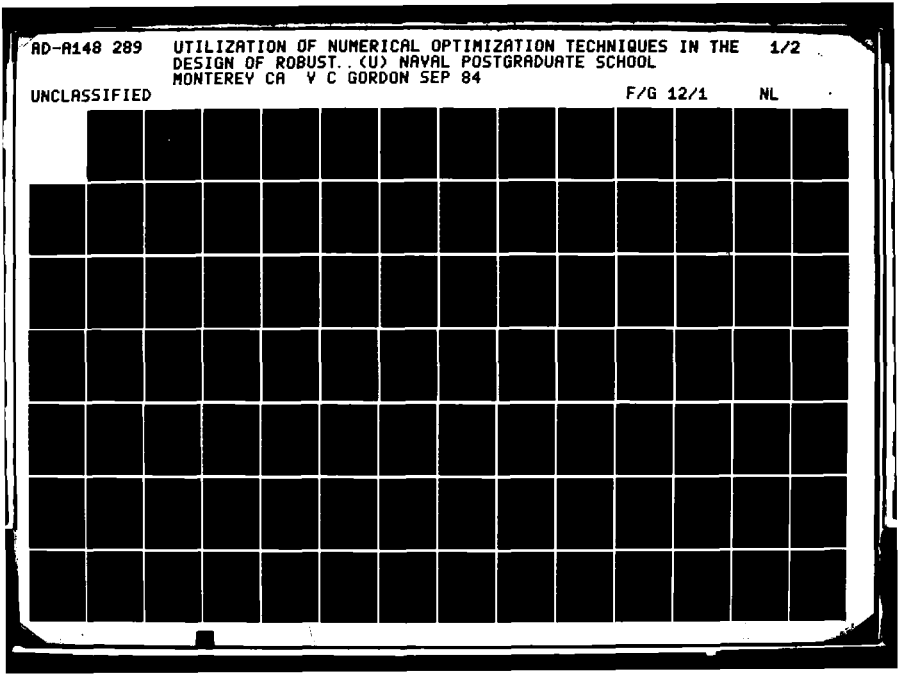
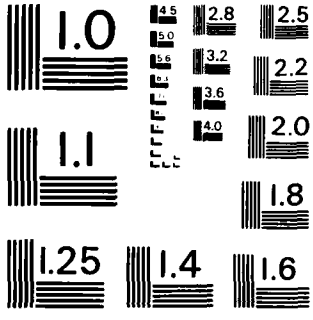
AD-A148 289 UTILIZATION OF NUMERICAL OPTIMIZATION TECHNIQUES IN THE 1/2
DESIGN OF ROBUST..(U) NAVAL POSTGRADUATE SCHOOL
MONTEREY CA Y C GORDON SEP 84

UNCLASSIFIED F/G 12/1 NL





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

AD-A148 289

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

DTIC FILE COPY

Approved for public release; distribution unlimited.

84 12 03 020

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO. AD-A148 284	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.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

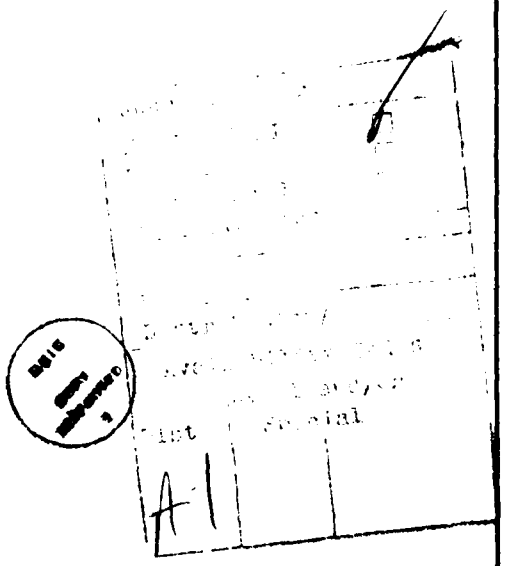
S N 0102-LF-014-6601

UNCLASSIFIED

1 SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

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.



Approved for public release; distribution unlimited.

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: Vernon C Gordon

APPROVED BY:

Max F. Platzer
M. F. Platzer
Professor of Aeronautics

T. H. Gawain
T. H. Gawain
Professor of Aeronautics

Robert D Strum
R. D. Strum
Professor of Electrical
Engineering

G. E. Latta
G. E. Latta
Professor of Mathematics

Daniel J Collins
D. J. Collins
Professor of Aeronautics
Thesis Supervisor

Approved by: Max F. Platzer
Chairman, Department of Aeronautics

Approved by: David A. Schrady
David A. Schrady, Academic Dean

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.

TABLE OF CONTENTS

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 Laptas 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.

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.

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 Bode 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. SINGLE-INPUT SINGLE-OUTPUT SYSTEMS

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 $\alpha(s)$ placed in series with the nominal system. When $\alpha = 1$ the system is nominal and stable. To determine the positive phase margin of the system the value of $\alpha(s) = \alpha(j\omega) = e^{j\varphi}$ will be changed by varying φ until the system just becomes unstable. This value of φ 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

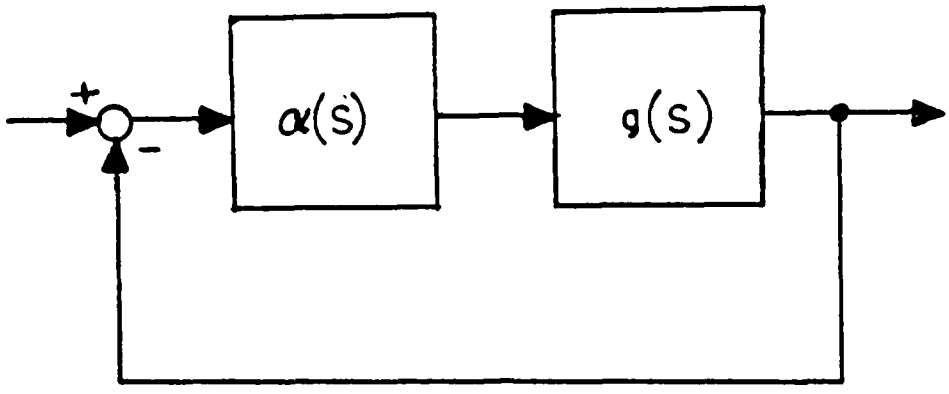


Figure 2.1 Basic Gain and Phase Margin Determination Model.

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

These gain and phase margins may not be adequate measures 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 destabilize the system, small simultaneous changes in gain and phase may destabilize the system. This is not a major difficulty 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 information. 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

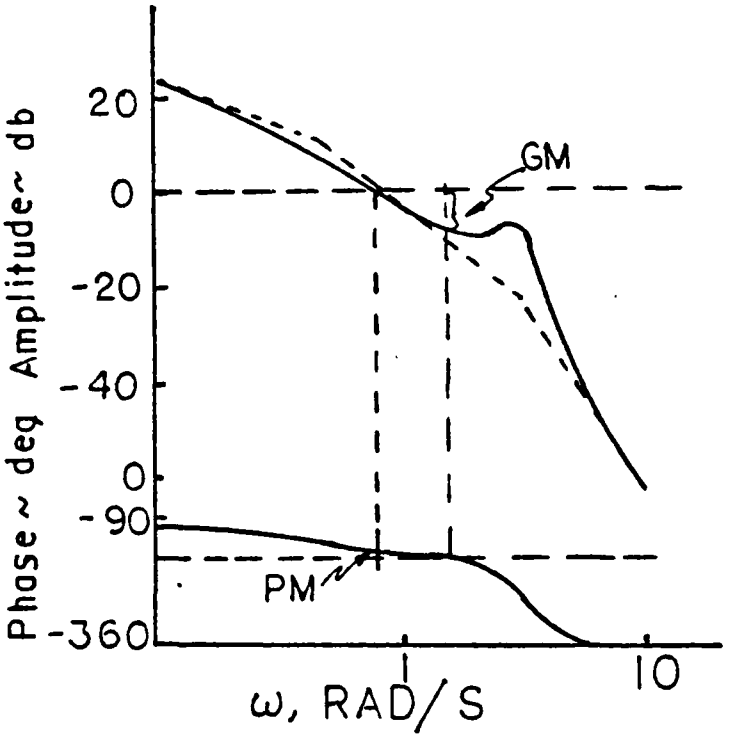


Figure 2.2 Classical Bode Plot.

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 contour. 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

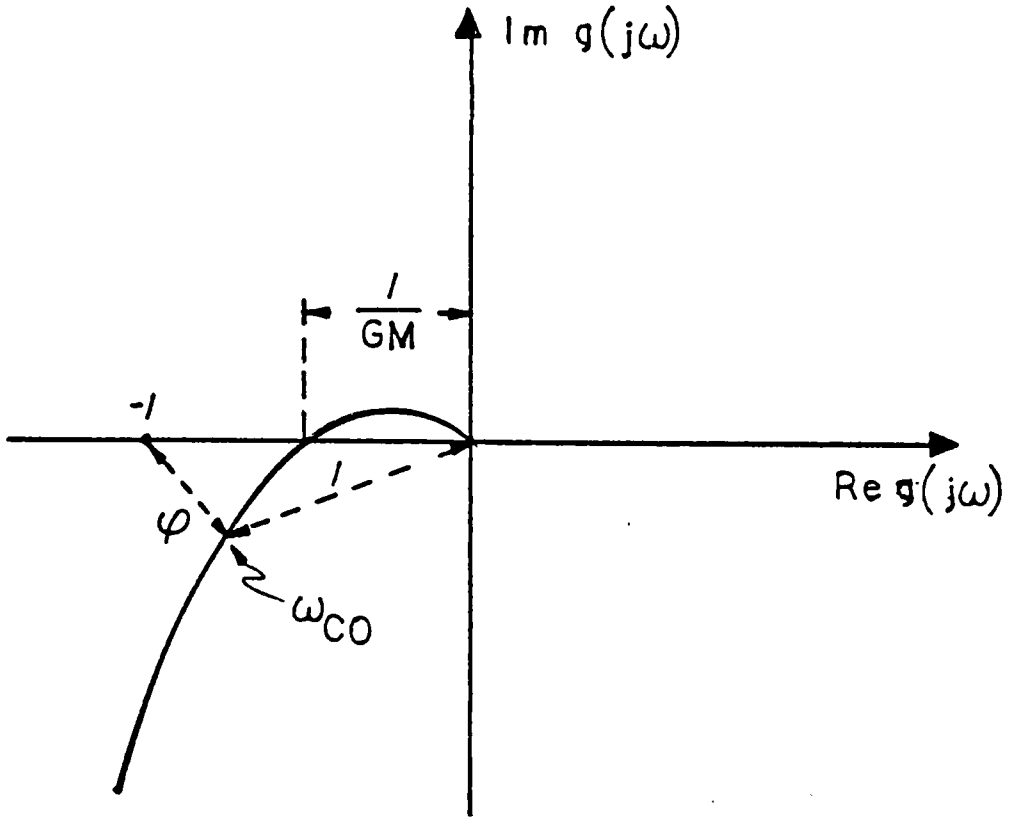


Figure 2.3 Nyquist Plot of Stable System.

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 of a multiplicative perturbation 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 perturbations are also stable the diagram may then be used to determine how near the system is to instability for the given perturbation.

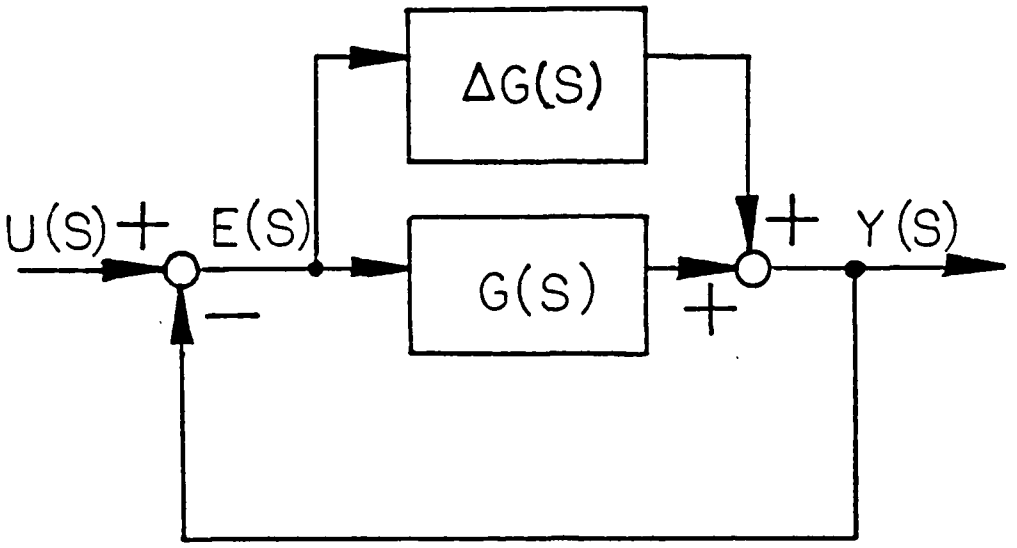


Figure 2.4 Additively Perturbed System.

Since the system is stable the (-1,0) point is encircled the correct number of times by the nominal plant. If the locus of $g(j\omega)$ in the diagram is warped until it passes beyond the (-1,0) point then clearly the number of encirclements of this point will change and the system will become 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.

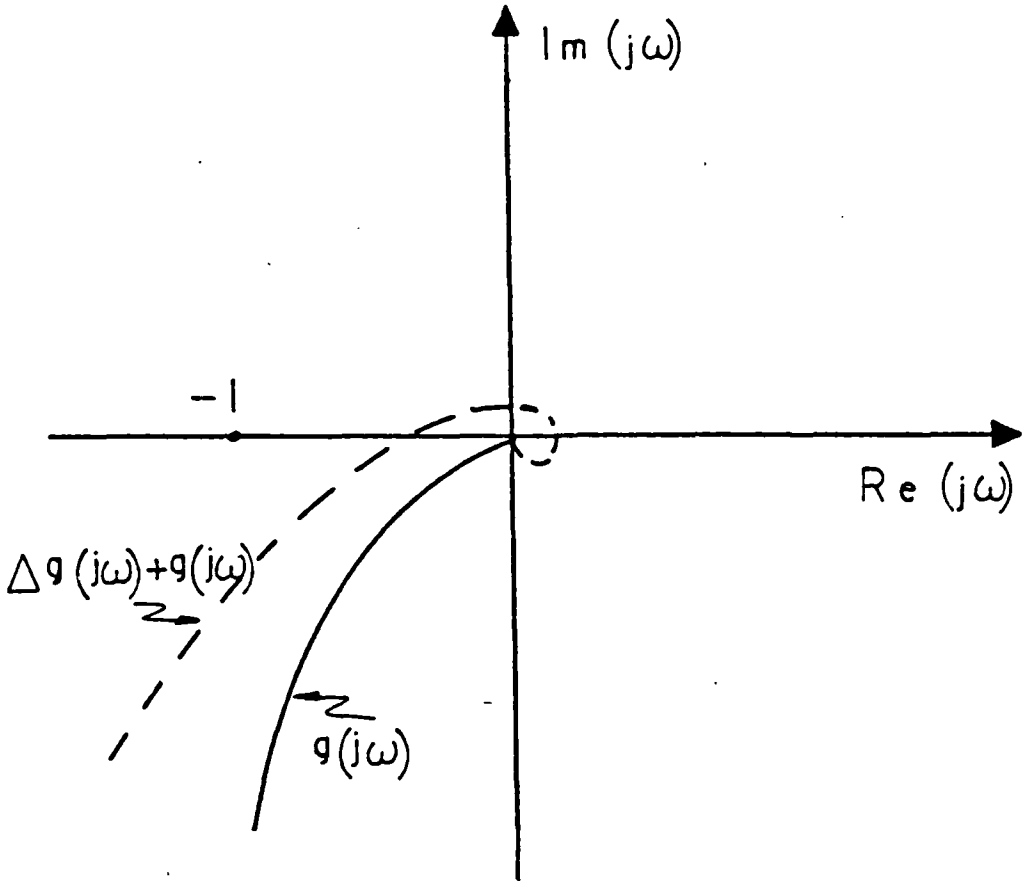


Figure 2.5 Additive Nyquist Plot.

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

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

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

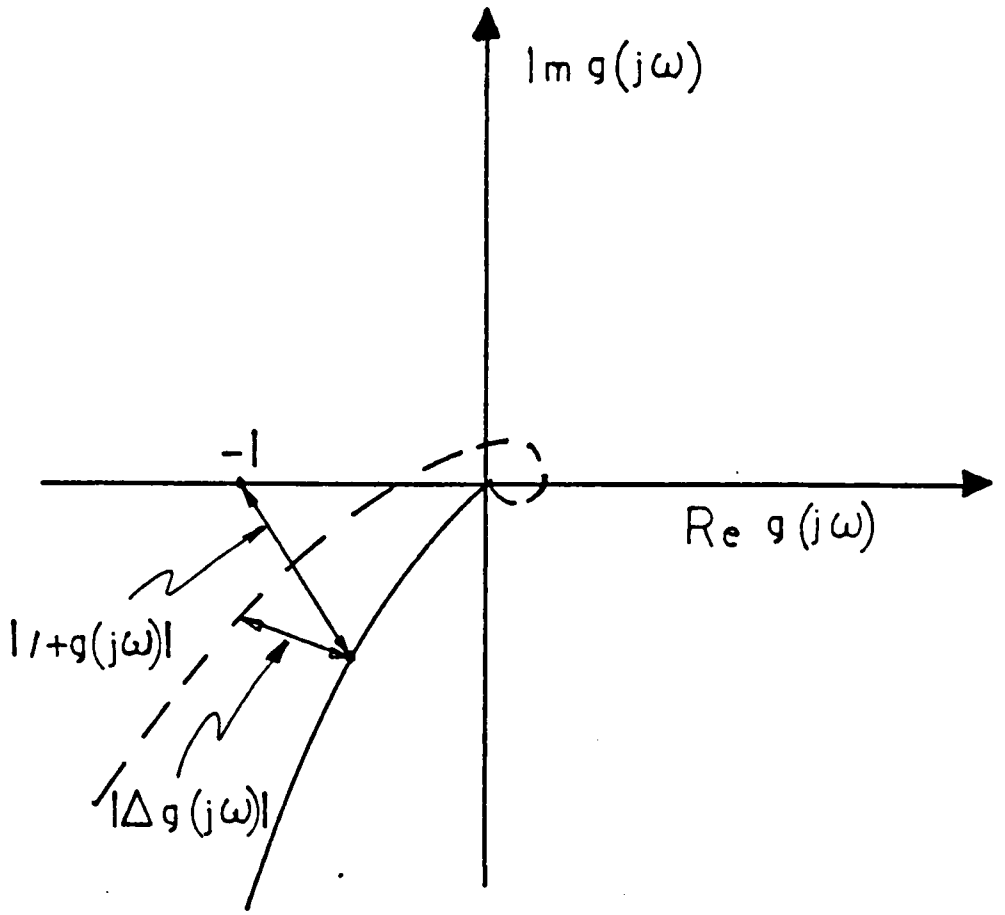


Figure 2.6 Nyquist for Inequality Additive Condition.

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

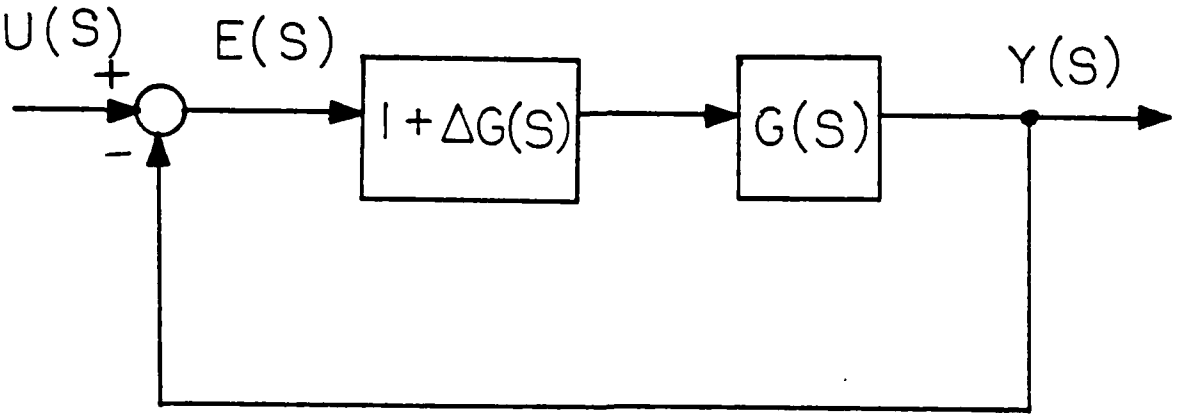


Figure 2.7 Multiplicative System.

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

where the $\underline{x}'\underline{x}$ term and the $\underline{u}'\underline{u}$ term form quadratics. The $\underset{\sim}{Q}$ and $\underset{\sim}{R}$ 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.

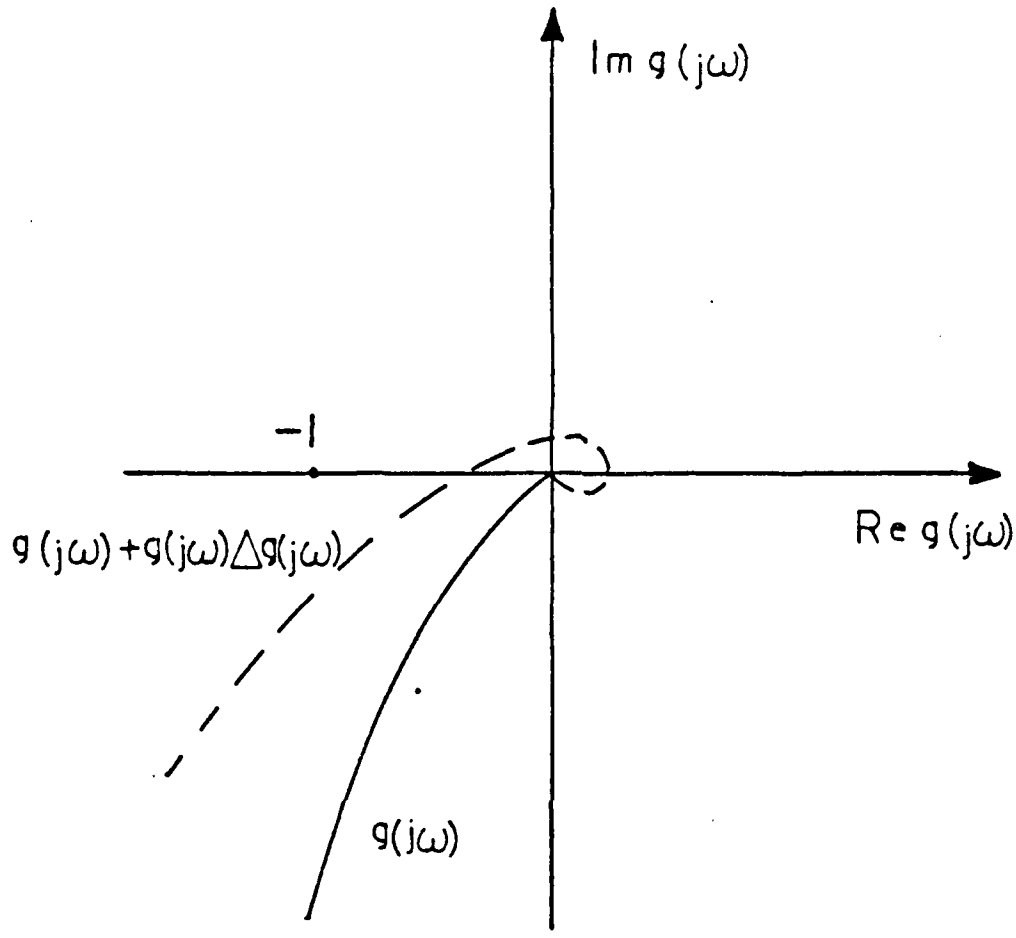


Figure 2.8 Nyquist Plot for Multiplicative System.

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

$$|1 + f'(j\omega I - a)^{-1}b| \geq 1 \qquad (2.4)$$

Writing the Kalman equation as 2.5

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

one 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.

$$|1 + G(s)H(s)| > 1 \qquad (2.7)$$

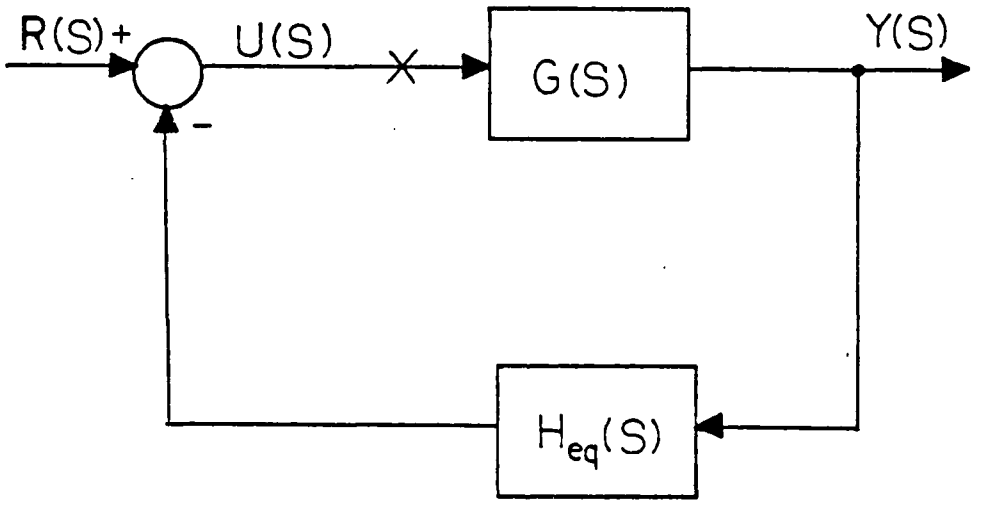


Figure 2.9 Typical Equivalent Feedback System.

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

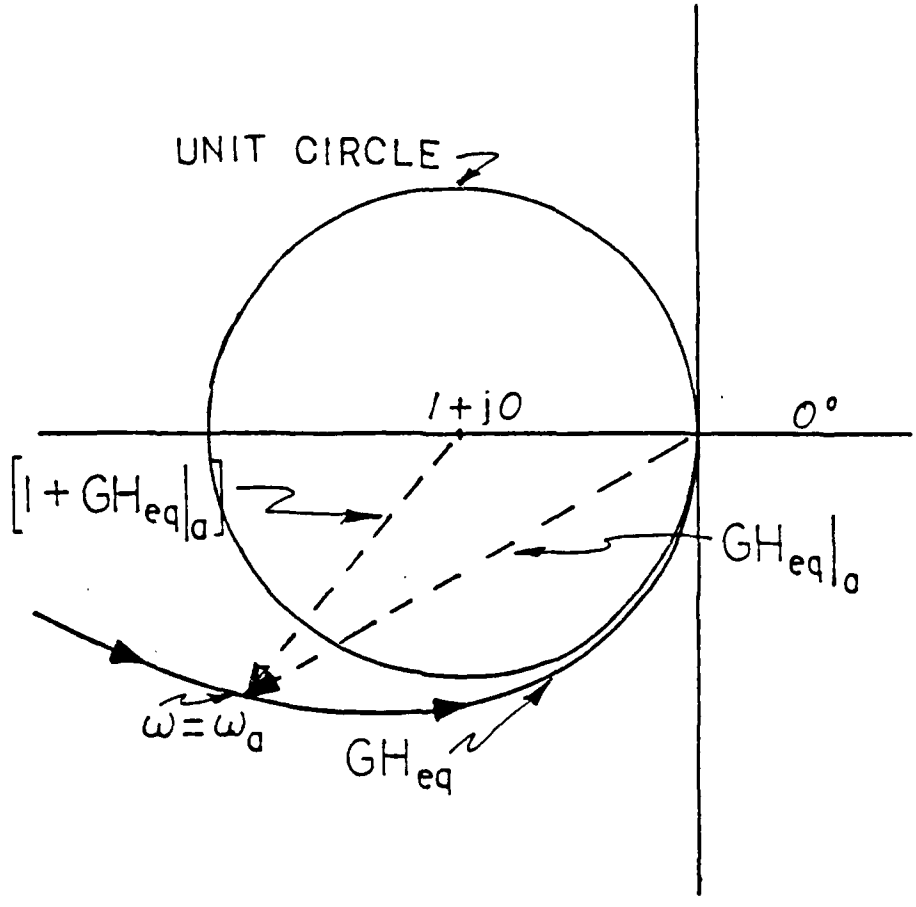


Figure 2.10 Polar Plot of an Optimal System.

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

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

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 $\underset{\sim}{Q}$ and/or $\underset{\sim}{R}$ 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 $\det(\underset{\sim}{I}+\underset{\sim}{G}(j\omega))$ 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

$$f(s) = -1 + \det[\underset{\sim}{I}+\underset{\sim}{G}(s)] = \varphi_{OL}(s)/\varphi_{CL}(s) - 1 \text{ and}$$

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, $\underset{\sim}{I} + \underset{\sim}{G}$, is nearly singular for some frequency. If $\underset{\sim}{I} + \underset{\sim}{G}$ is nearly singular a small change in $\underset{\sim}{G}$ may make $\underset{\sim}{I} + \underset{\sim}{G}$ exactly singular. This causes the $\det(\underset{\sim}{I} + \underset{\sim}{G})$ to become zero and the Nyquist encirclement count to change indicating an unstable system. It is possible for very small changes in $\underset{\sim}{I}+\underset{\sim}{G}$ to produce large changes in the determinant of $\underset{\sim}{I}+\underset{\sim}{G}$. The matrix $\underset{\sim}{I}+\underset{\sim}{G}$

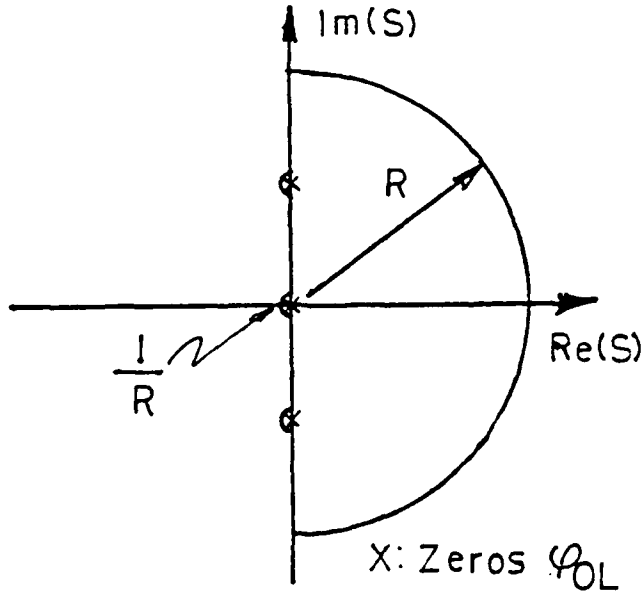


Figure 3.1 Nyquist D Contour.

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

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 $\det(\underset{\sim}{I} + \underset{\sim}{G})$ 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 $\det(\underset{\sim}{I}+\underset{\sim}{G})$ is zero it is important to find how near the return difference matrix $\underset{\sim}{I}+\underset{\sim}{G}$ is to being singular. This nearness to singularity can be interpreted as closeness of the matrix G

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

$$\underline{\sigma}(A) = \min_{i} \left(\lambda_i \left(\underset{\sim}{A}^H \underset{\sim}{A} \right) \right)^{1/2} \qquad (3.1)$$

where $\lambda_i (\underset{\sim}{A}^H \underset{\sim}{A})$ is the eigenvalue of the complex conjugate transpose of $\underset{\sim}{A}$ times $\underset{\sim}{A}$. A basic MIMO linear system is

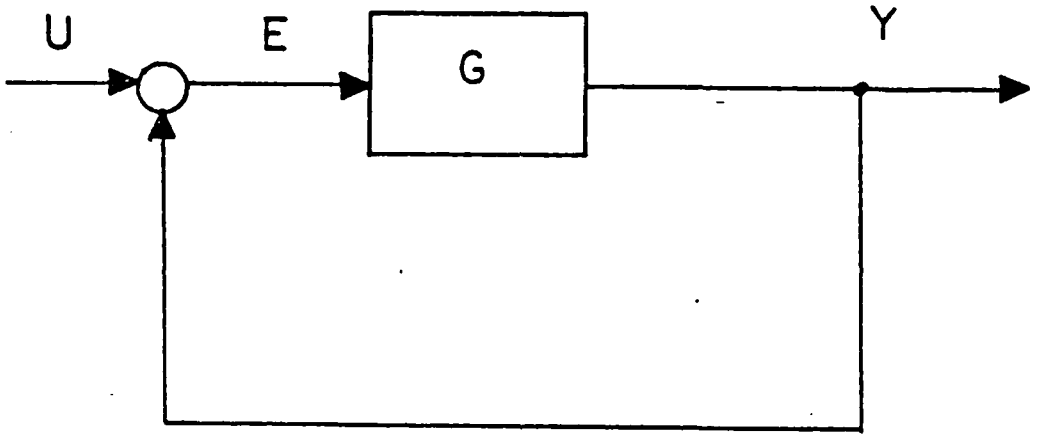


Figure 3.2 Basic Multi-input, Multi-output System.

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

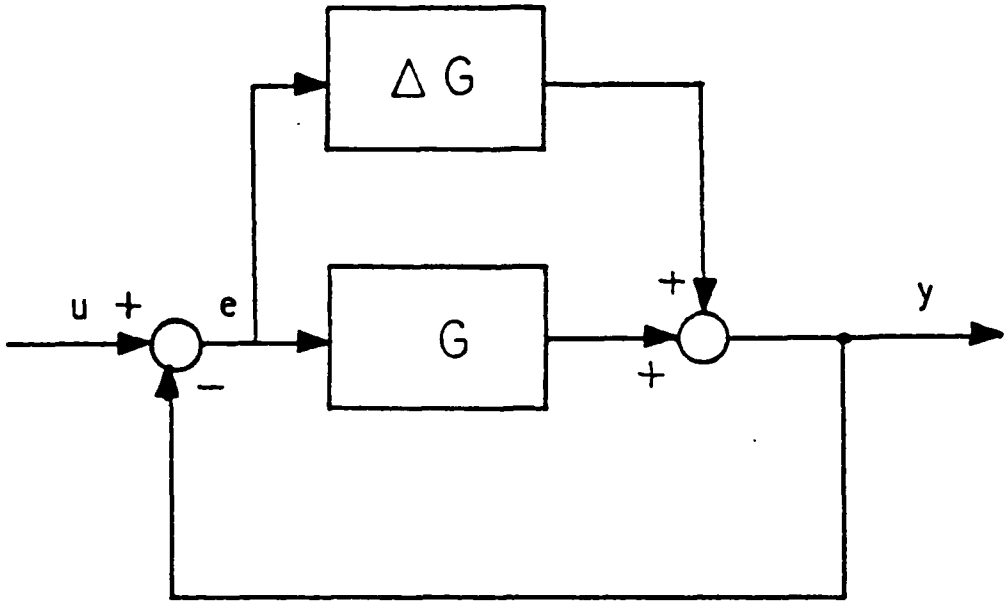


Figure 3.3 Additive Perturbation.

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

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

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

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

$$\|\Delta \underset{\sim}{G}(j\omega)\| < 1/\|(\underset{\sim}{I} + (\underset{\sim}{G})^{-1})^{-1}\| \tag{3.4}$$

$\omega \geq 0$. which may be expressed as;

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

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.

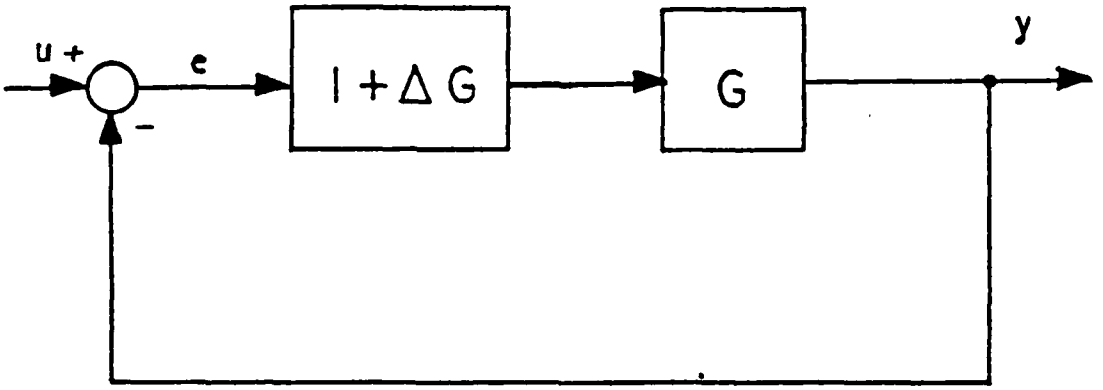


Figure 3.4 Multiplicative Perturbation.

IV. OPTIMIZATION

The purpose of this chapter will be to briefly describe several of the currently employed optimization techniques and the Autcmated 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

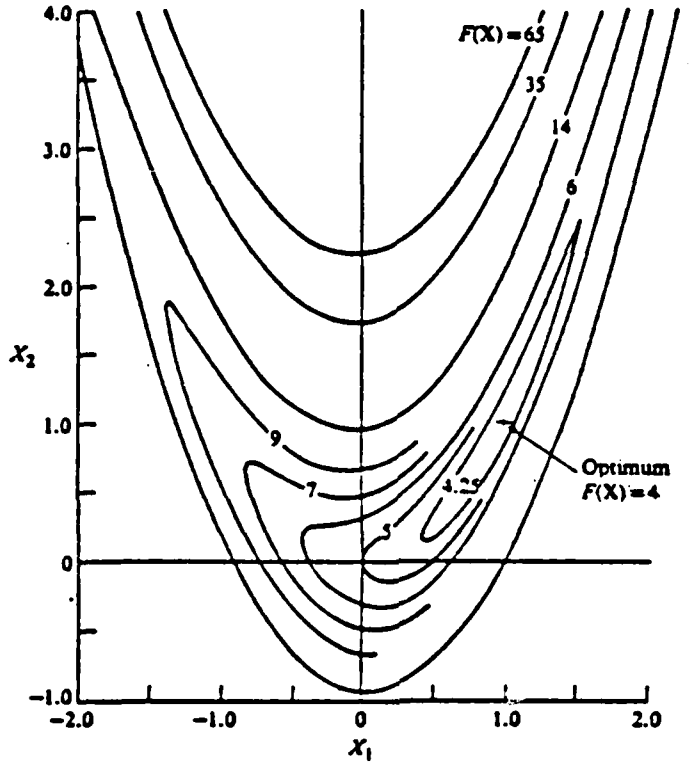


Figure 4.1 Design Space for Example Problem.

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

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

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

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:

$$W = \rho A h = \rho \pi D h \tag{4.4}$$

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

$$\sigma = P/A = P/\pi D t \tag{4.5}$$

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:

$$\text{minimize} \quad W = \rho \pi D t h \tag{4.6}$$

for the constraints

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

$$g(2) = \sigma/\sigma_b - 1 \leq 0 \tag{4.8}$$

$$g(3) = \sigma/\sigma_s - 1 \leq 0 \tag{4.9}$$

with D≥ 10e-06 and t ≥10e-06.

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

$\bar{\sigma}$, σ_b , σ_s refer to the allowable design stress, Euler buckling stress and the shell buckling stress respectively.

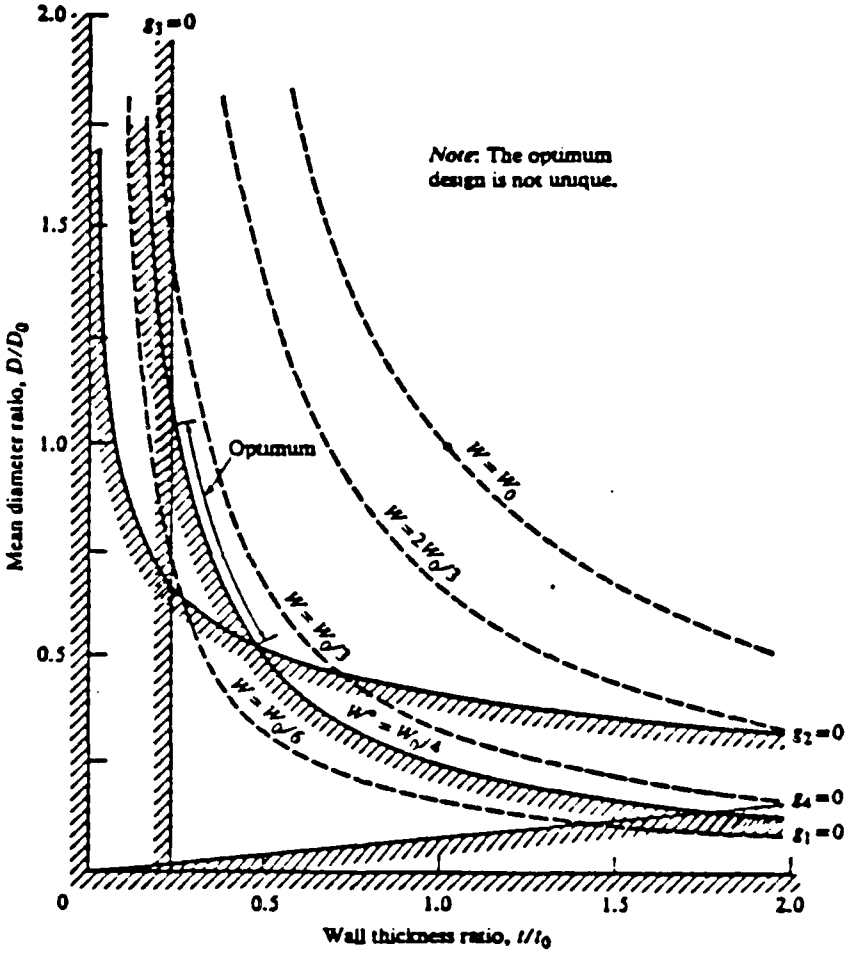


Figure 4.2 Design Space for Column Problem.

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.

The general optimization problem is then written as:

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

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

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

$$x_i^l \leq x_i \leq x_i^u \quad i=1,n \quad \text{side constraint} \tag{4.14}$$

where $\underline{X}$ = col (x1,x2,...,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

$$\underline{X}^1 = \underline{X}^0 + \alpha^* \underline{S}^1 \tag{4.15}$$

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

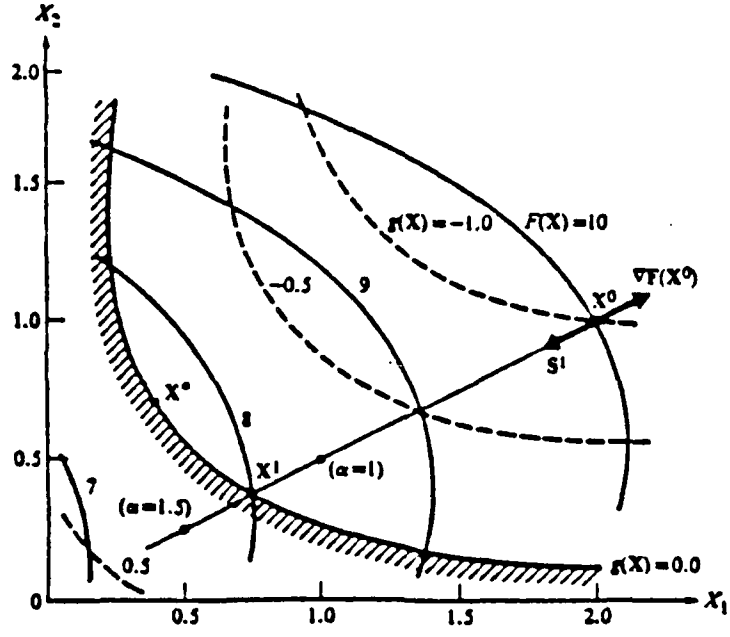


Figure 4.3 Illustrative Example for Iteration.

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:

$$\underline{X}^2 = \underline{X}^1 + \alpha^* \underline{S}^2 \qquad (4.16)$$

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

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 absolute 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:

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

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

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

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 definite 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.

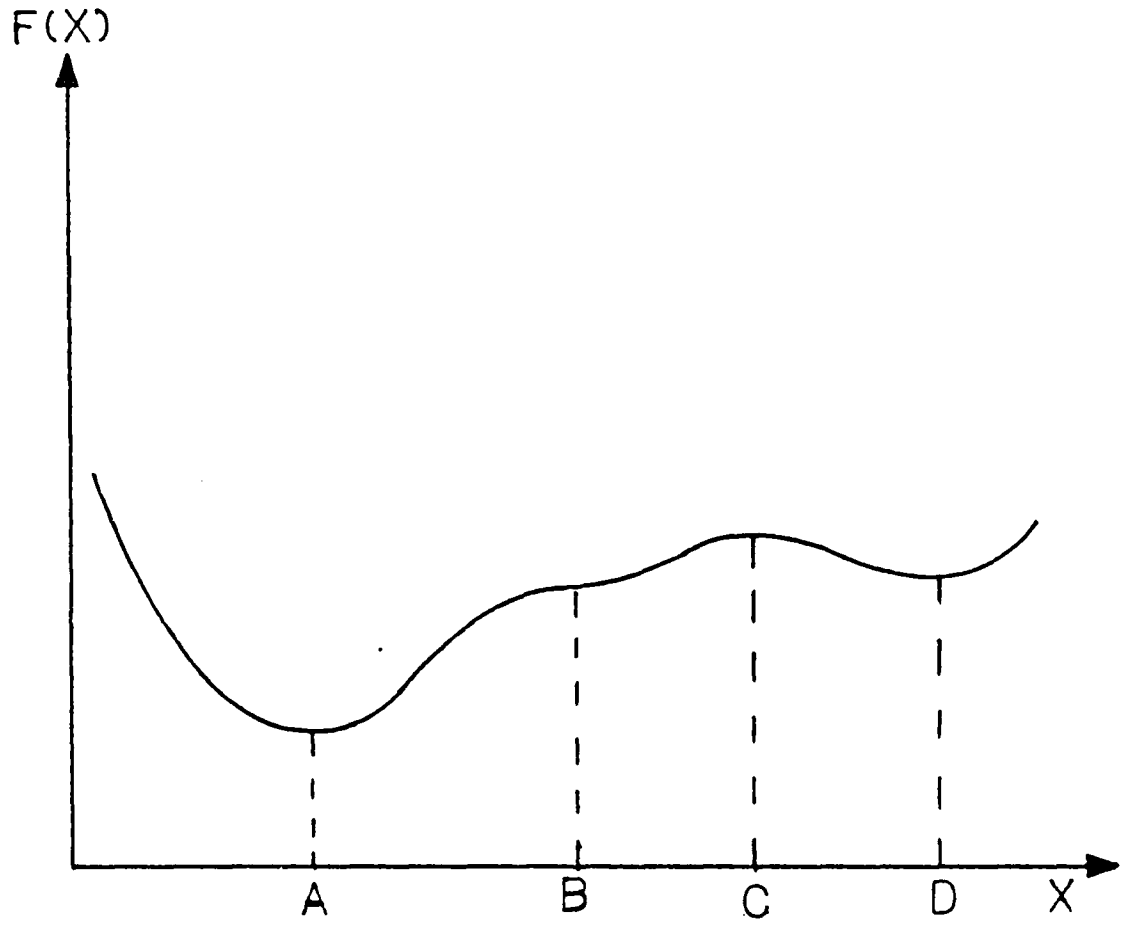


Figure 4.4 Plot of Various Points With Zero Gradient.

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

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

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

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

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 opposite 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:

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

where $\lambda_j \geq 0$ and λ_{k+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, optimization 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

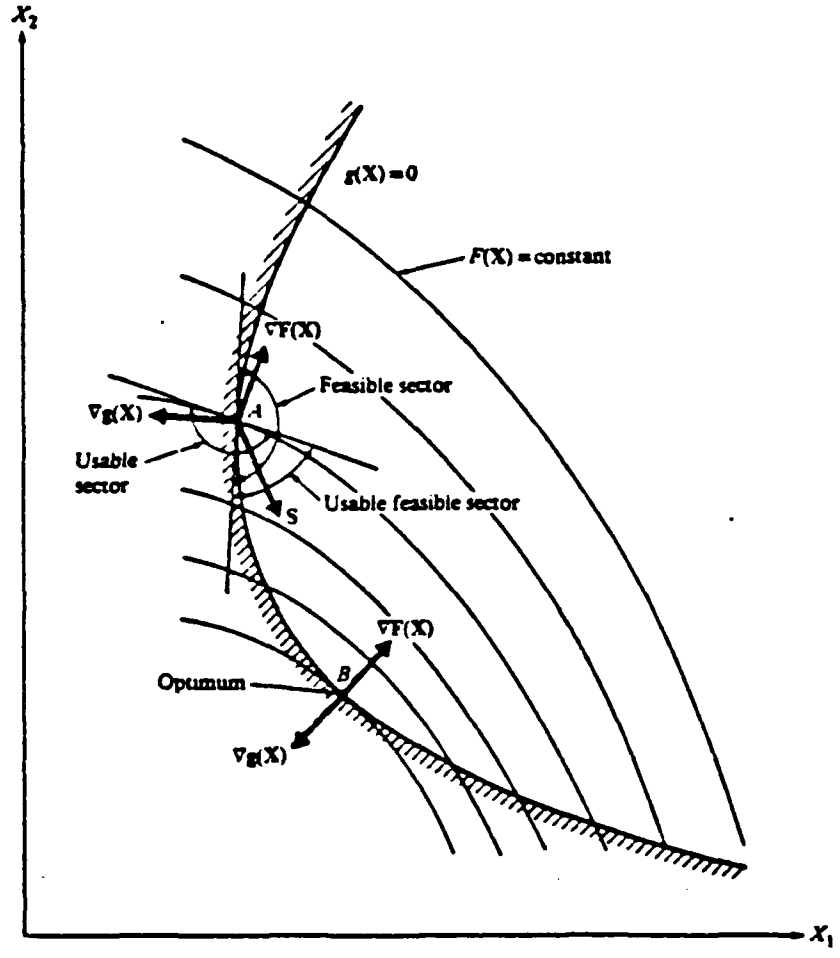


Figure 4.5 Constrained Optimization Example.

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 available 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

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 $F(\underline{x})$ evaluated at each point. The point corresponding to the smallest value of $F(\underline{x})$ could then be considered the optimum value of the one dimensional search. This method is hit and miss and better methods of

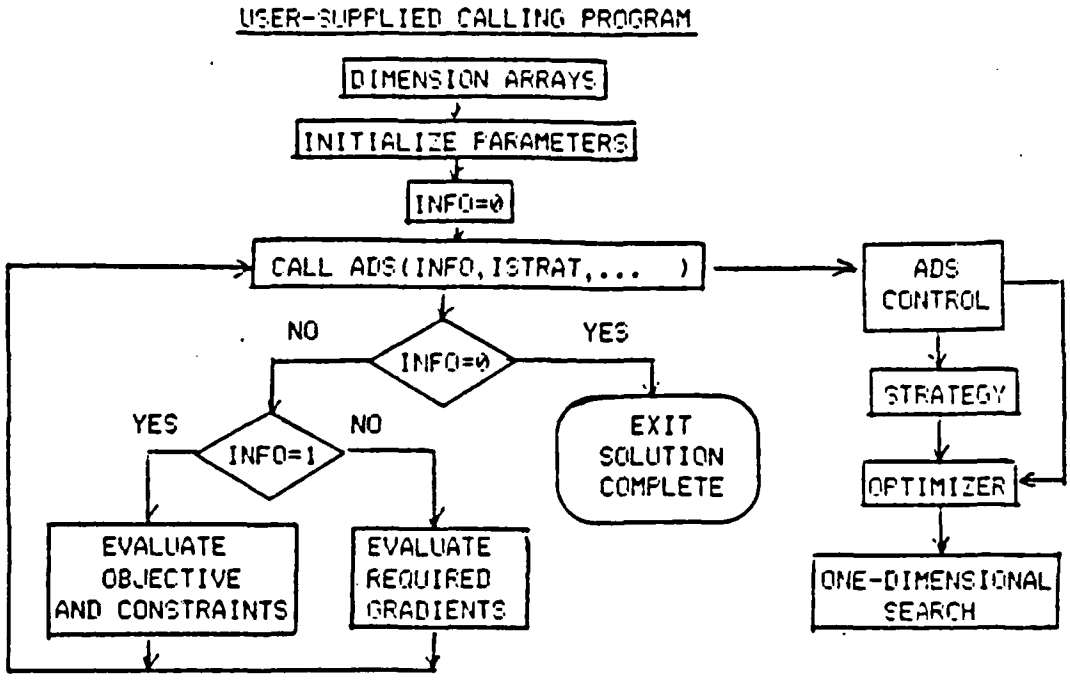


Figure 4.6 Organization of ADS Program.

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

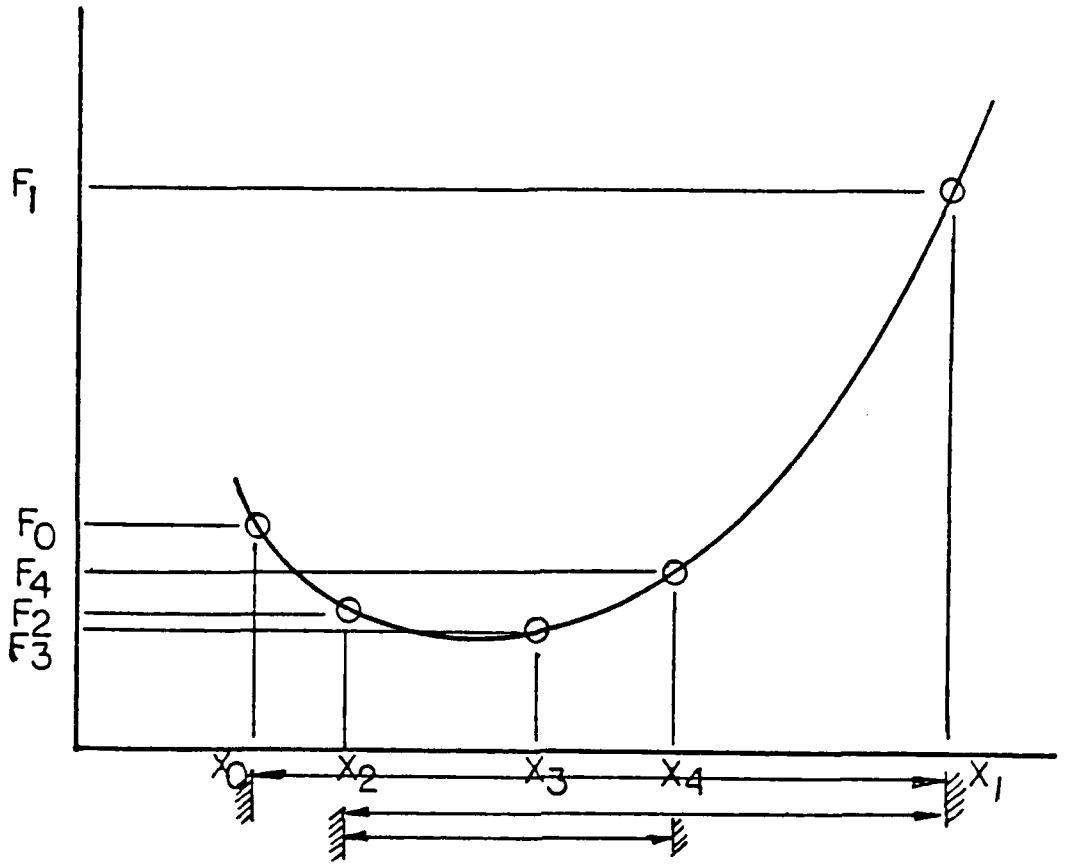


Figure 4.7 Golden Section Diagram.

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

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:

$$X^2 = (1 - \tau) X^0 + \tau X^1 \tag{4.22}$$

$$X^3 = \tau X^0 + (1 - \tau) X^1 \tag{4.23}$$

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:

$$F = a_0 + a_1 X + a_2 X^2 \tag{4.24}$$

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

$$X^* = -a_1 / 2a_2 \tag{4.25}$$

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

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

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

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

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).

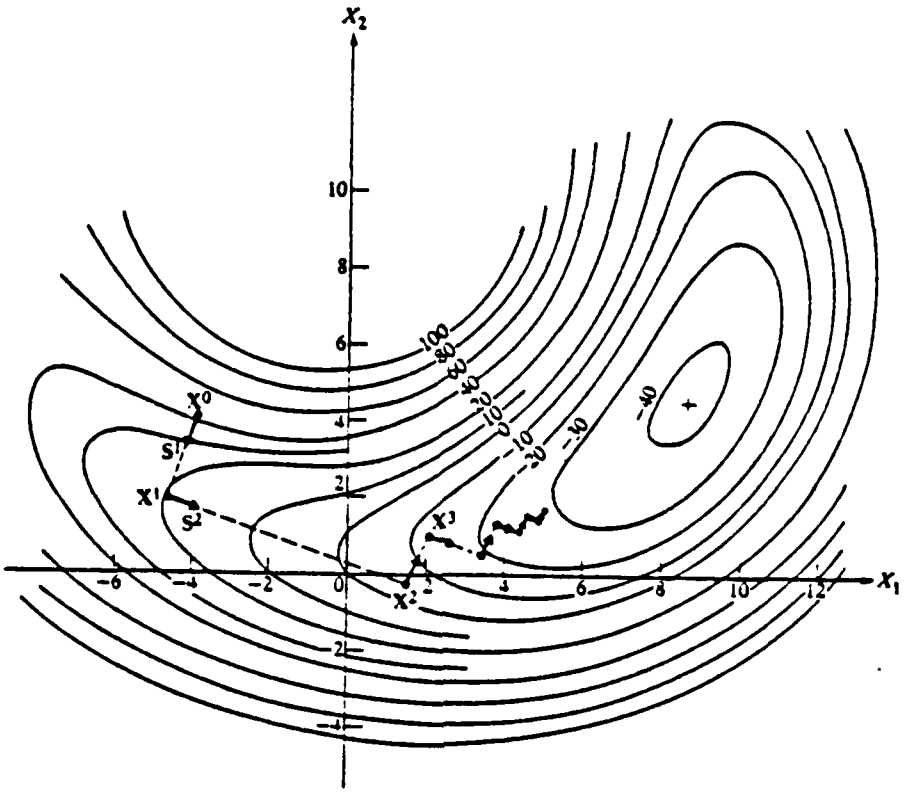


Figure 4.8 Steepest Descent Algorithm.

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 :

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

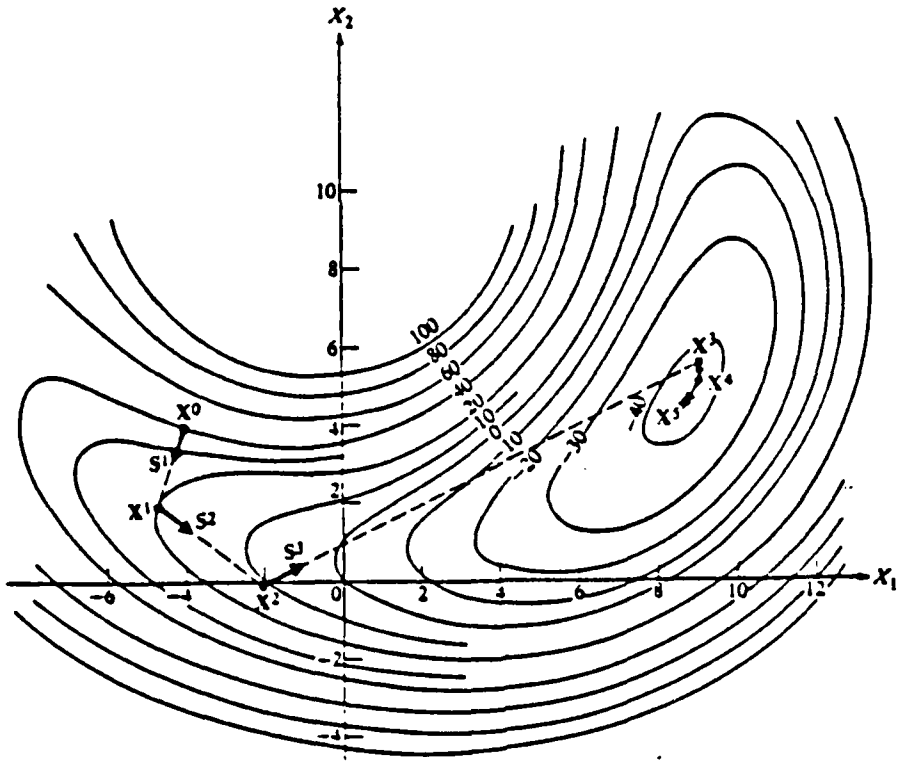


Figure 4.9 Fletcher-Reeves Conjugate Direction Algorithm.

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 :

$$\Phi(\underline{x}, r_p) = F(\underline{x}) + r_p P(\underline{x}) \qquad (4.29)$$

The constant r_p 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.

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:

$$\text{minimize} \quad F(\underline{x}) \tag{4.30}$$

$$\text{subject to } h_K(\underline{x}) = 0 \quad k=1,l \tag{4.31}$$

Next a Lagrangian is created such that

$$L(\underline{x},\lambda) = F(\underline{x}) + \sum_{K=1}^{l} \lambda_K h_K(x) \tag{4.32}$$

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

$$A(\underline{x},\lambda,r_p) = F(\underline{x}) + \sum_{K=1}^{l} (\lambda_K h_K(\underline{x}) + r_p (h_K(\underline{x}))^2) \tag{4.33}$$

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:

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

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

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

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

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

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:

$$\text{gain margin} = GM = 1/(1+\alpha_0) \tag{5.3}$$

$$\text{phase margin} = PM = \pm\cos^{-1}(1-\alpha_0^2/2) \tag{5.4}$$

provided that equation 5.5 holds.

$$\underset{\sim}{\sigma}(\underset{\sim}{I}+\underset{\sim}{G}) \geq \alpha_0 \tag{5.5}$$

for some $\alpha_0 \leq 1$

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.

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

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].

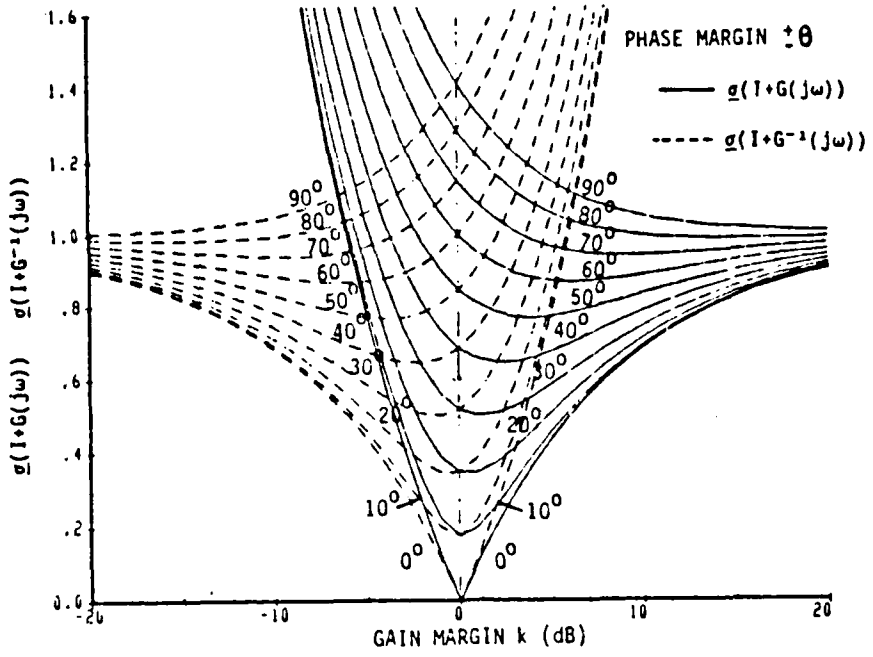


Figure 5.1 Universal Gain and Phase Singular Value Plot.

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

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

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 formulation can be used as a negative constraint function it has not been implemented as such within this program. There are numerous

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:

$$\dot{\underline{x}} = \underset{\sim}{A}\underline{x} + \underset{\sim}{B}\underline{u} \tag{5.8}$$

$$\underline{y} = \underset{\sim}{C}\underline{x} \tag{5.9}$$

$$\underline{u} = -\underset{\sim}{F}\underline{x} \tag{5.10}$$

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

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;

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

$$A_c = A - FC$$

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

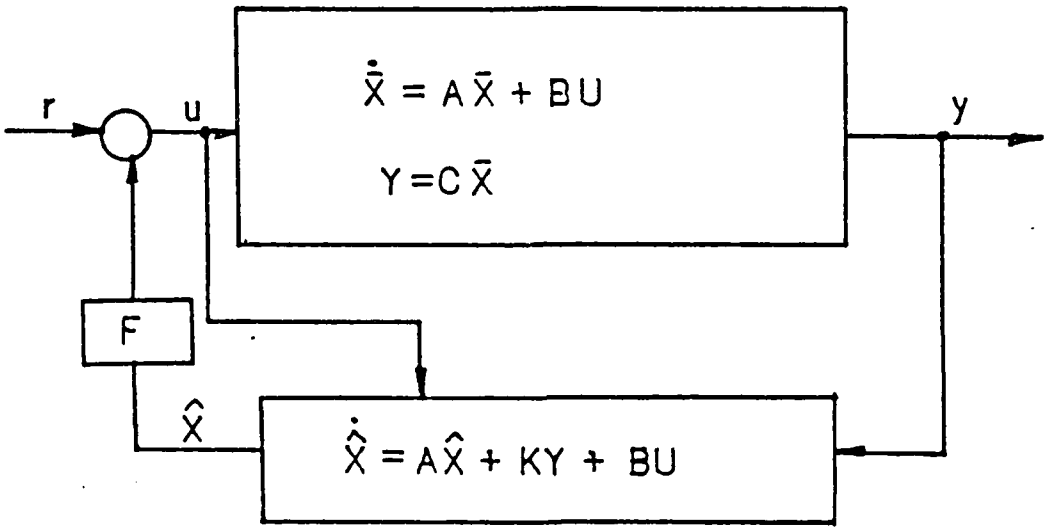


Figure 5.2 Observer Implementation.

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,

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

$$\begin{aligned} y1 &= x1 \\ y2 &= x2 \end{aligned} \tag{6.2}$$

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

$$\underset{\sim}{G} = \underset{\sim}{C}(s\underset{\sim}{I}-\underset{\sim}{A})^{-1}\underset{\sim}{B} \tag{6.5}$$

the transfer matrix may be written as equation 6.6.

$$\underset{\sim}{G} = \begin{bmatrix} 1/s+1 & b_{12}/s+1 \\ 0 & 1/s+1 \end{bmatrix} \tag{6.6}$$

which gives the return difference matrix, equation 6.7.

$$\underset{\sim}{I}+\underset{\sim}{G}(s) = \begin{bmatrix} s+2/s+1 & b_{12}/s+1 \\ 0 & s+2/s+1 \end{bmatrix} \tag{6.7}$$

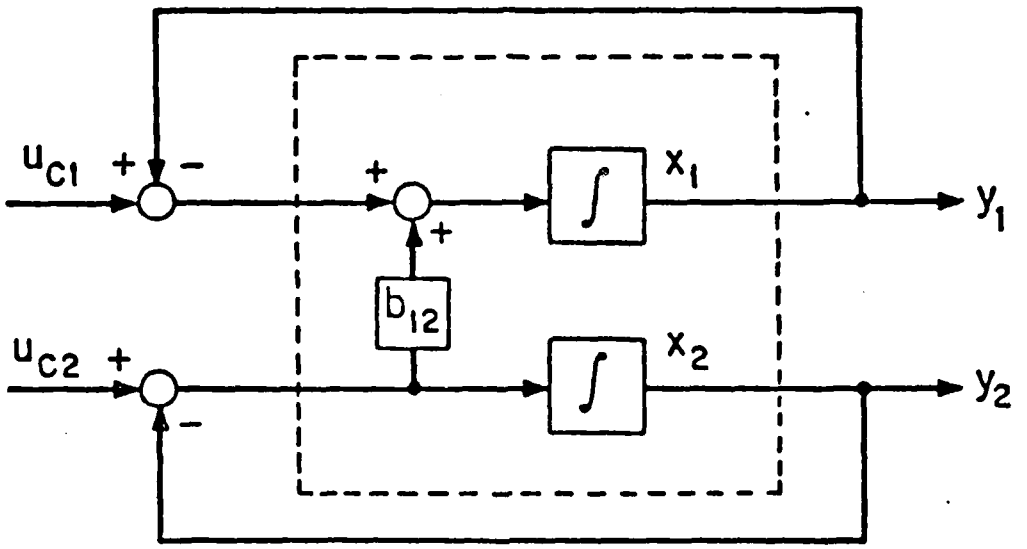


Figure 6.1 Basic Multi-input Multi-output System.

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 difference matrix $(\underset{\sim}{I}+\underset{\sim}{G})$, the determinant may be written as equation 6.8.

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

Therefore, the multivariable Nyquist diagram will be as

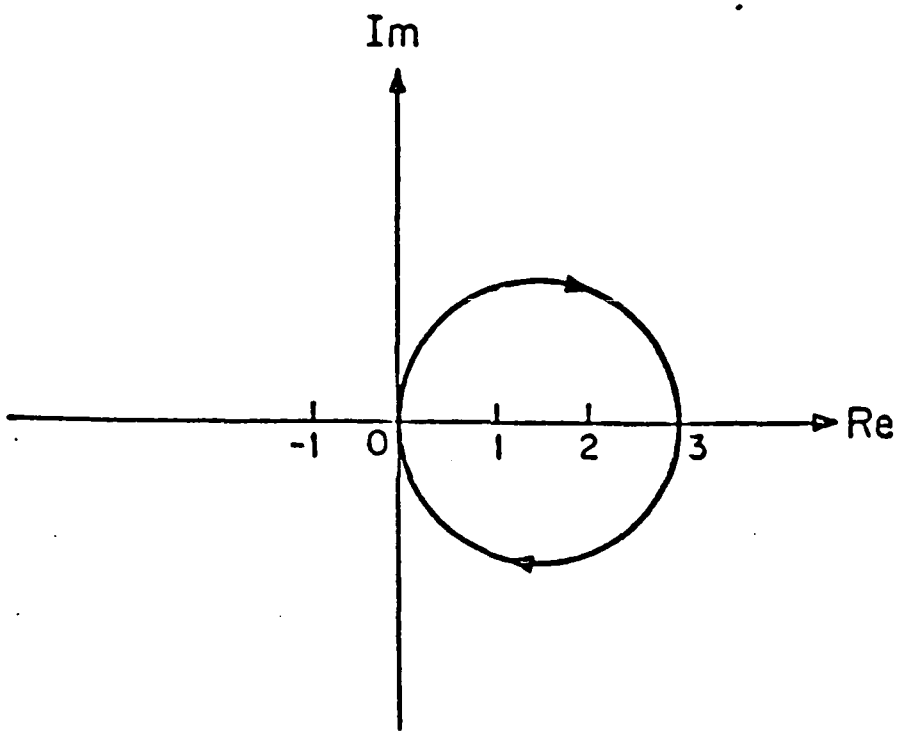


Figure 6.2 Multivariable Nyquist for $2s+3/(s+1)^2$.

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

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

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 $(\underset{\sim}{I}+\underset{\sim}{G})$, $\underline{\sigma}$. (For numerical calculations a value of $b_{12} = 50$ 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

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

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 $(\underset{\sim}{I}+\underset{\sim}{G}(j\omega))$

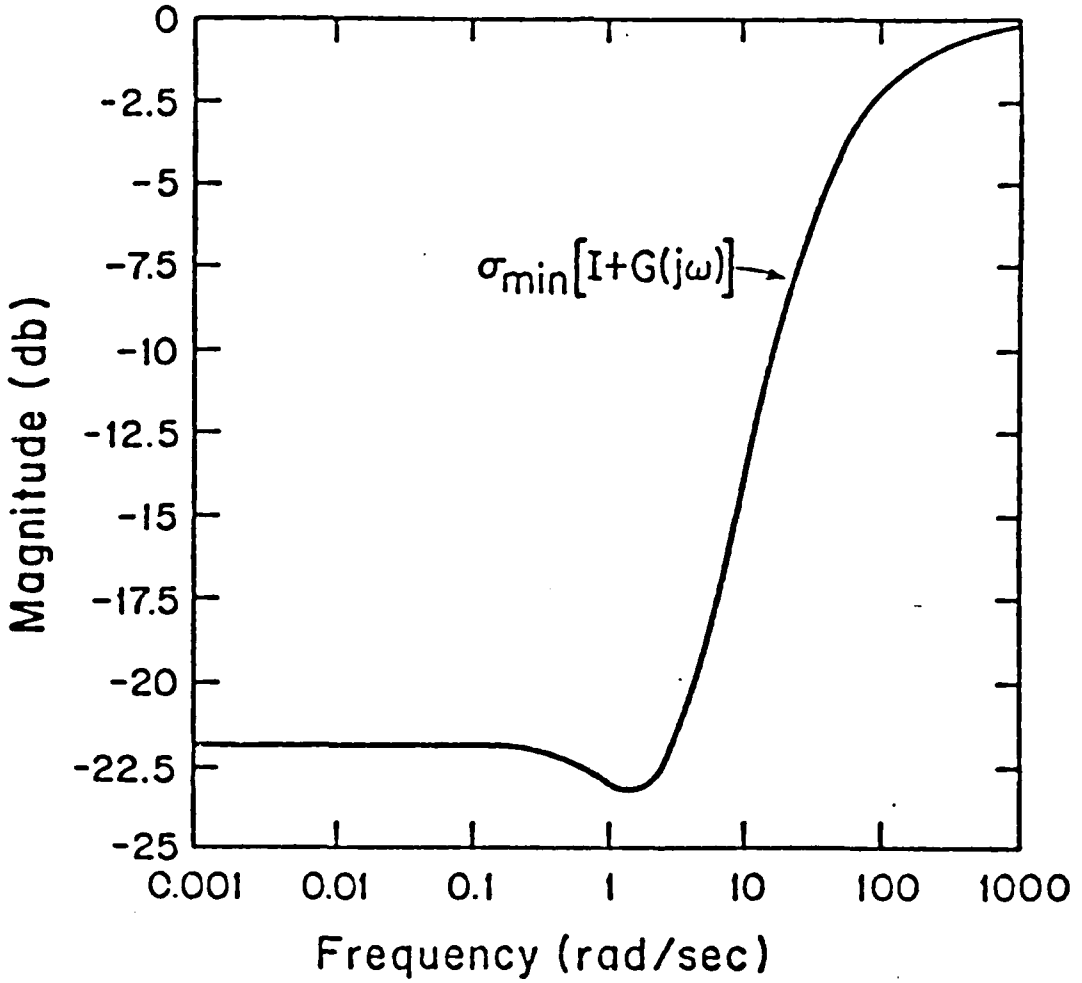


Figure 6.3 Minimum Singular Value Plot, Example Problem.

is nearly singular at some frequency ω_o, then the multivariable system will not be robust with regard to any modelling errors within the system. This is because any small change in $\underset{\sim}{G}(j\omega)$ can then make $\underset{\sim}{I}+\underset{\sim}{G}(j\omega)$ singular and the $\det(\underset{\sim}{I}+\underset{\sim}{G})$ 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

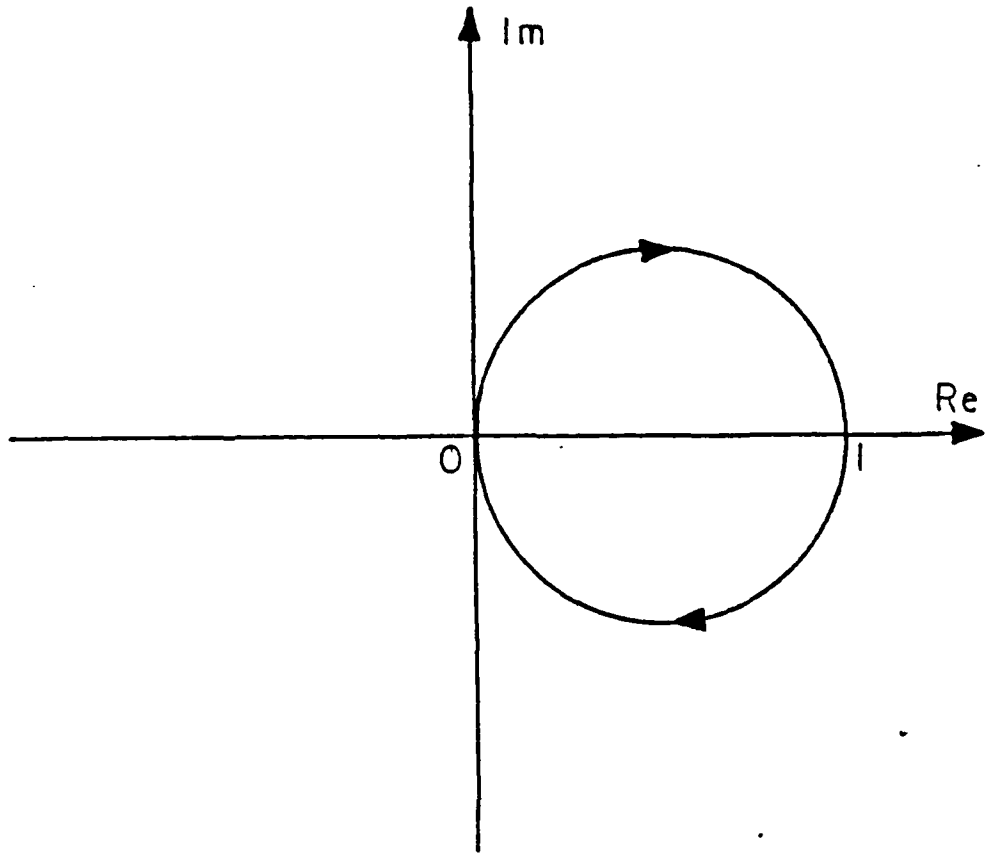


Figure 6.4 Nyquist Diagram of 1/s+1.

can be demonstrated . The first step is to establish a baseline 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 feedback 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

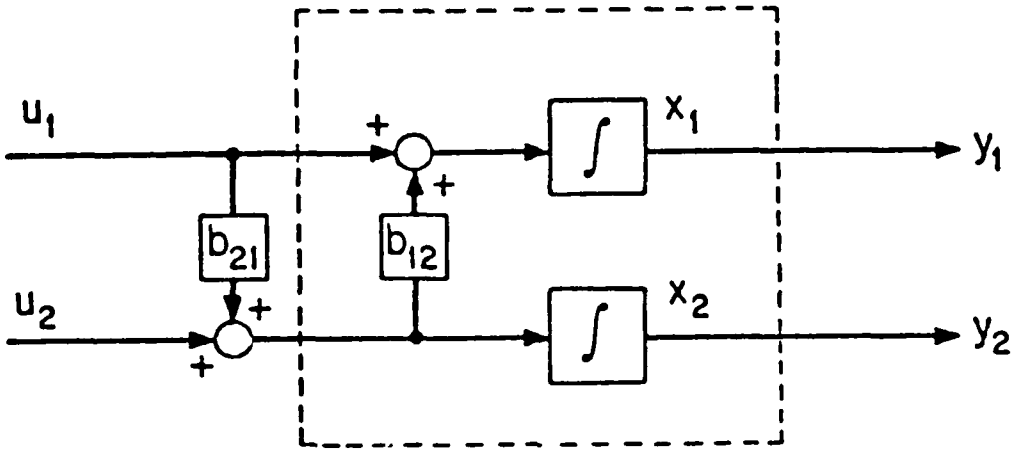


Figure 6.5 Perturbed System.

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

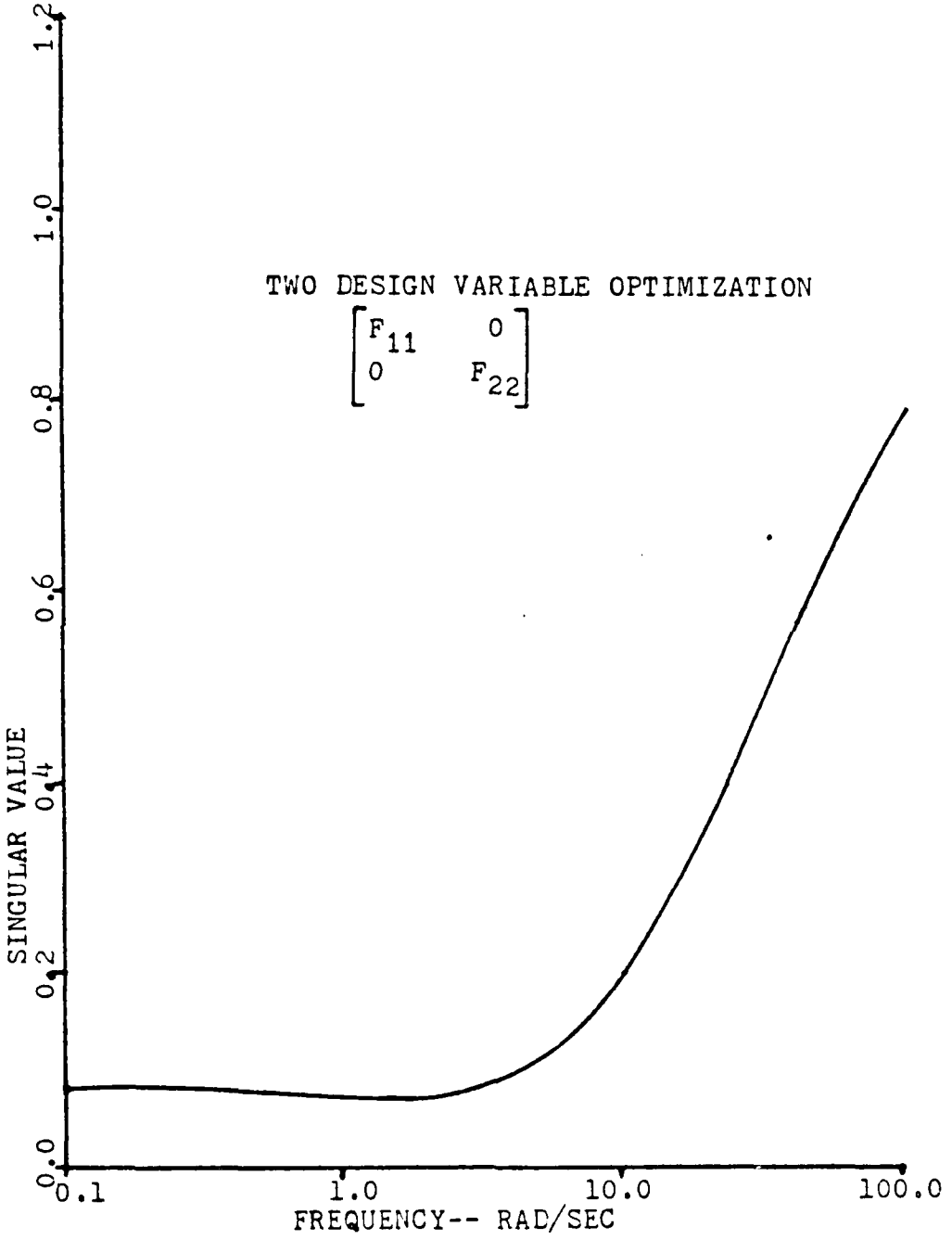
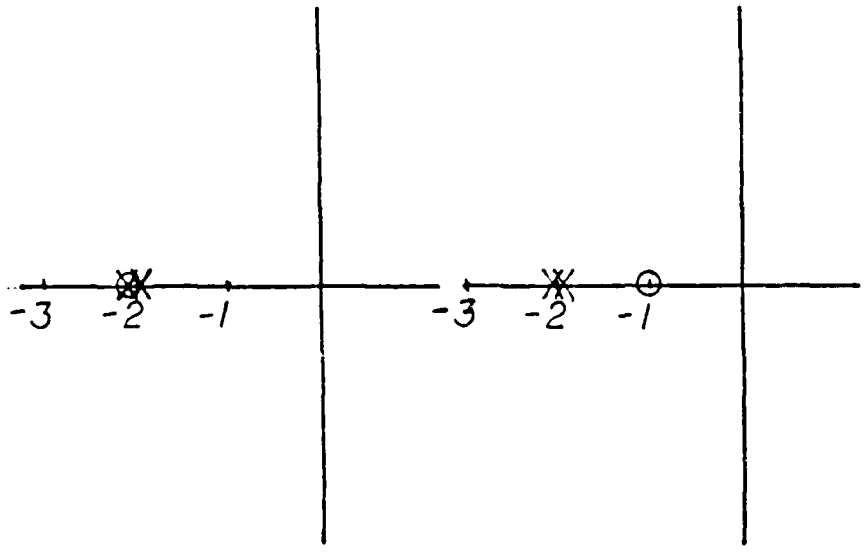
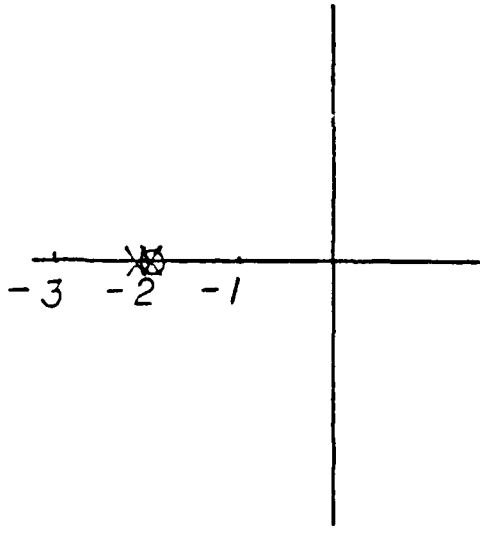


Figure 6.6 Singular Value Plot for Simple Problem.



1 TO 1 2 TO 1



2 TO 2

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

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.

$$\underset{\sim}{A}_c = \underset{\sim}{A} - \underset{\sim}{B}\underset{\sim}{F} \tag{6.11}$$

where equation 6.12 gives A .

$$\underset{\sim}{A}_c = \begin{bmatrix} -1-f_{11} & -f_{12}-b_{12}f_{22} \\ 0 & -1-f_{22} \end{bmatrix} \tag{6.12}$$

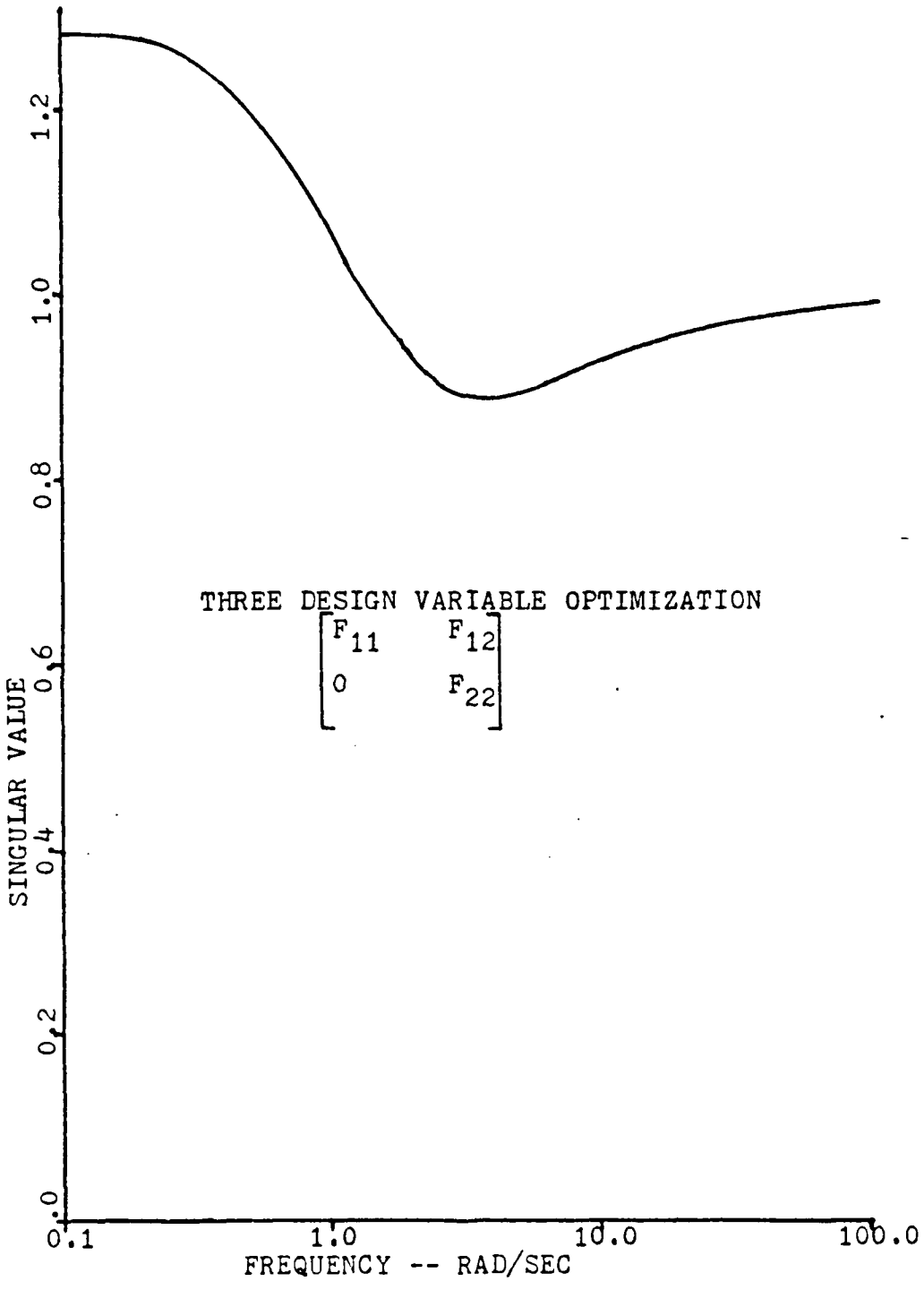


Figure 6.8 Singular Value Plot Case Three.

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

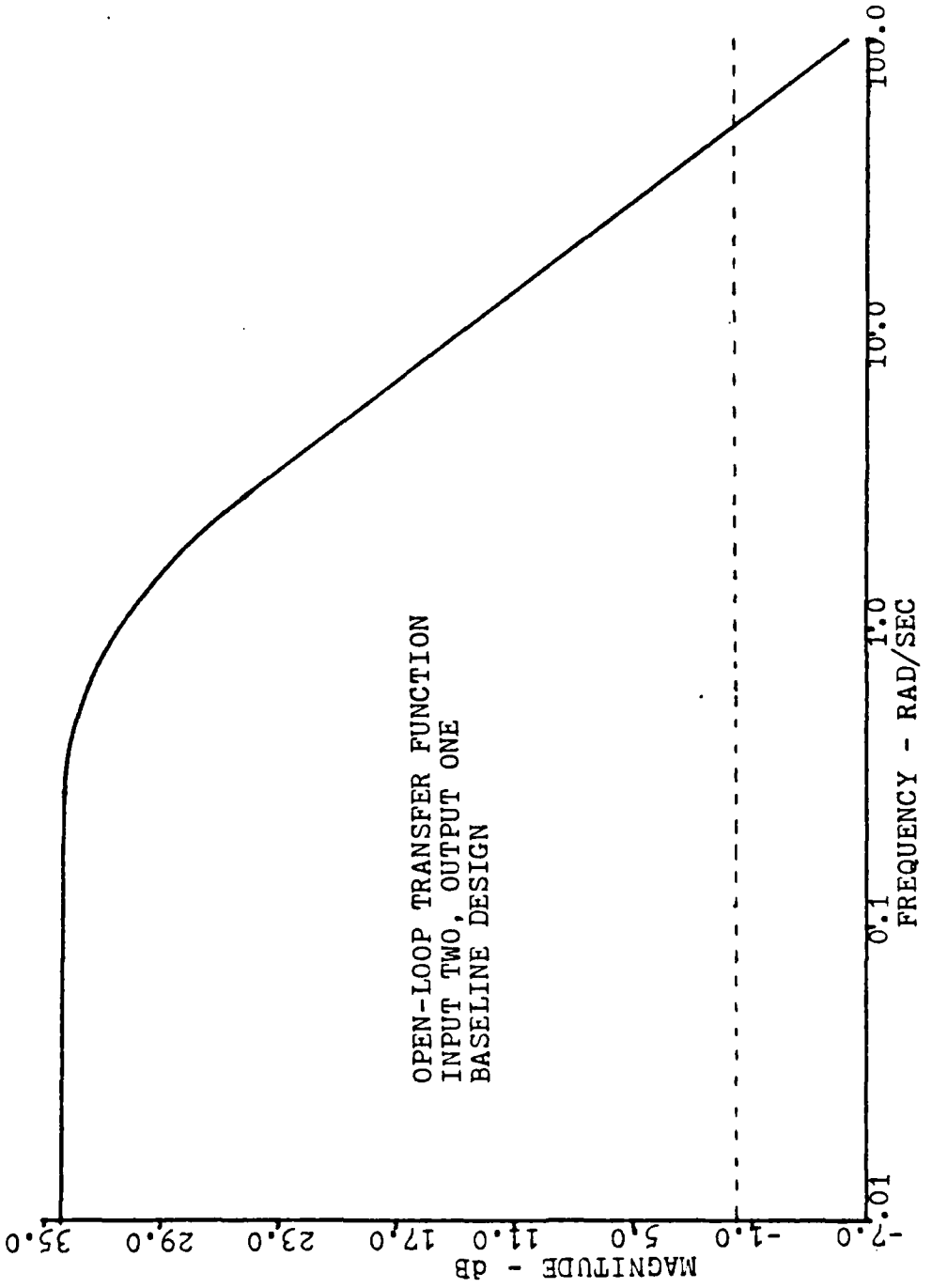


Figure 6.9 Open-loop Transfer Function 2-1 for Baseline.

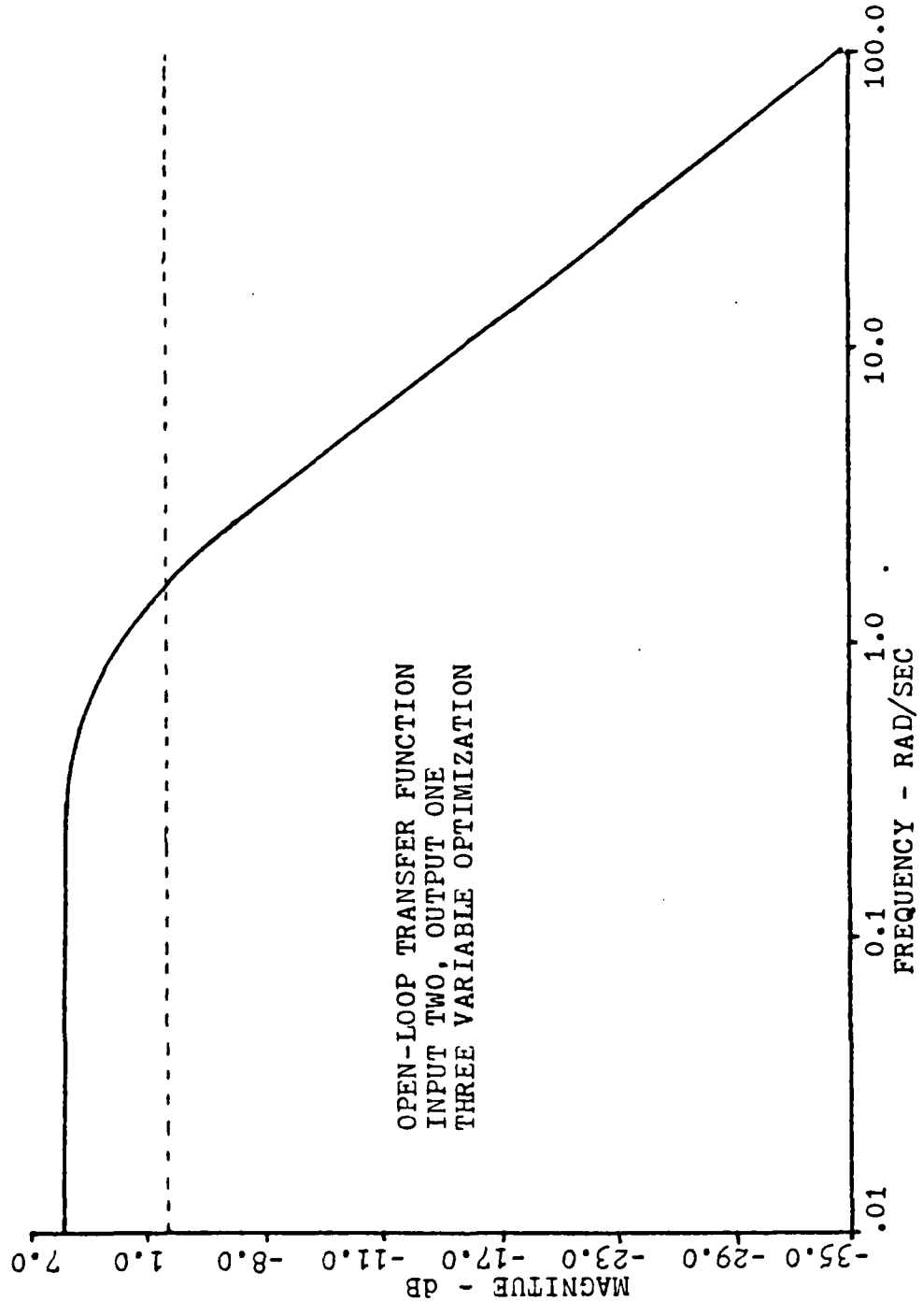


Figure 6.10 Transfer Function 2-1 Optimized(3 Var).

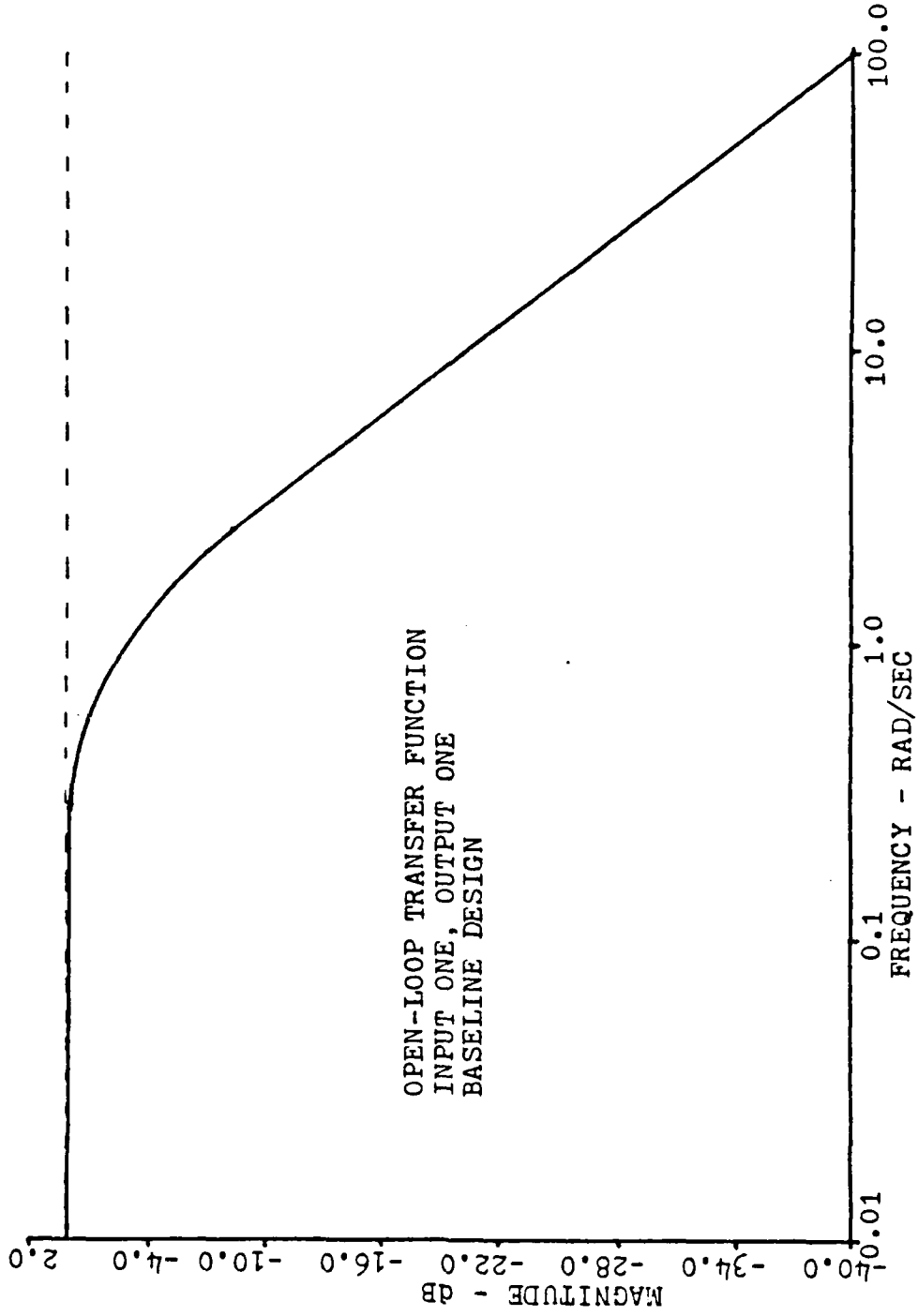


Figure 6.11 Transfer Function 1-1 Baseline.

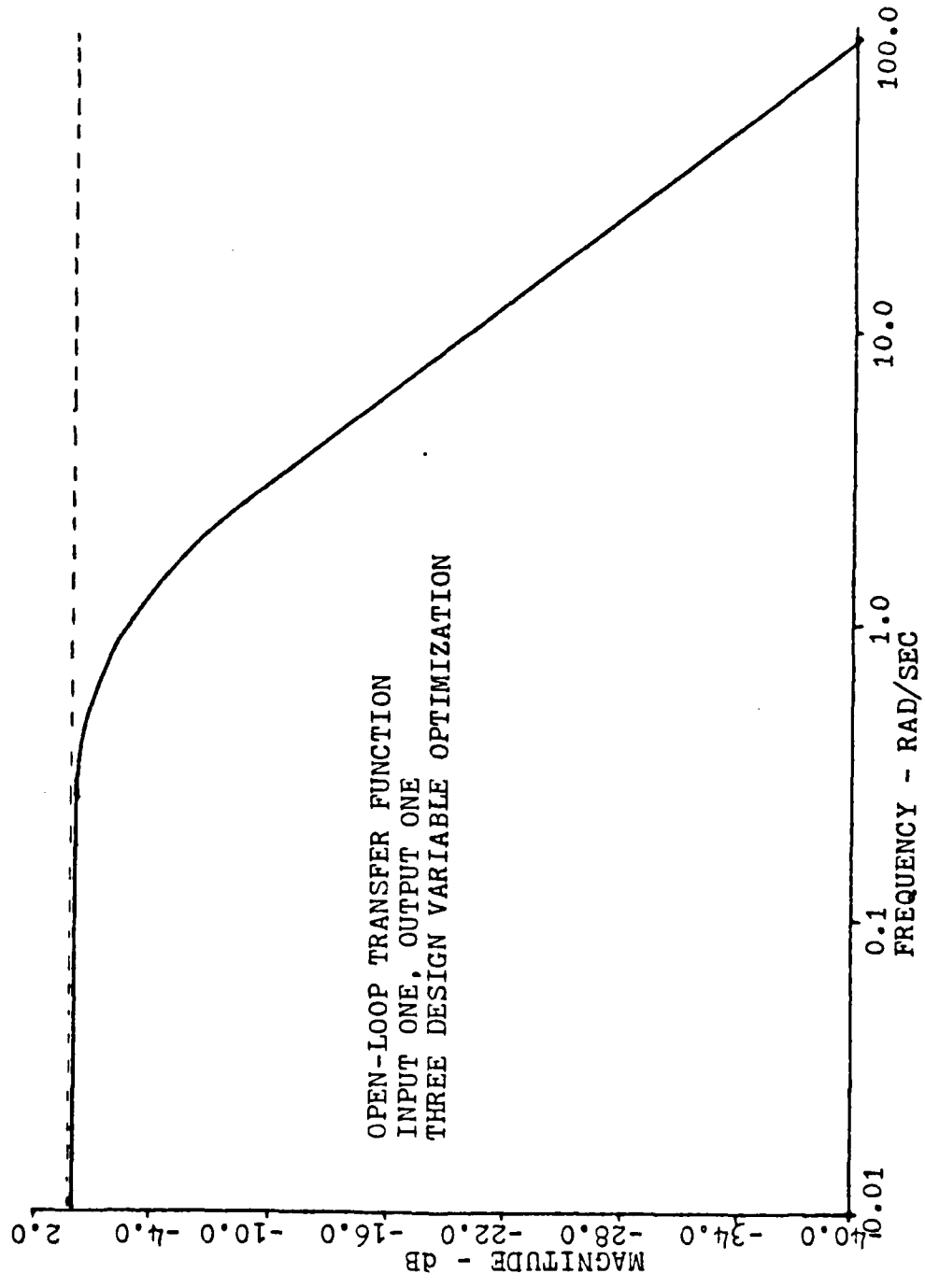


Figure 6.12 Transfer Function 1-1 Optimized (3 variables).

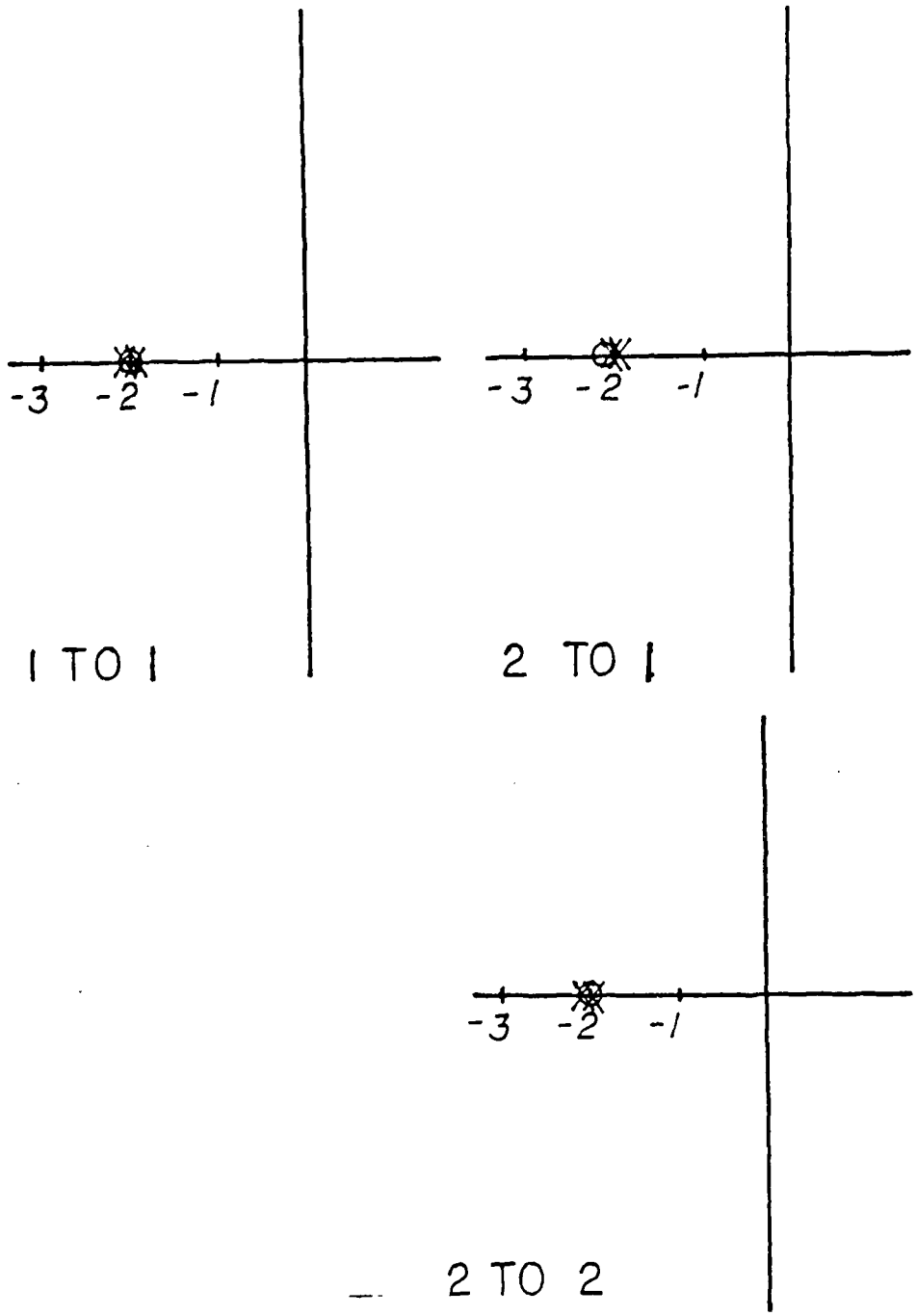


Figure 6.13 Closed-loop Pole-Zero Plot Case 3.

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 ADS 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 $\underline{I}+\underset{\sim}{F}\underset{\sim}{G}$ while the return difference for point 2 is $\underline{I}+\underset{\sim}{G}\underset{\sim}{F}$. 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

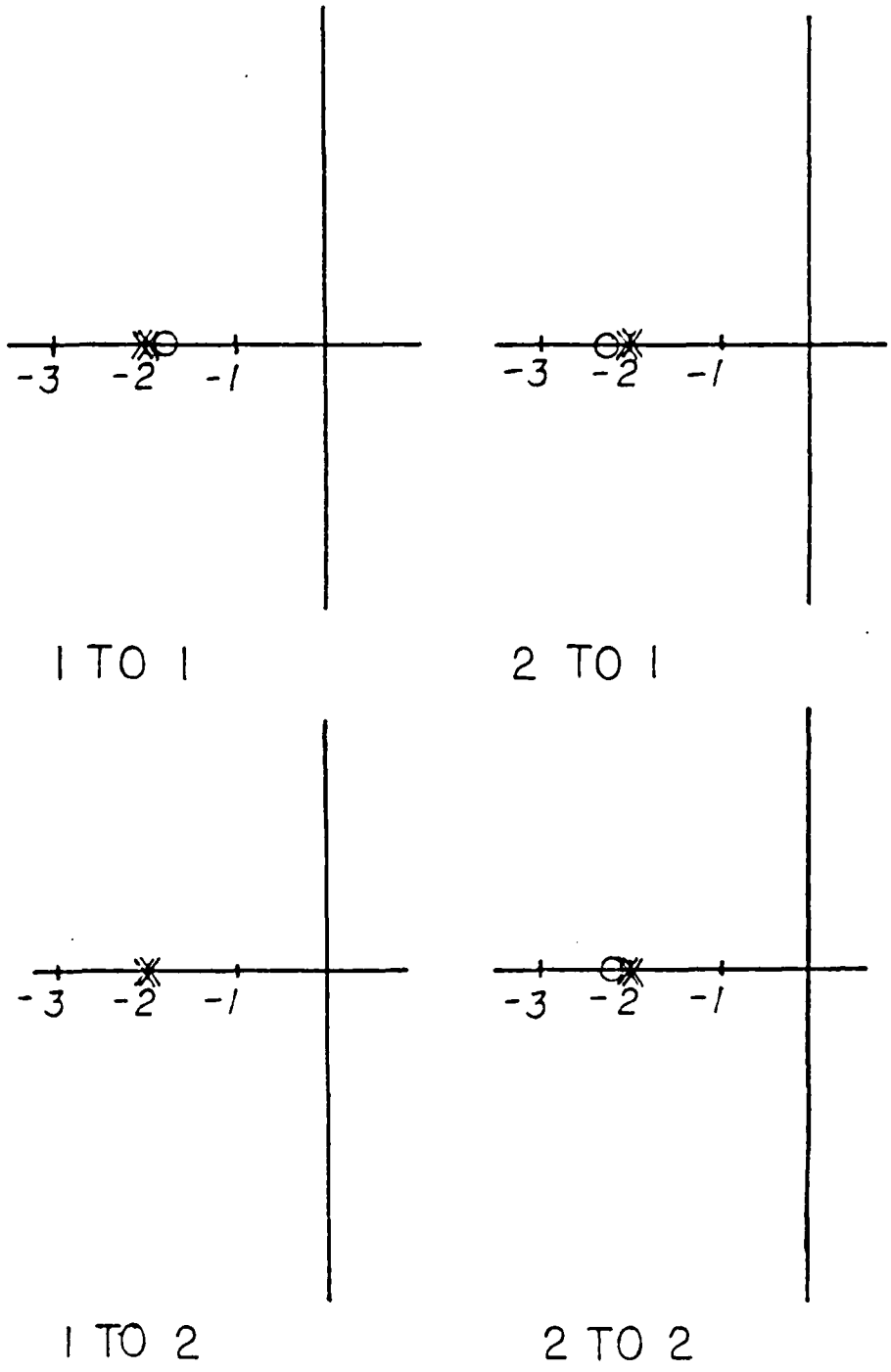


Figure 6.14 Pole-Zero Plot Simple Problem (4 Variables).

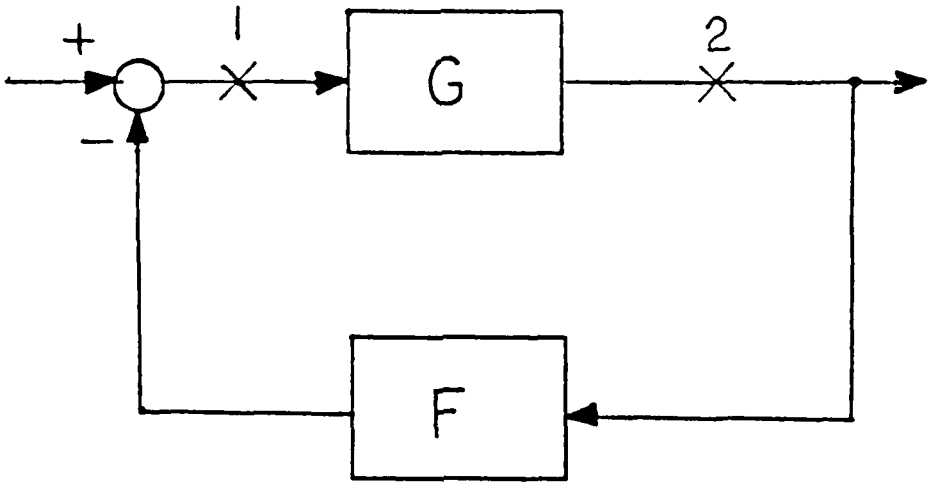


Figure 6.15 System Block Diagram.

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.

One 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.

To 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 of each element of the transfer matrix and look 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 on the pole-zero diagram, are shifted. This zero 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.

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

$$\dot{\underline{x}} = \underset{\sim}{A}\,\underline{x} + \underset{\sim}{B}\,\underline{u} \tag{7.1}$$

$$X = (v, p, r, \varphi) \tag{7.2}$$

$$u = (\delta_a, \delta_r) \tag{7.3}$$

where

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

and where

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

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.

$$\underline{u} = -\underset{\sim}{F}_i x + h_i \varphi_c \tag{7.4}$$

$i = 1, 2$ or 3

where the following values of F and h were are:

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

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

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

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

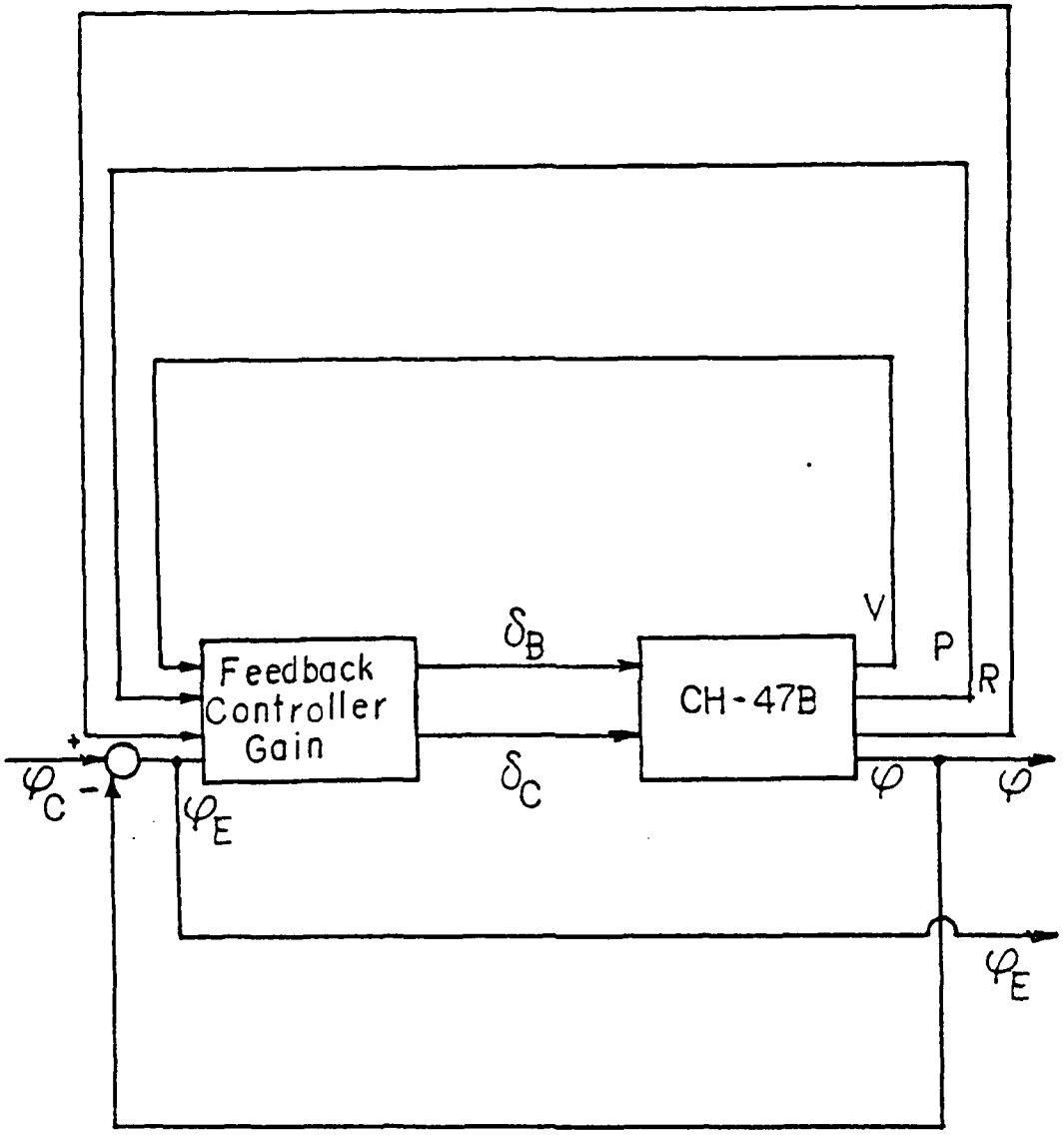


Figure 7.1 Feedback Control Structure.

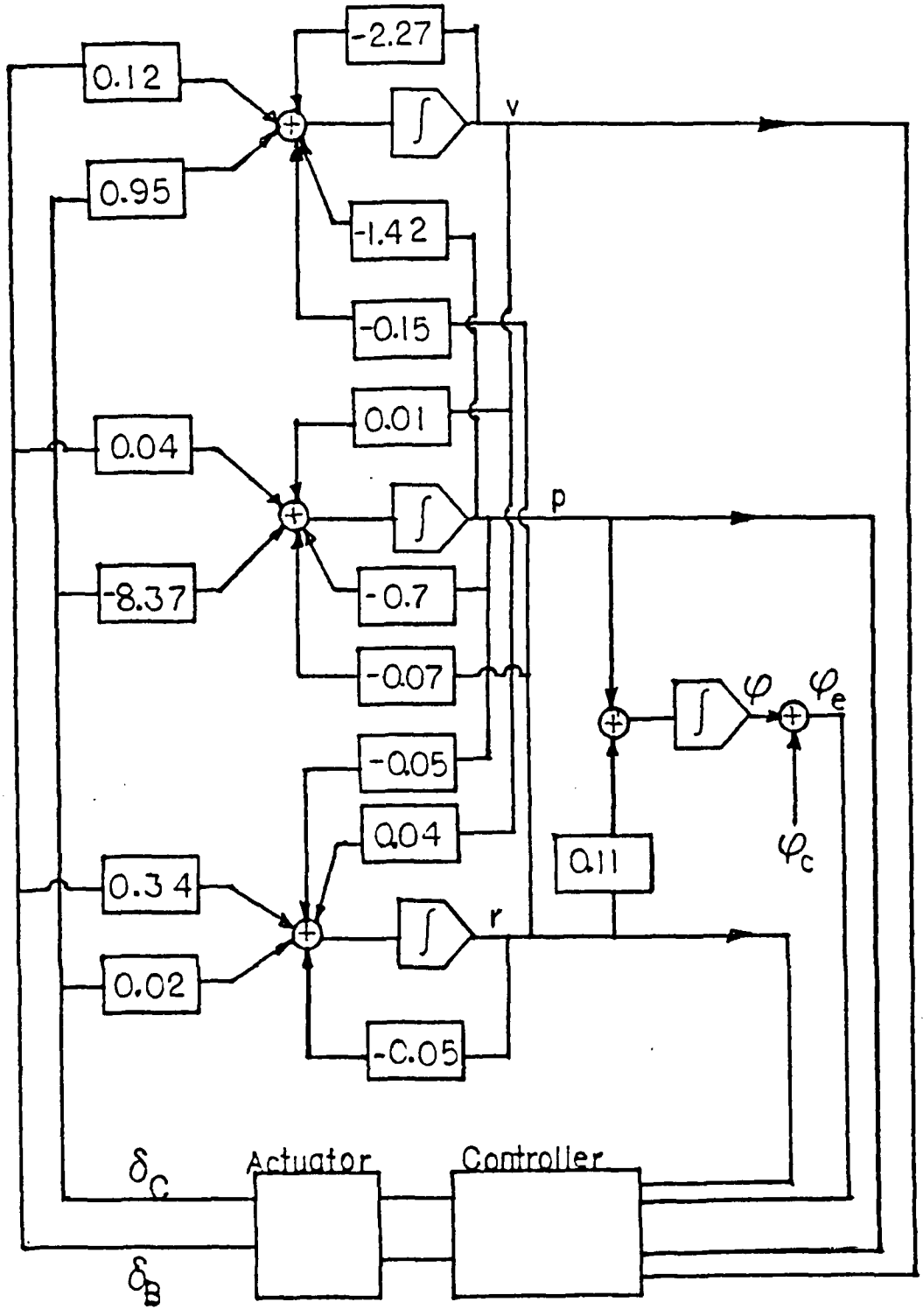


Figure 7.2 System Diagram.

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 δ_c 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

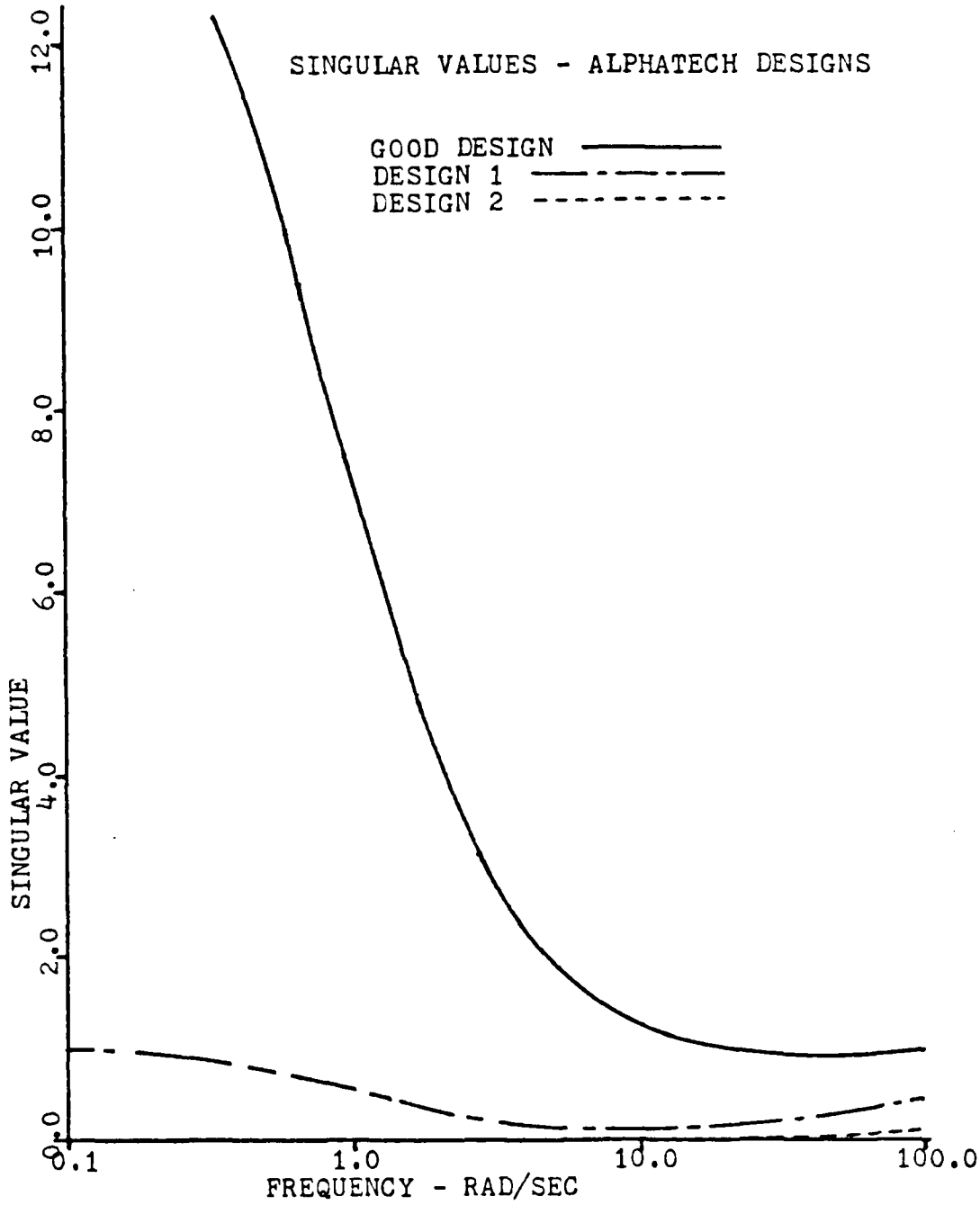


Figure 7.3 Alphatech Design Singular Value Plots.

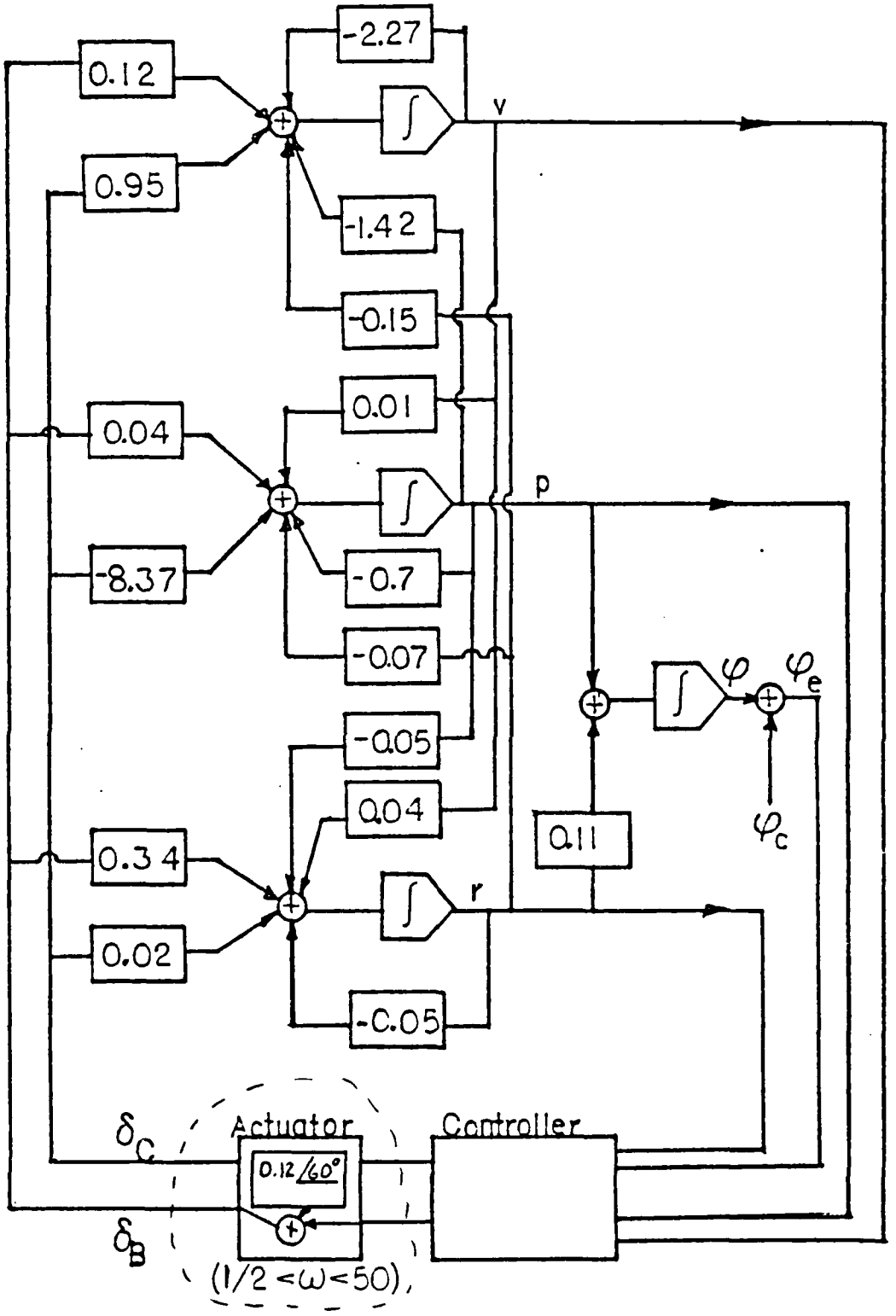


Figure 7.4 Design One Perturbation Input.

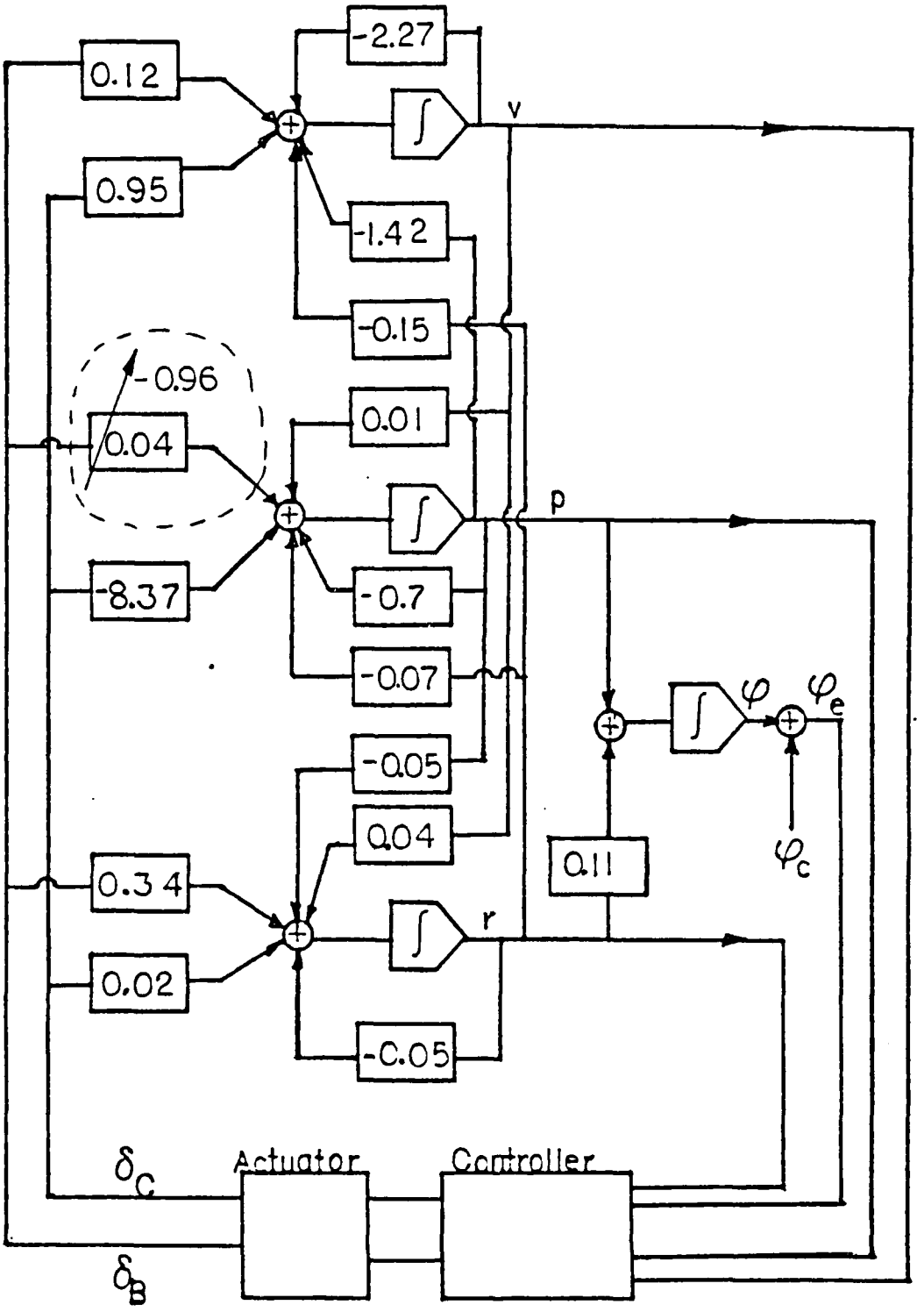


Figure 7.5 Design Two Perturbation Input.

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

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

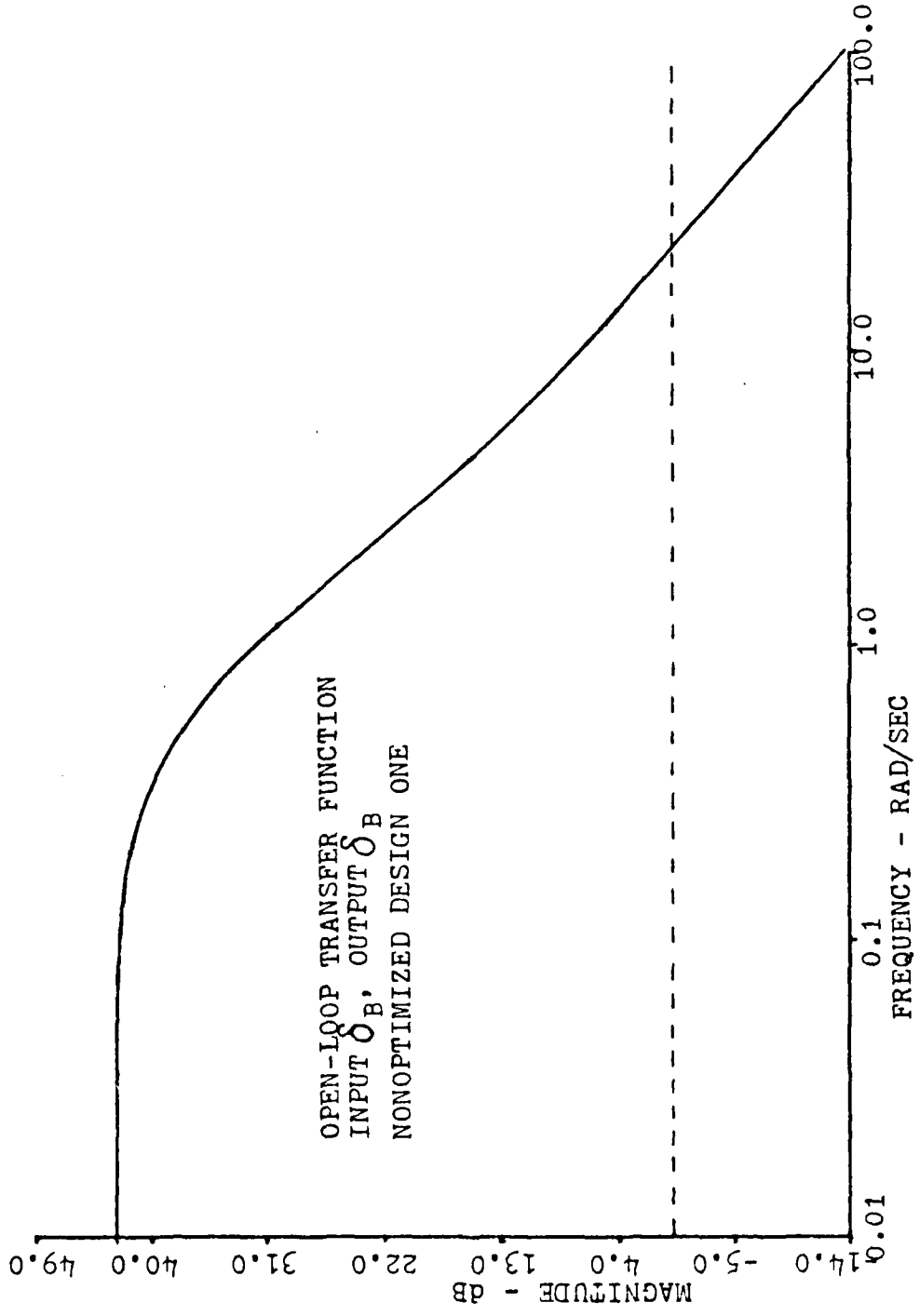


Figure 7.6 Transfer Function δ_B-δ_B Design One, Nonoptimized.

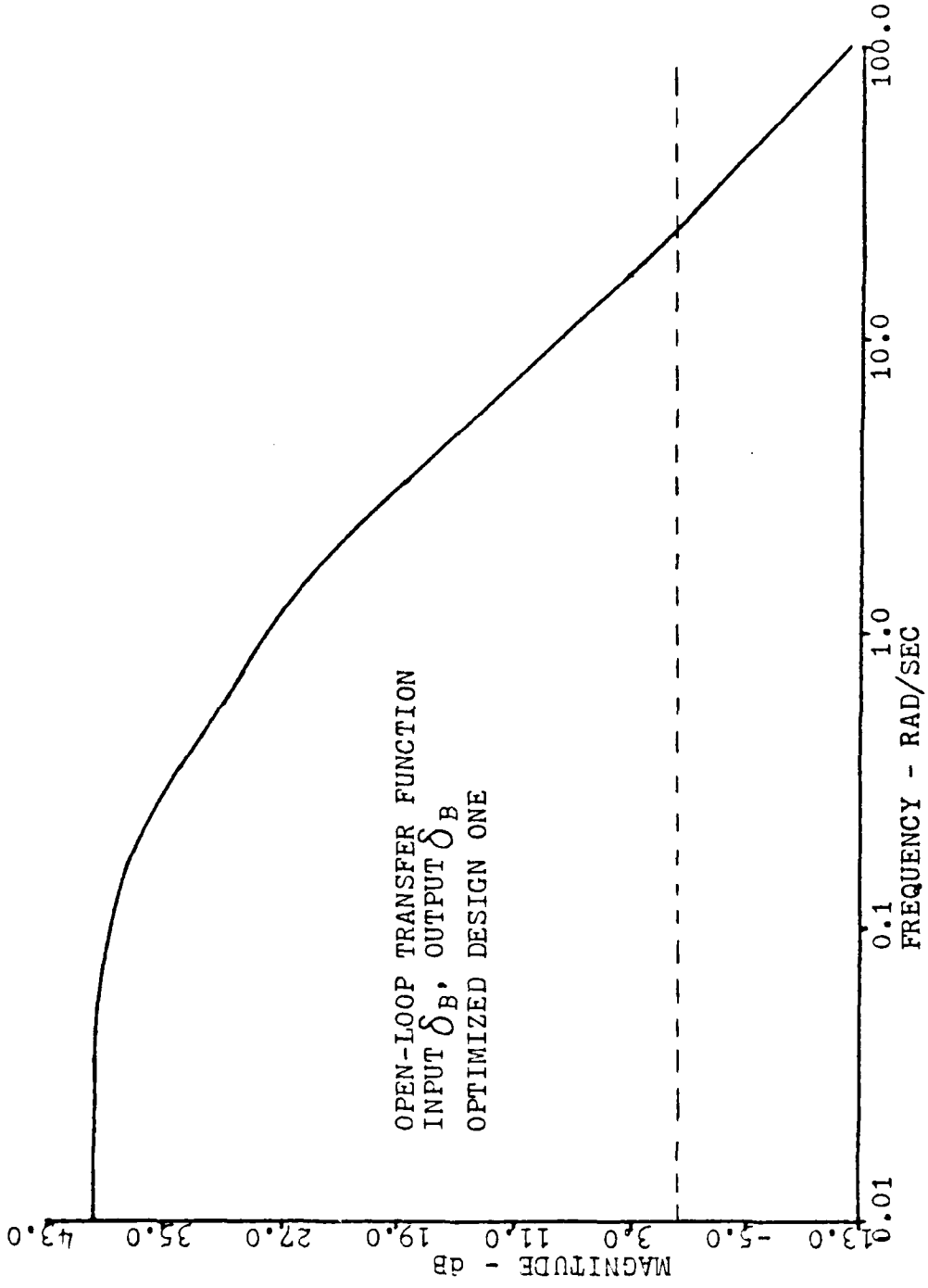


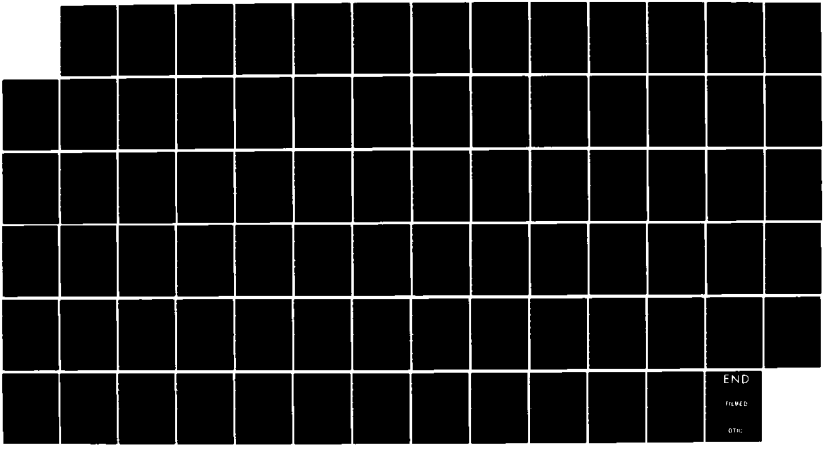
Figure 7.7 Transfer Function δ_B-δ_B Design One , Optimized.

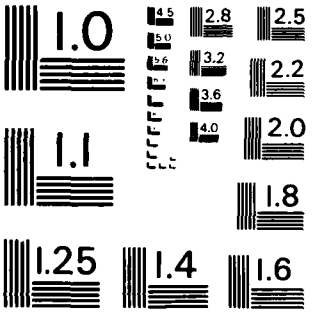
100 rad/sec to about 30 rad/sec. The reduction of bandwidth and gain in the loop δ_c to δ_B yields an increased tolerance to perturbation. The transfer function for δ_c to δ_B is shown in figures 7.8 and 7.9. In the transfer functions for δ_B to δ_c the bandwidth is slightly increased from 0.6 to 0.8 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 $-6 \pm 3j$. 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

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

UNCLASSIFIED F/G 12/1 NL





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

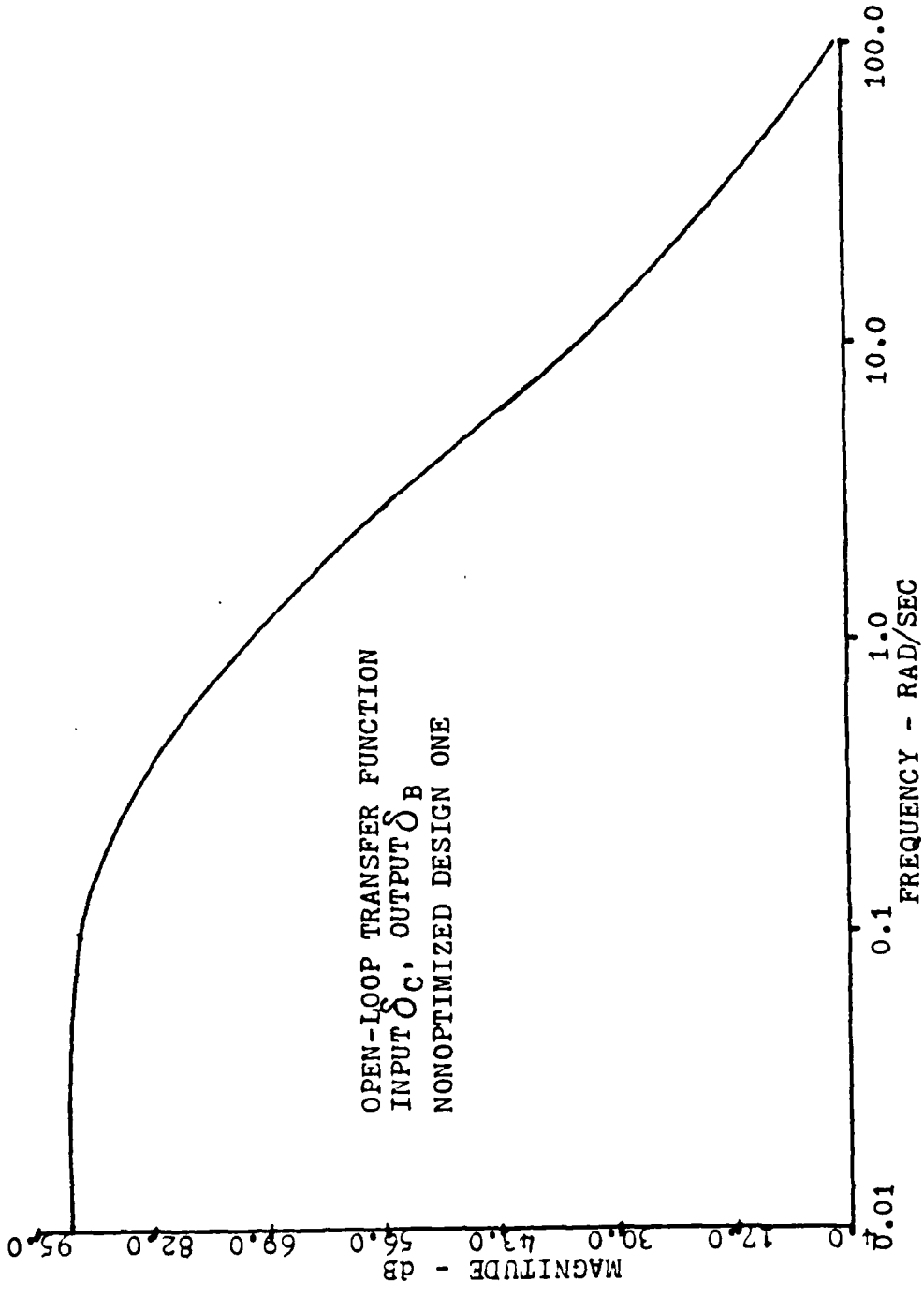


Figure 7.8 Transfer Function δ_C-δ_B Design One, Nonoptimized.

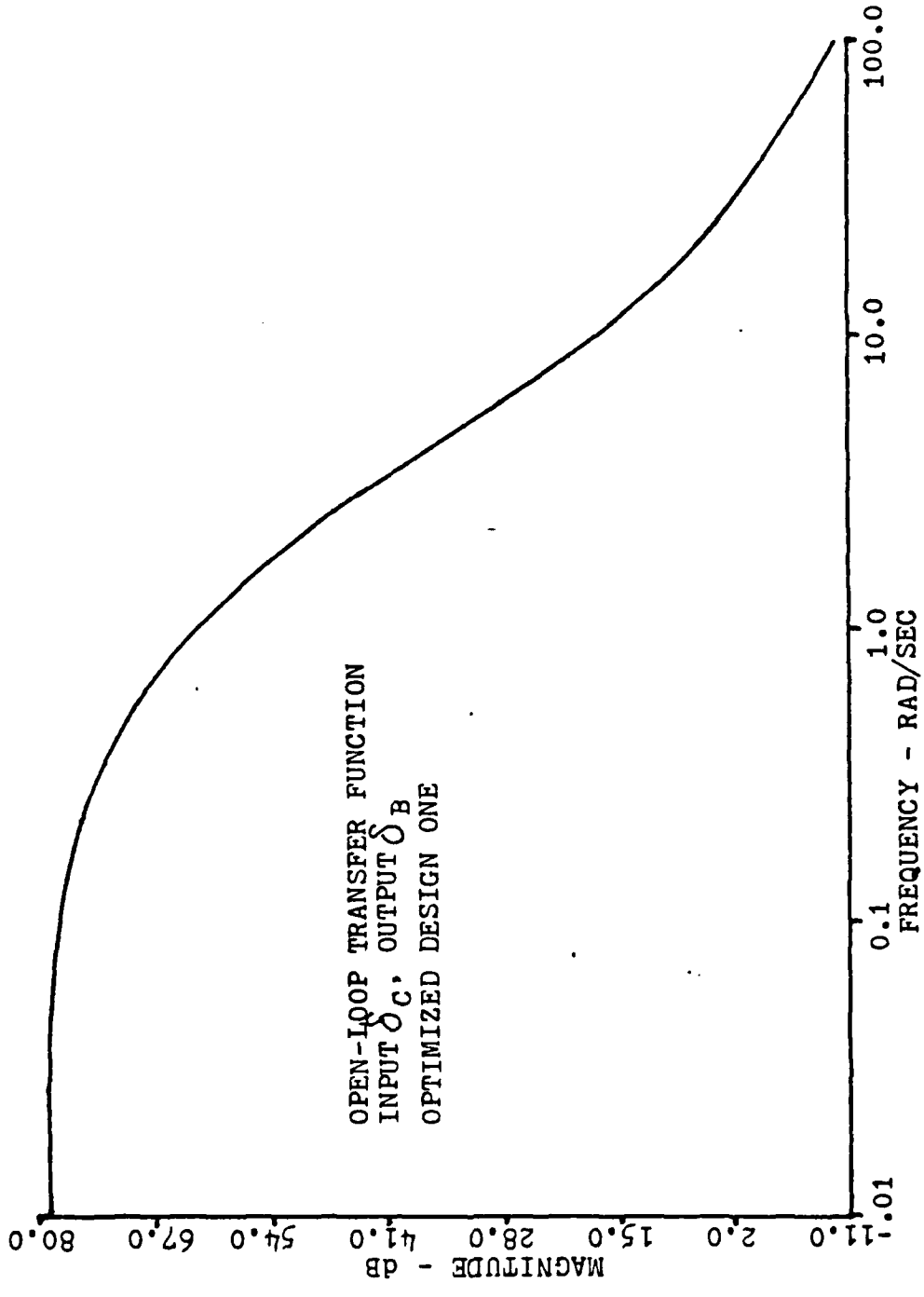


Figure 7.9 Transfer Function $\delta_c - \delta_B$ Design One, Optimized.

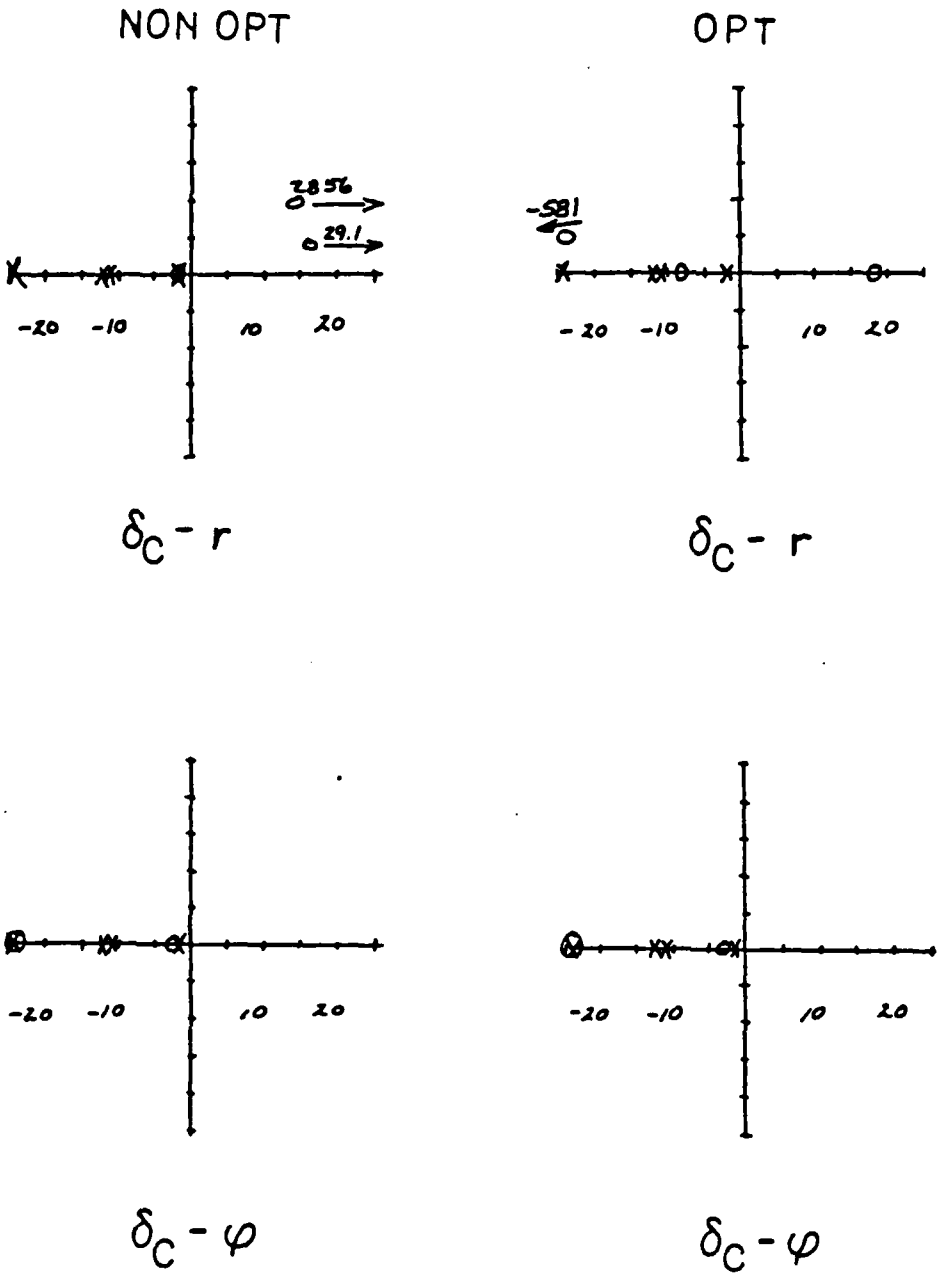


Figure 7.10 Pole-Zero Plots for Design 1.

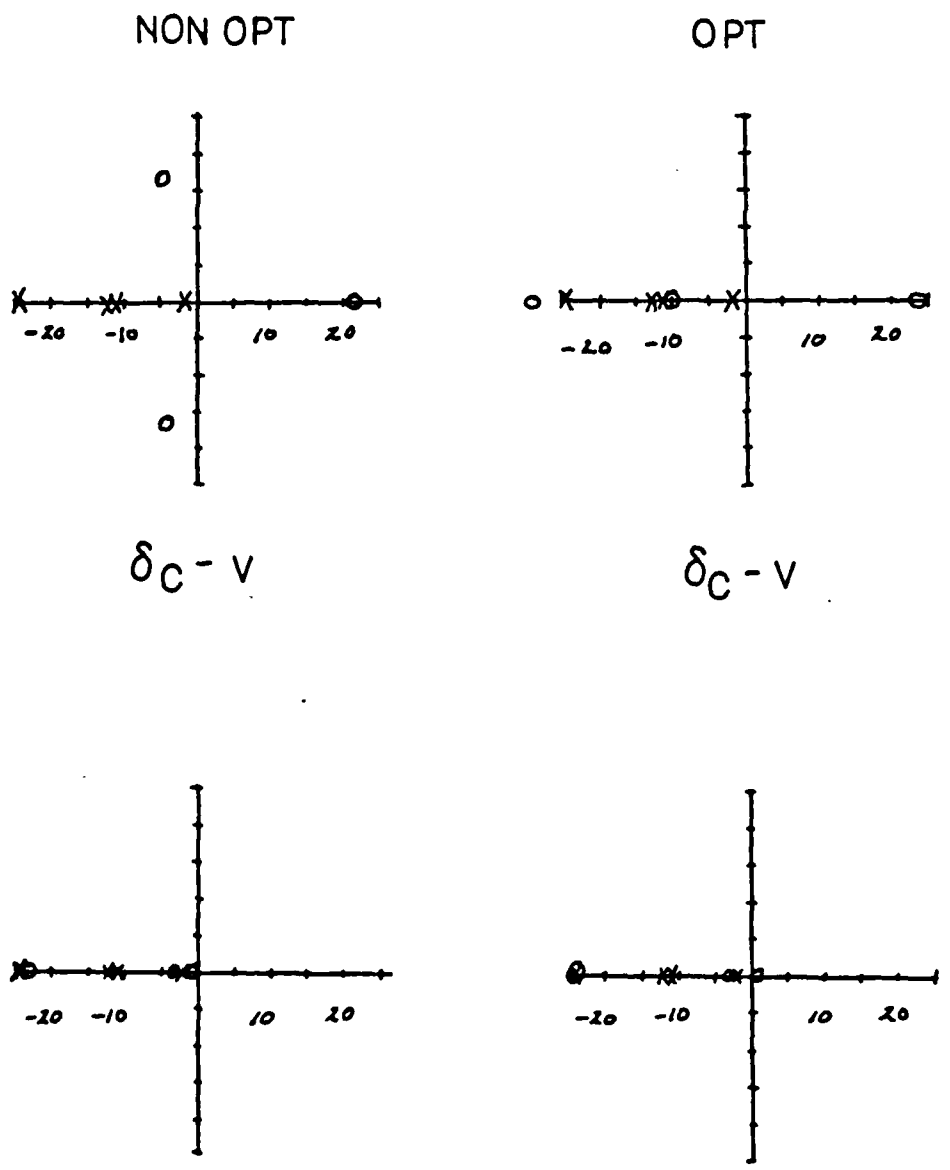


Figure 7.11 Pole-Zero Plots for Design 1 (cont.).

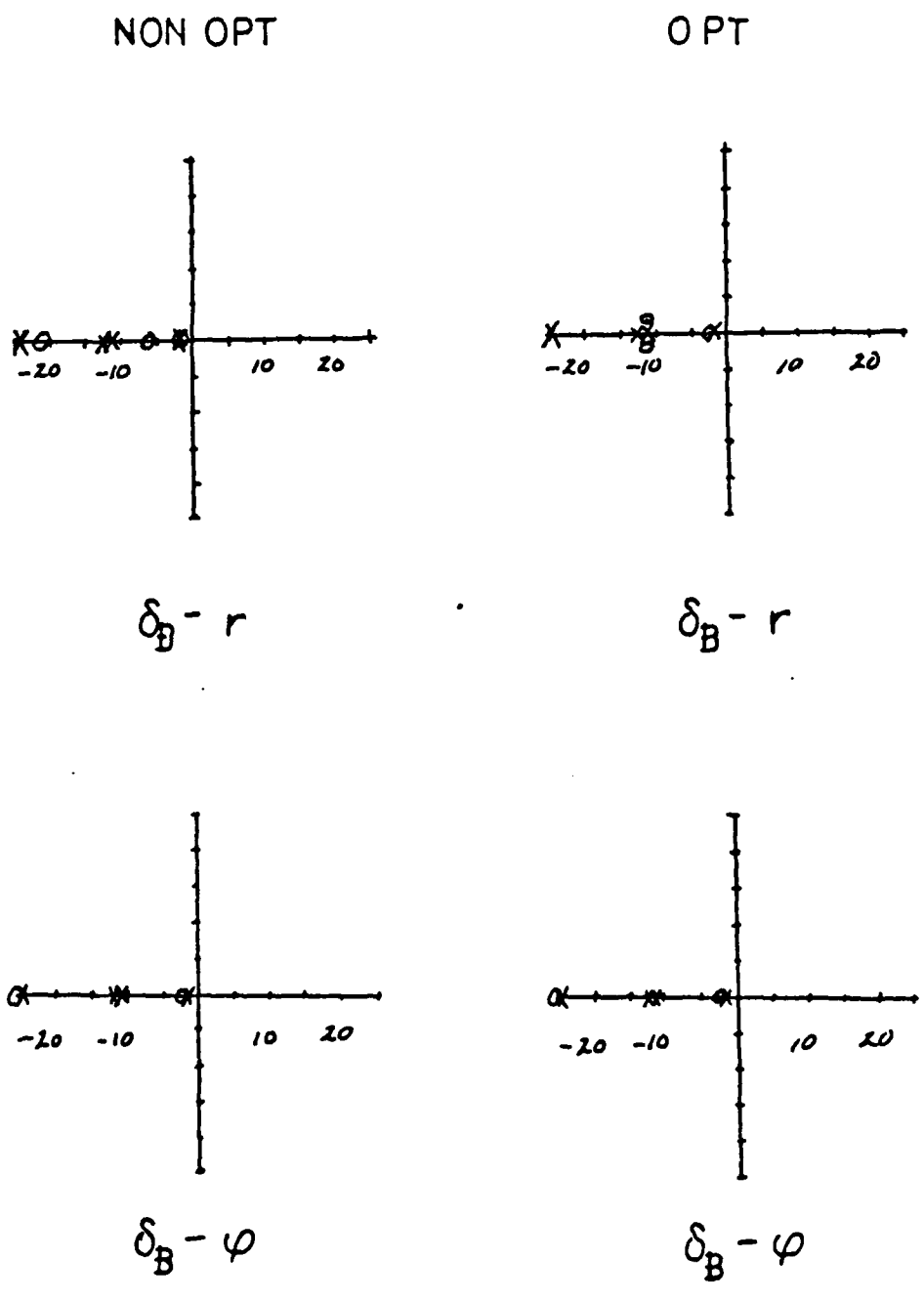


Figure 7.12 Pole-Zero Plots for Design 1 (cont.).

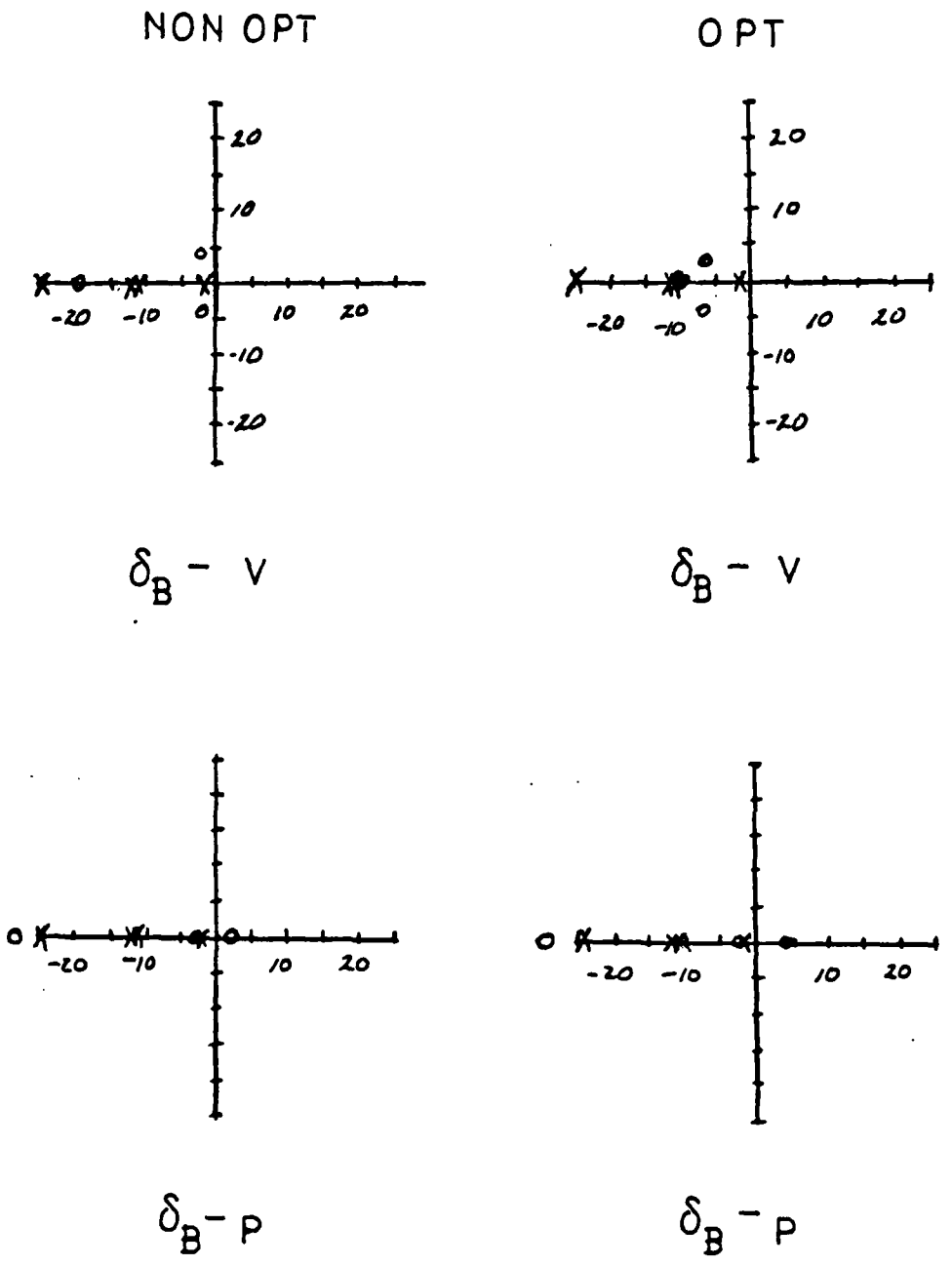


Figure 7.13 Pole-Zero Plots for Design 1 (cont.).

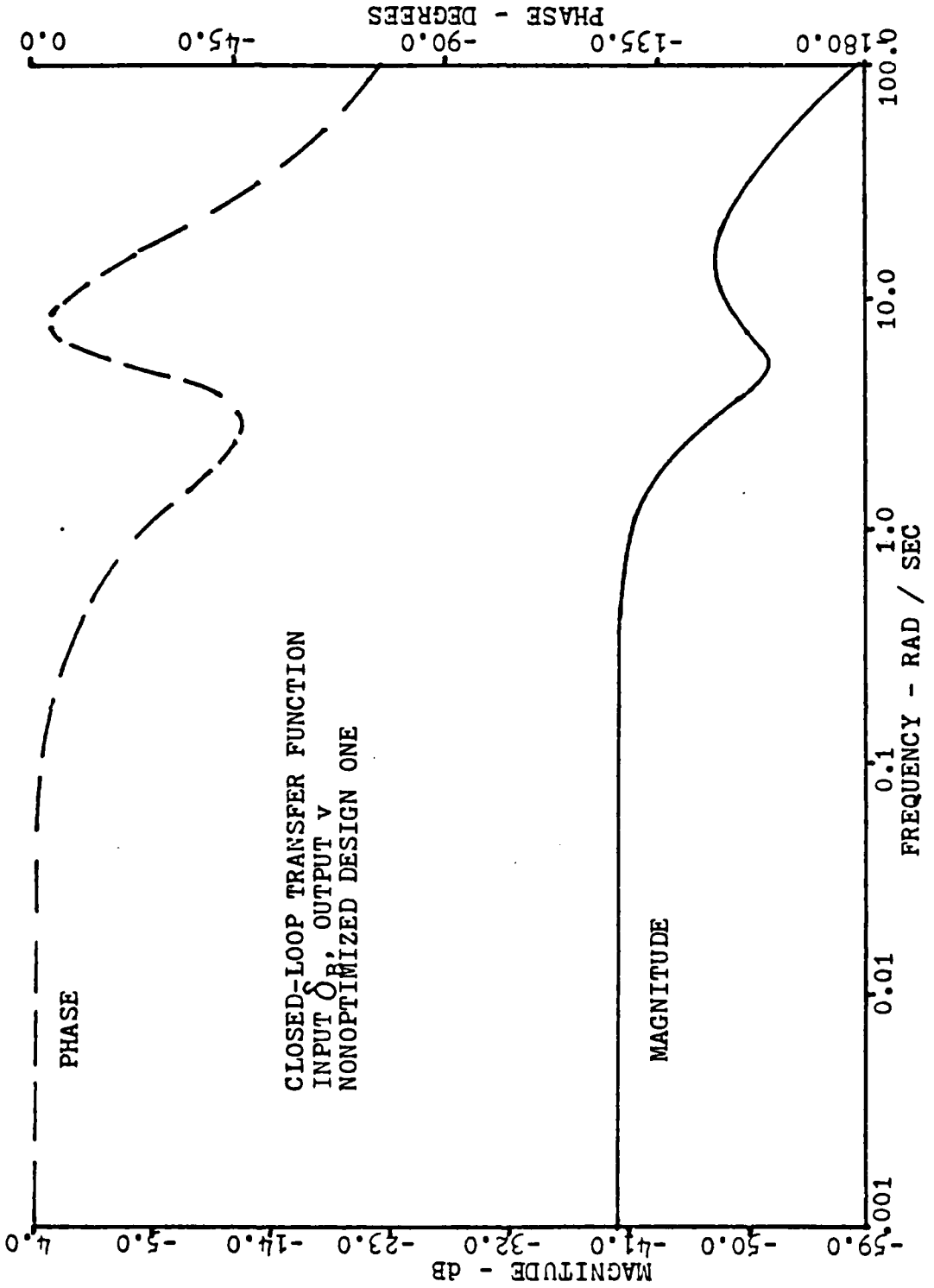


Figure 7.14 The δ_B to v Frequency Response, Nonopt..

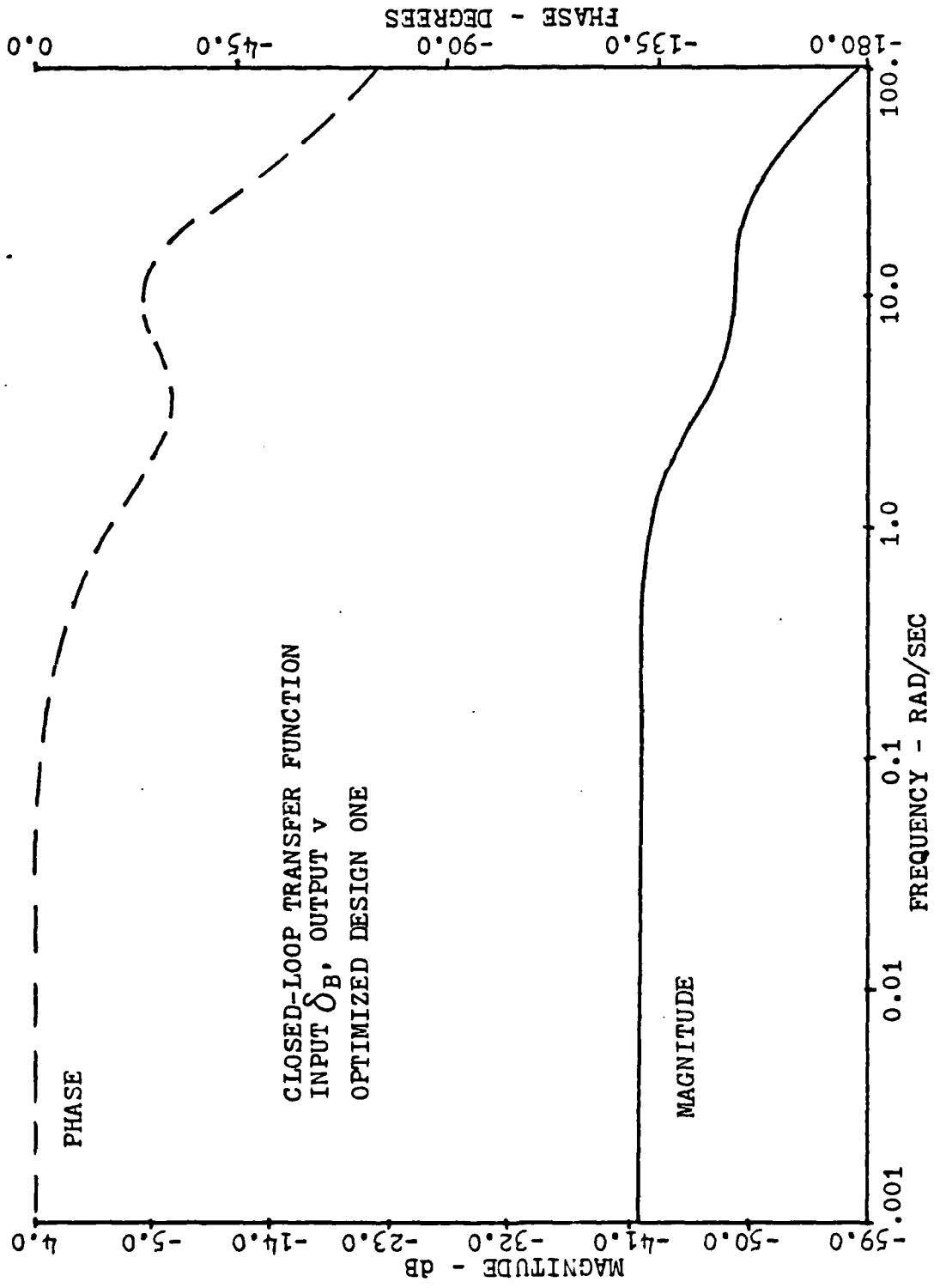


Figure 7.15 The δ_B to v Frequency Response, Optimized.

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 figures7.8 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].

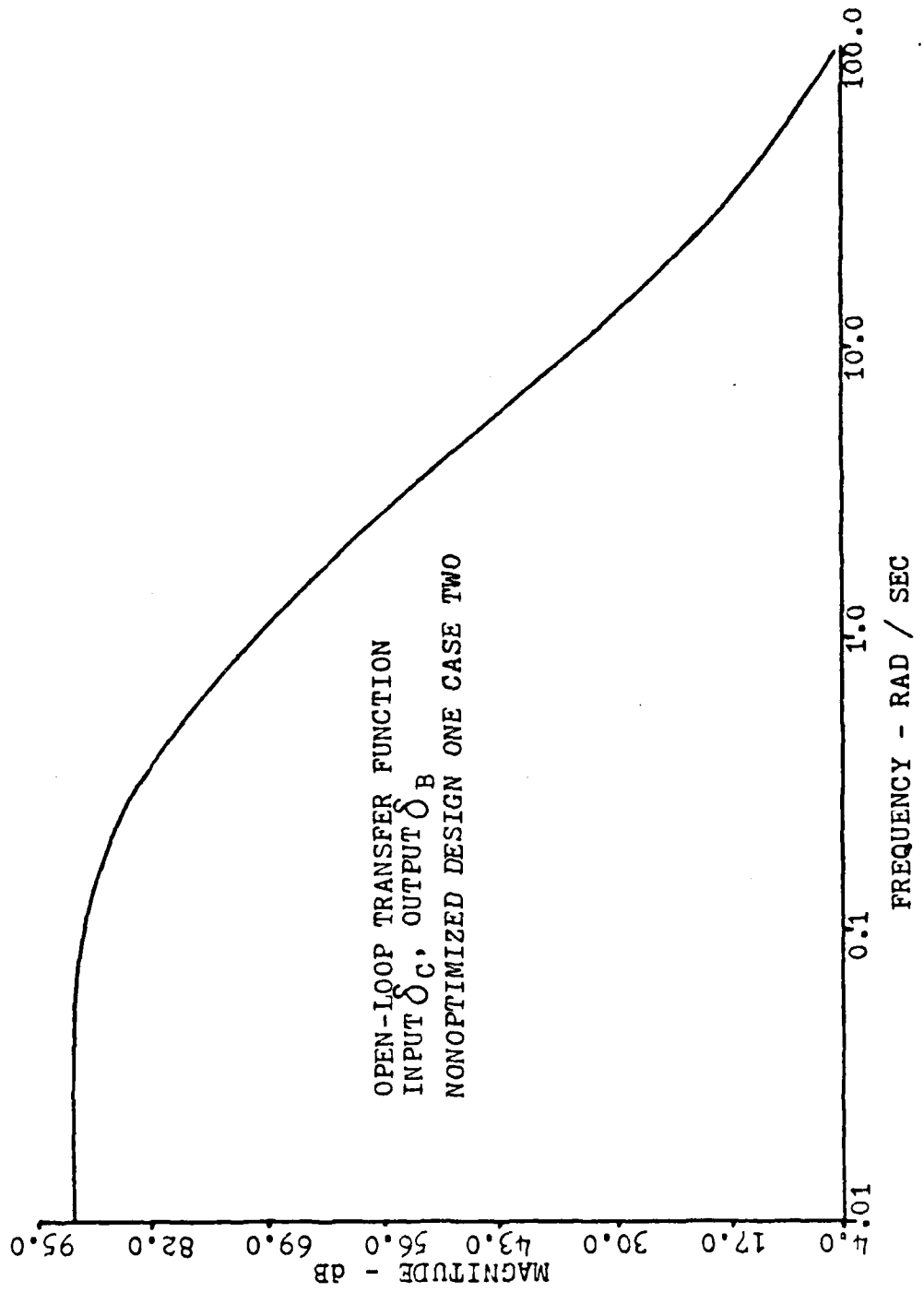


Figure 7.16 Transfer Function $\delta_c - \delta_B$ Design 1 Case2, Nonopt..

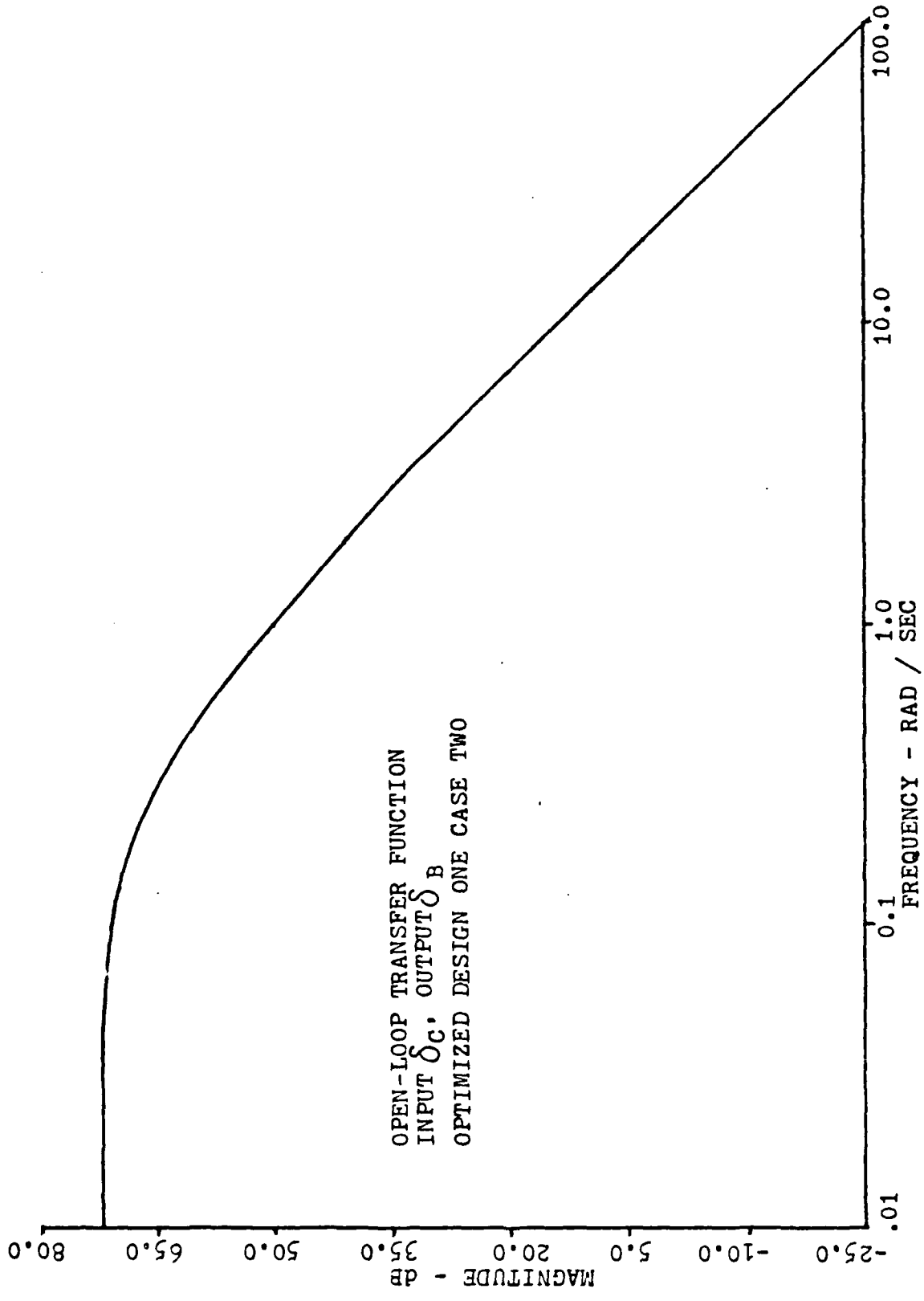


Figure 7.17 Transfer Function $\delta_c - \delta_B$ Design 1 Case 2, Opt.

Design two which is the design that is subject to parameter uncertainty in the δ_B 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_B 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_B 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

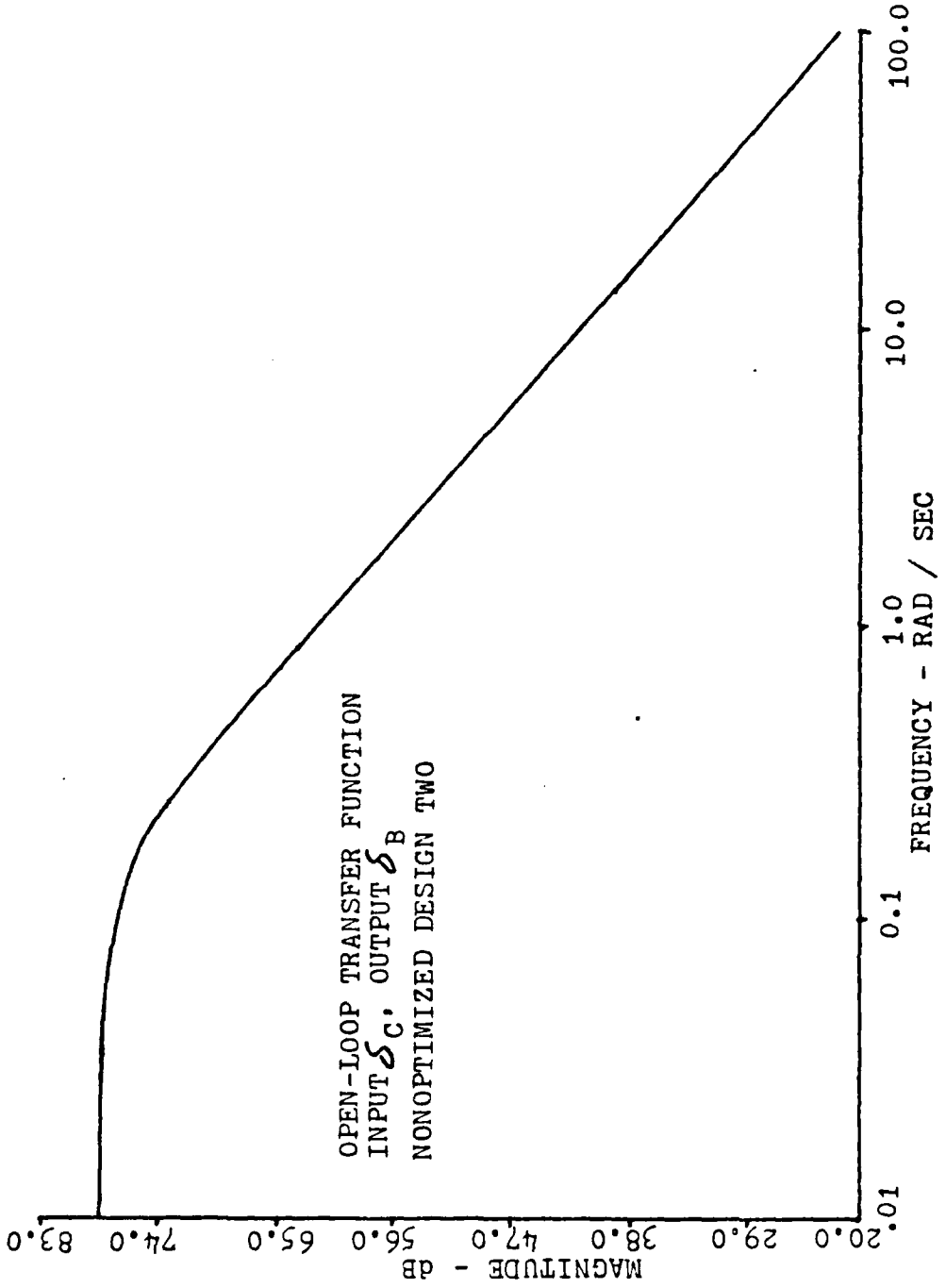


Figure 7.18 Transfer Function $\delta_c - \delta_B$ Design Two, Nonopt..

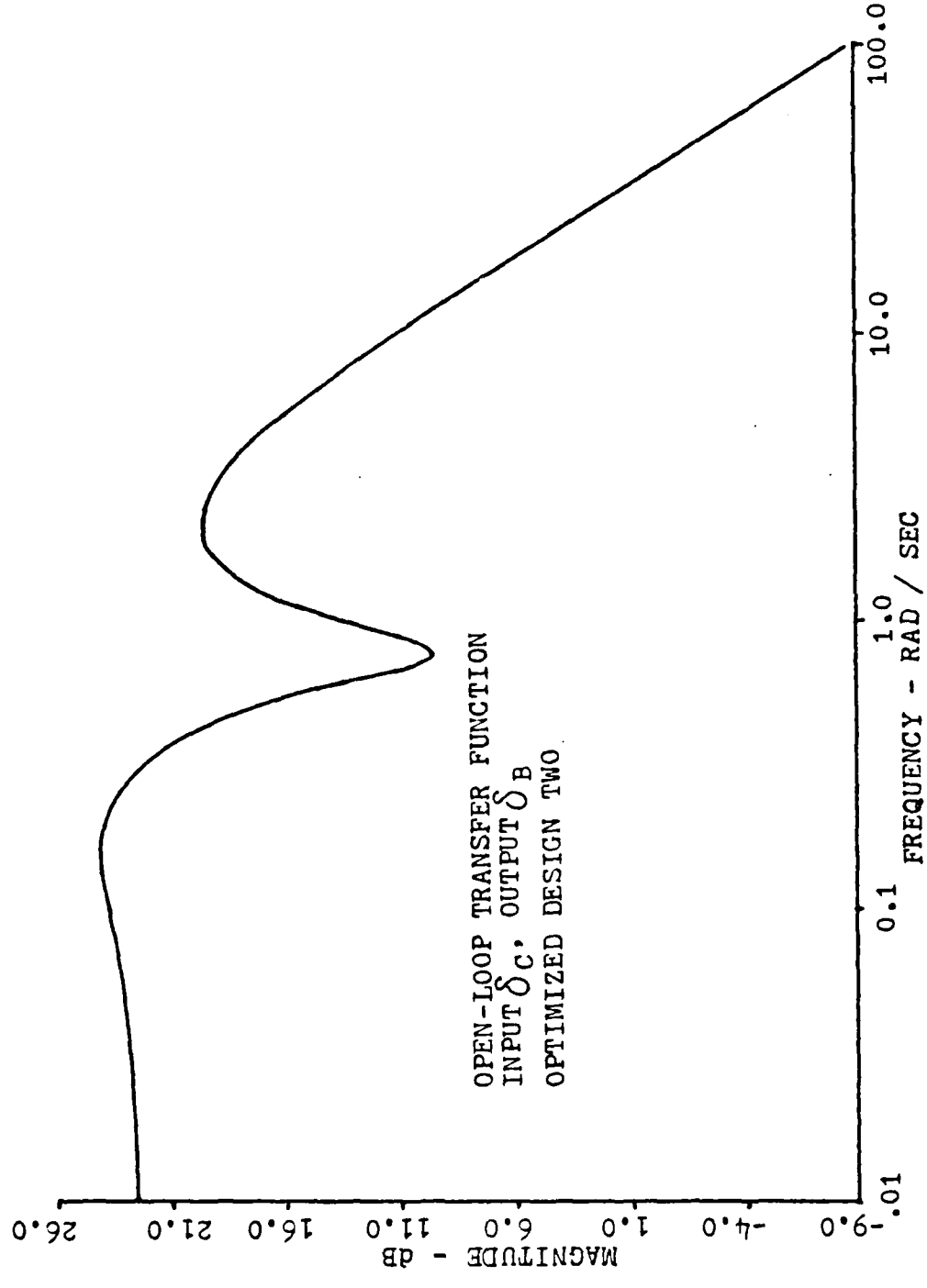


Figure 7.19 Transfer Function $\delta_c - \delta_B$ Design Two, Optimized.

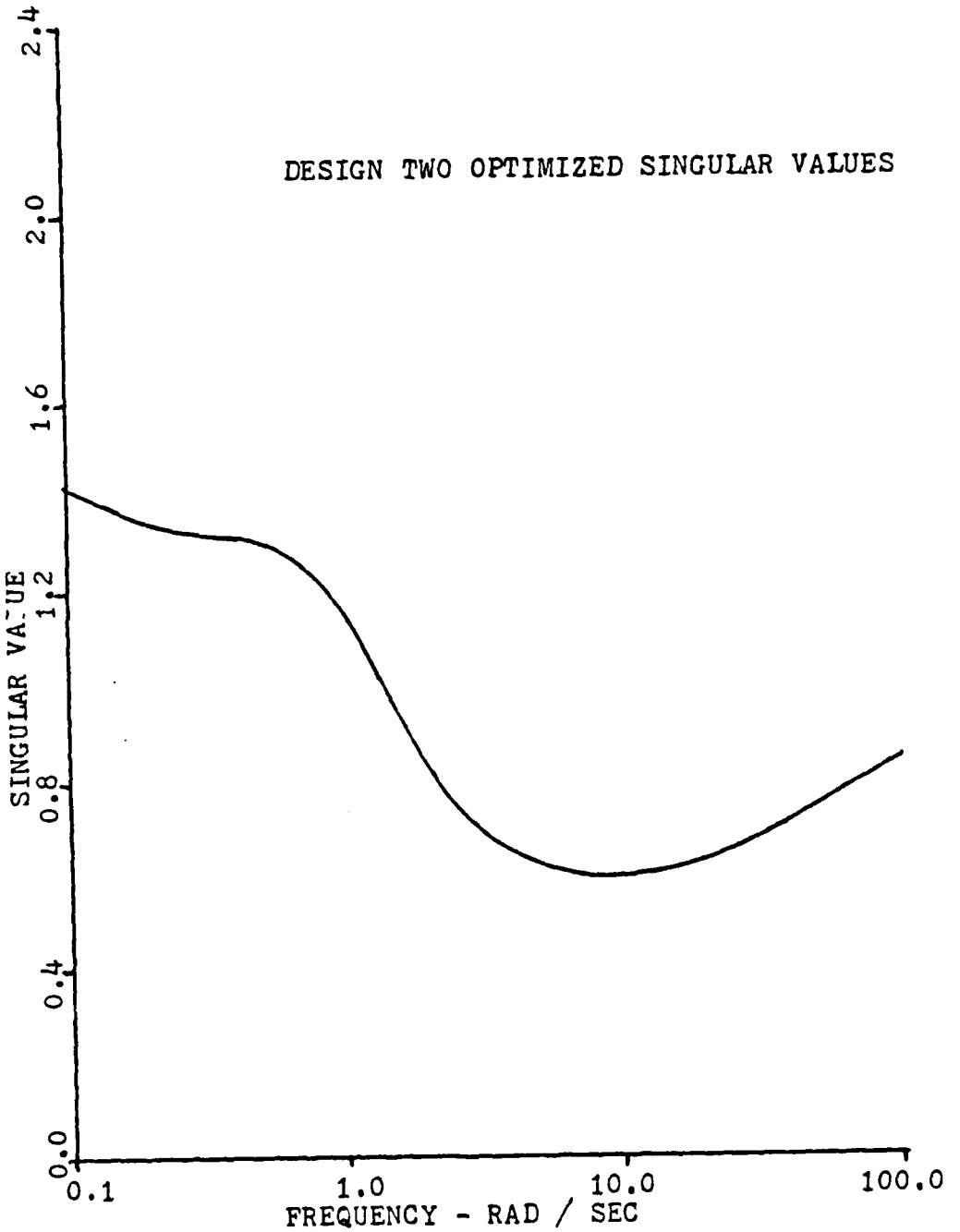


Figure 7.20 Singular Value Plot Design Two.

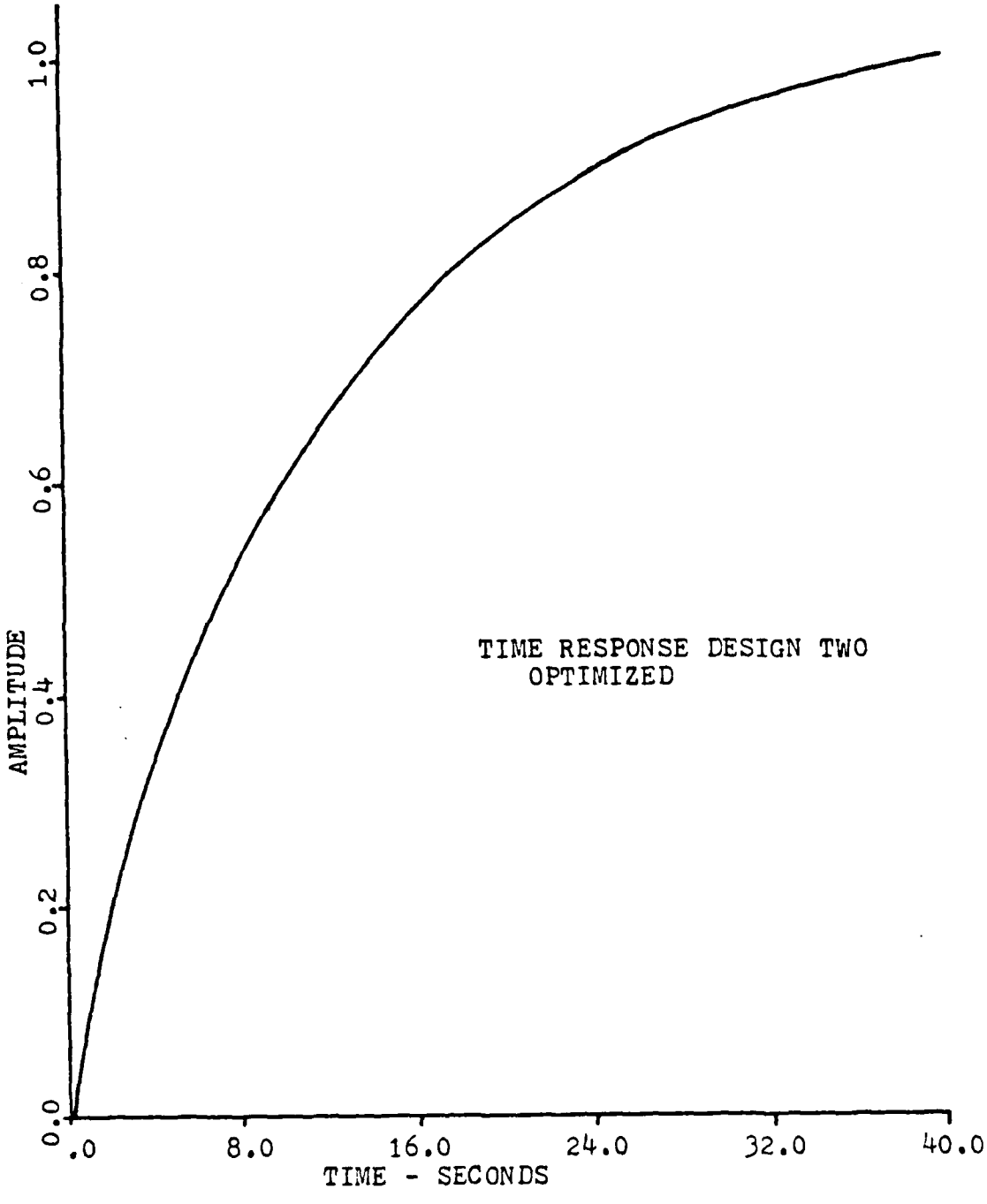


Figure 7.21 Time Response Design Two.

and have been shown tc be an alternative indicator of channels that may be affected by cross-feed perturbations. The pole-zero diagrams of the closed-loop transfer functicns 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.

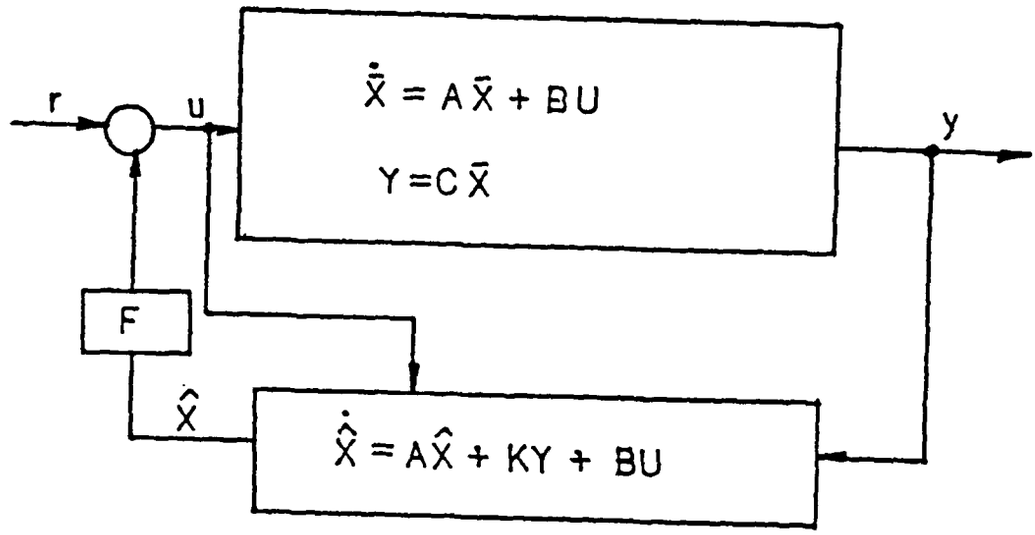


Figure 8.1 Simple Observer.

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

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

$$A_c = A - KC$$

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

locations and then a second optimization run with set feedback 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 of 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,

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

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

where $E(\bar{\gamma}) = E(\eta) = 0$ and $E(\bar{\gamma}(t)\bar{\gamma}(\tau)) = E(\eta(t)\eta(\tau)) = \delta(t-\tau)$ 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

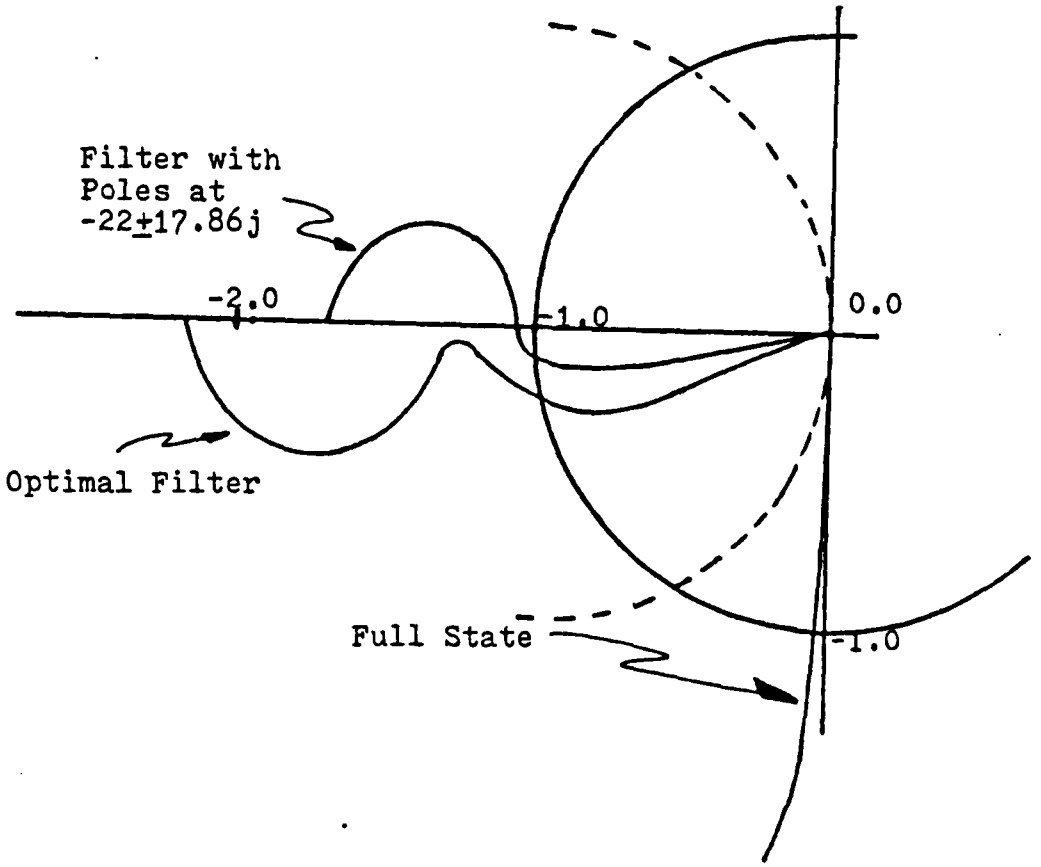


Figure 8.2 Nyquist Plot.

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±2j	- 6.75	15	97 -163	-163 277	221 -613	-613 2070	30 -60
Fast Filter Adjustment Procedure	-22±17.9j	- .98	10	6280 -12200	-12200 23800	130 -613	-613 8520	720 -1400
Fictitious Noise Adjustment Procedure $q^2 = 100$	-4.3 -13.1	- 7.73	79	107 -184	-184 319	236 -613	-613 1810	26.8 -40.2
$q^2 = 500$	-2.9 -24	-10.9	33	163 -301	-301 564	268 -613	-613 1500	20.4 -17.7
$q^2 = 10^3$	-2.5 -33	-13.9	42	204 -385	-385 743	285 -613	-613 1360	16.7 -1.9
$q^2 = 10^6$	-2.1 -100	-37	74	290 -570	-570 1170	317 -613	-613 1200	6.9 84.6

POPLAR CASE I	1.00018 -2.02742	GM=-5.5 dB PM=117 deg
POPLAR CASE II	18.22148 32.59476	GM=-19 dB 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

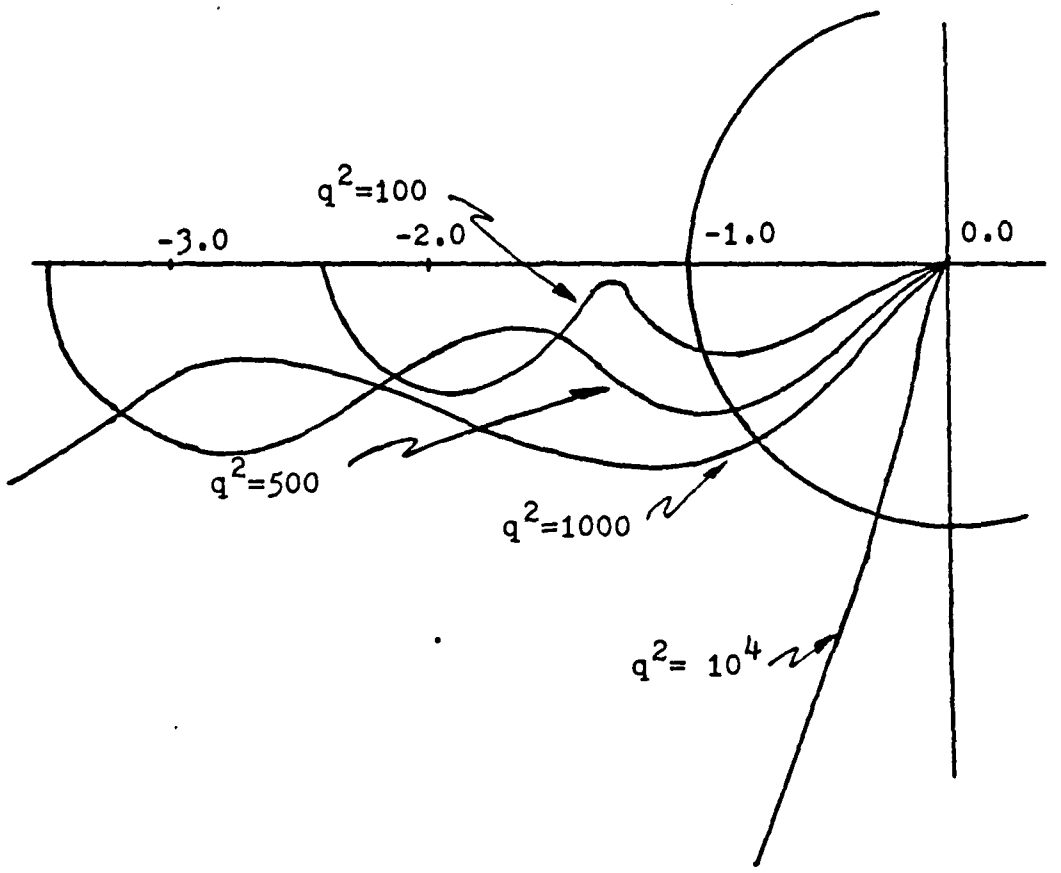


Figure 8.3 Nyquist Plot for Robustness Recovery.

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

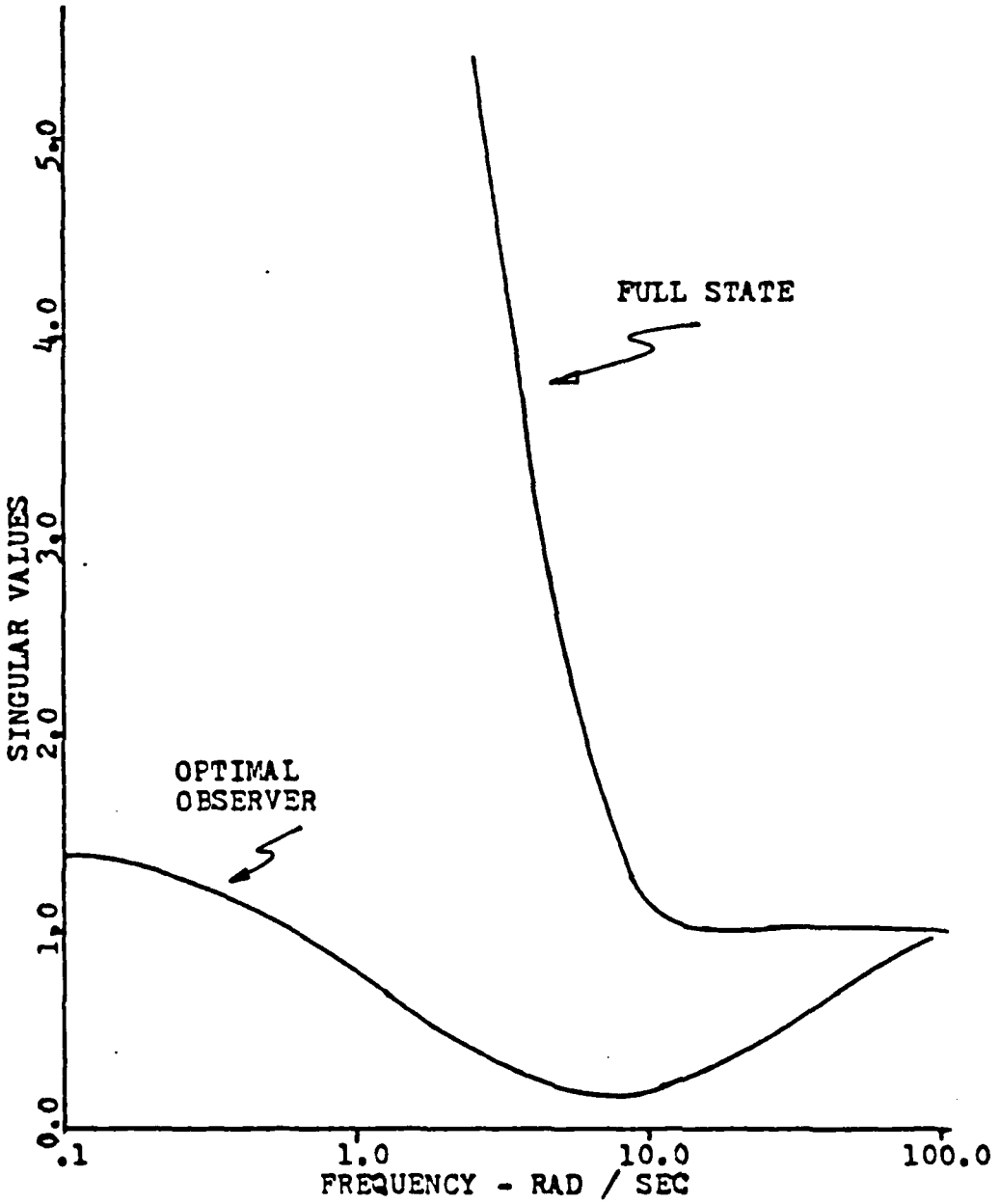


Figure 8.4 Singular Values of Observer System.

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 $-1.99 \pm j\ .001$, 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

$$\begin{matrix} 306.5 & -456.5 \\ -456.5 & 700.1 \end{matrix}$$

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

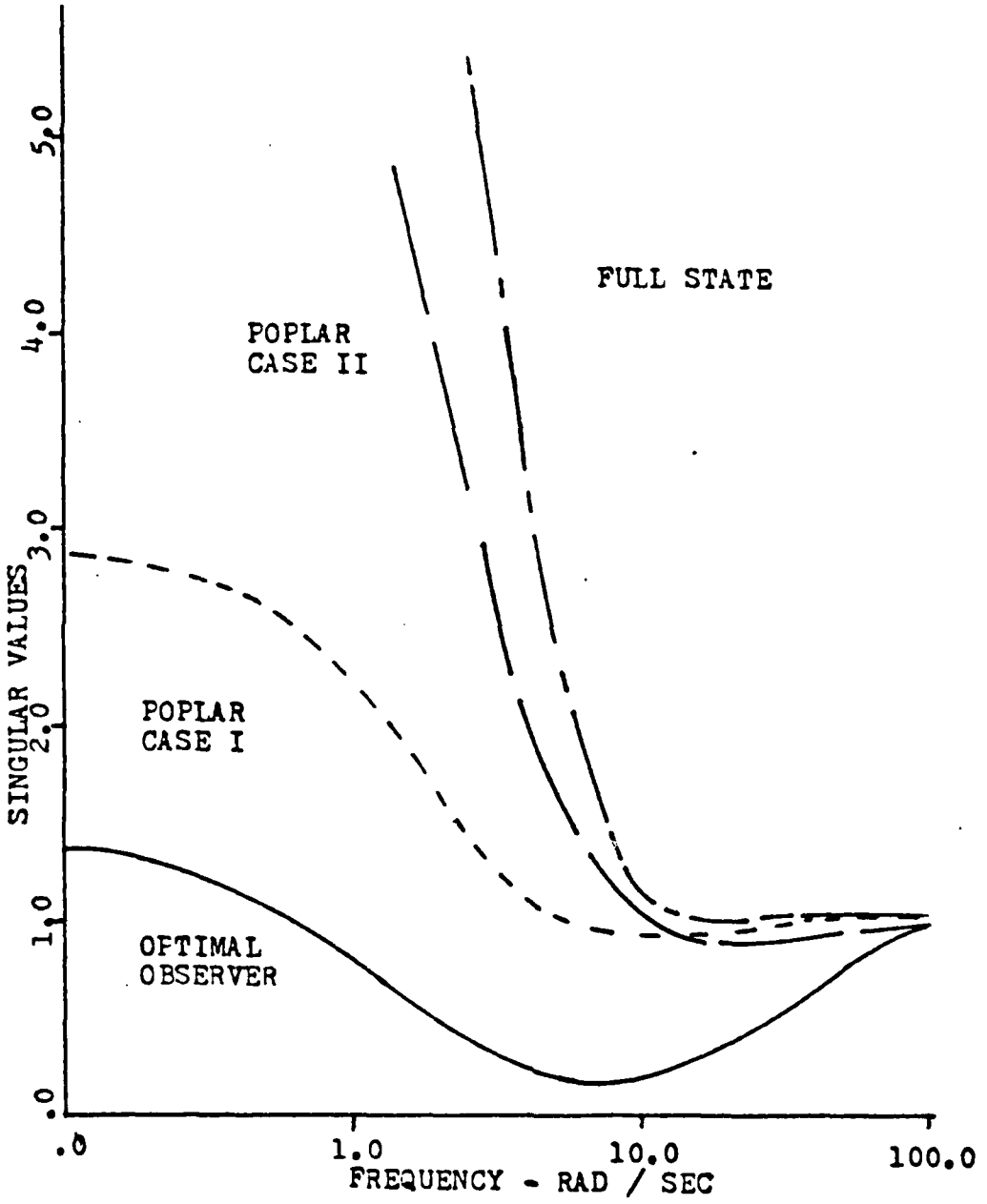


Figure 8.5 Singular Value Comparison Plot.

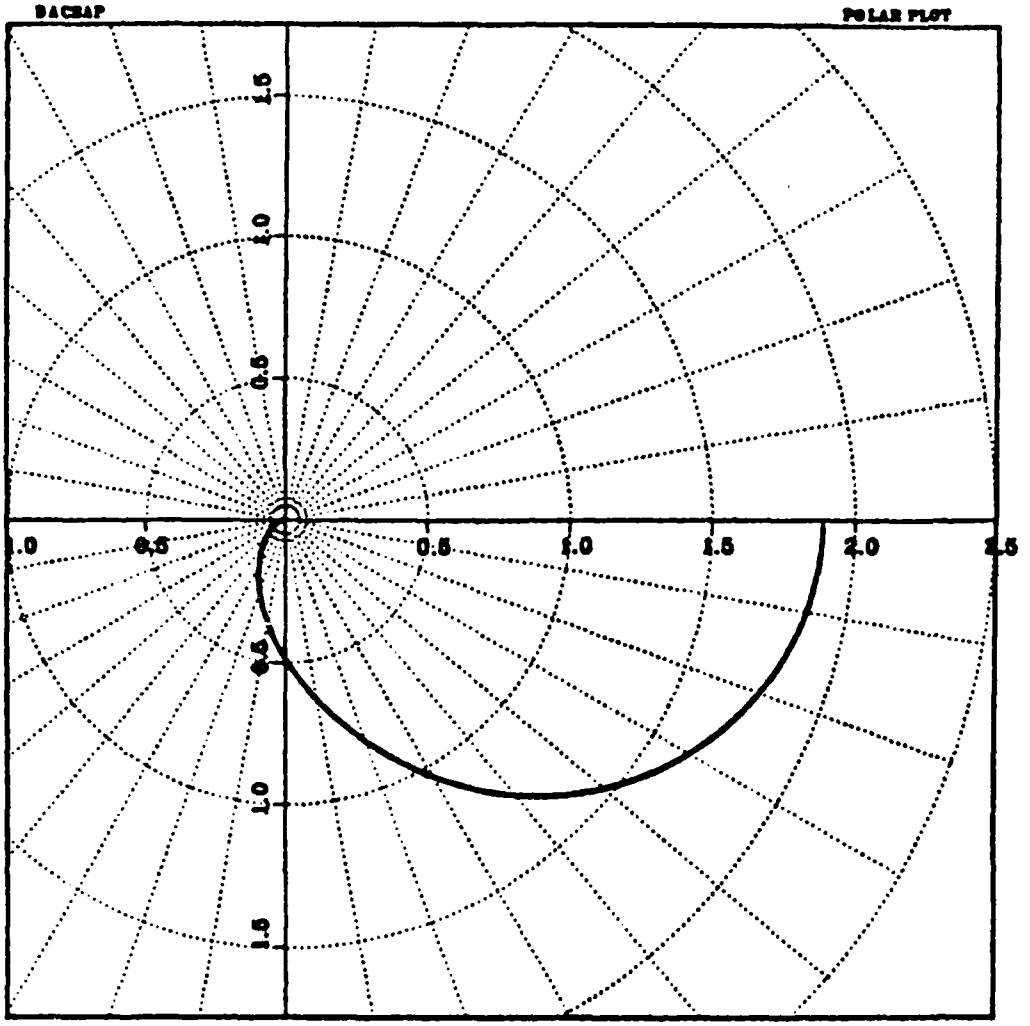


Figure 8.6 Nyquist for Computed Robustness Recovery.

which corresponds to the same level of pole movement found when q^2 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 of 58 degrees. The gain margin was about -19 db. Even though some of the constraint conditions on the design were not met the design demonstrates excellent robustness recovery.

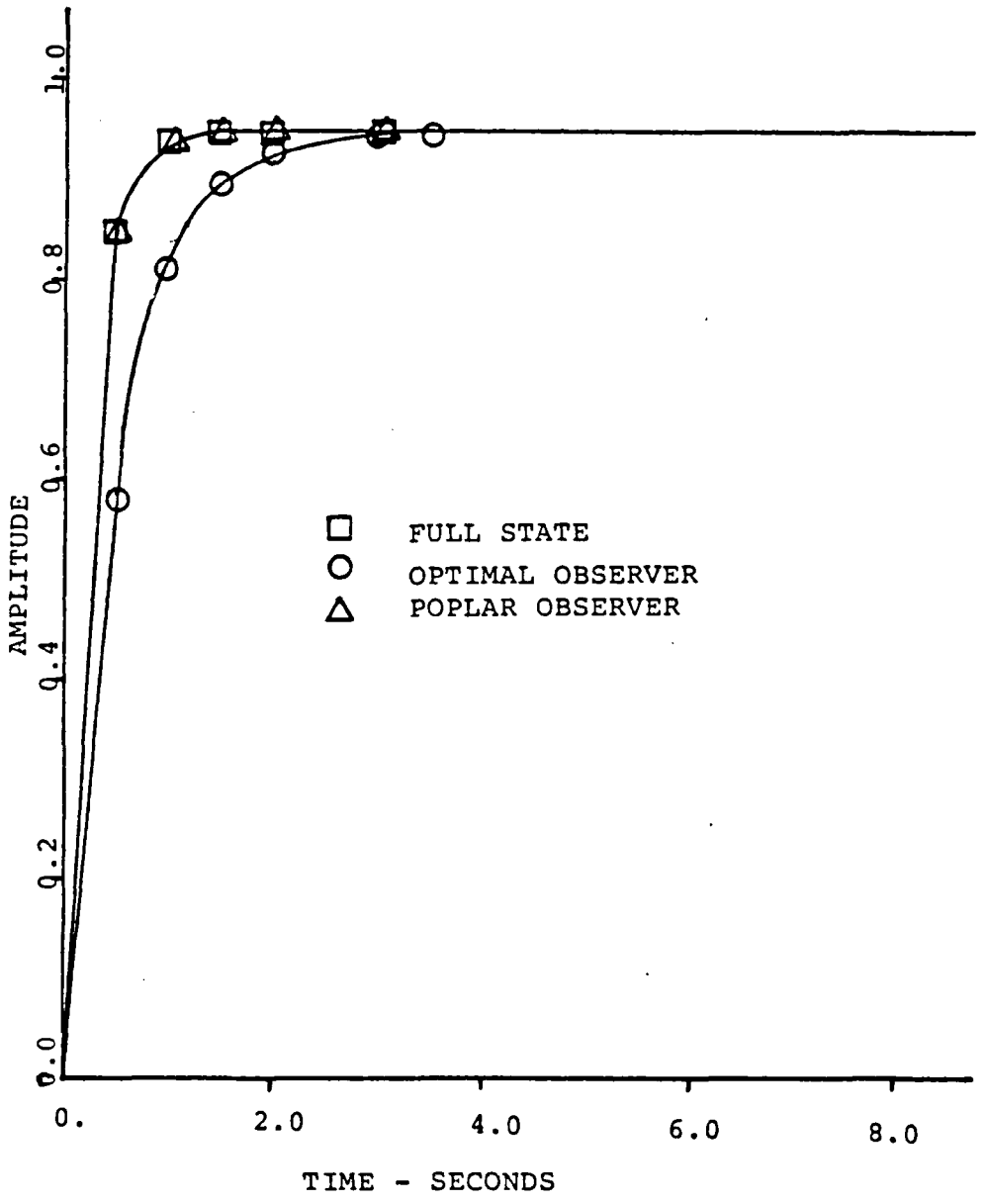


Figure 8.7 Time Response Plot for Simple Observer.

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

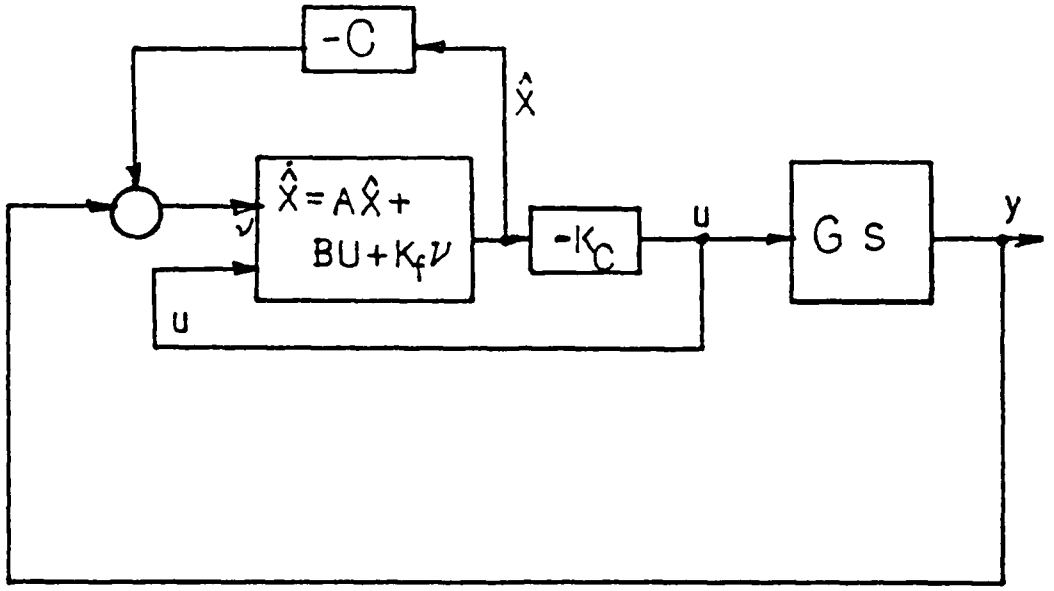


Figure 9.1 Observer Based Controller.

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

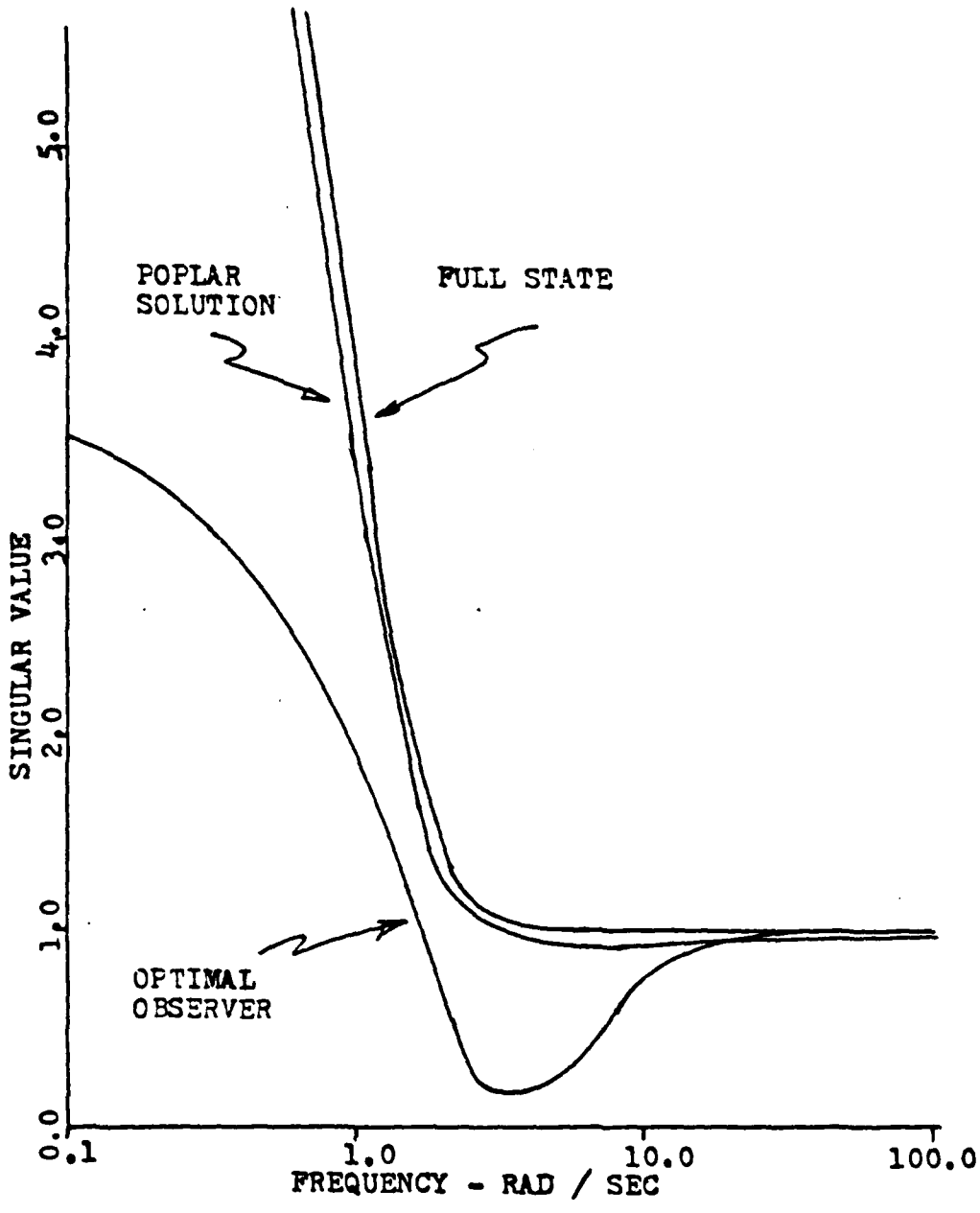


Figure 9.2 Singular Value Plot of Observer Results.

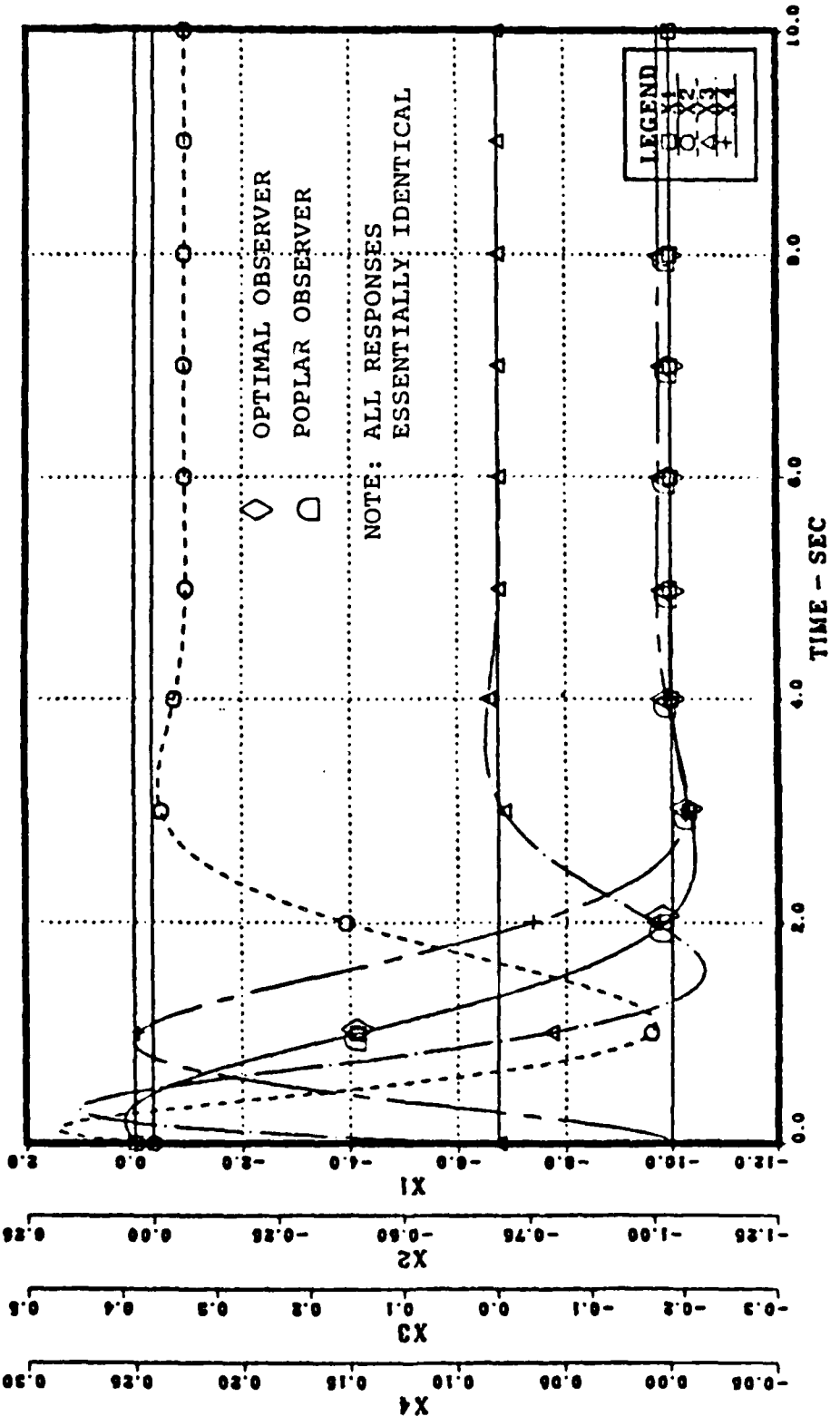


Figure 9.3 Time Response for Observer System.

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

recovery. By providing direct gain adjustment the pole placement and robustness procedure results in a practical design with relatively low observer gains and good performance. 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

of 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.

```
C     ****************************************************************** CONOCC10
C               PPPPPPP  OOOOOO PPPPPP L          AAAAAAA RRRRRR          CCN00020
C               P     P  O    O P     P L          A     A R     R         CCN00030
C               PPPPPPP  O    O PPPPPP  L          AAAAAAA RRRRRR          CON00040
C               P        O    O P       L          A     A R   R           CCN00050
C               P        OOOOOO P       LLLLLLL    A     A R     R         CONOCC60
C                                                                          CONOCC70
C                                                                          CONOCC80
C            POLE PLACEMENT AND ROBUSTNESS DESIGN PROGRAM                  CONOCC90
C*****************************************************************       CON00100
C     * THIS PROGRAM IS DESIGNED TO EMPLOY SINGULAR VALUE ANALYSIS AND  *  CCN00110
C     * THE USE OF AN OPTIMIZATION ROUTINE TO AID IN POLE PLACEMENT     *  CONO0120
C     *  CONTROL SYSTEM DESIGN FOR LINEAR MULTIVARIABLE SYSTEMS.        *  CCN00130
C     *  ROBUSTNESS OF THE SYSTEM IS ALSO CONSIDERED BY ESTABLISHING    *  CCN00140
C     *  SINGULAR VALUE LEVELS WHICH CORRESPOND TO MULTILOOP GAIN       *  CON00150
C     *  AND PHASE MARGINS DETERMINED FROM THE UNIVERSAL GAIN PHASE     *  CON00160
C     *  DIAGRAM DEVELOPED BY NEWSOM AND MUKHAPADHYAY AT NASA LANGLEY.  *  CONOC170
C     *            VERNON C GORDON , CDR , USN                          *  CON00180
C     *             VERSION 1.1 , 8-10-84                               CCN0C190
C     ****************************************************************** CONOC200
C     *                                                                *  CCN00210
C     *                                                                *  CONOC220
C     * THIS FIRST BLOCK SETS DIMENSIONS AND DECLARES REAL AND         *  CON00230
C     *  COMPLEX FUNCTIONS. THE FOLLOWING IS A LIST OF PROGRAM SYMBOLS: *  CCN00240
C     ****************************************************************** CONOC250
C     ****************************************************************** CCN00260
C     *                                                                *  CCNOC270
C     *              REAL VARIABLES BY NAME AND BRIEF DESCRIPTION      *  CCN00280
C     *     THE SYSTEM UNDER CONSIDERATION IS GIVEN AS:                *  CCN0C290
C     *         XDOT = A*X + B*U ; Y= C*X    , FOR STATE OR            *  CUN0C300
C     *           OUTPUTFEEDBACK OF THE FORM U=-FX +R OR U=-FY + R     *  CON00310
C     *      A   -  THE PLANT A MATRIX (NSTATE X NSTATE)               *  CCN00320
C     *      B   -  THE CONTROL MATRIX  (NSTATE X NCONTROL)            *  CONOC330
C     *      C   -  THE PLANT OBSERV. OR OUTPUT MATRIX (NOUTPUT X NSTATE; *  CCN00340
C     *               FOR STATE FEEDBACK PROBLEMS)                     *  CCNOC350
C     *      F   -  THE FEEDBACK GAIN MATRIX(NCONTROL X NSTATE;        *  CONOC360
C     *               FOR STATE FEEDBACK PROBLEMS)                     *  CCN00370
C     *      REALMU - THE INPUT DESIRED REAL POLE LOCATION             *  CONOC380
C     *      IMAGMU - THE INPUT DESIRED IMAGINARY POLE LOCATION        *  CONOC390
C     *      FC   -  F*C                                               *  CCN00400
C     *      BFC  -  B*F*C                                             *  CUNOC410
C     *      AMBFC - A - B*F*C                                         *  CCN00420
C     *      REIG - REAL COMPUTED EIGENVALUE OF THE SYSTEM             *  CCNOC430
C     *      WW   - PROGRAM WORK SPACE                                 *  CONOC440
C     *      WK - WORK SPACE USED BY SUBROUTINES                       *  CCN00450
C     *      WA - DIMENSIONED WORK SPACE                               *  CONOC460
C     *      OMEGA - FREQUENCY                                         *  CCN00470
C     *      SV - SINGULAR VALUE OF (I + F * PLANT TRANSFER FUNCTION(G)) *  CCNOC480
```

```
C     *      SVMI - SINGULAR VALUE OF (I + INV(F* G ))                    * CONOC490
C     *      SVAC - SINGULAR VALUE OF ( I + G * F)                        * CONOC500
C     *      SVMC - SINGULAR VALUE OF ( I + INV (G * F))                  * CONOC510
C     *               ONLY OF INTEREST IF NO. OUTPUTS = NO. INPUTS.       * CONOC520
C     *      SIGNM1 - FIRST SINGULAR VALUE OF (I + F*G)                   * CONOC530
C     *      SIGNM2 - SECOND SINGULAR VALUE OF (I + F*G)                  * CONOC540
C     *      SIGNMX - MAXIMUM SINGULAR VALUE OF (I + F*G)                 * CONOC550
C     *      SIGPRO - SQRT(FIRST SING VAL * SECOND SING VAL)              * CONOC560
C     *      SVACMO - MIN ADDITIVE OUTPUT SINGULAR VALUE (I + G*F)        * CONOC570
C     *      SVACXO - MAX ADDITIVE OUTPUT SINGULAR VALUE                  * CONOC580
C     *      SVMIM - MIN MULTPLICATIVE INPUT SINGULAR VALUE (I+INV(FG))   * CONOC590
C     *      SVMIX - MAX MULTIPLICATIVE INPUT SINGULAR VALUE              * CONOC600
C     *      SVMMC - MIN MULTIPLICATIVE OUTPUT SINGULAR VALUE (I+INV(GF)) * CONOC610
C     *      SVMXO - MAX MULTIPLICATIVE OUTPUT SINGULAR VALUE             * CONOC620
C     *                                                                   * CONOC630
C     ********************************************************************* CONOC640
      REAL*4 A(10,10),B(10,10),C(10,10),F(10,10),REALMU(50),IMAGMU(50),    CONOC650
     1FC(10,10),BFC(10,10),AMBFC(10,10),WK(5000),                          CONOC660
     2REIG(50),SV(10),WW(10,10),WA(100),                                   CONOC670
     3OMEGA(500),SIGNM1(500),SIGNM2(500),SIGNMX(500),SIGPRO(500),          CONOC680
     4SVACMO(500),SVACXO(500),SVMIM(500),SVMIX(500),                       CONOC690
     5SVMMO(500),SVMXO(500),SVMI(10),SVAC(10),SVMC(10)                     CONOC700
C     ********************************************************************* CONOC710
C     ********************************************************************* CONOC720
C     *          THE COMPLEX PARAMETERS USED IN THE PROGRAM               * CONOC730
C     *          AX - COMPLEX A                                           * CONOC740
C     *          BX - COMPLEX B                                           * CONOC750
C     *          CX - COMPLEX C                                           * CONOC760
C     *          FX - COMPLEX F                                           * CONOC770
C     *          EIG - EIGENVALUE OF A- B*F*C                             * CONOC780
C     *          Z - EIGENVECTOR OF A - B*F*C                             * CONOC790
C     *          MU - INPUT DESIRED POLE LOCATION                         * CONOC800
C     *          OMU - ORDERED VALUE OF MU                                * CONOC810
C     *          OEIG - ORDERED VALUE OF EIG                              * CONOC820
C     *          OZ - ORDERED EIGENVECTOR                                 * CONOC830
C     *          XI - IDENTITY MATRIX                                     * CONOC840
C     *          XXI - IDENTITY MATRIX                                    * CONOC850
C     *          LI - INPUT ADDITIVE SINGULAR VECTOR ASSOCIATED WITH SV   * CONOC860
C     *          VI - INPUT ADDITIVE SINGULAR VECTOR ASSOCIATED WITH SV   * CONOC870
C     *          UO - OUTPUT ADDITIVE SINGULAR VECTOR (SVAO)              * CONOC880
C     *          VO - OUTPUT ADDITIVE SINGULAR VECTOR (SVAO)              * CONOC890
C     *          UMI- INPUT MULTIPLICATIVE SINGULAR VECTOR                * CONOC900
C     *          VMI - INPUT MULTIPLICATIVE SINGULAR VECTOR               * CONOC910
C     *          UMO - OUTPUT MULTIPLICATIVE SINGULAR VECTOR              * CONOC920
C     *          VMO - OUTPUT MULTIPLICATIVE SINGULAR VECTOR              * CONOC930
C     *          FXPLT - F * PLANT TRANSFER FUNCTION (G)                  * CONOC940
C     *          XIPFXP - I + F*G                                         * CONOC950
C     *          PLAN - PLANT TRANSFER FUNCTION (G)                       * CONOC960
```

```
C     *          PLTFX  - G*F                                                    *  CONOC970
C     *          XIPPFX - I + G*F                                                *  CONOC980
C     *          PLTFXI -   INV(F*G)                                             *  CONOC990
C     *          FXPLTI -   INV(G*F)                                             *  CONO1000
C     *          XIFFLI -   I + INV (F*G)                                        *  CONO1010
C     *          XIPLFI -   I + INV (F*F)                                        *  CONO1020
C     *                                                                          *  CONO1030
C     *                                                                          *  CONO1040
C     ****************************************************************************  CONO1050
C                                                                                   CONO1060
C                                                                                   CONO1070
      COMPLEX *8EIG(50),Z(10,10),MU(50),OMU(50),OEIG(50),GZ(10,10),AX(10CONO1080
     1,10),BX(10,10),CX(10,10),FX(10,10),FXPLT(10,10),XI(10,10),XIPFXP(1CONO1090
     20,10),CI(10,10),VI(10,10),PLAN(10,10),XXI(10,10),UO(10,10),VO(10,1CONO1100
     30),UMI(10,10),VMI(10,10),UMO(10,10),VMC(10,10),PLTFX(10,10),XIPPFXCONO1110
     4(10,10),PLTFXI(10,10),FXPLTI(10,10),XIFFLI(10,10),XIPLFI(10,10)    CONO1120
C     ****************************************************************************CONO1130
C     ****************************************************************************CONO1140
C        SET UP COMPLEX WORKING PARAMETERS                                          CONO1150
C     ****************************************************************************CONO1160
C                                                                                   CONO1170
      COMPLEX ZN,TEMP1,TEMP2                                                       CONO1180
C     ****************************************************************************CONO1190
C     *    SET UP DIMENSIONS FOR OPTIMIZER PARAMETERS AND OTHER FUNCTIONS        *CONO1200
C     *                                                                          *CONO1210
C     *     X  - THE DESIGN VARIABLE FOR THE OPTIMIZER                           *CONO1220
C     *     VLB - DESIGN VARIABLE LOWER BOUND                                    *CONO1230
C     *     VUB - DESIGN VARIABLE UPPER BOUND                                    *CONO1240
C     *     G  - OPTIMIZER CONSTRAINT VECTOR                                     *CONO1250
C     *     IDG - OPTIMIZATION CONSTRAINT TYPE                                   *CONO1260
C     *     IC  - OPTIMIZER PARAMETER                                            *CONO1270
C     *     DF  - OPTIMIZER GRADIENT VECTOR                                      *CONO1280
C     *     XA  - OPTIMIZER STORAGE ARRAY                                        *CONO1290
C     *     WK1 - OPTIMIZER WORK SPACE                                           *CONO1300
C     *     IWK - OPTIMIZER WORK SPACE                                           *CONO1310
C     *     IF - ARRAY USED TO CHOOSE DESIGN VARIABLES FROM INPUT DATA           *CONO1320
C     *     IR - USED BY SORT ROUTINE                                            *CONO1330
C     *     KEY - USED BY SORT ROUTINE                                           *CONO1340
C     *     R  - USED BY SORT ROUTINE                                            *CONO1350
C     *                                                                          *CONO1360
C     ****************************************************************************CONO1370
C                                                                                   CONO1380
      DIMENSION X(50),VLB(50),VUB(50),G(50),IDG(50),IC(50),DF(50),XA(50,CONO1390
     150),WK1(10000),IWK(1000),IF(10,10),IR(50),KEY(50),R(50)             CONO1400
      DATA NRA/50/,NCA/50/,NRWK/10000/,NRIWK/1000/                         CONO1410
C     ****************************************************************************CONO1420
C                                                                                   CONO1430
C        INPUT DATA INTO THE PROGRAM TO SET UP THE SYSTEM                           CONO1440
C           XDOT=AX+BU ; Y=CX
```

```
C        WHERE U=-FY+R   OR U=-FX+ R  I.E. C=DIA(I)                    CONO1450
C    ****************************************************************** CONO1460
C    ****************************************************************** CONO1470
C    *       DATA FORMAT PREPARATION                                  * CONO1480
C    * CARD 1 KODE, KONTRL; FORMAT (I5,I5)                            * CONO1490
C    * KODE IS USER SUPPLIED NUMBER TO IDENTIFY THE DATA RUN            CONO1500
C    * KONTRL SETS FLAG TO TELL PROGRAM HOW TO HANDLE RUN MAY BE:     * CONO1510
C    *          0  ENTER OPTIMIZER ON FIRST PASS THROUGH THE PROGRAM  * CONO1520
C    *          1  COMPUTE VALUES THROUGH PROGRAM WITHOUT OPTIMIZER   * CONO1530
C    *               ON FIRST PASS THEN OPTIMIZE                      * CONO1540
C    *          2 OR MORE COMPUTE VALUES DO NOT OPTIMIZE              * CONO1550
C    * CARD 2 WMAX,WI,DELW; FORMAT(3F10.0)                            * CONO1560
C    *        WMAX - MAXIMUM FREQUENCY TO BE USED                     * CONO1570
C    *        WI - STARTING FREQUENCY                                 * CONO1580
C    *        DELW - FREQUENCY STEP FOR FIRST DECADE , THIS STEP IS   * CONO1590
C    *               MODIFIED BY PROGRAM AT EACH DECADE SHIFT         * CONO1600
C    * CARD 3 WT1,WT2,WT3; FORMAT(3F10.0)                             * CONO1610
C    *        WT1 WEIGHTING FACTOR TO CAUSE THE OPTIMIZER TO USE THE  * CONO1620
C    *            POLE PLACEMENT PORTION OF THE OBJECTIVE FUNCTION    * CONO1630
C    *             IN THE OPTIMIZATION ROUTINE                        * CONO1640
C    *        WT2 WEIGHTING FACTOR WHICH CAUSES THE OPTIMIZER TO      * CONO1650
C    *            CONSIDER THE INPUT SINGULAR VALUE OF (I+FG) IN      * CONO1660
C    *            THE OPTIMIZATION OBJECTIVE FUNCTION                 * CONO1670
C    *        WT3 WEIGHTING FACTOR WHICH CAUSES THE OPTIMIZER TO      * CONO1680
C    *             USE THE OUTPUT SINGULAR VALUE (I+GF) IN THE        * CONO1690
C    *             OBJECTIVE. IF THE OBJECTIVE NEEDS FURTHER PARAMETER * CONO1700
C    *             ADDITION THEY MAY BE WEIGHTED IN SIMIALR FASHION   * CONO1710
C    * CARD 4  SVMINI,SVMINO,RJ,NIDG; FORMAT(3F10.0,I5)               * CONO1720
C    *         SVMINI - DESIRED SINGULAR VALUE LEVEL FOR THE S.V. OF  * CONO1730
C    *               (I+FG).FOUND USING UNIVERSAL GAIN AND PHASE CHART * CONO1740
C    *               TO GIVE A DESIRED PHASE AND GAIN MARGIN          * CONO1750
C    *         SVMINO - OUTPUT VERSION OF SVMINI                      * CONO1760
C    *         RJ - FACTOR TO SET PERCENT POLE PLACEMENT ERROR THE    * CONO1770
C    *             DESIGNER WISHES TO PUT INTO THE OPTIMIZER          * CONO1780
C    *         NIDG - PARAMETER THAT SETS TYPE CONSTRAINT CONSIDERED  * CONO1790
C    *                SEE ADS MANUAL FOR USE                          * CONO1800
C    * CARD 5  IGRAD,NDV,NCON,ISTRAT,ICPT,ICNED,IPRINT,INFO           * CONO1810
C    *            FORMAT(8I5)                                         * CONO1820
C    *         IGRAD - PARAMETER CONCERNING GRADIENT COMP SEE ADS     * CONO1830
C    *         NDV - NUMBER OF DESIGN VARIABLES                       * CONO1840
C    *         NCON - NUMBER OF CONSTRAINTS                           * CONO1850
C    *         ISTRAT - OPTIMIZER STRATEGY SEE ADS MANUAL             * CONO1860
C    *         ICPT - OPTIMIZATION METHOD SEE ADS                     * CONO1870
C    *         ICNED - ONE DIMENSIONAL SEARCH TECHNIQUE SEE ADS       * CONO1880
C    *         IPRINT - ADS PRINT CONTROL SEE ADS                     * CONO1890
C    *         INFC - OPTIMIZER CONTROL PARAMETER SEE ADS             * CONO1900
C    *CARD 6 NROWA,NCCLA,NROWB,NCCLB,NROWC,NCCLC,NROWF,NCOLF,NMU      * CONO1910
C    *          FORMAT(9I2)                                           * CONO1920
```

```
C     *          NUMBER OF ROWS AND COLUMNS OF INPUT MATRICES AND          *  CON01930
C     *          NUMBER OF POLES TO BE INPUT FOR PLACEMENT                 *  CON01940
C     * CARD 7 A MATRIX ; READ BY ROWS AS:                                 *  CON01950
C     *               A(1,1)    A(1,2)                                     *  CON01960
C     *               A(2,1)    A(2,2)                                     *  CON01970
C     * THERE WILL BE ONE OR MORE CARDS FOR EACH ROW OF A                  *  CON01980
C     * AFTER THE PROPER CARDS ARE READ FOR A THE NEXT MATRIX MUST         *  CON01990
C     * BE READ IN. THE REMAINDER OF THIS COMMENT WILL REFER TO EACH NEW   *  CON02000
C     * DATA SET AS A CARD. REMEMBER, THERE MAY BE SEVERAL CARDS PER       *  CON02010
C     * SET SO THE NUMBERS DON'T CORRESPOND DIRECTLY WITH  THE DATA        *  CON02020
C     * LISTED IN THE INPUT FILE.                                          *  CON02030
C     * CARD 8 B MATRIX; READ ROW BY ROW                                   *  CON02040
C     * CARD 9 C MATRIX; READ ROW BY ROW                                   *  CON02050
C     * CARD 10 F MATRIX; READ ROW BY ROW                                  *  CON02060
C     * CARD 11 IF MATRIX ; ROW BY ROW                                     *  CON02070
C     *          FORMAT(8I2) ; THIS MATRIX DEFINES THE DESIGN VARIABLE     *  CON02080
C     *          TO THE OPTIMIZER. THE ENTRY WOULD BE OF THE FORM:         *  CON02090
C     *             1 2                                                    *  CON02100
C     *             0 3                                                    *  CON02110
C     *          THIS WOULD TELL THE OPTIMIZER TO MAKE F(1,1) THE FIRST    *  CON02120
C     *          DESIGN VARIABLE (X(1)) AND SO FORTH. THE ZERO TELLS THE   *  CON02130
C     *          OPTIMIZER THAT F(2,1) (IN THIS CASE) IS NOT A DESIGN      *  CON02140
C     *          VARIABLE AND IS NOT TO BE CHANGED.                        *  CON02150
C     * CARD 12  VLB,VUB; FORMAT(2F10.0)                                   *  CON02160
C     *          THERE WILL BE ONE CARD FOR EACH DESIGN VARIABLE           *  CON02170
C     * CARD 13  REALMU,IMAGMU ; FORMAT (2F10.0)                           *  CON02180
C     *           INPUT REAL AND IMAGINARY DESIRED POLE LOCATION           *  CON02190
C     *           ONE POLE PER CARD                                        *  CON02200
C     *                                                                    *  CON02210
C     ********************************************************************  CON02220
C                                                                            CON02230
C                                                                            CON02240
C     INPUT RUN CODE AND PROGRAM CONTROL                                     CON02250
      READ (1,750) KODE,KONTRL                                             CON02260
C                                                                            CON02270
C                                                                            CON02280
C     INPUT MAXN, INITIAL FREQUENCY,DELTA FREQ, OPTIMIZER WEIGHTS            CON02290
      READ (1,710) WMAX,WI,DELW                                            CON02300
      READ (1,570) WT1,WT2,WT3                                             CON02310
C                                                                            CON02320
C     INPUT DESIRED GAIN AND PHASE MARGIN PARAMETERS                         CON02330
C                                                                            CON02340
      READ (1,580) SVMINI,SVMINO,RJ,NIDG                                   CON02350
      W=WI                                                                 CON02360
      NC=1                                                                 CON02370
C                                                                            CON02380
C     INPUT OPTIMIZER INPUT DATA (SEE ADS MANUAL)                            CON02390
C                                                                            CON02400
```

```
      READ (1,760) IGRAD,NDV,NCON,ISTRAT,IOPT,IONED,IPRINT,INFO          CONO2410
C                                                                        CONO2420
C     INPUT THE PARAMETER VALUES AND THE MATRICES A,B,C,F                CONO2430
C                                                                        CONO2440
      READ (1,770) NROWA,NCOLA,NROWB,NCOLB,NROWC,NCOLC,NROWF,NCOLF,NMU   CONO2450
      CALL READ (A,NROWA,NCOLA)                                          CONO2460
      CALL READ (B,NROWB,NCOLB)                                          CONO2470
      CALL READ (C,NROWC,NCOLC)                                          CONO2480
      CALL READ (F,NROWF,NCOLF)                                          CONO2490
C                                                                        CONO2500
C     SPECIFY THE DESIGN VARIABLES WITHIN THE F MATRIX                   CONO2510
C                                                                        CONO2520
      DO 10 J=1,NROWF                                                    CONO2530
10    READ (1,780) (IF(J,K),K=1,NCOLF)                                   CONO2540
C                                                                        CONO2550
C     SET THE DESIGN VARIABLE BOUNDS                                     CONO2560
C                                                                        CONO2570
      DO 20 J=1,NDV                                                      CONO2580
20    READ (1,790) VLB(J),VUB(J)                                         CONO2590
C                                                                        CONO2600
C     FORMULATE THE DESIGN VARIABLE X FROM THE INPUT GUESS FOR F         CONO2610
C                                                                        CONO2620
      DO 50 J=1,NROWF                                                    CONO2630
      DO 40 K=1,NCOLF                                                    CONO2640
      IF (IF(J,K).EQ.0) GO TO 30                                         CONO2650
      KK=IF(J,K)                                                         CONO2660
      X(KK)=F(J,K)                                                       CONO2670
30    CONTINUE                                                           CONO2680
40    CONTINUE                                                           CONO2690
50    CONTINUE                                                           CONO2700
C                                                                        CONO2710
C     READ THE DESIRED EIGENVALUES AS REAL(X)+REAL(Y)*I                  CONO2720
C                                                                        CONO2730
      DO 60 I=1,NMU                                                      CONO2740
60    READ (1,860) REALMU(I),IMAGMU(I)                                   CONO2750
      DO 70 I=1,NMU                                                      CONO2760
      MU(I)=CMPLX(REALMU(I),IMAGMU(I))                                   CONO2770
70    CONTINUE                                                           CONO2780
C                                                                        CONO2790
C   SORT THE INPUT EIGENVALUESUSING IMSL ROUTINE VSRTR                   CONO2800
C                                                                        CONO2810
      DO 80 I=1,NMU                                                      CONO2820
80    IR(I)=I                                                            CONO2830
      CALL VSRTR (REALMU,NMU,IR)                                         CONO2840
      DO 90 J=1,NMU                                                      CONO2850
      K=IR(J)                                                            CONO2860
90    OMU(J)=MU(K)                                                       CONO2870
      JJ=NMU-1                                                           CONO2880
```

```
      DO 110 J=1,JJ                                                    CCNO2890
      IF (REAL(CMU(J)).NE.REAL(CMU(J+1))) GO TO 100                    CCNO2900
      IF (AIMAG(CMU(J)).LT.AIMAG(CMU(J+1))) GO TO 100                  CCNO2910
      TEMP1=CML(J)                                                     CCNO2920
      CML(J)=CML(J+1)                                                  CCNO2930
      CML(J+1)=TEMP1                                                   CCNO2940
  100 CONTINUE                                                         CCNO2950
  110 CONTINUE                                                         CCNO2960
C  ***************************************************************** CCNO2970
C                                                                      CCNO2980
C        WRITE THE INPUT DATA FOR CHECKING AND REFERENCE               CCNO2990
C                                                                      CCNO3000
C  ***************************************************************** CCNO3010
      WRITE (6,800) KODE                                               CCNO3020
      WRITE (6,720) WMAX,HI,DELW                                       CCNO3030
      WRITE (6,590) HT1,HT2,HT3                                        CCNO3040
      WRITE (6,600) SVMINI,SVMINO,RJ,NIDG                              CCNO3050
      WRITE (6,610) NROWA,NMU                                          CCNO3060
      WRITE (6,810)                                                    CCNO3070
      CALL WRITE (A,NROWA,NCOLA)                                       CCNO3080
      WRITE (6,820)                                                    CCNO3090
      CALL WRITE (B,NROWB,NCOLB)                                       CCNO3100
      WRITE (6,830)                                                    CCNO3110
      CALL WRITE (C,NROWC,NCOLC)                                       CCNO3120
      WRITE (6,840)                                                    CCNO3130
      CALL WRITE (F,NROWF,NCOLF)                                       CCNO3140
      WRITE (6,850)                                                    CCNO3150
      CALL CVECWR (CMU,NML)                                            CCNO3160
C                                                                      CCNO3170
C   IF KCNTRL SET GREATER THAN 0 THE FIRST PASS JUMPS THE OPTIMIZER    CCNO3180
C                                                                      CCNO3190
      IF (KCNTRL.GT.0) GO TO 130                                       CCNO3200
C  ***************************************************************** CCNO3210
C                                                                      CCNO3220
C       CALL THE ADS OPTIMIZER TO SOLVE FOR THE BEST F'S               CCNO3230
C                                                                      CCNO3240
C  ***************************************************************** CCNO3250
C                                                                      CCNO3260
C   THIS FIRST CALL SETS THE SCALE FLAG SEE ADS MANUAL                 CCNO3270
      INFO=-2                                                          CCNO3280
      CALL ADS (INFO,ISTRAT,ICPT,IONED,IPRINT,IGRAD,NDV,NCON,X,VLB,VUB,OCCNO3290
     1BJ,G,ICG,NGT,IC,CF,XA,NRA,NCA,WK1,NRWK,IWK,NRIWK)                CCNO3300
      IWK(2)=1                                                         CCNO3310
C***************************************************************      CCNO3320
C                                                                      CCNO3330
C       THIS CALL STARTS THE OPTIMIZATION PROCESS                      CCNO3340
C ***************************************************************    CCNO3350
  120 CALL ADS (INFO,ISTRAT,ICPT,IONED,IPRINT,IGRAD,NDV,NCON,X,VLB,VUB,OCCNO3360
```

```
     1BJ,G,IEC,NGT,IC,DF,XA,NRA,NCA,WK1,NRWK,IWK,NRIWK)                    CCNO3370
      IF (INFC.EQ.0) GO TO 480                                             CCNO3380
      IF (INFC.GT.1) GO TO 470                                             CCNO3390
C  ***************************************************************       CCNO3400
C                                                                          CCNO3410
C     THIS PORTION OF THE PROGRAM DOES THE ACTUAL COMPUTATION              CCNO3420
C                                                                          CCNO3430
C  ***************************************************************       CCNO3440
130   DO 160 J=1,NROWF                                                     CCNO3450
      DO 150 K=1,NCOLF                                                     CCNO3460
C                                                                          CCNO3470
C     EQUATE DESIGN VECTOR X WITH THE DESIRED SYSTEM MATRIX                CCNO3480
C                                                                          CCNO3490
      IF (IF(J,K).EQ.0) GO TO 140                                          CCNO3500
      KK=IF(J,K)                                                           CCNO3510
      F(J,K)=X(KK)                                                         CCNO3520
140   CONTINUE                                                             CCNO3530
150   CONTINUE                                                             CCNO3540
160   CONTINUE                                                             CCNO3550
C                                                                          CCNO3560
C      COMPLEX ALL THE INPUT SYSTEM MATRICES FOR THE TRANSFER FUNCTION     CCNO3570
C                                                                          CCNO3580
      CALL CPLXCV (AX,A,NROWA,NCOLA)                                       CCNO3590
      CALL CPLXCV (BX,B,NROWB,NCOLB)                                       CCNO3600
      CALL CPLXCV (CX,C,NROWC,NCOLC)                                       CCNO3610
      CALL CPLXCV (FX,F,NROWF,NCOLF)                                       CCNO3620
C                                                                          CCNO3630
C    COMPUTE THE A-B*F*C SYSTEM MATRIX FOR EIGEN COMPUTATION               CCNO3640
C                                                                          CCNO3650
      CALL MPLL (F,C,NROWF,NCOLF,NCOLC,FC)                                 CCNO3660
      CALL MPLL (B,FC,NROWB,NCOLB,NCOLC,BFC)                               CCNO3670
      DO 180 I=1,NROWA                                                     CCNO3680
      DO 170 J=1,NCOLA                                                     CCNO3690
170   AMBFC(I,J)=A(I,J)-BFC(I,J)                                           CCNO3700
180   CONTINUE                                                             CCNO3710
C  ***************************************************************       CCNO3720
C                                                                          CCNO3730
C    COMPUTE AND SORT THE EIGENVALUES AND VECTORS(IMSL ROUTINES)           CCNO3740
C                                                                          CCNO3750
C***************************************************************         CCNO3760
      CALL EIGRF (AMBFC,NROWA,10,2,EIG,Z,10,WK,IER)                        CCNO3770
      DO 190 J=1,NROWA                                                     CCNO3780
      REIG(J)=REAL(EIG(J))                                                 CCNO3790
190   CONTINUE                                                             CCNO3800
      DO 200 J=1,NROWA                                                     CCNO3810
200   KEY(J)=J                                                             CCNO3820
      CALL VSRTR (REIG,NROWA,KEY)                                          CCNO3830
      DO 220 J=1,NCOLA                                                     CCNO3840
```

```
      K=KEY(J)                                                          CONO3E50
      CEIG(J)=EIG(K)                                                    CONO3E60
      DO 210 KK=1,NROWA                                                 CONO3E70
      CZ(KK,J)=Z(KK,K)                                                  CONO3E80
  210 CONTINUE                                                          CONO3E90
  220 CONTINUE                                                          CONO3S00
      JJ=NCOLA-1                                                        CONO3S10
      DO 250 J=1,JJ                                                     CONO3S20
      IF (REAL(CEIG(J)).NE.REAL(CEIG(J+1))) GO TO 240                   CONO3S30
      IF (AIMAG(CEIG(J)).LT.AIMAG(CEIG(J+1))) GO TO 240                 CONO3S40
      TEMP1=CEIG(J)                                                     CONO3S50
      CEIG(J)=CEIG(J+1)                                                 CONO3S60
      CEIG(J+1)=TEMP1                                                   CONO3S70
      DO 230 KK=1,NROWA                                                 CONO3S80
      TEMP2=CZ(KK,J)                                                    CONO3S90
      CZ(KK,J)=CZ(KK,J+1)                                               CONO4000
      CZ(KK,J+1)=TEMP2                                                  CONO4010
  230 CONTINUE                                                          CONO4020
  240 CONTINUE                                                          CONO4030
  250 CONTINUE                                                          CONO4040
C     ****************************************************************  CONO4050
C                                                                       CONO4060
C     CALL THE PLANT TRANSFER MATRIX ROUTINE AND FORM THE SYSTEM        CONO4070
C       RETURN DIFFERENCE MATRICES AS REQUIRED                          CONO4080
C                                                                       CONO4090
C     ****************************************************************  CONO4100
      DEL=DELH                                                          CONO4110
      HII=HI                                                            CONO4120
      NC=1                                                              CONO4130
      H=HI                                                              CONO4140
      SMINAI=0.                                                         CONO4150
      SMINMI=0.                                                         CONO4160
      SMINAO=0.                                                         CONO4170
      SMINMO=0.                                                         CONO4180
      SMAXAI=0.                                                         CONO4190
      SMAXMI=0.                                                         CONO4200
      SMAXAO=0.                                                         CONO4210
      SMAXMO=0.                                                         CONO4220
  260 CALL PLANT (PLAN,AX,BX,CX,W,NROWA,NCOLA,NROWB,NCOLB,NROWC,NCOLC)  CONO4230
      CALL CMATML (FX,PLAN,NROWF,NCOLF,NCOLB,FXPLT)                     CONO4240
      CALL CMATML (PLAN,FX,NROWC,NCOLB,NCOLF,PLTFX)                     CONO4250
      DO 280 J=1,NROWF                                                  CONO4260
      DO 270 K=1,NCOLB                                                  CONO4270
  270 XI(J,K)=0.0                                                       CONO4280
  280 CONTINUE                                                          CONO4290
      DO 290 J=1,NROWF                                                  CONO4300
  290 XI(J,J)=1.0                                                       CONO4310
      DO 310 J=1,NROWC                                                  CONO4320
```

```
      DO 300 K=1,NCOLF                                                  CCN04330
      XXI(J,K)=0.0                                                      CCN04340
      XXI(J,J)=1.0                                                      CCN04350
      XIPPFX(J,K)=XXI(J,K)+PLTFX(J,K)                                   CCN04360
300   CONTINUE                                                          CCN04370
310   CONTINUE                                                          CCN04380
      DO 330 J=1,NROWF                                                  CCN04390
      DO 320 K=1,NCOLB                                                  CCN04400
320   XIPFXP(J,K)=XI(J,K)+FXPLT(J,K)                                    CCN04410
330   CONTINUE                                                          CCN04420
      DO 350 J=1,NROWF                                                  CCN04430
      DO 340 K=1,NROWF                                                  CCN04440
      FXPLTI(J,K)=0.0                                                   CCN04450
340   FXPLTI(J,J)=1.0                                                   CCN04460
350   CONTINUE                                                          CCN04470
      DO 370 J=1,NROWC                                                  CCN04480
      DO 360 K=1,NROWC                                                  CCN04490
      PLTFXI(J,K)=0.0                                                   CCN04500
360   PLTFXI(J,J)=1.                                                    CCN04510
370   CONTINUE                                                          CCN04520
C     CALL LECTIC (FXPLT,NROWF,10,FXPLTI,NROWF,10,0,WA,IER)             CCN04530
C     CALL LECTIC (PLTFX,NROWC,10,PLTFXI,NROWC,10,0,WA,IER)             CCN04540
      DO 390 J=1,NROWF                                                  CCN04550
      DO 380 K=1,NCOLB                                                  CCN04560
      XIFPLI(J,K)=XI(J,K)+FXPLTI(J,K)                                   CCN04570
380   CONTINUE                                                          CCN04580
390   CONTINUE                                                          CCN04590
      DO 410 J=1,NROWC                                                  CCN04600
      DO 400 K=1,NROWC                                                  CCN04610
      XIPLFI(J,K)=XXI(J,K)+PLTFXI(J,K)                                  CCN04620
400   CONTINUE                                                          CCN04630
410   CONTINUE                                                          CCN04640
C  ***************************************************************    CCN04650
C                                                                       CCN04660
C    DO SINGULAR VALUE DECOMP AND QUANTIFY ALL DESIRED SV'S             CCN04670
C                                                                       CCN04680
C***************************************************************      CCN04690
      CALL CSVD (XIPFXP,10,10,NROWF,NROWF,0,NROWF,NROWF,SV,UI,VI)       CCN04700
      CALL CSVD (XIPPFX,10,10,NROWC,NROWC,0,NROWC,NROWC,SVAO,UO,VO)     CCN04710
      CALL CSVD (XIFPLI,10,10,NROWF,NROWF,0,NROWF,NROWF,SVMI,UMI,VMI)   CCN04720
      CALL CSVD (XIPLFI,10,10,NROWC,NROWC,0,NROWC,NROWC,SVMO,UMO,VMO)   CCN04730
      OMEGA(NC)=W                                                       CCN04740
      SIGNM1(NC)=SV(NROWF)                                              CCN04750
      SIGNM2(NC)=SV(NROWF-1)                                            CCN04760
      SIGNMX(NC)=SV(1)                                                  CCN04770
      SVADMO(NC)=SVAO(NROWC)                                            CCN04780
      SVADXO(NC)=SVAO(1)                                                CCN04790
      SVMIM(NC)=SVMI(NROWF)                                             CCN04800
```

```
      SVMIX(NC)=SVMI(1)                                                    CCN04810
      SVMMO(NC)=SVMO(NRUNC)                                                CCN04820
      SVMXO(NC)=SVMO(1)                                                    CCN04830
      SIGPRC(NC)=SQRT(SIGNM1(NC)*SIGNM2(NC))                               CCN04840
      SMINAI=(SMINAI+(AMAX1(0.0,SVMINI-SIGNM1(NC)))**2)                    CCN04850
      SMINAO=(SMINAO+(AMAX1(0.0,SVMINO-SVADMC(NC)))**2)                    CCN04860
C                                                                          CCN04870
C  COMPUTE LOOP FOR ALL DESIRED FREQUENCY BANDWIDTH                        CCN04880
C                                                                          CCN04890
      W=W+DEL                                                              CCN04900
      WD=10.0*WII                                                          CCN04910
      IF (W.LT.WD) GO TO 420                                               CCN04920
      DEL=10.0*DEL                                                         CCN04930
      WII=10.0*WII                                                         CCN04940
420   NC=NC+1                                                              CCN04950
      IF (W.LE.WMAX) GO TO 260                                             CCN04960
      MAXN=NC-1                                                            CCN04970
C                                                                          CCN04980
C                                                                          CCN04990
C  FORM QUANTITIES TO USE IN THE COST CRITERIA OF OPTIMIZER                 CCN05000
      SVMIN=SIGNM1(1)                                                      CCN05010
      DO 430 J=1,MAXN                                                      CCN05020
      IF (SIGNM1(J).GE.SVMIN) GO TO 430                                    CCN05030
      SVMIN=SIGNM1(J)                                                      CCN05040
430   CONTINUE                                                             CCN05050
      COST=0.                                                              CCN05060
      DO 440 J=1,NMU                                                       CCN05070
      COST=COST+((REAL(OMU(J))-REAL(CEIG(J)))**2+(AIMAG(OMU(J))-AIMAG(DE   CCN05080
     1IG(J)))**2)                                                          CCN05090
440   CONTINUE                                                             CCN05100
C                                                                          CCN05110
C                                                                          CCN05120
C    THE OBJECTIVE FUNCTION IS INSERTED HERE                               CCN05130
C     OBJ=WT1*COST+WT2/SVMIN                                               CCN05140
      OBJ=WT1*COST+WT2*SMINAI+WT3*SMINAO                                   CCN05150
C     OBJ=WT2*SMINAI                                                       CCN05160
      IF (NCON.EQ.0) GO TO 460                                             CCN05170
      DO 450 J=1,NMU                                                       CCN05180
      IDG(J)=NDG                                                           CCN05190
C     R(J)=RJ                                                              CCN05200
      R(J)=RJ*SQRT((REAL(CMU(J))-REAL(CEIG(J)))**2+(AIMAG(OMU(J))-AIMAG(   CCN05210
     1CEIG(J)))**2)                                                        CCN05220
450   G(J)=SQRT((REAL(CMU(J))-REAL(CEIG(J)))**2+(AIMAG(OMU(J))-AIMAG(CEI   CCN05230
     1G(J)))**2)-R(J)                                                      CCN05240
460   IF (KCNTRL.GT.0) GO TO 480                                           CCN05250
      GO TO 120                                                            CCN05260
470   CONTINUE                                                             CCN05270
C                                                                          CCN05280
```

```
C  THE GRADIENT COMPUTATION ROUTINE GOES HERE IF NEEDED                    CCNO5290
C      GO TO 125                                                          CCNO5300
480    CONTINUE                                                           CCNO5310
C                                                                         CCNO5320
C  OUTPUT THE COMPUTED FINAL EIGEN VALUES AND VECTORS FROM OPT F'S        CCNO5330
C                                                                         CCNO5340
       CALL MPLL (F,C,NROWF,NCOLF,NCOLC,FC)                               CCNO5350
       CALL MPLL (B,FC,NROWB,NCOLB,NCOLC,BFC)                             CCNO5360
       DO 500 I=1,NROWA                                                   CCNO5370
       DO 490 J=1,NCOLA                                                   CCNO5380
490    AMBFC(I,J)=A(I,J)-BFC(I,J)                                         CCNO5390
500    CONTINUE                                                           CCNO5400
       IF (KONTRL.EQ.0) GO TO 510                                         CCNO5410
       WRITE (6,620)                                                      CCNO5420
       GO TO 520                                                          CCNO5430
510    WRITE (6,630)                                                      CCNO5440
520    CALL EIGRF (AMBFC,NROWA,10,2,EIG,Z,10,WK,IER)                      CCNO5450
       WRITE (6,870)                                                      CCNO5460
       CALL CVECWR (OEIG,NCOLA)                                           CCNO5470
       DO 530 J=1,NMU                                                     CCNO5480
530    WRITE (3,640) OEIG(J)                                              CCNO5490
       WRITE (6,880)                                                      CCNO5500
       CALL CWRITE (OZ,NROWA,NCOLA)                                       CCNO5510
       WRITE (6,890)                                                      CCNO5520
       CALL WRITE (F,NROWF,NCOLF)                                         CCNO5530
       WRITE (6,650) OBJ                                                  CCNO5540
C                                                                         CCNO5550
C OUTPUT THE OPTIMIZED SINGULAR VALUE DATA                                CCNO5560
C                                                                         CCNO5570
       WRITE (6,730)                                                      CCNO5580
       WRITE (3,660) MAXN                                                 CCNO5590
       DO 540 J=1,MAXN                                                    CCNO5600
       WRITE (3,670) OMEGA(J),SIGNM1(J),SIGNM2(J),SIGNMX(J),SIGPRO(J)     CCNO5610
540    WRITE (6,740) OMEGA(J),SIGNM1(J),SIGNM2(J),SIGNMX(J),SIGPRO(J)     CCNO5620
       WRITE (6,680)                                                      CCNO5630
       DO 550 J=1,MAXN                                                    CCNO5640
       WRITE (3,690) SVADMO(J),SVADXO(J),SVMIN(J),SVMIX(J),SVMMO(J),SVMXO CCNO5650
     1(J)                                                                 CCNO5660
550    WRITE (6,700) OMEGA(J),SVADMO(J),SVADXO(J),SVMIN(J),SVMIX(J),SVMMO CCNO5670
     1(J),SVMXO(J)                                                        CCNO5680
       IF (KONTRL.EQ.0.OR.KONTRL.GT.1) GO TO 560                          CCNO5690
       KONTRL=0                                                           CCNO5700
       GO TO 120                                                          CCNO5710
560    STOP                                                               CCNO5720
C ***************************************************************       CCNO5730
C      FORMAT STATEMENTS                                                  CCNO5740
C ***************************************************************       CCNO5750
570    FORMAT (3F10.0)                                                    CCNO5760
```

```
580    FORMAT (3F10.0,I5)                                                    CCN05770
590    FORMAT (/,10H WEIGHT 1=,F10.5,9HWEIGHT 2=,F10.5,9HWEIGHT 3=,F10.5,    CCN05780
      1/)                                                                    CCN05790
600    FORMAT (/,9H SVMINI= ,F10.5,8H SVMINO=,F10.5,5H  RJ=,F10.5,6H  IDG    CCN05800
      1=,I5,/)                                                               CCN05810
610    FORMAT (2I5)                                                          CCN05820
620    FORMAT (/,28H *** THE INITIAL DATA IS ***,/)                          CCN05830
630    FORMAT (/,41H *** THE CPTIMIZER COMPUTED OUTPUT IS ***,/)             CCN05840
640    FORMAT (2E12.5)                                                       CCN05850
650    FORMAT (//,34H  **** THE OBJECTIVE FUNCTION IS =,F10.5,6H  ****,/)    CCN05860
660    FORMAT (I5)                                                           CCN05870
670    FORMAT (5E12.5)                                                       CCN05880
680    FORMAT (/,2X,7H  FREQ ,5X,6HSVADMO,4X,6HSVADXO,4X,5HSVMIM,4X,5HSVM    CCN05890
      1IX,4X,5HSVMMO,4X,5HSVMXO,/)                                           CCN05900
690    FORMAT (6E12.5)                                                       CCN05910
700    FORMAT (7F10.5)                                                       CCN05920
710    FORMAT (3F10.0)                                                       CCN05930
720    FORMAT (/,6H WMAX=,F10.5,1X,10HINIT FREQ=,F10.5,1X,10HFREQ STEP=,F    CCN05940
      110.5,//)                                                              CCN05950
730    FORMAT (//,10H FREQUENCY,5X,13HMIN ADD IN SV,4X,13H2ND ADD IN SV,3    CCN05960
      1X,13HMAX ADD IN SV,2X,18HMIN PROD ADD IN SV)                          CCN05970
740    FORMAT (5(F10.5,5X))                                                  CCN05980
750    FORMAT (I5,I5)                                                        CCN05990
760    FORMAT (E15)                                                          CCN06000
770    FORMAT (5I2)                                                          CCN06010
780    FORMAT (E12)                                                          CCN06020
790    FORMAT (2F10.0)                                                       CCN06030
800    FORMAT (1H1,///,12H<<<<< RUN # ,I5,6H >>>>>,///)                      CCN06040
810    FORMAT (27H *** THE A PLANT MATRIX ***,///)                           CCN06050
820    FORMAT (//,35H *** THE B CONTROL INPUT MATRIX ***,///)                CCN06060
830    FORMAT (//,33H *** THE C OBSERVATION MATRIX ***,///)                  CCN06070
840    FORMAT (//,28H *** THE          F FEEDBACK MATRIX ***,///)            CCN06080
850    FORMAT (//,44H *** THE ORDERED COMPLEX EIGENVALUES (INPUT),//)        CCN06090
860    FORMAT (2F10.0)                                                       CCN06100
870    FORMAT (//,37H *** THE ORDERED COMPUTED EIGENVALUES,///)              CCN06110
880    FORMAT (//,29H *** THE ORDERED EIGENVECTORS,///)                      CCN06120
890    FORMAT (//,36H *** THE          F FEEDBACK MATRIX ***,///)            CCN06130
       END                                                                   CCN06140
```

```
C*********************************************************************
C
CSUBROUTINE PACKAGE FOR CCNXSV AND CCNOB1
C
C   ******************************************************************
C          WRITE THE COMPLEX MATRIX
C   ******************************************************************
      SUBROUTINE CWRITE (A,N,K)
      COMPLEX *8A(10,10)
      INTEGER N,K,I,J
      DO 10 I=1,N
      WRITE (6,20) (A(I,J),J=1,K)
10    CONTINUE
      RETURN
20    FORMAT (8F12.5)
      END
C   ******************************************************************
C     WRITE THE COMPLEX VECTOR
C   ******************************************************************
      SUBROUTINE CVECWR (A,N)
      COMPLEX *8A(10)
      INTEGER N,J
      WRITE (6,10) (A(J),J=1,N)
      RETURN
10    FORMAT (2F12.5)
      END
C    *****************************************************************
C      WRITE THE REAL VECTOR
C    *****************************************************************
      SUBROUTINE VECWR (A,N,IR)
      REAL*4 A(10)
      DIMENSION IR(50)
      INTEGER N,J
      WRITE (6,10) (A(J),IR(J),J=1,N)
      RETURN
10    FORMAT (F12.5,5X,I5)
      END
C    *****************************************************************
C      MATRIX MULTIPLICATION WITH COMPLEX MATRIX
C    *****************************************************************
      SUBROUTINE CMATML (R,T,M,LL,N,U)
      COMPLEX *8R(10,10),T(10,10),U(10,10)
      INTEGER M,LL,N
      DO 30 I=1,M
      DO 20 J=1,N
      U(I,J)=0.0
      DO 10 INDEX=1,LL
      U(I,J)=U(I,J)+R(I,INDEX)*T(INDEX,J)
```

```
10      CONTINUE                                                        CWROC490
20      CONTINUE                                                        CWROC500
30      CONTINUE                                                        CWROC510
        RETURN                                                          CWROC520
        END                                                             CWROC530
C   ****************************************************************** CWROC540
C       REAL MATRIX MULTIPLICATION ROUTINE                              CWROC550
C   ****************************************************************** CWROC560
        SUBROUTINE MMUL (R,T,M,LL,N,U)                                  CWROC570
        REAL*4 R(10,10),T(10,10),U(10,10)                               CWROC580
        INTEGER M,LL,N                                                  CWROC590
        DO 30 I=1,M                                                     CWROC600
        DO 20 J=1,N                                                     CWROC610
        U(I,J)=0.0                                                      CWROC620
        DO 10 INDEX=1,LL                                                CWROC630
        U(I,J)=U(I,J)+R(I,INDEX)*T(INDEX,J)                             CWROC640
10      CONTINUE                                                        CWROC650
20      CONTINUE                                                        CWROC660
30      CONTINUE                                                        CWROC670
        RETURN                                                          CWROC680
        END                                                             CWROC690
C   ****************************************************************** CWROC700
C    CSVD ALGORITHM FOR COMPLEX MATRICES; ANY OF THE ROUTINES OF IMSL   CWROC710
C    OR LINPACK COULD ALSO BE USED WITH PROPER CAUTIONS                 CWROC720
C   ****************************************************************** CWROC730
        SUBROUTINE CSVD (A,MMAX,NMAX,M,N,P,NU,NV,S,U,V)                 CWROC740
        COMPLEX A(MMAX,1),U(MMAX,1),V(NMAX,1)                           CWROC750
        INTEGER M,N,P,NU,NV                                             CWROC760
        REAL S(1)                                                       CWROC770
        COMPLEX C,R                                                     CWROC780
        REAL B(100),C(100),T(100)                                       CWROC790
        DATA ETA,TOL/1.5E-8,1.E-31/                                     CWROC800
        MP=M+P                                                          CWROC810
        N1=N+1                                                          CWROC820
C                                                                       CWROC830
C   HOUSEHOLDER REDUCTION                                               CWROC840
        C(1)=0.EO                                                       CWROC850
        K=1                                                             CWROC860
10      K1=K+1                                                          CWROC870
C                                                                       CWROC880
C    ELIMINATION OF A(I,K),I=K+1,....,M                                 CWROC890
        Z=0.E0                                                          CWROC900
        DO 20 I=K,M                                                     CWROC910
20      Z=Z+REAL(A(I,K))**2+AIMAG(A(I,K))**2                            CWROC920
        B(K)=0.EO                                                       CWROC930
        IF (Z.LE.TOL) GO TO 70                                          CWROC940
        Z=SQRT(Z)                                                       CWROC950
        B(K)=Z                                                          CWROC960
```

```
      W=CABS(A(K,K))                                                    CWR01070
      Q=(1.EC,C.EC)                                                     CWR01080
      IF (W.NE.0.EO) Q=A(K,K)/W                                         CWR01090
      A(K,K)=Q*(Z+W)                                                    CWR01000
      IF (K.EC.NP) GO TO 70                                             CWR01010
      DO 50 J=K1,NP                                                     CWR01020
      Q=(0.EC,C.EC)                                                     CWR01030
      DO 30 I=K,M                                                       CWR01040
30    Q=Q+CONJG(A(I,K))*A(I,J)                                          CWR01050
      Q=Q/(Z*(Z+W))                                                     CWR01060
      DO 40 I=K,M                                                       CWR01070
40    A(I,J)=A(I,J)-Q*A(I,K)                                            CWR01080
50    CONTINUE                                                          CWR01090
C                                                                       CWR01100
C     PHASE TRANSFORMATION                                              CWR01110
      Q=-CONJG(A(K,K))/CABS(A(K,K))                                     CWR01120
      DO 60 J=K1,NP                                                     CWR01130
60    A(K,J)=Q*A(K,J)                                                   CWR01140
C                                                                       CWR01150
C     ELIMINATION OF A(K,J),J=K+2,...,N                                 CWR01160
70    IF (K.EC.N) GO TO 140                                             CWR01170
      Z=C.EO                                                            CWR01180
      DO 80 J=K1,N                                                      CWR01190
80    Z=Z+REAL(A(K,J))**2+AIMAG(A(K,J))**2                              CWR01200
      C(K1)=C.EO                                                        CWR01210
      IF (Z.LE.TOL) GO TO 130                                           CWR01220
      Z=SQRT(Z)                                                         CWR01230
      C(K1)=Z                                                           CWR01240
      W=CABS(A(K,K1))                                                   CWR01250
      Q=(1.EC,C.EO)                                                     CWR01260
      IF (W.NE.C.EO) Q=A(K,K1)/W                                        CWR01270
      A(K,K1)=Q*(Z+W)                                                   CWR01280
      DO 110 I=K1,M                                                     CWR01290
      Q=(0.EC,0.EC)                                                     CWR01300
      DO 90 J=K1,N                                                      CWR01310
90    Q=Q+CONJG(A(K,J))*A(I,J)                                          CWR01320
      Q=Q/(Z*(Z+W))                                                     CWR01330
      DO 100 J=K1,N                                                     CWR01340
100   A(I,J)=A(I,J)-Q*A(K,J)                                            CWR01350
110   CONTINUE                                                          CWR01360
C                                                                       CWR01370
C     PHASE TRANSFORMATION                                              CWR01380
      Q=-CONJG(A(K,K1))/CABS(A(K,K1))                                   CWR01390
      DO 120 I=K1,M                                                     CWR01400
120   A(I,K1)=A(I,K1)*Q                                                 CWR01410
130   K=K1                                                              CWR01420
      GO TO 10                                                          CWR01430
C                                                                       CWR01440
```

```
C  TOLERENCE FOR NEGLIGIBLE ELEMENTS                                  CWR01450
140   EPS=0.EO                                                        CWR01460
      DO 150 K=1,N                                                    CWR01470
      S(K)=B(K)                                                       CWR01480
      T(K)=C(K)                                                       CWR01490
150   EPS=AMAX1(EPS,S(K)+T(K))                                        CWR01500
      EPS=EPS*ETA                                                     CWR01510
C                                                                     CWR01520
C    INITIALIZATION OF U AND V                                        CWR01530
      IF (NU.EQ.0) GO TO 180                                          CWR01540
      DO 170 J=1,NU                                                   CWR01550
      DO 160 I=1,M                                                    CWR01560
160   U(I,J)=(0.EO,0.EO)                                              CWR01570
170   U(J,J)=(1.EO,0.EO)                                              CWR01580
180   IF (NV.EQ.0) GO TO 210                                          CWR01590
      DO 200 J=1,NV                                                   CWR01600
      DO 190 I=1,N                                                    CWR01610
190   V(I,J)=(0.EO,0.EO)                                              CWR01620
200   V(J,J)=(1.EO,0.EO)                                              CWR01630
C                                                                     CWR01640
C  QR DIAGONALIZATION                                                 CWR01650
210   DO 380 KK=1,N                                                   CWR01660
      K=N1-KK                                                         CWR01670
C                                                                     CWR01680
C    TEST FOR SPLIT                                                   CWR01690
220   DO 230 LL=1,K                                                   CWR01700
      L=K+1-LL                                                        CWR01710
      IF (ABS(T(L)).LE.EPS) GO TO 290                                 CWR01720
      IF (ABS(S(L-1)).LE.EPS) GO TO 240                               CWR01730
230   CONTINUE                                                        CWR01740
C                                                                     CWR01750
C    CANCELLATION OF E(L)                                             CWR01760
240   CS=0.EO                                                         CWR01770
      SN=1.EO                                                         CWR01780
      L1=L-1                                                          CWR01790
      DO 280 I=L,K                                                    CWR01800
      F=SN*T(I)                                                       CWR01810
      T(I)=CS*T(I)                                                    CWR01820
      IF (ABS(F).LE.EPS) GO TO 290                                    CWR01830
      H=S(I)                                                          CWR01840
      W=SQRT(F*F+H*H)                                                 CWR01850
      S(I)=W                                                          CWR01860
      CS=H/W                                                          CWR01870
      SN=-F/W                                                         CWR01880
      IF (NU.EQ.0) GO TO 260                                          CWR01890
      DO 250 J=1,N                                                    CWR01900
      X=REAL(U(J,L1))                                                 CWR01910
      Y=REAL(U(J,I))                                                  CWR01920
```

```
      U(J,L1)=CMPLX(X*CS+Y*SN,0.E0)                                     CWR01930
250   U(J,I)=CMPLX(Y*CS-X*SN,0.E0)                                      CWR01940
260   IF (NP.EQ.N) GO TO 280                                            CWR01950
      DO 270 J=N1,NP                                                    CWR01960
      C=A(L1,J)                                                         CWR01970
      R=A(I,J)                                                          CWR01980
      A(L1,J)=C*CS+R*SN                                                 CWR01990
270   A(I,J)=R*CS-C*SN                                                  CWR02000
280   CONTINUE                                                          CWR02010
C                                                                       CWR02020
C     TEST FOR CONVERGENCE                                              CWR02030
290   W=S(K)                                                            CWR02040
      IF (L.EQ.K) GO TO 360                                             CWR02050
C                                                                       CWR02060
C     ORIGIN SHIFT                                                      CWR02070
      X=S(L)                                                            CWR02080
      Y=S(K-1)                                                          CWR02090
      G=T(K-1)                                                          CWR02100
      H=T(K)                                                            CWR02110
      F=((Y-W)*(Y+W)+(G-H)*(G+H))/(2.E0*H*Y)                            CWR02120
      G=SQRT(F*F+1.E0)                                                  CWR02130
      IF (F.LT.0.E0) G=-G                                               CWR02140
      F=((X-W)*(X+W)+(Y/(F+G)-H)*H)/X                                   CWR02150
C                                                                       CWR02160
C     QR STEP                                                           CWR02170
      CS=1.E0                                                           CWR02180
      SN=1.E0                                                           CWR02190
      L1=L+1                                                            CWR02200
      DO 350 I=L1,K                                                     CWR02210
      G=T(I)                                                            CWR02220
      Y=S(I)                                                            CWR02230
      H=SN*G                                                            CWR02240
      G=CS*G                                                            CWR02250
      W=SQRT(H*H+F*F)                                                   CWR02260
      T(I-1)=W                                                          CWR02270
      CS=F/W                                                            CWR02280
      SN=H/W                                                            CWR02290
      F=X*CS+G*SN                                                       CWR02300
      G=G*CS-X*SN                                                       CWR02310
      H=Y*SN                                                            CWR02320
      Y=Y*CS                                                            CWR02330
      IF (NV.EQ.0) GO TO 310                                            CWR02340
      DO 300 J=1,N                                                      CWR02350
      X=REAL(V(J,I-1))                                                  CWR02360
      W=REAL(V(J,I))                                                    CWR02370
      V(J,I-1)=CMPLX(X*CS+W*SN,0.E0)                                    CWR02380
300   V(J,I)=CMPLX(W*CS-X*SN,0.E0)                                      CWR02390
310   W=SQRT(W*H+F*F)                                                   CWR02400
```

```
      S(I-1)=H                                                          CWR02410
      CS=F/H                                                            CWR02420
      SN=G/H                                                            CWR02430
      X=CS*G+SN*Y                                                       CWR02440
      X=CS*Y-SN*G                                                       CWR02450
      IF (NU.EQ.0) GO TO 330                                            CWR02460
      DO 320 J=1,N                                                      CWR02470
      Y=REAL(U(J,I-1))                                                  CWR02480
      W=REAL(U(J,I))                                                    CWR02490
      U(J,I-1)=CMPLX(Y*CS+W*SN,0.E0)                                    CWR02500
320   U(J,I)=CMPLX(W*CS-Y*SN,0.E0)                                      CWR02510
330   IF (N.EQ.NP) GO TO 350                                            CWR02520
      DO 340 J=N1,NP                                                    CWR02530
      Q=A(I-1,J)                                                        CWR02540
      R=A(I,J)                                                          CWR02550
      A(I-1,J)=Q*CS+R*SN                                                CWR02560
340   A(I,J)=R*CS-Q*SN                                                  CWR02570
350   CONTINUE                                                          CWR02580
      T(L)=0.E0                                                         CWR02590
      T(K)=F                                                            CWR02600
      S(K)=X                                                            CWR02610
      GO TO 220                                                         CWR02620
C                                                                       CWR02630
C  CONVERGENCE                                                          CWR02640
360   IF (W.GE.0.E0) GO TO 380                                          CWR02650
      S(K)=-H                                                           CWR02660
      IF (NV.EQ.0) GO TO 380                                            CWR02670
      DO 370 J=1,N                                                      CWR02680
370   V(J,K)=-V(J,K)                                                    CWR02690
380   CONTINUE                                                          CWR02700
C                                                                       CWR02710
C  SORT SINGULAR VALUES                                                 CWR02720
      DO 450 K=1,N                                                      CWR02730
      G=-1.E0                                                           CWR02740
      J=K                                                               CWR02750
      DO 390 I=K,N                                                      CWR02760
      IF (S(I).LE.G) GO TO 390                                          CWR02770
      G=S(I)                                                            CWR02780
      J=I                                                               CWR02790
390   CONTINUE                                                          CWR02800
      IF (J.EQ.K) GO TO 450                                             CWR02810
      S(J)=S(K)                                                         CWR02820
      S(K)=G                                                            CWR02830
      IF (NV.EQ.0) GO TO 410                                            CWR02840
      DO 400 I=1,N                                                      CWR02850
      Q=V(I,J)                                                          CWR02860
      V(I,J)=V(I,K)                                                     CWR02870
400   V(I,K)=Q                                                          CWR02880
```

```
  410   IF (NL.EQ.0) GO TO 430                                          CWR02690
        DO 420 I=1,N                                                    CWR02700
        C=L(I,J)                                                        CWR02710
        U(I,J)=L(I,K)                                                   CWR02720
  420   U(I,K)=C                                                        CWR02730
  430   IF (N.EQ.NP) GO TO 450                                          CWR02740
        DO 440 I=N1,NP                                                  CWR02750
        C=A(J,I)                                                        CWR02760
        A(J,I)=A(K,I)                                                   CWR02770
  440   A(K,I)=C                                                        CWR02780
  450   CONTINUE                                                        CWR02790
C                                                                       CWR03000
C  BACK TRANSFORMATION                                                  CWR03010
        IF (NU.EQ.0) GO TO 510                                          CWR03020
        DO 500 KK=1,N                                                   CWR03030
        K=N1-KK                                                         CWR03040
        IF (B(K).EQ.0.E0) GO TO 500                                     CWR03050
        Q=-A(K,K)/CABS(A(K,K))                                          CWR03060
        DO 460 J=1,NU                                                   CWR03070
  460   U(K,J)=Q*U(K,J)                                                 CWR03080
        DO 490 J=1,NU                                                   CWR03090
        Q=(0.E0,0.E0)                                                   CWR03100
        DO 470 I=K,N                                                    CWR03110
  470   Q=Q+CONJG(A(I,K))*U(I,J)                                        CWR03120
        Q=Q/(CABS(A(K,K))*B(K))                                         CWR03130
        DO 480 I=K,N                                                    CWR03140
  480   U(I,J)=U(I,J)-Q*A(I,K)                                          CWR03150
  490   CONTINUE                                                        CWR03160
  500   CONTINUE                                                        CWR03170
  510   IF (NV.EQ.0) GO TO 570                                          CWR03180
        IF (N.LT.2) GO TO 570                                           CWR03190
        DO 560 KK=2,N                                                   CWR03200
        K=N1-KK                                                         CWR03210
        K1=K+1                                                          CWR03220
        IF (C(K1).EQ.0.E0) GO TO 560                                    CWR03230
        Q=-CONJG(A(K,K1))/CABS(A(K,K1))                                 CWR03240
        DO 520 J=1,NV                                                   CWR03250
  520   V(K1,J)=Q*V(K1,J)                                               CWR03260
        DO 550 J=1,NV                                                   CWR03270
        Q=(0.E0,0.E0)                                                   CWR03280
        DO 530 I=K1,N                                                   CWR03290
  530   Q=Q+A(K,I)*V(I,J)                                               CWR03300
        Q=Q/(CABS(A(K,K1))*C(K1))                                       CWR03310
        DO 540 I=K1,N                                                   CWR03320
  540   V(I,J)=V(I,J)-Q*CONJG(A(K,I))                                   CWR03330
  550   CONTINUE                                                        CWR03340
  560   CONTINUE                                                        CWR03350
  570   RETURN                                                          CWR03360
```

```
      END
C     ****************************************************************
C        READ A REAL MATRIX
C     ****************************************************************
      SUBROUTINE READ (A,N,K)
      REAL*4 A(10,10)
      INTEGER N,K
      DO 10 I=1,N
      READ (1,20) (A(I,J),J=1,K)
10    CONTINUE
      RETURN
20    FORMAT (8F10.0)
      END
C     ****************************************************************
C        WRITE A REAL MATRIX
C     ****************************************************************
      SUBROUTINE WRITE (A,N,K)
      REAL*4 A(10,10)
      INTEGER N,K
      DO 10 I=1,N
      WRITE (6,20) (A(I,J),J=1,K)
10    CONTINUE
      RETURN
20    FORMAT (8F10.5)
      END
C     ****************************************************************
C     CONVERT REAL MATRIX TO COMPLEX FORM
C     ****************************************************************
      SUBROUTINE CPLXCV (X,A,N,K)
      COMPLEX *8X(10,10)
      REAL*4 A(10,10)
      INTEGER N,K
      DO 20 I=1,N
      DO 10 J=1,K
10    X(I,J)=CMPLX(A(I,J),0.00)
20    CONTINUE
      RETURN
      END
C     ****************************************************************
C        THIS ROUTINE WILL COMPUTE THE TRANSFER FUNCTION AS A
C         FUNCTION OF FREQUENCY FOR A GIVEN A, B, C SET OF
C                         MATRICES
C
C     ****************************************************************
      SUBROUTINE PLANT (GP,AX,BX,CX,W,NRA,NCA,NRB,NCB,NRC,NCC)
      COMPLEX *8CX(10,10),AX(10,10),BX(10,10),GP(10,10),I(10,10),SIA(10,
     110),SIAI(10,10),SIAIB(10,10),SI(10,10)
      COMPLEX S
```

```
      REAL*4 WA(100)                                                    CWR03E50
      S=CMPLX(0.0,W)                                                    CWR03E60
      DO 20 J=1,NRA                                                     CWR03E70
      DO 10 K=1,NCA                                                     CWR03E80
      SI(J,K)=0.0                                                       CWR03E90
10    I(J,K)=0.0                                                        CWR03900
20    CONTINUE                                                          CWR03910
      DO 30 J=1,NCA                                                     CWR03920
      I(J,J)=1.0                                                        CWR03930
30    SI(J,J)=S*I(J,J)                                                  CWR03940
      DO 50 J=1,NRA                                                     CWR03950
      DO 40 K=1,NCA                                                     CWR03960
40    SIA(J,K)=SI(J,K)-AX(J,K)                                          CWR03970
50    CONTINUE                                                          CWR03980
      CALL LEQT1C (SIA,NCA,10,I,NCA,10,0,WA,IER)                        CWR03990
      CALL CMATML (I,BX,NCA,NCA,NCB,SIAIB)                              CWR04000
      CALL CMATML (CX,SIAIB,NRC,NCC,NCB,GP)                             CWR04010
      RETURN                                                            CWR04020
      END                                                               CWR04030
C  ***************************************************************      CWR04040
C     THIS ROUTINE IS INPUT FOR DE SOLVER                               CWR04050
CWHEN DE SOLVER IS ADDED DIRECTLY TO CONXSV                             CWR04060
C  ***************************************************************      CWR04070
      SUBROUTINE FCN (N,T,X,XDOT)                                       CWR04080
      REAL*4 T,X(N),XDOT(N),AMBFC(10,10),A(10,10),B(10,10),C(10,10),    CWR04090
     1F(10,10),PHI(10),BFC(10,10),FC(10,10),AMBFCX(10),BPHI(10)         CWR04100
      COMMON AMBFC,A,B,F,C,PHI,AMBFCX,BPHI,NCOLB                        CWR04110
C     WRITE(6,4004)N,NCOLB                                              CWR04120
C4004 FORMAT(2I5)                                                       CWR04130
C     WRITE(6,4006)(PHI(J),J=1,NCOLB)                                   CWR04140
C4006 FORMAT(2F10.5)                                                    CWR04150
C     CALL WRITE(B,N,NCOLB)                                             CWR04160
C     CALL WRITE(AMBFC,N,N)                                             CWR04170
      DO 10 J=1,N                                                       CWR04180
      BPHI(J)=0.0                                                       CWR04190
      DO 20 K=1,N                                                       CWR04200
20    BPHI(J)=BPHI(J)+B(J,K)*PHI(K)                                     CWR04210
10    CONTINUE                                                          CWR04220
C     WRITE(6,5500)(BPHI(J),J=1,N)                                      CWR04230
C5500 FORMAT(6F10.5)                                                    CWR04240
      DO 30 J=1,N                                                       CWR04250
      AMBFCX(J)=0.0                                                     CWR04260
      DO 40 K=1,N                                                       CWR04270
   4C AMBFCX(J)=AMBFCX(J)+AMBFC(J,K)*X(K)                               CWR04280
   3C CONTINUE                                                          CWR04290
C     WRITE(6,5501)(AMBFCX(J),J=1,N)                                    CWR04300
C5501 FORMAT(6F10.5)                                                    CWR04310
      DO 50 J=1,N                                                       CWR04320
```

```
   5C XDOT(J)=AMBFCX(J)+BPHI(J)                                          CWR04330
C       WRITE(6,5502)(XDOT(J),J=1,N)                                     CWR04340
C5502 FORMAT(2F10.5)                                                     CWR04350
      RETURN                                                             CWR04360
      END                                                                CWR04370
CTHIS SUBROUTINE COMPUTES THE CONTROLLER TRANSFER FUNCTION               CWR04380
CFOR THE OBSERVER ROBUSTNESS PROBLEM                                     CWR04390
C                                                                        CWR04400
C                                                                        CWR04410
      SUBROUTINE CONTRL(H,AX,FX,FKX,FKC,BX,NRA,NCA,NRF,NCF,NRK,NCK,W,    CWR04420
     1NRE)                                                               CWR04430
      COMPLEX*8 H(10,10),AX(10,10),FX(10,10),FKX(10,10),FKCX(10,10),     CWR04440
     *BX(10,10),
     1BFX(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 30 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 CWRITE(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
C       CALL CWRITE(SIA,NRA,NCA)                                         CWR04710
      CALL CFLXCV(FKCX,FKC,NRA,NCA)                                      CWR04720
      CALL CMATML(BFX,BX,FX,NRB,NCF)                                     CWR04730
      DO 60 J=1,NRA                                                      CWR04740
      DO 70 K=1,NCA                                                      CWR04750
   7C AALG(J,K)=SIA(J,K)+FKCX(J,K)+BFX(J,K)                              CWR04760
C       CALL CWRITE(AAUG,NRA,NCA)                                        CWR04770
   6C CONTINUE                                                           CWR04780
      CALL LEQT1C(AAUG,NCA,10,I,NCA,10,0,WA,IER)                         CWR04790
                                                                         CWR04800
```

```
      CALL CMAIML(I,FKX,NRA,NCA,NCK,AAUK)                               CWR04810
      CALL CMAIML(FX,AAUK,NRF,NCF,NCK,H)                                CWR04820
C     CALL CWFITE(H,NRF,NCK)                                            CWR04830
      RETURN                                                            CWR04840
      END                                                               CWR04850
```

```
C****************************************************************************
C*POLE PLACEMENT AND ROBUSTNESS (POPLAR) DESIGN PROGRAM       *
C*MODIFIED VERSION FOR OBSERVER ROBUSTNESS RECOVERY*
C    ************************************************************************
C    ************************************************************************
C    ************************************************************************
C    * THIS PROGRAM IS DESIGNED TO EMPLOY SINGULAR VALUE ANALYSIS AND       *
C    * THE USE OF A OPTIMIZATION ROUTINE TO AID IN POLE PLACEMENT          *
C    *  CONTROL SYSTEM DESIGN FOR LINEAR MULTIVARIABLE SYSTEMS.            *
C    *  ROBUSTNESS OF THE SYSTEM IS ALSO CONSIDERED BY ESTABLISHING        *
C    *  SINGULAR VALUE LEVELS WHICH CORRESPOND TO MULTILOOP GAIN           *
C    *  AND PHASE MARGINS DETERMINED FROM THE UNIVERSAL GAIN PHASE         *
C    *  DIAGRAM DEVELOPED BY NEWSOM AND MUKHAPADHYAY AT NASA LANGLEY.      *
C    *            VERNON C GORDON , CDR , USN                              *
C    *             VERSION 1.0 , 6-28-84
C    ************************************************************************
C    *                                                                     *
C    *                                                                     *
C    * THIS FIRST BLOCK SETS DIMENSIONS AND DECLARES REAL AND              *
C    *  COMPLEX FUNCTIONS. THE FOLLOWING IS A LIST OF PROGRAM SYMBOLS:     *
C    ************************************************************************
C    ************************************************************************
C    *                                                                     *
C    *             REAL VARIABLES BY NAME AND BRIEF DESCRIPTION            *
C    *      THE SYSTEM UNDER CONSIDERATION IS GIVEN AS:                    *
C    *          XDOT = A*X + B*U ; Y= C*X    , FOR STATE OR                *
C    *             OUTPUTFEEDBACK OF THE FORM U=-FX +R OR U=-FY + R        *
C    *       THIS PROGRAM HAS BEEN MODIFIED TO WORK AS A DOUBLE PASS       *
C    *          PROGRAM TO RECOVER ROBUSTNESS IN OBSERVER DESIGNS          *
C    *         BY USING THE FILTER GAINS AS THE RECOVERY VARIABLE.         *
C    *                                                                     *
C    *                                                                     *
C    *      A  -  THE PLANT A MATRIX (NSTATE X NSTATE)                     *
C    *      B  -  THE CONTROL MATRIX  (NSTATE X NCONTRUL)                  *
C    *      C  -  THE PLANT OBSERV. OR OUTPUT MATRIX (NOUTPUT X NSTATE;    *
C    *              FOR STATE FEEDBACK PROBLEMS)                           *
C    *      F  -  THE FEEDBACK GAIN MATRIX(NCONTROL X NSTATE;              *
C    *              FOR STATE FEEDBACK PROBLEMS)                           *
C    *      FK - THE ONSERVER GAIN MATRIX FOR FILTER PROBLEMS              *
C    *      REALMU - THE INPUT DESIRED REAL POLE LOCATION                  *
C    *      IMAGMU - THE INPUT DESIRED IMAGINARY POLE LOCATION             *
C    *      BF - B*F                                                       *
C    *      FC  -   F*C                                                    *
C    *      BFC -   B*F*C                                                  *
C    *      FKC - FK*C                                                     *
C    *      AMBFC - A - B*F*C                                              *
C    *      AMFKC - A - FK*C                                               *
C    *      AMBF - A - B * F                                               *
```

```
C     *      REIG - REAL COMPUTED EIGENVALUE OF THE SYSTEM                    *  CONOC490
C     *      WW   - PROGRAM WORK SPACE                                        *  CONOC500
C     *      WK   - WORK SPACE USED BY SUBROUTINES                            *  CONOC510
C     *      WA   - DIMENSIONED WORK SPACE                                    *  CONOC520
C     *      WW   - WORKING SPACE                                             *  CONOC530
C     *      OMEGA - FREQUENCY                                                *  CONOC540
C     *      SVGFM - SINGULAR VALUE OF (I +G*H)                                  CONOC550
C     *      SVMI  - SINGULAR VALUE OF (I + INV(F* G ))                       *  CONOC560
C     *      SVAC  - SINGULAR VALUE OF ( I + G * F)                           *  CONOC570
C     *      SVMC  - SINGULAR VALUE OF ( I + INV (G * F))                     *  CONOC580
C     *                ONLY OF INTEREST IF NO. OUTPUTS = NO. INPUTS.          *  CONOC590
C     *      SVHG1   - FIRST SINGULAR VALUE OF (I + H*G)                      *  CONOC600
C     *      SVHG2   - SECOND SINGULAR VALUE OF (I + H*G)                     *  CONOC610
C     *      SVHGX   - MAXIMUM SINGULAR VALUE OF (I + H*G)                    *  CONOC620
C     *      SIGFFG - SQRT(FIRST SING VAL * SECOND SING VAL)                  *  CONOC630
C     *      SVGH1   - MIN ADDITIVE OUTPUT SINGULAR VALUE (I + G*H)           *  CONOC640
C     *      SVGH2   - SECOND SINGULAR VALUE                                  *  CONOC650
C     *      SVGHX   - MAX ADDITIVE OUTPUT SINGULAR VALUE                     *  CONOC660
C     *      SVHGM1- MIN MULTIPLICATIVE INPUT SINGULAR VALUE (I+INV(HG))      *  CONOC670
C     *      SVHGM2  SECOND SINGULAR VALUE                                    *  CONOC680
C     *      SVHGMX- MAX MULTIPLICATIVE INPUT SINGULAR VALUE                  *  CONOC690
C     *      SVGHM1- MIN MULTIPLICATIVE OUTPUT SINGULAR VALUE (I+INV(GH))     *  CONOC700
C     *      SVGHM2 - SECOND SINGULAR VALUE                                   *  CONOC710
C     *      SVGHMX- MAX MULTIPLICATIVE OUTPUT SINGULAR VALUE                 *  CONOC720
C     *      PHGAD  - SQRT (SVHG1*AVHG2)                                      *  CONOC730
C     *      PGHAD  - SQRT(SVGH1*SVGH2)                                       *  CONOC740
C     *      PHGMU  - SQRT (SVHGM1*SVHGM2)                                    *  CONOC750
C     *      PGHMU  - SQRT (SVGHM1*SVGHM2)                                    *  CONOC760
C     *      SVHGA  - ADDITIVE SINGULAR VALUE OF I+HG                         *  CONOC770
C     *      SVGHA  - ADDITIVE SINGULAR VALUE OF I+GH                         *  CONOC780
C     *      SVHGM  - MULTIPLICATIVE SINGULAR VALLE OF I+HG                   *  CONOC790
C     *      SVGHM  - MULTIPLICATIVE SINGULAR VALLE OF I+GH                   *  CONOC800
C     *                                                                       *  CONOC810
C     ************************************************************************  CONOC820
      REAL*4 A(10,10),B(10,10),C(10,10),F(10,10),REALMU(50),IMAGMU(50),        CONOC830
     1FK(10,10),BF(10,10),AMFKC(10,10),                    WK(5000),           CONOC840
     2REIG(50),          WW(10,10),WA(100),          FKC(10,10),SVGHM(10),    CONOC850
     3OMEGA(500),SVHG1(500),SVHG2(500),SVHGX(500),PHGAD(500),PGHAD(500)       CONOC860
     4,SVGH1(500),SVGH2(500),SVGHX(500),SVHGM1(500),       AMBF(10,10),       CONOC870
     5SVHGM2(500),SVHGMX(500),          SVHGA(10),SVGHA(10),SVHGM(10),        CONOC880
     6SVGHM1(500),SVGHM2(500),SVGHMX(500),PHGMU(500),PGHMU(500)               CONOC890
C     ************************************************************************  CONOC900
C     *          THE COMPLEX PARAMETERS USED IN THE PROGRAM                   *  CONOC910
C     *          AX - COMPLEX A                                               *  CONOC920
C     *          BX - COMPLEX B                                               *  CONOC930
C     *          CX - COMPLEX C                                               *  CONOC940
C     *          FX - COMPLEX F                                               *  CONOC950
C     *          FKX - COMPLEXOBSERVER GAIN K                                    CONOC960
```

```
C     *          EIG - EIGENVALUE OF A- B*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                                       * CONO1C00
C     *          CEIG - ORDERED VALUE OF EIG                                     * CONO1C10
C     *          CZ - ORDERED EIGENVECTOR                                        * CONO1C20
C     *          XI - IDENTITY MATRIX                                            * CONO1C30
C     *          XXI - IDENTITY MATRIX                                           * CONO1C40
C     *          UI - INPUT ADDITIVE SINGULAR VECTOR ASSOCIATED WITH SV          * CONO1C50
C     *          VI - INPUT ADDITIVE SINGULAR VECTOR ASSOCIATED WITH SV          * CONO1C60
C     *          UC - OUTPUT ADDITIVE SINGULAR VECTOR (SVAD)                     * CONO1C70
C     *          VC - OUTPUT ADDITIVE SINGULAR VECTOR (SVAD)                     * CONO1C80
C     *          UMI- INPUT MULTIPLICATIVE SINGULAR VECTOR                       * CONO1C90
C     *          VMI - INPUT MULTIPLICATIVE SINGULAR VECTOR                      * CONO1100
C     *          UMC - OUTPUT MULTIPLICATIVE SINGULAR VECTOR                     * CONO1110
C     *          VMO - OUTPUT MULTIPLICATIVE SINGULAR VECTOR                     * CONO1120
C     *          FG - CONTROL * PLANT TRANSFER FUNCTION                          * CONO1130
C     *          GH - PLANT * CONTROL TRANSFER FUNCTION                          * CONO1140
C     *          XIPFG  - I + H*G                                                * CONO1150
C     *          PLAN - PLANT TRANSFER FUNCTION (G)                              * CONO1160
C     *          CONT - CONTROL TRANSFER FUNCTION (H)                            * CONO1170
C     *          FLTFX - G*F                                                     * CONO1180
C     *          XIPGH  - I + G*H                                                * CONO1190
C     *          FGI  -    INV(H*G)                                              * CONO1200
C     *          GHI  -    INV(G*H)                                              * CONO1210
C     *          XIPFGI -  I + INV (H*G)                                         * CONO1220
C     *          XIPGHI -  I + INV (G*H)                                         * CONO1230
C     *          EIGK - OBSERVER EIGENVALUE                                      * 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
     1,10),BX(10,10),CX(10,10),FX(10,10),XIPHG(10,10),XI(10,10),XIPHGI(1CONO1280
     20,10),UI(10,10),VI(10,10),PLAN(10,10),XXI(10,10),UC(10,10),VC(10,1CONO1290
     30),UMI(10,10),VMI(10,10),UMC(10,10),VMC(10,10),GH(10,10),XIPGH(10,CONO1300
     410),XIPGHI(10,10),FGI(10,10),HG(10,10),GHI(10,10),ZK(10,10),FKX(1CONO1310
     5,10),EIGK(50),CONT(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 *CONO1390
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                               *CONO1440
```

```
C     *      IDG - 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     *      VLB1 - LOWER BOUND FOR PASS 2                                       *CCNO1590
C     *      VUB1 - UPPER BOUND FOR PASS 2                                       *CCNO1600
C     ************************************************************************CCNO1610
C                                                                               CCNO1620
      DIMENSION X(50),VLB(50),VUB(50),G(50),IDG(50),IC(50),DF(50),XA(50,CCNO1630
     150),WK1(10000),IWK(1000),IF(10,10),XIN(50),IR(50),KEY(50),R(50),IFCCNO1640
     2K(10,10),X1(50),G1(50),VLB1(50),VUB1(50)                                  CCNO1650
C     EXTERNAL FCN                                                              CONO1660
C     COMMON                                                                    CCNO1670
      DATA NRA/50/,NCA/50/,NRWK/10000/,NRIWK/1000/                              CONO1680
C     ************************************************************************CCNO1690
C     ************************************************************************ CCNO1700
C     ************************************************************************ CONO1710
C     *          DATA FORMAT PREPARATION                                      * CCNO1720
C     *  CARD 1  KODE, KONTRL; FORMAT (I5,I5)                                 * CCNO1730
C     *  KODE IS USER NUMBER TO IDENTIFY THE DATA RUN                         * CONO1740
C     *  KONTRL SETS FLAG TO TELL PROGRAM HOW TO HANDLE RUN MAY BE:           * CCNO1750
C     *              0  ENTER OPTIMIZER ON FIRST PASS THROUGH THE PROGRAM     * CONO1760
C     *              1  COMPUTE VALUES THROUGH PROGRAM WITHOUT OPTIMIZER      * CONO1770
C     *                 ON FIRST PASS THEN OPTIMIZE                           * CONO1780
C     *              2 OR MORE COMPUTE VALUES TO ACT OPTIMIZE                 * CONO1790
C     *  CARD 2  WMAX,WI,DELW; FORMAT(3F10.0)                                 * CONO1800
C     *          WMAX - MAXIMUM FREQUENCY TO BE USED                          * CCNO1810
C     *          WI - STARTING FREQUENCY                                      * CONO1820
C     *          DELW - FREQUENCY STEP FOR FIRST DECADE , THIS STEP IS        * CCNO1830
C     *                 MODIFIED BY PROGRAM AT EACH DECADE SHIFT              * CONO1840
C     *  CARD 3  WT1,WT2,WT3; FORMAT(3F10.0)                                  * CCNO1850
C     *          WT1 WEIGHTING FACTOR TO CAUSE THE OPTIMIZER TO USE THE       * CCNO1860
C     *              POLE PLACEMENT PORTION OF THE OBJECTIVE FUNCTION         * CCNO1870
C     *               IN THE OPTIMIZATION ROUTINE                             * CCNO1880
C     *          WT2 WEIGHTING FACTOR WHICH CAUSES THE OPTIMIZER TO           * CCNO1890
C     *              CONSIDER THE INPUT SINGULAR VALUE OF (I+FG) IN           * CCNO1900
C     *              THE OPTIMIZATION OBJECTIVE FUNCTION                      * CCNO1910
C     *          WT3 WEIGHTING FACTOR WHICH CAUSES THE OPTIMIZER TO           * CCNO1920
```

```
C     *            USE THE OUTPUT SINGULAR VALUE (1+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(3F10.0,I5)                         * CONO1960
C     *          SVMINI - DESIRED SINGULAR VALUE LEVEL FOR THE S.V. OF            * CONO1970
C     *               (1+FG).FOUND USING UNIVERSAL GAIN AND PHASE CHART           * CONO1980
C     *             TO GIVE A DESIRED PHASE AND GAIN MARGIN                       * 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     *                                                                           * CONO2100
C     *  CARD 6  IGRAD,NDV,NCON,ISTRAT,IOPT,ICNEC,IPRINT,INFO                     * CONO2110
C     *            FORMAT(8I5)                                                    * CONO2120
C     *          IGRAD - PARAMETER CONCERNING GRADIENT COMP SEE ADS               * CONO2130
C     *          NDV - NUMBER OF DESIGN VARIABLES                                 * CONO2140
C     *          NCON - NUMBER OF CONSTRAINTS                                     * CONO2150
C     *          ISTRAT - OPTIMIZER STRATEGY SEE ADS MANUAL                       * CONO2160
C     *          IOPT - OPTIMIZATION METHOD SEE ADS                               * CONO2170
C     *          ICNED - ONE DIMENSIONAL SEARCH TECHNIQUE SEE ADS                 * CONO2180
C     *          IPRINT - ADS PRINT CONTROL SEE ADS                               * CONO2190
C     *          INFO - OPTIMIZER CONTROL PARAMETER SEE ADS                       * CONO2200
C     *CARD 7 NROWA,NCOLA,NROWB,NCOLB,NROWC,NCOLC,NROWF,NCOLF,NMU                * CONO2210
C     *            FORMAT(9I2)                                                    * CONO2220
C     *          NUMBER OF ROWS AND COLUMNS OF INPUT MATRICES AND                 * CONO2230
C     *          NUMBER OF POLES TO BE INPUT FOR PLACEMENT                        * CONO2240
C     *  CARD 8 A MATRIX ; READ BY ROWS AS:                                       * CONO2250
C     *               A(1,1)     A(1,2)                                           * CONO2260
C     *               A(2,1)     A(2,2)                                           * CONO2270
C     *  THERE WILL BE ONE OR MORE CARDS FOR EACH ROW OF A                        * CONO2280
C     *  AFTER THE PROPER CARDS ARE READ FOR A THE NEXT MATRIX MUST               * CONO2290
C     *  BE READ IN. THE REMAINDER OF THIS COMMENT WILL REFER TO EACH NEW          * CONO2300
C     *  DATA SET AS A CARD. REMEMBER, THERE MAY BE SEVERAL CARDS PER             * CONO2310
C     *  SET SO THE NUMBERS DON'T CORRESPOND DIRECTLY WITH  THE DATA              * CONO2320
C     *  LISTED IN THE INPUT FILE.                                                * CONO2330
C     *  CARD 9 B MATRIX; READ ROW BY ROW                                         * CONO2340
C     *  CARD 10 C MATRIX; READ ROW BY ROW                                        * CONO2350
C     *  CARD 11 F MATRIX; READ ROW BY ROW                                        * CONO2360
C     *  CARD 12 FK MATRIX (OBSERVER GUESS); READ ROW BY ROW                      * CONO2370
C     *  CARD 13 IF MATRIX ; ROW BY ROW                                           * CONO2380
C     *            FORMAT(8I2) ; THIS MATRIX DEFINES THE DESIGN VARIABLE          * CONO2390
C     *            TO THE OPTIMIZER. THE ENTRY WOULD BE OF THE FORM:              * CONO2400
```

```
C     *            1 2                                                    *  CCNO2410
C     *            0 3                                                    *  CCNO2420
C     *          THIS WOULD TELL THE OPTIMIZER TO MAKE F(1,1) THE FIRST   *  CCNO2430
C     *          DESIGN VARIABLE (X(1)) AND SO FORTH. THE ZERO TELLS THE  *  CCNO2440
C     *          OPTIMIZER THAT F(2,1) (IN THIS CASE) IS NOT A DESIGN     *  CCNO2450
C     *          VARIABLE AND IS NOT TO BE CHANGED.                       *  CCNO2460
C     *  CARD 14  IFK THE DESIGN VARIABLE SPECIFICATION FOR THE           *  CCNO2470
C     *           OBSERVER DONE AS ABOVE                                  *  CCNO2480
C     *  CARD 15  VLB,VUB; FORMAT(2F10.0)                                 *  CCNO2490
C     *           THERE WILL BE ONE CARD FOR EACH DESIGN VARIABLE         *  CCNO2500
C     *  CARD 16  REALMU,IMAGMU ; FORMAT (2F10.0)                         *  CCNO2510
C     *           INPUT REAL AND IMAGINARY DESIRED POLE LOCATION          *  CCNO2520
C     *           ONE POLE PER CARD                                       *  CCNO2530
C     *  CARD 17 IGRAD,NDV,NCON,ISTRAT,IOPT,IONED,IPRINT,INFO             *  CCNO2540
C     *           SAME FORMAT AS PREVIOUS CARD BUT FOR SECOND OPTIMIZER   *  CCNO2550
C     *  CARD 18  VLB1,VUB1; FORMAT(2F10.0)*                              CCNO2560
C     *           THERE WILL BE ONE CARD FOR EACH DESIGN VARIABLE         *  CCNO2570
C     *                                                                   *  CCNO2580
C     ********************************************************************  CCNO2590
C     ********************************************************************  CCNO2600
C                                                                            CCNO2610
C         INPUT DATA INTO THE PROGRAM TO SET UP THE SYSTEM                   CCNO2620
C            XDOT=AX+BU ; Y=CX                                               CCNO2630
C                                                                            CCNO2640
C     ********************************************************************   CCNO2650
C                                                                            CCNO2660
C       INPUT RUN CODE AND PROGRAM CONTROL                                   CCNO2670
        READ (1,1040) KODE,KONTRL                                            CCNO2680
C                                                                            CCNO2690
C       INPUT MAXN, INITIAL FREQUENCY,DELTA FREQ, OPTIMIZER WEIGHTS          CCNO2700
C                                                                            CCNO2710
        READ (1,1000) WMAX,WI,DELW                                           CCNO2720
        READ (1,780) WT1,WT2,WT3                                             CCNO2730
        READ (1,790) SVMINI,SVMINO,RJ,NIDG                                   CCNO2740
        READ (1,780) EIGX,EIGM,EIRT                                          CCNO2750
        W=WI                                                                 CCNO2760
        NC=1                                                                 CCNO2770
C                                                                            CCNO2780
C       INPUT OPTIMIZER INPUT DATA (SEE ADS MANUAL)                          CCNO2790
C                                                                            CCNO2800
        READ (1,1050) IGRAD,NDV,NCON,ISTRAT,IOPT,IONED,IPRINT,INFO           CCNO2810
C                                                                            CCNO2820
C     INPUT THE PARAMETER VALUES AND THE MATRICES A,B,C,F,FK                 CCNO2830
C                                                                            CCNO2840
        READ (1,1060) NROWA,NCOLA,NROWB,NCOLB,NROWC,NCOLC,NROWF,NCOLF,NMU    CCNO2850
        READ (1,800) NROWK,NCOLK                                             CCNO2860
        CALL READ (A,NROWA,NCOLA)                                            CCNO2870
        CALL READ (B,NROWB,NCOLB)                                            CCNO2880
```

```
      CALL READ (C,NROWC,NCOLC)                                          CONO2890
      CALL READ (F,NROWF,NCOLF)                                          CONO2900
      CALL READ (FK,NROWK,NCOLK)                                         CONO2910
C                                                                        CONO2920
C     SPECIFY THE DESIGN VARIABLES WITHIN THE F MATRIX                   CONO2930
C                                                                        CONO2940
      DO 10 J=1,NROWF                                                    CONO2950
10    READ (1,1070) (IF(J,K),K=1,NCOLF)                                  CONO2960
      DO 20 J=1,NROWK                                                    CONO2970
20    READ (1,1070) (IFK(J,K),K=1,NCOLK)                                 CONO2980
C                                                                        CONO2990
C     SET THE DESIGN VARIABLE BOUNDS                                     CONO3000
C                                                                        CONO3010
      DO 30 J=1,NDV                                                      CONO3020
30    READ (1,1080) VLB(J),VUB(J)                                        CONO3030
      DO 60 J=1,NROWF                                                    CONO3040
      DO 50 K=1,NCOLF                                                    CONO3050
      IF (IF(J,K).EQ.0) GO TO 40                                         CONO3060
      KK=IF(J,K)                                                         CONO3070
      X(KK)=F(J,K)                                                       CONO3080
40    CONTINUE                                                           CONO3090
50    CONTINUE                                                           CONO3100
60    CONTINUE                                                           CONO3110
C                                                                        CONO3120
C     SPEC DESIGN VAR WITHIN THE FK MATRIX                               CONO3130
C                                                                        CONO3140
C                                                                        CONO3150
C     READ THE DESIRED EIGENVALUES AS REAL(X)+REAL(Y)*I                  CONO3160
      DO 70 I=1,NMU                                                      CONO3170
70    READ (1,1150) REALMU(I),IMAGMU(I)                                  CONO3180
      DO 80 I=1,NMU                                                      CONO3190
      MU(I)=CMPLX(REALMU(I),IMAGMU(I))                                   CONO3200
80    CONTINUE                                                           CONO3210
C                                                                        CONO3220
C   SORT THE INPUT EIGEN VALUES                                          CONO3230
C                                                                        CONO3240
      DO 90 I=1,NMU                                                      CONO3250
90    IR(I)=I                                                            CONO3260
      CALL VSRTR (REALMU,NMU,IR)                                         CONO3270
      DO 100 J=1,NMU                                                     CONO3280
      K=IR(J)                                                            CONO3290
100   OMU(J)=MU(K)                                                       CONO3300
      JJ=NMU-1                                                           CONO3310
      DO 120 J=1,JJ                                                      CONO3320
      IF (REAL(OMU(J)).NE.REAL(OMU(J+1))) GO TO 110                      CONO3330
      IF (AIMAG(OMU(J)).LT.AIMAG(OMU(J+1))) GO TO 110                    CONO3340
      TEMP1=OMU(J)                                                       CONO3350
      OMU(J)=OMU(J+1)                                                    CONO3360
```

```
      CML(J+1)=TEMP1                                                    CCNO3370
110   CONTINUE                                                          CCNO3380
120   CONTINUE                                                          CCNO3390
C  ****************************************************************** CCNO3400
C                                                                       CCNO3410
C        WRITE THE INPUT DATA FOR CHECKING AND REFERENCE                CCNO3420
C                                                                       CCNO3430
C  ****************************************************************** CCNO3440
      WRITE (6,1050) KODE                                               CCNO3450
      WRITE (6,1010) WMAX,WI,DELW                                       CCNO3460
      WRITE (6,830) WT1,WT2,WT3                                         CCNO3470
      WRITE (6,840) SVMINI,SVMING,RJ,NIDG                               CCNO3480
      WRITE (6,820) EIGX,EIGM,EIRT                                      CCNO3490
      WRITE (3,850) NRCWA,NML                                           CCNO3500
      WRITE (6,1100)                                                    CCNO3510
      CALL WRITE (A,NRCWA,NCOLA)                                        CCNO3520
      WRITE (6,1110)                                                    CCNO3530
      CALL WRITE (B,NRCWB,NCOLB)                                        CCNO3540
      WRITE (6,1120)                                                    CCNO3550
      CALL WRITE (C,NRCWC,NCOLC)                                        CCNO3560
      WRITE (6,1130)                                                    CCNO3570
      CALL WRITE (F,NRCWF,NCOLF)                                        CCNO3580
      WRITE (6,660)                                                     CCNO3590
      CALL WRITE (FK,NROWK,NCOLK)                                       CCNO3600
      WRITE (6,1140)                                                    CCNO3610
      CALL CVECWR (OMU,NML)                                             CCNO3620
      IF (KCNTRL.GT.0) GO TO 140                                        CCNO3630
C  ****************************************************************** CCNO3640
C                                                                       CCNO3650
C        CALL THE ADS OPTIMIZER TO SOLVE FOR THE BEST F'S               CCNO3660
C                                                                       CCNO3670
C  ****************************************************************** CCNO3680
      INFO=-2                                                           CCNO3690
      CALL ADS (INFO,ISTRAT,IOPT,IONED,IPRINT,IGRAD,NDV,NCON,X,VLB,VUB,GCCNO3700
     1BJ,G,IDG,NGT,IC,DF,XA,NRA,NCA,WK1,NRWK,IWK,NRIWK)                 CCNO3710
      IWK(2)=1                                                          CCNO3720
130   CALL ADS (INFO,ISTRAT,IOPT,IONED,IPRINT,IGRAD,NDV,NCON,X,VLB,VUB, CCNO3730
     1BJ,G,IDG,NGT,IC,DF,XA,NRA,NCA,WK1,NRWK,IWK,NRIWK)                 CCNO3740
      IF (INFO.EQ.0) GO TO 310                                          CCNO3750
      IF (INFO.GT.1) GO TO 300                                          CCNO3760
C  ****************************************************************** CCNO3770
C                                                                       CCNO3780
C      THIS PORTION OF THE PROGRAM DOES THE ACTUAL COMPUTATION          CCNO3790
C                                                                       CCNO3800
C  ****************************************************************** CCNO3810
C                                                                       CCNO3820
C     EQUATE DESIGN VECTOR X WITH THE DESIRED SYSTEM MATRIX             CCNO3830
140   DO 170 J=1,NROWF                                                  CCNO3840
```

```
      DO 160 K=1,NCOLF                                                  CCN03E50
      IF (IF(J,K).EQ.0) GO TO 150                                       CCN03E60
      KK=IF(J,K)                                                        CCN03E70
      F(J,K)=X(KK)                                                      CCN03E80
150   CONTINUE                                                          CCN03E90
160   CONTINUE                                                          CCN03F00
170   CONTINUE                                                          CCN03F10
C                                                                       CCN03F20
C                                                                       CCN03F30
C     COMPUTE THE  SYSTEM MATRIX FOR EIGEN COMPUTATION                  CCN03F40
      CALL MPLL (B,F,NROWB,NCOLB,NCOLF,BF)                              CCN03F50
      DO 190 I=1,NROWA                                                  CCN03F60
      DO 180 J=1,NCOLA                                                  CCN03F70
180   AMBF(I,J)=A(I,J)-BF(I,J)                                          CCN03F80
190   CONTINUE                                                          CCN03F90
C                                                                       CCN04C00
C     COMPUTE AND SORT THE EIGENVALUES AND VECTORS                      CCN04C10
C                                                                       CCN04C20
      CALL EIGRF (AMBF,NROWA,10,2,EIG,Z,10,WK,IER)                      CCN04C30
      DO 200 J=1,NROWA                                                  CCN04C40
      REIG(J)=REAL(EIG(J))                                              CCN04C50
200   CONTINUE                                                          CCN04C60
      DO 210 J=1,NROWA                                                  CCN04C70
210   KEY(J)=J                                                          CCN04C80
      CALL VSRTR (REIG,NROWA,KEY)                                       CCN04C90
      DO 230 J=1,NCOLA                                                  CCN04100
      K=KEY(J)                                                          CCN04110
      CEIG(J)=EIG(K)                                                    CCN04120
      DO 220 KK=1,NROWA                                                 CCN04130
      OZ(KK,J)=Z(KK,K)                                                  CCN04140
220   CONTINUE                                                          CCN04150
230   CONTINUE                                                          CCN04160
      JJ=NCOLA-1                                                        CCN04170
      DO 260 J=1,JJ                                                     CCN04180
      IF (REAL(CEIG(J)).NE.REAL(CEIG(J+1))) GO TO 250                   CCN04190
      IF (AIMAG(CEIG(J)).LT.AIMAG(CEIG(J+1))) GO TO 250                 CCN04200
      TEMP1=CEIG(J)                                                     CCN04210
      CEIG(J)=CEIG(J+1)                                                 CCN04220
      CEIG(J+1)=TEMP1                                                   CCN04230
      DO 240 KK=1,NROWA                                                 CCN04240
      TEMP2=CZ(KK,J)                                                    CCN04250
      CZ(KK,J)=CZ(KK,J+1)                                               CCN04260
      CZ(KK,J+1)=TEMP2                                                  CCN04270
240   CONTINUE                                                          CCN04280
250   CONTINUE                                                          CCN04290
260   CONTINUE                                                          CCN04300
C                                                                       CCN04310
      COST=0.                                                           CCN04320
```

```
      DO 270 J=1,NMU                                                    CCN04330
      COST=COST+((REAL(OMU(J))-REAL(OEIG(J)))**2+(AIMAG(OMU(J))-AIMAG(OECCN04340
     1IG(J)))**2)                                                       CCN04350
270   CONTINUE                                                          CCN04360
      OBJ=WT1*COST                                                      CCN04370
      IF (NCON.EQ.0) GO TO 290                                          CCN04380
      DO 280 J=1,NMU                                                    CCN04390
      IDG(J)=NIDG                                                       CCN04400
C     R(J)=RJ                                                           CCN04410
      R(J)=RJ*SQRT((REAL(OMU(J))-REAL(OEIG(J)))**2+(AIMAG(OMU(J))-AIMAG(CCN04420
     1OEIG(J)))**2)                                                     CCN04430
280   G(J)=SQRT((REAL(OMU(J))-REAL(OEIG(J)))**2+(AIMAG(OMU(J))-AIMAG(CEICCN04440
     1G(J)))**2)-R(J)                                                   CCN04450
290   IF (KONTRL.GT.0) GO TO 310                                        CCN04460
      GO TO 130                                                         CCN04470
300   CONTINUE                                                          CCN04480
C  THE GRADIENT COMPUTATION ROUTINE GOES HERE IF NEEDED                 CCN04490
C                                                                       CCN04500
310   CONTINUE                                                          CCN04510
      WRITE (6,1160)                                                    CCN04520
      CALL CVECWR (OEIG,NCOLA)                                          CCN04530
      DO 320 J=1,NMU                                                    CCN04540
320   WRITE (3,910) OEIG(J),EIGK(J)                                     CCN04550
      WRITE (6,1170)                                                    CCN04560
      CALL CWRITE (OZ,NROWA,NCOLA)                                      CCN04570
      WRITE (6,1180)                                                    CCN04580
      CALL WRITE (F,NRCWF,NCOLF)                                        CCN04590
      READ (1,1050) IGRAD,NDV,NCON,ISTRAT,IOPT,IONED,IPRINT,INFO        CCN04600
      DO 330 J=1,NDV                                                    CCN04610
330   READ (1,1080) VLB1(J),VUB1(J)                                     CCN04620
C                                                                       CCN04630
      DO 360 J=1,NROWK                                                  CCN04640
      DO 350 K=1,NCOLK                                                  CCN04650
      IF (IFK(J,K).EQ.0) GO TO 340                                      CCN04660
      KKK=IFK(J,K)+KK                                                   CCN04670
      X1(KKK)=FK(J,K)                                                   CCN04680
340   CONTINUE                                                          CCN04690
350   CONTINUE                                                          CCN04700
360   CONTINUE                                                          CCN04710
      IF (KONTRL.GT.0) GO TO 141                                        CCN04720
      INFO=-2                                                           CCN04730
C     ****************************************************************  CCN04740
C     *          CALL ADS ROUTINE TO FIND FILTER FEEDBACK GAINS FOR  *  CCN04750
C     *             THE ROBUSTNESS RECOVERY WITH THE OBSERVER GAINS  *  CCN04760
C     ****************************************************************  CCN04770
      CALL ADS (INFO,ISTRAT,IOPT,IONED,IPRINT,IGRAD,NDV,NCON,X1,VLB1,VUBCCN04780
     11,OBJ1,C1,IDG,NGT,IC,DF,XA,NRA,NCA,WK1,NRWK,IWK,NRIWK)            CCN04790
      IWK(2)=1                                                          CCN04800
```

```
  370  CALL ADS (INFO,ISTRAT,IOPT,IONED,IPRINT,IGRAD,NDV,NCON,X1,VLB1,VUBCCN04E10
     11,CBJ1,G1,ICG,NGT,IC,DF,XA,NRA,NCA,WK1,NRWK,IWK,NRIWK)              CCN04E20
       IF (INFC.EQ.0) GO TO 700                                           CCN04E30
       IF (INFC.GT.1) GO TO 690                                           CCN04E40
C                                                                         CCN04E50
C                                                                         CCN04E60
  141 CONTINUE                                                            CCN04E70
       DO 400 J=1,NROWK                                                   CCN04880
       DO 390 K=1,NCOLK                                                   CCN04E90
       IF (IFK(J,K).EQ.0) GO TO 380                                       CCN04900
       KKKK=IFK(J,K)                                                      CCN04910
C       KKKK=KK+KKK                                                       CCN04920
       FK(J,K)=X1(KKKK)                                                   CCN04930
380    CONTINUE                                                           CCN04940
390    CONTINUE                                                           CCN04950
400    CONTINUE                                                           CCN04960
       CALL MPLL (FK,C,NROWK,NCOLK,NCOLC,FKC)                             CCN04970
       DO 420 I=1,NROWA                                                   CCN04980
       DO 410 J=1,NCOLA                                                   CCN04990
410    AMFKC(I,J)=A(I,J)-FKC(I,J)                                         CCN05C00
420    CONTINUE                                                           CCN05C10
       CALL EIGRF (AMFKC,NROWA,10,2,EIGK,ZK,10,WK,IER)                    CCN05C20
C                                                                         CCN05C30
C       COMPLEX ALL THE INPUT SYSTEM MATRICES FOR THE TRANSFER FUNCTION   CCN05C40
C                                                                         CCN05C50
       CALL CPLXCV (AX,A,NROWA,NCCLA)                                     CCN05C60
       CALL CPLXCV (BX,B,NROWB,NCCLB)                                     CCN05C70
       CALL CPLXCV (CX,C,NROWC,NCCLC)                                     CCN05C80
       CALL CPLXCV (FX,F,NROWF,NCCLF)                                     CCN05C90
       CALL CPLXCV (FKX,FK,NROWK,NCOLK)                                   CCN05100
C                                                                         CCN05110
C   CALL THE PLANT TRANSFER MATRIX ROUTINE AND FORM THE SYSTEM            CCN05120
C     RETURN DIFFERENCE MATRICES AS REQUIRED                              CCN05130
C                                                                         CCN05140
       DEL=DELW                                                           CCN05150
       WII=WI                                                             CCN05160
       NC=1                                                               CCN05170
       W=WI                                                               CCN05180
       SMINAI=C.                                                          CCN05190
       SMINMI=C.                                                          CCN05200
       SMINAO=0.                                                          CCN05210
       SMINMO=C.                                                          CCN05220
       SMAXAI=C.                                                          CCN05230
       SMAXMI=0.                                                          CCN05240
       SMAXAO=C.                                                          CCN05250
       SMAXMO=C.                                                          CCN05260
430    CALL PLANT (PLAN,AX,BX,CX,W,NRCWA,NCCLA,NROWB,NCOLB,NROWC,NCOLC)   CCN05270
       CALL CONTRP (CONT,AX,FX,FKX,FKC,BF,NROWA,NCCLA,NROWF,NCOLF,NROWK,NCCN05280
```

```
     1COLK,W)                                                                  CCNO5290
      CALL CMATML (CCNT,PLAN,NROWF,NCOLK,NCOLB,HG)                             CCNO5300
      CALL CMATML (PLAN,CCNT,NROWC,NCOLB,NCOLK,GH)                             CCNO5310
      DO 450 J=1,NROWF                                                         CCNO5320
      DO 440 K=1,NCOLB                                                         CCNO5330
  440 XI(J,K)=0.0                                                              CCNO5340
  450 CONTINUE                                                                 CCNO5350
      DO 460 J=1,NROWF                                                         CCNO5360
  460 XI(J,J)=1.0                                                              CCNO5370
      DO 480 J=1,NROWC                                                         CCNO5380
      DO 470 K=1,NCOLK                                                         CCNO5390
      XXI(J,K)=0.0                                                             CCNO5400
  470 CONTINUE                                                                 CCNO5410
  480 CONTINUE                                                                 CCNO5420
      DO 490 J=1,NCOLK                                                         CCNO5430
  490 XXI(J,J)=1.0                                                             CCNO5440
      DO 510 J=1,NROWF                                                         CCNO5450
      DO 500 K=1,NCOLB                                                         CCNO5460
C500    XIPHG(J,K)=HG(J,K)                                                     CCNO5470
  500 XIPHG(J,K)=XI(J,K)+HG(J,K)                                               CCNO5480
  510 CONTINUE                                                                 CCNO5490
      DO 530 J=1,NROWC                                                         CCNO5500
      DO 520 K=1,NCOLK                                                         CCNO5510
      XIPGH(J,K)=XXI(J,K)+GH(J,K)                                              CCNO5520
  520 CONTINUE                                                                 CCNO5530
  530 CONTINUE                                                                 CCNO5540
      DO 550 J=1,NROWF                                                         CCNO5550
      DO 540 K=1,NROWF                                                         CCNO5560
      HGI(J,K)=0.0                                                             CCNO5570
  540 HGI(J,J)=1.0                                                             CCNO5580
  550 CONTINUE                                                                 CCNO5590
      DO 570 J=1,NROWC                                                         CCNO5600
      DO 560 K=1,NROWC                                                         CCNO5610
      GHI(J,K)=0.0                                                             CCNO5620
  560 GHI(J,J)=1.                                                              CCNO5630
  570 CONTINUE                                                                 CCNO5640
      CALL LEQT1C (HG,NROWF,10,HGI,NROWF,10,0,WA,IER)                          CCNO5650
C     CALL LEQT1C(GH,NROWC,10,GHI,NROWC,10,0,WA,IER)                           CCNO5660
      DO 590 J=1,NROWF                                                         CCNO5670
      DO 580 K=1,NCOLB                                                         CCNO5680
      XIPHGI(J,K)=XI(J,K)+HGI(J,K)                                             CCNO5690
  580 CONTINUE                                                                 CCNO5700
  590 CONTINUE                                                                 CCNO5710
      DO 610 J=1,NROWC                                                         CCNO5720
      DO 600 K=1,NROWC                                                         CCNO5730
      XIPGHI(J,K)=XXI(J,K)+GHI(J,K)                                            CCNO5740
  600 CONTINUE                                                                 CCNO5750
  610 CONTINUE                                                                 CCNO5760
```

```
C                                                                       CCNO5770
C     DO SINGULAR VALUE DECOMP AND QUANTIFY ALL DESIRED SV'S            CCNO5780
C                                                                       CCNO5790
      CALL CSVD (XIPHG,10,10,NROWF,NRCWF,0,NRCWF,NROWF,SVHGA,UI,VI)     CCNO5800
      CALL CSVD (XIPGH,10,10,NROWC,NROWC,0,NROWC,NROWC,SVGHA,UO,VO)     CCNO5810
      CALL CSVD (XIPHGI,10,10,NRCWF,NROWF,0,NROWF,NRCWF,SVHGM,UMI,VMI)  CCNO5820
      CALL CSVD (XIPGHI,10,10,NROWC,NROWC,0,NROWC,NROWC,SVGHM,UMC,VMC)  CCNO5830
      OMEGA(NC)=W                                                       CCNO5840
      SVHG1(NC)=SVHGA(NRCWF)                                            CCNO5850
      SVHG2(NC)=SVHGA(NRCWF-1)                                          CCNO5860
      SVHGX(NC)=SVHGA(1)                                                CCNO5870
      SVGH1(NC)=SVGHA(NRCWC)                                            CCNO5880
      SVGH2(NC)=SVGHA(NRCWC-1)                                          CCNO5890
      SVGHX(NC)=SVGHA(1)                                                CCNO5900
      SVHGM1(NC)=SVHGM(NROWF)                                           CCNO5910
      SVHGM2(NC)=SVHGM(NRCWF-1)                                         CCNO5920
      SVHGMX(NC)=SVHGM(1)                                               CCNO5930
      SVGHM1(NC)=SVGHM(NRCWC)                                           CCNO5940
      SVGHM2(NC)=SVGHM(NRCWC-1)                                         CCNO5950
      SVGHMX(NC)=SVGHM(1)                                               CCNO5960
      PHGAD(NC)=SQRT(SVHG1(NC)*SVHG2(NC))                               CCNO5970
      PGHAD(NC)=SQRT(SVGH1(NC)*SVGH2(NC))                               CCNO5980
      PHGMU(NC)=SQRT(SVHGM1(NC)*SVHGM2(NC))                             CCNO5990
      PGHMU(NC)=SQRT(SVGHM1(NC)*SVGHM2(NC))                             CCNO6000
      SMINAI=(SMINAI+(AMAX1(0.0,SVMIN1-SVHG1(NC)))**2)                  CCNO6010
      SMINAO=(SMINAO+(AMAX1(0.0,SVMIN0-SVGH1(NC)))**2)                  CCNO6020
C                                                                       CCNO6030
C     COMPUTE LOOP FOR ALL DESIRED FREQUENCY BANDWIDTH                  CCNO6040
C                                                                       CCNO6050
      W=W+DEL                                                           CCNO6060
      WD=10.0*WII                                                       CCNO6070
      IF (W.LT.WC) GO TO 620                                            CCNO6080
      DEL=10.0*DEL                                                      CCNO6090
      WII=10.0*WII                                                      CCNO6100
620   NC=NC+1                                                           CCNO6110
      IF (W.LE.WMAX) GO TO 430                                          CCNO6120
      MAXN=NC-1                                                         CCNO6130
C                                                                       CCNO6140
C     FORM QUANTITIES TO USE IN THE COST CRITERIA OF OPTIMIZER          CCNO6150
C                                                                       CCNO6160
      SVMIN=SVHG1(1)                                                    CCNO6170
      DO 630 J=1,MAXN                                                   CCNO6180
      IF (SVGH1(J).GE.SVMIN) GO TO 630                                  CCNO6190
      SVMIN=SVGH1(J)                                                    CCNO6200
630   CONTINUE                                                          CCNO6210
C                                                                       CCNO6220
C                                                                       CCNO6230
C     THE OBJECTIVE FUNCTION IS INSERTED HERE                           CCNO6240
```

```
      OBJ1=WT2*SMINAI+WT3*SMINAO                                        CONO6250
C     OBJ=WT2*SMINAI                                                    CONO6260
      IF (NCCN.EQ.0) GO TO 680                                          CONO6270
      DO 640 J=1,NCON                                                   CONO6280
640   IDG(J)=NICG                                                       CONO6290
      SUM1=0.0                                                          CONO6300
      DO 650 J=1,NMU                                                    CONO6310
C     JJ=J+NML                                                          CONO6320
      JJ=J                                                              CONO6330
      G1(JJ)=SUM1+(AMAX1(0.0,REAL(EIGK(JJ)-EIGX))**2                    CONO6340
650   CONTINUE                                                          CONO6350
      SUM2=0.                                                           CONO6360
      DO 660 J=1,NMU                                                    CONO6370
C     JJ=J+2*NML                                                        CONO6380
      JJ=J+NML                                                          CONO6390
      G1(JJ)=SUM2+(AMAX1(0.0,EIGM-REAL(EIGK(J))))**2                    CONO6400
660   CONTINUE                                                          CONO6410
      SUM3=0.                                                           CONO6420
      DO 670 J=1,NMU                                                    CONO6430
C     JJ=J+3*NMU                                                        CONO6440
      JJ=J+2*NML                                                        CONO6450
      G1(JJ)=SUM3+(AMAX1(0.,(AIMAG(EIGK(J))/REAL(EIGK(J)))-EIRT))**2    CONO6460
670   CONTINUE                                                          CONO6470
680   IF (KCNTRL.GT.0) GO TO 700                                        CONO6480
      GO TO 690                                                         CONO6490
690   CONTINUE                                                          CONO6500
C  THE GRADIENT COMPUTATION ROUTINE GOES HERE IF NEEDED                 CONO6510
700   CONTINUE                                                          CONO6520
C                                                                       CONO6530
C  OUTPUT THE COMPUTED FINAL EIGEN VALUES AND VECTORS FROM OPT F'S      CONO6540
C                                                                       CONO6550
      IF (KCNTRL.EQ.0) GO TO 710                                        CONO6560
      WRITE (6,870)                                                     CONO6570
      GO TO 720                                                         CONO6580
710   WRITE (6,880)                                                     CONO6590
720   WRITE (6,890)                                                     CONO6600
      CALL CVECWR (EIGK,NCOLA)                                          CONO6610
      WRITE (6,900)                                                     CONO6620
      CALL CWRITE (ZK,NRCWA,NCOLA)                                      CONO6630
      WRITE (6,1160)                                                    CONO6640
      CALL CVECWR (OEIG,NCOLA)                                          CONO6650
      DO 730 J=1,NMU                                                    CONO6660
730   WRITE (6,910) OEIG(J),EIGK(J)                                     CONO6670
      WRITE (6,1170)                                                    CONO6680
      CALL CWRITE (OZ,NRCWA,NCOLA)                                      CONO6690
      WRITE (6,1160)                                                    CONO6700
      CALL WRITE (F,NRCWF,NCOLF)                                        CONO6710
      WRITE (6,920)                                                     CONO6720
```

```
      CALL WRITE (FK,NROWK,NCOLK)                                       CCN06730
C                                                                       CCN06740
C OUTPUT THE OPTIMIZED SINGULAR VALUE DATA                              CCN06750
C                                                                       CCN06760
      WRITE (6,1020)                                                    CCN06770
      WRITE (3,940) MAXN                                                CCN06780
      DO 740 J=1,MAXN                                                   CCN06790
      WRITE (3,950) OMEGA(J),SVHG1(J),SVGH1(J),SVHGM1(J),SVGHM1(J)      CCN06800
740   WRITE (6,1020) OMEGA(J),SVHG1(J),SVGH1(J),SVHGM1(J),SVGHM1(J)     CCN06810
      WRITE (6,960)                                                     CCN06820
      DO 750 J=1,MAXN                                                   CCN06830
      WRITE (3,980) OMEGA(J),SVHG2(J),SVGH2(J),SVHGM2(J),SVGHM2(J)      CCN06840
750   WRITE (6,990) OMEGA(J),SVHG2(J),SVGH2(J),SVHGM2(J),SVGHM2(J)      CCN06850
      WRITE (6,970)                                                     CCN06860
      DO 760 J=1,MAXN                                                   CCN06870
      WRITE (3,980) OMEGA(J),SVHGX(J),SVGHX(J),SVHGMX(J),SVGHMX(J)      CCN06880
760   WRITE (6,990) OMEGA(J),SVHGX(J),SVGHX(J),SVHGMX(J),SVGHMX(J)      CCN06890
      IF (KONTRL.EQ.0.OR.KONTRL.GT.1) GO TO 770                         CCN06900
      KONTRL=0                                                          CCN06910
      GO TO 130                                                         CCN06920
C*********************************************************************  CCN06930
C                                                                       CCN06940
770   STOP                                                              CCN06950
C ***************************************************************     CCN06960
C       FORMAT STATEMENTS                                               CCN06970
C ***************************************************************     CCN06980
780   FORMAT (3F10.0)                                                   CCN06990
790   FORMAT (3F10.0,I5)                                                CCN07000
800   FORMAT (2I2)                                                      CCN07010
810   FORMAT (8I2)                                                      CCN07020
820   FORMAT (/,12HOBS MAX EIG=,F10.5,8HMIN EIG=,F10.5,8HEIG RAT=,F10.5,CCN07030
     1/)                                                                CCN07040
830   FORMAT (/,10H WEIGHT 1=,F10.5,9HWEIGHT 2=,F10.5,9HWEIGHT 3=,F10.5,CCN07050
     1/)                                                                CCN07060
840   FORMAT (/,9H SVMINI= ,F10.5,8H SVMINO=,F10.5,5H  RJ=,F10.5,6H  IDGCCN07070
     1=,I5,/)                                                           CCN07080
850   FORMAT (2I5)                                                      CCN07090
860   FORMAT (/,26H     THE INITIAL K MATRIX  ,/)                       CCN07100
870   FORMAT (/,26H *** THE INITIAL DATA IS ***,/)                      CCN07110
880   FORMAT (/,41H *** THE OPTIMIZER COMPUTED OUTPUT IS ***,/)         CCN07120
890   FORMAT (/,27H  THE OBSERVER EIGENVALUES ,/)                       CCN07130
900   FORMAT (/,27H THE OBSERVER EIGENVECTORS ,/)                       CCN07140
910   FORMAT (4E12.5)                                                   CCN07150
920   FORMAT (/,20H  THE K GAIN MATRIX ,/)                              CCN07160
930   FORMAT (//,34H  **** THE OBJECTIVE FUNCTION IS =,F10.5,6H  ****,/)CCN07170
940   FORMAT (I5)                                                       CCN07180
950   FORMAT (5E12.5)                                                   CCN07190
960   FORMAT (/,2X,7H  FREQ ,5X,6HSVADI2,4X,6HSVADO2,4X,5HSVMI2,4X,5HSVMCCN07200
```

```
     102,/)                                                                    CONO7210
970   FORMAT (/,2X,8H   FREQ ,5X,5HSVHGX,4X,5HSVGFX,4X,6HSVHGMX,4X,6HSVG     CCNO7220
     1HMX,/)                                                                   CONO7230
980   FORMAT (5E12.5)                                                         CCNO7240
990   FORMAT (5F10.5)                                                         CCNO7250
1000  FORMAT (3F10.0)                                                         CCNO7260
1010  FORMAT (/,6H WMAX=,F10.5,1X,10HINIT FREQ=,F10.5,1X,10HFREQ STEP=,F      CCNO7270
     110.5,//)                                                                CCNO7280
1020  FORMAT (//,10H FREQUENCY,5X,13HMIN ADD IN SV,4X,13HMIN ADD CT SV,3      CCNO7290
     1X,13HMIN MUL IN SV,2X,18HMIN MUL  CLT SV   )                            CCNO7300
1030  FORMAT (5(F10.5,5X))                                                    CCNO7310
1040  FORMAT (I5,I5)                                                          CCNO7320
1050  FORMAT (8I5)                                                            CCNO7330
1060  FORMAT (9I2)                                                            CCNO7340
1070  FORMAT (8I2)                                                            CCNO7350
1080  FORMAT (2F10.0)                                                         CCNO7360
1090  FORMAT (1H1,///,12H<<<<< RUN #  ,I5,6H >>>>>,///)                       CCNO7370
1100  FORMAT (27H *** THE A PLANT MATRIX ***,///)                             CCNO7380
1110  FORMAT (//,35H *** THE B CONTROL INPUT MATRIX ***,///)                  CCNO7390
1120  FORMAT (//,33H *** THE C OBSERVATION MATRIX ***,///)                    CCNO7400
1130  FORMAT (//,38H *** THE          F FEEDBACK MATRIX ***,///)              CCNO7410
1140  FORMAT (//,44H *** THE ORDERED COMPLEX EIGENVALUES (INPUT),//)          CCNO7420
1150  FORMAT (2F10.0)                                                         CCNO7430
1160  FORMAT (//,37H *** THE ORDERED COMPUTED EIGENVALUES,//)                 CCNO7440
1170  FORMAT (//,29H *** THE ORDERED EIGENVECTORS,//)                         CCNO7450
1180  FORMAT (//,36H *** THE        F FEEDBACK MATRIX ***,///)                CCNO7460
      END                                                                     CCNO7470
CTHIS SUBROUTINE COMPUTES THE CONTROLLER TRANSFER FUNCTION                    CCNO7480
CFOR THE OBSERVER ROBUSTNESS PROBLEM                                          CCNO7490
C                                                                             CCNO7500
C                                                                             CCNO7510
      SUBROUTINE CONTRP(F,AX,FX,FKX,FKC,BF,NRA,NCA,NRF,NCF,NRK,NCK,W)         CCNO7520
      COMPLEX*8 H(10,10),AX(10,10),FX(10,10),FKX(10,10),FKCX(10,10),          CCNO7530
     1BFX(10,10),SI(10,10),I(10,10),SIA(10,10),AALG(10,10),AALK(10,10)       CCNO7540
      REAL*4 F(100),FKC(10,10),BF(10,10)                                      CCNO7550
      COMPLEX S                                                               CCNO7560
      S=CMPLX(0.0,W)                                                          CCNO7570
C      CALL CWRITE(AX,NRA,NCA)                                                CCNO7580
C     CALL CWRITE(FX,NRF,NCF)                                                 CCNO7590
C     CALL CWRITE(FKX,NRK,NCK)                                                CCNO7600
C     CALL WRITE(BF,NRB,NCF)                                                  CCNO7610
      DO 20 J=1,NRA                                                           CCNO7620
      DO 10 K=1,NCA                                                           CCNO7630
   10 SI(J,K)=C.                                                              CCNO7640
   20 CONTINUE                                                                CCNO7650
      DO 12 J=1,NRA                                                           CCNO7660
      DO 11 K=1,NCA                                                           CCNO7670
   11 I(J,K)=0.0                                                              CCNO7680
```

```
   12 CONTINUE                                                          CCNO7690
      DO 30 J=1,NCA                                                     CONO7700
   30 I(J,J)=1.0                                                        CONO7710
      DO 31 J=1,NRA                                                     CONO7720
      DO 32 K=1,NCA                                                     CONO7730
   32 SI(J,J)=S*I(J,J)                                                  CCNO7740
   31 CONTINUE                                                          CCNO7750
C     CALL CHFITE(SI,NRA,NCA)                                           CONO7760
      DO 50 J=1,NRA                                                     CCNO7770
      DO 40 K=1,NCA                                                     CONO7780
   40 SIA(J,K)=SI(J,K)-AX(J,K)                                          CONO7790
   50 CONTINUE                                                          CCNO7800
C     CALL CHFITE(SIA,NRA,NCA)                                          CONO7810
      CALL CFLXCV(FKCX,FKC,NRA,NCA)                                     CONO7820
      CALL CFLXCV(BFX,BF,NRA,NCA)                                       CONO7830
      DO 60 J=1,NRA                                                     CONO7840
      DO 70 K=1,NCA                                                     CONO7850
   70 AALG(J,K)=SIA(J,K)+FKCX(J,K)+BFX(J,K)                             CONO7860
C     CALL CHFITE(AAUG,NRA,NCA)                                         CONO7870
   60 CONTINUE                                                          CCNO7880
      CALL LEGTIC(AAUG,NCA,10,I,NCA,10,0,WA,IER)                        CONO7890
      CALL CMATML(I,FKX,NRA,NCA,NCK,AAUK)                               CONO7900
      CALL CMATML(FX,AAUK,NRF,NCF,NCK,H)                                CCNO7910
C     CALL CHFITE(H,NRF,NCK)                                            CONO7920
      RETURN                                                            CCNO7930
      END                                                               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.

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