Pole Placement Only Case.

68

base level of 0.6. This level would correspond, using the universal gain margin chart, to a gain margin of -4 db to 8 db and a phase margin of ± 35 degrees. With only the two design variables to work with the pole placement and robustness design program was unable to place the poles and adjust the singular value level to the required value. The poles were placed at -2, -2 but the singular value minimum was still on the order of -23 db or 0.07. The pole-zero plot remained almost unchanged. It is clear that additional degrees of freedom for the pole placement and robustness design program must be opened if robustness is to be accounted for.

Case three was then run on the pole placement and robustness program by adding on additional degree of freedom. This case used the f gain as the additional design variable. A good choice as will be seen. Allowing the optimizer routine the extra freedom to adjust the additional feedback gain term an excellent design was found. The singular value minimum became 0.885 as shown in figure 6.8. Using the universal gain margin chart this corresponds to gain and phase margins of -6 db - 18 db and ±52 degrees respectively. This is a considerable improvement over the original design. The factor that changed the design was the upper diagonal feedback term which provides a cancelling factor for the cross-coupling term b_{12} = 50. This can be seen by looking at the system matrix equation 6.11.

$$\underline{A}_c = \underline{A} - \underline{B}\underline{F} \qquad (6.11)$$

where equation 6.12 gives A .

$$\underline{A}_c = \begin{bmatrix} -1-f_{11} & -f_{12} -b_{12}\, f_{22} \\ 0 & -1-f_{22} \end{bmatrix} \qquad (6.12)$$

69

THREE DESIGN VARIABLE OPTIMIZATION
$$\begin{bmatrix} F_{11} & F_{12} \\ 0 & F_{22} \end{bmatrix}$$
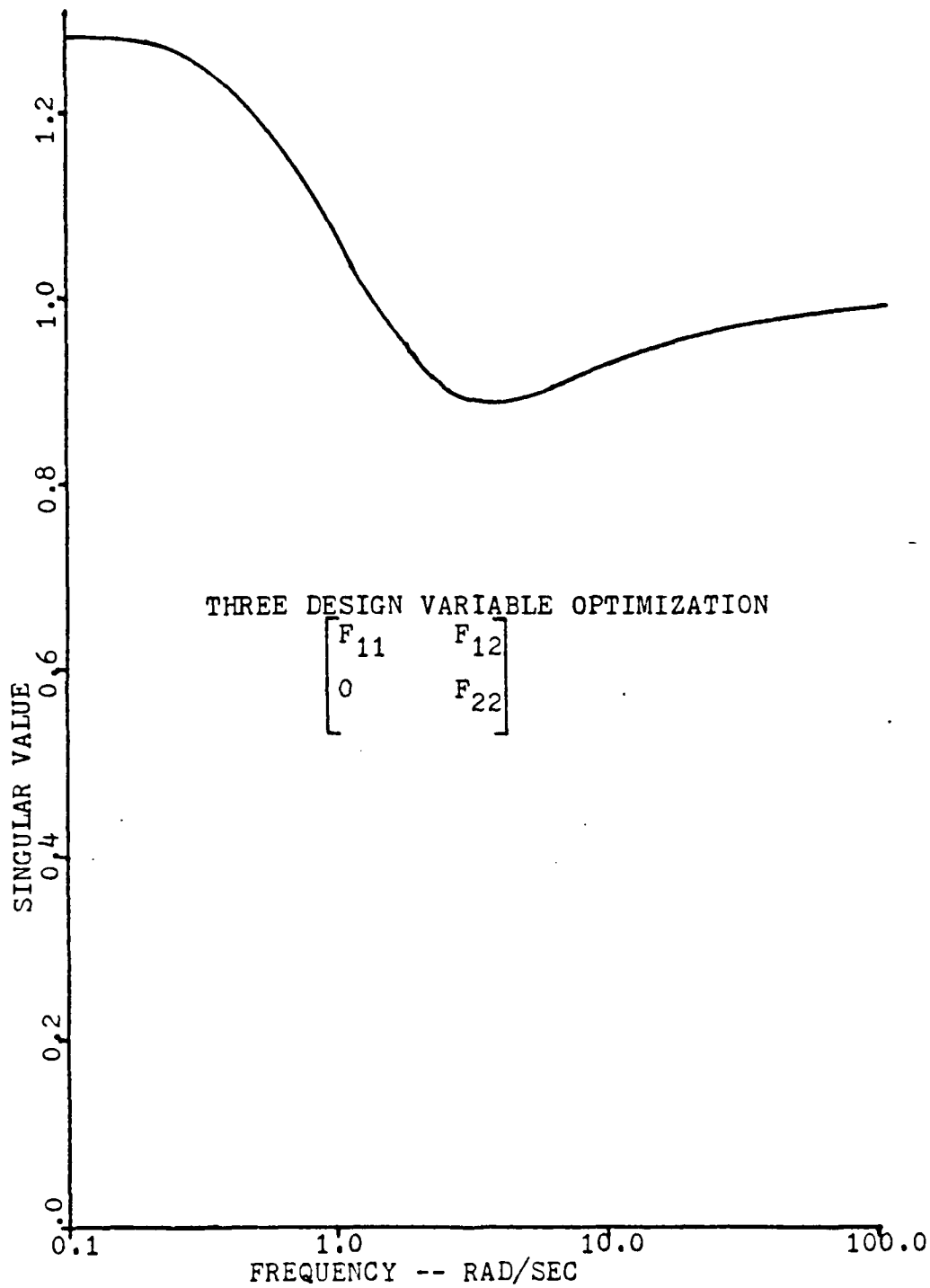
Figure 6.8   Singular Value Plot Case Three.

70

If the upper diagonal term of the matrix can be driven to zero the system will be decoupled into a diagonal system. To place the poles of this system at $-2,-2$ the diagonal feedback gains must be one. The optimizer feedback gains for this case were $f_{11} = 1.00006$, $f_{22} = 0.99998$ and, most importantly, $f_{12} = -51.83401$. Using these values in the A matrix

$$A = \begin{matrix} -2.00006 & 1.83501 \\ 0.0 & -1.99998 \end{matrix}$$

The value in the upper right position in the system matrix has been lowered considerably from the value of near 50 that appears in this position in the low singular value cases. Lowering this system gain value decreases the cross-coupling perturbation effects on the system. This can be more graphically demonstrated by figures 6.9 and 6.10. In figure 6.9 the transfer function shows a high gain of approximately 35 db and a bandwidth of 50 rad/sec. In figure 6.10 this gain has been reduced to 6.0 db with a bandwidth of almost 2 rad/sec. Two things are indicated by the figures, one, the open-loop Bode plot of the transfer function of the cross-coupled channel can be used to indicate the robustness problem as evidenced by the high gain and bandwidth relative to the other feedback gains and, two, the mechanism used by the pole placement and robustness design procedure to recover robustness is to reduce the relative gain and associated bandwidth within the affected channel. Figures 6.11 and 6.12 which are for the input one to output one channel are approximately the same as are the Bode plots for the input two to output two channel which are not shown. These figures indicate that no problem exists in the diagonal or direct coupling terms. The pole-zero diagram for optimization run case three is included in figure 6.13. The only significant change in this plot as compared to figure 6.7 is the movement of the zero in the off diagonal pole-zero plot. The zero is seen to shift to one of the pole locations

71
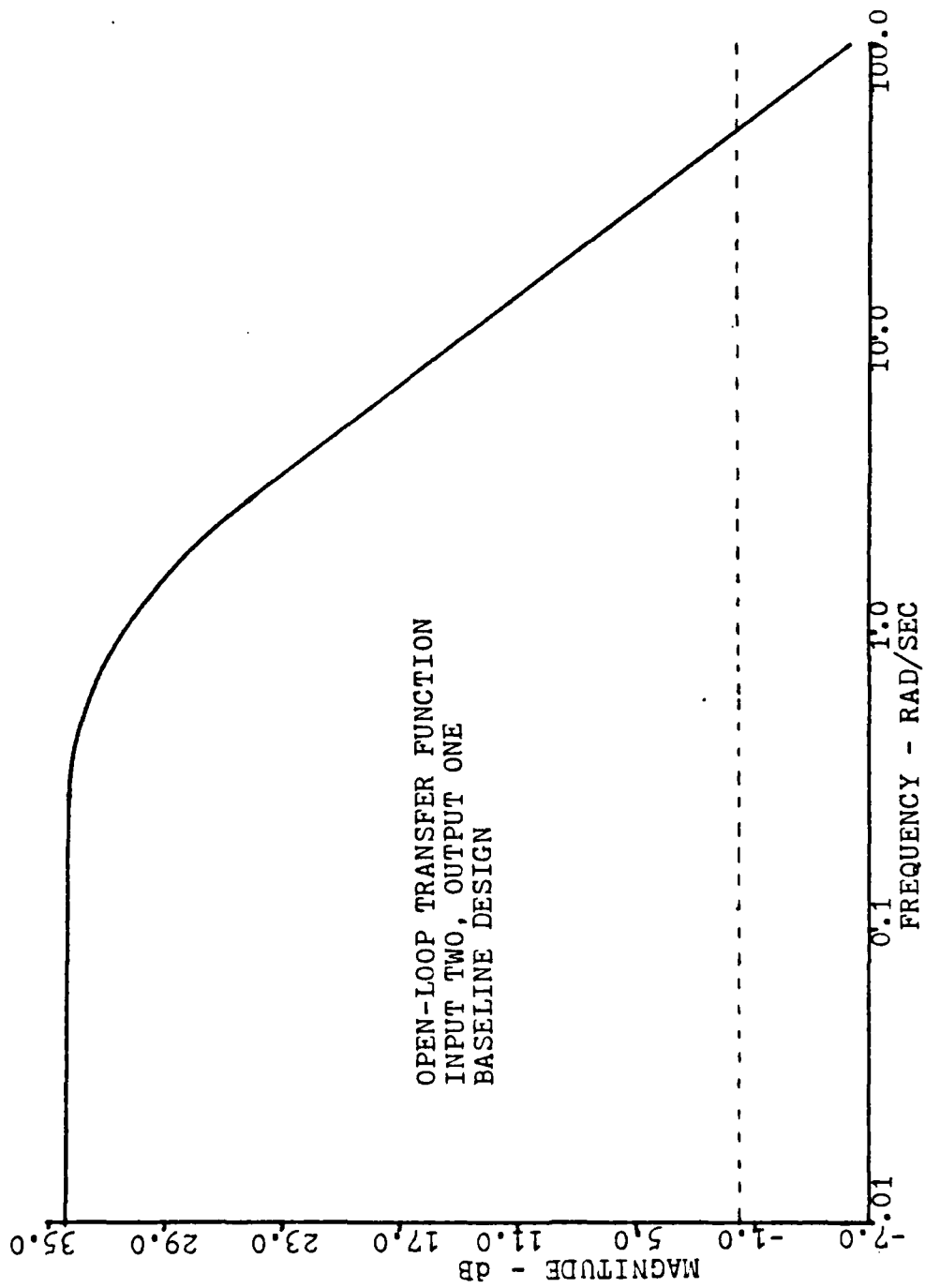
Figure 6.9   Open-loop Transfer Function 2-1 for Baseline.

72
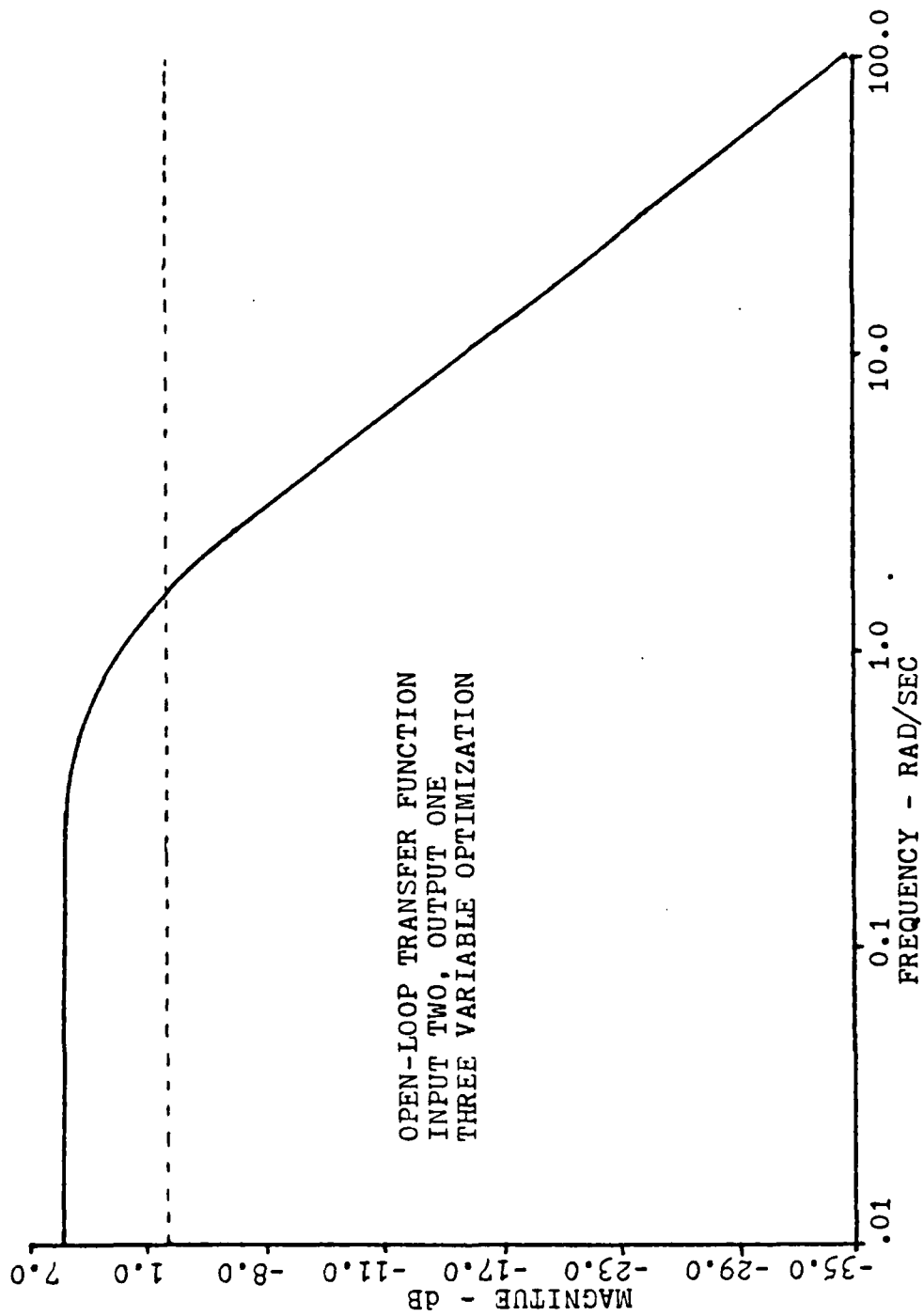
Figure 6.10   Transfer Function 2-1 Optimized(3 Var).

73
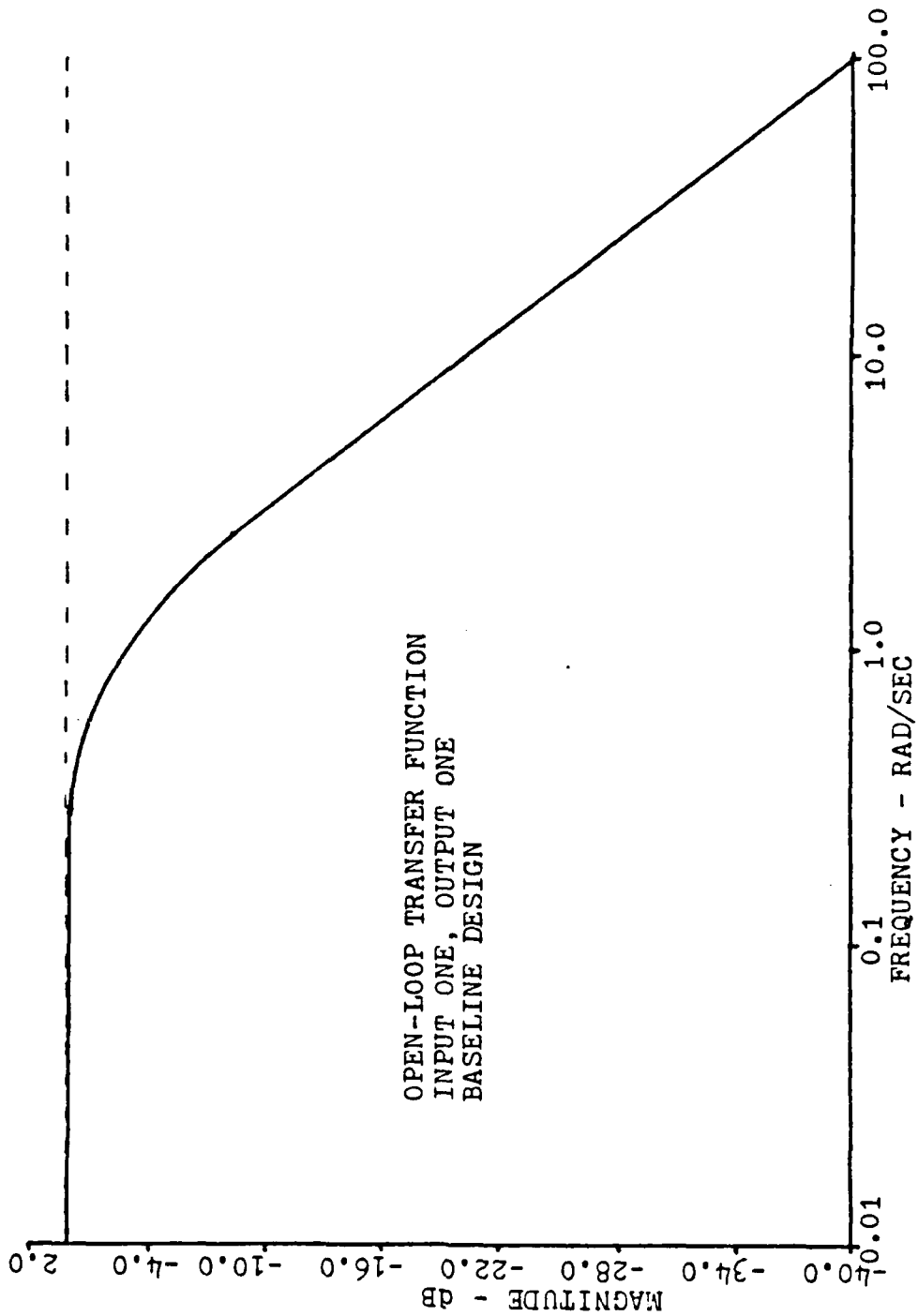
Figure 6.11   Transfer Function 1-1 Baseline.
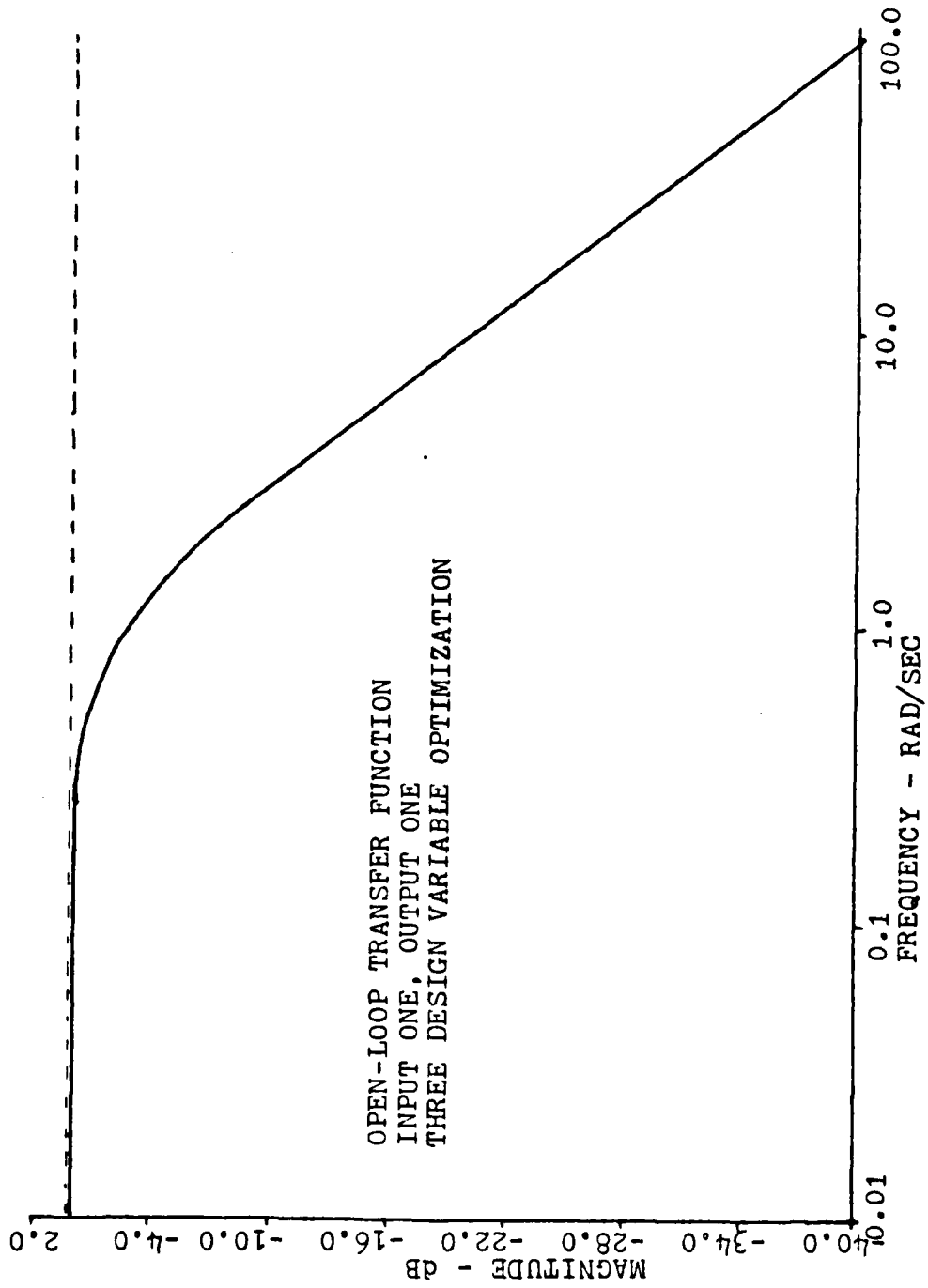
Figure 6.12   Transfer Function 1-1 Optimized (3 variables).

75

ΙΤΟΙ 　　　　　　 2 TO Ι

2 TO 2
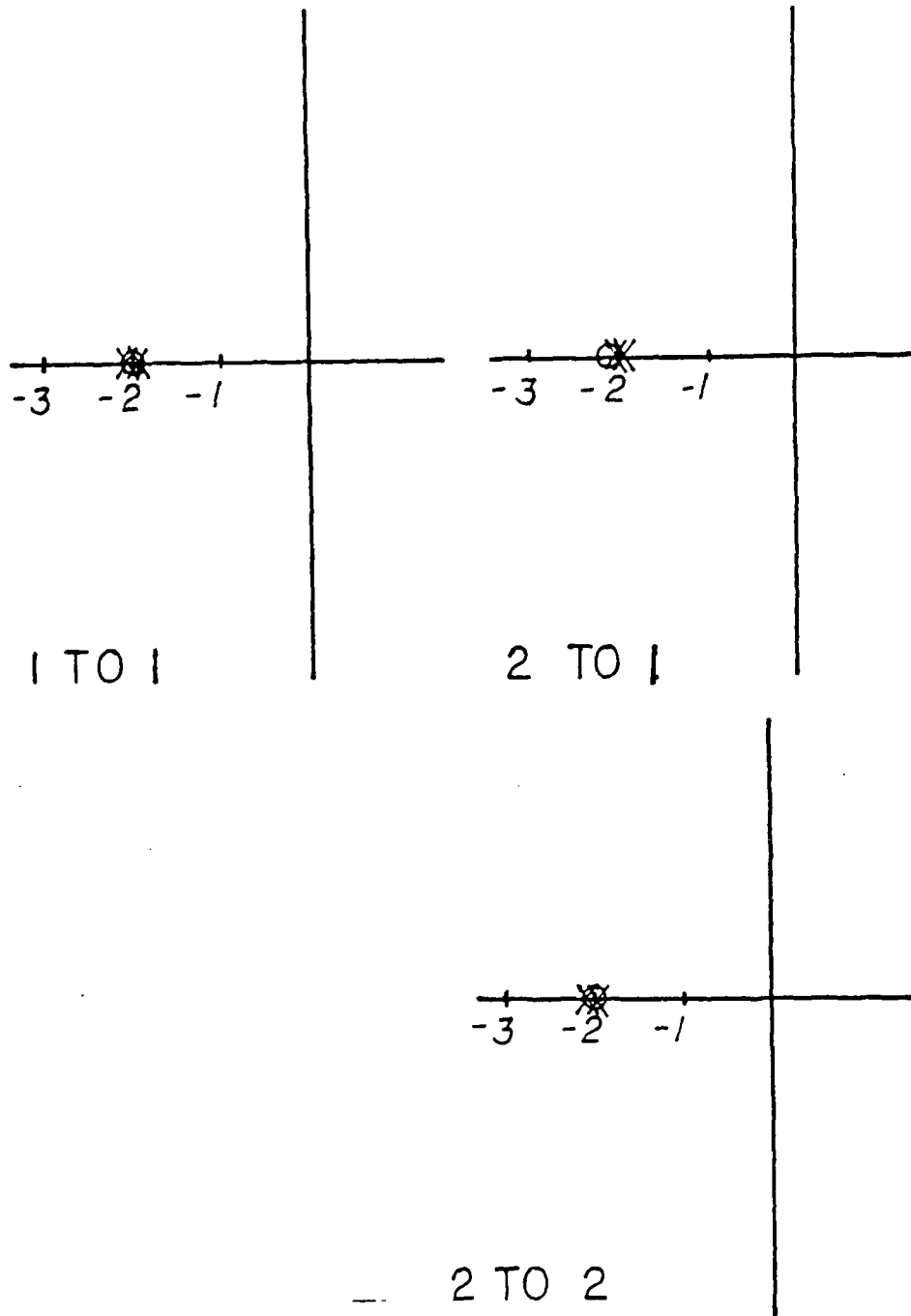
Figure 6.13　　Closed-loop Pole-Zero Plot Case 3.

during the optimization. This zero shift has the effect of smoothing the frequency response curve in the vicinity of the frequency of the minimum singular value providing a more uniform gain distribution around this point. Table 1 shows comparison results for the feedback gains for several cases of this basic problem.

The logical extension of the problem to the general case is to allow all four feedback gains to become design vari-

TABLE 1
Comparative Results Simple Problem

	Feedback Gains			
	f_{11}	f_{12}	f_{21}	f_{22}
f_{12} variable $f_{21}=0$	1.00006	-51.83401	0.0	0.9998
f_{21} variable $f_{12}=0$	0.25254	0.0	-0.01071	1.52332
f_{21} variable $f_{12}=-50$	.99960	-50.0	0.00025	0.99038
All f's variable	1.11936	-60.4718	0.00280	0.75478

ables. The pole placement and robustness routine can then use full freedom in choosing all four of these feedback gains to compensate for any cross-coupling effects within the system. Based on the previous analysis the two diagonal gains would be expected to approach 1 while the upper off diagonal gain moves to -50 and the lower one moves to 0. The flexibility of the ACS program was required for the four design variable study. Several runs were made with various combinations of starting parameters for the feedback gains and optimizer routines before a good design for the case

incorporating all four feedback gains as variable was obtained. While this points out one of the limitations of optimization design routines the program was able to develop an improved design over the baseline while employing four design variables. In the design produced for this case the singular value level was placed above 0.6 for a gain margin of -4 db to 9 db and phase margin of ± 35 degrees. In this formulation the optimizer was able to place the poles and meet the design singular value level. The feedback gains produced by the optimizer were:

1.11936	-60.4718
0.0028	0.75478

which are approaching the analytic design gains. After obtaining the feedback gains the OPTSYS program was used to obtain the necessary data to do a closed-loop pole-zero map. This plot is shown in figure 6.14 These plots indicate a similar pole-zero location to that found in the previous three design variable problem. Again the gain in the affected channel has been reduced to compensate for the high cross-coupling perturbation within the system.

The design studies presented to this point have been based on breaking the system loop at the input as shown in figure 6.15. In multivariable theory the location of the break in the loop changes the return difference for the system and the transfer function formulation. In the figure number 1 depicts a system with an input loop break point while number 2 is an output loop break point for output return difference determination. The return difference function for the point 1 is written as $I+FG$ while the return difference for point 2 is $I+GF$. The baseline system not only has low singular values of the input return difference matrix, the lowest being 0.0706, but also has low singular values for the output return difference matrix of point 2 of about the same order. To demonstrate the versatility of the
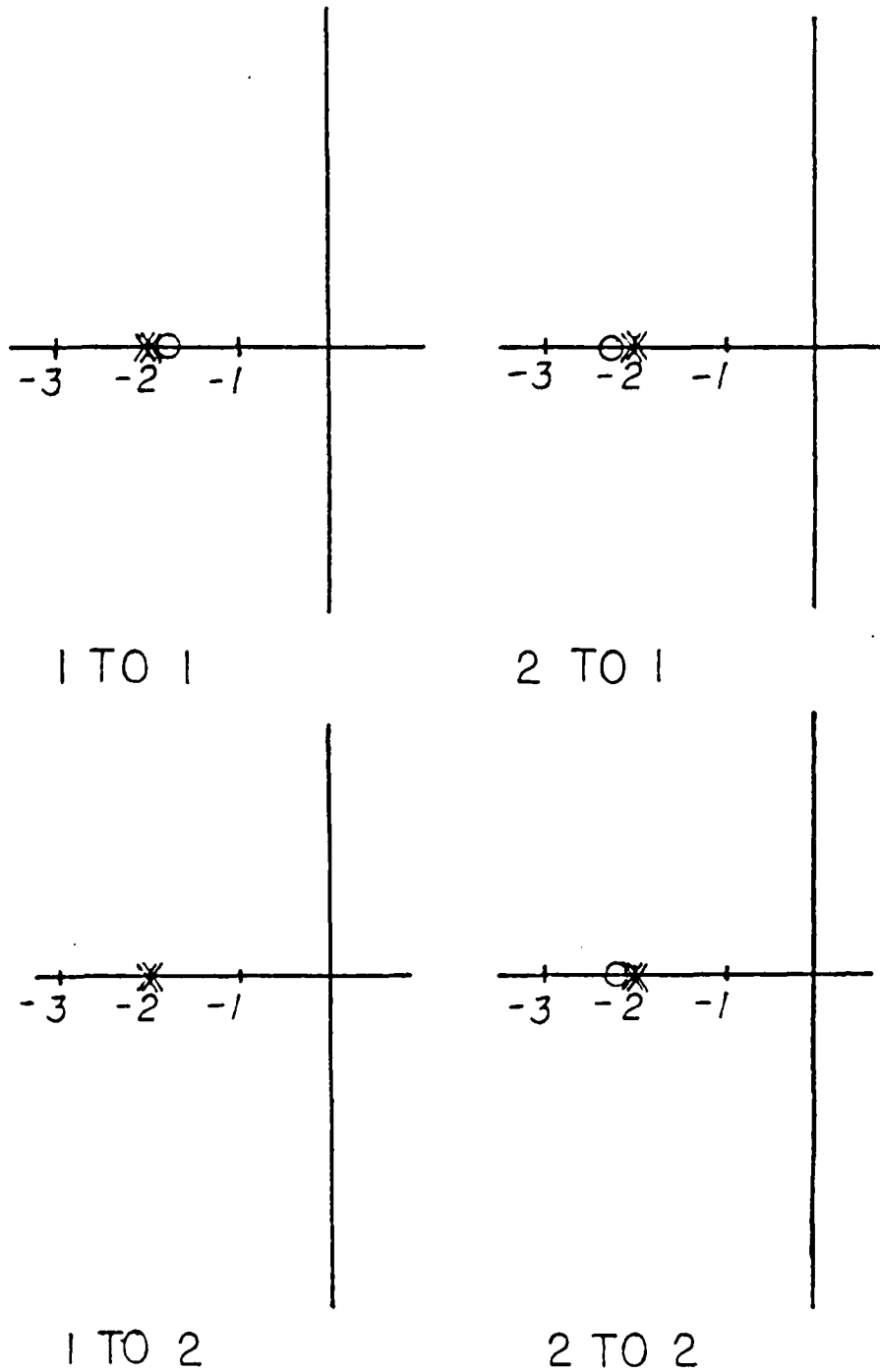
Figure 6.14   Pole-Zero Plot Simple Problem (4 Variables).
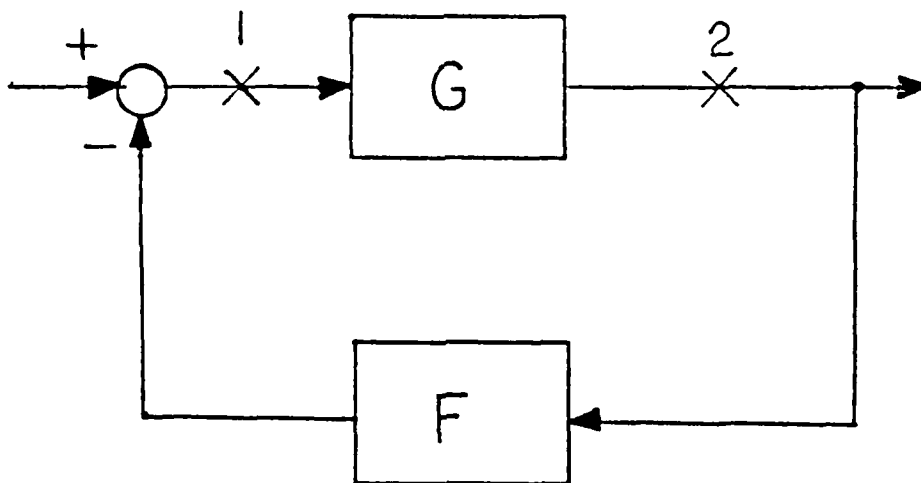
Figure 6.15   System Block Diagram.

pole placement and   robustness routine a design   was made in
which the output singular values were specified   instead of
the input values.   The pole placement and robustness routine
produced a design with gains that provided a highly robust
system on the output side with gain and phase margin of -6
db to   and phase margin of ±60 degrees. The design was not
robust on the input side.   Thus, designing for robustness at
one point in the system does not necessarily give robustness
at all points within the system.

Cne final case that should be discussed is that of
setting both input and output robustness criteria at the
same time.   Excellent results were obtained for this case.
The design routine placed the poles at -1.97±.009j with
feedback gains of

               0.85793   -45.91757
               0.00425   0.87137

The input singular values were raised to a level of 0.74984
or -4.5 db to 12 db gain margin and ±43 degrees phase

margin. The output values were raised to above 0.822 which corresponds to -5 to 15 db gain margin and ±49 degrees phase margin.

Ic summarize, it can be stated that the robustness problem for this system exists in the upper cross-coupling channel (input two, output one). The lack of robustness can be discovered in two ways. The first method is to plot the open-loop Bode plots cf each element of the transfer matrix and lock for extremely high gains and bandwidths relative to the other transfer functions. The second method examines the singular values of the return difference matrix for magnitude. Low singular values correspond to low robustness. The pole placement and robustness design routine can increase robustness by modifying feedback gains to reduce the effect of cross-coupling within the system. Observing the gain modification made by the pole placement and robustness routine the critical channel within the system that affects the robustness may be determined from the Bode plots. The pole placement and robustness routine feedback gain changes also cause zero shifts during the robustness recovery. The gain on the open loop Bode plot for the affected cross-coupling channel is adjusted and the closed-loop zeros, as seen cn the pole-zero diagram, are shifted. This zerc shift is in a direction which will combine with system poles to smooth the frequency response diagram in the vicinity of the minimum singular value.

81

VII. A HELICOPTER STABILITY PROBLEM

This chapter will deal with a more practical application of the numerical optimization program. In this problem the combined pole placement, robustness design procedure will be applied to the linear lateral dynamic channels of a CH-47 helicopter. The model is a highly coupled two-input two-output system that has been studied for its basic robustness characteristics [Ref. 16]. The usual procedure for design of highly coupled systems is to obtain a diagonally dominant closed-loop system. This diagonal system will be stable but not robust to cross-feed parameters. Sandell, et al, produced three designs. Two of these designs, while meeting basic performance criteria, had poor robustness. The third design was a relatively good design. The numerical optimization technique developed in the thesis was applied to the two poor designs and shown to provide substantial improvement in robustness.

The systems were designed to satisfy specifications to step input response and stability margins as stated in military specifications. Specific design parameters for each of the three designs presented were not available. The designs were all considered to meet the performance specification criteria and stability margin requirements. It was shown that two of these designs were extremely sensitive to model errors. The classic Nyquist techniques did not predict this sensitivity. The singular value analysis did indicate sensitivity problems.

The model is that of a CH-47B helicopter lateral dynamic system in hover. The dynamic model of the system is

$$\dot{\underline{x}} = \underline{A}\,\underline{x} + \underline{B}\,\underline{u} \qquad (7.1)$$

$$X = (v, p, r, \psi) \qquad (7.2)$$

$$\underline{u} = (\delta_\beta, \delta_c) \qquad (7.3)$$

where

$$\underline{A} = \begin{bmatrix} 2.27 & -1.42 & -0.15 & 31.99 \\ 0.01 & -0.7 & -0.07 & 0 \\ 0.04 & -0.05 & -0.5 & 0 \\ 0 & 1 & 0.11 & 0 \end{bmatrix}$$

and where

$$\underline{B} = \begin{bmatrix} 0.12 & 0.95 \\ 0.04 & -8.37 \\ .34 & 0.02 \\ 0 & 0 \end{bmatrix}$$

with full state available for feedback. Table 3 is a summary of parameters. The system is not open-loop stable.

Three control laws are formulated to satisfy the desired performance specifications. Equation 7.4 is the basic control law.

$$\underline{u} = -\underline{F}_i x + h_i \, \varphi_c \qquad (7.4)$$
$$i = 1, 2 \text{ or } 3$$

where the following values of F and h were are:

$$F_1 = \begin{bmatrix} -1.72 & -23.5 & 70.6 & 595. \\ 0.024 & -2.71 & 0.368 & -7.99 \end{bmatrix}$$

$$F_2 = \begin{bmatrix} 0.198 & 154.0 & 18.3 & 142.0 \\ -0.01 & -1.592 & -0.189 & -1.47 \end{bmatrix}$$

$$F_3 = \begin{bmatrix} 0 & 0 & 25.5 & 0 \\ 0 & -4 & 0 & -27 \end{bmatrix}$$

83

TABLE 2

CH-46 Helicopter Parameter Definitions

Variable	Units	Description
v	ft/sec	Vehicle body y-axis earth relative velocity component
p	rad/sec	Roll rate
r	rad/sec	Yaw rate
ϕ	rad	Roll attitude angle
δ_B	in	Yaw rate rotor deflection control
δ_C	in	Roll rate rotor deflection control

$$h_1 = \begin{bmatrix} 597.0 \\ -7.99 \end{bmatrix}$$

$$h_2 = \begin{bmatrix} 142.0 \\ -1.47 \end{bmatrix}$$

$$h_3 = \begin{bmatrix} 0 \\ -27 \end{bmatrix}$$

The φ_c is a step input command that must track. Figure 7.1 is a diagram of the control structure for the feedback control laws. Figure 7.2 is a detailed layout of the system. All three of the designs have negative real eigenvalues and provide stable overdamped responses. All three of the above designs meet the desired design specifications.

The loop Bode plots, [Ref. 16], indicate all three designs to have good stability margins when considered a loop at a time. Since these designs were all full state LQ

84
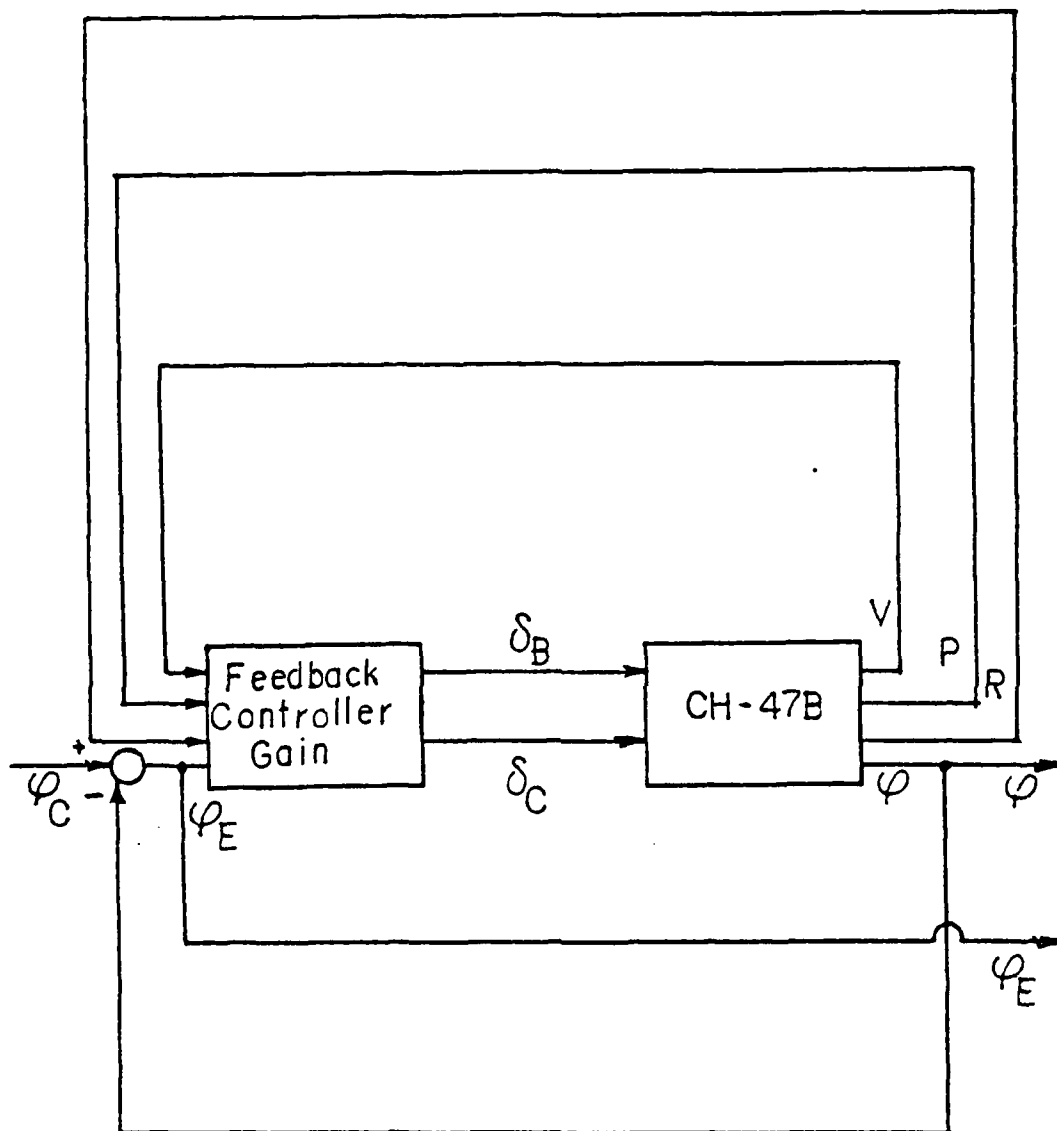
Figure 7.1   Feedback Control Structure.

85
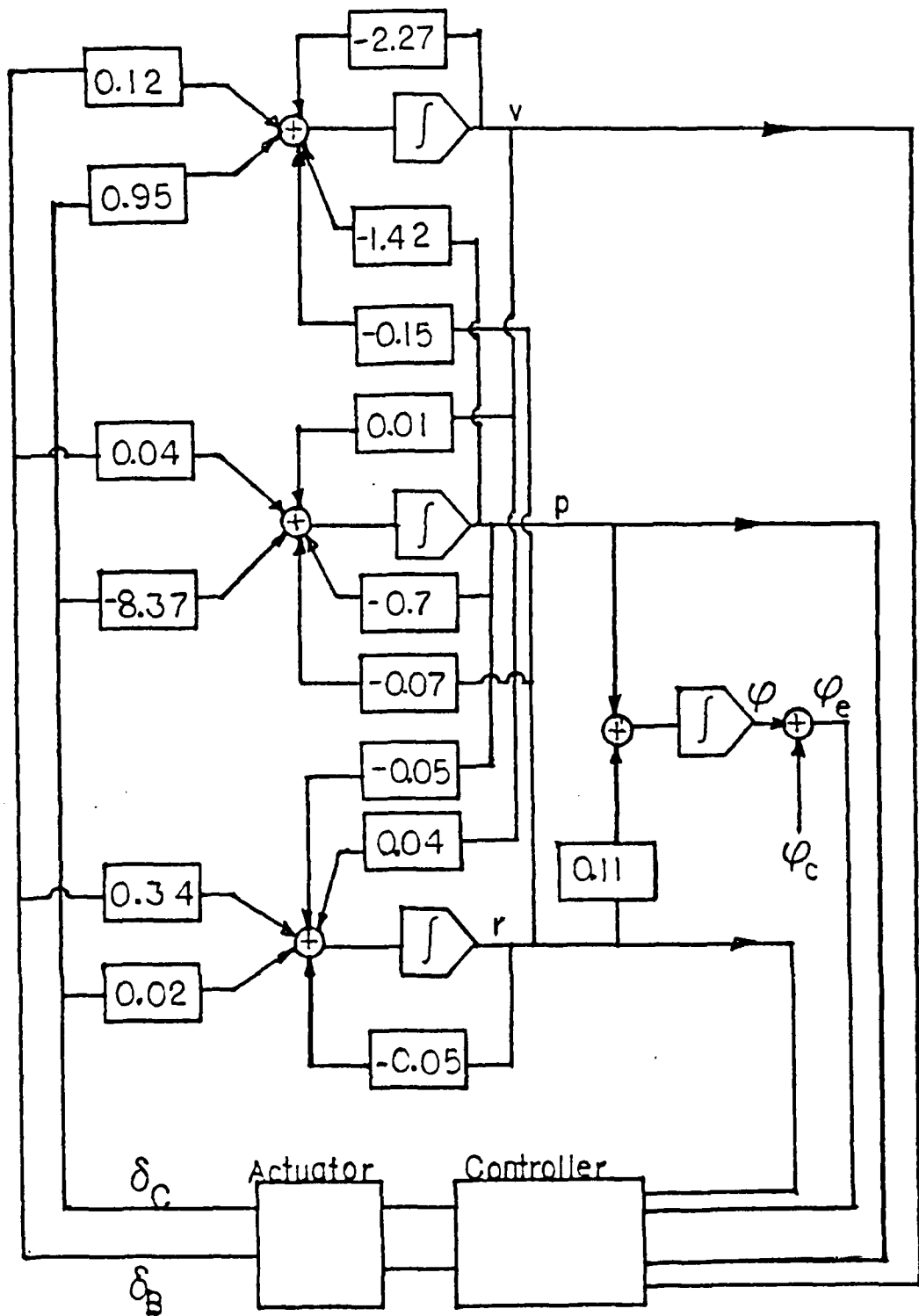
Figure 7.2   System Diagram.

86

designs with a diagonal control weighting matrix they should posses at least -6 db to infinite gain margin and 60 degrees of phase margin. When the singular values of the three designs are computed a robustness problem is indicated by low singular values of two of the designs. Figure 7.3 presents singular value plots of all three designs. This plot shows that designs 1 and 2 both have very low minimum singular values for the return difference matrix. Design 1 goes as low as -20 db near 10 rad/sec in frequency while design 2 is down to -34 db at frequencies up to 1000 rad/sec. Design 3 is a good design with singular values that remain above one throughout the frequency range of interest. Using the universal gain and phase diagram as discussed in Chapter 6 this equates to a gain margin of about -6 db to infinity and a phase margin of 60 degrees as expected.

The system stability in design 1 may be affected by perturbations occurring in the actuators as shown in figure 7.4. If the output axis coupling from δc spills into δ_B in the frequency range from 0.5 to 50.0 rad/sec with a magnitude of 0.12 and a phase of 60 degrees then the system can become unstable. This could be caused by nonlinear terms, worn parts, or system saturation.

The second design as shown in figure 7.5 may have a stability derivative variation between δ_B and p. If this derivative varies from about 0.04 to -0.96 the system can become unstable. The center of gravity location, trim of the aircraft and rotor coupling can all affect this derivative.

The pole placement and robustness design technique developed for this thesis research was applied to designs 1 and 2 to obtain an improved robustness for these designs. Design 1 will be considered first. In this design a cross-feed perturbation through the actuator can produce instability as indicated by the low singular values at lower frequencies. To study this problem with the pole placement
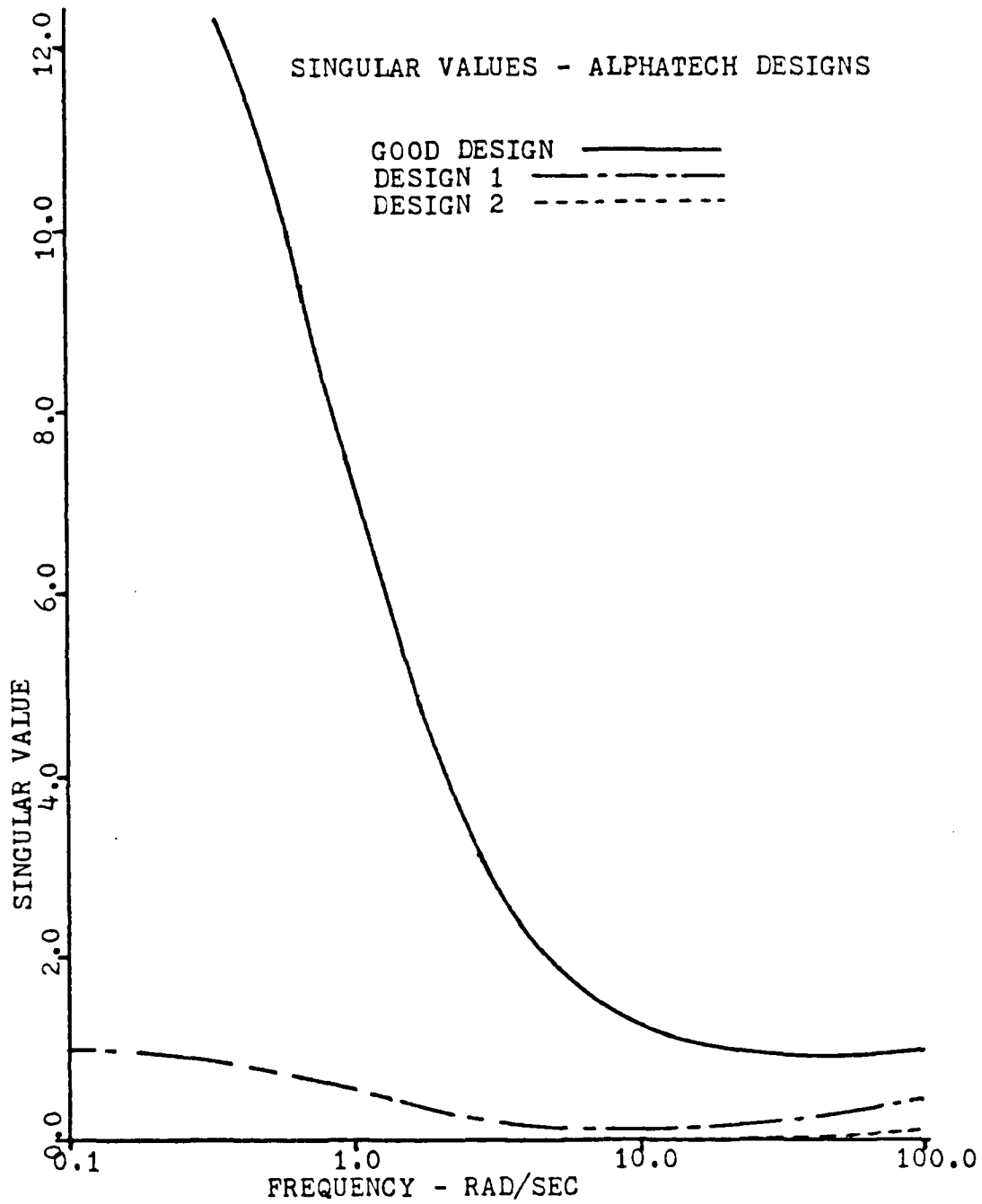
Figure 7.3   Alphatech Design Singular Value Plots.
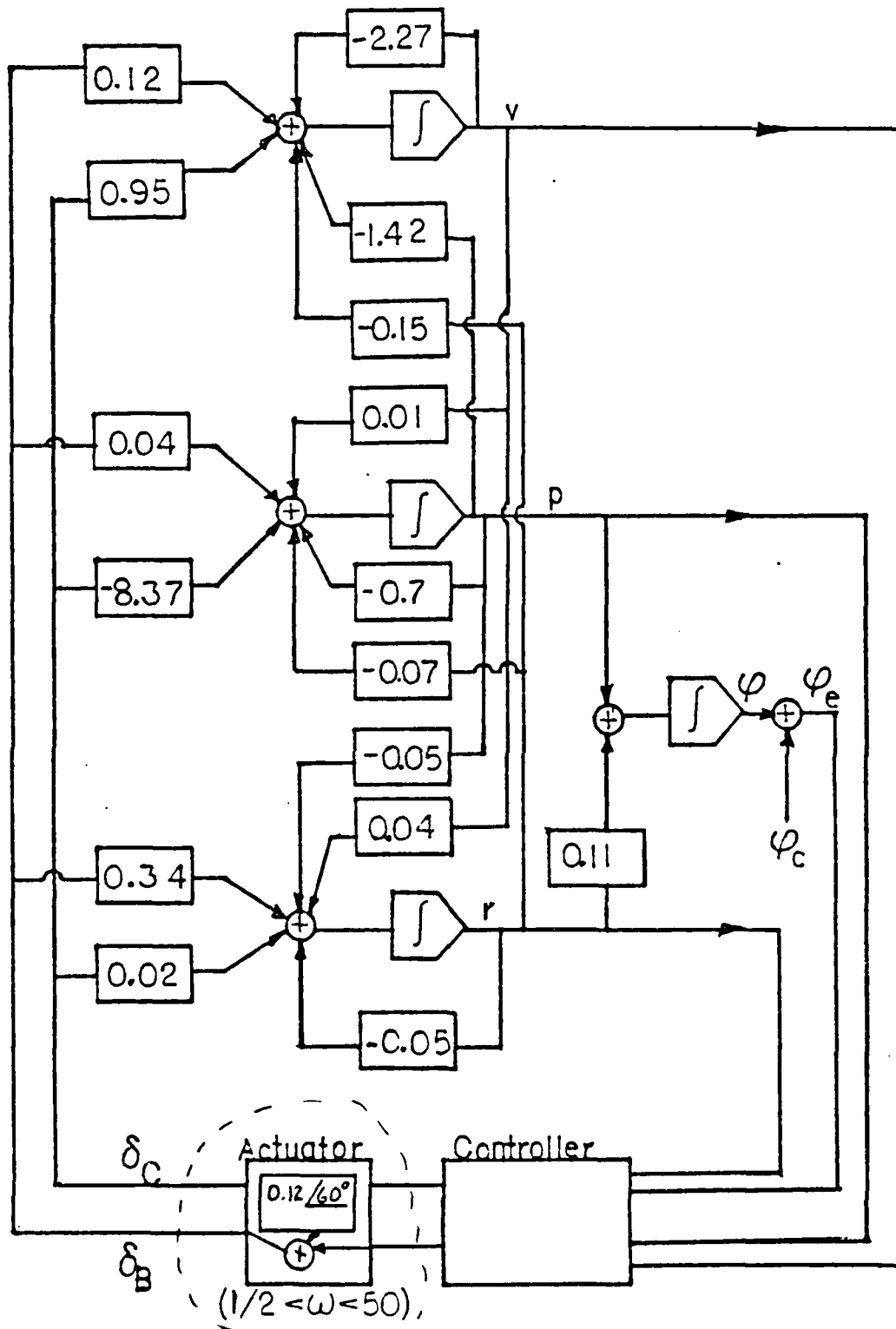
Figure 7.4 Design One Perturbation Input.

89
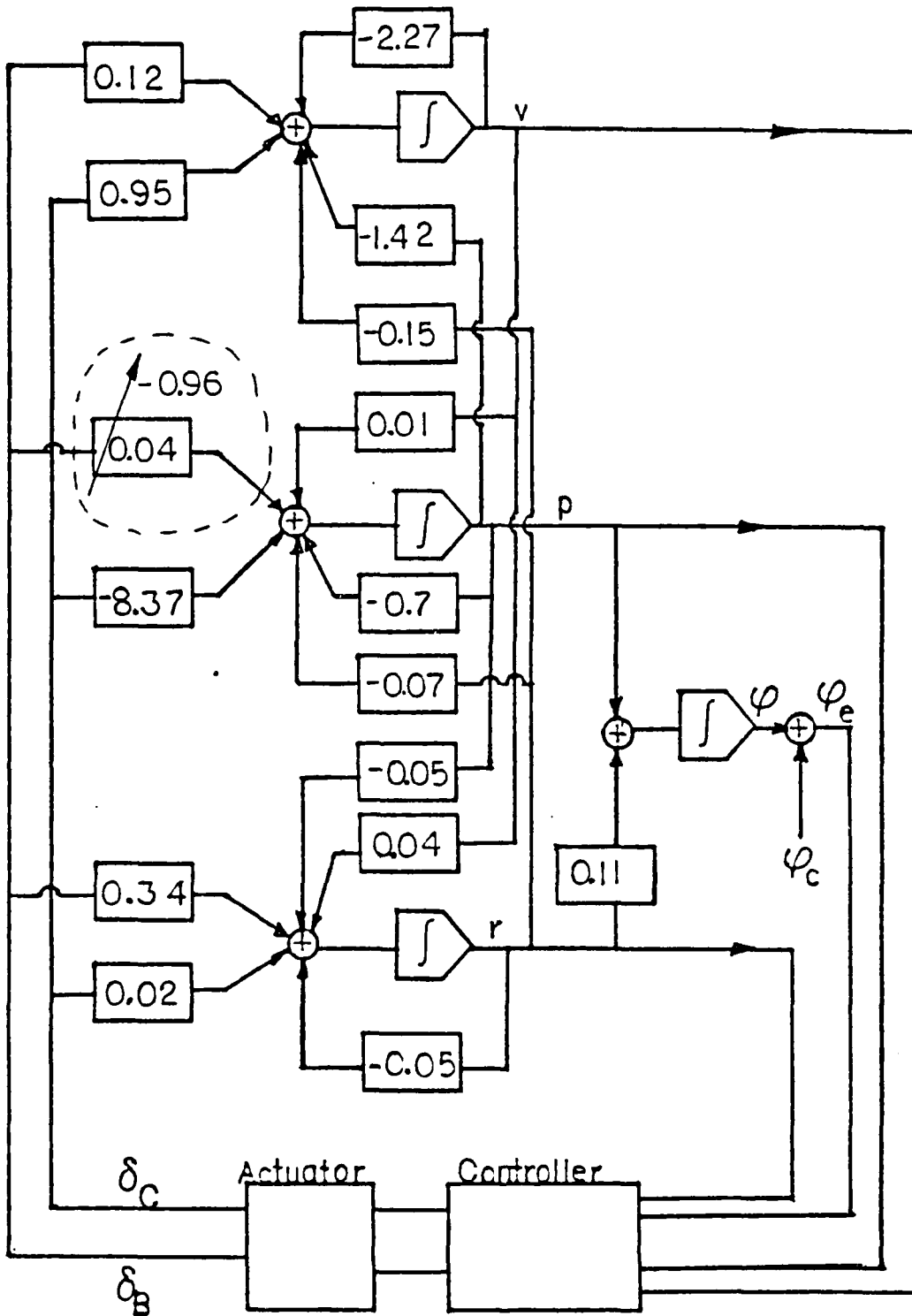
Figure 7.5   Design 1:0 Perturbation Input.

and robustness design method it is assumed that the eigenvalues (pole locations) of the system as developed in [Ref. 16]. are the required poles for the performance criteria. Once the pole locations are set, robustness criteria must be selected. From the universal gain and phase margin curve discussed earlier two choices of singular value levels were made for this problem. The first singular value level chosen was 0.6. This corresponds to a gain margin of -4.0 db to 8 db and a phase margin of about 35 degrees. The second value chosen was 1.0. With corresponding gain margin of -6 db to ∞ and a phase margin of 60 degrees which are the characteristics of a LQ regulator design with diagonal weighting matrices.

For singular value level 0.6 the pole placement and robustness design routine places the poles as shown in table 2 The slight differences in these pole locations appear to have insignificant effect on the performance as shown in the response curves of the system. The feedback gain adjustment moves the minimum singular value from about 0.11 with very poor phase and gain margins to a level of 0.66 . This is above the desired values of gain and phase. The improvement in robustness came from modifying the feedback gains in channel δ_B. By greatly reducing the gains in channel δ_B the optimizer minimizes the influence of the cross-coupling from channel δ_c. In this way a much larger spill over of channel δ_c may be tolerated through the actuator before the system will become unstable.

The primary mechanism of robustness improvement in this problem was a reduction in the gain levels of the affected channel. The feedback gains presented in table 4 show the modification of these gains from those utilized in [Ref. 16]. While all the gains are modified the f_{14} gain undergoes a significantly larger change than the other design 1 case 1 feedback gains. Looking at the open-loop

91

TABLE 3
Design One Pole Placement

Pole	Desired Location	Actual Location
1	-24.7977	-24.7893
2	-11.3635	-12.0083
3	-10.3288	-10.7752
4	- 2.1005	- 2.1181

transfer functions of this optimized problem and comparing them with a non-optimized set of transfer functions for δ_B to   as shown in figure 7.6 and 7.7 show very little change in the system gain, both are around 42.0 db, and a bandwidth of about 20 rad/sec. There is a significant change in the phase diagram which is caused by the zero location shift.

The transfer function for δ_c to δ_c depicted a gain increase of about 3 db for the optimized design while the phase remained similar for both transfer functions. In transfer function δ_c to δ_B some important aspects of the problem are observed. The Bode diagram of the open-loop transfer function of δ_c to δ_B clearly indicates the cross-coupling problems and the pole placement and robustness design routine's mechanism of optimizing the system gains to increase robustness. The gain is reduced from 93 db to about 78 db and the bandwidth is reduced from above

92
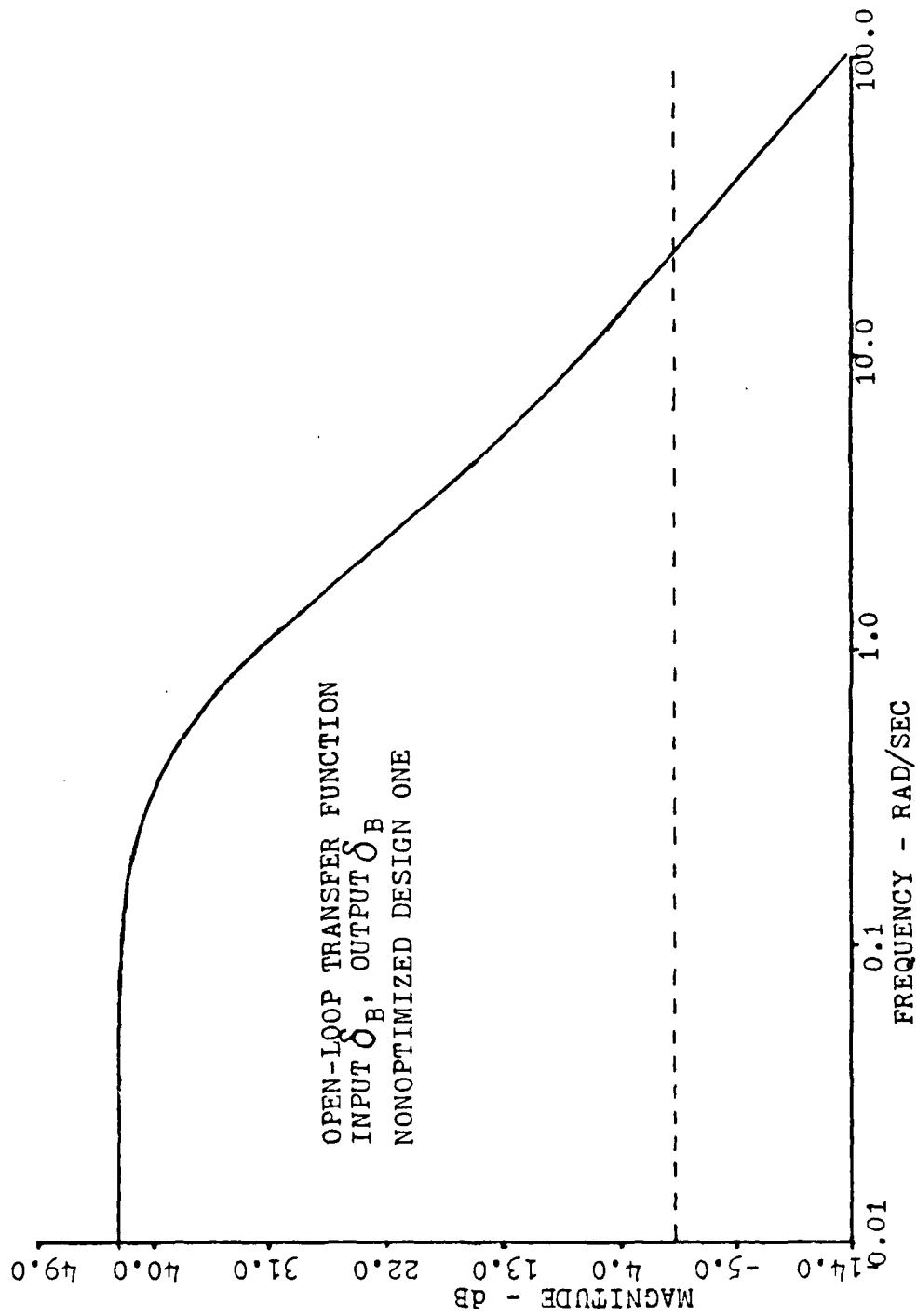
Figure 7.6   Transfer Function $\delta_B - \delta_B$ Design One, Nonoptimized.

93
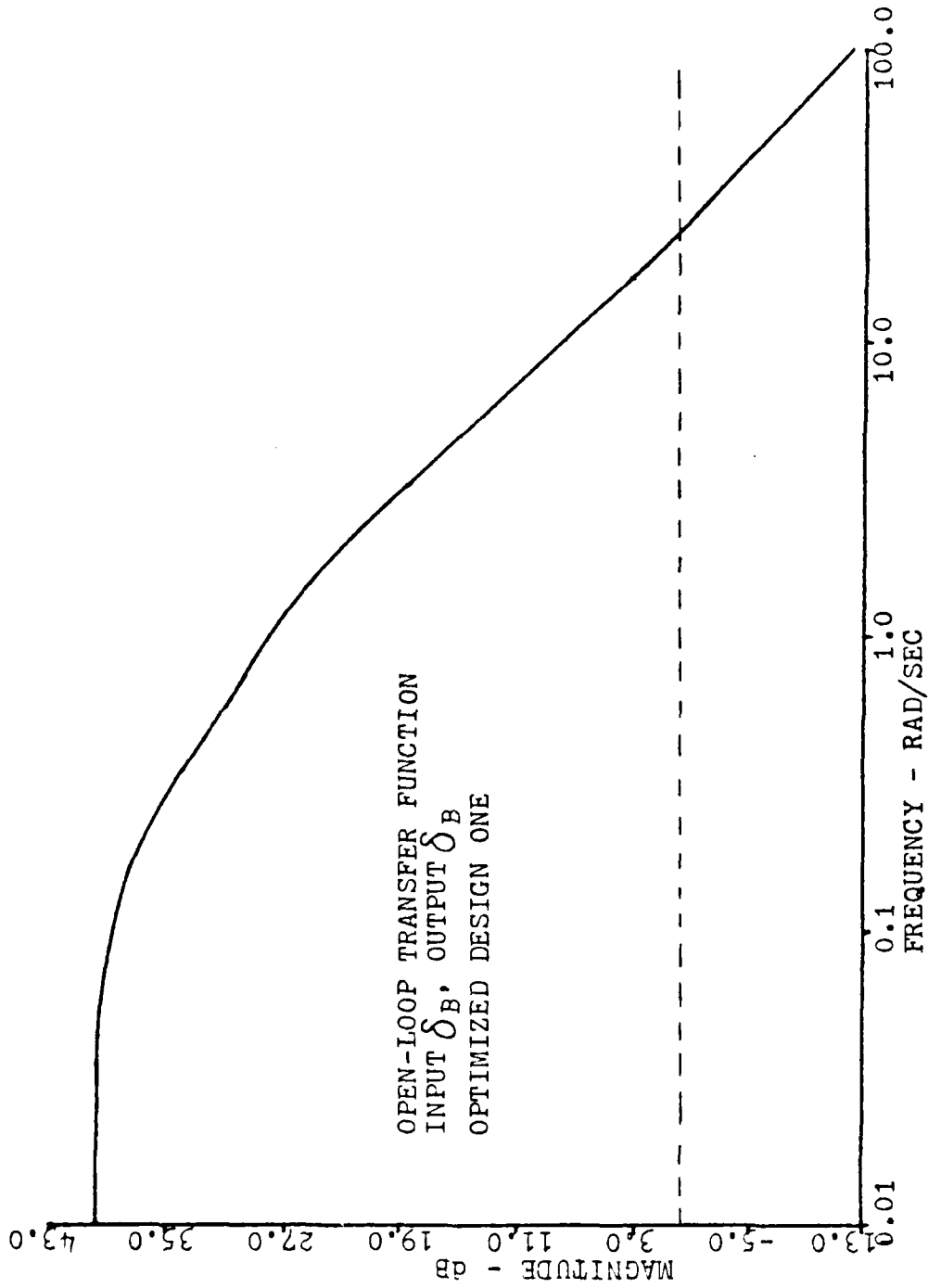
Figure 7.7   Transfer Function δ_B-δ_B Design One , Optimized.

94

OPEN-LOOP TRANSFER FUNCTION
INPUT δ_B, OUTPUT δ_B
OPTIMIZED DESIGN ONE

100 rad/sec to about 30 rad/sec. The reduction of bandwidth and gain in the loop δ_c to δ_B yields an increased tolerance to perturbation. The transfer function for δ_c to δ_B is shown in figures 7.8 and 7.9. In the transfer functions for δ_B to δ_c the bandwidth is slightly increased from 0.6 to 0.8 rad/sec and the gain actually increased from 9 db to 14 db. The big change in the overall system, however, is in the transfer function from δ_c to δ_B . This is the channel that the destabilizing perturbation enters and by greatly reducing the gain and bandwidth in this channel through a change in feedback gain, the optimizer routine has brought the entire system gains to more balanced conditions and recovered a highly robust design.

The gain changes associated with the robustness improvement cause the zeros of the various closed-loop pole-zero diagram to move. A comparison of the eight pole-zero diagrams is shown in figures 7.10 to 7.13. The significant feature of these pole-zero diagrams is the shift of the zeros of the optimized design in a direction that attempts to equalize or balance the frequency response for frequencies in the vicinity of the minimum singular values. The pole-zero diagram of δ_B to v will be discussed as an example of this effect. In 7.10 the nonoptimized zeros are located about $-2 \pm 4.5j$ and -19.8. When the pole placement and robustness routine has completed the feedback gain modification these zeros have shifted near -11 and $-6\pm3j$. The effect of these zero shifts is to combine with the pole locations to equalize the frequency response as depicted in figures 7.14 and 7.15. Zero shifts for the remainder of the transfer functions provide similar results in the other channels. By moving toward the frequencies associated with the minimum singular values the zeros have balanced the overall frequency response of the system in each channel. While the channel gain modification is the primary mechanism

95
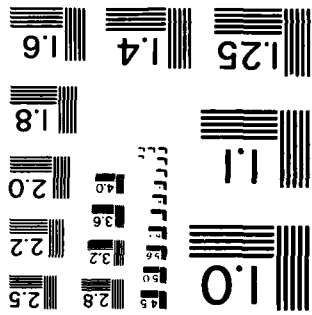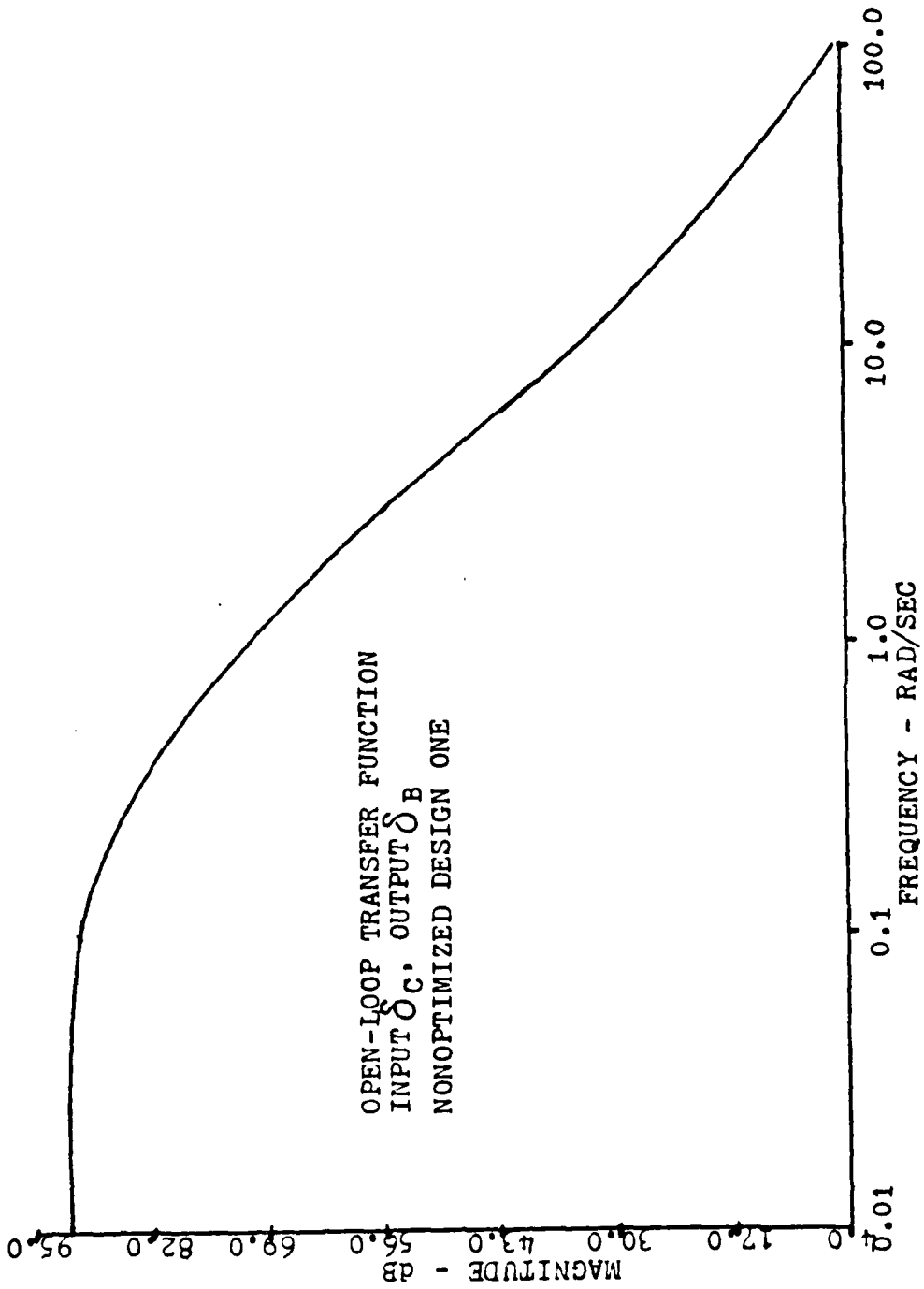
Figure 7.8   Transfer Function $\delta_c - \delta_B$ Design One, Nonoptimized.
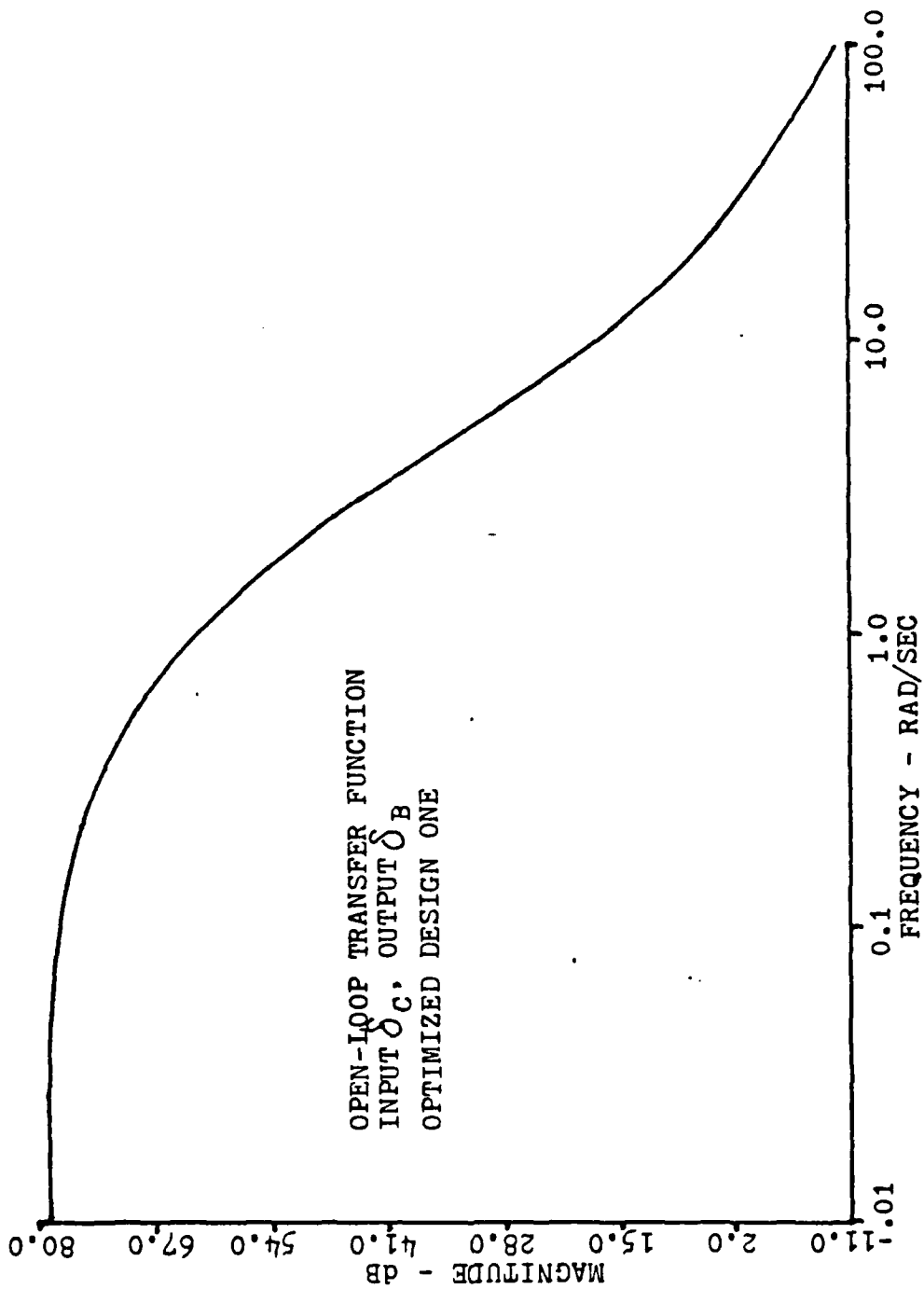
Figure 7.9   Transfer Function $\delta_c - \delta_g$ Design One, Optimized.

97
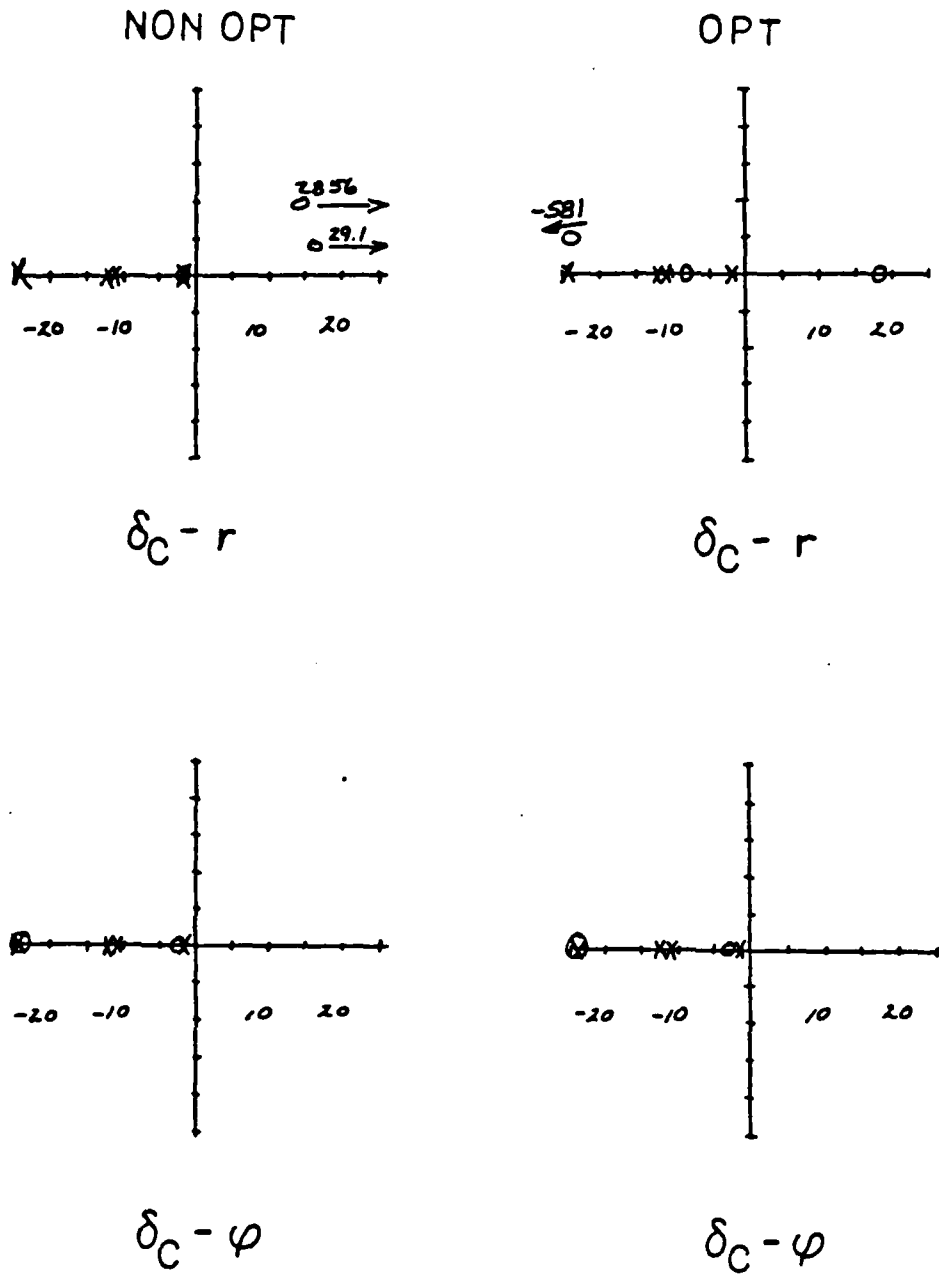
Figure 7.10   Pole-Zero Plots for Design 1.

98

NON OPT

OPT

$\delta_C - V$
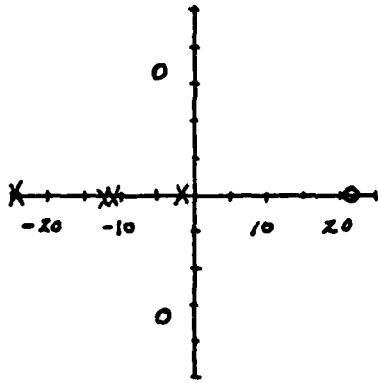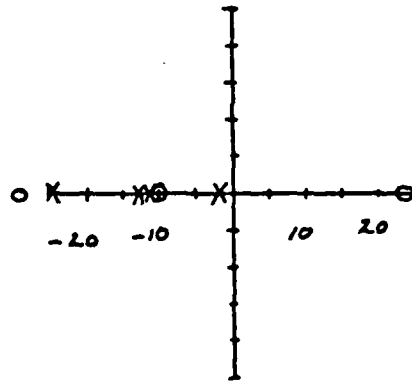
$\delta_C - V$
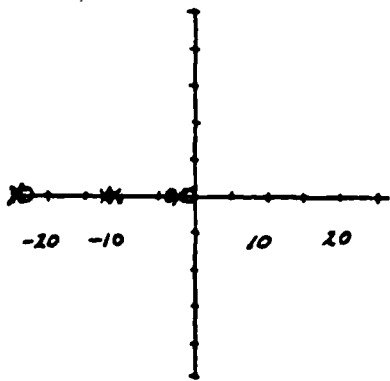
$\delta_C - P$
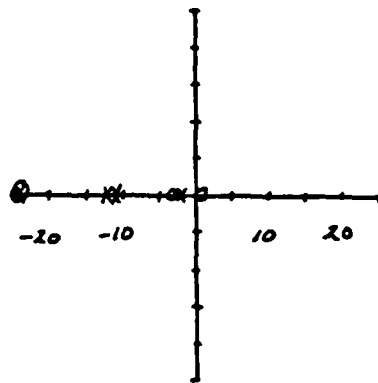
$\delta_C - P$

Figure 7.11   Pole-Zero Plots for Design 1 (cont.).

99

NON OPT   O PT
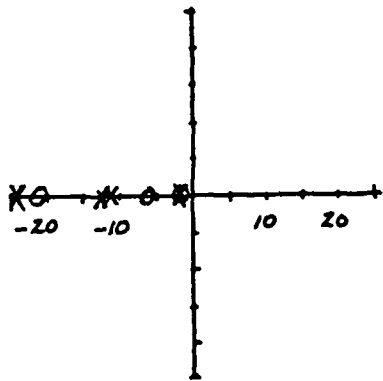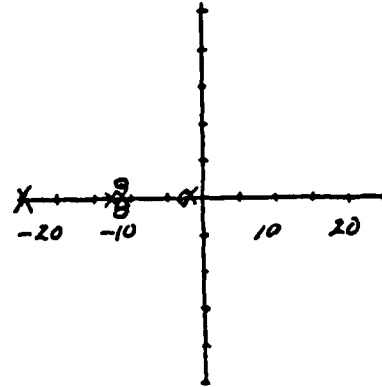
$\delta_B - r$ $\delta_B - r$
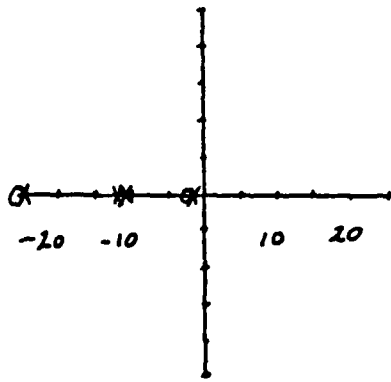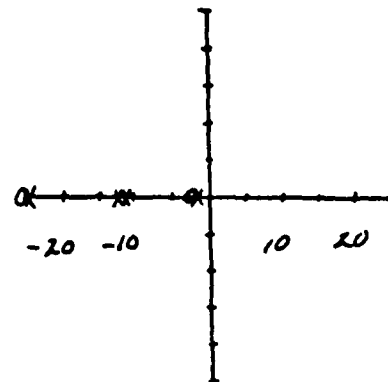
$\delta_B - \varphi$ $\delta_B - \varphi$

Figure 7.12   Pole-Zero Plots for Design 1 (cont.).

100

NON OPT   OPT
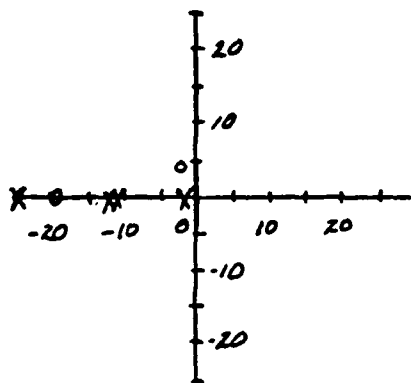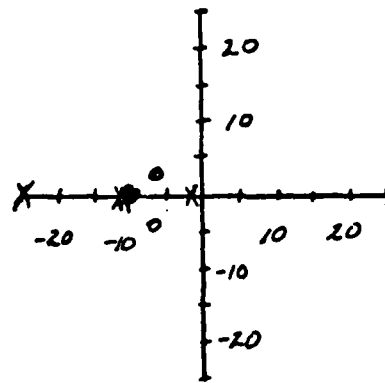
$\delta_B - V$ $\delta_B - V$
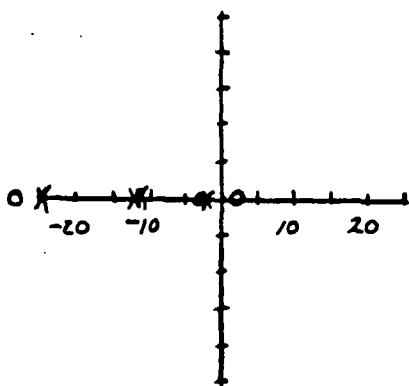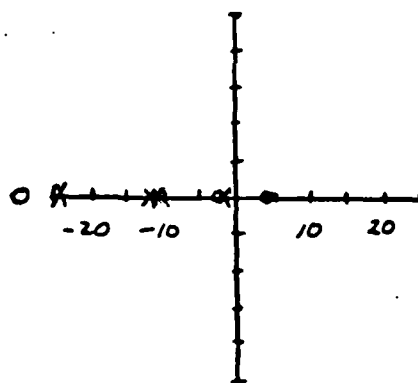
$\delta_B - P$ $\delta_B - P$

Figure 7.13   Pole-Zero Plots for Design 1 (cont.).

101
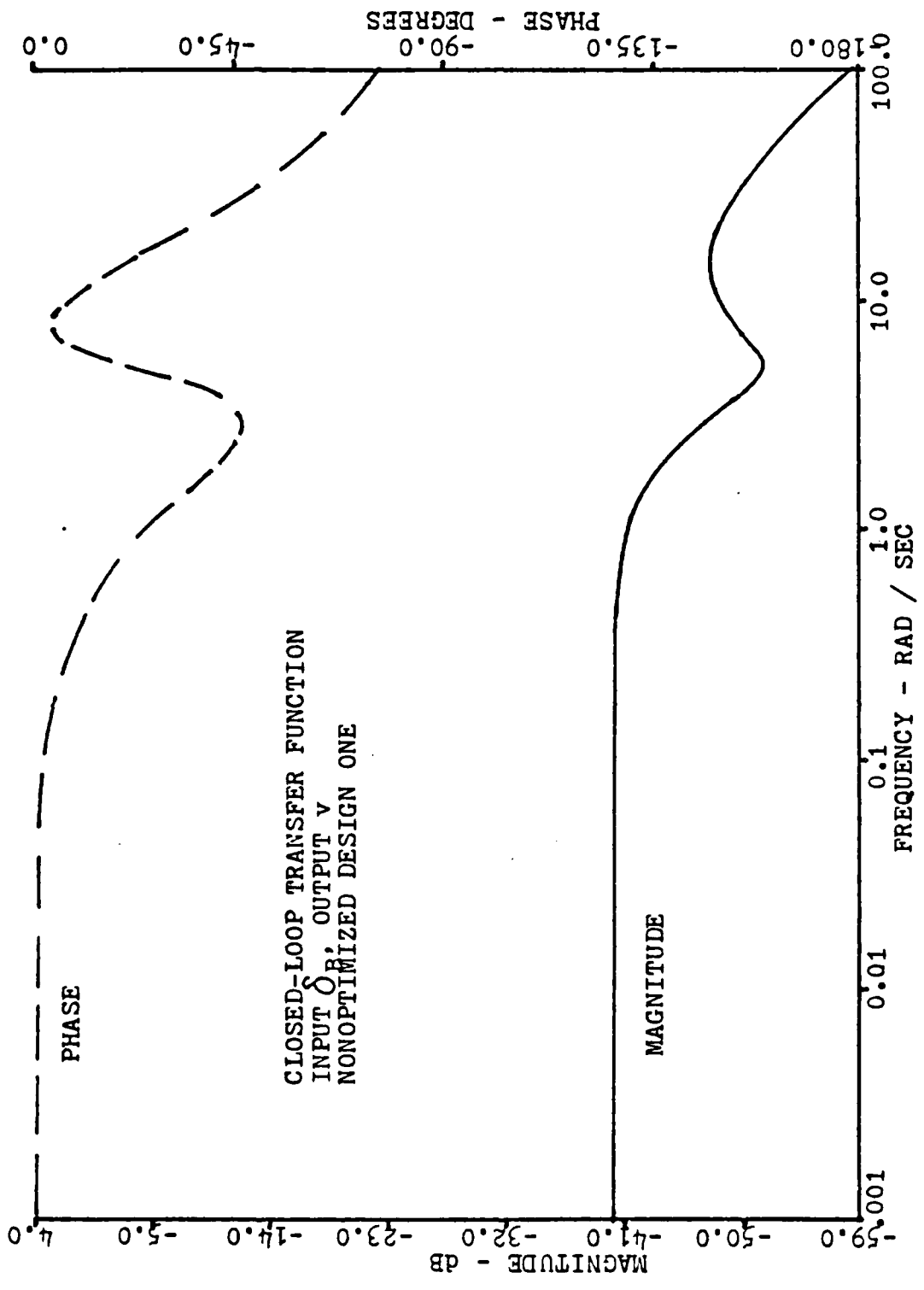
Figure 7.14   The δ_B to v Frequency Response, Nonopt..

102
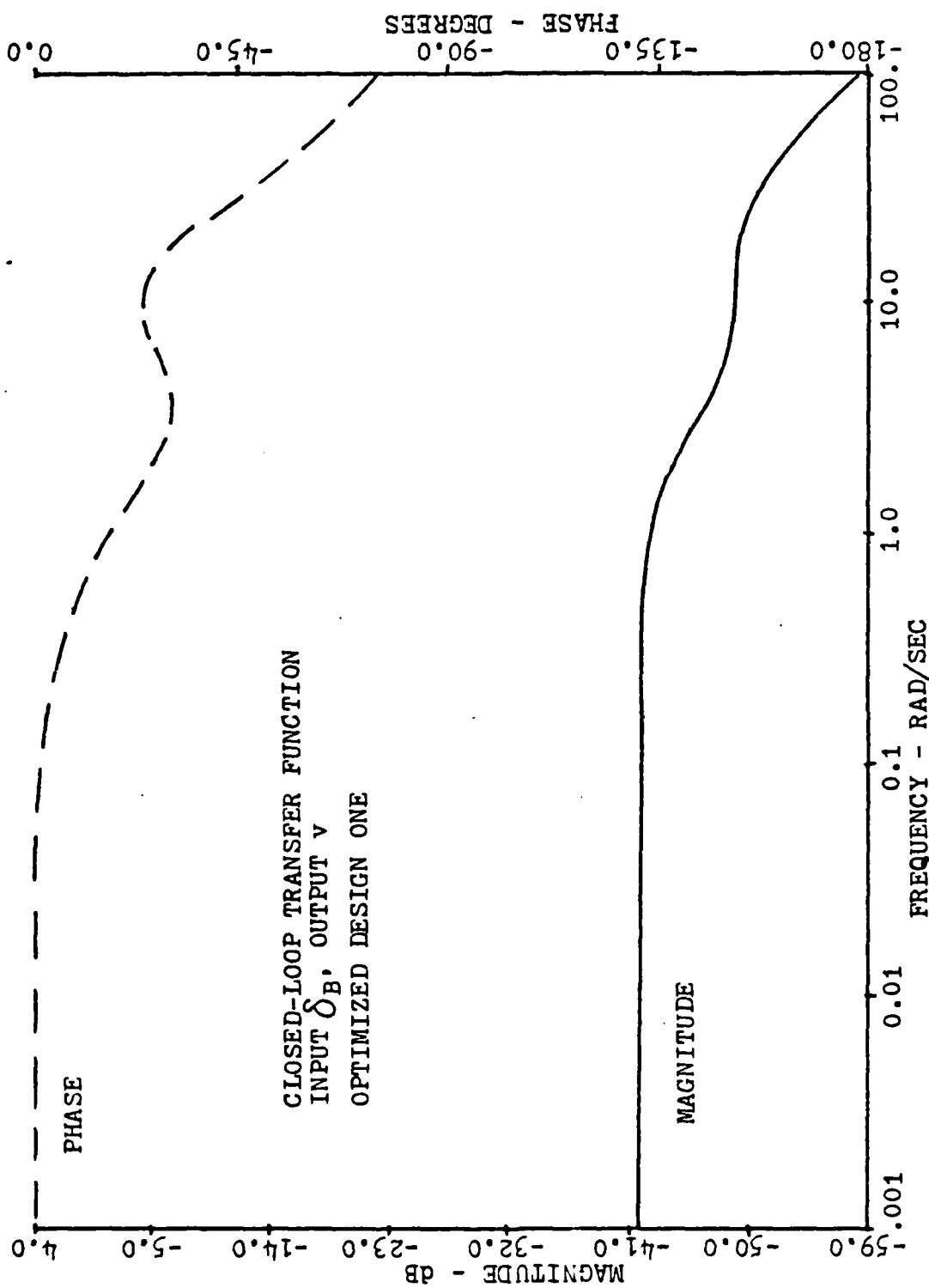
Figure 7.15   The δ_B to v Frequency Response, Optimized.

103

for robustness recovery the zero shift associated with the feedback gain changes is directly related to the overall frequency response of the system.

In design 1 case 2 the minimum singular value level was increased from 0.6 to 1.0. The pole placement and robustness design procedure also gave a better design for case 2

TABLE 4
Helicopter Problem Feedback Gains

DESIGN	GAIN VALUES $\begin{bmatrix} f_{11} & f_{12} & f_{13} & f_{14} \\ f_{21} & f_{22} & f_{23} & f_{24} \end{bmatrix}$		
One Case 1	-14.77726 2.15858	77.96629	-32.91595
	-0.00567 -2.55646	0.39039	-15.04805
One Case 2	-6.36249 -1.53746	72.85013	74.20387
	0.00813 -2.60073	0.65395	-13.47128
Two	-0.96660 -4.54597	0.75436	11.96808
	0.00916 -3.03069	-0.36391	-0.34991

by modifying the feedback gains as shown in table 4. The change in the transfer functions for δ_c to δ_B (figure 7.16 and 7.17) is similar to the change seen in figures 7.8 and 7.9. The gain was reduced from above 90 db to about 75 db and the bandwidth cut from above 100 rad/sec to about 25 rad/sec. The added requirement of increased robustness did not result in a significant change in performance. [Ref. 16].
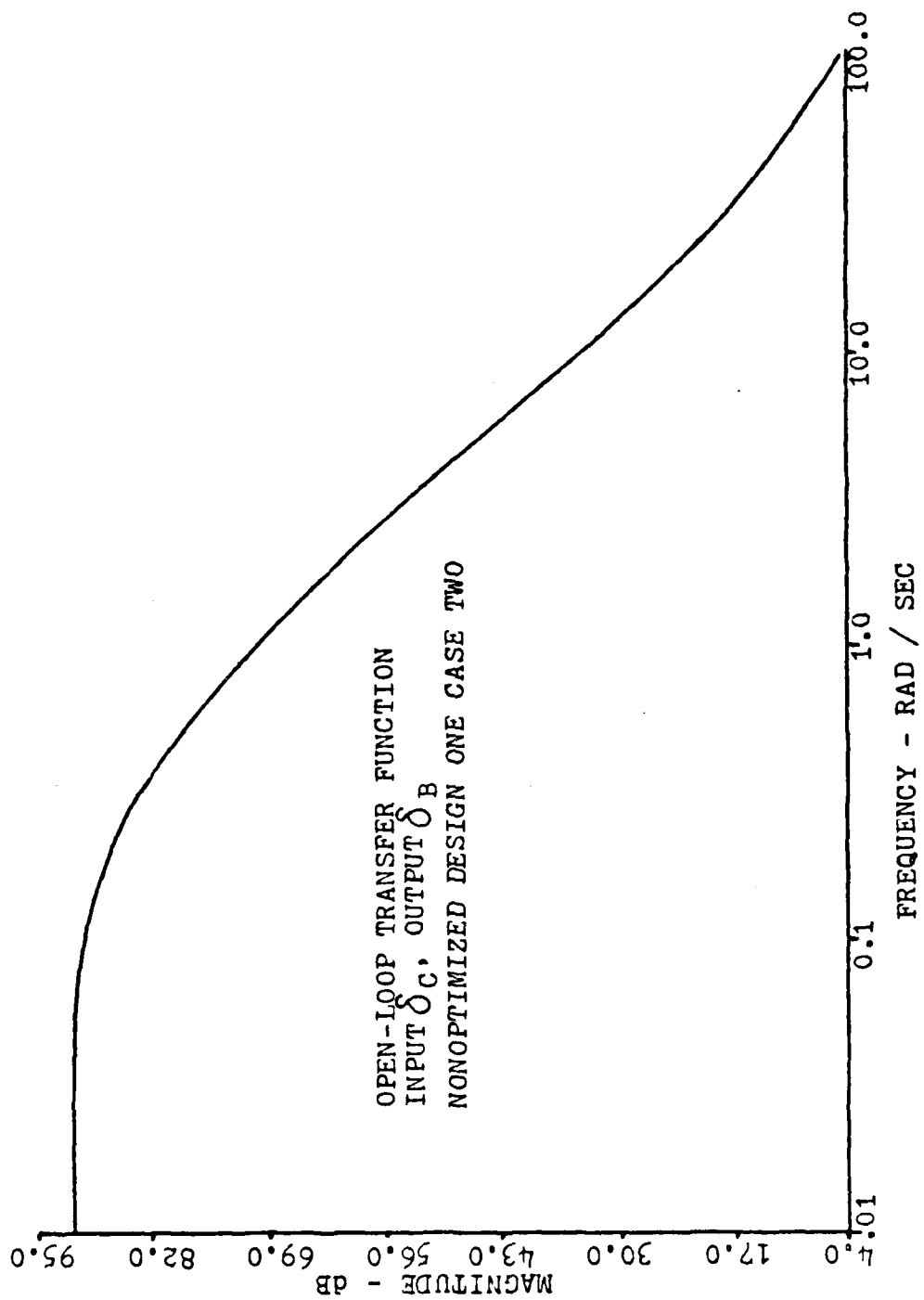
Figure 7.16 Transfer Function $\delta_c - \delta_B$ Design 1 Case2, Nonopt..
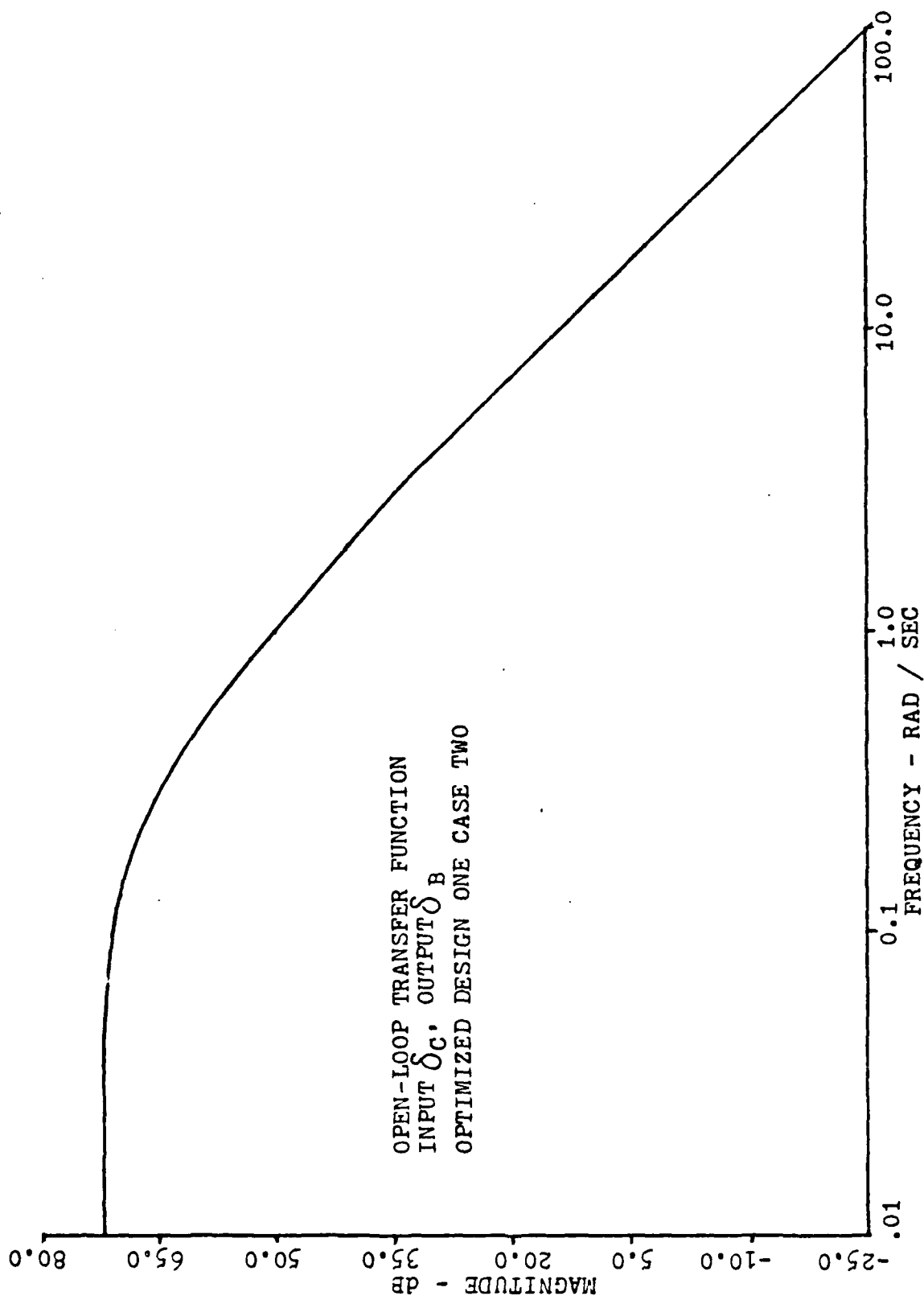
Figure 7.17   Transfer Function $\delta_c - \delta_B$ Design 1 Case 2, Opt.

106

Design two which is the design that is subject to parameter uncertainty in the δ_β to p channel was also studied using the pole placement and robustness design routine. In this design the singular values were very low until frequencies above 100 rad/sec were reached. To improve the design it was assumed that the pole locations that corresponded to [Ref. 16] were the required pole locations and that a robustness singular value level of 0.6 would provide adequate gain and phase margin for the design. The pole placement and robustness routine adjusted the gains in this problem until a minimum singular value of 0.6 was obtained. During this adjustment the gains in channel δ_β were considerably reduced to offset the cross-coupling between the two channels. Again, in this design as before the channel cross-coupling between δ_c and δ_β shows a marked change in bandwidth and gain from above 100 to 35 rad/sec and about 78 db to 22 db respectively. Figures 7.18 and 7.19 show the transfer function plots for this term. Figure 7.20 shows the singular value improvement. In figure 7.21 the time response is plotted. As can be seen in the plot the improvement in robustness for this problem results in very sluggish response and degraded performance.

To summarize for this problem, a given performance level has been chosen in terms of pole locations. The level of robustness has been set for a desired gain and phase margin based on the universal gain and phase margin curve. The pole placement and robustness routine has been able to improve the robustness level by changing the feedback gains that affect the channel δ_c cross-coupling. This robustness recovery is affected by modification of the system feedback gains in such a manner that cross coupling gains are reduced so that small cross-coupling perturbations do not drive the system into instability. The open-loop transfer function plots have been used to indicate how this mechanism operates

107
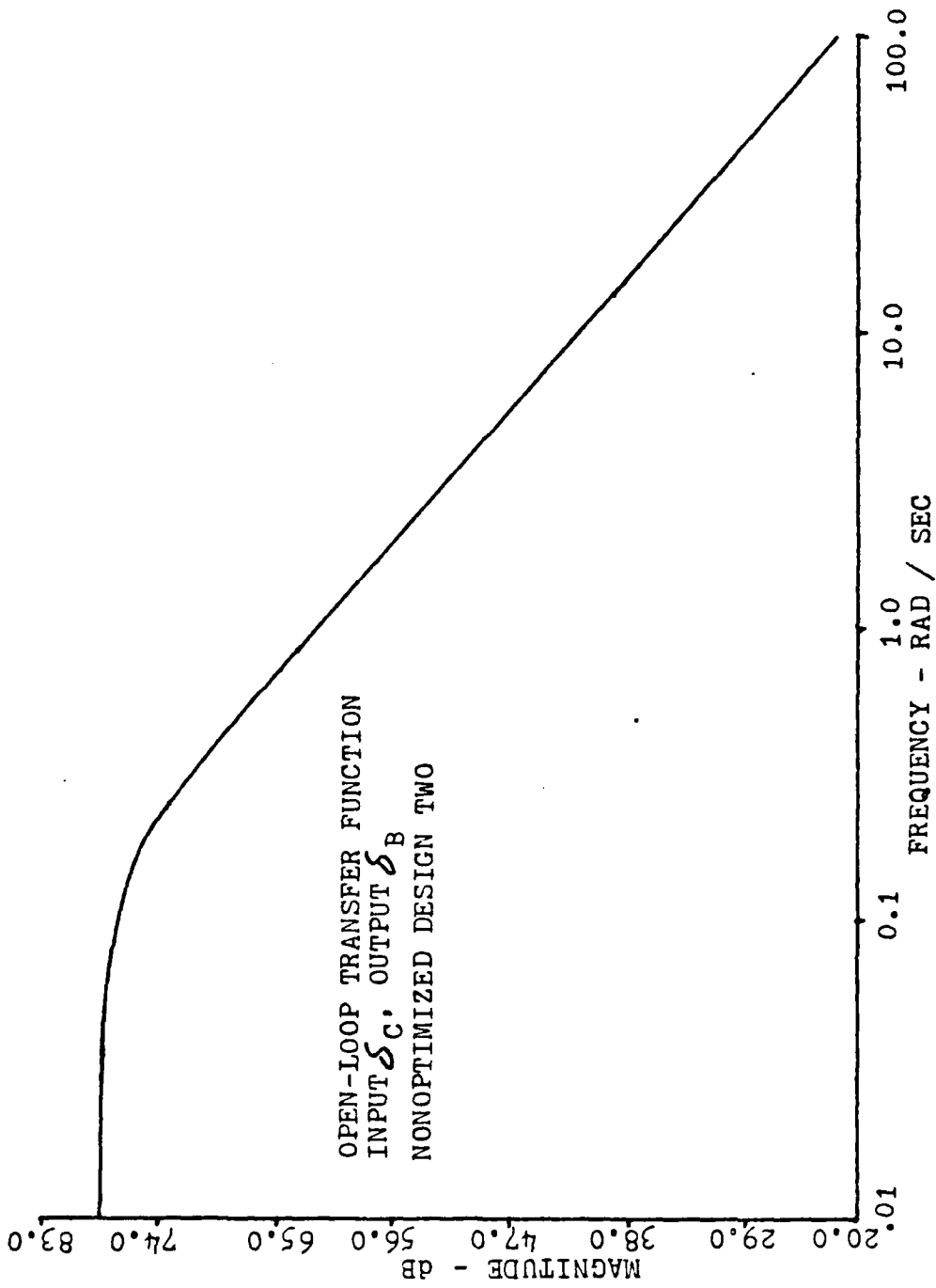
Figure 7.18   Transfer Function $\delta_c - \delta_B$ Design Two, Nonopt..

108
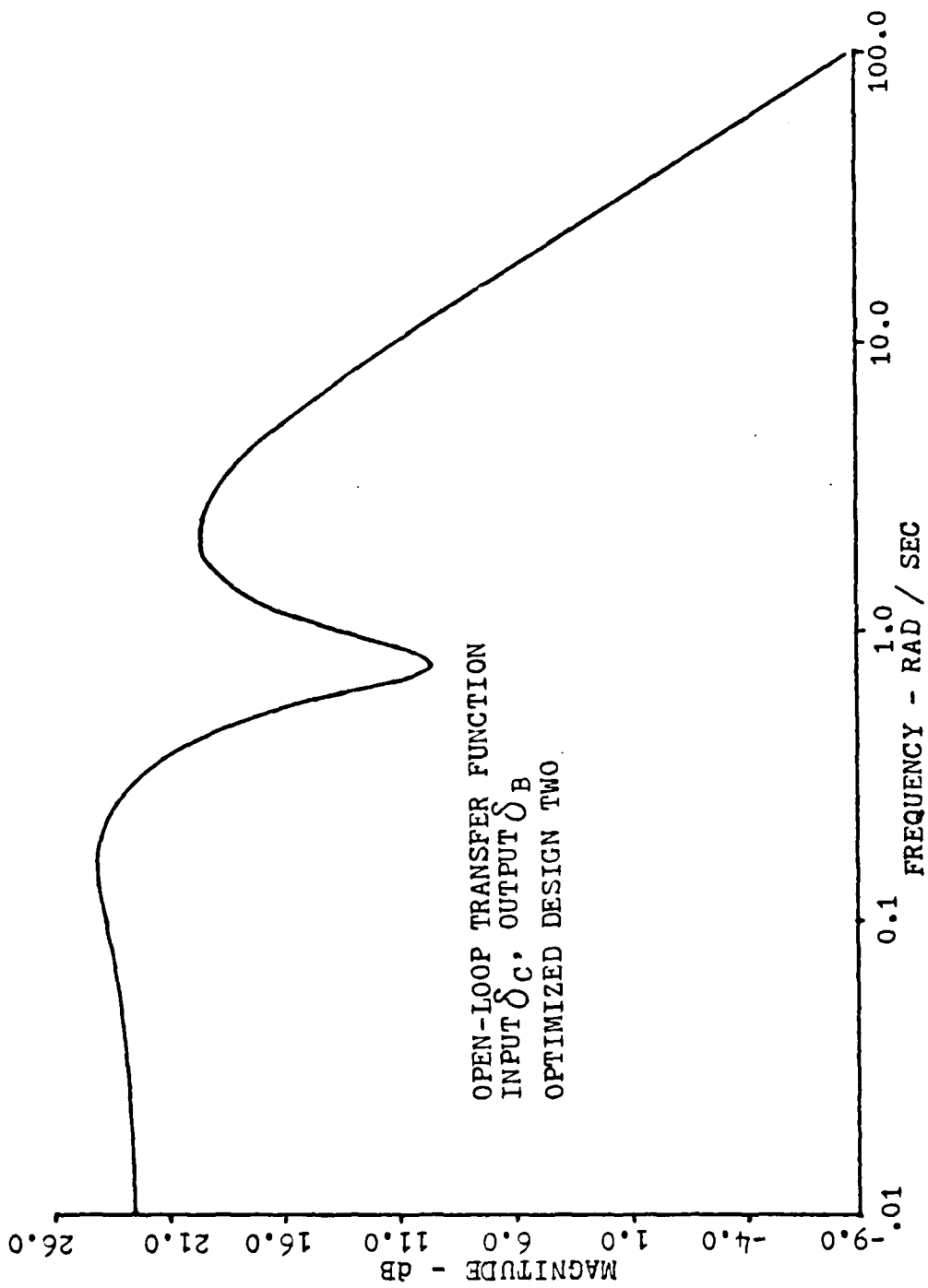
Figure 7.19   Transfer Function $\delta_c - \delta_B$ Design Two, Optimized.

DESIGN TWO OPTIMIZED SINGULAR VALUES
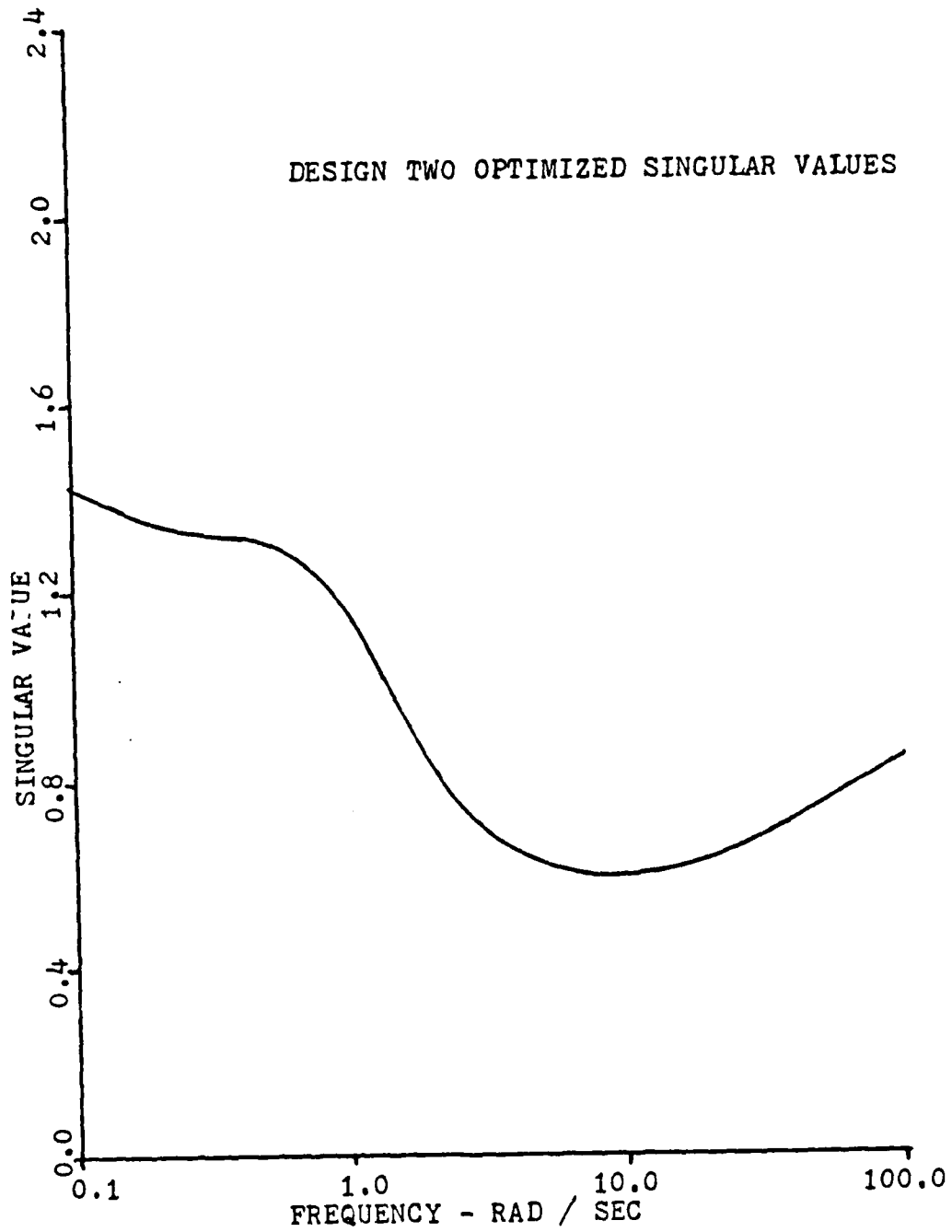
Figure 7.20   Singular Value Plot Design Two.

110
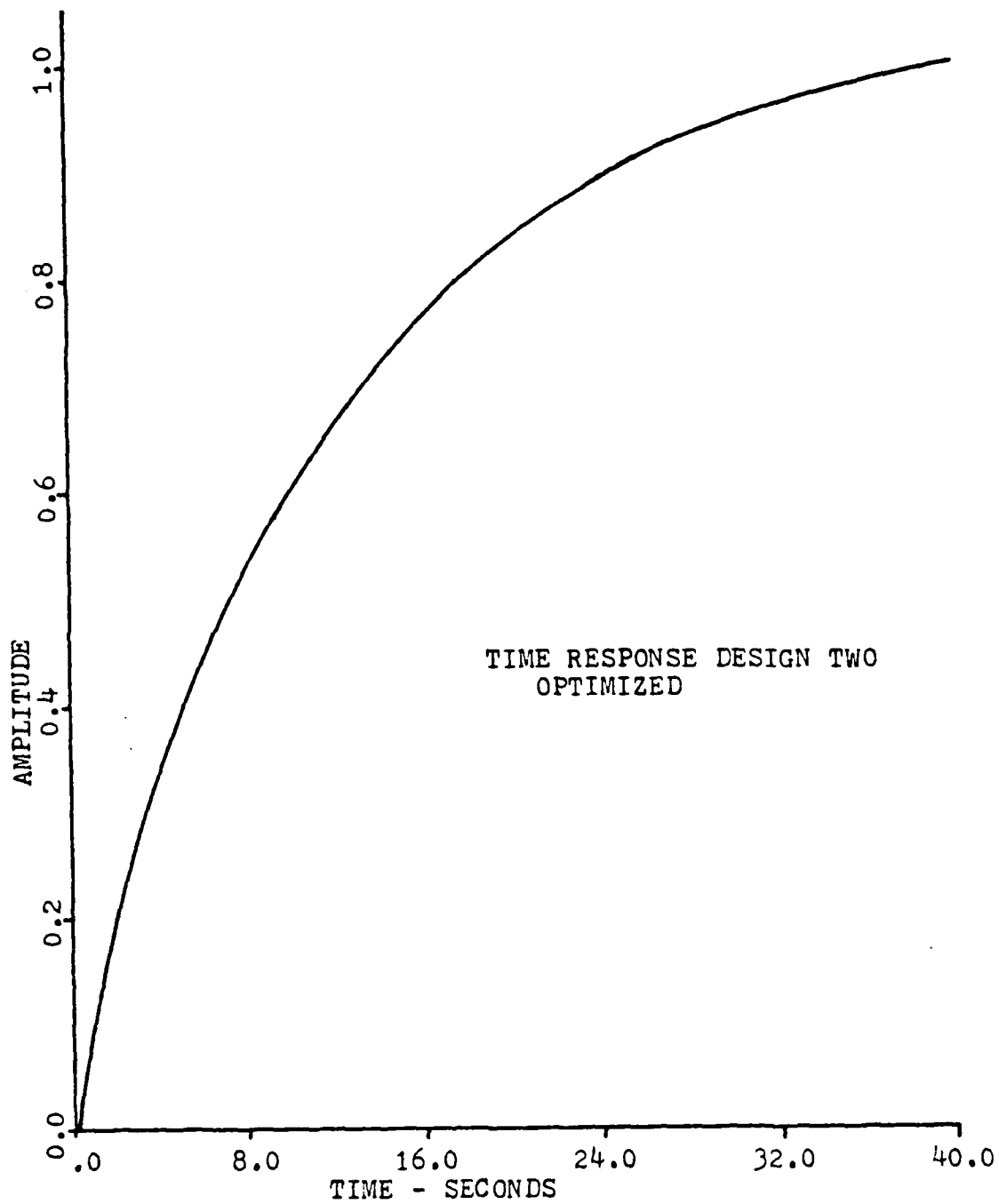
Figure 7.21   Time Response Design Two.

and have been shown to be an alternative indicator of channels that may be affected by cross-feed perturbations. The pole-zero diagrams of the closed-loop transfer functions of the transfer matrix further indicate that zero movement is in a direction that equalizes the gain level of the frequency response curves in the vicinity of the lowest singular values providing a more balanced system response.

VIII. SIMPLE OBSERVER

The pole placement and robustness design procedure can also be used for robustness recovery in observer design.
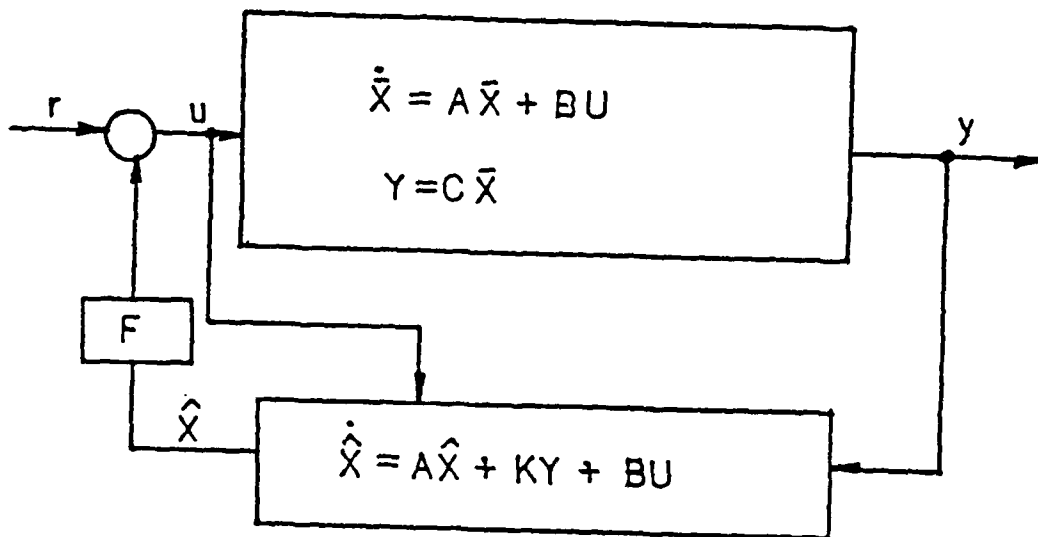
Figure 8.1   Simple Observer.

Given an observer as represented in figure 8.1 it has been shown that the system differential equation may be written as equation 8.1

$$\begin{aligned}
\begin{matrix} \dot{x} \\ \dot{\hat{x}} \end{matrix} &= \begin{bmatrix} A & -BF \\ KC & A_c - BF \end{bmatrix} \begin{bmatrix} x \\ \hat{x} \end{bmatrix} + \begin{bmatrix} B \\ B \end{bmatrix} r \\
A_c &= A - KC
\end{aligned} \tag{8.1}$$

where x is the state and x is the observer variable, and the solution may be separated to independent solutions for the feedback gains and the observer gains. This separation allows the use of the feedback gains to set the pole

113

locations and then a second optimization run with set feed-back gains so that the observer gains may be computed to adjust the system robustness level. The observer pole location could also be placed using the observer gains, K. In the currently implementation of the pole placement and robustness design routine the observer poles are simply restricted to areas cf the left half plane.

In this chapter a simple stable observer system will be analyzed based on [Ref. 17]. Given the system, equation 8.2,

$$\dot{\underline{x}} = \begin{bmatrix} 0 & 1 \\ -3 & -4 \end{bmatrix} \underline{x} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u + \begin{bmatrix} 35 \\ -61 \end{bmatrix} \gamma \qquad (8.2)$$

$$y = \begin{bmatrix} 2 & 1 \end{bmatrix} x + \eta$$

where $E(\gamma) = E(\eta) = 0$ and $E(\gamma(t)\gamma(\tau)) = E(\eta(t)\eta(\tau)) = \delta(t-\tau)$ with the feedback law of equation 8.3

$$u = -(50 \quad 10) x + 50r \qquad (8.3)$$

An analysis has been done to compare results of the numerical optimization procedure with results presented in [Ref. 17]. The results for an optimal regulator design using quadratic cost criteria as stated in equation 8.4 are given in table 5

$$J = \int (x^T H^T H x + u^2) \, dt \qquad (8.4)$$

with $H = 4\sqrt{5} \, (\sqrt{35} \quad 1)$

Figure 8.2 shows a Nyquist plot of the full state regulator, the optimal filter and a fast filter. The full state design had poles at s= -7.0±j 2.0 and feedback gains of 50 and 10. The optimal filter as shown in table 5 had poles of -7.0 ± j 2.0 with gain and phase margins of -6.75 db and ±15 degrees. The optimum filter gains were 30 and -50. Using a faster
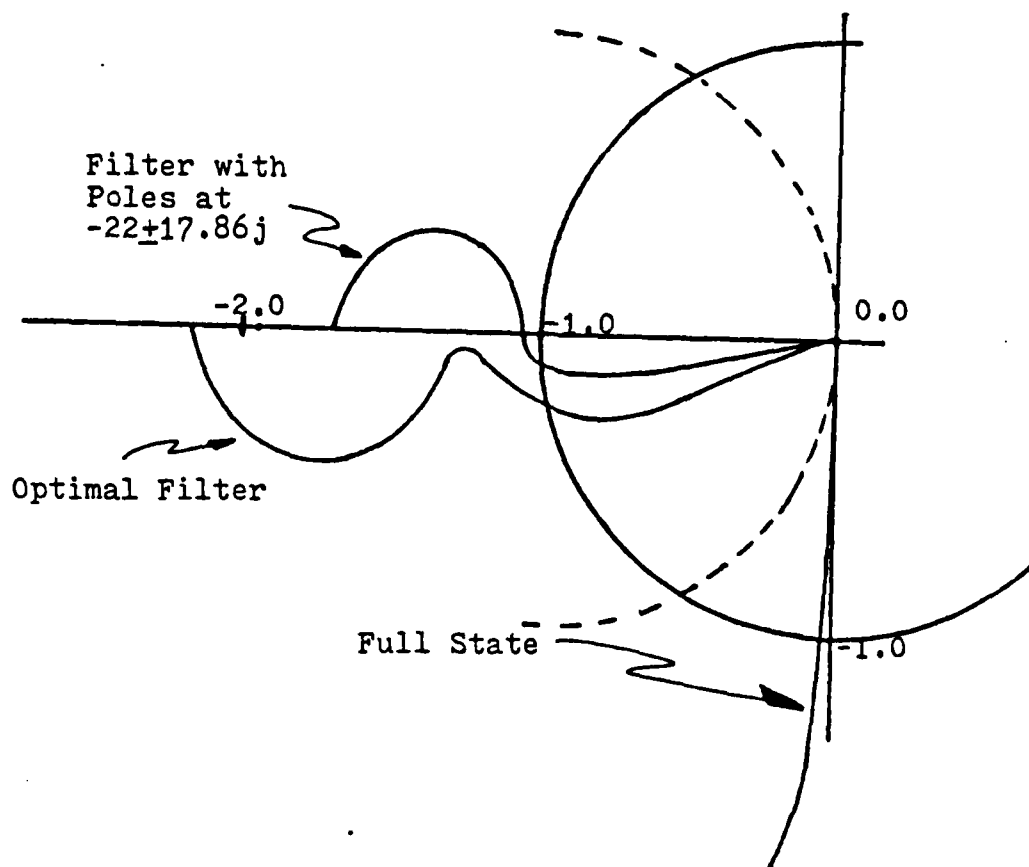
Figure 8.2   Nyquist Plot.

filter also gives poor gain and phase margins.   The gain
margins are on the order of -.98 while the phase margin is
less than 10 degrees.   The bandwidth also increased from 12
to 40 rad/sec. A recovery procedure based on a modification
to the process noise matrix [Ref. 17] may be applied to the
problem.   This procedure can recover a large amount of the
robustness that was lost with the observer addition. Figure
8.3 shows data obtained for several trials of the fictitious
noise procedure.   The gain,   phase margin and other
parameters may be found in table 5.

TABLE 5

Observer Parameter Data

	FILTER POLES	GAIN MARGIN db	PHASE MARGIN deg	ERROR COVARIANCE $E[(x-\hat{x})(x-\hat{x})^T]$		STATE COVARIANCE $E(xx^T)$		FILTER GAIN
Optimal LQG Design	$-7\pm2j$	-6.75	15	27	-163	221	-613	30
				-163	277	-613	2070	-60
Fast Filter Adjustment Procedure	$-22\pm17.9j$	$-.98$	10	6280	-12200	130	-613	720
				-12200	23800	-613	8520	-1400
Fictitious Noise Adjustment Procedure $q^2 = 100$	-4.3	-7.73	19	107	-184	236	-613	26.8
	-13.1			-184	319	-613	1810	-40.2
$q^2 = 500$	-2.9	-10.9	33	163	-301	268	-613	20.4
	-24			-301	564	-613	1500	-17.7
$q^2 = 10^3$	-2.5	-13.9	42	204	-385	285	-613	16.7
	-33			-385	743	-613	1360	-1.9
$q^2 = 10^4$	-2.1	-37	74	290	-570	317	-613	6.9
	-100			-570	1170	-613	1200	84.6

```
POPLAR CASE I        1.00018          GM=-5.5 dB
                    -2.02742          PM=117 deg

POPLAR CASE II      18.22148          GM=-19 dB
                    32.59476          PM=58 deg
```

The singular values of the state feedback and optimal observer systems were computed for comparison with results produced using the pole placement and robustness design recovery procedure. Figure 8.4 indicates a loss in robustness represented by the observer singular values. Lower singular values are less robust. For this single-input single-output observer the Nyquist diagram will be used to define gain and phase margins. Since a singular value of 1.0 is indicative of a linear quadratic level of gain and phase margin, i.e. GM=-6 db, and PM=±60 degrees, this was chosen
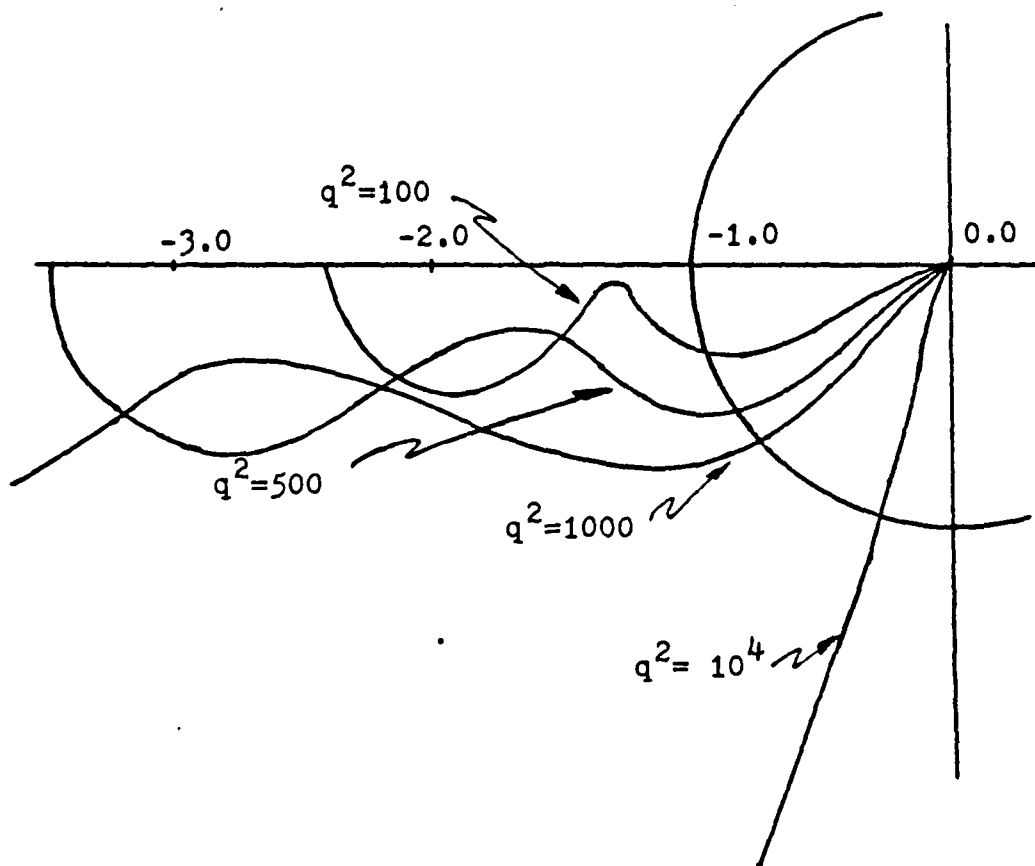
Figure 8.3   Nyquist Plot for Robustness Recovery.

as the design level. The pole placement and robustness
routine was used to recover robustness while setting the
pole placement at the state feedback pole locations of
$-7\pm j2$. The minimum input singular value level was set at
1.0. The pole placement and robustness routine was also set
to place the observer poles anywhere between -100 and -2
that would provide robust design. The plant poles were
placed at -7.05 $\pm j$ 1.82. The feedback gains for this run
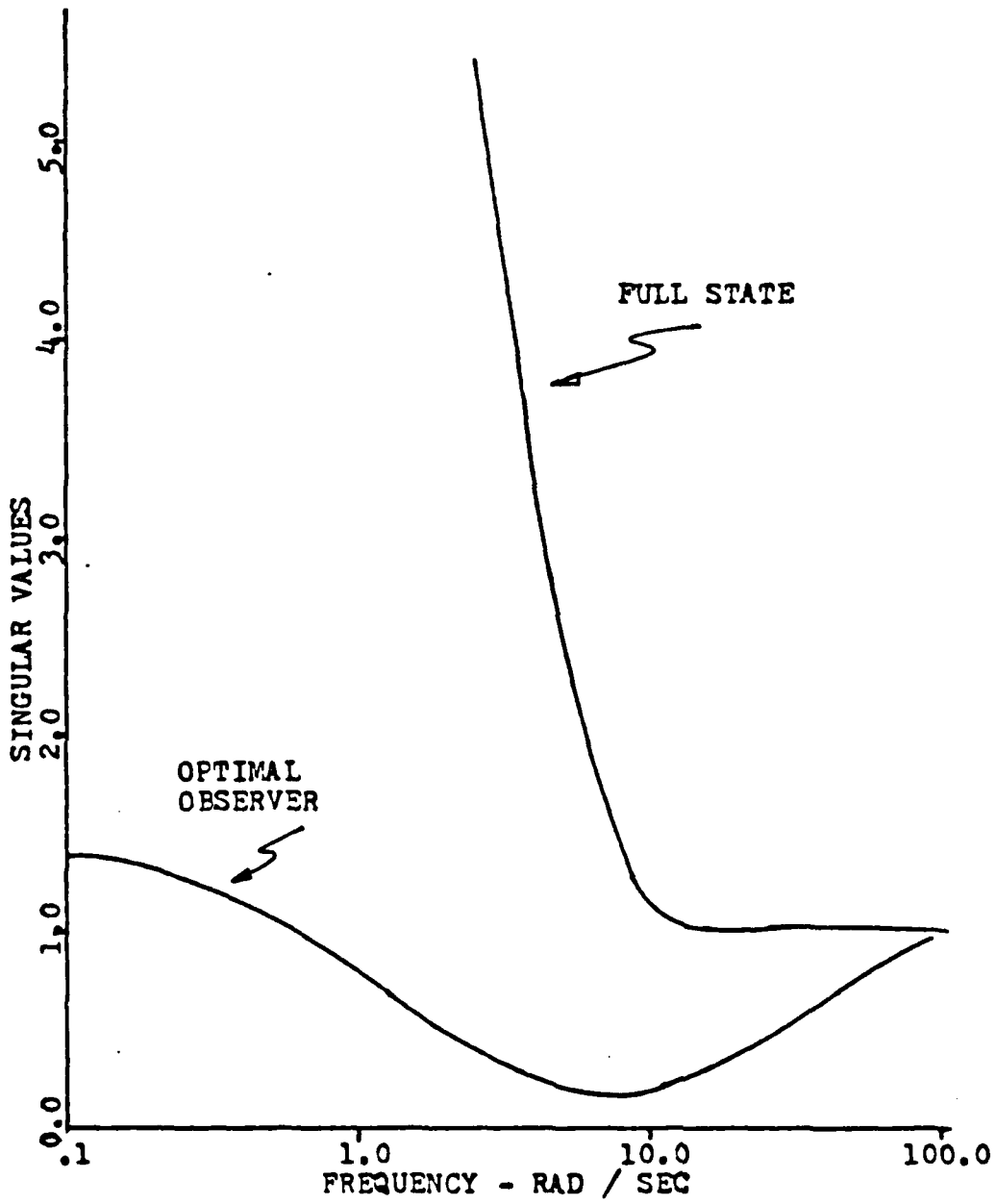
Figure 8.4   Singular Values of Observer System.

118

were 50 and 10.09. The optimizer output produced a filter
with a gains of 1.00018 and -2.02742. The observer pole
locations were -1.99 ± j .001, very near the plant zero
location. A significant change was produced in singular
values. Figure 8.5 shows all singular value curves plotted
together. The optimizer solution for this problem is well
above the optimal filter curve at low frequency. Figure 8.6
shows the Nyquist plot of the optimizer developed design.
The system has a gain margin of -6 db and a phase margin of
117 degrees. The most significant differences between the
two designs being that the observer poles are close to the
plant zero locations and the filter gains are much lower for
the optimizer solution.

Using the OPTSYS program with the pole placement and
robustness routine computed gains as the design parameters
the data for the observer filter was computed. The error
covariance matrix was found to be

306.5   -456.5

-456.5   700.1

These values compare favorably with the trends established
in table 5. The last comparison of the pole placement and
robustness design recovery procedure was an analysis of the
time response curves. Figure 8.7 shows the comparison plot.
The design obtained using the pole placement and robustness
routine did not degrade system performance.

One additional analysis was conducted that set the
desired design parameters slightly differently. In this run
the pole placement and robustness routine was set to place
the observer poles between -10 and -100. The pole placement
and robustness routine was unable to totally satisfy this
requirement. It violated one of the constraints and moved
the larger observer pole to -2.25 which is near the optimum
pole location and also the high q^2 values of the fictitious
noise procedure. The smaller pole was moved out to -70.8

119
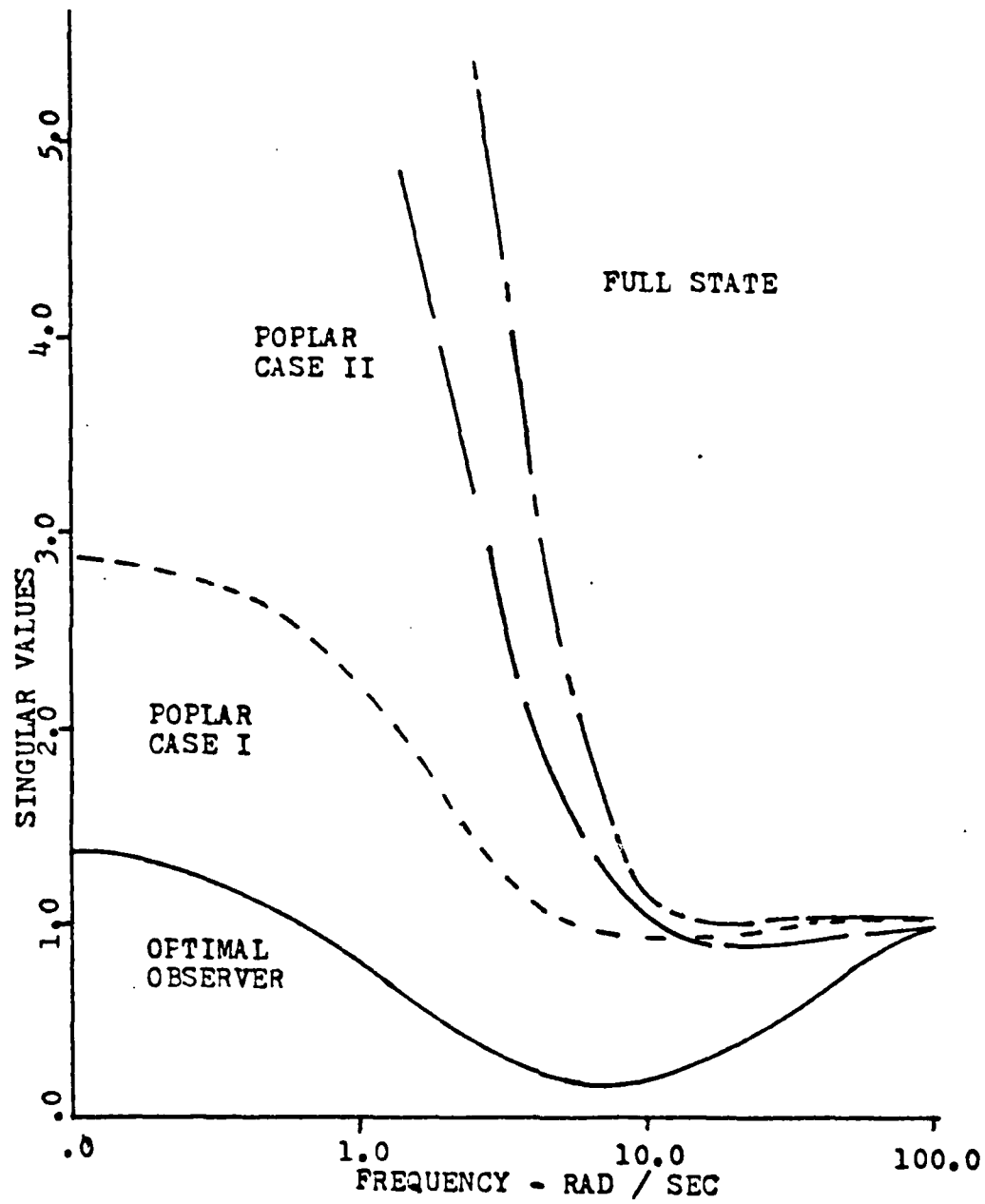
Figure 8.5   Singular Value Comparison Plot.
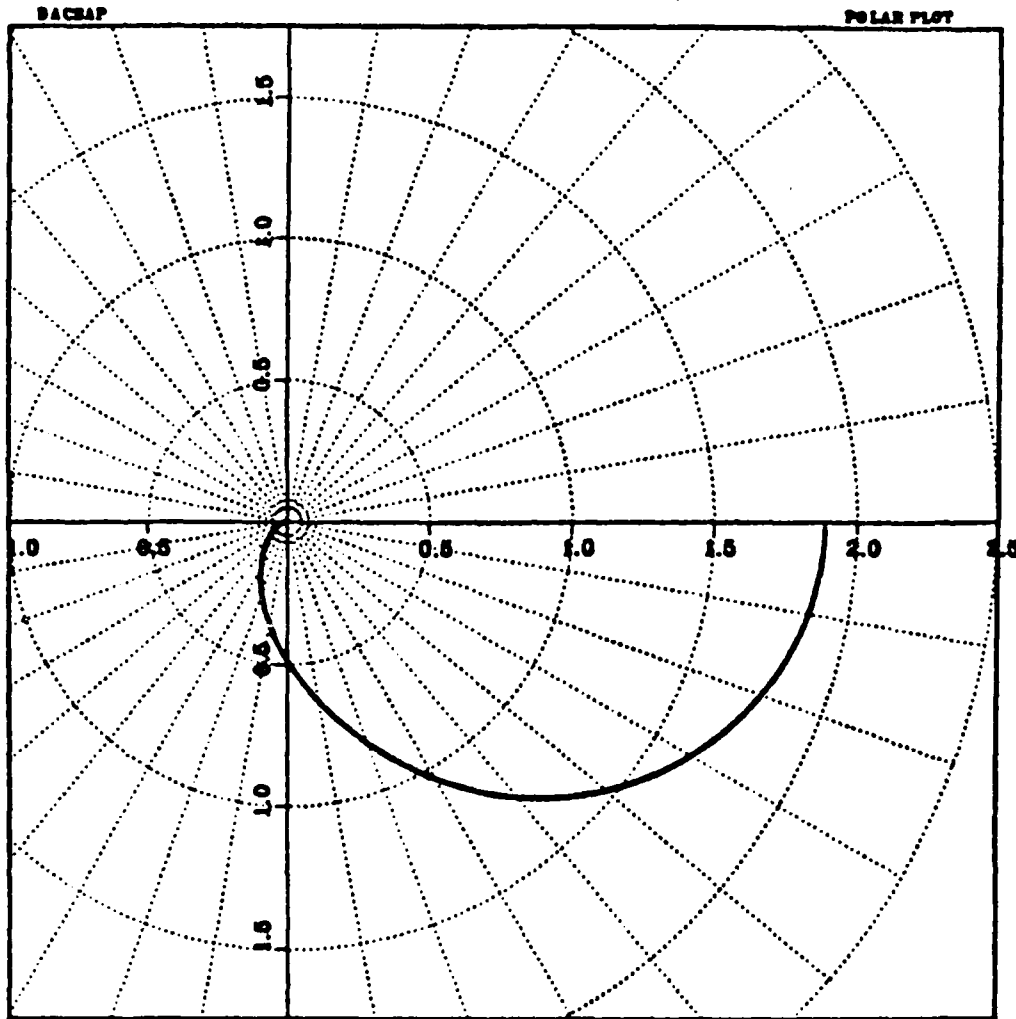
Figure 8.6   Nyquist for Computed Robustness Recovery.

which corresponds to the same level of pole movement found
when q² becomes large. The singular value levels were raised
significantly as shown in figure 8.5.   The state and error
covariance matrices were of the same order as the matrices
found in table 5.   This design has a phase margin cf 58
degrees. The gain margin was about -19 db.   Even though scme
of the ccnstraint conditions on the design were not met the
design demonstrates excellent robustness recovery.

121
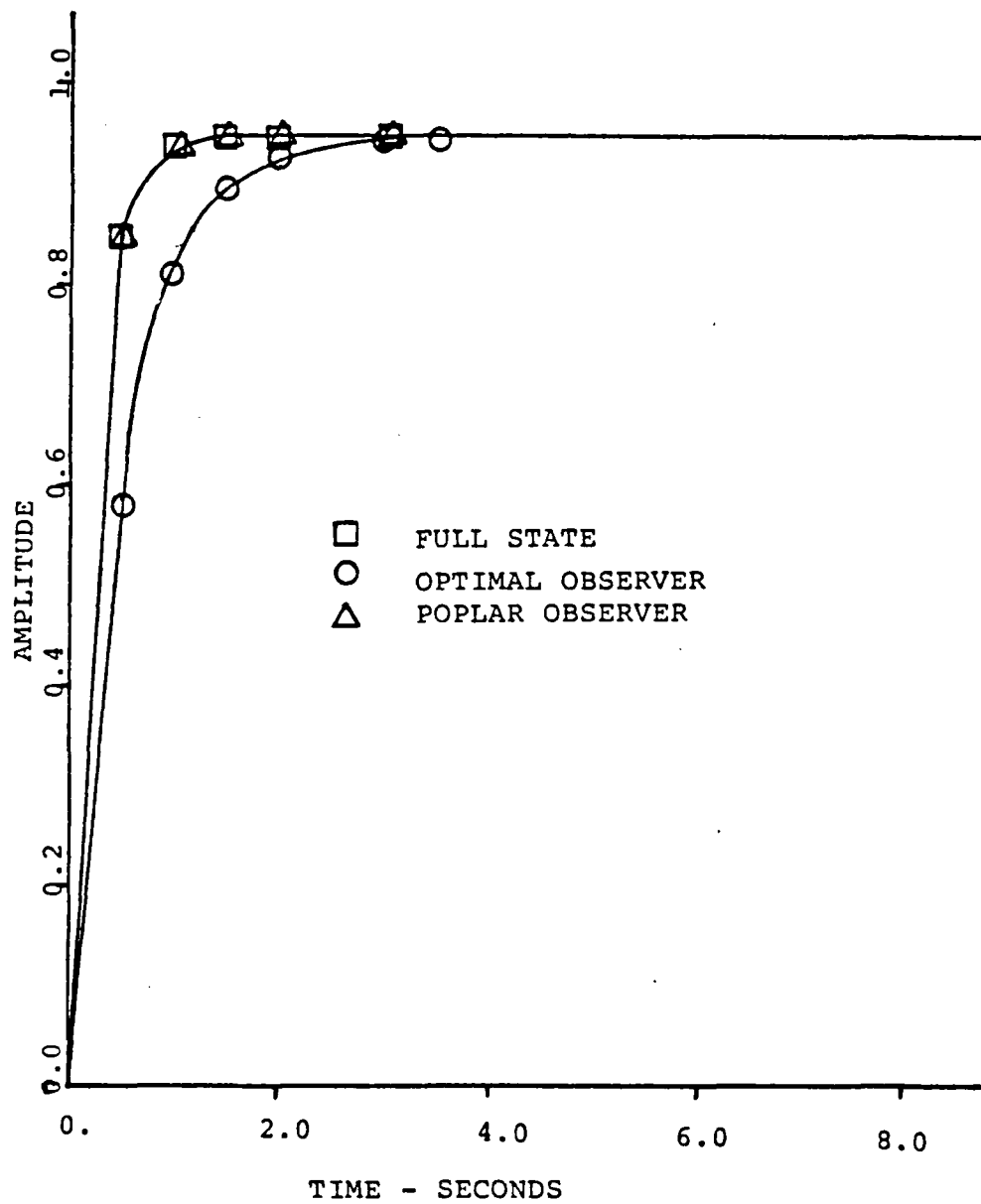
Figure 8.7   Time Response Plot for Simple Observer.

122

The pole placement and robustness design routine has produced a robust design for this observer based system. This design is obtained by using a numerical optimization to directly manipulate the feedback and filter gains. Modification of the LQ functional equation as done in the fictitious noise procedure is not required. The pole placement and robustness routine solution for this problem has lower gains than those found using the fictitious noise adjustment. The routine provides a good, direct methodology for selecting the feedback and filter gains for a robust observer design with excellent performance.

IX. ROBUST OBSERVER DESIGN

This chapter will be devoted to a short discussion of the robustness recovery of a fourth order observer based controller. The problem is a helicopter problem, [Ref. 6]. In this case the helicopter model is that of the longitudinal control loop of a CH-47. The nominal model is taken to be the system of equations 9.1 and 9.2 for an aircraft speed of forty knots.

$$\dot{\underline{x}} = \begin{bmatrix} -0.02 & 0.005 & 2.4 & -32. \\ -0.14 & 0.44 & -1.3 & -30. \\ 0. & 0.018 & -1.6 & 1.2 \\ 0. & 0. & 1. & 0. \end{bmatrix} \underline{x} + \begin{bmatrix} 0.14 & -.12 \\ .36 & -8.6 \\ .35 & .009 \\ 0. & 0. \end{bmatrix} \underline{u} \qquad (9.1)$$

$$\underline{y} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 57.3 \end{bmatrix} \underline{x} \qquad (9.2)$$

In this problem the controller is formulated as in figure 9.1 which leads to an open-loop transfer function of the form of equation 9.3

$$K(s)G(s) = F(sI-A+BF+KC)^{-1}KC(sI-A)^{-1}B \qquad (9.3)$$

The pole placement and robustness recovery procedure was applied to this problem.

First, the standard full state feedback design was carried out using the Naval Postgraduate version of OPTSYS. This design produced excellent singular value output for the return difference as shown in figure 9.2. The lowest singular value being essentially 1, corresponding to a LQ design with -6 db to ∞ gain margin and 60 degrees of phase margin. The time response of the system was good as shown in

124
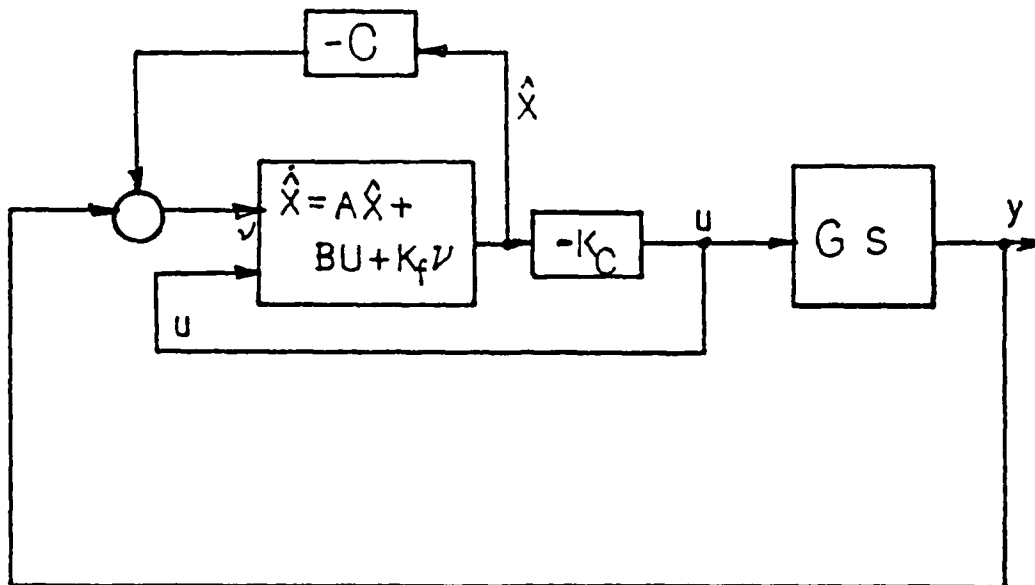
Figure 9.1   Observer Based Controller.

figure 9.3, reaching steady-state in about four seconds with
only a slight overshoot. Since the characteristics for this
system are acceptable no further design iteration was
carried out. The full state feedback became the baseline
design. Assuming that full state feedback was not available
and only two measurements could be produced an observer was
developed to control the two measured outputs, vertical
velocity and pitch attitude. Using the measurement matrix of
equation 9.2, OPTSYS was used to develop an optimal observer
system for this problem. The singular values of the return
difference matrix and the time response were plotted for
comparison to the full state design. These are shown in
figures 9.2 and 9.3 respectively. Note that the singular
value of the return difference matrix is as low as 0.16 at 4
rad/sec. This equates to a gain margin of -1 db to 1 db and
less than 10 degrees phase margin. The time response is
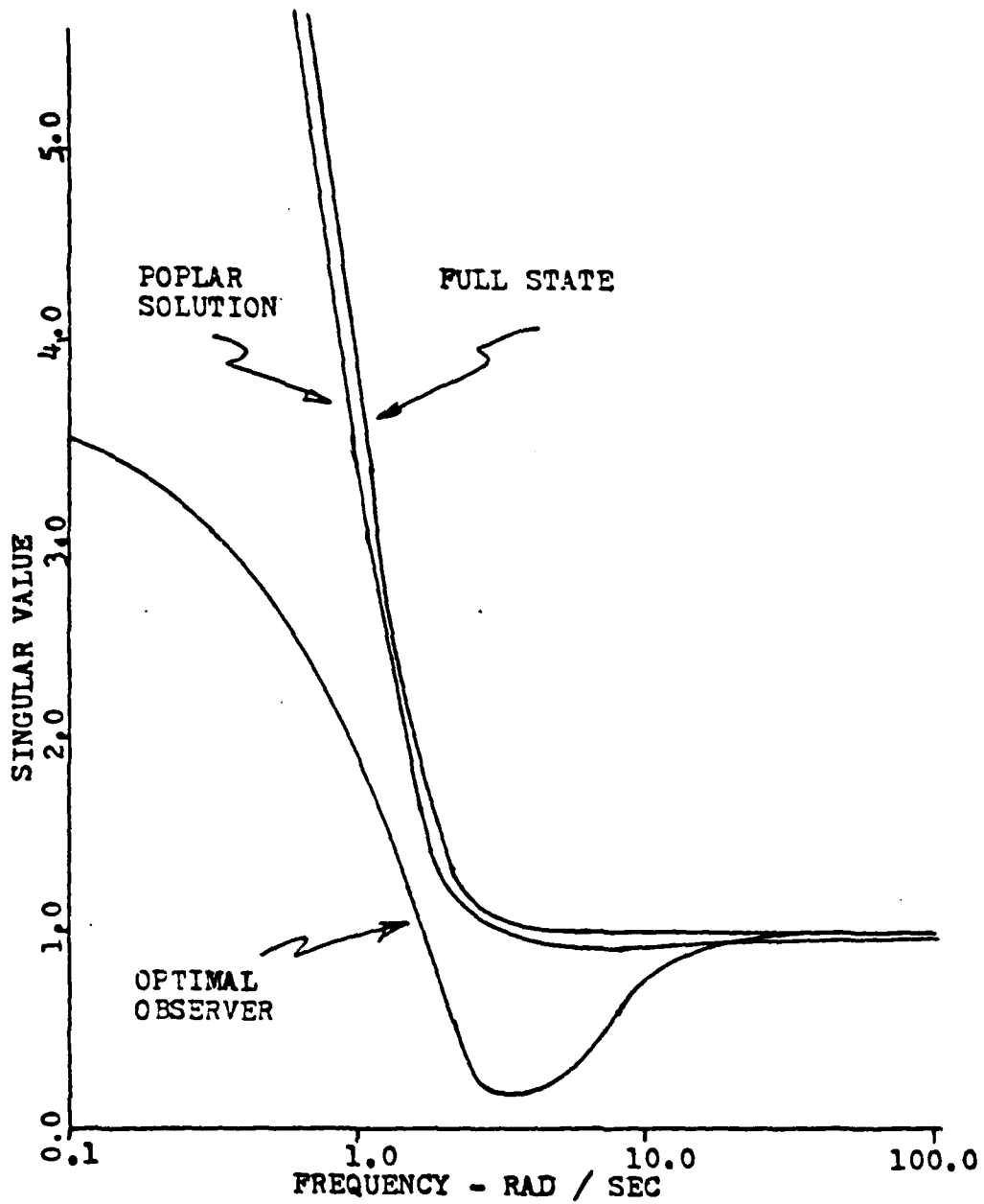
Figure 9.2   Singular Value Plot of Observer Results.

126
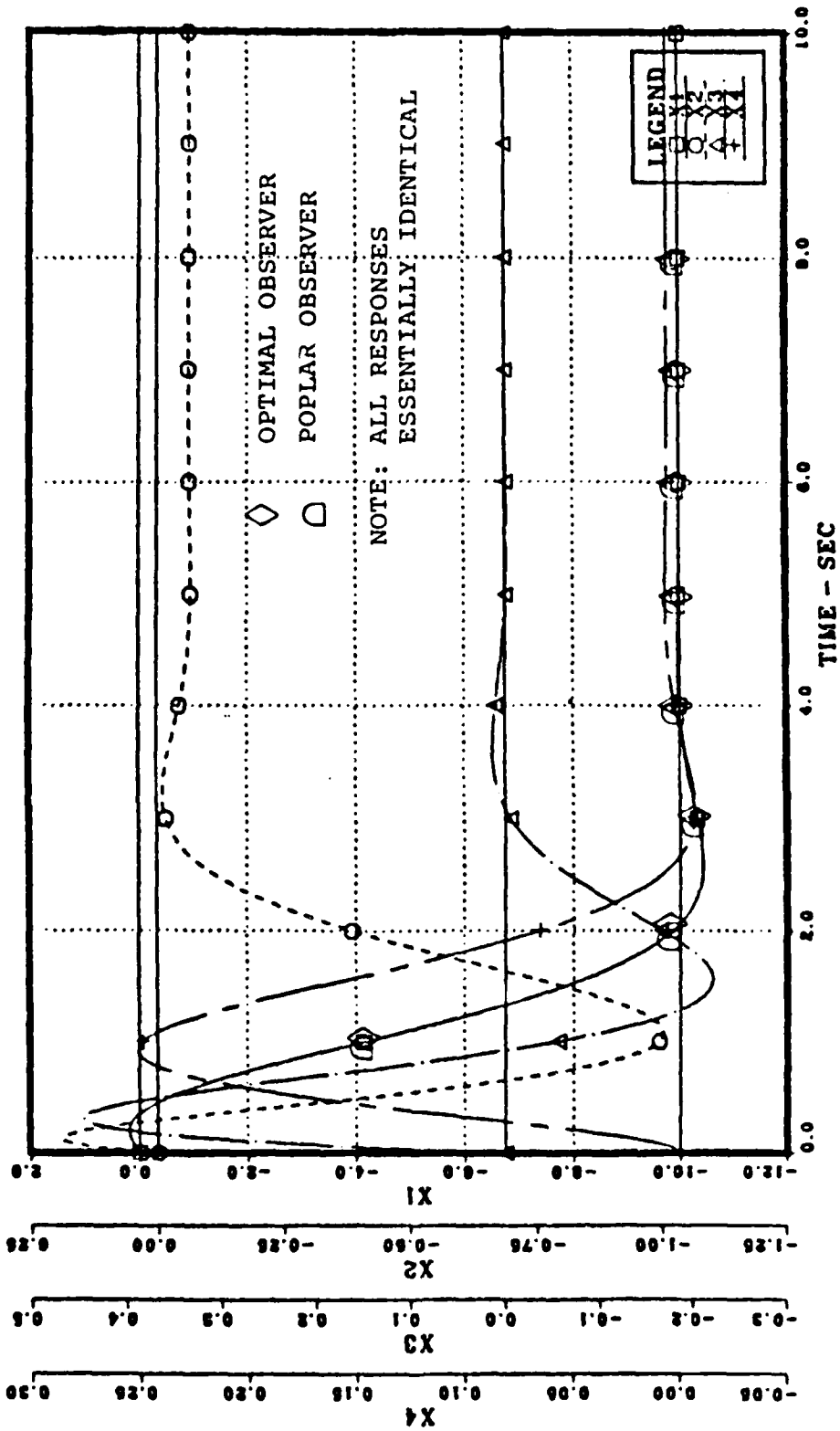
Figure 9.3   Time Response for Observer System.

127

plotted in figure 9.3. The steady-state was reached for the velocity after a slight overshoot in about four seconds.

As the final step in the analysis the pole placement and robustness routine was employed to recover robustness of the observer based system. The pole placement and robustness routine was first used to set the poles at approximately the same location as the poles of the LQ regulator. These were assumed to be the desired pole locations. The pole placement and robustness routine was then used to vary the filter gains, K, until the desired level of robustness was reached. The desired singular value level chosen was 1.0 which corresponds to -6 db to ∞ db gain margin and 60 degrees of phase margin. Figure 9.2 shows that the pole placement and robustness procedure failed to totally recover the robustness level to 1.0. The minimum singular value reached was only 0.936. This singular value equates to a gain margin of -5.0 db to ∞ and a phase margin of 55 degrees. While this is slightly less than the design objective, it is far superior to the optimal observer design discussed in the previous paragraphs. This recovery was made by making an optimization run, finding the filter gains with near zero values and freezing these values at zero. This reduced the number of design variables the pole placement and robustness routine was required to manipulate in a second optimization run and gave a higher robustness solution.

This analysis and the second order observer analysis presented earlier clearly indicate that it is possible to use the pole placement and robustness procedure and separation principle to develop a robustness recovery procedure. The pole placement and robustness routine provides robustness recovery by direct modification of the feedback and filter gains. This procedure requires no modification to LQ cost functionals or other parameters as done in the fictitious noise adjustment method commonly used for robustness

128

recovery.   By providing direct gain adjustment the pole
placement and   robustness procedure results in a practical
design with relatively low observer gains and good perform-
ance. The procedure is simple and straight forward with the
only difficulty being a requirement to sometimes modify
initial starting values or optimizer codes to force the
solution toward the desired point.

X. CONCLUSIONS

An effective method of robustness multivariable control design utilizing a numerical optimization based algorithm has been developed. The pole placement and robustness design routine coupled with the Automated Design Synthesis program provides the designer an excellent tool with which to attack the robust design problem.

The pole placement and robustness design routine has demonstrated the capability of providing designs that solve the problems caused by cross-coupling perturbations which reduce robustness in multivariable systems. This design improvement is accomplished by modifying the system feedback gains in such a manner that the gain in channels that are affected by cross-coupling perturbations is equalized with other system gains to reduce this cross-coupling effect. The gain changes are accompanied by zero shifts which also influence the gain distribution and frequency response of the system.

Perturbation problems in multivariable systems have been shown to be detectable by singular value analysis and by using the Bode magnitude diagram of the open-loop transfer functions of the system. In the open loop transfer function large differentials in Bode gains and bandwidths are indicative of problem areas for cross-coupling perturbations. Robustness is obtained by the pole placement and robustness design program by modifying those gains and bandwidths associated with the cross-coupling perturbations thus reducing the amount of energy coupled from the perturbation into other channels. An associated zero shift has been observed when these gain modifications take place. This zero shift is in the direction of poles that are located in the vicinity

130

cf the frequency of the minimum singular value and tends to equalize the frequency response curve gains in this region.

The use of numerical optimization to recover robustness in observer based designs was demonstrated. The pole placement and robustness routine was applied to problems previously solved using the fictitious noise procedure for robustness recovery. The direct manipulation of feedback and filter gains by the pole placement and robustness routine provided a highly robust design with relatively low filter gains. The problem of robustness recovery in filter-observer designs has been solved in a straight forward and highly practical manner.

```
      ****************************************************   CCNOCC10
      *     PPPPPP  OOOOO  PPPPP  L       AAAAAA  RRRRRR     CCNO0020
      *     P    P  O   O  P   P  L       A    A  R    R     CCN00030
      *     PPPPPP  O   O  PPPPP  L       AAAAAA  RRRRRR     CCN00040
      *     P       O   O  P      L       A    A  R   R      CCN00050
      *     P       OOOOO  P      LLLLLL  A    A  R    R     CCN00070
      *                                                      CCN00080
      *  POLE PLACEMENT AND ROBUSTNESS DESIGN PROGRAM    *   CCN00090
      ****************************************************   CCN00100
      * THIS PROGRAM IS DESIGNED TO EMPLOY SINGULAR VALUE ANALYSIS AND   * CCN00110
      * THE USE OF AN OPTIMIZATION ROUTINE TO AID IN POLE PLACEMENT      * CCN00120
      * CONTROL SYSTEM DESIGN FOR LINEAR MULTIVARIABLE SYSTEMS.          * CCN00130
      * ROBUSTNESS OF THE SYSTEM IS ALSO CONSIDERED BY ESTABLISHING      * CCN00140
      * SINGULAR VALUE LEVELS WHICH CORRESPOND TO MULTILOOP GAIN         * CCN00150
      * AND PHASE MARGINS DETERMINED FROM THE UNIVERSAL GAIN PHASE       * CCN00160
      * DIAGRAM DEVELOPED BY NEWSOM AND MUKHAPADHYAY AT NASA LANGLEY.    * CCN00170
      *                 VERNON C GORDON , CDR , LSN                      * CCN00180
      *                 VERSION 1.1  8-10-84                             * CCN00190
      ****************************************************   CCN00200
      * THIS FIRST BLOCK SETS DIMENSIONS AND DECLARES REAL AND          * CCN00210
      * COMPLEX FUNCTIONS. THE FOLLOWING IS A LIST OF PROGRAM SYMBOLS:  * CCN00220
      ****************************************************   CCN00230
      *                                                      CCN00240
      *   THE SYSTEM UNDER CONSIDERATION IS GIVEN AS:        CCN00250
      *   XDOT = A*X + B*U ; Y=C*X , FOR STATE OR            CCN00260
      *   OUTPUTFEEDBACK OF THE FORM  U=-FX +R OR U=-FY + R  CCN00270
      *                                                      CCN00280
      *   A  - THE PLANT A MATRIX (NSTATE X NSTATE)          CCN00290
      *   B  - THE CONTROL MATRIX (NSTATE X NCONTROL)        CCN00300
      *   C  - THE PLANT OBSERV. OR OUTPUT MATRIX (NOUTPUT X NSTATE;     CCN00310
      *             FOR STATE FEEDBACK PROBLEMS)             CCN00320
      *   F  - THE FEEDBACK GAIN MATRIX(NCONTROL X NSTATE;   CCN00330
      *             FOR STATE FEEDBACK PROBLEMS)             CCN00340
      *                                                      CCN00350
      * REALMU - THE INPUT DESIRED REAL POLE LOCATION        CCN00360
      * IMAGMU - THE INPUT DESIRED IMAGINARY POLE LOCATION   CCN00370
      * FC  -  F*C                                           CCN00390
      * BFC -  B*F*C                                         CCN00400
      * AMBFC - A - B*F*C                                    CCN00410
      * REIG - REAL COMPUTED EIGENVALUE OF THE SYSTEM        CCN00430
      * WW  - PROGRAM WORK SPACE                             CCN00440
      * WK  - WORK SPACE USED BY SUBROUTINES                 CCN00450
      * WA  - DIMENSIONED WORK SPACE                         CCN00460
      * OMEGA - FREQUENCY                                    CCN00470
      * SV  - SINGULAR VALUE OF (I + F * PLANT TRANSFER FUNCTION(G) )    CCN00480
```

132

```
SVMI  -  SINGULAR VALUE OF (I + INV(F* G ))                              CCNOC490
SVAC  -  SINGULAR VALUE OF ( I + G* F)                                   CCNOC500
SVFC  -  SINGULAR VALUE OF ( I + INV (G * F))                            CCNOC510
              ONLY OF INTEREST IF NO. OUTPUTS = NO. INPUTS.              CCNOC520
SIGAM1 - FIRST SINGULAR VALUE OF (I + F*G)                               CCNOC530
SIGAM2 - SECOND SINGULAR VALUE OF (I + F*G)                              CCNOC540
SIGAMX - MAXIMUM SINGULAR VALUE OF (I + F*G)                             CCNOC550
SIGFRO - SQRT(FIRST SING VAL * SECOND SING VAL)                          CCNOC560
SVACMO - MIN ADDITIVE OUTPUT SINGULAR VALUE                              CCNOC570
SVACXO - MAX ADDITIVE OUTPUT SINGULAR VALUE                              CCNOC580
SVMIM  - MIN MULTIPLICATIVE INPUT SINGULAR VALUE (I+INV(FG))             CCNOC590
SVMIX  - MAX MULTIPLICATIVE INPUT SINGULAR VALUE (I+INV(FG))             CCNOC600
SVMFC  - MIN MULTIPLICATIVE OUTPLT SINGULAR VALUE (I+INV(GF))            CCNOC610
SVMXC  - MAX MULTIPLICATIVE OUTPLT SINGULAR VALUE                        CCNOC620
**************************************************************           CCNOC630
REAL*4 A(10,10),B(10,10),C(10,10),F(10,10),REALMU(50),IMAGMU(50),       CCNOC640
1FC(10,10),BFC(10,10),AMBFC(10,10),WK(58C0),                            CCNOC650
2REIG(50),SV(10),WW(10,10),WA(100),                                     CCNOC660
3OMEGA(50),SIGNMI(500),SIGNM2(500),SIGNMX(500),SIGPRU(500),             CCNOC670
4SVACMO(500),SVACXO(500),SVMIM(500),SVMI(50),                           CCNOC680
5SVMMO(500),SVMMJ(500),SVMC(10),SVAC(10),SVMC(10)                       CCNOC690
**************************************************************           CCNOC700
THE COMPLEX PARAMETERS USED IN THE PROGRAM                              CCNOC710
AX  - COMPLEX A                                                         CCNOC720
BX  - COMPLEX B                                                         CCNOC730
CX  - COMPLEX C                                                         CCNOC740
FX  - COMPLEX F                                                         CCNOC750
EIG - EIGENVALUE OF A - B*F*C                                           CCNOC760
Z   - INPUT DESIRED POLE LOCATION                                       CCNOC770
MU  - EIGENVALUE OF A - B*F*C                                           CCNOC780
OMU - ORDERED VALUE OF MU                                               CCNOC790
CEIG - ORDERED VALUE OF EIG                                             CCNOC800
CZ  - ORDERED EIGENVECTOR                                               CCNOC810
XXI - IDENTITY MATRIX                                                   CCNOC820
LI  - INPUT ADDITIVE SINGULAR VECTOR ASSOCIATED WITH SV                 CCNOC830
VI  - INPUT ADDITIVE SINGULAR VECTOR ASSOCIATED WITH SV                 CCNOC840
UG  - OUTPUT ACCITIVE SINGULAR VECTOR (SVAO)                            CCNOC850
UMI - INPUT ACCITIVE SINGULAR VECTOR (SVAO)                             CCNOC860
VMI - INPUT MULTIPLICATIVE SINGULAR VECTOR                              CCNOC870
UMO - OUTPUT MULTIPLICATIVE SINGULAR VECTOR                             CCNOC880
VMG - OUTPUT MULTIPLICATIVE SINGULAR VECTOR                             CCNOC890
FXPLT - F * PLANT TRANSFER FUNCTION (G)                                 CCNOC900
IPFXP - I + F*G                                                         CCNOC910
PLAN  - PLANT TRANSFER FUNCTION (G)                                     CCNOC920
                                                                        CCNOC930
                                                                        CCNOC940
                                                                        CCNOC950
                                                                        CCNOC960
```

135

```
      COMPLEX FUNCTION  PLTFX - G*F                                      CCNO0670
                        XIPPFX - I + G*F                                 CCNO0680
                        PLTFXI - INV(F*G)                               CCNO0690
                        FXPLTI - INV(G*F)                               CCNO1000
                        XIFFLI - I + INV (F*G)                          CCNO1010
                        XIPLFI - I + INV (F*F)                          CCNO1020
C *********************************************************************  CCNO1030
      COMPLEX *8 EIG(50),ZEIG(10,10),MU(50),OPU(50),QEIG(50),GZ(10,10),AX(10 CCNO1040
     1 10),BX(10,10),CX(10,10),FX(10,10),FXPLTI(10,10),XI(10,10),XIPF,XP(1 CCNO1050
     2 20),UMI(10,10),VMI(10,10),PLAN(10,10),XXI(10,10),UO(10,10),VO(10,1 CCNO1060
     3 30),PLTFXI(10,10),VMI(10,10),UMO(10,10),VMO(10,10),PLTFX(10,10),XIPPF CCNO1070
     4 (10,10),PLTFXI(10,10),FXPLTI(10,10),XIFFLI(10,10),XIPLFI(10,10)    CCNO1080
C *********************************************************************  CCNO1090
C     SET UP COMPLEX WORKING PARAMETERS                                  CCNO1100
      COMPLEX ZN,TEMP1,TEMP2                                             CCNO1110
C *********************************************************************  CCNO1120
C     SET UP DIMENSIONS FOR OPTIMIZER PARAMETERS AND OTHER FUNCTIONS     CCNO1130
C *********************************************************************  CCNO1140
C                                                                        CCNO1150
C     X   - THE DESIGN VARIABLE FOR THE OPTIMIZER                        CCNO1160
C     VLE - DESIGN VARIABLE LOWER BOUND                                  CCNO1170
C     VUB - DESIGN VARIABLE UPPER BOUND                                  CCNO1180
C     G   - OPTIMIZER CONSTRAINT VECTOR                                  CCNO1190
C     IDG - OPTIMIZATION CONSTRAINT TYPE                                 CCNO1200
C     IC  - OPTIMIZER PARAMETER                                          CCNO1210
C     DF  - OPTIMIZER GRADIENT VECTOR                                    CCNO1220
C     XA  - OPTIMIZER STORAGE ARRAY                                      CCNO1230
C     WK1 - OPTIMIZER WORK SPACE                                         CCNO1240
C     IWK - OPTIMIZER WORK SPACE                                         CCNO1250
C     IR  - ARRAY USED TO CHOOSE DESIGN VARIABLES FROM INPUT DATA        CCNO1260
C     KEY - USED BY SORT ROUTINE                                         CCNO1270
C     R   - USED BY SORT ROUTINE                                         CCNO1280
C *********************************************************************  CCNO1290
      DIMENSION X(50),VLE(50),VUB(50),G(50),IDG(50),IC(50),DF(50),XA(50, CCNO1300
     1 50),WK1(1000),IWK(1000),IF(10,10),IR(50),KEY(50),R(50)            CCNO1310
      DATA NRA/50/,NCA/50/,NRWK/1000/,NRIWK/1000/                        CCNO1320
C *********************************************************************  CCNO1330
C                                                                        CCNO1400
C     INPUT DATA INTO THE PROGRAM TO SET UP THE SYSTEM                   CCNO1410
C     XDCI=AX+BU ; Y=CX                                                  CCNO1420
```

134

```
WHERE U=-FY+R   OR U=-FX+R   I.E.  C=DIA(1)
*********************************************************
*********************************************************
*********************************************************
*********************************************************
*********************************************************
*********************************************************
*********************************************************
*
CARD 1  KCDE& KONTRL FORMAT(I5,I5)
KDCE IS USER SUPPLIED NUMBER TO IDENTIFY THE DATA RUN
KONTRL SETS FLAG TO TELL PROGRAM HOW TO HANDLE RUN MAY BE:
        0 ENTER OPTIMIZER ON FIRST PASS THROUGH THE PROGRAM
        1 COMPUTE VALUES THROUGH PROGRAM WITHOUT OPTIMIZER
          ON FIRST PASS THEN OPTIMIZE
        2 OR MORE COMPUTE VALUES DO NOT OPTIMIZE
CARD 2  WMAX,WI,LELW; FORMAT(3F10.0)
WMAX - MAXIMUM FREQUENCY TO BE USED
WI   - STARTING FREQUENCY
LELW - FREQUENCY STEP FOR FIRST DECADE THIS STEP IS
       MODIFIED BY PROGRAM AT EACH DECADE SHIFT
CARD 3  WT1,WT2,WT3; FORMAT(3F10.0)
WT1 WEIGHTING FACTOR TO CAUSE THE OPTIMIZER TO USE THE
    PULE PLACEMENT PORTION OF THE OBJECTIVE FUNCTION
    IN THE OPTIMIZATION ROUTINE
WT2 WEIGHTING FACTOR WHICH CAUSES THE OPTIMIZER TO
    CONSIDER THE INPUT SINGULAR VALUE OF (1+FG) IN
    THE OPTIMIZATION OBJECTIVE FUNCTION
WT3 WEIGHTING FACTOR WHICH CAUSES THE OPTIMIZER TO
    USE THE OUTPUT SINGULAR VALUE (1+GFJ IN THE
    OBJECTIVE. IF THE OBJECTIVE NEEDS FURTHER PARAMETER
    ADDITION THEY MAY BE WEIGHTED IN SIMILAR FASHION
CARD 4  SVMINI,SVMINO,RJ,NIDG; FORMAT(2F10.0,I5)
SVMINI - DESIRED SINGULAR VALUE LEVEL FOR THE S.V. OF
         (1+FGJ FOUND USING UNIVERSAL GAIN AND PHASE CHART
         TO GIVE A DESIRED PHASE AND GAIN MARGIN
SVMINO - OUTPUT VERSION OF SVMINI
RJ - FACTOR TO SET PERCENT POLE PLACEMENT ERROR THE
     DESIGNER WISHES TO PUT INTO THE OPTIMIZER
NIDG - PARAMETER THAT SETS TYPE CONSTRAINT CONSIDERED
       SEE ADS MANUAL FOR USE
CARD 5  IGRAD,NDV,NCON,ISTRAT,ICPT,ICNEC,IPRINT,INFO
        FORMAT(8I5)
IGRAD - PARAMETER CONCERNING GRADIENT COMP SEE ADS
NDV - NUMBER OF DESIGN VARIABLES
NCON - NUMBER OF CONSTRAINTS
ISTRAT - OPTIMIZER STRATEGY SEE ADS MANUAL
ICPT - OPTIMIZATION METHOD SEE ADS
ICNED - ONE DIMENSIONAL SEARCH TECHNIQUE SEE ADS
IPRINT - ADS PRINT CONTROL SEE ADS
INFO - OPTIMIZER CONTROL PARAMETER SEE ADS
CARD 6  NROWA,NCCLA,NROWB,NCCLB,NROWC,NCCLC,NROWF,NCOLF,NMU
        FORMAT(9I2)
*********************************************************
*********************************************************
```

```
C      NUMBER CF RCWS AND COLUMNS CF INPUT MATRICES AND         *   CONO1930
C      NUMBER OF POLES TO BE INPUT FOR PLACEMENT                *   CCNO1940
C  CARD 7  A MATRIX ; READ BY ROWS AS:                          *   CCNO1950
C              A(1,1)     A(1,2)                                 *   CCNO1960
C              A(2,1)     A(2,2)                                 *   CONO1970
C                                                               *   CONO1980
C  THERE WILL BE GNE CR MCRE CARDS FOR EACH ROW OF A            *   CONO1990
C  AFTER THE PROPER CARDS ARE READ FOR A THE NEXT MATRIX MUST   *   CONO2000
C  BE READ IN. THE REMAINDER OF THIS COMMENT WILL REFER TO EACH NEW*  CONO2C10
C  DATA SET AS A CARD. REMEMBER, THERE MAY BE SEVERAL CARDS PER *   CONO2020
C  SET SO THE NUMBERS DON'T CORRESPOND DIRECTLY WITH THE DATA   *   CONO2C30
C  LISTED IN THE INPUT FILE.                                    *   CCNO2C40
C  CARD 8  E MATRIX; READ ROW BY ROW                            *   CCNO2C50
C  CARD 9  C MATRIX; READ ROW BY RCW                            *   CCNO2C60
C  CARD 10 F MATRIX; REAC ROW BY ROW                            *   CONO2070
C  CARD 11 IF MATRIX: ROW BY ROW                                *   CONO2C80
C           FORMAT(8I2) THIS MATRIX CEFINES THE DESIGN VARIABLE *   CONO2C90
C           TO THE CPTIMIZER. THE ENTRY WOULD BE OF THE FORM:   *   CONO2100
C                1                                              *   CONO2110
C                2     THIS WOULD TELL THE OPTIMIZER TO MAKE F(1,1) THE FIRST*  CONO2120
C                0     DESIGN VARIABLE (X(1)) AND SO FORTH. THE ZERO TELLS THE*  CCNO2130
C                3     OPTIMIZER THAT F(2,1) (IN THIS CASE) IS NOT A DESIGN  *   CONO2140
C                      VARIABLE AND IS NCT TC BE CHANGED.       *   CONO2150
C  CARD 12  VLB,VUB; FORMAT(2F10.0)                             *   CONO2160
C           THERE WILL BE GNE CARD FOR EACH DESIGN VARIABLE     *   CCNO2170
C  CARD 13  REALMU,IMAGMU; FORMAT (2F10.0)                      *   CCNO2180
C           INPUT REAL AND IMAGINARY CESIRED POLE LOCATION      *   CCNO2190
C           ONE POLE PER CARD                                   *   CONO2200
C ************************************************************** *   CONO2210
C ***************************************************************    CONO2220
C                                                                    CCNO2230
C  INPUT RLN COCE AND PROGRAM CONTRCL                                CCNO2240
C  REAL (1,750) KODE,KONTRL                                          CCNO2250
C                                                                    CCNO2260
C  INPUT MAXN, INITIAL FREQUENCY,DELTA FREG, GPTIMIZER WEIGHTS       CCNO2270
C  REAL (1,710) WMAX,WI,DELW                                         CONO2280
C  READ (1,570) WT1,WT2,WT3                                          CONO2290
C                                                                    CCNO2300
C  INPUT DESIREC GAIN AND PHASE MARGIN PARAMETERS                    CCNO2310
C                                                                    CCNO2320
C  REAL (1,580) SVMINI,SVMINO,RJ,NIDG                                CCNO2330
C  W=WI                                                              CCNO2340
C  NC=1                                                              CCNO2350
C                                                                    CCNO2360
C  INPUT CFTIMIZER INPUT CATA (SEE ADS MANUAL)                       CCNO2370
C                                                                    CONO2380
C                                                                    CCNO2390
C                                                                    CCNO2400
```

136

```
C      READ (1,760) IGRAD,ADV,NCON,ISTRAT,IOPT,IONED,IPRINT,INFO     CGNO2410
C                                                                     CGNO2420
C      INPUT THE PARAMETER VALUES AND THE MATRICES A,B,C,F            CGNO2430
C                                                                     CGNO2440
       READ (1,770) NROWA,NCOLA,NROWB,NCOLB,NROWC,NCOLC,NROWF,NCOLF,NMU CGNO2450
       CALL READ (A,NROWA,NCOLA)                                      CGNO2460
       CALL READ (B,NROWB,NCOLB)                                      CGNO2470
       CALL READ (C,NROWC,NCOLC)                                      CGNO2480
       CALL READ (F,NROWF,NCOLF)                                      CGNO2490
C                                                                     CGNO2500
C      SPECIFY THE DESIGN VARIABLES WITHIN THE F MATRIX               CGNO2510
C                                                                     CGNO2520
       DO 10 J=1,NROWF                                                CGNO2530
   10  READ (1,780) (IF(J,K),K=1,NCOLF)                               CGNO2540
C                                                                     CGNO2550
C      SET THE DESIGN VARIABLE BOUNDS                                 CGNO2560
C                                                                     CGNO2570
       DO 20 J=1,NCV                                                  CGNO2580
   20  READ (1,790) VLB(J),VUB(J)                                     CGNO2590
C                                                                     CGNO2600
C      FORMULATE THE DESIGN VARIABLE X FROM THE INPUT GUESS FOR F     CGNO2610
C                                                                     CGNO2620
       DO 50 J=1,NROWF                                                CGNO2630
       DO 40 K=1,NCOLF                                                CGNO2640
       IF (IF(J,K).EQ.0) GO TO 30                                     CGNO2650
       KK=IF(J,K)                                                     CGNO2660
       X(KK)=F(J,K)                                                   CGNO2670
   30  CONTINUE                                                       CGNO2680
   40  CONTINUE                                                       CGNO2690
   50  CONTINUE                                                       CGNO2700
C                                                                     CGNO2710
C      READ THE DESIRED EIGENVALUES AS REAL(X)+REAL(Y)*I              CGNO2720
C                                                                     CGNO2730
       DO 60 I=1,NMU                                                  CGNO2740
   60  READ (1,660) REALMU(I),IMAGMU(I)                               CGNO2750
       DO 70 I=1,NMU                                                  CGNO2760
       MU(I)=CMPLX(REALMU(I),IMAGMU(I))                               CGNO2770
   70  CONTINUE                                                       CGNO2780
C                                                                     CGNO2790
C      SORT THE INPUT EIGENVALUES USING IMSL ROUTINE VSRTR           CGNO2800
C                                                                     CGNO2810
       DO 80 I=1,NMU                                                  CGNO2820
   80  IR(II)=I                                                       CGNO2830
       CALL VSRTR (REALMU,NMU,IR)                                     CGNO2840
       DO 90 J=1,NMU                                                  CGNO2850
       K=IR(J)                                                        CGNO2860
   90  OMU(JJ)=MU(K)                                                  CGNO2870
       JJ=NMU-1                                                       CGNO2880
```

157

```
      DO 110 J=1,JJ
      IF (REAL(CMU(J)).NE.REAL(OMU(J+1))) GO TO 100
      IF (AIMAG(GPU(J)).LT.AIMAG(CMU(J+1))) GC TO 100
      TEMP1=CPL(J)
      CML(J)=CPL(J+1)
      CML(J+1)=TEMP1
100   CONTINUE
110   CONTINUE
C *****************************************************************
C     WRITE THE INPUT CATA FGR CHECKING ANC REFERENCE
C *****************************************************************
      WRITE (6,800) KGDE
      WRITE (6,720) WMAX,WI,CELW
      WRITE (6,590) WT1,WT2,WT3
      WRITE (6,600) SVMINI,SVMING,RJ,AIDG
      WRITE (3,610) NRCWA,NMU
      WRITE (6,810)
      CALL WRITE (A,NROWA,NCOLA)
      WRITE (6,820)
      CALL WRITE (B,NRGWB,NCCLB)
      WRITE (6,830)
      CALL WRITE (C,NRCWC,NCCLC)
      CALL WRITE (F,NRCWF,NCCLF)
      WRITE (6,850)
      CALL CVECWR (OMU,NML)
C
      IF KCNTRL SET GREATER THAN O THE FIRST PASS JUMPS THE OPTIMIZER
C
      IF (KCNTFL.GT.0) GC TO 130
C *****************************************************************
C     CALL THE ADS CPTIMIZER TC SOLVE FCR THE BEST F'S
C *****************************************************************
C     THIS FIRST CALL SETS THE SCALE FLAG SEE ACS MANUAL
      INFO=-2
      CALL ACS (INFO,ISTRAT,ICPT,IONED,IFRINT,IGRAD,NCV,NCUN,X,VLB,VUB,O
     1BJ,G,IEC,AGT,IC,LF,XA,NRA,NCA,WKI,NRWK,IWK,NRIWK)
      IWK(2)=1
C *****************************************************************
C     THIS CALL STARTS THE CPTIMIZATICN FROCESS
C *****************************************************************
120   CALL ACS (INFO,ISTRAT,ICPT,IUNED,IFRINT,IGRAD,NDV,NCUN,X,VLB,VUB,O
```

```
      1BJ,C,IEC,NGT,IC,DF,XA,NRA,NCA,KK1,NRKK,IWK,ARIWK)        CCNO3370
      IF (INFC.EQ.0) GC TC 480                                 CCNO3380
      IF (INFC.GT.1) GO TC 470                                 CCNO3390
C *********************************************************    CCNO3400
C     THIS PORTION OF THE PROGRAM DOES THE ACTUAL COMPUTATION  CCNO3410
C *********************************************************    CCNO3420
C *********************************************************    CCNO3430
130   DO 160 J=1,NROWF                                         CCNO3440
      DO 150 M=1,NCOLF                                         CCNO3450
C                                                              CCNO3460
C     EQUATE DESIGN VECTOR X WITH THE DESIRED SYSTEM MATRIX    CCNO3470
C                                                              CCNO3480
      IF (IF(.,K).EQ.0) GC TC 140                              CCNO3490
      KK=IF(J,K)                                               CCNO3500
      F(J,K)=X(KK)                                             CCNO3510
140   CONTINUE                                                 CCNO3520
150   CONTINUE                                                 CCNO3530
160   CONTINUE                                                 CCNO3540
C                                                              CCNO3550
C     COMPLEX ALL THE INPUT SYSTEM MATRICES FOR THE TRANSFER FUNCTION  CCNO3560
C                                                              CCNO3570
      CALL CFLXCV (AX,A,NROWA,NCCLA)                           CCNO3580
      CALL CFLXCV (BX,B,NROWB,NCCLB)                           CCNO3590
      CALL CFLXCV (CX,C,NROWC,NCCLC)                           CCNO3600
      CALL CFLXCV (FX,F,NROWF,NCCLF)                           CCNO3610
C                                                              CCNO3620
C     COMPUTE THE A-B*F*C SYSTEM MATRIX FOR EIGEN COMPUTATION  CCNO3630
C                                                              CCNO3640
      CALL MPLL (F,C,NROWF,NCOLF,NCOLC,FC)                     CCNO3650
      CALL MPLL (B,FC,NRCWB,NCOLE,NCOLC,BFC)                   CCNO3660
      DO 180 I=1,NROWA                                         CCNO3670
      DO 170 J=1,NCOLA                                         CCNO3680
      AMBFC(I,J)=A(I,J)-BFC(I,J)                               CCNO3690
170   CONTINUE                                                 CCNO3700
180   CONTINUE                                                 CCNO3710
C *********************************************************    CCNO3720
C     COMPUTE AND SORT THE EIGENVALUES AND VECTORS(IMSL ROUTINES)   CCNO3730
C *********************************************************    CCNO3740
C *********************************************************    CCNO3750
      CALL EIGRF (AMBFC,NROWA,10,2,EIG,Z,10,WK,IER)            CCNO3760
      DO 190 J=1,NROWA                                         CCNO3770
      REIG(J)=REAL(EIG(J))                                     CCNO3780
190   CONTINUE                                                 CCNO3790
      DO 200 J=J,NROWA                                         CCNO3800
      KEY(J)=J                                                 CCNO3810
200   CALL VSFTR (KEIG,NRCWA,KEY)                              CCNO3820
      DU 220 J=1,NCOLA                                         CCNO3830
                                                               CCNO3840
```

139

```fortran
      K=KEY(J)                                                          CCNO3650
      EIG(J)=EIG(K)                                                     CCNO3660
      DO 210 KK=1,NROWA                                                 CCNO3670
      OZ(KK,J)=Z(KK,K)                                                  CONO3680
210   CONTINUE                                                          CONO3890
220   CONTINUE                                                          CCNO3900
      JJ=ACCLA-1                                                        CCNO3910
      DO 250 J=1,JJ                                                     CONO3920
      IF (REAL(CEIG(J)).NE.REAL(CEIG(J+1))) GC TO 240                   CCNO3930
      IF (AIMAG(OEIG(J)).LT.AIMAG(OEIG(J+1))) GO TO 240                 CONO3940
      TEMP1=CEIG(J)                                                     CCNO3950
      CEIG(J)=CEIG(J+1)                                                 CCNO3960
      CEIG(J+1)=TEMP1                                                   CCNO3970
      DO 230 KK=1,NROWA                                                 CCNO3980
      TEMP2=CZ(KK,J)                                                    CCNO3990
      OZ(KK,J)=OZ(KK,J+1)                                               CCNO4003
      OZ(KK,J+1)=TEMP2                                                  CCNO4010
230   CONTINUE                                                          CCNO4020
240   CONTINUE                                                          CCNO4030
250   CONTINUE                                                          CCNO4040
C     **********************************************************        CCNO4050
C                                                                       CCNO4C60
C     CALL THE PLANT TRANSFER MATRIX ROUTINE AND FORM THE SYSTEM        CCNO4C70
C     RETURN DIFFERENCE MATRICES AS REQUIRED                           CCNO4C80
C                                                                       CLNO4C90
C     **********************************************************        CCNO4100
      CEL=DELB                                                          CCNO4110
      WII=WI                                                            CCNO4120
      NC=1                                                              CCNO4130
      W=WI                                                              CCNO4140
      SMINAI=C.                                                         CCNO4150
      SMINMI=C.                                                         CCNO4160
      SMINAG=C.                                                         CCNO4170
      SMINMO=C.                                                         CCNO4180
      SMAXAI=C.                                                         CCNO4190
      SMAXMI=C.                                                         CCNO4200
      SMAXAO=C.                                                         CCNO4210
      SMAXMC=0.                                                         CCNO4220
260   CALL PLANT (PLAN,AX,BX,CX,W,NRCWA,NCCLA,NROWB,NCOL B,NRROWC,NCOL C)CCNO4230
      CALL CPATML (FX,PLAN,NRUWF,NCOLF,NCCLB,NCCLLB,FXPLT)              CCNO4240
      CALL CPATML (PLAN,FX,NRCWC,NCOLB,NCCLLF,PLTFX)                    CUNO4250
      DO 280 J=1,NROWF                                                  CCNO4260
      DO 270 K=1,NCOLB                                                  CCNO4273
      XI(J,K)=C.0                                                       CCNO4280
270   CONTINUE                                                          CCNO4290
280   CONTINUE                                                          CONO4300
      DO 290 J=1,NROWF                                                  CCNO4310
290   XI(J,J)=1.0                                                       CLNO4320
      DO 310 J=1,NROWC
```

140

```
        DO 300 K=1,NCOLF                                      CCNO04430
        XXI(J,K)=0.0                                          CCNO04440
        XXI(J,K)=1.0                                          CCNO04350
        XIPPFX(J,K)=XXI(J,K)+PLTFX(J,K)                       CCNO04360
300     CONTINUE                                              CCNO04370
310     CONTINUE                                              CCNO04380
        DO 330 J=1,NROWF                                      CCNO04390
        DO 320 K=1,NCOLB                                      CCNO04400
        XIPFXP(J,K)=XXI(J,K)+FXFLT(J,K)                       CCNO04410
        CONTINUE                                              CCNO04420
320                                                           CCNO04430
330     DO 350 J=1,NROWF                                      CCNO04440
        DO 340 K=1,NROWF                                      CCNO04450
        FXPLTI(J,K)=0.0                                       CCNO04460
        FXFLTI(J,K)=1.0                                       CCNO04470
        DO 370 J=1,NROWC                                      CCNO04480
        DO 360 K=1,NROWC                                      CCNO04490
        PLTFXI(J,K)=0.0                                       CCNO04500
        PLTFXI(J,K)=1.                                        CCNO04510
360     CONTINUE                                              CCNO04520
370     CALL LECTIC (FXPLT,NROWF,10,FXPLTI,NROWF,10,0,WA,IER) CCNO04530
C       CALL LECTIC (PLTFX,NROWC,10,PLTFXI,NROWC,10,0,WA,IER) CCNO04540
C       DO 390 J=1,NROWF                                      CCNO04550
        DO 380 K=1,NCOLB                                      CCNO04560
        XIFPLI(J,K)=XI(J,K)+FXPLTI(J,K)                       CCNO04570
        CONTINUE                                              CCNO04580
380                                                           CCNO04590
390     DO 410 J=1,NROWC                                      CCNO04600
        DO 400 K=1,NROWC                                      CCNO04610
        XIPLFI(J,K)=XXI(J,K)+PLTFXI(J,K)                      CCNO04620
400     CONTINUE                                              CCNO04630
410     CONTINUE                                              CCNO04640
C       *********************************************         CCNO04650
C       *********************************************         CCNO04660
C CO SINGULAR VALUE DECOMP AND QUANTIFY ALL DESIRED SV'S      CCNO04670
C       *********************************************         CCNO04680
C       *********************************************         CCNO04690
        CALL CSVC (XIPFXP,10,10,NRCWF,NROWF,0,NRO F,NROWF,SV,UI,VI)     CCNO04700
        CALL CSVC (XIPPFX,10,10,NRCWC,NROWC,0,NROWC,NRCWC,SVAO,UO,VO)   CCNO04710
        CALL CSVC (XIFPLI,10,10,NRCWF,NROWF,0,NROWF,NRGWF,SVMI,LMI,VMI) CCNO34720
        CALL CSVC (XIPLFI,10,10,NRCWC,NROWC,0,NROWC,NROWC,SVMO,LMO,VMO) CCNO04730
        OMEGA(NC)=W                                           CCNO04740
        SIGNM1(NC)=SV(NRGWF)                                  CCNO04750
        SIGNM2(NC)=SV(NROWF-1)                                CCNO04763
        SIGNMX(NC)=SV(1)                                      CCNO04773
        SVACMG(NC)=SVAC(NRCWC)                                CCNO04780
        SVADXO(NC)=SVAO(1)                                    CCNO04790
        SVMIM(NC)=SVMI(NROWF)                                 CCNO04800
```

141

```
      SVMIX(NC)=SVMI(1)
      SVMAO(NC)=SVMO(NRUWC)
      SVMXO(NC)=SVMO(1)
      SIGPRC(NC)=SQRT(SIGMA1(NC)*SIGAM2(NC))
      SMINAI=(SMINAI+(AMAXI(0.0;SVMINI-SIGNM1(NC)))**2)
      SMINAO=(SMINAO+(AMAXI(0.0;SVMINO-SVADMC(NC)))**2)
C
C     COMPUTE LCOF FCR ALL DESIRED FREQUENCY BANDWIDTH
C
      W=W+DEL
      WD=10.-C*WII
      IF(W.LT.WD) GO TO 420
      DEL=10.-C*DEL
      WII=10.-C*WII
  420 NC=NC+1
      IF(W.LE.WMAX) GO TO 260
      MAXN=NC-1
C
C     FORM QUANTITES TO USE IN THE COST CRITERIA OF OPTIMIZER
C
      SVMIN=SIGNMI(1)
      DO 430 J=1,MAXN
      IF(SIGNMI(J).GE.SVMINI) GO TO 430
      SVMIN=SIGNMI(J)
  430 CONTINUE
      COST=0.
      DO 440 J=1,MU
      COST=CCST+((REAL(OMU(J))-REAL(CEIG(J)))**2+(AIMAG(OMU(J))-AIMAG(OE
     1IG(J)))**2)
  440 CONTINUE
C
C     THE OBJECTIVE FUNCTION IS INSERTED HERE
C
      OBJ=WT1*COST+WT2/SVMIN
      OBJ=WT1*CCST+WT2*SVMINAI+WT3*SMINAU
      OBJ=WT2*SMINAI
      IF(NCCA.EQ.0) GO TO 460
      DO 450 J=1,MU
      IDG(J)=NIDG
      R(J)=RJ
      R(J)=RJ+SVFT((REAL(CMU(J))-REAL(CEIG(J)))**2
  1CEIG(J))**2)+SVFT((REAL(OMU(J))-REAL(OEIG(J)))**2+(AIMAG(CEI
  450 IG(J)))=SVFT((REAL(OML(J))-REAL(GEIG(J)))**2+(AIMAG(OMU(J))-AIMAG(CE
  460 IF(KCATFL.GT.0) GO TO 480
      GO TO 16C
      CONTINUE
C
  470 CONTINUE
```

```
C     THE GRADIENT COMPUTATION ROUTINE GOES HERE IF NEEDED       CCN05290
      GO TO 125                                                  CCN05300
480   CONTINUE                                                   CCN05310
C                                                                CCN05320
C     OUTPUT THE COMPUTED FINAL EIGEN VALUES AND VECTORS FROM OPT F'S  CCN05330
C                                                                CCN05340
      CALL MPLL (F,C,NROWF,NCOLF,FC)                             CCN05350
      CALL MPLL (B,FC,NROWB,NCOLB,NCCLC,BFC)                     CCN05360
      DO 500 I=1,NROWA                                           CCN05370
      DO 490 J=1,NCOLA                                           CCN05380
490   AMBFC(I,J)=A(I,J)-BFC(I,J)                                 CCN05390
500   CONTINUE                                                   CCN05400
      IF (KCNTRL.EQ.0) GO TO 510                                 CCN05410
      WRITE (6,620)                                              CCN05420
      GO TO 530                                                  CCN05430
510   WRITE (6,630)                                              CCN05440
520   CALL EIGRF (AMBFC,NROWA,10,2,EIG,Z,10,WK,IER)             CCN05450
      WRITE (6,870)                                              CCN05460
      CALL CVECWR (OEIG,NCOLA)                                   CCN05470
      DO 530 J=1,AMJ                                             CCN05480
      WRITE (2,640) OEIG(J)                                      CCN05490
      WRITE (6,880)                                              CCN05500
      CALL CMFITE (OZ,NRCWA,NCOLA)                               CCN05510
530   WRITE (6,890)                                              CCN05520
      CALL WRITE (F,NROWF,NCOLF)                                 CCN05530
      WRITE (6,650) OBJ                                          CCN05540
C                                                                CCN05550
C     OUTPUT THE OPTIMIZED SINGULAR VALUE DATA                   CCN05560
C                                                                CCN05570
      WRITE (6,73C) MAXN                                         CCN05580
      WRITE (5,66C) MAXN                                         CCN05590
      DO 540 J=1,MAXN                                            CCN05600
      WRITE (3,67C) OMEGA(J),SIGNM1(J),SIGNM2(J),SIGNMX(J),SIGPRG(J)  CCN05610
540   WRITE (4,74C) OMEGA(J),SIGNMI(J),SIGNM2(J),SIGNMX(J),SIGPRO(J)  CCN05620
      DO 550 J=1,MAXN                                            CCN05630
      WRITE (2,69C) SVADPC(J),SVADXO(J),SVMIP(J),SVMIX(J),SVMMO(J),SVMXO  CCN05640
     1(J)                                                        CCN05650
550   WRITE (6,70C) OMEGA(J),SVACMO(J),SVADXC(J),SVMIM(J),SVMIX(J),SVMMU  CCN05660
     1(J),SVMXU(J)                                               CCN05670
      IF (KCNTFL.EQ.0.OR.KONTRL.GT.1) GO TO 560                  CCN05680
      KONTRL=C                                                   CCN05690
      GO TO 12C                                                  CCN05700
560   STCP                                                       CCN05710
C     ***********************************************            CCN05720
C     FORMAT STATEMENTS                                          CCN05730
C     ***********************************************            CCN05740
570   FORMAT (3F10.0)                                            CCN05760
```

143

```
      580 FORMAT (2F10.0,15)                                             CCNO5770
      590 FORMAT (/,1CH WEIGHT 1=,F10.5,9HWEIGHT 2=,F10.5,9HWEIGHT 3=,F10.5,CCNO5780
     1/)                                                                 CCNO5790
      600 FORMAT (/,9F SVMINI= ,F10.5,8H SVMINO=,F10.5,5H RJ=,F10.5,6H IDGCONO5800
     1=,15,/)                                                            CCNO5810
      610 FORMAT (215)                                                    CCNO5820
      620 FORMAT (/,2EH *** THE INITIAL DATA IS ***,/)                    CCNO5830
      630 FORMAT (/,41H *** THE CPTIMIZER CONFUTEC OUTPUT IS ***,/)       CCNO5840
      640 FORMAT (2E12.5)                                                 CCNO5850
      650 FORMAT (//,24H *** THE OBJECTIVE FUNCTION IS =,F10.5,6H ***,/)  CCNO5860
      660 FORMAT (15)                                                     CCNO5870
      670 FORMAT (5E12.5)                                                 CCNO5880
      680 FORMAT (//,2X,7H FREQ ,5X,6HSVADMO,4X,6HSVALXO,4X,5HSVMIM,4X,5HSVMCCNO5890
     1IX,4X,5HSVMMO,4X,5HSVMXO,/)                                        CCNO5900
      690 FORMAT (6E12.5)                                                 CCNO5910
      700 FORMAT (7F1C.5)                                                 CCNO5920
      710 FORMAT (3F10.5)                                                 CCNO5930
      720 FORMAT (/,6H WMAX=,F10.5,1X,10HINIT FREG=,F10.5,1X,10HFREQ STEP=,FCCNO5940
     110.5,/)                                                            CCNO5950
      730 FORMAT (//,10H FREQUENCY,5X,13HMIN ADD IN SV,4X,13H2ND ADD IN SV,3CCNO5960
     1X,13HMAX ADD IN SV,2X,18HMIN PRGD ACD IN SV)                       CCNO5970
      740 FORMAT (5(F10.5,5X))                                            CCNO5980
      750 FORMAT (15,15)                                                  CCNO5990
      760 FORMAT (E15)                                                    CCNO6000
      770 FORMAT (E12)                                                    CCNO6010
      780 FORMAT (E12)                                                    CCNO6020
      790 FORMAT (2F1C.0)                                                 CCNO6030
      800 FORMAT (1H1,///,12F+<<<<< RUN #,15,6H >>>>>,///)               CCNO6040
      810 FORMAT (27H *** THE A PLANT MATRIX ***,///)                     CCNO6050
      820 FORMAT (/,35H *** THE B CCNTRCL INPUT MATRIX ***,///)           CCNO6060
      830 FORMAT (/,33H *** THE C OBSERVATICN MATRIX ***,///)             CCNO6070
      840 FORMAT (/,38H *** THE F FEEDBACK MATRIX ***,///)                CCNO6080
      850 FORMAT (/,44H *** THE CRDERED COMPLEX EIGENVALUES (INPUT),///)  CCNO6090
      860 FORMAT (2F1C.0)                                                 CONO6100
      870 FORMAT (/,37H *** THE CRDERED COMPUTEC EIGENVALUES,///)         CCNO6110
      880 FORMAT (/,29H *** THE CRDERED EIGENVECTORS,///)                 CCNO6120
      890 FORMAT (/,26H *** THE F FEEDEACK MATRIX ***,///)                CCNO6130
          ENC                                                            CCNO6140
```

144

```
C*********************************************************************     CWROCC010
CSUBROUTINE PACKAGE FOR CCNXSV AND CCNQB1                                  CWROCC020
C*********************************************************************     CWROCC030
C*********************************************************************     CWROCC040
C*********    WRITE THE COMPLEX MATRIX                                     CWROCC050
C*********************************************************************     CWROCC060
C*********************************************************************     CWROCC070
      SUBROUTINE CWRITE (A,N,K)                                            CWROCC080
      COMPLEX *8A(10,10)                                                   CWROCC090
      INTEGER N,K,I,J                                                      CWROCC100
      DO 10 I=1,N                                                          CWROCC110
      WRITE (6,20) (A(I,J),J=1,K)                                          CWROCC120
   10 CONTINUE                                                             CWROCC130
      RETURN                                                               CWROCC140
   20 FORMAT (8F12.5)                                                      CWROCC150
      END                                                                  CWROCC160
C*********************************************************************     CWROCC170
C*********    WRITE THE COMPLEX VECTOR                                     CWROCC180
C*********************************************************************     CWROCC190
      SUBROUTINE CVECWR (A,N)                                              CWROCC200
      COMPLEX *8A(10)                                                      CWROCC210
      INTEGER N,J                                                          CWROCC220
      WRITE (6,10) (A(J),J=1,N)                                            CWROCC230
      RETURN                                                               CWROCC240
   10 FORMAT (2F12.5)                                                      CWROCC250
      END                                                                  CWROCC260
C*********************************************************************     CWROCC270
C*********    WRITE THE REAL VECTOR                                        CWROCC280
C*********************************************************************     CWROCC290
      SUBROUTINE VECWR (A,N,IR)                                            CWROCC300
      REAL*4 A(10)                                                         CWROCC310
      DIMENSION IR(50)                                                     CWROCC320
      INTEGER N,J                                                          CWROCC330
      WRITE (6,10) (A(J),IR(J),J=1,N)                                      CWROCC340
      RETURN                                                               CWROCC350
   10 FORMAT (F12.5,5X,I5)                                                 CWROCC360
      END                                                                  CWROCC370
C*********************************************************************     CWROCC380
C*********    MATRIX MULTIPLICATION WITH COMPLEX MATRIX                    CWROCC390
C*********************************************************************     CWROCC400
      SUBROUTINE CMATML (R,T,M,LL,N,U)                                     CWROCC410
      COMPLEX *8R(10,10),T(10,10),U(10,10)                                 CWROCC420
      INTEGER M,LL,N                                                       CWROCC430
      DO 30 I=1,M                                                          CWROCC440
      DO 20 J=1,N                                                          CWROCC450
      U(I,J)=C.0                                                           CWROCC460
      DO 10 INDEX=1,LL                                                     CWROCC470
      U(I,J)=U(I,J)+R(I,INDEX)*T(INDEX,J)                                  CWROCC480
```

145

```
10    CONTINUE                                                    CWROC490
20    CONTINUE                                                    CWROC500
30    CONTINUE                                                    CWROC510
      RETURN                                                      CWROC520
      END                                                         CWROC540
C     ***********************************************************  CWROC550
C     ******* MATRIX MULTIPLICATION ROUTINE *******************    CWROC560
C     ***********************************************************  CWROC570
      SUBROUTINE MMUL (R,T,M,LL,N,U)                               CWROC580
      REAL*4 F(10,10),T(10,10),U(10,10)                            CWROC590
      INTEGER M,LL,N                                               CWROC600
      DO 30 I=1,M                                                  CWROC610
      DO 20 J=1,N                                                  CWROC620
      U(I,J)=0.0                                                   CWROC630
      DU 10 INDEX=1,LL                                             CWROC640
      U(I,J)=U(I,J)+R(I,INDEX)*T(INDEX,J)                          CWROC650
10    CONTINUE                                                     CWROC660
20    CONTINUE                                                     CWROC670
30    CONTINUE                                                     CWROC680
      RETURN                                                       CWROC690
      END                                                          CWROC700
C     ***********************************************************  CWROC710
C     CSVD ALGORITHM FOR COMPLEX MATRICES; ANY OF THE ROUTINES OF IMSL  CWROC720
C     OR LINPACK COULD ALSO BE USED WITH PROPER CAUTIONS           CWROC730
C     ***********************************************************  CWROC740
      SUBROUTINE CSVD (A,MMAX,NMAX,M,N,P,NU,NV,S,L,V)              CWROC750
      COMPLEX A(MMAX,1),L(MMAX,1),V(NMAX,1)                        CWROC760
      INTEGER M,N,P,NU,NV                                          CWROC770
      REAL S(1)                                                    CWROC780
      COMPLEX Q,R                                                  CWROC790
      REAL B(100),C(100),T(100)                                    CWROC800
      DATA ETA,TOL/1.5E-8,1.E-31/                                  CWROC810
      NP=N+P                                                       CWROC820
      N1=N+1                                                       CWROC830
C     HOUSEHOLDER REDUCTION                                        CWROC840
      C(1)=0.EO                                                    CWROC850
      K=1                                                          CWROC860
10    K1=K+1                                                       CWROC870
C     ELIMINATION OF A(I,K),I=K+1,....,M                           CWROC880
      Z=C.EO                                                       CWROC890
      DO 20 I=K,M                                                  CWROC900
      Z=Z+REAL(A(I,K))**2+AIMAG(A(I,K))**2                         CWROC910
      B(K)=0.EO                                                    CWROC920
      IF (Z.LE.TOL) GO TO 70                                       CWROC930
      Z=SQRT(Z)                                                    CWROC940
      B(K)=Z                                                       CWROC960
```

146

```
      W=CABS(A(K,K))
      Q=(1.EC,C.EC)
      IF(W.NE.0.EO) Q=A(K,K)/W
   30 A(K,K)=C*(Z+W)
      IF (K.EC.NP) GO TO 70
      DO 50 J=K1,NP
      Q=(0.EC,C.EC)
      DO 30 I=K1M
      Q=G+CCNJG(A(I,K))*A(I,J)
      Q=Q/(Z*(Z+W))
      DO 40 I=K,M
   40 A(I,J)=A(I,J)-Q*A(I,K)
   50 CONTINLE
C
C     PHASE TRANSFORMATION
   60 Q=-CCNJG(A(K,K))/CAES(A(K,K))
      DO 60 J=K1,NP
      A(K,J)=Q*A(K,J)
C
C     ELIMINATION OF A(K,J),J=K+2,...,N
   70 IF (K.EC.N) GO TO 140
      Z=C.EO
      DO 80 J=K1,N
      Z=Z+REAL(A(K,J))**2+AIMAG(A(K,J))**2
   80 C(K1)=C.EO
      IF (Z.LE.TOL) GO TO 130
      Z=SCRT(Z)
      C(K1)=Z
      W=CABS(A(K,K1))
      C=(1.EC,C.EO)
      IF (W.NE.C.EO) Q=A(K,K1)/W
      A(K,K1)=C*(Z+W)
      DO 110 I=K1,M
      Q=(0.EC,0.EC)
      DO 90 J=K1,N
   90 Q=G+CONJG(A(K,J))*A(I,J)
      C=C/(Z*(Z+W))
      DO 100 J=K1,N
  100 A(I,J)=A(I,J)-Q*A(K,J)
  110 CONTINLE
C
C     PHASE TRANSFORMATION
  120 Q=-CCNJG(A(K,K1))/CABS(A(K,K1))
      DO 120 I=K1,M
      A(I,K1)=A(I,K1)*Q
  130 K=K1
      GO TO 10
C
```

```
C   TOLERENCE FOR NEGLIGIBLE ELEMENTS                    CHRO1450
140    EPS=0.E0                                          CHRO1460
       DO 150 K=1,N                                      CHRO1470
       S(K)=B(K)                                         CHRO1480
       T(K)=C(K)                                         CHRO1490
150    EPS=AMAX1(EPS,S(K)+T(K))                          CHRO1500
       EPS=EPS*ETA                                       CHRO1510
C                                                        CHRO1520
C   INITIALIZATION OF U AND V                            CHRO1530
       IF (NU.EC.0) GO TO 180                            CHRO1540
       DO 170 I=1,NU                                     CHRO1550
160    DO 160 J=1,M                                      CHRO1560
170    U(I,J)=(0.E0,0.E0)                                CHRO1570
180    U(J,J)=(1.E0,0.E0)                                CHRO1580
       IF (NV.EC.0) GO TO 210                            CHRO1590
       DO 200 I=1,NV                                     CHRO1600
190    DO 190 J=1,N                                      CHRO1610
       V(I,J)=(0.E0,0.E0)                                CHRO1620
200    V(J,J)=(1.E0,0.E0)                                CHRO1630
C   QR DIAGONALIZATION                                   CHRO1640
210    DO 380 KK=1,N                                     CHRO1650
       K=N-KK                                            CHRO1660
C                                                        CHRO1670
C   TEST FOR SPLIT                                       CHRO1680
220    DO 230 LL=1,K                                     CHRO1690
       L=K+1-LL                                          CHRO1700
       F=SN*T(L)                                         CHRO1710
       T(L)=CS*T(L)                                      CHRO1720
       IF (ABS(T(L)).LE.EPS) GO TO 290                   CHRO1720
       IF (ABS(S(L-1)).LE.EPS) GO TO 240                 CHRO1740
230    CONTINUE                                          CHRO1750
C   CANCELLATION OF E(L)                                 CHRO1760
240    CS=C.E0                                           CHRO1770
       SN=1.E0                                           CHRO1780
       LL=L-1                                            CHRO1790
       DO 280 I=L,K                                      CHRO1800
       F=SN*T(I)                                         CHRO1810
       T(I)=CS*T(I)                                      CHRO1820
       IF (ABS(F).LE.EPS) GO TO 290                      CHRO1830
       H=S(I)                                            CHRO1840
       H=SQRT(F*F+H*H)                                   CHRO1850
       S(I)=H                                            CHRO1860
       CS=H/H                                            CHRO1870
       SN=-F/H                                           CHRO1880
       IF (NU.EC.0) GO TO 260                            CHRO1890
       DO 250 J=1,N                                      CHRO1900
       X=REAL(L(J;L))                                    CHRO1910
       Y=REAL(L(J,I))                                    CHRO1920
```

148

```
250   U(J,L1)=CMPLX(X*CS+Y*SN,0.E0)                              CWRO1930
260   U(J+1)=CMPLX(Y*CS-X*SN,0.E0)                               CWRO1940
      IF (NP.EC.N) GO TO 280                                     CWRO1950
      DO 270 J=N1,NP                                             CWRO1960
      G=A(L1,J)                                                  CWRO1970
      R=A(I,J)                                                   CWRO1980
      A(L1,J)=G*CS+R*SN                                          CWRO1990
270   A(I,J)=R*CS-G*SN                                           CWRO2000
280   CONTINUE                                                   CWRO2010
C                                                                CWRO2020
C     TEST FOR CONVERGENCE                                       CWRO2030
290   W=S(K)                                                     CWRO2040
      IF (L.EC.K) GO TO 360                                      CWRO2050
C                                                                CWRO2060
C     ORIGIN SHIFT                                               CWRO2070
      X=S(L)                                                     CWRO2080
      Y=S(K-1)                                                   CWRO2090
      G=T(K-1)                                                   CWRO2100
      H=T(K)                                                     CWRO2110
      F=((Y-H)*(Y+H)+(G-H)*(G+H))/(2.E0*H*Y)                     CWRO2120
      G=SCRT(F*F+1.E0)                                           CWRO2130
      IF (F-L1.O.E0) G=-G                                        CWRO2140
      F=((X-H)*(X+H)+(Y/(F+G)-H)*H)/X                            CWRO2150
C                                                                CWRO2160
C     GR STEP                                                    CWRO2170
      CS=1.E0                                                    CWRO2180
      SN=1.E0                                                    CWRO2190
      L1=L+1                                                     CWRO2200
      DO 350 I=L1,K                                              CWRO2210
      G=T(I)                                                     CWRO2220
      Y=S(I)                                                     CWRO2230
      H=SN*G                                                     CWRO2240
      G=CS*G                                                     CWRO2250
      W=SCRT(G*G+F*F)                                            CWRO2260
      T(I-1)=W                                                   CWRO2270
      CS=F/W                                                     CWRO2280
      SN=H/W                                                     CWRO2290
      F=X*CS+G*SN                                                CWRO2300
      G=G*CS-X*SN                                                CWRO2310
      H=Y*SN                                                     CWRO2320
      Y=Y*CS                                                     CWRO2330
      IF (NV.EC.O) GO TO 310                                     CWRO2340
      DO 300 J=1,N                                               CWRO2350
      X=REAL(V(J,I-1))                                           CWRO2360
      W=REAL(V(J,I))                                             CWRO2370
      V(J,I-1)=CMPLX(X*CS+W*SN,0.E0)                             CWRO2380
300   V(J,I)=CMPLX(W*CS-X*SN,0.E0)                               CWRO2390
310   W=SCRT(H*H+F*F)                                            CWRO2400
```

149

```
         S(I-1)=H
         CS=F/H
         SN=H/H
         I=CS*G+SN*Y
         X=CS*Y-SN*G
         IF (NU.EQ.0) GO TO 330
         DO 320 J=1,N
         Y=REAL(U(J,I-1))
         W=REAL(U(J,I))
         U(J,I-1)=CMPLX(Y*CS+W*SN,0.E0)
  320    U(J,I)=CMPLX(W*CS-Y*SN,0.E0)
  330    IF (N.EQ.NP) GO TO 350
         DO 340 J=N1,NP
         G=A(I-1,J)
         R=A(I,J)
         A(I-1,J)=Q*CS+R*SN
  340    A(I,J)=R*CS-Q*SN
  350    CONTINUE
         T(L)=0.E0
         T(K)=F
         S(K)=X
         GO TO 220
C
C CONVERGENCE
  360    IF (W.GE.0.E0) GO TO 380
         S(K)=-W
         J=-K
         IF (NV.EQ.0) GO TO 380
         DO 370 J=1,N
  370    V(J,K)=-V(J,K)
  380    CONTINUE
C
C SORT SINGULAR VALUES
         DO 450 K=1,N
         G=-1.EC
         J=K
         DO 390 J=K,N
         IF (S(I).LE.G) GO TO 390
         G=S(I)
         J=I
  390    CONTINUE
         IF (J.EC.K) GO TO 450
         S(J)=S(K)
         S(K)=G
         IF (NV.EQ.0) GO TO 410
         DO 400 I=1,N
         G=V(I,J)
         V(I,J)=V(I,K)
  400    V(I,K)=G
```

```
410   IF (NL.EQ.0) GO TO 430
      DO 420 J=1,N
      G=L(I,J)
      U(I,J)=L(I,K)
420   U(I,K)=G
430   IF (N.EQ.NP) GO TO 450
      DO 440 I=N1,NP
      G=A(J,I)
      A(J,I)=A(K,I)
440   A(K,I)=G
450   CONTINUE
C
C     BACK TRANSFORMATION
      IF (NL.EQ.0) GO TO 510
      DO 500 KK=1,N
      K=N1-KK
      IF (B(K).EQ.0.E0) GO TC 500
      Q=-A(K,K)/CABS(A(K,K))
      DO 460 I=1,NU
460   U(K,J)=Q*U(K,J)
      DO 490 J=1,NU
      Q=(0.E0,C.E0)
      DO 470 I=K,N
470   Q=Q+CONJG(A(I,K))*U(I,J))
      Q=C/(CABS(A(K,K))*B(K))
      DO 480 I=K,N
480   U(I,J)=U(I,J)-Q*A(I,K)
490   CONTINUE
500   CONTINUE
510   IF (NV.EQ.0) GO TO 570
      IF (N.LT.2) GO TG 570
      DO 560 KK=2,N
      K=N1-KK
      K1=K+1
      IF (C(K1).EC.0.E0) GO TO 560
      Q=-CONJG(A(K,K1))/CABS(A(K,K1))
      DO 520 J=1,NV
520   V(K1,J)=Q*V(K1,J)
      DO 530 J=1,NV
      Q=(0.E0,0.E0)
      DO 530 I=K1,N
530   Q=Q+A(K1,I)*V(I,J))*C(K1))
      Q=C/(CABS(A(K,K1))*C(K1))
      DO 540 I=K1,N
540   V(I,J)=V(I,J)-Q*CONJG(A(K,I))
550   CONTINUE
560   CONTINUE
570   RETURN
```

151

```
      END
C***** READ A REAL MATRIX ***********                              CWRO 370
C*********************************                                 CWRO 380
      SUBROUTINE READ (A,N,K)                                      CWRO 390
      REAL*4 A(10,10)                                              CWRO 400
      INTEGER N,K                                                  CWRO 410
      DO 10 I=1,N                                                  CWRO 420
      READ (1,20) (A(I,J),J=1,K)                                   CWRO 430
10    CONTINUE                                                     CWRO 440
      RETURN                                                       CWRO 450
20    FORMAT (8F10.0)                                              CWRO 460
      END                                                          CWRO 470
C***** WRITE A REAL MATRIX ***********                             CWRO 480
C*********************************                                 CWRO 490
      SUBROUTINE WRITE (A,N,K)                                     CWRO 500
      REAL*4 A(10,10)                                              CWRO 510
      INTEGER N,K                                                  CWRO 520
      DO 10 I=1,N                                                  CWRO 530
      WRITE (6,20) (A(I,J),J=1,K)                                  CWRO 540
10    CONTINUE                                                     CWRO 550
      RETURN                                                       CWRO 560
20    FORMAT (8F10.5)                                              CWRO 570
      END                                                          CWRO 580
C***** CONVERT REAL MATRIX TO COMPLEX FORM ****                    CWRO 590
C*********************************                                 CWRO 600
      SUBROUTINE CPLXCV (X,A,N,K)                                  CWRO 610
      COMPLEX*8 X(10,10)                                           CWRO 620
      REAL*4 A(10,10)                                              CWRO 630
      INTEGER N,K                                                  CWRO 640
      DO 20 I=1,N                                                  CWRO 650
      DO 10 J=1,K                                                  CWRO 660
      X(I,J)=CMPLX(A(I,J),0.00)                                    CWRO 670
10    CONTINUE                                                     CWRO 680
20    CONTINUE                                                     CWRO 690
      RETURN                                                       CWRO 700
      END                                                          CWRO 710
C*********************************                                 CWRO 720
C     THIS ROUTINE WILL COMPUTE THE TRANSFER FUNCTION AS A         CWRO 730
C     FUNCTION OF FREQUENCY FOR A GIVEN A, B, C SET OF             CWRO 740
C              MATRICES                                            CWRO 750
C*********************************                                 CWRO 760
      SUBROUTINE FLANT (GP,AX,BX,CX,W,NRA,NCA,NRB,NCB,NRC,NCC)     CWRO 770
      COMPLEX*8CX(10,10),AX(10,10),BX(10,10),GP(10,10),SIA(10,     CWRO 780
     110),SIAI(10,10),SIAIB(10,10),SI(10,10)                       CWRO 790
      COMPLEX SI                                                   CWRO 800
                                                                   CWRO 810
                                                                   CWRO 820
                                                                   CWRO 830
                                                                   CWRO 840
```

152

```
      REAL*4 WA(100)
      S=CMPLX(C.0,W)
      DO 20 J=1,NRA
      DO 10 K=1,NCA
      SI(J,K)=C.0
   10 I(J,K)=C.0
   20 CONTINUE
      DO 30 J=1,NCA
      I(J,J)=1.0
   30 SI(J,J)=S*I(J,J)
      DO 50 J=1,NRA
      DO 40 K=1,NCA
      SIA(J,K)=SI(J,K)-AX(J,K)
   40 CONTINUE
   50 CONTINUE
      CALL LECTIC (SIA,NCA,10,I,NCA,10,I,NCA,IER)
      CALL CMAIML (II,BX,NCA,NCA,NCB,SIAIB)
      CALL CMAIML (CX,SIAIB,NRC,ACC,NCB,GP)
      RETURN
      END
C ****************************************************
C     THIS ROUTINE IS INPUT FOR DE SOLVER
CWHEN DE SOLVER IS ADDED DIRECTLY TO CONXSV
C ****************************************************
      SUBROUTINE FCN (N,T,X,XDOT)
      REAL*4 T,X(N),XDOT(N),AMBFC(10,10),A(10,10),B(10,10),C(10,10),
     1F(10,10),PHI(10),BFC(10,10),FC(10,10),AMBFCX(10),BPHI(10)
      COMMON AMBFC,A,B,F,C,PHI,AMBFCX,BPHI,NCOLB
      WRITE(6,4004)N,NCOLB
 4004 FORMAT(2I5)
      WRITE(6,4006)(PHI(J),J=1,NCOLB)
 4006 FORMAT(2F10.5)
      CALL WRITE(E,N,NCOLB)
      CALL WRITE(AMBFC,N,N)
      DO 10 J=1,N
      BPHI(J)=C.0
      DO 20 K=1,N
   20 BPHI(J)=BPHI(J)+B(J,K)*PHI(K)
   10 CONTINUE
C     WRITE(6,5500)(BPHI(J),J=1,N)
C5500 FORMAT(8F10.5)
      DO 40 J=1,N
      AMEFCX(J)=0.0
      DO 40 K=1,N
   40 AMEFCX(J)=AMBFCX(J)+AMEFC(J,K)*X(K)
   30 CONTINUE
C     WRITE(6,5501)(AMBFCX(J),J=1,N)
C5501 FORMAT(8F10.5)
      DO 50 J=1,N
```

153

```
 5C   XDCT(J)=AMBFCX(JJ+BPHI(J)                              CWR04430
      WRITE(6,5502)(XDGT(J),J=1,N)                           CWR04540
C5502 FORMAT(2F10.5)                                         CWR04350
      RETURN                                                 CWR04460
      END                                                    CWR04470
CTHIS SUBROUTINE COMPUTES THE CONTROLLER TRANSFER FUNCTION   CWR04383
CFOR  THE CBSERVER ROBUSTNESS PROBLEM                        CWR04390
C                                                            CWR04400
      SUBROUTINE CONTRL(M,AX,FX,FKX,FKC,BX,NRA,NCA,NRF,NCF,NRK,NCK,W,  CWR04410
     INRE)                                                   CWR04420
     *COMPLEX*8 H(10,10),AX(10,10),FX(10,10),FKX(10,10),FKCX(10,10),   CWR04430
     *BX(10,10),                                             CWR04440
     IBFX(10,10),SI(10,10),I(10,10),SIA(10,10),AALG(10,10),AALK(10,10) CWR04450
      REAL*4 WA(100),FKC(10,10),BF(10,10)                    CWR04460
      COMPLEX S                                              CWR04470
      S=CMPLX(0.0,W)                                         CWR04480
C     CALL CWRITE(AX,NRA,NCA)                                CWR04490
C     CALL CWRITE(FX,NRF,NCF)                                CWR04500
C     CALL CWRITE(FKX,NRK,NCK)                               CWR04510
      DO 20 J=1,NRA                                          CWR04520
      DO 10 K=1,NCA                                          CWR04530
 1C   SI(J,K)=C.                                             CWR04540
 2C   CONTINUE                                               CWR04550
      DO 12 J=1,NRA                                          CWR04560
      DO 11 K=1,NCA                                          CWR04570
 11   I(J,K)=C.0                                             CWR04580
 12   CONTINUE                                               CWR04590
      DO 20 J=1,NCA                                          CWR04600
 3C   I(J,J)=1.0                                             CWR04610
      DO 31 J=1,NRA                                          CWR04620
      DO 32 K=1,NCA                                          CWR04630
 32   SI(J,J)=S*I(J,J)                                       CWR04640
 31   CONTINUE                                               CWR04650
C     CALL CWFITE(SI,NRA,NCA)                                CWR04660
      DO 50 J=1,NRA                                          CWR04670
      DO 40 K=1,NCA                                          CWR04680
 40   SIA(J,K)=SI(J,K)-AX(J,K)                               CWR04690
 5C   CONTINUE                                               CWR04700
      CALL CWFITE(SIA,NRA,NCA)                               CWR04710
      CALL CFLXCV(FKCX,FKC,NRA,NCA)                          CWR04720
      CALL CMZIML(BF X,BX,FX,NRB,NCF)                        CWR04730
      DO 60 J=1,NRA                                          CWR04740
      DO 70 M=1,NCA                                          CWR04750
 7C   AALG(J,K)=SIA(J,K)+FKCX(J,K)+BFX(J,K)                  CWR04760
      CALL CWFITE(AAUG,NRA,NCA)                              CWR04770
 6C   CONTINUE LECTIC(AAUG,NCA,10,I,NCA,10,0,WA,IER)         CWR04790
                                                             CWR04800
```

154

```
C
      CALL CPMPL(I,FKX,ARA,ACA,ACK,AAUK)       CWR04810
      CALL CMATML(FX,AAUK,NRF,NCF,NCK,H)        CWR04820
      CALL CMFITE(H,NRF,NCK)                    CWR04830
      RETURN                                     CWR04840
      END                                        CWR04850
```

155

```
C**POLE PLACEMENT AND ROBUSTNESS (POPLAR) DESIGN PROGRAM    *       CCNO0010
C**MODIFIED VERSION FOR OBSERVER ROBUSTNESS RECOVERY*               CCNO0020
                                                                   CCNO0030
      THIS PROGRAM IS DESIGNED TO EMPLOY SINGULAR VALUE ANALYSIS AND  CCNO0040
      THE USE OF A OPTIMIZATION ROUTINE TO AID IN POLE PLACEMENT    CCNO0050
      CONTROL SYSTEM DESIGN FOR LINEAR MULTIVARIABLE SYSTEMS.       CCNO0060
      ROBUSTNESS OF THE SYSTEM IS ALSO CONSIDERED BY ESTABLISHING   CCNO0070
      SINGULAR VALUE LEVELS WHICH CORRESPOND TO MULTILOOP GAIN      CCNO0080
      AND PHASE MARGINS DETERMINED FROM THE UNIVERSAL GAIN PHASE    CCNO0090
      DIAGRAM DEVELOPED BY NEWSOM AND MUKHAPADHYAY AT NASA LANGLEY. CCNO0100
                                                                   CCNO0110
      VERNGN ( GORDON , CCR , USN                                  CCNO0120
      VERSION 1.0 , 6-28-84                                        CCNO0130
                                                                   CCNO0140
                                                                   CCNO0150
                                                                   CCNO0160
      THIS FIRST BLOCK SETS DIMENSIONS AND DECLARES REAL AND       CCNO0170
      COMPLEX FUNCTIONS. THE FOLLOWING IS A LIST OF PROGRAM SYMBOLS: CCNO0180
                                                                   CCNO0190
                                                                   CCNO0200
      REAL VARIABLES BY NAME AND BRIEF DESCRIPTION                 CCNO0210
THE SYSTEM UNDER CONSIDERATION IS GIVEN AS:                        CCNO0220
XDOT= A*X + B*U  Y=C*X , FOR STATE OR                              CCNO0230
      OUTPUT FEEDBACK OF THE FORM, U=-FX +R OR U=-FY + R.          CCNO0240
THIS PROGRAM HAS BEEN MODIFIED TO WORK AS A DOUBLE PASS            CCNO0250
      PROGRAM TO RECOVER ROBUSTNESS IN OBSERVER DESIGNS            CCNO0260
      BY USING THE FILTER GAINS AS THE RECOVERY VARIABLE.          CCNO0270
                                                                   CCNO0280
                                                                   CCNO0290
                                                                   CCNO0300
      A  -  THE PLANT A MATRIX (NSTATE X NSTATE)                   CCNO0310
      B  -  THE CONTROL MATRIX (NSTATE X NCONTROL)                 CCNO0320
      C  -  THE PLANT OBSERV. OR OUTPUT MATRIX (NOUTPUT X NSTATE;  CCNO0330
                  FOR STATE FEEDBACK PROBLEMS)                     CCNO0340
      F  -  THE FEEDBACK GAIN MATRIX(NCONTROL X NSTATE;            CCNO0350
                  FOR STATE FEEDBACK PROBLEMS)                     CCNO0360
      FK -  THE ONSERVER GAIN MATRIX FOR FILTER PROBLEMS           CCNO0370
      REALMU - THE INPUT DESIRED REAL POLE LOCATION                CCNO0380
      IMAGMU - THE INPUT DESIRED IMAGINARY POLE LOCATION           CCNO0390
      BF - B*F                                                     CCNO0400
      FC - F*C                                                     CCNO0410
      BFC - B*F*C                                                  CCNO0420
      FKC - FK*C                                                   CCNO0430
      AMBFC - A - B*F*C                                            CCNO0440
      AMFKC - A - FK*C                                             CCNO0450
      AMEF - A - B*F                                               CCNO0460
                                                                   CCNO0470
                                                                   CCNO0480
```

```
C   REIG  - REAL COMPUTED EIGENVALUE OF THE SYSTEM          CCNOC490
C   WK    - PROGRAM WORK SPACE                              CCNOC500
C   WK    - WORK SPACE USED BY SUBROUTINES                  CCNOC510
C   WA    - DIMENSIONED WORK SPACE                          CCNOC520
C   WA    - WORKING SPACE                                   CCNOC530
C   OMEGA - FREQUENCY                                       CCNOC540
C   SVGP1 - SINGULAR VALUE OF (I +G*H)                      CCNOC550
C   SVPI  - SINGULAR VALUE OF ( I + INV(F* G ))             CCNOC560
C   SVPC  - SINGULAR VALUE OF ( I + G* F)                   CCNOC570
C   SVPC  - SINGULAR VALUE OF ( I + INV(G * F))             CCNOC580
C           ONLY OF INTEREST IF NO. OUTPUTS = NO. INPUTS.   CCNOC590
C   SVTG1  - FIRST SINGULAR VALUE CF (I + H*G)              CCNOC600
C   SVTG2  - SECOND SINGULAR VALUE CF ( I + H*G)            CCNOC610
C   SVTGA  - MAXIMUM SINGULAR VALUE CF (I + H*G)            CCNOC620
C   SIGFFG - SQRT(FIRST SING VAL * SECOND SING VAL)         CCNOC630
C   SVGT1  - MIN ADDITIVE OUTPUT SINGULAR VALUE (I + G*H)   CCNOC640
C   SVGT2  - SECOND SINGULAR VALUE                          CCNOC650
C   SVHGM1 - MAX ADDITIVE OUTPUT SINGULAR VALUE             CCNOC660
C   SVHGM2 - MIN MULTIPLICATIVE INPUT SINGULAR VALUE        CCNOC670
C   SVTGMX - SECOND SINGULAR VALUE                          CCNOC680
C   SVTGMX - MAX MULTIPLICATIVE INPUT SINGULAR VALUE        CCNOC690
C   SVTGM1 - MIN MULTIPLICATIVE OUTPUT SINGULAR VALUE       CCNOC700
C   SVGHM2 - SECOND MULTIPLICATIVE VALUE                    CCNOC710
C   SVGHMX - MAX MULTIPLICATIVE OUTPUT SINGULAR VALUE       CCNOC720
C   PHGAC  - SQRT( SVHG1*AVHG2)                             CCNOC730
C   PHGML  - SQRT(SVHGI1*SVG-2)                             CCNOC740
C   PHGML  - SQRT (SVHGM1*SVHGM2)                           CCNOC750
C   PHGMU  - SQRT(SVHGMI*SVHGM2)                            CCNOC760
C   SVHGA  - ADDITIVE SINGULAR VALUE CF I+HG               CCNOC770
C   SVGHA  - ADDITIVE SINGULAR VALUE CF I+GH               CCNOC780
C   SVHGM  - MULTIPLICATIVE SINGULAR VALUE OF I+HG          CCNOC790
C   SVGHM  - MULTIPLICATIVE SINGULAR VALUE OF I+GH          CCNOC800
C*******************************************************   CCNOC810
      REAL*4 A(10,10),B(10,10),C(10,10),F(10,10),MK(5000),IMAGMU(50),   CCNOC820
     1FK(10,10),BF(10,10),AMFK(10,10),FEALMU(50),FKC(10,10),            CCNOC830
     2REIG(50),WH(10,10),WA(100),F(10,10),SVGHM(10),                    CCNOC840
     3OMEGA(500),SVHG1(500),SVHG2(500),SVHGX(500),SVGHM(10),            CCNOC850
     4,SVHGH1(500),SVHGH2(500),SVGHX(500),PHGAC(500),PGHAC(500),        CCNOC860
     5SVHGM2(500),SVHGMX(500),SVHGA(10),SVGHA(10),AMBF(10,10),          CCNOC870
     6SVGHMI(500),SVGHMX(500),PHG/U(500),SVHGM(10),                     CCNOC880
C*******************************************************   CCNOC890
C   THE COMPLEX PARAMETERS USEC IN THE PROGRAM             CCNOC900
C   AX  - COMPLEX A                                        CCNOC910
C   BX  - COMPLEX B                                        CCNOC920
C   CX  - COMPLEX C                                        CCNOC930
C   FX  - COMPLEX F                                        CCNOC940
C   FKX - COMPLEX OBSERVER GAIN K                          CCNOC950
```

```
C*********************************************************************
C
C     EIG  - EIGENVALUE OF A- E*F*C                              CONOC570
C     Z    - EIGENVECTOR OF A- B*F*C                             CONOC580
C     MU   - INPUT DESIRED POLE LOCATION                         CONOC590
C     CMU  - ORDERED VALUE OF MU                                 CCNO1C10
C     CEIG - ORDERED VALUE OF EIG                                CCNO1C20
C     CZ   - ORDERED EIGENVECTOR                                 CCNO1C30
C     XXI  - IDENTITY MATRIX                                     CCNO1C40
C     UI   - INPUT ADDITIVE SINGULAR VECTOR ASSOCIATED WITH SV   CCNO1C50
C     VI   - INPUT ADDITIVE SINGULAR VECTOR ASSOCIATED WITH SV   CCNO1C60
C     UC   - OUTPUT ADDITIVE SINGULAR VECTOR (SVAU)              CONO1070
C     VC   - OUTPUT ADDITIVE SINGULAR VECTOR (SVAU)              CONO1C80
C     UMI  - INPUT MULTIPLICATIVE SINGULAR VECTOR               CONO1090
C     VMI  - INPUT MULTIPLICATIVE SINGULAR VECTOR               CONO1100
C     UMC  - OUTPUT MULTIPLICATIVE SINGULAR VECTOR              CONO1110
C     VMC  - OUTPUT MULTIPLICATIVE SINGULAR VECTOR              CONO1120
C     HG   - CONTROL * CONTROL TRANSFER FUNCTION                CONO1130
C     GH   - PLANT * CONTROL TRANSFER FUNCTION                  CONO1140
C     XIPHG - PLANT TRANSFER FUNCTION (G)                       CONO1150
C     PLAN - PLANT TRANSFER FUNCTION (G)                        CONO1160
C     CCNT - CONTROL TRANSFER FUNCTION (H)                      CONO1170
C     FLTFX - G*F                                               CONO1180
C     XIPGH - I   + G*H                                         CONO1190
C     FGI  - I   + INV (H*G)                                    CONO1200
C     GHI  - I   + INV (G*H)                                    CONO1210
C     XIPHGI - I   + INV (H*G)                                  CONO1220
C     XIPGHI - I   + INV (G*H)                                  CONO1230
C     EIGK - EIGENVALUE OF THE OBSERVER                         CONO1240
C     ZK   - EIGENVECTOR OF THE OBSERVER                        CONO1250
C*********************************************************************CONO1260
      COMPLEX *8EIG(50),Z(10,10),MU(50),CMU(50),CEIG(50),CZ(10,10),AX(10CONO1270
     18,10),BX(10,18),LI(18,10),VI(18,10),FX(10,10),XIPHG(10,10),XI(10,10CONO1280
     28,10),UMI(10,10),VMI(10,10),PLAN(10,10),XXI(18,10),UC(18,10),VC(10,CONO1290
     30),XIFGH(10,10),UMO(10,10),VMC(10,10),X(10,10),XIPGH(10,CONO1300
     410),XIFGHI(10,10),FGI(10,10),HG(10,10),GHI(10,10),ZK(10,10),FKX(1CCONO1310
     5,1C),EIGK(50),CCNT(10,10)                                 CONO1320
C                                                                CONO1330
CSET COMPLEX WORKING VARIABLES                                   CONO1340
C                                                                CONO1350
      COMPLEX ZN,TEMP1,TEMP2                                     CONO1360
C                                                                CONO1370
C*********************************************************************CONO1380
C     SET UP DIMENSIONS FOR OPTIMIZER PARAMETERS AND OTHER FUNCTIONS*CCNO1390
C*********************************************************************CONO1400
C     X  - THE DESIGN VARIABLE FOR THE OPTIMIZER                 CONO1410
C     VLB - DESIGN VARIABLE LOWER BOUND                          CONO1420
C     VUB - DESIGN VARIABLE UPPER BOUND                          CONO1430
C     G  - OPTIMIZER CONSTRAINT VECTOR                           CCNO1440
```

```
C     IOG   - OPTIMIZATION CONSTRAINT TYPE                              *CCNO1450
C     IC    - OPTIMIZER PARAMETER                                       *CCNO1460
C     DF    - OPTIMIZER GRADIENT VECTOR                                 *CCNO1470
C     XA    - OPTIMIZER STORAGE ARRAY                                   *CCNO1480
C     WK1   - OPTIMIZER WORK SPACE                                      *CCNO1490
C     IWK   - OPTIMIZER WORK SPACE                                      *CCNO1500
C     IF    - ARRAY USED TO CHOOSE DESIGN VARIABLES FROM INPUT DATA     *CCNO1510
C     IR    - USED BY SORT ROUTINE                                      *CCNO1520
C     KEY   - USED BY SORT ROUTINE                                      *CCNO1530
C     R     - USED BY SORT ROUTINE                                      *CCNO1540
C     IFK   - ARRAY USED TO CHOOSE DESIGN VARIABLE FOR SECOND           *CCNO1550
C             OPTIMIZER RUN, I.E. OBSERVER GAINS                        *CCNO1560
C     X1    - DESIGN VECTOR FOR SECOND OPTIMIZER PASS                   *CCNO1570
C     G1    - CONSTRAINT VECTOR FOR SECOND OPTIMIZER PASS               *CCNO1580
C     VLE1  - LOWER BOUND FOR PASS 2                                    *CCNO1590
C     VUE1  - UPPER BOUND FOR PASS 2                                    *CCNO1600
C     **********************************************************        *CCNO1610
      DIMENSION X(50),VLE(50),VUB(50),G(50),IOG(50),IOG(50),XA(50       *CCNO1620
     150),WK1(1000),IWK(1000),IF(10,10),XIN(50),IR(50),KEY(50),R(50),IF *CCNO1630
     2K(10,10),X1(50),G1(50),VLB1(50),VUE1(50)                           CCNO1640
      EXTERNAL FCN                                                       CCNO1650
      COMMON                                                             CCNO1660
      DATA NF/50/,NCA/50/,NRWK/10000/,NRIWK/1000/                        CCNO1670
C     **********************************************************        *CCNO1680
C     **********************************************************        *CCNO1690
C                                                                      **CCNO1700
C     DATA FORMAT PREPARATION                                          **CCNO1710
C     CARD 1  KODE; KONTRL; FORMAT (I5,I5)                             **CCNO1720
C             KODE IS USER NUMBER TO IDENTIFY THE DATA RUN             **CCNO1730
C             KONTRL SETS FLAG TO TELL PROGRAM HOW TO HANDLE RUN MAY BE:**CCNO1740
C             0 ENTER OPTIMIZER ON FIRST PASS THROUGH THE PROGRAM      **CCNO1750
C             1 COMPUTE VALUES THEN PROGRAM WITHOUT OPTIMIZER          **CCNO1760
C               ON FIRST PASS THEN OPTIMIZE                            **CCNO1770
C     CARD 2  QR MORE COMPUTE VALUES DO NOT OPTIMIZE                   **CCNO1780
C             WMAX,WI,DELW; FORMAT(3F10.0)                            **CCNO1790
C             WMAX - MAXIMUM FREQUENCY TO BE USED                      **CCNO1800
C             WI  - STARTING FREQUENCY                                 **CCNO1810
C             DELW - FREQUENCY STEP FOR FIRST DECADE THIS STEP IS      **CCNO1820
C             MODIFIED BY PROGRAM AT EACH DECADE SHIFT                 **CCNO1830
C     CARD 3  WT1,WT2,WT3; FORMAT(3F10.0)                             **CCNO1840
C             WT1 WEIGHTING FACTOR TO CAUSE THE OPTIMIZER TO USE THE   **CCNO1850
C                 POLE PLACEMENT PORTION OF THE OBJECTIVE FUNCTION     **CCNO1860
C                 IN THE OPTIMIZATION ROUTINE                          *CCNO1870
C             WT2 WEIGHTING FACTOR WHICH CAUSES THE OPTIMIZER TO       *CCNO1880
C                 CONSIDER THE INPUT SINGULAR VALUE GF (I+FG) IN       *CCNO1890
C                 THE OPTIMIZATION OBJECTIVE FUNCTION                   CCNO1900
C             WT3 WEIGHTING FACTOR WHICH CAUSES THE OPTIMIZER TO        CCNO1910
```

```
C
C
C
C          USE THE OUTPUT SINGULAR VALUE (I+GF) IN THE                    CONO1930
C          OBJECTIVE. IF THE OBJECTIVE NEEDS FURTHER PARAMETER            CONO1940
C          ADDITION THEY MAY BE WEIGHTED IN SIMILAR FASHION               CONO1950
C  CARD 4  SVMINI,SVMINO,RJ,NIDG; FORMAT(3F10.C,I5)                       CONO1960
C          SVMINI - DESIRED SINGULAR VALUE LEVEL FOR THE S.V. OF          CONO1980
C                   (I+FG).FOUND USING UNIVERSAL GAIN AND PHASE CHART     CONO1990
C          SVMINO - OUTPUT VERSION OF SVMINI                             CONO2000
C          RJ - FACTOR TO SET PERCENT POLE PLACEMENT ERROR THE           CONO2010
C               DESIGNER WISHES TO PUT INTO THE OPTIMIZER                 CONO2020
C          NIDG - PARAMETER THAT SET TYPE CONSTRAINT CONSIDERED          CONO2030
C                 SEE ADS MANUAL FOR USE                                  CONO2040
C  CARD 5  EIGX,EIGM, EIRT ; FORMAT(3F10.0)                              CONO2050
C          EIGX - MAXIMUM OBSERVER EIGENVALUE DESIRED                    CONO2060
C          EIGM - MINIMUM OBSERVER EIGENVALUE DESIRED                    CONO2070
C          EIRT - EIGEN VALUE REAL TO IMAG RATIO DESIRED FOR             CONO2080
C                 OBSERVER                                               CONO2090
C  CARD 6  IGRAD,NDV,NCON,ISTRAT,IOPT,ICNEC,IPFINT,INFG                  CONO2100
C          FORMAT(8I5)                                                   CONO2110
C          IGRAD - PARAMETER CONCERNING GRADIENT COMP SEE ADS            CONO2120
C          NDV - NUMBER OF DESIGN VARIABLES                              CONO2130
C          NCON - NUMBER OF CONSTRAINTS                                  CONO2140
C          ISTRAT - OPTIMIZER STRATEGY SEE ADS MANUAL                    CONO2150
C          IOPT - OPTIMIZATION METHOD SEE ADS                           CONO2160
C          ICNED - ONE DIMENSIONAL SEARCH TECHNIQUE SEE ADS             CONO2170
C          IPRINT - ADS PRINT CONTROL PARAMETER SEE ADS                 CONO2180
C          INFC - OPTIMIZER CONTROL PARAMETER SEE ADS                   CONO2190
C  CARD 7  NCOLA,NCOLARCMB,NCCLB,NRGWC,NRGWC,NCCLC,NRGWF,NCOLF,NMU       CONO2200
C          FORMAT(9I2)                                                   CONO2210
C          NUMBER OF ROWS AND COLUMNS OF INPUT MATRICES AND              CONO2220
C          NUMBER OF POLES TO BE INPUT FOR PLACEMENT                     CONO2230
C  CARD 8  A MATRIX; READ BY ROWS AS:                                    CONO2240
C          A(1,1)    A(1,2)                                              CONO2250
C          A(2,1)    A(2,2)                                              CONO2260
C          THERE WILL BE ONE OR MORE CARDS FOR EACH ROW OF A             CONO2270
C          AFTER THE PROPER CARDS ARE READ FOR A THE NEXT MATRIX MUST    CONO2280
C          BE READ IN. THE REMAINDER OF THIS COMMENT WILL REFER TO EACH NEW CONO2290
C          DATA SET AS A CARD. REMEMBER, THERE MAY BE SEVERAL CARDS PER  CONO2300
C          SET SO THE NUMBERS DON'T CORRESPOND DIRECTLY WITH THE DATA    CONO2310
C          LISTED IN THE INPUT FILE.                                     CONO2320
C  CARD 9  B MATRIX; READ ROW BY ROW                                     CONO2330
C  CARD 10 C MATRIX; READ ROW BY ROW                                     CONO2340
C  CARD 11 F MATRIX (OBSERVER GUESS); READ ROW BY ROW                    CONO2350
C  CARD 12 FK MATRIX; ROW BY ROW                                         CONO2360
C  CARD 13 IF MATRIX FORMAT(8I2) THIS MATRIX DEFINES THE DESIGN VARIABLE CONO2370
C          FCRMAT(8I2) THIS MATRIX DEFINES THE DESIGN VARIABLE           CONO2380
C          TO THE OPTIMIZER. THE ENTRY WOULD BE OF THE FORM:             CONO2390
C                                                                        CONO2400
```

160

```
C          1   2
C          0   3
C      THIS WOULD TELL THE OPTIMIZER TO MAKE F(1) THE FIRST          CCNO2410
C      DESIGN VARIABLE (X(1)) AND SO FORTH. THE ZERO TELLS THE       CCNO2420
C      OPTIMIZER THAT F(2,1) (IN THIS CASE) IS NOT A DESIGN          CCNO2430
C      VARIABLE AND IS NOT TO BE CHANGED.                            CCNO2440
C      IFK THE DESIGN VARIABLE SPECIFICATION FOR THE                 CCNO2450
C      OBSERVER DONE AS ABOVE                                        CCNO2460
C CARD 14  VLE,VUBi FORMAT(2F10.0)                                   CCNO2470
C      THERE WILL BE ONE CARD FOR EACH DESIGN VARIABLE               CCNO2480
C CARD 15  REALMU,IMAGMU; FORMAT(2F10.0)                             CCNO2490
C      INPUT REAL AND IMAGINARY DESIRED POLE LOCATION                CCNO2500
C      ONE POLE PER CARD                                             CCNO2510
C CARD 16  IGRAD,NDV,NCON,ISTRAT,IOPT,IONED,IPRINT,INFO              CCNO2520
C      SAME FORMAT AS PREVIOUS CARD BUT FOR SECOND OPTIMIZER         CCNO2530
C CARD 17  VLBI,VUBI; FORMAT(2F10.0)*                                CCNO2540
C      THERE WILL BE ONE CARD FOR EACH DESIGN VARIABLE               CCNO2550
C CARD 18                                                            CCNO2560
C ************************************************************        CCNO2570
C ************************************************************        CCNO2580
C ************************************************************        CCNO2590
C                                                                    CCNO2600
C ************************************************************        CCNO2610
C ************************************************************        CCNO2620
C      INPUT DATA INTO THE PROGRAM TO SET UP THE SYSTEM              CCNO2630
C      XCCT=AX+BU ; Y=CX                                             CCNO2640
C ************************************************************        CCNO2650
C ************************************************************        CCNO2660
C                                                                    CCNO2670
C      INPUT RUN CODE AND PROGRAM CONTROL                            CCNO2680
       READ (1,1040) KCCE,KONTRL                                     CCNO2690
C                                                                    CCNO2700
C      INPUT MAXN, INITIAL FREQUENCY,DELTA FREQ, OPTIMIZER WEIGHTS   CCNO2710
C                                                                    CCNO2720
       READ (1,1000) WMAX,WI,DELW                                    CCNO2730
       READ (1,780) WT1,WT2,WT3                                      CCNO2740
       READ (1,750) SVMINI,SVWINO,RJ,NIDG                            CCNO2750
       READ (1,780) EIGX,EIGM,EIRF                                   CCNO2760
       W=WI                                                          CCNO2770
       NC=1                                                          CCNO2780
C                                                                    CCNO2790
C      INPUT OPTIMIZER INPUT DATA (SEE ADS MANUAL)                   CCNO2800
C                                                                    CCNO2810
       READ (1,1050) IGRAC,NCV,NCCN,ISTRAT,IOPT,IONED,IPRINT,INFO    CCNO2820
C                                                                    CCNO2830
C      INPUT THE PARAMETER VALUES AND THE MATRICES A,B,C,F,FK        CCNO2840
C                                                                    CCNO2850
       READ (1,1060) NRCWA,NCCLA,NRCWE,NCCLE,NFCWC,NCCLC,NROWF,NCCLF,NMU CCNO2860
       READ (1,800) NRCWK,NCCLK                                      CCNO2870
       CALL REAC (A,NRGWA,NCCLA)                                     CCNO2880
       CALL REAC (B,NROWB,NCGLB)
```

161

```
      CALL READ (C,NRONC,NCOLC)
      CALL READ (F,NROWF,NCOLF)
      CALL READ (FK,NROWK,NCOLK)
C
C     SPECIFY THE DESIGN VARIABLES WITHIN THE F MATRIX
C
      DO 10 J=1,NROWF
      READ (1,1070) (IF(J,K),K=1,NCOLF)
   10 DO 20 J=1,NROWK
      READ (1,810) (IFK(J,K),K=1,NCOLK)
   20
C
C     SET THE DESIGN VARIABLE BOUNDS
C
      DO 30 J=1,NCV
      READ (1,1080) VLB(J),VUB(J)
   30 DO 50 K=1,NCOLF
      IF (IF(J,K).EQ.0) GO TO 40
      KK=IF(J,K)
      X(KK)=F(J,K)
   40 CONTINUE
   50 CONTINUE
   60 CONTINUE
C
C     SPEC DESIGN VAR WITHIN THE FK MATRIX
C
      READ THE DESIRED EIGENVALUES AS REAL(X)+REAL(Y)*I
      DO 70 I=1,NMU
      READ (1,115C) REALML(I),IMAGMU(I)
   70 DO 80 I=1,NMU
      MU(I)=CMPLX(REALMU(I),IMAGMU(I))
   80 CONTINUE
C
C     SORT THE INPUT EIGEN VALUES
C
      DO 90 I=1,NMU
      IR(I)=I
   90 CALL VSFIR (REALMU,NMU,IR)
      DO 100 J=1,NMU
      K=IR(J)
  100 OMU(J)=MU(K)
      JJ=NMU-1
      DO 120 J=1,JJ
      IF (REAL(CMU(J)).NE.REAL(OMU(J+1))) GO TO 110
      IF (AIMAG(CMU(J)).LT.AIMAG(OMU(J+1))) GO TO 110
      TEMP1=CMU(J)
      CMU(J)=CMU(J+1)
```

CCN02490
CCN02500
CCN02510
CCN02520
CCN02530
CCN02540
CCN02550
CCN02560
CCN02570
CCN02580
CCN02590
CCN03000
CCN03010
CCN03020
CCN03030
CCN03040
CCN03050
CCN03060
CCN03070
CCN03080
CCN03090
CCN03100
CCN03110
CCN03120
CCN03130
CCN03140
CCN03150
CCN03160
CCN03170
CCN03180
CCN03190
CCN03200
CCN03210
CCN03220
CCN03230
CCN03240
CCN03250
CCN03260
CCN03270
CCN03280
CCN03290
CCN03300
CCN03310
CCN03320
CCN03330
CCN03340
CCN03350
CCN03360

```
      CML(J+1)=TEMP1                                               CONO0370
  110 CONTINUE                                                     CONO0380
  120 CONTINUE                                                     CONO0390
C *************************************************************     CONO0400
C     WRITE THE INPUT DATA FOR CHECKING AND REFERENCE             CONO0420
C *************************************************************     CONO0430
      WRITE (6,1050) KODE                                          CONO0440
      WRITE (6,1010) HMAX,HWI,DELH                                 CONO0450
      WRITE (6,830) HT1,HT2,HT3                                    CONO0460
      WRITE (6,840) SVMINI,SVMING,RJ,NIDG                          CONO0470
      WRITE (6,820) EIGX,EIGM,EIRT                                 CONO0480
      WRITE (3,850) NRCWA,NML                                      CONO0490
      CALL WRITE (A,NRCWA,NCCLA)                                   CONO0500
      WRITE (6,1110) NRCWB,NCCLB)                                  CONO0510
      CALL WRITE (B,NRCWB,NCCLB)                                   CONO0520
      WRITE (6,1120) NRCWC,NCCLC)                                  CONO0530
      CALL WRITE (C,NRCWC,NCCLC)                                   CONO0540
      WRITE (6,1130) NRCWF,NCCLF)                                  CONO0550
      CALL WRITE (F,NRCWF,NCCLF)                                   CONO0560
      WRITE (6,860)                                                CONO0570
      CALL WRITE (FK,NROWK,NCCLK)                                  CONO0580
      WRITE (6,1140)                                               CONO0590
      CALL CVE(WR (OMU,NML)                                        CONO0600
      IF (KCNTRL.GT.0) GO TO 140                                   CONO0610
C *************************************************************     CONO0620
C *************************************************************     CONO0630
C     CALL THE ADS OPTIMIZER TO SOLVE FOR THE BEST F'S            CONO0640
C *************************************************************     CONO0650
C *************************************************************     CONO0660
      INFO=-2                                                      CONO0670
      CALL ADS (INFO,ISTRAT,IOPT,IONED,IPRINT,IGRAD,NDV,NCON,X,VLB,VUB,CONO0680
     1BJ,G,IC,NGT,IC,DF,XA,NRA,NCA,WKI,NRWK,IWK,NRIWK)             CONO0690
      IWK(2)=1                                                     CONO0700
      CALL ADS (INFO,ISTRAT,IOPT,IONED,IPRINT,IGRAD,NDV,NCON,X,VLB,VUB,CONO0710
     1BJ,G,IC,NGT,IC,DF,XA,NRA,NCA,WKI,NRWK,IWK,NRIWK)             CONO0720
  130 IF (INFO.EQ.0) GO TO 310                                     CONO0740
      IF (INFO.GT.1) GO TO 300                                     CONO0750
C *************************************************************     CONO0770
C     THIS PORTION OF THE PROGRAM DOES THE ACTUAL COMPUTATION     CONO0780
C *************************************************************     CONO0790
C *************************************************************     CONO0800
C     EQUATE DESIGN VECTOR X WITH THE DESIRED SYSTEM MATRIX       CONO0820
  140 DO 170 J=1,NROWF                                             CONO0840
```

```
      DO 160 K=1,NCOLF
      IF(IF(J,K).EQ.0) GO TO 150
      KK=IF(J,K)
      F(J,K)=X(KK)
150   CONTINUE
160   CONTINUE
170   CONTINUE
C
C     COMPUTE THE SYSTEM MATRIX FOR EIGEN COMPUTATION
      CALL MPLL (B,F,NROWB,NCOLB,NCOLF,BF)
      DO 190 I=1,NROWA
      DO 180 J=1,NCOLA
      AMBF(I,J)=A(I,J)-BF(I,J)
180   CONTINUE
190   CONTINUE
C
C     COMPUTE AND SORT THE EIGENVALUES AND VECTORS
      CALL EIGRF (AMBF,NROWA,10,2,EIG,Z,10,WK,IER)
      DO 200 J=1,NROWA
      REIG(J)=REAL(EIG(J))
200   CONTINUE
      DO 210 J=1,NROWA
      KEY(J)=J
210   CONTINUE
      CALL VSRTR (REIG,NROWA,KEY)
      DO 230 J=1,NCOLA
      K=KEY(J)
      GEIG(J)=EIG(K)
      DO 220 KK=1,NRCWA
      CZ(KK,J)=Z(KK,K)
220   CONTINUE
230   CONTINUE
      JJ=NCOLA-1
      DO 260 J=1,JJ
      IF (REAL(CEIG(J)).NE.REAL(QEIG(J+1))) GO TO 250
      IF (AIMAG(GEIG(J)).LT.AIMAG(QEIG(J+1))) GO TO 250
      TEMP1=CEIG(J)
      CEIG(J)=CEIG(J+1)
      CEIG(J+1)=TEMP1
      DO 240 KK=1,NROWA
      TEMP2=CZ(KK,J)
      CZ(KK,J)=CZ(KK,J+1)
      CZ(KK,J+1)=TEMP2
240   CONTINUE
250   CONTINUE
260   CONTINUE
C
      CUST=0.
```

CCNO3850
CCNO3860
CCNO3870
CCNO3880
CCNO3890
CCNO3900
CCNO3910
CCNO3920
CCNO3930
CCNO3940
CCNO3950
CCNO3960
CCNO3970
CCNO3980
CCNO3990
CCNO4000
CONO4010
CCNO4020
CCNO4030
CCA34040
CUNO4050
CCNO4060
CCNO4070
CCNO4080
CCNO4090
CONO4100
CCNO4110
CCNO4120
CCNO4130
CCNO4140
CCNO4150
CCNO4160
CCNO4170
CCNO4180
CCNO4190
CONO4200
CCNO4210
CCNO4220
CCNO4230
CCNO4240
CCNO4250
CCNO4260
CCNO4270
CCNO4280
CCNO4290
CCNO4300
CCNO4310
CONO4320

164

```
      DO 270 J=1,NMU
      COST=CCST+((REAL(OML(J))-REAL(CEIG(J)))**2+(AIMAG(OMU(J))-AIMAG(OE  CONO04330
     1IG(J)))**2)                                                          CCNO04340
270   CONTINUE                                                             CONO04350
      CBJ=WTI*COST                                                         CONO04360
      IF (NCCA.EC.0) GO TO 290                                             CONO04370
      DO 280 J=1,NMU                                                       CONO04380
      IDG(J)=NIDG                                                          CONO04390
C     R(J)=RJ                                                              CONO04400
      R(J)=RJ*SQRT((REAL(CML(J))-REAL(CEIG(J)))**2+(AIMAG(OMU(J))-AIMAG(   CONO04410
     10EIG(J))**2)                                                         CONO04420
280   IG(J)=SQFT((FEAL(CMU(J))-REAL(OEIG(J)))**2+(AIMAG(OMU(J))-AIMAG(CEI  CONO04430
     1G(J))**2)-R(J)                                                       CONO04440
290   IF (KCNTRL.GT.0) GG TO 310                                           CONO04450
      GO TO 120                                                            CONO04460
300   CONTINUE                                                             CONO04470
C     THE GRADIENT CCMPUTATION ROUTINE GOES HERE IF NEEDED                 CCNO04480
C                                                                          CONO04490
310   CONTINUE                                                             CONO04500
      WRITE (6,1160)                                                       CONO04510
      CALL CVECHR (OEIG,NCOLA)                                             CCNO04520
      DO 320 J=1,NMU                                                       CONO04530
      WRITE (6,910)  CEIG(J),EIGK(J)                                       CONO04540
320   WRITE (6,1170)                                                       CONO04550
      CALL CWRITE (OZ,NROWA,NCOLA)                                         CONO04560
      WRITE (6,1160)                                                       CONO04570
      CALL WRITE (F,NRCWF,NCCLF)                                           CONO04580
      REAL (1,1050) IGRAC,NEV,NCCN,ISTRAT,IOPT,IONED,IPKINT,INFO           CONO04590
      DO 330 J=1,NDV                                                       CCNO04600
330   REAL (1,108CJ) VLB1(J),VUB1(J)                                       CGNO04610
C                                                                          CCNO04620
      DO 360 J=1,NRDWK                                                     CCNO04630
      DO 350 K=1,NCOLK                                                     CONO04640
      IF (IFK(J,K).EQ-0) GO TC 340                                         CCNO04650
      KKK=IFK(J,K)+KK                                                      CONO04660
      X1(KKK)=FK(J,K)                                                      CONO04670
340   CONTINUE                                                             CONO04680
350   CONTINUE                                                             CONO04690
360   CONTINUE                                                             CCNO04700
      IF (KONTRL.GT.0) GC TO 141                                           CCNO04710
      INFC=-2                                                              CCNO04720
C     **************************************************                   CCNO04730
C     *         CALL ADS ROUTINE TO FIND FILTER FEEDBACK GAINS FCR    *    CCNO04740
C     *         THE ROBUSTNESS RECOVERY WITH THE OBSERVER GAINS        *   CCNO04750
C     **************************************************                   CCNO04760
      CALL ADS (INFO,ISTRAT,IOPT,IONED,IFRINT,IGRAD,NDV,NCUN,X1,VLB1,VUB   CCNO04770
     1,CEJ1,C1,IG,NGT,IC,DF,XA,NRA,NCA,NRWK,WK1,NRIWK,IWK,NRIWK)           CONO04780
      IWK(2)=1                                                             CONO04790
                                                                           CONO04800
```

```
 370  CALL ACS (INFO,ISTRAT,IOPT,IONED,IPRINT,IGRAD,NDV,NCON,X1,VLB1,VUB  CCNO4810
     1I,CBJ1,G1,LCG,NGT,IC,DF,XA,NRA,NCA,WK1,NRWK,IWK,NRIWK)               CCNO4820
      IF (INFC.EQ.0) GC TO 700                                            CCNO4830
      IF (INFC.GT.1) GC TO 690                                            CCNO4840
C                                                                         CCNO4850
C                                                                         CCNO4860
 141  CONTINUE                                                            CCNO4870
      DO 400 J=1,NROWK                                                    CCNO4880
      DO 390 K=1,NCOLK                                                    CCNO4890
      IF (IFK(J,K).EQ.0) GO TO 380                                        CCNO4900
      KKKK=IFK(J,K)                                                       CCNO4910
      FK(J,KK)=X1(KKKK)                                                   CCNO4920
 380  CONTINUE                                                            CCNO4930
 390  CONTINUE                                                            CCNO4940
 400  CONTINUE                                                            CCNO4950
      CALL MMUL (FK,C,NROWK,NCOLK,NCOLC,FKC)                              CCNO4960
      DO 420 I=1,NROWA                                                    CCNO4970
      DO 410 J=1,NCOLA                                                    CCNO4980
      AMFKC(I,J)=A(I,J)-FKC(I,J)                                          CCNO4990
 410                                                                      CCNO5000
 420  CONTINUE                                                            CCNO5010
      CALL EIGRF (AMFKC,NROWA,10,2,EIGK,ZK,10,WK,IER)                     CCNO5020
C                                                                         CCNO5030
C     COMPLEX ALL THE INPUT SYSTEM MATRICES FOR THE TRANSFER FUNCTION     CCNO5040
C                                                                         CCNO5050
      CALL CFLXCV (AX,A,NRCWA,NCCLA)                                      CCNO5060
      CALL CFLXCV (BX,B,NROWB,NCCLB)                                      CCNO5070
      CALL CFLXCV (CX,C,NRCWC,NCCLC)                                      CCNO5080
      CALL CFLXCV (FX,F,NROWF,NCCLF)                                      CCNO5090
      CALL CFLXCV (FKX,FK,NROWK,NCOLK)                                    CCNO5100
C                                                                         CCNO5110
C     CALL THE PLANT TRANSFER MATRIX ROUTINE AND FORM THE SYSTEM          CCNO5120
C     RETLRN DIFFERENCE MATRICES AS REQUIRED                             CCNO5130
C                                                                         CCNO5140
      DEL=DELW                                                            CCNO5150
      WII=WI                                                              CCNO5160
      NC=1                                                                CCNO5170
      W=WI                                                                CCNO5180
      SMINAI=C.                                                           CCNO5190
      SMINMI=C.                                                           CCNO5200
      SMINAC=0.                                                           CCNO5210
      SMINMO=C.                                                           CCNO5220
      SMAXAI=C.                                                           CCNO5230
      SMAXMI=0.                                                           CCNO5240
      SMAXAO=C.                                                           CCNO5250
      SMAXMO=C.                                                           CCNO5260
 430  CALL PLANT (PLAN,AX,BX,CX,W,NRCWA,NCCLA,NROWB,NCOLB,NROWC,NCOLC)    CCNO5270
      CALL CONTRP (CONT,AX,FX,FKX,FKC,BF,NROWA,NCCLA,NROWF,NCOLF,NROWK,NC CCNO5280
```

166

```
      1COLK,K)
      CALL CMATML (CCNT,PLAN,NROWF,NCOLK,NCOLE,HG)
      CALL CMATML (PLAN,CCNT,NROWC,NCOLB,NCOLK,GH)
      DO 450 K=1,NROWF
      XI(J,K)=0.0
440   CONTINUE
450   CONTINUE
      DO 460 J=1,NROWF
      XI(J,J)=1.0
460   DO 480 J=1,NROWC
      DO 470 K=1,NCOLK
      XXI(J,K)=0.0
470   CONTINUE
480   CONTINUE
      DO 490 J=1,NCOLK
      XXI(J,J)=1.0
490   DO 500 K=1,NROWF
      DO 500 K=1,NCOLB
      XIPHG(J,K)=HG(J,K)
C500  XIPHG(J,K)=XI(J,K)+HG(J,K)
510   CONTINUE
      DO 530 J=1,AROWC
      DO 520 K=1,NCOLK
      XIFGH(J,K)=XXI(J,K)+Gh(J,K)
520   CONTINUE
530   CONTINUE
      DO 550 K=1,AROWF
      DO 540 K=1,NROWF
      HGI(J,K)=0.0
540   HGI(J,K)=1.0
550   CONTINUE
      DO 570 K=1,AROWC
      DO 560 K=1,AROWC
      GHI(J,K)=0.0
560   GHI(J,K)=1.
570   CONTINUE
      CALL LEGTIC (HG,NROWF,10,HGI,NROWF,10,0,WA,IER)
      CALL LEGTIC (GH,NROWC,10,GHI,NRCWC,10,0,WA,IER)
      DO 590 K=1,NCOLB
      DO 580 K=1,ACOLB
      XIPHGI(J,K)=XI(J,K)+HGI(J,K)
580   CONTINUE
590   CONTINUE
      DO 610 J=1,NROWC
      DO 600 K=1,NROWC
      XIFGHI(J,K)=XXI(J,K)+GhI(J,K)
600   CONTINUE
610   CONTINUE
```

167

```
C
C     CO SINGULAR VALUE CECCMP AND QUANTIFY ALL DESIRED SV'S
C
      CALL CSVC (XIPHG,10,10,NROWF,NRCWF,0,NRCWF,NRGWF,SVHGA,VHGA,U1,V1)    CCNO5770
      CALL CSVC (XIPGH,10,10,NROWC,NRCWC,0,NRCWF,NROWC,SVGHA,VGHA,U0,V0)    CCNO5780
      CALL CSVC (XIPHGI,10,10,NRCWF,NROWC,0,NROWF,NROWC,SVHGM,VHGM,UA1,VM1)  CCNO5790
      CALL CSVC (XIPGHI,10,10,NROWC,NROWC,0,NROWC,NROWC,SVGHM,VGHM,UMC,VMC)  CCNO5800
      OMEGA(AC)=W                                                           CCNO5810
      SVHG1(NC)=SVHGA(NRCWF)                                                CCNO5820
      SVHG2(AC)=SVHGA(NRCWF-1)                                              CCNO5830
      SVHGX(AC)=SVHGA(1)                                                    CCNO5840
      SVGH1(NC)=SVGHA(NRGWC)                                                CCNO5850
      SVGH2(AC)=SVGHA(NRCWC-1)                                              CCNO5860
      SVGHX(NC)=SVGHA(1)                                                    CCNO5870
      SVFGM1(AC)=SVHGM(NROWF)                                               CCNO5880
      SVFGM2(AC)=SVHGM(NRCWF-1)                                             CCNO5890
      SVHGMX(NC)=SVHGM(1)                                                   CCNO5900
      SVGFM1(AC)=SVGHM(NRCWC)                                               CCNO5910
      SVGHM2(AC)=SVGHM(NRCWC-1)                                             CCNO5920
      SVGHMX(AC)=SVGHM(1)                                                   CCNO5930
      PHGAD(AC)=SCRT(SVHG1(NC)*SVHG2(NC))                                   CCNO5940
      PGHAD(AC)=SCRT(SVGH1(NC)*SVGH2(NC))                                   CCNO5950
      PHGMU(AC)=SCRT(SVHGM1(NC)**SVHGM2(NC))                                CCNO5960
      PGHMU(AC)=SCRT(SVGHM1(NC)**SVGHM2(NC))                                CCNO5970
      SMINAI=(SMINAI+(AMAX1(0.0,SVMIN1-SVHG1(NC)))**2)                      CCNO5980
      SMINAC=(SMINAO+(AMAX1(0.0,SVMINO-SVGH1(NC)))**2)                      CCNO5990
C                                                                          CCNO6000
C     CCMPUTE LCCP FCR ALL DESIRED FREQUENCY BANCWIDTH                     CCNO6010
C                                                                          CCNO6020
      W=W+DEL                                                              CCNO6030
      WD=10.-C*WII                                                         CCNO6040
      IF (W.LT.WC) GO TO 620                                               CCNO6050
      DEL=10.-C*DEL                                                        CCNO6060
      WII=10.-C*WII                                                        CCNO6070
620   NC=NC+1                                                              CCNO6080
      IF (W.LE.WMAX) GO TO 420                                             CCNO6090
      MAXA=NC-1                                                            CCNO6100
C                                                                          CCNO6110
C     FCRM QUANTIIES TO USE IN THE COST CRITERIA GF CPTIMIZER              CCNO6120
C                                                                          CCNO6130
      SVMIN=SVFG1(1)                                                       CCNO6140
      DO 630 J=1,MAXN                                                      CCNO6150
      IF (SVGFI(J).GE.SVMIN) GO TO 630                                     CCNO6160
      SVMIN=SVGHI(J)                                                       CCNO6170
630   CONTINLE                                                             CCNO6180
C                                                                          CCNO6190
C     THE CEJECTIVE FUNCTICN IS INSERTED HERE                              CCNO6200
```

```
      OBJ1=HT1*SMINAI+HT3*SMINAO                                       CCN06250
      OBJ=HT2*SMINAI                                                   CCN06260
      IF (NCCA.EQ.0) GC TC 680                                         CCN06270
      DO 640 J=1,NCON                                                  CCN06280
      IDG(J)=NICG                                                      CCN06290
      SUM1=0.0                                                         CCN06300
      DO 650 J=1,NMU                                                   CCN06310
      JJ=J+NPL                                                         CCN06320
      G1(JJ)=SUM1+(AMAX1(0.0,REAL(EIGK(JJ)-EIGX)))**2                  CCN06330
  650 CONTINUE                                                         CCN06340
      SUM2=0.                                                          CCN06350
      DO 660 J=1,NML                                                   CCN06360
      JJ=J+2*NPL                                                       CCN06370
      JJ=J+NPL                                                         CCN06380
      G1(JJ)=SUM2+(AMAX1(0.0,EIGN-REAL(EIGK(JJ))))**2                  CCN06390
  660 CONTINUE                                                         CCN06400
      SUM3=0.                                                          CCN06410
      DO 670 J=1,NMU                                                   CCN06420
      JJ=J+3*NPL                                                       CCN06430
      JJ=J+2*NPL                                                       CCN06440
      G1(JJ)=SLM3+(AMAX1(0.,(AIMAG(EIGK(JJ)/REAL(EIGK(JJ))-EIRT))**2   CCN06450
  670 CONTINUE                                                         CCN06460
  680 IF (KCNTRL.GT.0) GC TC 700                                       CCN06470
      GO TO 710                                                        CCN06480
  690 CONTINUE                                                         CCN06490
C THE GRADIENT COMPUTATION ROUTINE GOES HERE IF NEEDED                 CCN06500
  700 CONTINUE                                                         CCN06510
C                                                                      CCN06520
C OUTPUT THE COMPUTED FINAL EIGEN VALUES AND VECTORS FROM OPT F'S      CCN06530
C                                                                      CCN06540
      IF (KCNTRL.EQ.0) GC TC 710                                       CCN06550
      WRITE (6,870)                                                    CCN06560
      GO TO 720                                                        CCN06570
  710 WRITE (6,880)                                                    CCN06580
  720 WRITE (6,890)                                                    CCN06590
      CALL CVECWR (EIGK,NCOLA)                                         CCN06600
      WRITE (6,990)                                                    CCN06610
      CALL CMFITE (ZK,NRCWA,NCOLA)                                     CCN06620
      WRITE (6,1160)                                                   CCN06630
      CALL CVECWR (OEIG,NCOLA)                                         CCN06640
      DO 730 J=1,NMU                                                   CCN06650
      WRITE (3,SIC) OEIG(J),EIGK(J)                                    CCN06660
      WRITE (6,1170)                                                   CCN06670
      CALL CMFITE (OZ,NRCWA,NCOLA)                                     CCN06680
      WRITE (6,1160)                                                   CCN06690
  730 CALL WFITE (F,NRCWF,NCCLF)                                       CCN06700
      WRITE (6,920)                                                    CCN06710
                                                                      CCN06720
```

169

```
      CALL WRITE (FK,NROWK,NCOLK)                                      CCNO6730
C                                                                      CCNO6740
C  OUTPUT THE CPTIMIZED SINGULAR VALUE DATA                            CCNO6750
C                                                                      CCNO6760
      WRITE (6,1020) MAXN                                              CCNO6770
      WRITE (3,540) MAXN                                               CCNO6780
      CO 740 J=1,MAXN                                                  CCNO6790
      WRITE (3,550) OMEGA(J),SVHG1(J),SVHG1(J),SVHGM1(J),SVHGMI(J)     CCNO6800
      WRITE (6,1030) OMEGA(J),SVHG1(J),SVHG1(J),SVHGM1(J),SVHGMI(J)    CCNO6810
  740 WRITE (4,560)                                                    CCNO6820
      CO 750 J=1,MAXN                                                  CCNO6830
      WRITE (3,980) OMEGA(J),SVHG2(J),SVHG2(J),SVHGM2(J),SVGHM2(J)     CCNO6840
      WRITE (6,550) OMEGA(J),SVHG2(J),SVHG2(J),SVHGM2(J),SVGHM2(J)     CCNO6850
  750 WRITE (4,570)                                                    CCNO6860
      CO 760 J=1,MAXN                                                  CCNO6870
      WRITE (3,980) OMEGA(J),SVHGX(J),SVHGX(J),SVHGMX(J),SVGHMX(J)     CCNO6880
      WRITE (6,990) OMEGA(J),SVHGX(J),SVHGX(J),SVHGMX(J),SVGHMX(J)     CCNO6890
  760 IF (KCNTRL.EQ.0.CR.KCNTRL.GT.1) GO TO 770                        CCNO6900
      KONTRL=C                                                         CCNO6910
      GO TO 150                                                        CCNO6920
C***********************************************************************CCNO6930
C***********************************************************************CCNO6940
  770 STCP                                                             CCNO6950
C***********************************************************************CCNO6960
C      FCRMAT STATEMENTS                                               CCNO6970
C***********************************************************************CCNO6980
  780 FORMAT (3F10.0)                                                  CCNO6990
  790 FORMAT (3F10.0,I5)                                               CCNO7000
  800 FORMAT (2I2)                                                     CCNO7010
  810 FORMAT (I2)                                                      CCNO7020
  820 FORMAT (/,12HOBS MAX EIG=,F10.5,8HMIN EIG=,F10.5,8HEIG RAT=,F10.5,CCNO7030
     1/)                                                               CCNO7040
  830 FORMAT (/,10H WEIGHT 1=,F10.5,9HWEIGHT 2=,F10.5,9HWEIGHT 3=,F10.5,CCNO7050
     1/)                                                               CCNO7060
  840 FORMAT (/,9F SVMINI= ,F10.5,8H SVMINO=,F10.5,5H RJ=,F10.5,6H IDG CCNO7070
     1=,I5,/)                                                          CCNO7080
  850 FORMAT (2I5)                                                     CCNO7090
  860 FORMAT (//,26H     THE INITIAL K MATRIX ,*(/)                    CCNO7100
  870 FORMAT (//,26H *** THE INITIAL DATA IS ***,/)                    CCNO7110
  880 FORMAT (//,41H *** THE CPTIMIZER COMPUTED OUTPUT IS ***,/)       CCNO7120
  890 FORMAT (//,27H    THE CBSERVER EIGENVALUES,/)                    CCNO7130
  900 FORMAT (//,27H THE CBSERVER EIGENVECTORS ,/)                     CCNO7140
  910 FORMAT (4E16.5)                                                  CCNO7150
  920 FORMAT (//,20H    THE K GAIN MATRIX,/)                           CCNO7160
  930 FORMAT (/,24H *** THE OBJECTIVE FUNCTION IS =,F10.5,6H ***,/)    CCNO7170
  940 FORMAT (I5)                                                      CCNO7180
  950 FURMAT (5E12.5)                                                  CCNO7190
  960 FORMAT (//,2X,7H FREQ ,5X,6HSVACI2,4X,6HSVACC2,4X,5HSVMI2,4X,5HSVMCCNO7200
```

```
 102,/)
970   FORMAT  (/,2X,8H    FREG  ,5X,5HSVHGX,4X,5HSVGFX,4X,6HSVHGMX,4X,6HSVG   CCNO07210
1HMX,/)                                                                      CCNO07220
980   FORMAT  (5E16.5)                                                       CCNO07230
990   FORMAT  (5F10.5)                                                       CCNO07240
1000  FORMAT  (3F10.0)                                                       CCNO07250
1010  FORMAT  (/,6H WMAX=,F10.5,1X,10HINIT FREG=,F10.5,1X,10HFREG STEP=,F    CCNO07260
110.5,//)                                                                    CCNO07270
1020  FORMAT  (//,10H FREGUENCY,5X,13HMIN ADD IN SV,4X,13HMIN ADD CT SV,3    CCNO07280
1X,13HMIN MUL IN SV,2X,18HMIN MUL CLT SV )                                   CCNO07290
1030  FORMAT  (5(F10.5,5X))                                                  CCNO07300
1040  FORMAT  (15,15)                                                        CCNO07310
1050  FORMAT  (8I5)                                                          CCNO07320
1060  FORMAT  (8I2)                                                          CCNO07330
1070  FORMAT  (8I2)                                                          CCNO07340
1080  FORMAT  (2F10.0)                                                       CCNO07350
1090  FORMAT  (1H1,//,12H<<<<< RUN #,15,6H >>>>,///)                         CCNO07360
1100  FORMAT  (27H *** THE A PLANT MATRIX ***,///)                           CCNO07370
1110  FORMAT  (///,35H *** THE B CONTRL INPUT MATRIX ***,///)                CCNO07380
1120  FORMAT  (///,38H *** THE C OBSERVATICN MATRIX ***,///)                 CCNO07390
1130  FORMAT  (///,44H *** THE     F FEEDBACK MATRIX ***,///)                CCNO07400
1140  FORMAT  (///,     THE ORDERED COMPLEX EIGENVALUES (INPUT),///)         CCNO07410
1150  FORMAT  (2F10.0)                                                       CCNO07420
1160  FORMAT  (///,27H *** THE ORDERED COMPLTEC EIGENVALUES,///)             CCNO07430
1170  FORMAT  (///,29H *** THE ORDERED EIGENVECTGRS,//)                      CCNO07440
1180  FORMAT  (///,36H *** THE     F FEEDBACK MATRIX ***,///)                CCNO07450
      ENC                                                                    CCNO07460
CTHIS SUEROUTINE COMPUTES THE CONTROLLER TRANSFER FUNCTICN                   CCNO07470
CFOR THE OBSERVER ROBUSTNESS PROBLEM                                         CCNO07480
C                                                                            CCNO07490
      SUBROUTINE CONTRP(F,AX,FX,FKX,FKC,BF,NRA,NCA,NRF,NCF,NRK,NCK,W)        CCNO07500
      COMPLEX*8 H(10,10),AX(10,10),FX(10,10),FX1(10,10),FKCX(10,10),AALK(10,10)  CCNO07520
1BFX(10,10),SI(10,10),FKC(10,10),SIA(10,10),AALG(10,10),FKCX(10,10),AALK(10,10)  CCNO07530
      REAL*4  BA(10,10),FKC(10,10),BF(10,10)                                 CCNO07540
      COMPLEX*8 W                                                            CCNO07550
      S=CMPLX(0.0,W)                                                         CCNO07560
C                                                                            CCNO07570
      CALL CMFITE(AX,NRA,NCA)                                                CCNO07580
C                                                                            CCNO07590
      CALL CMFITE(FX,NRF,NCF)                                                CCNO07600
C                                                                            CCNO07610
      CALL CMFITE(FKX,NRK,NCK)                                               CCNO07620
      CALL WRITE(BF,NRB,NCF)                                                 CCNO07630
      DO 20 J=1,NRA                                                          CCNO07640
10    SI(J,K)=C.                                                             CCNO07650
20    CONTINLE                                                               CCNO07660
      DO 12 J=1,NRA                                                          CCNO07670
      DO 11 K=1,NCA                                                          CCNO07680
11    I(J,K)=0.0
```

171

```
 12   CONTINLE
 3C   DO 30 J=1,NCA
      I(J,J)=1.0
      DO 31 J=1,NRA
      DO 32 K=1,NCA
 32   SI(J,J)=SI(J,J)
 31   CONTINUE
      CALL CWFITE(SI,NRA,NCA)
      DO 50 J=1,NRA
      DO 40 K=1,NCA
 40   SIA(J,K)=SI(J,K)-AX(J,K)
 5C   CONTINLE
      CALL CWFITE(SIA,NRA,NCA)
      CALL CFLXCV(FKCX,FKC,NRA,NCA)
      CALL CFLXCV(BFX,BF,NRA,NCA)
      DO 60 J=1,NRA
      DO 70 K=1,NCA
 7C   AALG(J,K)=SIA(J,K)+FKCX(J,K)+BFX(J,K)
      CALL CWRITE(AAUG,NRA,NCA)
 6C   CONTINLE
      CALL LECTIC(AAUG,NCA,10,1,NCA,10,0,AA,IER)
      CALL CPATML(I,FKX,NRA,NCA,ACK,AAUK)
      CALL CPAIML(FX,AAUK,NRF,NCF,NCK,H)
      CALL CWFITE(H,NRF,NCK)
      RETURN
      ENC
```

CCNO7690
CONO7700
CCGNO7710
CONO7720
CONO7730
CONO7740
CCCNO7750
CCCNO7760
CONO7770
CCGNO7780
CCCNO7790
CCGNO7800
CCGNO7810
CCCNO7820
CCGNO7830
CCGNO7840
CCGNO7850
CCCNO7860
CCCNO7870
CCCNO7880
CCCNO7890
CCCNO7900
CCCNO7910
CONO7920
CCCNO7930
CONO7940

LIST OF REFERENCES

1. Rosenbrock, H. H., Computer-Aided Control Systems Design, Academic Press, 1974.

2. Sandell, N. R., Jr., Recent Developments in the Robustness Theory of Multivariable Systems, Office of Naval Research Report CR-215-271-1F, 1979.

3. Kalman, R. E., " When Is A Linear Control System Optimal?", ASME J. Basic Engr., pp51-60, March 1964.

4. Anderson, B. D. C. and Moore, J. B., Linear Optimal Control, Prentice Hall, Inc., 1971.

5. Rosenbrock, H. H., "Design of Multivariable Control Systems Using the Inverse Nyquist Array", Proc. of IEE, Vol 116, pp 1929-1936, Nov 1969.

6. Doyle, J. C. and Stein, G., " Multivariable Feedback Design: Concepts for a Classical/Modern Synthesis", IEEE Trans. Auto. Control, Vol AC-26, No.1, pp4-16, Feb 1981.

7. Safanov, M. G. and Athans, M., " Gain and Phase Margin of Multiloop LQG Regulators", IEEE Trans. Auto. Control, April 1977.

8. Sandell, N. S., Jr., Lehtomaki, N. A., and Athans, M., " Robustness Results in Linear Quadratic Gaussian Based Multivariable Control DEsigns", IEEE Trans. Auto. Control, Vol 26, No. 1, pp75-92, Feb. 1981.

9. Lehtomaki, N. A., Practical Robustness Measures in Multivariable Control System Analysis, Ph.D. Thesis, Massachusetts Institute of Technology, May 1981.

10. Vanderplaats, G. N., ADS - A Fortran Program for Automated Design Synthesis, Version 1.0, Naval Postgraduate School, Monterey, Ca., May 1984.

11. Vanderplaats, G. N., Numerical Optimization Techniques for Engineering Design: With Applications, McGraw-Hill, 1984.

12. Vanderplaats, G. N., CONMIN-A Fortran Program for Constrained Function Minimization - User Manual, NASA TM X-622282, AUG 1973.

13. Shapiro, E. Y., et. al.," Pole Placement with Output Feedback", AIAA J. of Guidance and Control, Vol 4, No 4, pp 441-443, July-Aug 1981.

14. Mukhopadhyay, V. and Newsom, J., Application of Matrix Singular Value Margins of Multiloop Systems, NASA TM-84524, July 1982.

15. Mukhopadhyay, V. and Newsom, J., " The use of Singular Value Gradients and Optimization Techniques to Design Robust Controllers for Multiloop Systems", AIAA Guidance and Control Papers, August 1983.

16. Sandell, N. R.,Jr., et al, Multivariable Stability Margins for Vehicle Flight Control Systems, TR -121, Alphatech Inc., Dec. 1981.

17. Doyle, J. C. and Stein, G., " Robustness with Observers", IEEE Trans. Auto. Control, Vol AC-24, pp 607-611, Aug. 1979.

174

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22314	2
2.	Professor D. J. Collins, Code 67Co Department of Aeronautics Naval Postgraduate School Monterey, California 93943	5
3.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943	2
4.	Department Chairman, Code 67 Department of Aeronautics Naval Postgraduate School Monterey, California 93943	1
5.	Prof M.F. Platzer, Code 67Pl Department of Aeronautics Naval Postgraduate School Monterey, California 93943	1
6.	Prof T.H. Gawain, Code 67Gn Department of Aeronautics Naval Postgraduate School Monterey, California 93943	1
7.	Prof R.D. Strum, Code 62St Department of Electrical Engineering Naval Postgraduate School Monterey, California 93943	1
8.	Prof G.E. Latta, Code 54Lz Department of Mathematics Naval Postgraduate School Monterey, California 93943	1
9.	Prof G.N. Vanderplats Department of Mechanical and Environmental Engineering University of California Santa Barbara, California 93106	1
9.	Commander Vernon C. Gordon, USN 315 Alameda Blvd. Coronado, California 92118	3

END

FILMED

1-85

DTIC