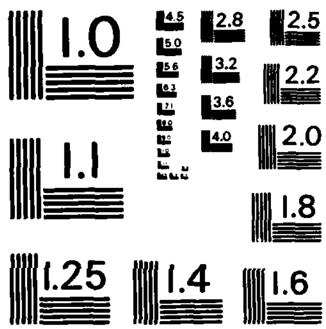
AD-A162 269 AUTOMATED POLE PLACEMENT ALGORITHM FOR MULTIVARIABLE OPTIMAL CONTROL SYNTHESIS(U) NAVAL POSTGRADUATE SCHOOL MONTEREY CA W K CHOW SEP 85 1/2

UNCLASSIFIED F/G 12/1 NL



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

(2)

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

AUTOMATED POLE PLACEMENT ALGORITHM

FOR MULTIVARIABLE OPTIMAL CONTROL SYNTHESIS

by

Chow, Wah Keh

September 1985

Thesis Advisor: D. J. Collins

Approved for public release; distribution unlimited.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Date Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO. AD-A162269	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Automated Pole Placement Algorithm for Multivariable Optimal Control Synthesis		5. TYPE OF REPORT & PERIOD COVERED Master's Thesis September 1985
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Chow, Wah Keh		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 1985
		13. NUMBER OF PAGES 170
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassifed
		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)

Linear Quadratic Control, Pole Placement, Pole Assignment, Multivariable System, Multi-input Multi-output Control, Multivariable System, Robustness, Weighting Matrix.

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

This work addresses the application of numerical optimization technique to the pole-placement problem in multivariable optimal control. An algorithm is developed to select a set of weighting matrix element such that the conventional transient response criteria are satisfied.
General properties of the optimal system in terms of stablity, robustness and relative weights between state and control

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

S N 0102- LF- 014- 6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

variables were explored by applying the method to the design of two multivariable systems. Results indicated that this method provides good insight to the problem for the designer and is therefore a useful tool in multivariable control synthesis.

Approved for public release; distribution is unlimited.

Automated Pole Placement Algorithm for Multivariable Optimal Control Synthesis

by

Chow, Wah Keh
B.A.(Hons), University of Oxford, 1978

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN AERONAUTICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
September 1985

Author: Chow, Wah Keh

Approved by: Daniel J. Collins, Thesis Advisor

Max F. Platzer, Chairman, Department of Aeronautics

John N. Dyer, Dean of Science and Engineering

ABSTRACT

This ~~work~~ thesis addresses the application of numerical optimization technique to the pole-placement problem in multivariable optimal control. An algorithm is developed to select a set of weighting matrix element such that the conventional transient response criteria are satisfied.

General properties of the optimal system in terms of stablity, robustness and relative weights between state and control variables were explored by applying the method to the design of two multivariable systems. Results indicated that this method provides good insight to the problem for the designer and is therefore a useful tool in multivariable control synthesis. Keywords: Linear Quadratic Control; control theory; Computer aided design.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGMENTS

I wish to gratefully acknowledge my thesis advisor, Prof. D. J. Collins, for suggesting the basis of this thesis work, and for his invaluable advice and guidance during the course of the thesis.

I would also like to express my appreciation to Prof. J. V. Sanders and R. D. Zucker for their support during my study at NPS, and to all others who have helped me directly or indirectly to successfully complete this thesis.

I. INTRODUCTION

A. BACKGROUND

For more than two decades, the need to solve control problems in aerospace application has been the primary driving force behind the modern control theory development. Problems in manoeuvring, guidance and tracking of aircrafts and space vehicles have motivated the development of various control design and synthesis methods. One of the methods, the so-called Linear Quadratic Control or LQ-control [Refs. 1,2] has been widely used and is treated extensively in the control literatures. Unlike most classical methods where the design are based on conventional time response criteria, the LQ theory treats the problem of designing controllers as that of minimizing a quadratic cost function of states and control inputs. The design problems become that of selecting suitable weighting matrices in the performance index. Two questions naturally arise from this method. First, how does one select the weighting matrices and second, once a set weighting matrices is selected, how does one know that it is a good design. There are generally two approaches to the first problem; the obvious one is to rely on physical arguments and a certain amount of trial and error [Refs. 3,4] Unfortunately, such formulation can be obtained in only a few cases. Reference 4 provides a few guidelines that can be employed. The second approach is to avoid the physical aspect of the performance index but instead try to relate the weighting matrices with some other performance specification. For example, Tyler and Tuteur [Ref. 5] expressed the characteristic polynomial as an explicit function of the weighting matrices for single input single output (SISO) system with diagonal weighting elements. Root-Locus type of procedure were used to show how the variation in weighting elements affect the eigenvalues of the closed-loop system. Similar relations were explored in [Refs. 6,7] in which more general expressions were obtained. Their uses were restricted to single input case due to difficulty in handling polynomial matrices. Solheim [Ref. 8] later developed a sequential design procedure based on diagonalized (decoupled) system. More recent results of eigenvalue placement in optimal control problem are presented in [Refs. 9,10,11,12].

The second issue of LQ design, i.e., whether a good design has been obtained once the performance index has been fixed, is related to the multivariable nature of system. Unlike the single input case where the closed-loop eigenvalues uniquely define the feedback gain and hence the weighting matrices, the MIMO structure provides additional dimension which allows further tradeoff for properties other than the closed-loop eigenvalue location. An example is the gain and phase margin, or in MIMO case, the so-called robustness criteria. Robust-control has been the subject of extensive researches in recent years and results relating to optimal control can be found in [Refs. 13,14,15].

It becomes evident from the above discussion that LQ control design and synthesis are not just a matter of specifying performance index. Optimal in the sense of satisfying performance index and perhaps pole location does not necessary means that a good design has been obtained. Other criteria like disturbance rejection, robustness and sensitivity need to be considered and incorporated in the design procedure. This is the motivation behind our present research which in turn leads to the development of the synthesis package. An overview of the thesis is described in the next section.

B. OVERVIEW

In this section an overview of the thesis is given. A background of multivariable optimal control theory in terms of its structure, frequency domain characteristic and asymptotic properties is first discussed together with derivation of some useful relationships in Chapter 2. General robustness concepts and its application in Linear Quadratic (LQ) Control is presented in Chapter 3. In Chapter 4, a computer aided design package for pole-placement synthesis based on numerical optimization technique is presented. This package provides a useful computational tool to support the material in the remaining chapter. A step-by-step pole-placement synthesis procedure is also presented to illustrate how the package can be used to design optimal control system that meet time response criteria and other properties. The use of the pole-placement synthesis package in two actual design problems is demonstrated in Chapter 5. Results in terms of frequency and time response properties ,robustness (singular value decomposition) are compared to those obtained by other design methods. Program listings and an example of a design session are included in the Appendix.

II. MULTIVARIABLE LINEAR QUADRATIC CONTROL

In this Chapter a brief review of Multivariable Linear Quadratic Control theory is presented. Structure of LQ system is first given, general stability properties are then discussed. In section two the general pole assignment problem is formulated for the MIMO state feedback system. Interpretation of the state and control input weighting matrices and their effect on the closed-loop time response behaviors and the asymptotic properties are presented.

A. LINEAR QUADRATIC SYSTEM

Consider the following linear time-invariant state-space system given by the equations:

$$\dot{x}(t) = Ax(t) + Bu(t) \qquad \text{(eqn 2.1)}$$

$$y(t) = Cx(t) \qquad \text{(eqn 2.2)}$$

where $x(t)$ is the n-dimensional state vector, $u(t)$ is the m-dimensional control vector and $y(t)$ is the p-dimensional output vector. A ,B and C are real constant matrices of dimension nxn, nxm and pxn respectively. Assuming that they form a controllable pair(A,B) and an observable pair(A,C), the optimal feedback control law is obtained by minimizing the following quadratic performance index.

$$J = \int_0^\infty (x^T Q x + \rho u^T R u)dt \qquad \text{(eqn 2.3)}$$

where R is positive definite (R>0) for bounded input and ρ is a scalar. Q is a semi-positive definite matrix ($Q \geq 0$).

When both Q and R are diagonal matrices, ρ defines the relative weight between the state and control weighting matrices in the performance index. The quadratic form x^TQx and u^TRu provide a weighted measure of the magnitude of the states and control vector respectively. The steady state control law that minimizes J is given by,

$$u(t) = -Fx(t) \qquad \text{(eqn 2.4)}$$

where F is the feedback gain matrix which is given by

$$F = -R^{-1}B^TP \qquad \text{(eqn 2.5)}$$

The positive definite matrix P is given by the solution to the steady state Riccati equation.

$$PA + A^TP + Q - PBR^{-1}B^TP = 0 \qquad \text{(eqn 2.6)}$$

Equation 2.5 and 2.6 are well-known results in optimal control theory that yield the optimal closed-loop system

$$\dot{x}(t) = [A + BF]x(t) \qquad \text{(eqn 2.7)}$$

whose closed-loop poles are given by the eigenvalues of the matrix [A+BF]. The LQ system given by equations 2.1 through 2.6 is closed-loop stable and can be represented by the feedback configuration in both the time and frequency domain as shown in Figure 2.1 .

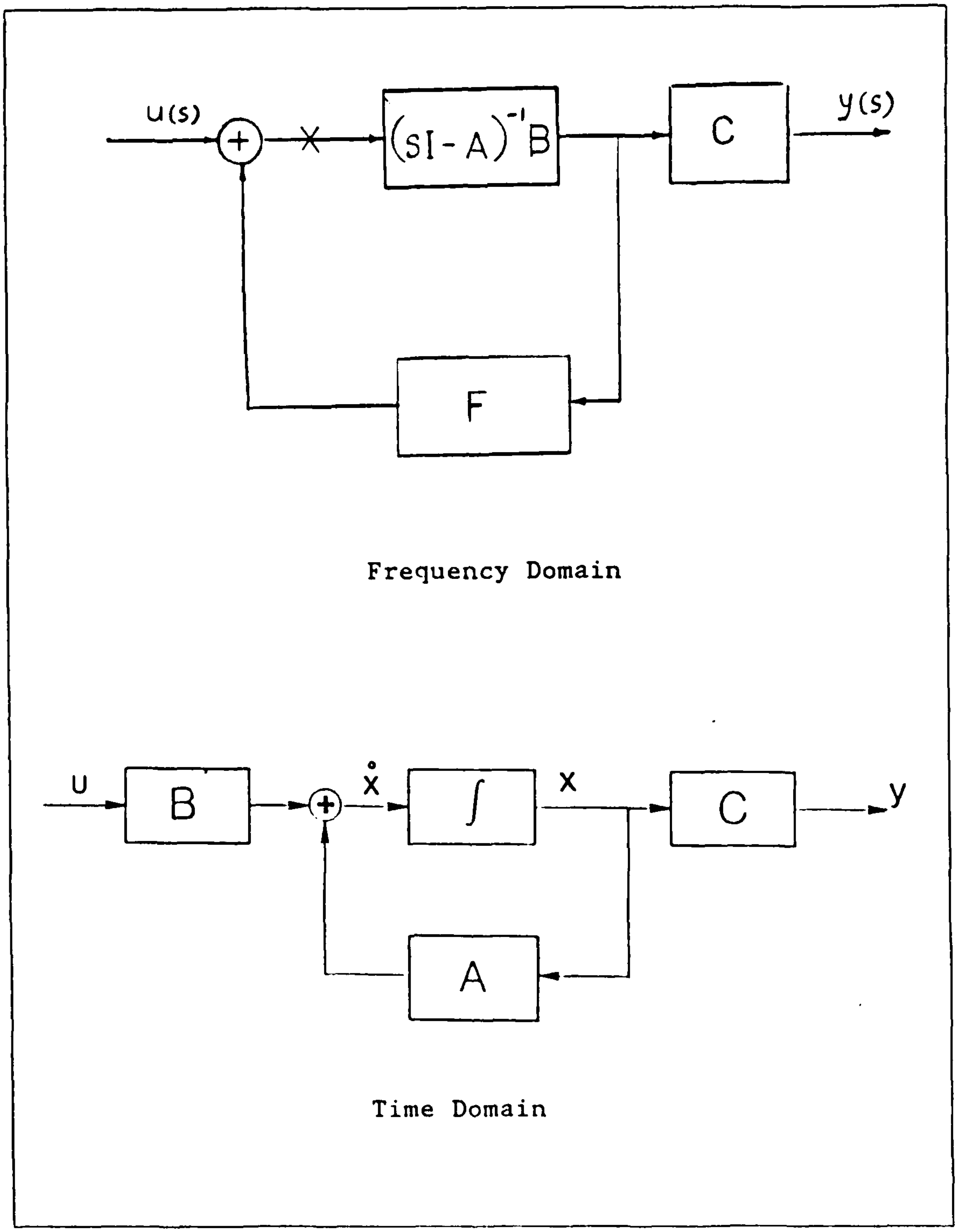


Figure 2.1 State Feedback System-Time and Frequency Domain.

In Figure 2.1, if the loop is broken at the input (X) as shown, the loop transfer function is given by

$$G(s) = F(sI - A)^{-1}B \qquad \text{(eqn 2.8)}$$

The matrix $[I + G(s)]$ is called the Return Difference Matrix and will be shown to have some important feedback properties in the following section.

B. POLE ASSIGNMENT PROBLEM

In the most general form, the state feedback pole assignment problem in control system design can be formulated precisely as follows:

"Given real matrices (A,B) of order (nxn, nxm) respectively and a set of n complex numbers ($\lambda_1, \lambda_2, \ldots \lambda_n$) closed under complex conjugation. Find a real mxn matrix F such that the eigenvalues of [A+BF] are λ_i, i = 1, 2,3... n."

In general, the closed-loop eigenvalues of (A+BF) can be arbitrary located in the complex plane, with the only restriction that complex characteristic eigenvalues must occur in complex conjugate pair. In other words, if the matrices (A,B) are a completely controllable pair, the stability of the system can always be improved by state feedback. If the (A,B) pair is not completely controllable, then the system is required to be 'stabilizeable' meaning that in its controlability canonical form given by equation 2.9 below, the matrix A_{22} is asymptotically stable or any unstable subspace of equation 2.9 is also in the controllable subspace.

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix}$$

(eqn 2.9)

The solution to the pole assignment problem in the single input (m=1) case, when it exits, can be shown to be unique. In the multiple input case (1<m<n), the solution of the so-called inverse eigenvalue problem is, in general, undetermined with many degrees of freedom. Additional conditions must be supplied in order to eliminate the extra degrees of freedom. This has been an area of active research in recent years and a number of approaches have been developed to relate the extra degrees of freedom with properties such as system eigenvectors, transmission zeros and robustness [Refs. 16,17]. In this work, it is shown that Linear Quadratic formulation incorporating equation 2.3 will partly remove the uncertainty that exists in the multiple inputs case. It will be shown that useful properties like robustness are guaranteed. It is also shown that the LQ type of pole placement formulation, when combined with eigenvectors assignment type of formulation, will produce some very useful design and synthesis procedures. In the present work, however, only the LQ eigenvalue placement problem is addressed; the problem of combined eigenvalue and eigenvector assignment is briefly described in Chapter 5.

C. WEIGHTING MATRICES AND SYSTEM PERFORMANCES

This section briefly reviews the effect of weighting matrices on system performance. The physical aspects of the weighting matrix for both single and multiple input systems are presented first together with some discussion on the asymptotic behavior of the LQ system.

It was shown in the last section that under complete controllable conditions, the time-invariant linear system can be stabilized by a linear feedback control law. For the regulator-type problem where the aim is to bring the system from an arbitrary initial state to the zero state, the closed loop poles can be chosen far to the left on the complex plane. Convergence to the zero state is fast but the large input required may not be practical. This naturally leads to an optimization problem where the trade off is between speed of convergence to zero and the magnitude of the input. This is reflected in the two quadratic terms in the performance index. The quantity x^TQx in the first term of the performance index is a measure of the extent in which the state at time t deviates from the zero state. The matrix Q determines how much weight is attached to each of the component of the state. The integral $\int(x^TQx)dt$ is a criteria for the cumulative deviation of x(t) from the zero state during the interval.

The problem of large control input is resolved by incooperating the second quadratic term, $\int(u^TRu)dt$. Larger value in the element of the control weighting matrix R will result in smaller input. It will be shown later that one can manipulate R to achieve some secondary design objectives. The remaining parameter ρ, that need to be specified, accounts for the relative weighting between the state and the control inputs. Selecting optimal value for ρ depends on the particular problem and the design requirement. As ρ

decreases, the integrated square regulating error decreases but the integrated square input increases. Very often, the optimal control problem is solved for many different values of ρ. A graph similar to that shown in figure 2.2 is obtained where the integrated square regulating error is plotted versus the integrated square input. An appropriate value of ρ is then selected to give sufficiently small regulator error without excessively large input.

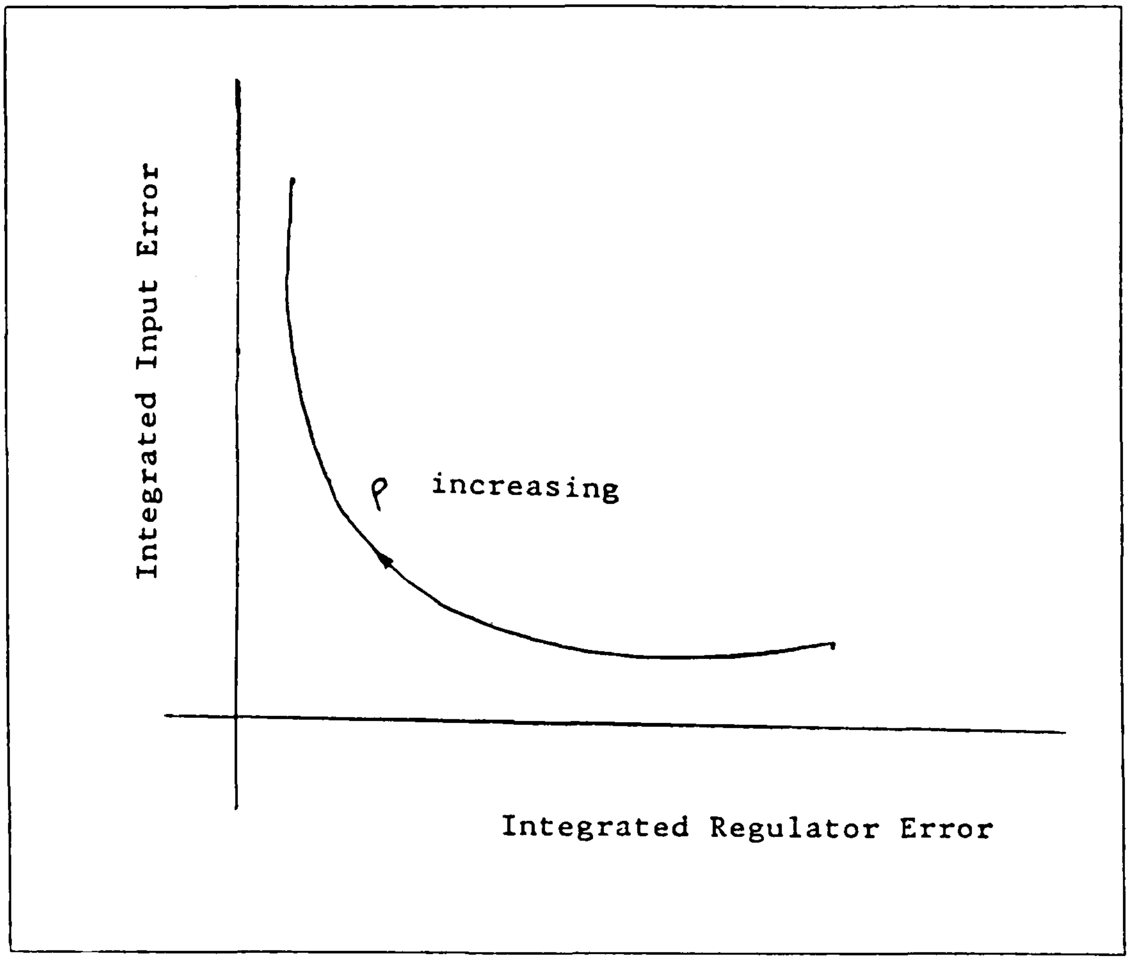


Figure 2.2 Selecting Relative Weighting Parameter ρ.

The special case where ρ approaches zero ,the so-called asymptotic properties, has been shown to provide good insight for LQ control system design. Some of the results from [Refs. 4,18] are summarized below.

As ρ decreases to zero for the system given by equations 2.1 through 2.6, some (say q) of the closed-loop poles go to infinity while other (n-q) stay finite. Those remaining finite approach the left half plane zero of $\det[B^T(-sI-A^T)^{-1}Q\,(sI-A)^{-1}B]$. If m is the dimension of the input vector, then at least m closed-loop poles approaches infinity. All closed-loop poles that go to infinity do so by grouping into several Butterworth patterns of different order and radii. For ρ approaching infinity, the closed-loop poles approach the mirror image of the plant open-loop poles. To illustrate the asymptotic concept, two examples from [Ref. 4] are given below.

An example of the asymptotic properties of a single input system is shown in Figure 2.3 . The closed-loop poles of a position control system using LQ type feedback are plotted as a function of ρ . The system has two open-loop poles at -4.6 and 0.0 . As ρ decreases, the closed-loop poles go to infinity along two straight lines that make an angle of $\pi/4$ with the negative real axis. As ρ approaches infinity, both closed-loop poles first meet on the negative real axis and then approach the open-loop poles at (-4.6, 0).

Figure 2.4 shows the asymptotic loci of the closed-loop poles for a multiple input system where n=4 and m=2. There are four open-loop poles at (-0.006123, ±j0.09353) and (-1.250, ±j1.394) and both the state and control weighting matrices are taken to be of a diagonal form. As ρ approaches to zero, one closed-loop pole stay finite at -1.002. The remaining three go to infinity, two of which assume a second order Butterworth pattern and the last one approaches on the negative real axis. When ρ approaches infinity, all closed-loop poles approach the open-loop poles.

Many researchers have explored this asymptotic properties. Stein [Refs. 9,10] developed a procedure to select the control weighting matrices for a desired closed-loop asymptotic eigenstructure. It will be shown in the subsequent Chapters that the asymptotic properties provide useful guideline for the design procedures to be described.

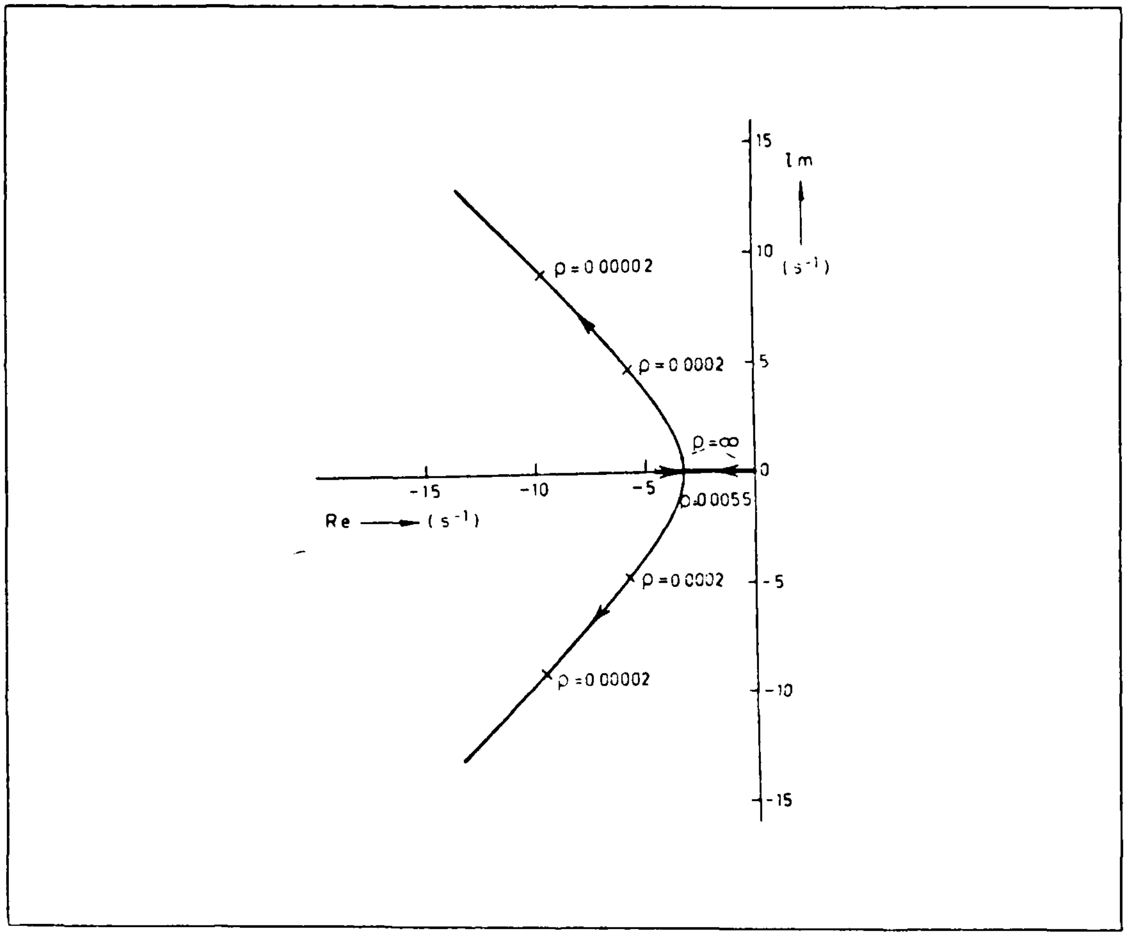


Figure 2.3 Single Input Asymptotic Root Loci.

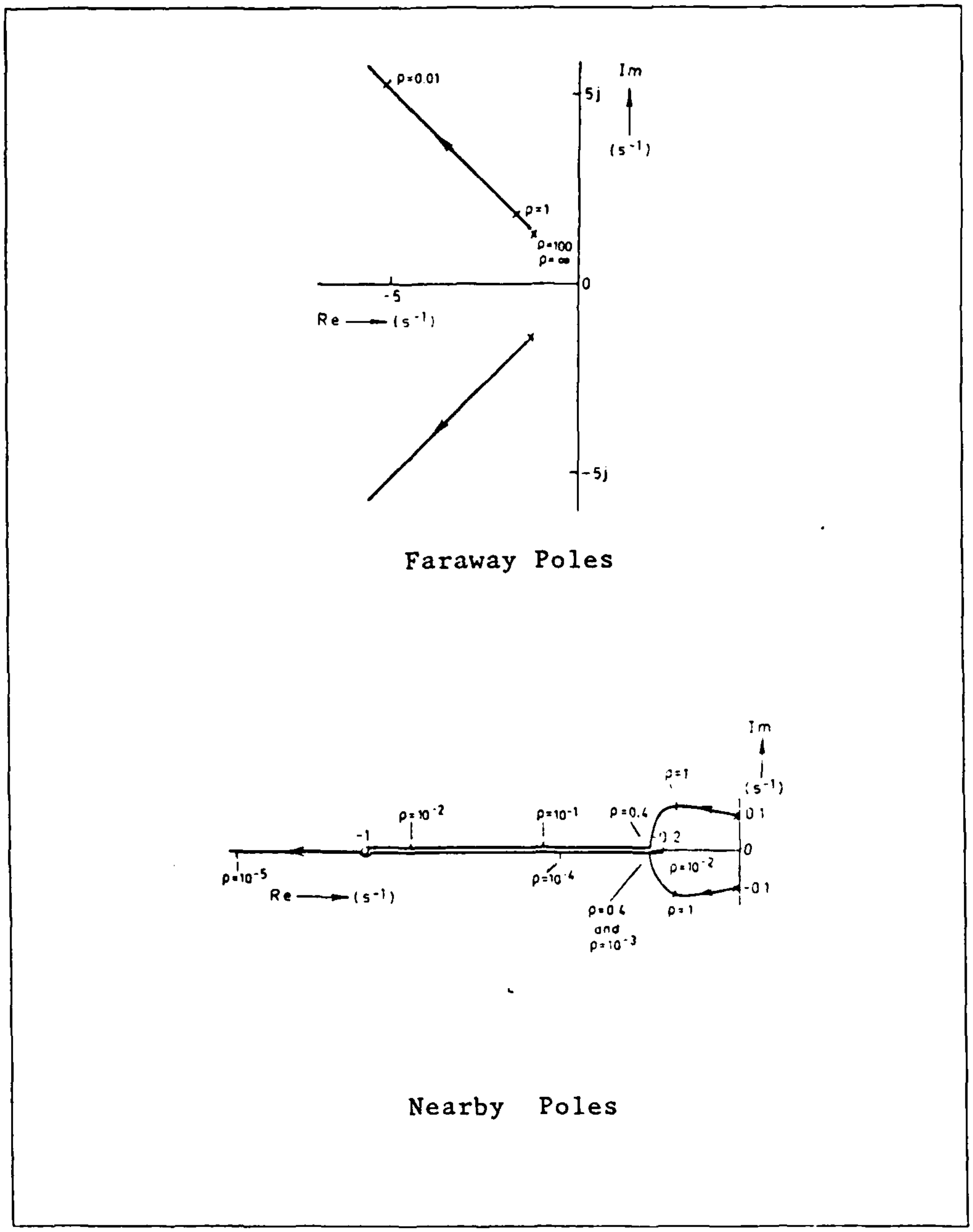


Figure 2.4 Multiple Input Asymptotic Root Loci.

III. ROBUSTNESS THEORY AND LINEAR QUADRATIC CONTROL

In the last Chapter, the effect of weighting matrices on the closed-loop pole of the LQ system was discussed. This Chapter addresses yet another important feedback property that control system designers are concerned with: Robustness. Robustness theory was developed when it was realized that the classical single loop Nyquist test was not adequate to guarantee stability when the multivariable open loop plant deviates from its model due to a variety of reasons [Ref. 15]. In the following section , the concept of robust design for both SISO and MIMO general feedback system is reviewed. The singular value analysis is discussed in term of multiplicative type of disturbances. Finally, robustness for linear quadratic state feedback are presented in terms of the effect of weighting matrices on the singular value curve.

A. ROBUSTNESS CONCEPTS

The concept of robustness for SISO system can best be described in terms of the definition of phase and gain margin. A system characterized by good gain and phase margin implies that changes in the plant model parameters and changes in the loop gain and/or phase may be accommodated without loss of stability. The gain and phase margins of a SISO system are defined with reference to the perturbed system in Figure 3.1

Assuming that the unperturbed system $(l(jw)=1)$ is stable, the positive (or negative) phase margin is the value of Φ greater (or less) than zero at which the perturbed system with $l(jw) = k^{*}\exp(j\Phi)$ becomes unstable. The upward

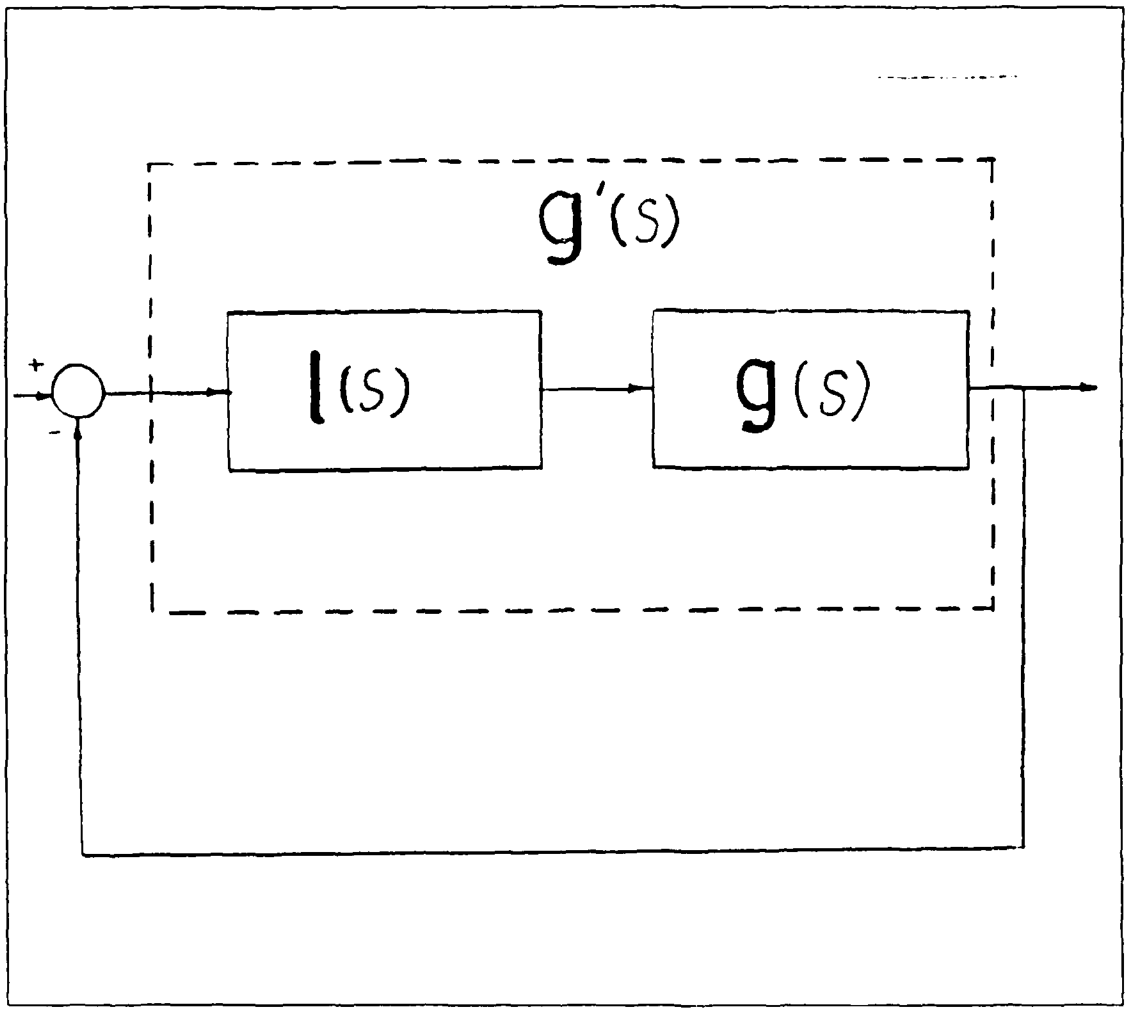


Figure 3.1 The Perturbed SISO System.

(or downward) gain margin is the smallest (or greatest) value of $l(jw)=k=\text{constant}$, for which the system become unstable. With reference to the classical Nyquist plot in Figure 3.2 , gain and phase margin are defined as,

$$GM\uparrow \text{(upward)} = 1/k_1 \qquad GM\downarrow \text{(downward)} = 1/k_2$$

$$PM^{+} = \alpha_1 \qquad PM^{-} = \alpha_2$$

A set of minimum guaranteed GM and PM may be obtained by defining $\alpha_o = \min[1 + g(jw)]$ as shown in Figure 3.3 where,

$$GM = 1/(1 \pm \alpha_o)$$

and

$$PM = \pm\cos^{-1}(1 - \alpha_o^2/2)$$

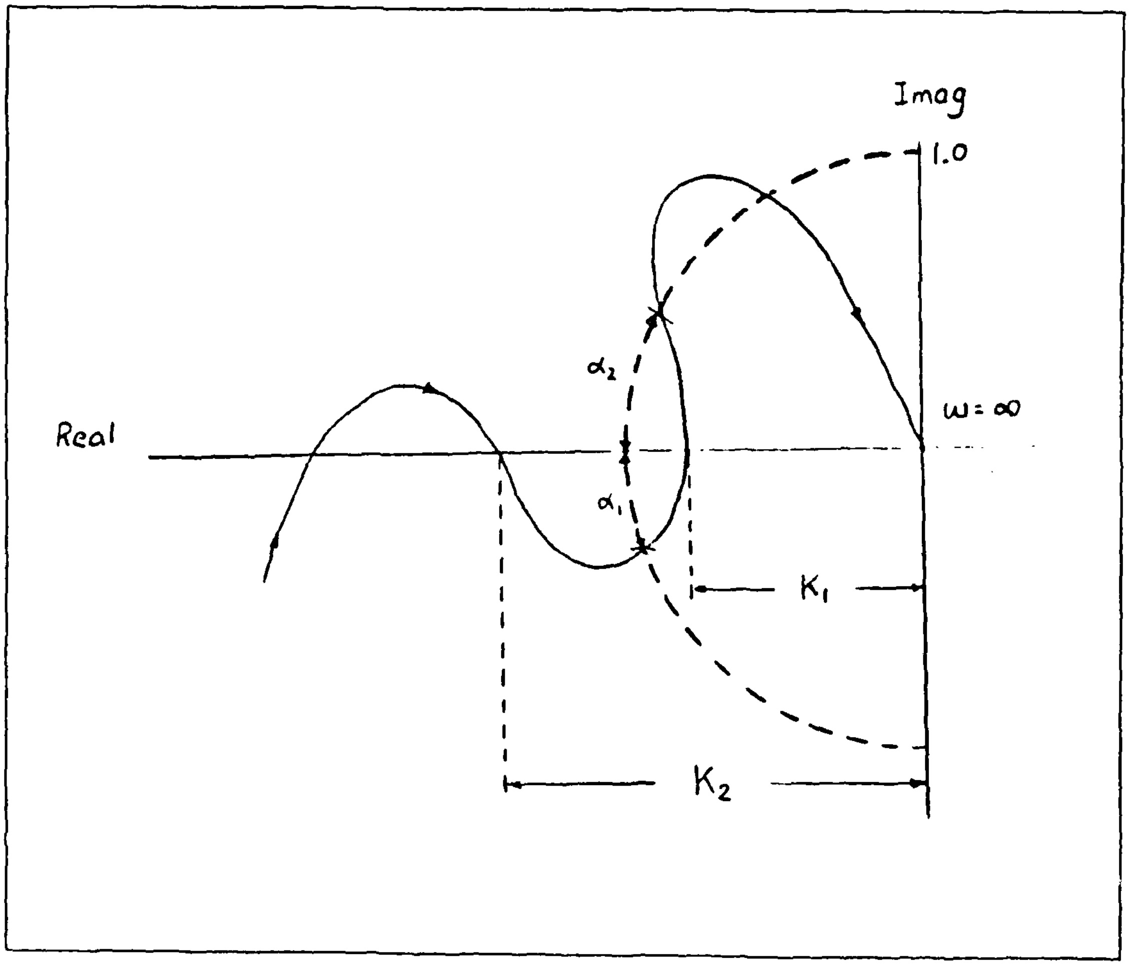


Figure 3.2 The Nyquist Plot - Gain and Phase Margin.

Note that in the above definition of GM and PM, either Φ_i 'or' K_i is allowed to change in the loop. The allowed changes are therefore very restrictive. A more useful definition of the gain and phase margin for MIMO system that accounts for simultaneous changes in both Φ_i and/or K_i (the so-called universal gain and phase margin) has been derived in [Ref. 19]. From Figure 3.3 it can be seen that α_0 may provide a rather conservative estimate of the actual gain and phase margin.

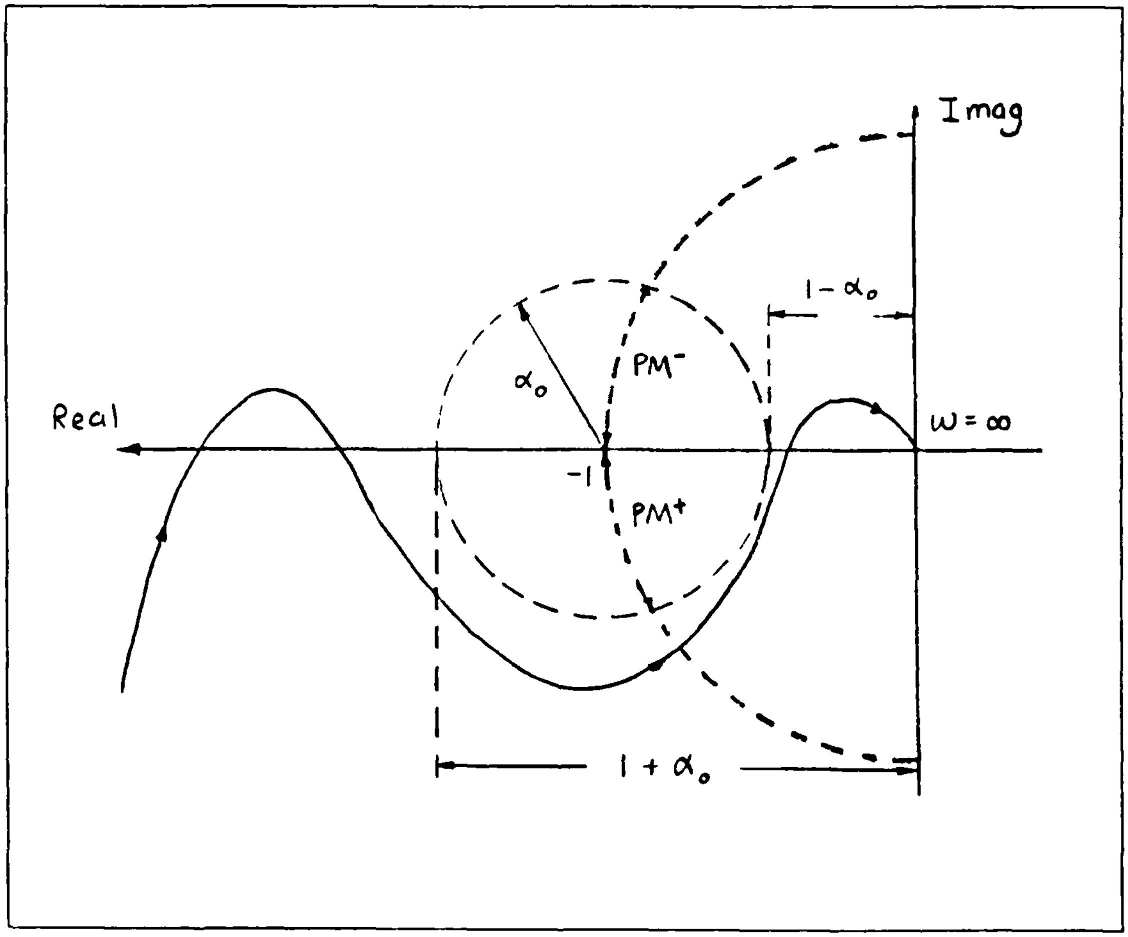


Figure 3.3 The Minimum Guaranteed GM and PM.

In MIMO system, gain and phase margin characterize the ability of the system to tolerate gain and/or phase changes within all loops simultaneously. Figure 3.4 shows a perturbed MIMO system with the assumption that $L(jw) = Dia(l_1(jw), l_2(jw)\ldots)$. As in the SISO case, the system will remain stable as longer as $l_i(jw)$ satisfy $GM\downarrow < k_i < GM\uparrow$ (assuming that Φ_is are zero). For the case when the magnitude of $l(jw)$ are constant, the system will remain stable for all Φ_is that satisfy $PM^- < \Phi_i < PM^+$.

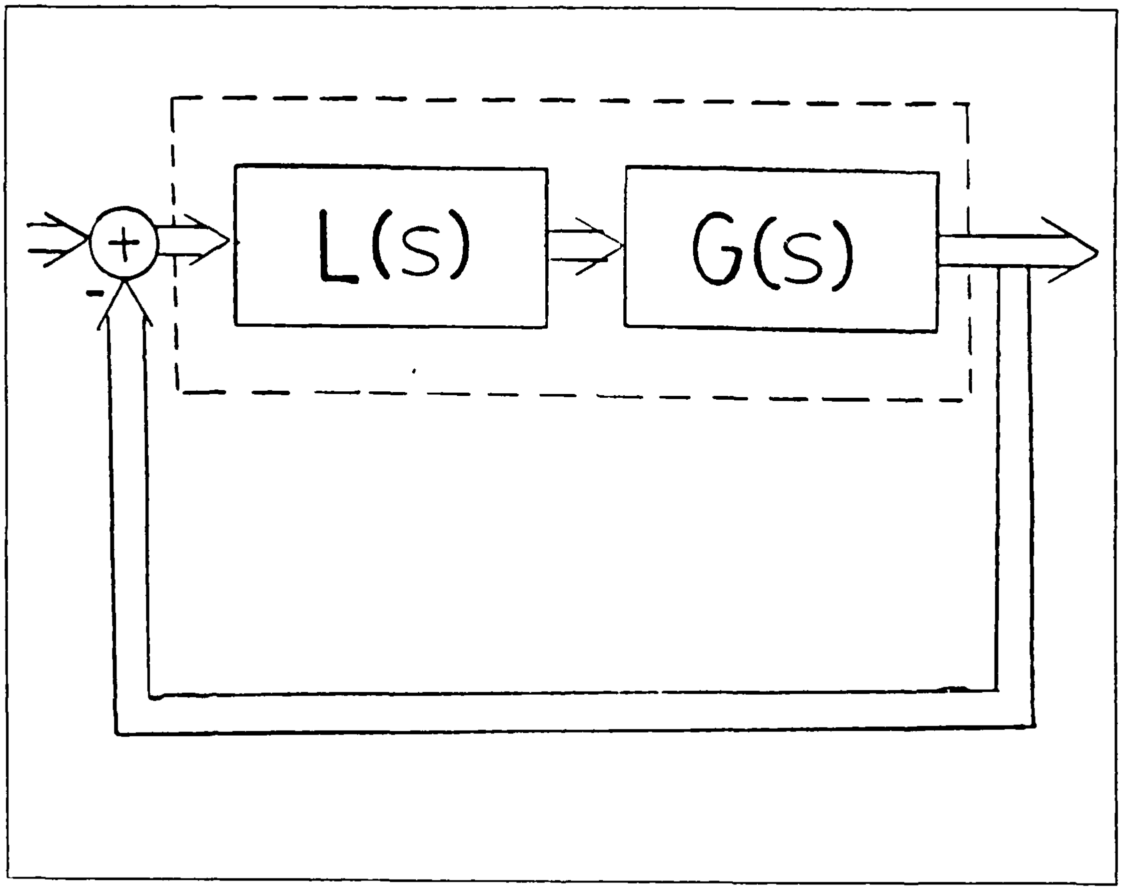


Figure 3.4 The Perturbed MIMO System.

The set of minimum guaranteed multivariable gain and phase margins are determined by the minimum singular value ($\underline{\sigma}$) of the return difference matrix $[I + G(s)]$, where $G(s)$ is the open loop transfer matrix and consists of the plant and its controller. Note the similarity between the SISO and MIMO cases,

α_o : nearness of $(1 + g(s))$ to the origin.

$\underline{\sigma}$: nearness of matrix $[I + G(s)]$ to singularity.

Two important results that are related to multivariable phase and gain margin developed in [Ref. 15] are next presented as a theorem.

THEOREM: The multiplicative perturbed system (Figure 3.4) is stable if either of the following conditions hold:

1. $\underline{\sigma}\,[I + G(s)] \; > \; \bar{\sigma}\,[L^{-1}(s) - I]$ (eqn 3.1)

2. $\underline{\sigma}\,[I + G^{-1}(s)] \; > \; \bar{\sigma}\,[L\,(s) - I]$ (eqn 3.2)

where $\bar{\sigma}$, $\underline{\sigma}$ denote the maximum and minimum singular values of $[I+G(s)]$ respectively.

It will be shown that condition 1 can be related to the optimality condition of LQ system and hence provides a useful relationship between the weighting matrices and robustness.

B. ROBUSTNESS IN LQ SYSTEM

Robustness properties pertaining to LQ system are closely related to the frequency domain optimality condition. For the SISO case, Kalman [Ref. 1] showed that the return difference transfer function satisfies the inequality

$$(1 + g(jw)) \geq 1 \quad (\alpha_o = 1) \qquad \text{(eqn 3.3)}$$

Inspection of the Nyquist diagram in Figure 3.5 clearly indicates that a SISO LQ state feedback has a guaranteed infinite upward gain margin , 0.5 downward gain margin and a minimum phase margin of ± 60 deg.

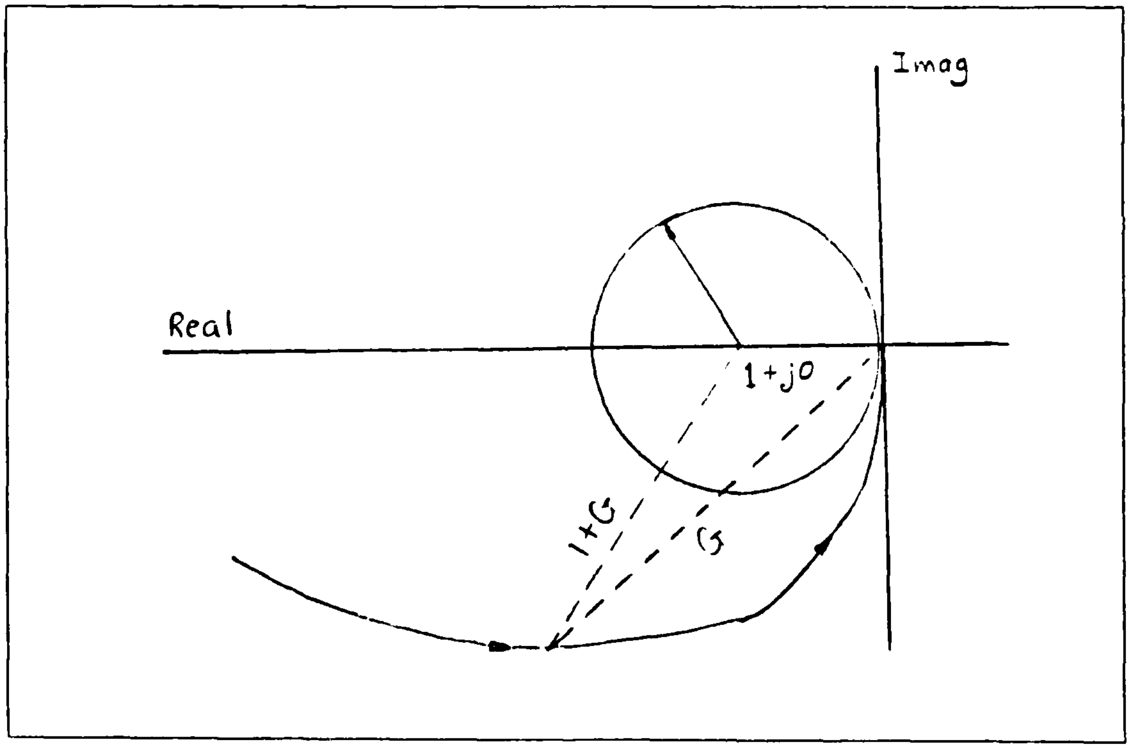


Figure 3.5 SISO LQ Nyquist.

In a similar manner, multivariable stability margin can be derived from the multivariable version of condition 1 in equation 3.1 ;

$$[I + G(s)]^H R \; [I + G(s)] > R \qquad \text{(eqn 3.4)}$$

where

$$G(s) = R^{-1} B^T P[sI - A]^{-1} B$$

and $P \geq 0$ satisfies the steady state Riccati equation.

It can be shown that condition 1 in equation 3.1 above can be written as

$$\underline{\sigma}\,[I + R^{1/2} G(jw)\, R^{-1/2}] \geq 1 \qquad \text{(eqn 3.5)}$$

which can be reduced to

$$\underline{\sigma}[I + G(jw)] > 1 \qquad \text{(eqn 3.6)}$$

where R is diagonal (i.e. $R = \rho I$ for some positive scalar ρ). Like the SISO case, the multivariable LQ regulator with loop transfer matrix G(s) that satisfies equation 3.5 and 3.6 has, in all the feedback loops, a guaranteed minimum gain and phase margin given by,

$$GM = 1/2, \text{ and } PM = \pm 60 \text{ deg}$$

The case when R is not diagonal is interesting as condition 3.6 no longer apply and additional trade-off can be obtained by including off diagonal terms in the control input weighting matrix. This is especially useful when the nature and structure of the disturbances are known. Design involving the off diagonal R matrix will be shown in Appendix A.

IV. DESIGN PROCEDURE

A. GENERAL

The background material described in Chapter 2 and 3 are the basis for developing a computer aided design package and the corresponding design procedure. The design philosophy presented in this Chapter is most useful when a designer is able to characterize the desired system in terms of closed-loop eigenvalues and time response. Some initial physical insight of the weighting matrices ,their asymptotic properties and the nature of the perturbation will be useful in the design process. This Chapter begins with a discussion of the various approaches to the LQ pole placement problem. The selected approach and the corresponding computer aided design package is presented. The design philosophy and design procedure for both the reduced order and full order model are then given.

B. APPROACHES

All the pole placement algorithms for LQ system that have been developed so far require an expression that relate the characteristic equation of the optimal system with the elements of the weighting matrices. Two such formulations for the stabilizeable and detectable time-invariant linear system (equation 2.1) and the quadratic criteria (equation 2.3) are given by

$$\phi_c(s)\phi_c(-s) = \phi(s)\phi(-s)\det[I + R^{-1} H^T(-s)\, Q\, H(s)] \quad \text{(eqn 4.1)}$$

and

$$\phi_c(s)\phi_c(-s) = \phi(s)\phi(-s)\det[I + 1/\rho R^{-\frac{1}{2}} H^T(-s) Q H(s) R^{-\frac{1}{2}}] \quad \text{(eqn 4.2)}$$

where $\phi_c(s) = \det[sI - A + BF]$ and $\phi(s) = \det[sI - A]$ are the closed-loop and open-loop characteristic polynomials respectively. $H(s) = C(sI-A)^{-1}B$ is the open loop transfer matrix of the system.

Both formulations have been used in root-locus type of procedure to investigate how the closed-loop poles move as weighting matrices changes [Refs. 5,6]. For MIMO case, there has been little success due to problems involving polynomial matrices. In this work, a different approach is adopted. Equation 4.1 or 4.2 is formulated as a numerical optimization problem in which the objective function is made equal to the determinant part of equation 4.1 or 4.2

$$Obj = \det[I + R^{-1} H^T(s)\, Q\, H(s)] \quad \text{(eqn 4.3)}$$

or

$$Obj = \det[I + 1/\rho\, R^{-\frac{1}{2}} H^T(-s)\, Q\, H(s) R^{-\frac{1}{2}}] \quad \text{(eqn 4.4)}$$

For a given desired closed loop pole $s = s_d$, equation 4.3 or 4.4 becomes

$$\phi_c(s_d)\,\phi_c(-s_d) = 0 \quad \text{(eqn 4.5)}$$

Providing that $\phi(s_d)\,\phi(-s_d)$ is not equal to zero, the objective function must equal to zero if the particular Q, R set is to correspond to the desired closed-loop poles. Convergence to zero for a given set of Q and R is therefor automatically guaranteed.

The pole placement problem can therefore be solved as an unconstrained multivariable optimization in which the elements of Q and R are varied to make the objective function in equation 4.3 and 4.4 approach zero. This was done during the early phase of the work. It was later discovered that more insight to the problem can be obtained by first transforming the problem to an appropriate coordinate system and then to perform pole placement one at a time . This is of advantage as a system designer is often satisfied with several open-loop poles in a large system. Reassigning poles in the the reduced-order model will reduce computer time and memory requirement.

The optimization routine selected for this work is the so-called SUMT method (Sequential Unconstrainted Minimization Techniques) obtained from the ADS package in [Refs. 20,21]. In this method, the objective function (eg. equation 4.3 or 4.4) and any constraint equation are formulated into an augmented objective function in which the problem is solved as an unconstraint optimization task.

C. POLE PLACEMENT ALGORITHMS

For ease of implementation and better insight, only the problem of determining the state weighting matrix Q (given R) that gives a set of closed-loop eigenvalues is considered. It must be emphasised that the algorithm can also be formulated to determine R (for a given Q) or to vary Q and R at the same time. In most cases, the present formulation is adequate as designers usually have some knowledge about the control weighting matrix. System matrix that has real and distinct eigenvalues is presented first, follows by cases where A has complex eigenvalues and repeated eigenvalues.

1. System with Real and Distinct Eigenvalues

The original system given by equation 2.1 and 2.2 is first transformed into a diagonal form using the transformation given by;

$$x(t) = Mz(t) \qquad \text{(eqn 4.6)}$$

where M is an eigenvector matrix corresponding to the system matrix A. and z(t) is the new state vector. The transformed system in the new coordinate is given by,

$$\dot{z}(t) = \Lambda z(t) + M^{-1}Bu(t) \qquad \text{(eqn 4.7)}$$

where Λ is a diagonal matrix dia[λ_1, λ_2 , λ_3 , . . .]

The performance index (equation 2.3), when expressed in terms of the new state vector z(t) becomes,

$$\begin{aligned} J &= \int_0^{\infty} (x^T Q x + u^T R u) \, dt \\ &= \int_0^{\infty} (z^T M^T QM z + u^T Ru) \, dt \\ &= \int_0^{\infty} (z^T \hat{Q} z + u^T R u) \, dt \qquad \text{(eqn 4.8)} \end{aligned}$$

where $\hat{Q} = M^T QM$.

It can be shown that to move an open-loop pole to its new location given by $s_i = s_d$, only $\hat{Q}_i$ is required and other $\hat{Q}$s' have no effect on the pole assignment. As an example, to move the open-loop pole at $s = \lambda_2$ for Λ = dia[λ_1 , λ_2 , λ_3 , . . .] to its new location $s = \bar{\lambda}_2$, only $\hat{Q}$= dia[0, $\hat{q}_2$, 0, . . .] is required.

$\hat{Q}$ can then be selected according to equation 4.3 or 4.4 using the optimization routine. As currently implemented in the program, there is no constraint equation formulation. Once the value of $\hat{Q}$ that satisfies the desired pole location is obtained, the system is transformed back to the original coordinate system via,

$$Q = M^{-T}\hat{Q}M^{-1} \qquad \text{(eqn 4.9)}$$

With Q known, and R given, the optimal feedback gain F can be obtained by solving the steady state Riccati equation as given in equation 2.5 and 2.6 . Since the pole placement is done in the decoupled coordinate system, only the eigenvalues that correspond to Q_i is reassigned; all other eigenvalues remain unchange. It can be also shown that the eigenvector corresponding to an eigenvalue is also unchange, this property will be shown to be useful in the reduced order formulation of the linear quadratic problem.

If desired, the problem here can also be formulated to move more than one eigenvalues in one run. This can be done by modifying the objective function to include more terms as follows;

$$Obj = \sum_{i=1}^{n} \det[I + R^{-1} H^{T}(-s)QH(s)R^{-1}] \qquad \text{(eqn 4.10)}$$

where n is the number of poles to be reassigned.

The augmented matrix A_{aug} = [A + BF] is then computed. If desired, the procedure may be repeated to move other open loop eigenvalue to its specified position using the new A_{aug} as the starting plant matrix. This will in turn result in another set of Q and F . The effective Q_e and F_e after n reassignments are given by

$$Q_e = Q_1 + Q_2 + \dots Q_n \qquad \text{(eqn 4.11)}$$

and

$$F_e = F_1 + F_2 + \dots F_n \qquad \text{(eqn 4.12)}$$

The above pole assignment procedure can also be applied to an optimal system where an initial starting Q and R are given. A good example is when the control system designer has some knowledge of the weighting matrix but would also like to meet a specific time response requirement.

2. System with Complex Eigenvalues

If the same similarity transformation mentioned in the last section is used, the transformation matrix will be complex. To be able to work with real matrix, an auxiliary transformation of the form given by equation 4.13 is used;

$$x(t) = Tz(t) \qquad \text{(eqn 4.13)}$$

T = ML and M is the eigenvector matrix (equation 4.6) The matrix L is given by,

$$L = \begin{bmatrix} 0.5 & -0.5j & 0.0. & \ldots & 0.0 \\ 0.5 & 0.5j & 0.0 & \ldots & . \\ 0.0 & 0.0 & 1.0 & \ldots & . \\ . & \ldots & \ldots & 1.0 & . \\ 0.0 & \ldots & \ldots & \ldots & 1.0 \end{bmatrix}$$

(eqn 4.14)

The transformed system is then given by,

$$\dot{z}(t) = \Lambda Z(t) + T^{-1}Bu(t) \qquad \text{(eqn 4.15)}$$

with performance index ,

$$J = \int (z^T T^T Q\ Tz + u^T Ru\ )\ dt \qquad \text{(eqn 4.16)}$$

where $\hat{Q}$ is now given by $\hat{Q} = T^T\ Q\ T$.

It can be shown that to move a pair of complex eigenvalues given by s = a + bj, a weighting matrix of the form

$$\hat{Q} = \begin{bmatrix} 0 & 0 & 0 \ldots & 0 \\ 0 & \hat{q}_i & 0 & . \\ 0 & 0 & \hat{q}_{i+1} & . \\ 0 & . & . & . \\ 0 & . & . \; . & 0 \end{bmatrix}$$

(eqn 4.17)

with $\hat{q}_i = \hat{q}_{i+1}$ is required.

In a similar manner, $\hat{Q}$ can be obtained by using the optimization routine, with the condition $\hat{q}_i = \hat{q}_{i+1}$ formulated as a constrained equation. Inverse transformation and determination of Q_e and F_e are identical to the distinct eigenvalue case with M in equation 4.9 replaced by T.

3. System With Repeated Eigenvalues

In this case, the system matrix cannot be diagonalized but the general procedure given above still apply. The system is first transformed into the Jordan canonical form ;

$$J = U^{-1} A U \qquad \text{(eqn 4.18)}$$

where U is a transformation matrix which is not the eigenvector matrix M. An example of the Jordan form is given below,

$$J = \begin{bmatrix} \lambda & 1 & 0 \\ 0 & \lambda & 1 \\ 0 & 0 & \lambda \end{bmatrix}$$

(eqn 4.19)

The only difference with the two procedures mentioned above is that the pole reassignment has to begin at the bottom of each Jordan block. For example, in the system given above (equation 4.19), the first re-assignment will result in a new system given by equation 4.20

$$J = \begin{bmatrix} \lambda & 1 & 0 \\ 0 & \lambda & 1 \\ 0 & 0 & s_d \end{bmatrix}$$

(eqn 4.20)

Determination of Q and F using the optimization and the inverse transformation routines are identical to the distinct eigenvalues case with M in equation 4.9 replaced by U.

D. COMPUTER PROGRAM DESCRIPTION

The computer aided design package developed in this thesis is illustrated in Figure 4.1 The program is developed using the top-down approach with special purpose subroutines called by the main control program. The driver program supports three independent modes with the data entry portion common to all modes:

1. Data Entry : The system matrices A, B, C and/or F, Q, R etc are entered through a data file. Design variables such as desired poles locations, elements of matrices to be varied etc, are also specified through the input data file.
2. Pole Placement Mode : In this mode, an arbitrary set of closed-loop eigenvalues is assigned by selecting the appropriate state weighting matrix. As shown in Figure 4.1, the transformation matrices for various

cases are computed first. The pole placement is achieved using the numerical optimization routine described in the last section; Q is obtained and then inverse-transformed to the original co-ordinate system. If desired, the results can be used as an input to the Linear Quadratic Control Program.

3. Linear Quadratic Control Mode : This part of the program is adopted from the OPTSYS program. Given a set of weighting matrices Q and R and the system matrices A, B and C, it computes the steady state feedback gain F, closed-loop eigenvalues, etc.
4. Singular Value Analysis Mode : This portion of the program allows the designer to analyze various designs obtained from the two modes mentioned above in term of singular value vs frequency plot. The main part of this program is adopted from [Ref. 22].

The three modes of operation mentioned above may be used in any order to implement specific design objectives. A typical design process will involve runs alternating between the three modes until a compromise between primary and secondary design objectives is achieved. Record of a typical design run together with a complete listing of the main program and their non-standard subroutine are given in Appendix B and C.

E. DESIGN PHILOSOPHY

The Linear Quadratic constant state feedback design philosophy for the linear time-invariant model is illustrated in Figure 4.2. It assumes that the location of the eigenvalues and system time response are the main design objectives. These objectives can be achieved using the pole placement procedure developed in this thesis. Single or multiple reassignment can be made in one run. A number of

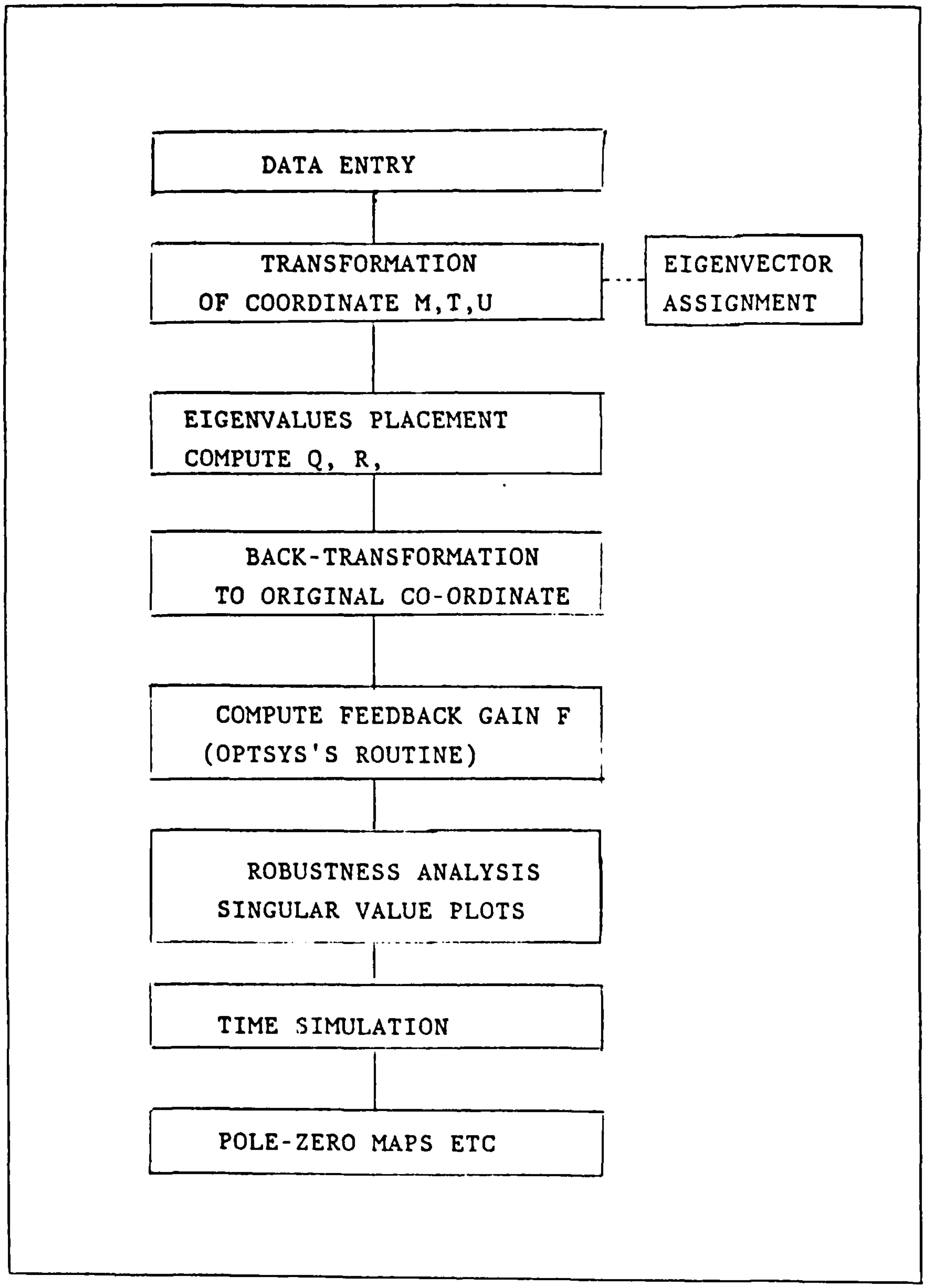


Figure 4.1 Computer Aided Design Package Organization.

designs can then be obtained using different starting control weighting matrices, different state weighting matrices and different assignment sequences. Physical constraints such as control input amplitude, control input energy as well as general system properties such as asymptotic behavior are heavily relied upon during this process. After the major objectives are satisfactorily achieved, secondary design objectives are considered. These include feedback gain reduction (by increasing the control weighting matrix), robustness (in terms of minimization of system sensitivity to modelling errors and/or parameter variations), zeros locations, eigenvectors assignments. The extra degrees of freedom available in the MIMO state feedback system can often provide a means to improve these secondary objectives while only slightly modifying the closed-loop pole assignment and thus the time response.

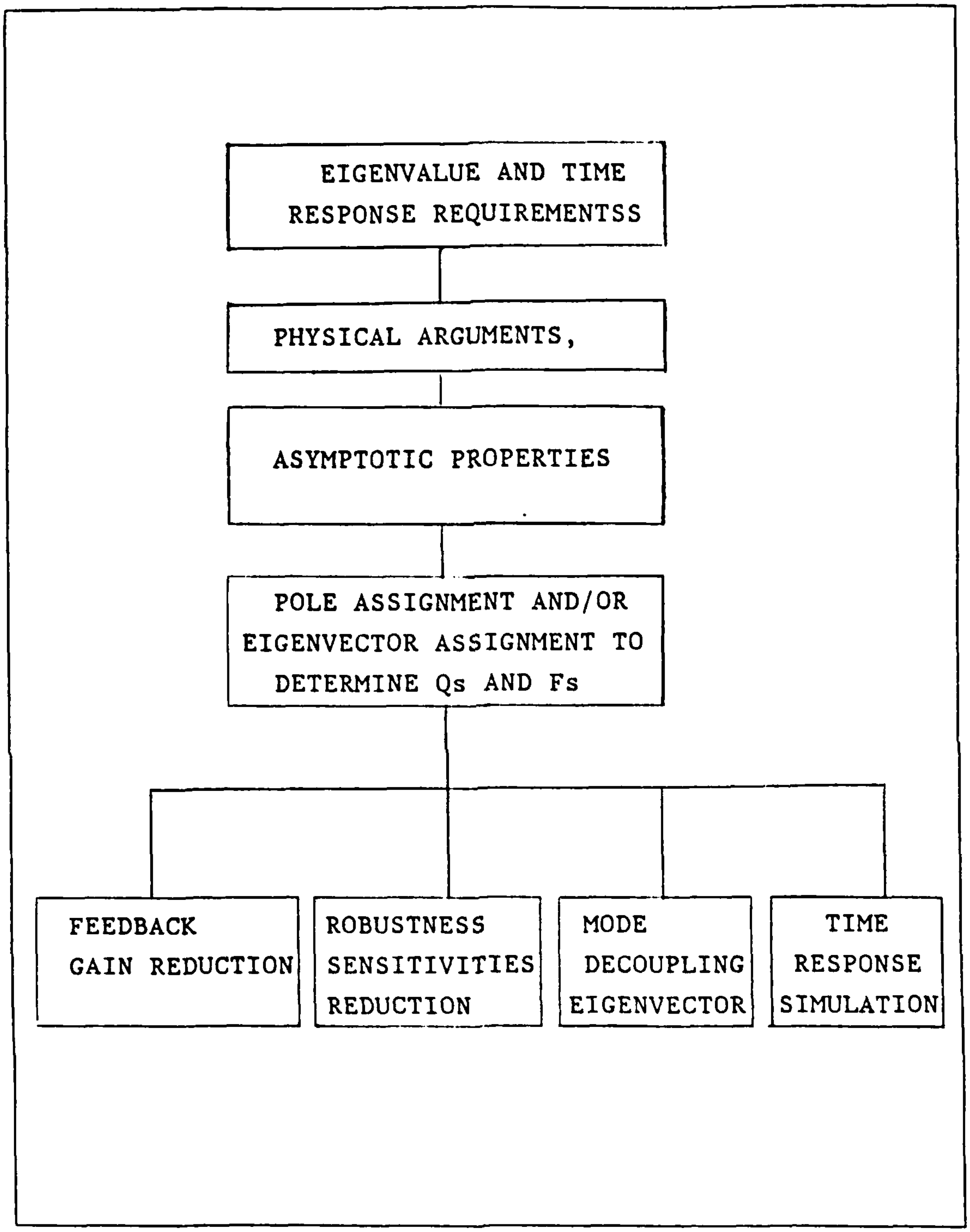


Figure 4.2 LQ Eigenvalue Assignment Philosophy.

V. DESIGN EXAMPLES

The design procedure described in Chapter 4 is illustrated in this Chapter by two design problems. A typical LQ structure and its properties, together with the pole placement procedure is first presented with a 2x2 model. A practical design problem is then presented for the highly coupled lateral channels of a CH-47 Helicopter. The resulting LQ designs are compared with other multivariable state feedback designs [Refs. 22,23]. It is shown that the procedure developed here is a viable tool for robust constant feedback controller design.

A. INTRODUCTORY 2X2 PROBLEM

This problem formulated in reference 23 serves to demonstrate how a highly cross-coupled multivariable control problem can be formulated and solved as a linear quadratic design problem, using the pole placement procedure. The problem provides excellent insight into the structure of the multivariable LQ system and its built-in robustness to cross-coupled perturbation.

Figure 5.1 shows a diagram of this basic 2X2 system in which the plant is given by

$$\dot{x}(t) = Ax(t) + Bu(t) \qquad \text{(eqn 5.1)}$$

$$y(t) = Cx(t) \qquad \text{(eqn 5.2)}$$

where

$$A = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} \qquad B = \begin{bmatrix} 1 & 50 \\ 0 & 1 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

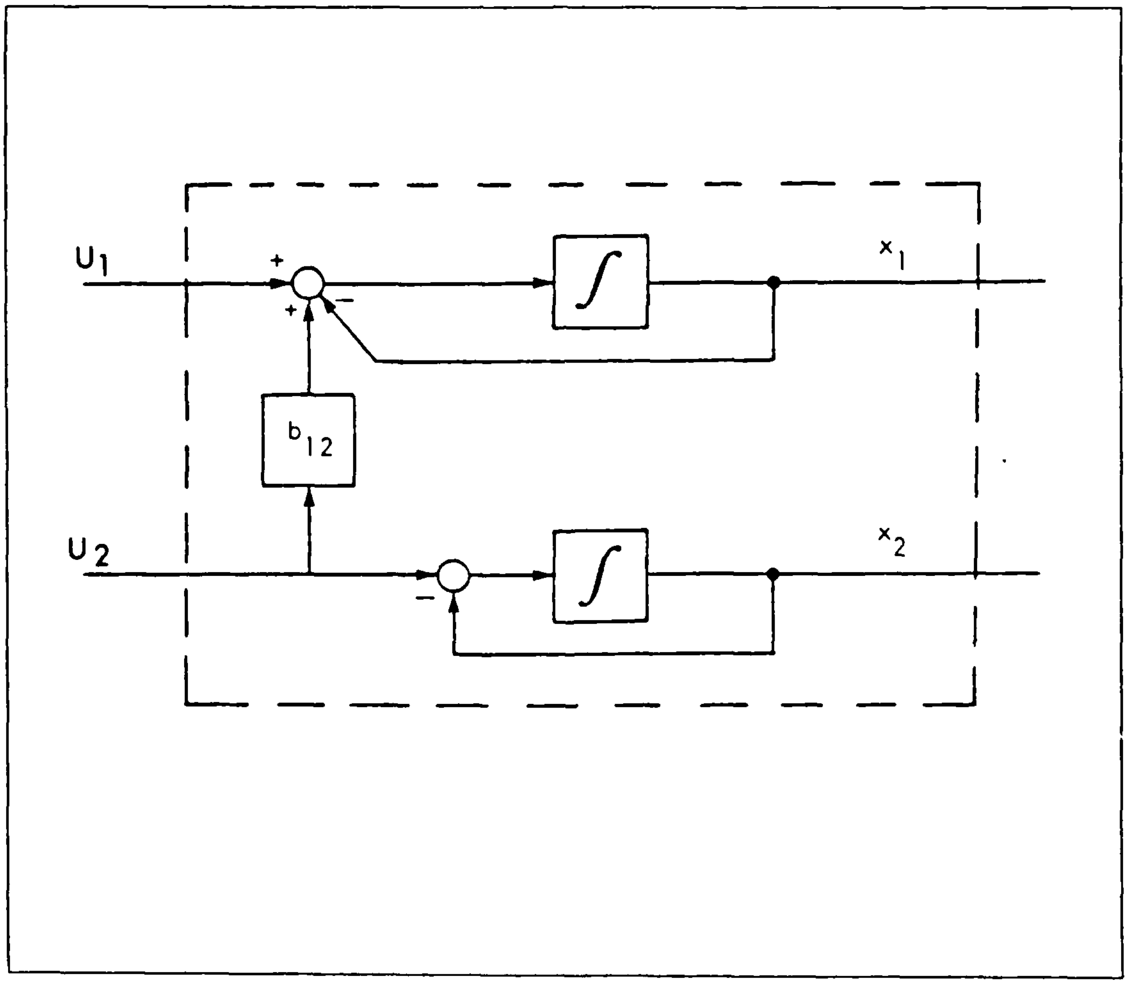


Figure 5.1 A Simple 2X2 MIMO Model.

This system has open-loop eigenvalue at -1, -1 and is therefore stable. The b_{12} term in the control matrix B is purposely made large to produce the cross-coupled effect from channel two to channel one(see Figure 5.1). The design requirement is to select a set of feedback gain F such that the closed-loop eigenvalues are at -2, -2. Assuming that this is the only requirement, it is not difficult to see that a unity feedback law of the form (Figure 5.2),

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} u_{c1} \\ u_{c2} \end{bmatrix}$$

(eqn 5.3)

will produce the desired closed-loop system given by,

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} -2 & 50 \\ 0 & -2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 & 50 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u_{c1} \\ u_{c2} \end{bmatrix}$$

(eqn 5.4)

The above design seems to be acceptable as far as eigenvalues or time response is concerned. It is now shown that when robustness of the system is considered, the unity feedback gain controller performs rather poorly. On the other hand, design using LQ Pole Placement type of formulation will result in robust controllers. To demonstrate the lack of robustness of the unity feedback design, the feedback gain matrix F as given in equation 5.4 is perturbed slightly (by +5%) and the eigenvalues of the resulting closed-loop

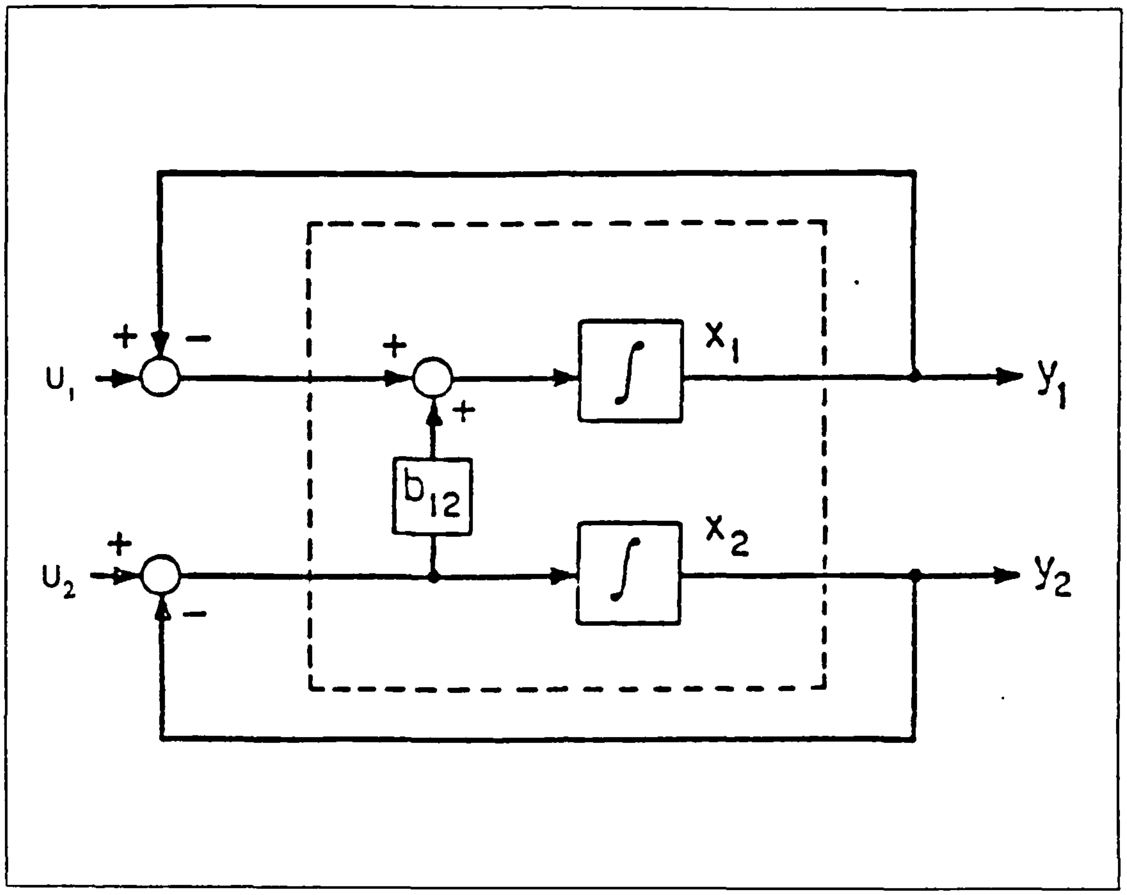


Figure 5.2 Unity Feedback for the 2X2 Model.

system matrix are calculated. The absolute and percentage errors in the eigenvalue due to the perturbation of the closed-loop system are given in Table I .

It can be seen from the above that the first cut design is very susceptible to model and feedback perturbation. A 5 % changes in the f_{21} term has resulted in a large shift (160%) in the closed-loop eigenvalue. Lack of robustness in the unity feedback design can also be seen in terms of the minimum singular value plot of the return difference matrix in the frequency domain. For the unity feedback gain system shown in Figure 5.2, the return difference matrix is given by,

TABLE I

PERTURBED EIGENVALUE FOR 2X2 MODEL

Perturbation in F (+5%)	Absolute Changes in Eigenvalues	Percent Changes in Eigenvalues
f_{11} and f_{22}	0.05 , 0.05	2.5 , 2.5
f_{12}	0.0 , 0.0	0.0 , 0.0
f_{21}	0.67 , +3.265	33.5 , 163.0
All	1,255, 5.346	37.25 , 167.0

$$I + G(s) = \begin{bmatrix} s+2/(s+1) & 50/s+1 \\ 0 & s+2/(s+1) \end{bmatrix} \qquad \text{(eqn 5.5)}$$

where $G(s) = C(sI-A)^{-1}B$ is the loop transfer matrix as indicated in Figure 5.3

The multivariable Nyquist diagram (locus of det[I+G(s)] for the system is shown in Figure 5.4 If this diagram is interpreted as for a single input system, the $(-1/2, \infty)$ gain margin and (± 160) phase margin would lead one to believe that the design is a good one . This has been shown to be not the case, a 5% perturbation in F would cause the system to become unstable. The above clearly demonstrate the inadequacy of the classical method in evaluating stability margin for MIMO system.

Robustness properties of the unity feedback controller will now be analyzed in terms of singular value as discussed

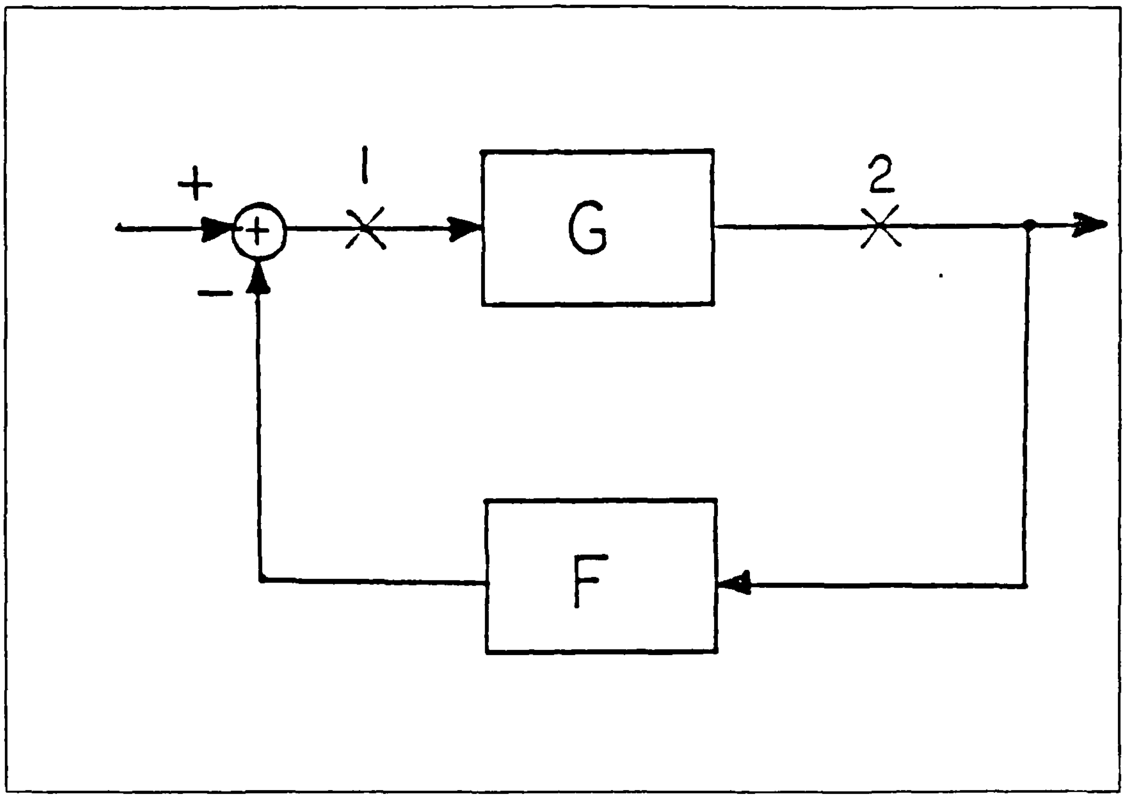


Figure 5.3 Loop Transfer Matrix - F = I.

in Chapter 3. The minimum singular value plot of [I+G(s)], is shown in Figure 5.5 as a function of frequency. The lack of robustness is clearly indicated by the relative small singular value at frequency around 2 rad/s. Using the universal phase and gain margin chart developed in [Ref. 19], the minimum singular value at this frequency corresponds to a gain margin of (0.91, 1.0) and a phase margin of (± 4 deg).

It is now shown that formulation using LQ approach and the pole placement procedure developed here will result in robust design that meet the time response requirement. Furthermore, better insight of the design process can be obtained from the procedure to be described here. The first step in the LQ design is to determine the asymptotic

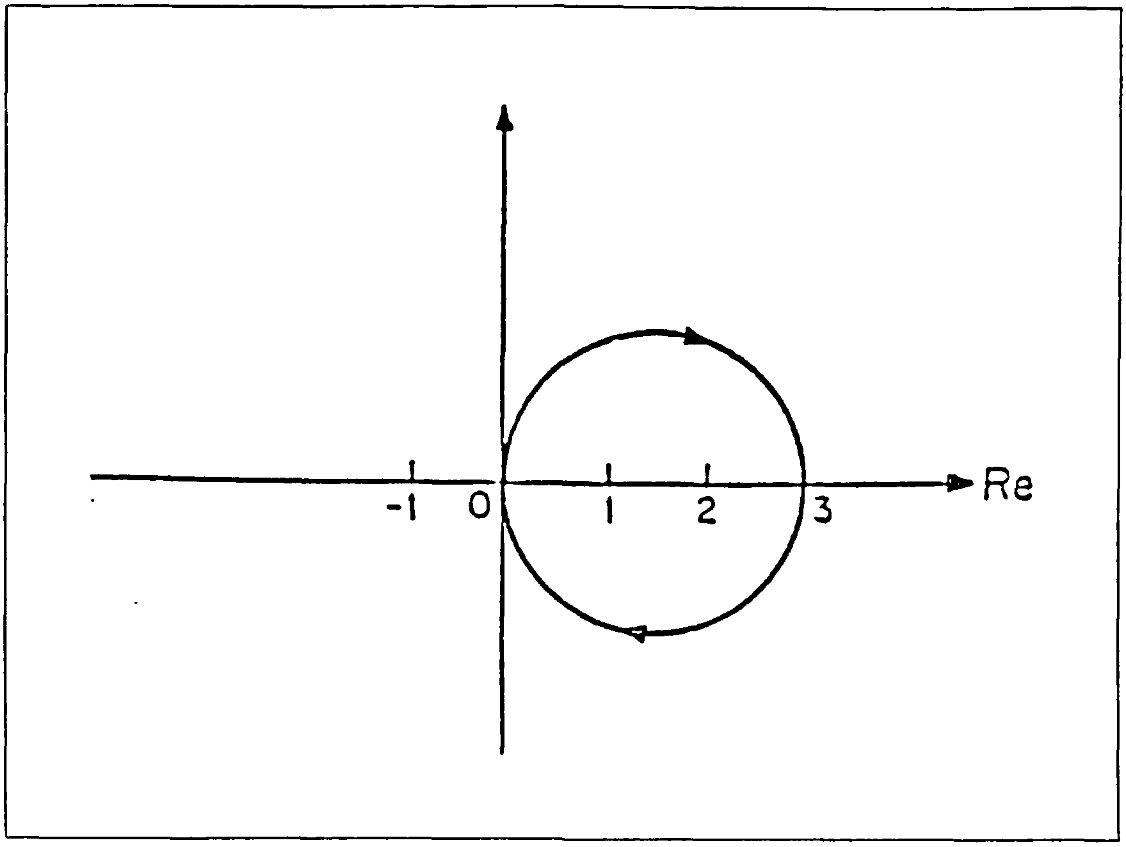


Figure 5.4 Multivariable Nyquist Plot.

properties of the system, i.e. movement of closed-loop poles as $R \to 0$. Using results from Chapter 2, it can be established that as R increase from 0 to ∞, both closed-loop poles move from infinity on the real axis to the open-loop poles location. None of the closed-loop poles stay finite as $R \to 0$ since the dimension of the input control vector is equal to the dimension of the state. Assuming that R = I, the pole placement is accomplished in two steps. The first step is to move the open-loop pole at -1 to -2.0. As the system matrix A given is already in Jordan form, no transformation is required. The pole placement program puts the pole at -1.9987 with,

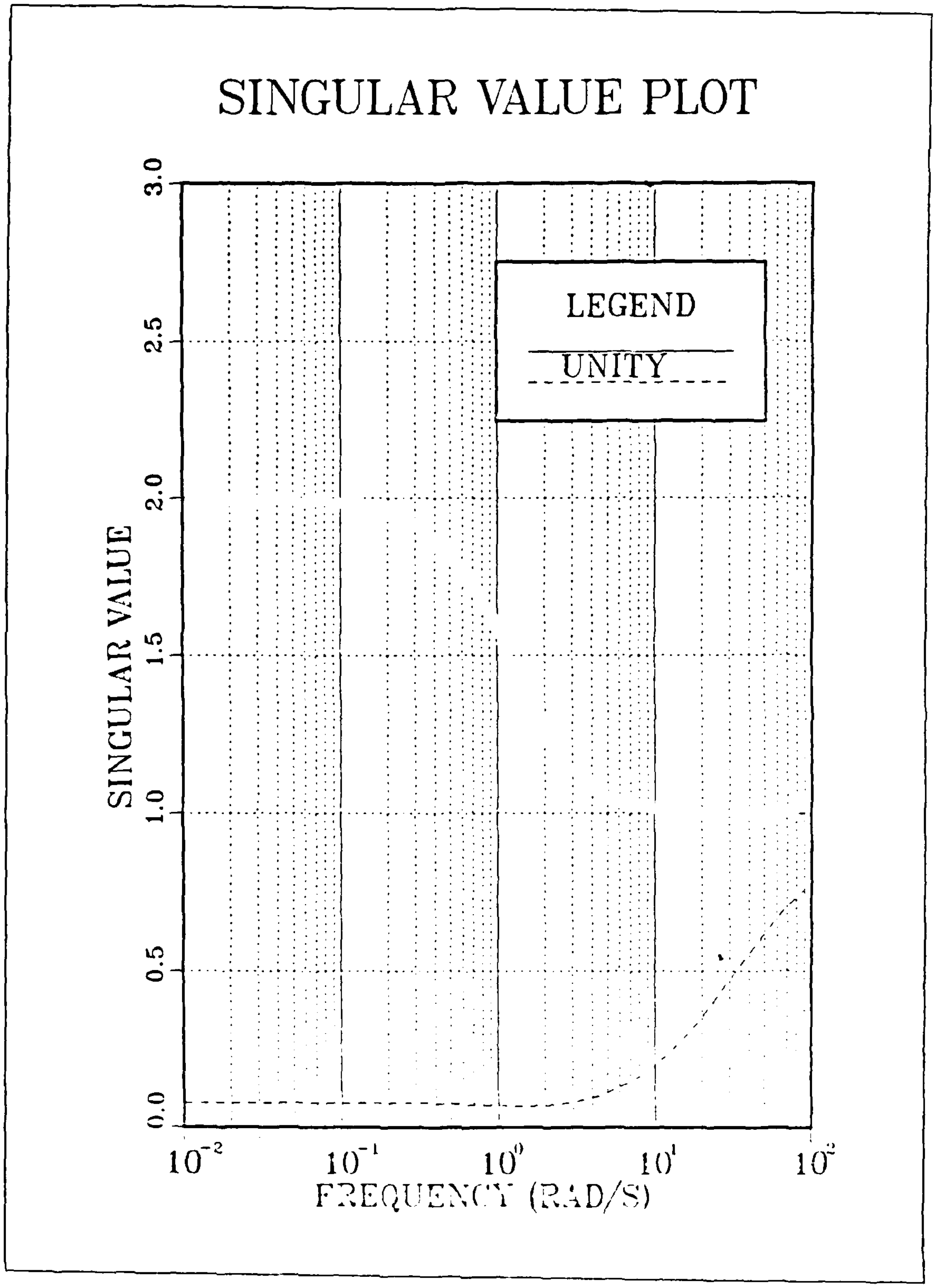


Figure 5.5 Singular Value Plots - Unity Feedback.

$$Q_1 = \begin{bmatrix} 0.00119 & 0 \\ 0 & 0 \end{bmatrix} \qquad F_1 = \begin{bmatrix} 0 & 0 \\ 0.01983 & 0 \end{bmatrix}$$

and

$$A_{aug} = \begin{bmatrix} 1.9915 & 0 \\ 0.01983 & 1.0 \end{bmatrix}$$

In the second step, the other open-loop pole at -1. is moved to -2.0 , using A_{aug} as the new plant matrix. The resulting Q and F and the augmented plant matrix become,

$$Q_2 = \begin{bmatrix} -2.998 & -149.9 \\ -149.9 & 7499.3 \end{bmatrix} \qquad F_2 = \begin{bmatrix} 0.9994 & -49.978 \\ -0.00841 & 0.4211 \end{bmatrix}$$

The effective Q_e and F_e required to move both open-loop poles at -1.0 to -2. are $Q_e = Q_1 + Q_2$ and $F_e = F_1 + F_2$ as shown in equation 5.6 below. The pole placement procedure is completed with the final eigenvalues placed at (-1.99255, ±j0.05628).

$$Q_e = \begin{bmatrix} 2.99919 & -149.9 \\ -149.9 & 7499.3 \end{bmatrix} \qquad F_e = \begin{bmatrix} 0.9994 & -49.978 \\ 0.0114 & 0.4211 \end{bmatrix}$$

(eqn 5.6)

The singular value plots of the case where R = I together with the unity feedback gain (non-LQ design) are shown in Figure 5.6 The well established fact that LQ design possesses ($1/2, \infty$) gain margin and ($\pm$60deg) phase margin can also be readily observed from the same figure as the minimum singular values of [I+G], $\underline{\sigma}$, is greater than one for all frequency. Changes in closed-loop eigenvalue for a small perturbation in F is again computed as shown in Table II . It can be seen that the LQ design is robust with the largest percentage change in eigenvalues location of only 10%, when compared with the 160% change in the unity feedback design.

TABLE II

PERTURBED EIGENVALUE FOR LQ DESIGN

Perturbation in F (+5%)	Absolute Changes in Eigenvalues	Percent Changes in Eigenvalues
f_{11} and f_{22}	0.2149 , 0.142	10.7 , 7.1
f_{12}	0.119 , 0.119	5.59 , 5.59
f_{21}	0.0592, 0.0578	2.90 , 2.89
All	0.07 , 0.070	3.52 , 3.52

The pole-zero plots of various closed-loop transfer functions of the closed-loop transfer matrix for both the unity feedback and LQ design are compared in Figure 5.7 and 5.8 . For the unity feedback design, zero at minus three for the input 2 to output 1 channel corresponds to the

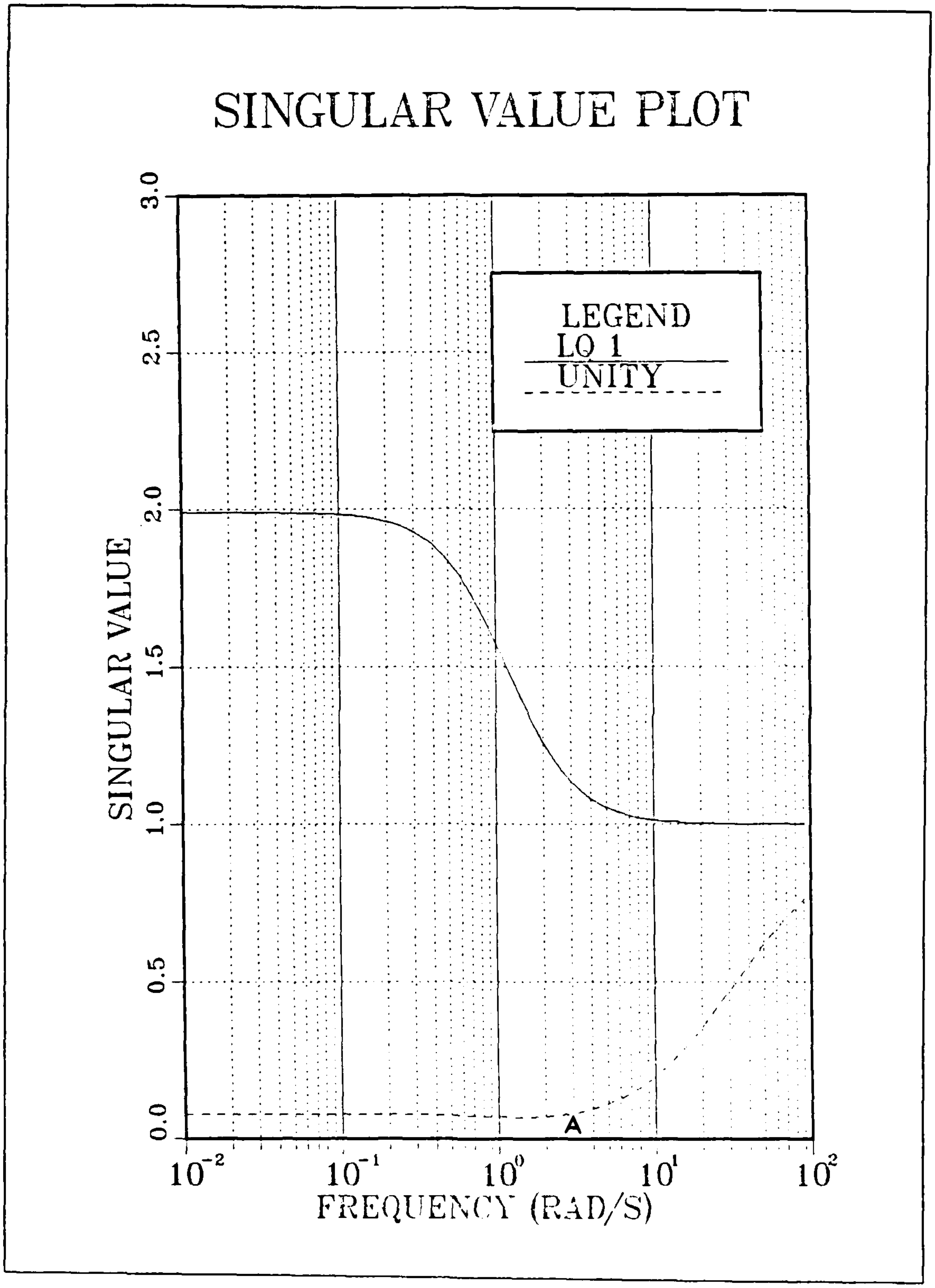


Figure 5.6 Comparison-Singular Value Plots (2x2 model).

minimum singular value frequency (point A in Figure 5.6). It is in fact this zero that causes the the system to be sensitive to perturbation. For the LQ design, the built in robustness cause the zero at minus three to move toward the pole location at -2.

Improvement in the robustness from LQ design is now analyzed in terms of Bode plots. The open-loop gain and phase vs frequency plots for a MIMO system can be obtained from the open-loop transfer function matrix, G(s). For full state feedback, $G(s)=F(sI-A)^{-1}B$. (if the open-loop plots without feedback is considered, $G(s)=C(sI-A)^{-1}B$) The matrix G is a matrix-valued rational function of s, it describes how the system (with or w/o feedback) appears to its environment. It is an external description of the system and is closely related to the zeros of the system. The open-loop Bode plots for the two designs are compared in Figures 5.9 to 5.11 As $b_{21}=0$, there is no coupling from channel 1 to channel 2. All channels have a -20 dB/decade slope at high frequency which is in agreement with earlier observation that each channels has one finite zero. It is interesting to note that the unity feedback and LQ design result in almost identical gain vs frequency plots for direct channels (i.e. channel 1-1 and 2-2). Any classical single loop type of analysis will not be able to detect any difference between the two designs. On the other hand, cross-coupling effect can be readily seen from the gain vs frequency plot in channel 2-1 (Figure 5.10). The unity feedback design is characterized by the rather large channel 2-1 gain at low frequency. A very small perturbation in channel one's parameter can change the system behavior considerably. This has been illustrated earlier by perturbing the feedback gain. It can be seen from the figure that LQ design reduces the crossfeed gain by large amount.

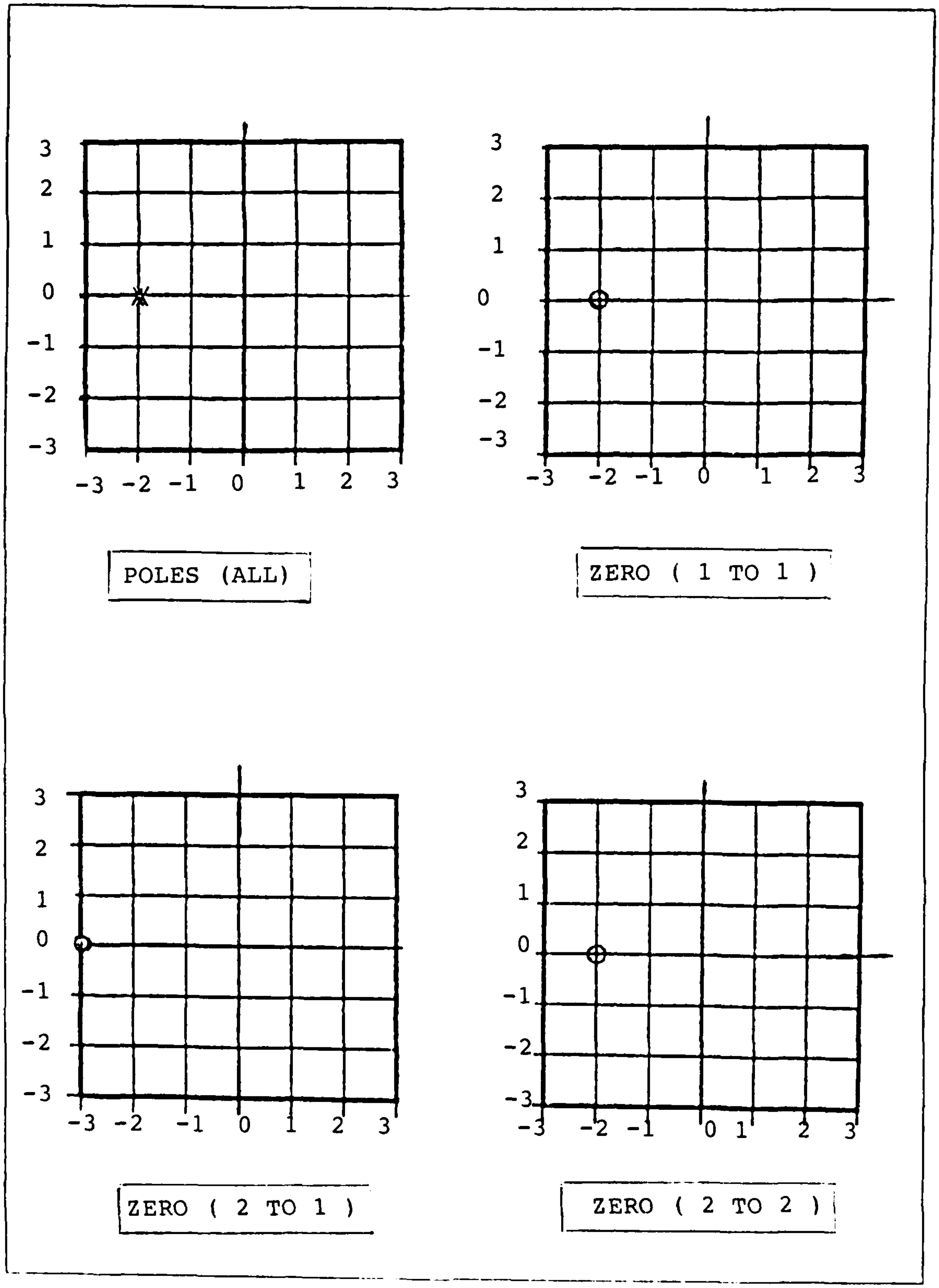


Figure 5.7 Closed-Loop Pole-Zero Plots (Unity Gain FB).

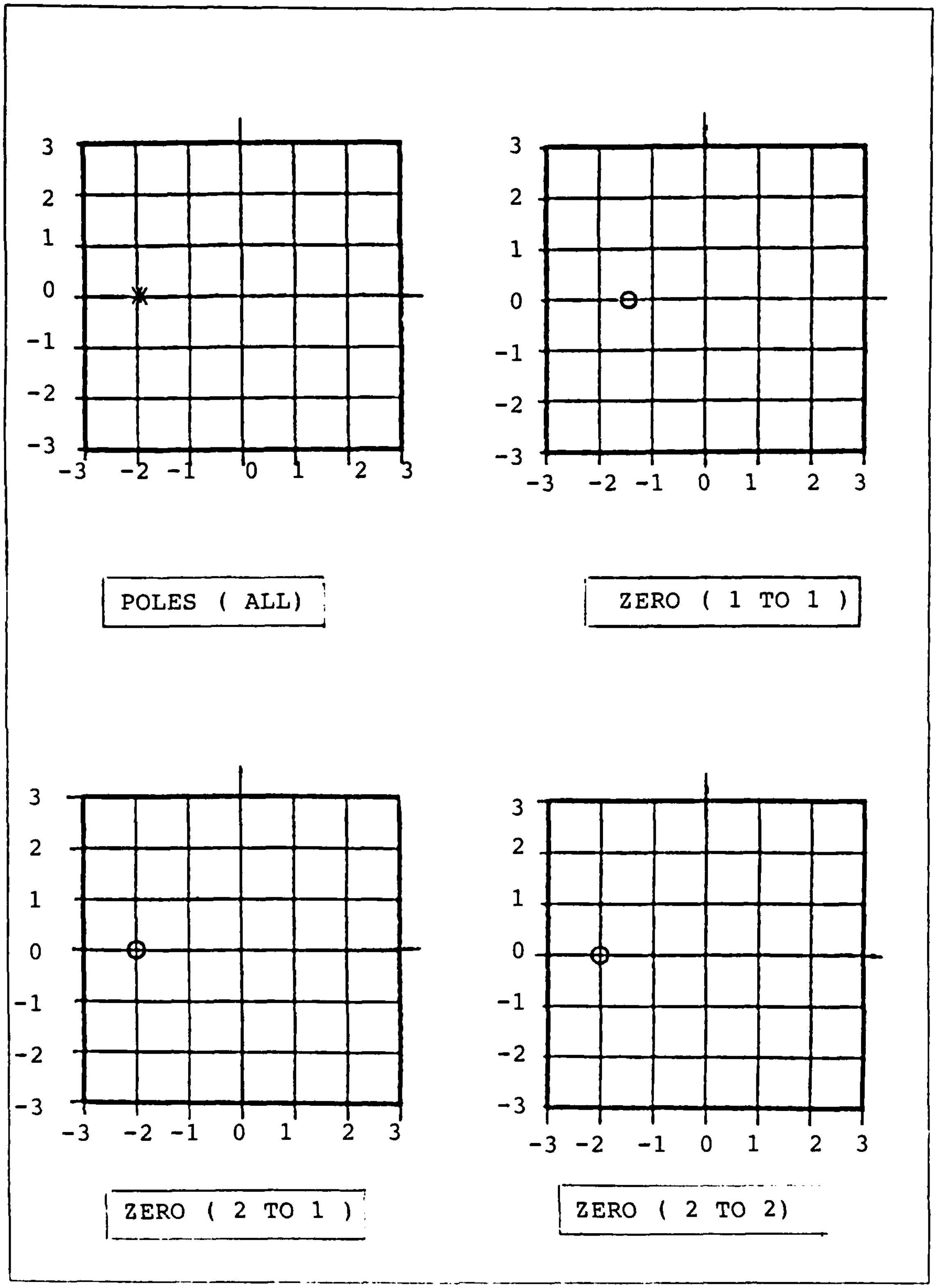


Figure 5.8 Closed-Loop Pole-Zero Plots (LQ Design).

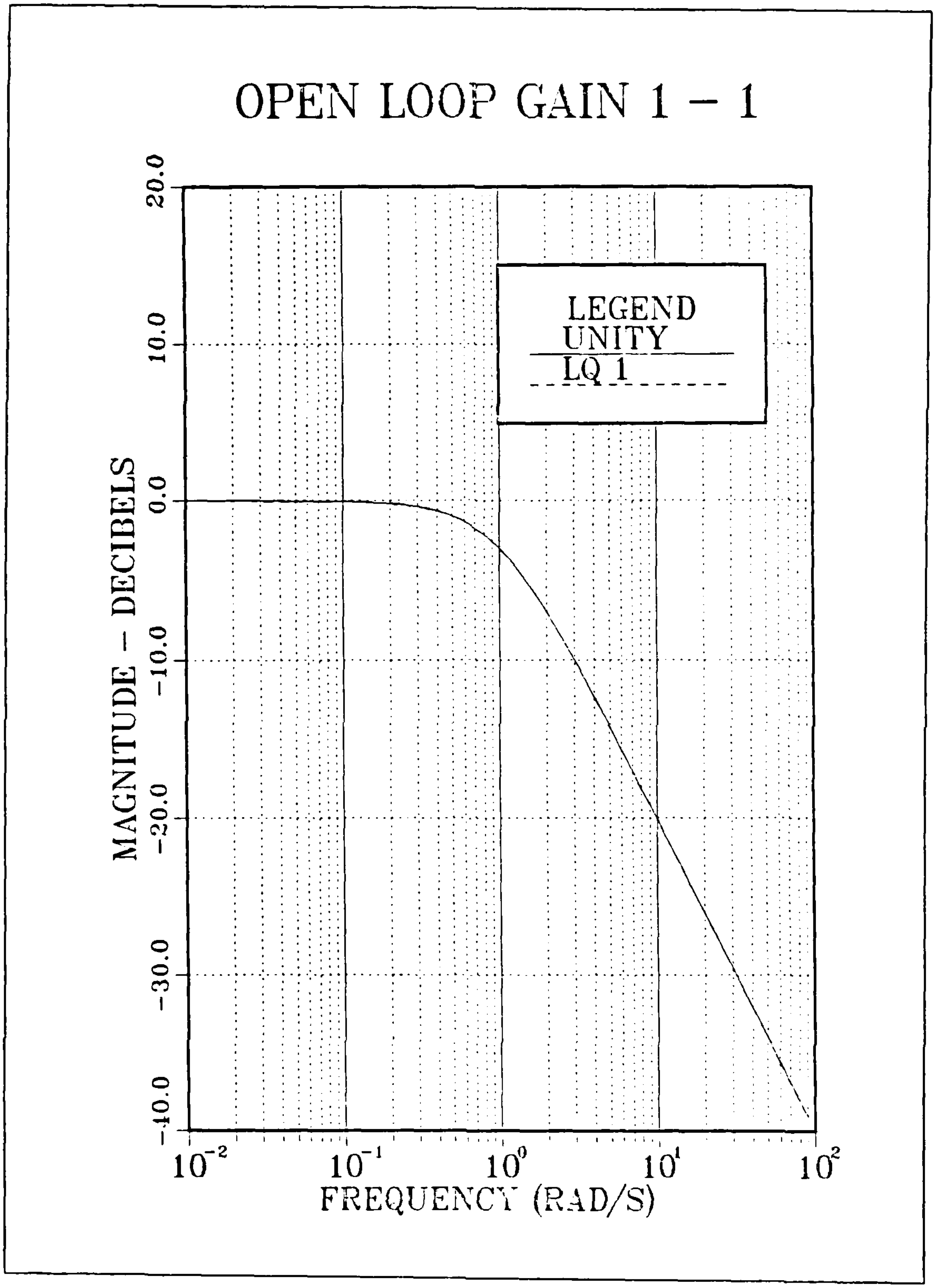


Figure 5.9 Open-Loop Bode Plots Channel 1-1.

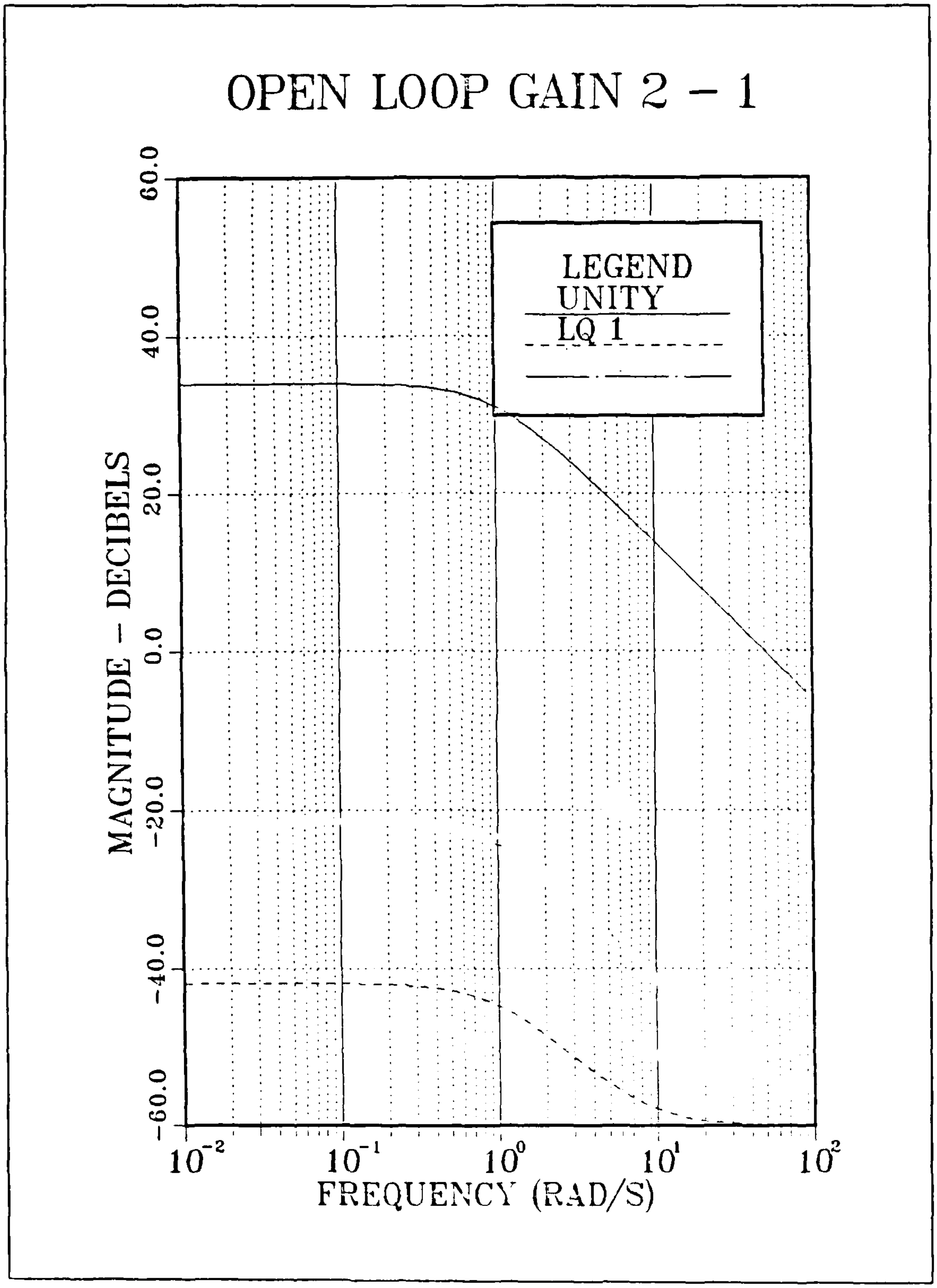


Figure 5.10 Open-Loop Bode Plots Channel 2-1.

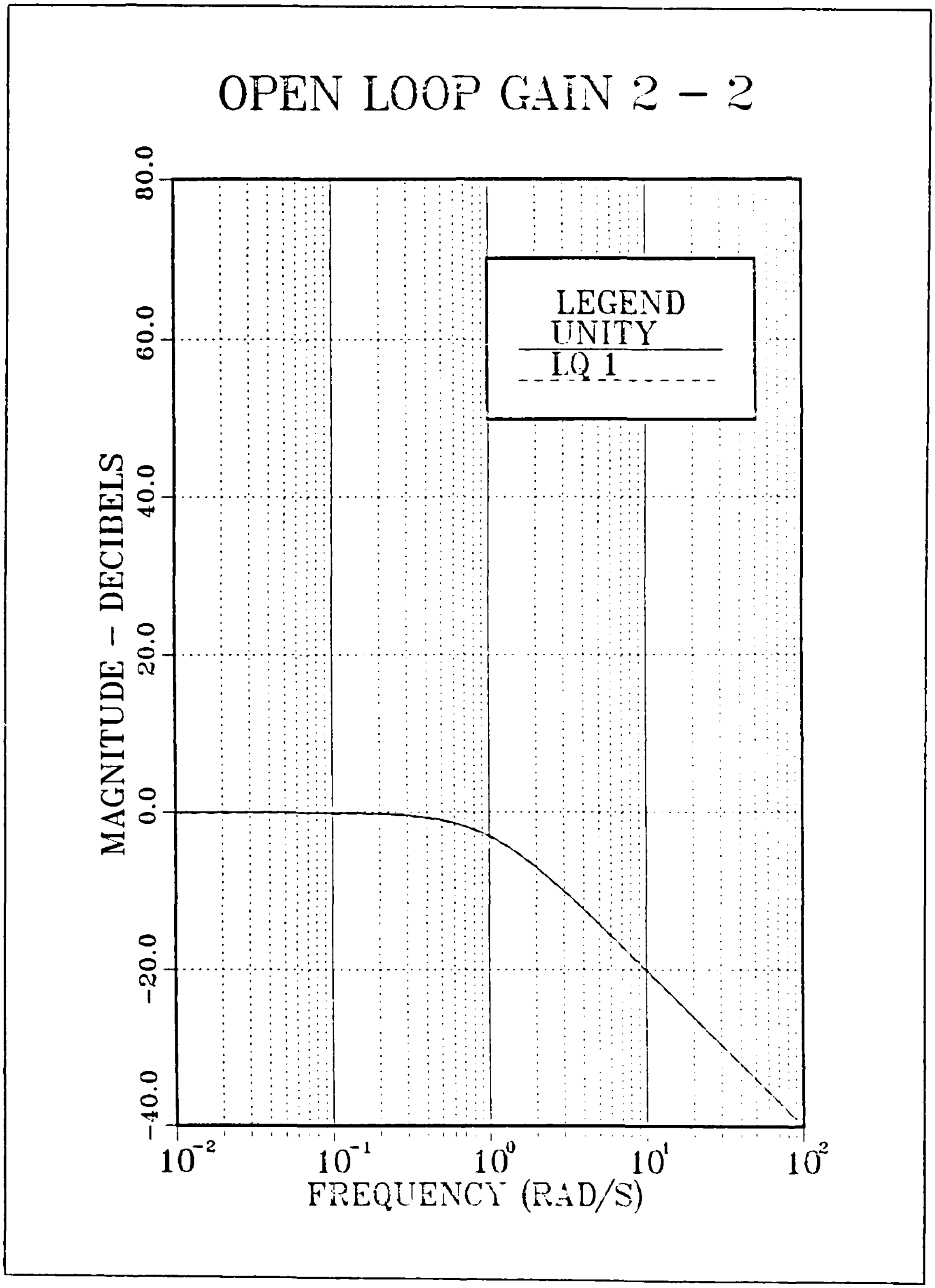


Figure 5.11 Open-Loop Bode Plots Channel 2-2.

B. A HELICOPTER DESIGN PROBLEM

The design procedure described in Chapter 4 is further illustrated in this section by an actual design problem. A controller is designed using the Linear Quadratic Pole Placement approach for the lateral dynamic model of a CH-47 helicopter. The resulting design is then compared to multivariable state feedback controller given in [Ref. 23].

The highly coupled two inputs lateral axis model of the CH-47 helicopter is used as a full order system. The state vector x(t) and control input vector u(t) are given by,

$x_1 = v =$ Y-axis velocity (ft/sec)
$x_2 = p =$ Roll rate (rad/sec)
$x_3 = r =$ Yaw rate (rad/sec)
$x_4 = \phi =$ Roll angle (rad)

$u_1 = \delta_b =$ Yaw rate rotor deflection control (inches.)
$u_2 = \delta_c =$ Roll rate rotor deflection control (inches.)

The state variables and the body axes of the aircraft are illustrated in Figure 5.12 The yaw and roll rotor deflection control produce changes in the yaw rate, slide slip angle, roll rate and bank angle. Assuming full state feedback, the A, B, and C system matrices are given by,

$$A = \begin{bmatrix} -2.27 & 1.420 & -0.15 & 31.99 \\ .01 & -0.7 & -0.07 & 0.0 \\ 0.04 & -0.05 & -0.05 & 0. \\ 0. & 1. & 0.11 & 0. \end{bmatrix}$$

$$B = \begin{bmatrix} 0.12 & 0.95 \\ 0.04 & -8.37 \\ .34 & .020 \\ 0.0 & .0 \end{bmatrix} \qquad C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

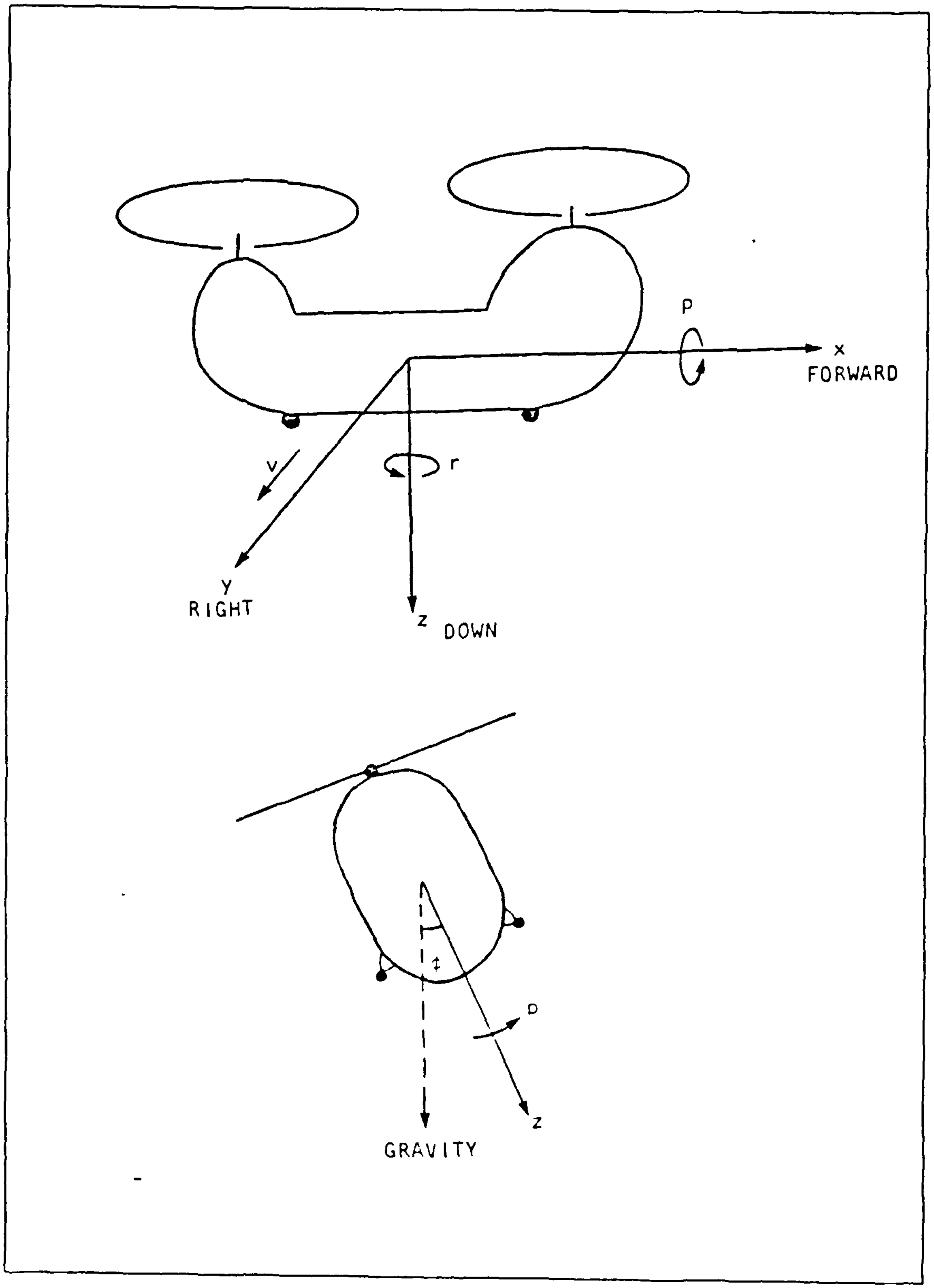


Figure 5.12   State Variables and Body Axes for CH-47.

The open loop eigenvalues of this system are;

$s_1 = +0.2065$
$s_2 = -0.0503$
$s_3 = -1.0498$
$s_4 = -2.1263$

The open-loop system is not stable and the time response is shown in Figure 5.13 for zero input and an initial condition of $\phi(0)=0.1$ rad

The design requirements are to satisfy specification in terms of the step input response of the roll attitude channel (ϕ / ϕ_c). Stability margin requirements are those given by standard military specification. In [Ref. 23] three designs were obtained to satisfy the desired time response performance specification. All three control laws are of the form given by,

$$u(t) = -Fx(t) + h\phi_c(t) \qquad \text{(eqn 5.7)}$$

The values of F and h are summaried in Table III . It was shown in [Ref. 23] that two of the designs (design 1 and 2) were extremely sensitive to model errors and perturbations. It is now shown that LQ formulation using the pole placement procedure developed here will result in robustness design and yet satisfy the conventional time response criteria.

It is assumed that the closed-loop poles requirement are the same as those obtained in the AlphaTech's design 1. (-25.12, -12.51, -9.652, -2.125). The first step in the design is to establish some asymptotic properties of the system. The dimension of the control input (2) is less than the dimension of the state(4). As shown in Chapter 2, this implies that at least two closed-loop poles approach minus infinity in the complex plane when $R \rightarrow 0$. When $R \rightarrow 0$ (meaning no control input allowed), the closed-loop poles

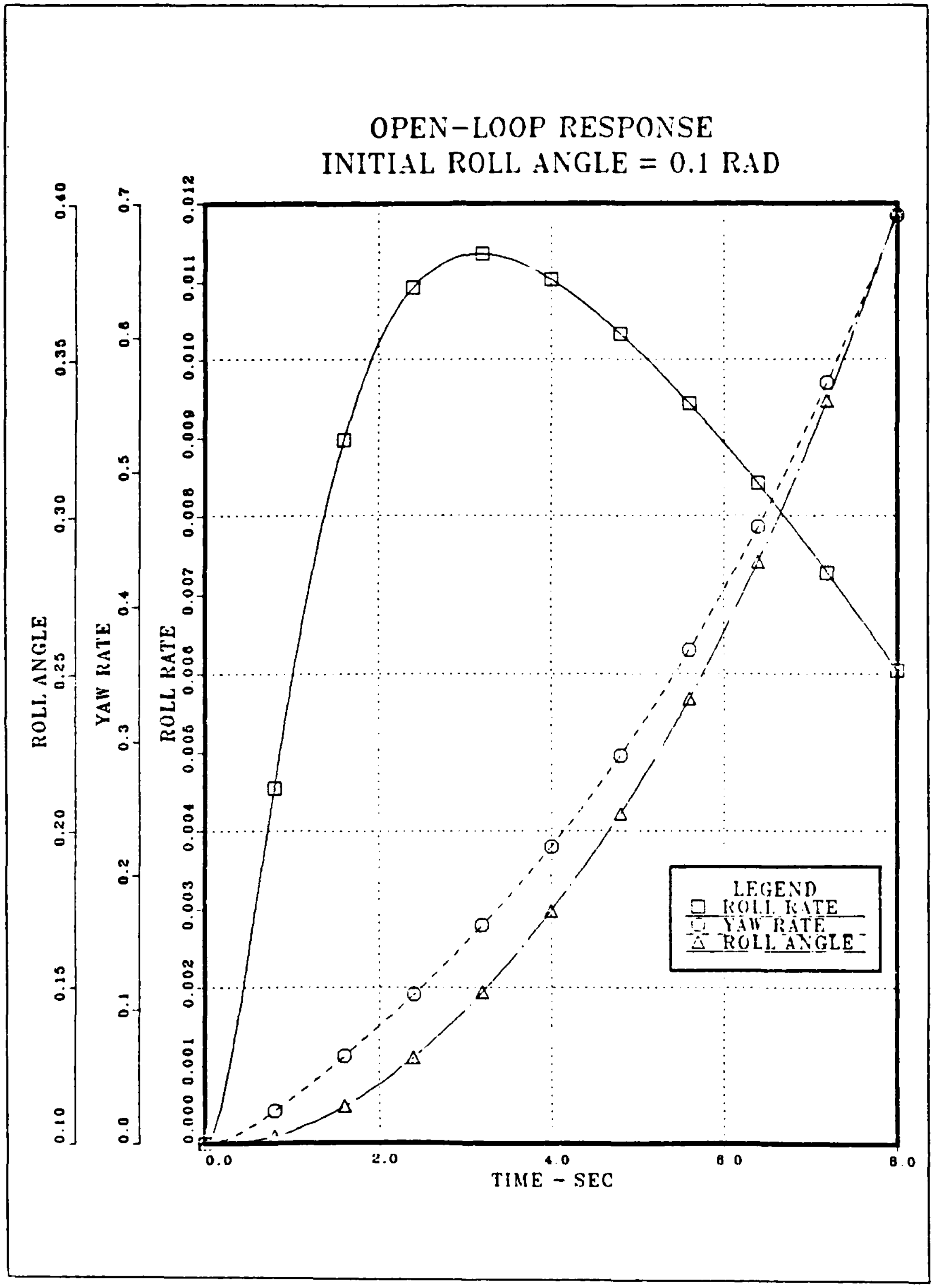


Figure 5.13 Open Loop Time Response $\phi(0) = 0.1$ rad.

TABLE III

ALPHATECH DESIGNS (1, 2, AND 3)

Designs	Feedback Gains (F)				h
one	-1.72 0.024	-23.5 -2.71	70.6 0.368	595 -7.99	595 -7.99
two	0.198 -0.01	154. -1.592	18.3 -0.189	142.0 -1.47	142.0 -1.47
three	0 0	0 -4	25.5 0	0 -27	0 -27

approach the open-loop poles or their mirror images if they are in the right hand plane. (i.e. -0.2065, -0.0503, -1.04987, and -2.12635).

The next step in the design procedure is to select a suitable starting control weighting matrix. Based on guideline given in [Ref. 4] and assuming that u_{1max} and u_{2max} are equal to 1 inch. R = I is selected for the initial design. (In the present formulation ρ is not needed as only Q is varied to place the pole).

The sequence of the reassignment is determined next. At present there is no known established guideline for selecting the prefered sequence of moving the poles. Two extreme sequences are considered as follows,

A. Move the left most open-loop pole to the left most desired closed-loop pole etc. The reassignment sequence then become:

1. move pole at -2.1265 to -25.12
2. move pole at -1.0483 to -12.51

3. move pole at -0.2065 to -9.652
4. move pole at -0.0503 to -2.125

B. Move the right most open-loop pole to the left most desired closed-loop pole etc. For the problem given, the reassignment sequence are :

1. move pole at +0.2065 to -25.12
2. move pole at -0.0503 to -12.51
3. move pole at -1.0498 to -9.652
4. move pole at -2.126 to -2.125
   (close, so no move required)

Using the pole placement program developed in this thesis, the corresponding Q and F for each of the reassignments are obtained and tabulated in Tables IV and V for the two cases selected (design LQ-A and LQ-B). It can be seen from the table that difference reassignment sequence results in different set of Q_e and F_e . In general, once the matrix R and the n closed-loop poles are selected, there are $n(n+1)/2$ extra degrees of freedom available in Q. This also verifies the well-known fact that the general solutions of Q and R for a given set of closed-loop eigenvalue are non-unique. The elements of the Q matrix obtained during each reassignment depends on the transformation matrix (M, L or U) and hence the eigenvectors that are used to construct them. The important of eigenvector type of assignment is evident as Q, F and the resulting closed-loop poles depend on the eigenvectors used. The designer can shape the design by choosing appropriate eigenvectors for the transformation matrix. The application of these extra degrees of freedom will be discussed in the following section. The resulting designs (LQ-A and LQ-B) are now compared with the first design in [Ref. 23].

TABLE IV

RESULTS FROM POLE PLACEMENT SEQUENCE(LQ-A)

Move	Q and F obtained during each reassignment			
Q_1	0.10027 0.96194 0.11959 -1.50849	0.96194 9.22847 1.14725 -14.47182	0.11959 1.14725 0.14262 -1.79908	-1.50849 -14.47182 -1.79908 22.69429
F_1	0.00387 -0.24260	0.03151 -2.83522	0.00399 -0.34541	-0.06416 3.11642
Q_2	6.12853 9198 0.51376 -95.14284	-2.99198 1.46070 -0.25082 46.44916	0.51376 -0.25082 0.04307 -7.97589	-95.14285 46.44916 -7.97589 1477.05225
F_2	0.05825 2.27529	-0.02844 -1.11079	0.00490 0.19079	-0.90426 -35.32260
Q_3	80.37563 18.91502 13.64761 218.27521	18.91502 4.45133 3.21173 51.36732	13.64762 3.21173 2.31734 37.06267	218.27522 51.36732 37.06267 592.76758
F_3	1.58772 -8.62241	0.37365 -2.02917	0.26961 -1.46414	4.31182 -23.41618
Q_4	0.15575 0.02839 -2.48989 .19453	0.02839 0.00517 -0.45383 0.03546	-2.48989 -0.45383 39.80322 -3.10981	0.19453 0.03546 -3.10981 0.24297
F_4	-0.38007 -0.06411	-0.06927 -0.01168	6.07574 1.02482	-0.47469 -0.08007
Q_e	86.76016 16.91336 11.79107 121.81841	16.91336 15.14567 3.65433 83.38010	11.79108 3.65433 42.30624 24.17787	121.81841 83.38010 24.17787 2092.75684
F_e	1.26977 -6.65383	0.30745 -5.98686	6.35424 -0.59394	2.86871 -55.70242

$$u(t) = -Fx(t) + h\,\phi_C(t), \qquad h = \begin{bmatrix} 2.8687 \\ -55.7024 \end{bmatrix}$$

TABLE V

RESULTS FROM POLE PLACEMENT SEQUENCE(LQ-B)

Move	Q and F obtained during each reassignment			
Q_1	0.00031 0.05289 0.00589 0.04781	0.05289 9.06491 1.00865 8.19296	0.00589 1.00865 0.11223 0.91163	0.04781 8.19296 0.91163 7.40489
F_1	0.00017 -0.01771	0.02851 -3.03548	0.00317 -0.33776	0.02576 -2.74347
Q_2	0.26982 0.46312 16.33746 10.89106	0.46312 0.79492 28.04222 18.69380	16.33746 28.04224 989.23975 659.45728	10.89106 18.69380 659.45752 439.61426
F_2	0.44607 -0.26207	0.76563 -0.44982	27.00978 -15.86824	18.00517 -10.57837
Q_3	0.16317 -0.35205 6.24519 -9.35123	-0.35205 0.75956 -13.47425 20.17563	6.24519 -13.47424 239.02527 -357.90381	-9.35123 20.17563 -357.90405 535.90698
F_3	0.19939 0.30220	-0.43018 -0.65201	7.63129 11.56628	-11.42668 -17.31876
Q_e	0.43330 0.16396 22.58853 1.58764	0.16396 10.61939 15.57661 47.06238	22.58853 15.57663 1228.37695 302.46509	1.58764 47.06238 302.46509 982.92603
F_e	0.64563 0.02242	0.36396 -4.13731	34.64423 -4.63971	6.60425 -30.64059

$$u(t) = -Fx(t) + h\,\phi_c(t), \qquad h = \begin{bmatrix} 6.60425 \\ -30.64059 \end{bmatrix}$$

Closed loop time response for a step input in roll attitude command (ϕ_c =0.1 rad) for the three designs are shown in Figures 5.14 to 5.16 . It can be seen from the response plots that although all three designs have almost identical closed-loop pole location, the step response for various states differs. The AlphaTech design's response is characterized by the large 'overshoot' in the yaw rate step response. In the Linear Quadratic Design (LQ-A), the 'overshoot' occurs with the roll attitude response rather than the yaw rate response; the yaw rate response is well damped. The LQ-B design appears to be a better design as coupling among modes are small and can not be detected from the time response plot. It is noted that the difference in response for the same set of closed-loop eigenvalue is due to coupling among various modes through their respective eigenvectors. The 'overshoot' in this case is obviously not due to complex conjugate poles as all closed-loop poles in the three designs are on the real axis. The set of final eigenvector for the three designs are tabulated in Table VI . Inter-mode coupling for the yaw rate response in the AlphaTech design and roll attitude response in the LQ design case A can be readily seen from the table. The issue of eigenvector assignment will be discussed in the next section.

TABLE VI

CLOSED-LOOP EIGENVECTORS

	Eigenvalues/Eigenvector			
	-24.79907	-2.12242	-9.72403	-11.9449
Alpha one	-0.39336 0.10767 -0.91306 -0.00029	-0.99976 0.00822 0.01955 -0.00488	-0.48016 -0.32948 -0.81181 0.04306	-0.41854 -0.33640 -0.84283 0.03593
	-2.12499	-9.26645	-12.52066	-25.20990
LQ-A	0.22870 0.18018 -0.95603 -0.03529	-0.15459 -0.98218 -0.01258 0.10614	0.01894 -0.99664 -0.00396 0.07964	-0.12653 0.99118 0.00031 -0.03932
	-25.17894	-9.62708	-12.51644	-2.12630
LQ-B	-0.12652 0.99118 0.00024 -0.03937	-0.13209 -0.98557 -0.02568 0.10267	0.05042 -0.99375 0.06075 0.07886	-0.99977 0.00797 0.01945 -0.00475

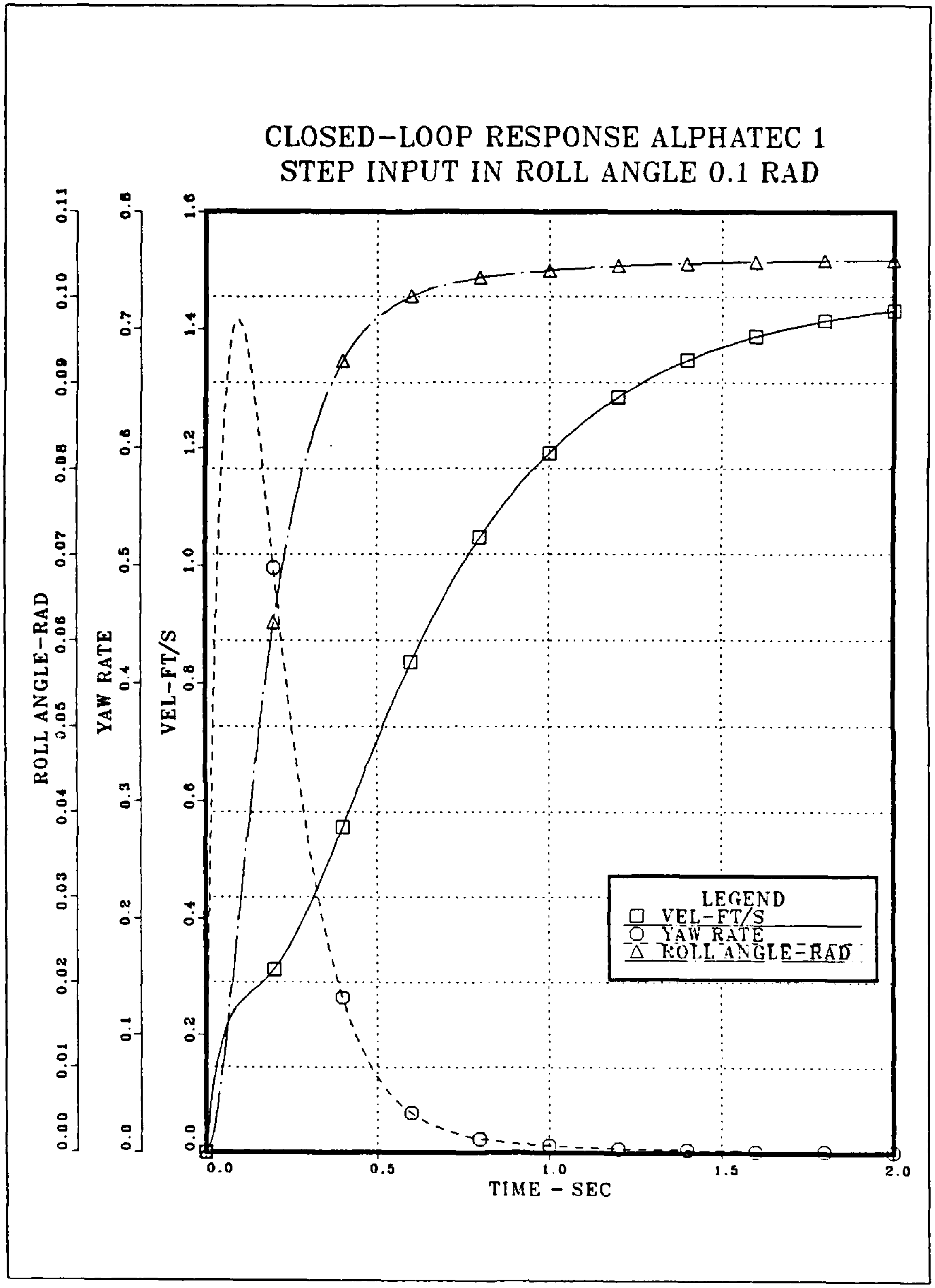


Figure 5.14 Closed-Loop Time Response Plot (AlphaTech 1).

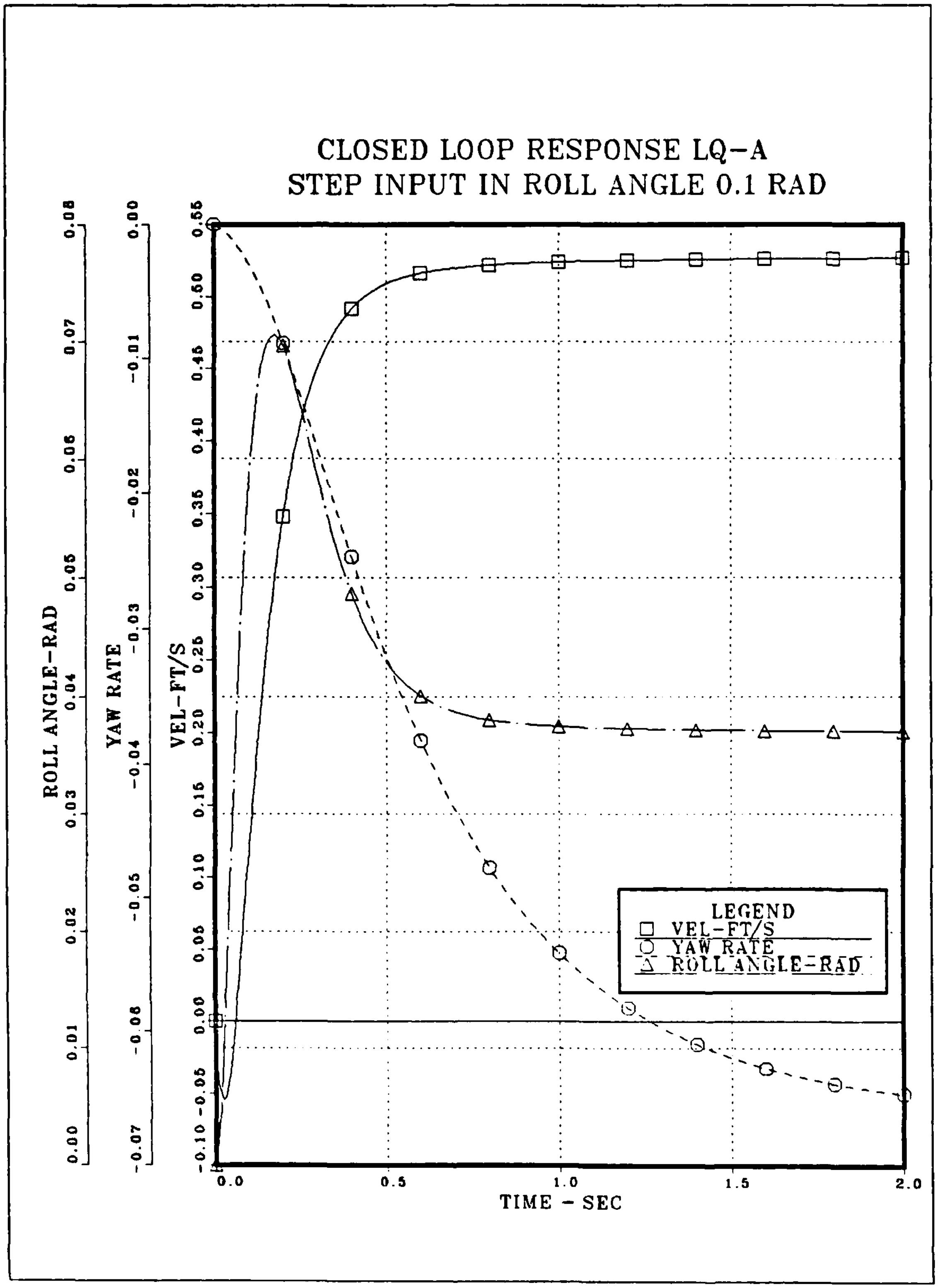


Figure 5.15 Closed-Loop Time Response Plot (LQ-A).

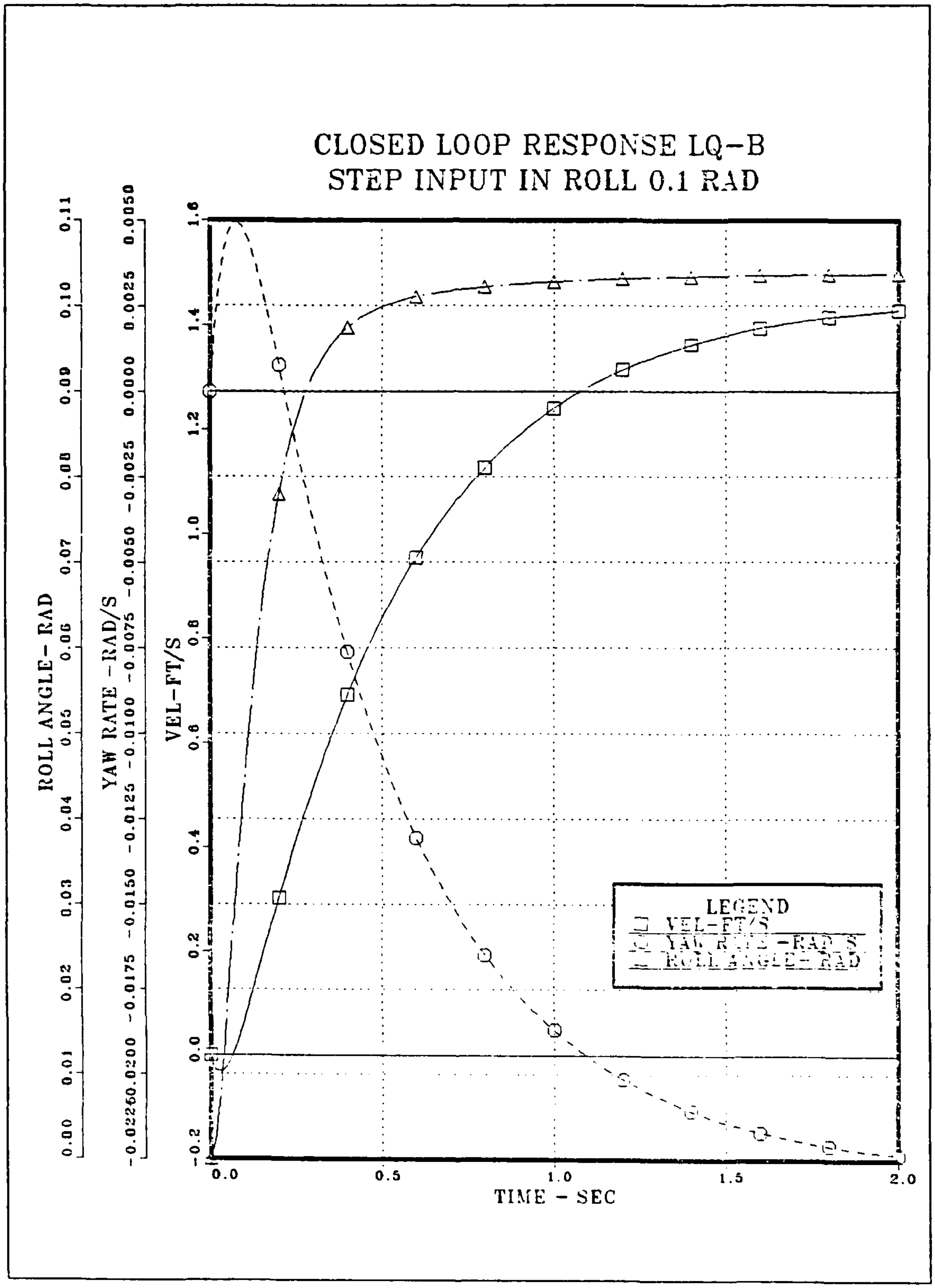


Figure 5.16 Closed-Loop Time Response Plot (LQ-B).

The difference in time response performance for the three designs can also be interpreted in term of the closed-loop pole-zero plots in the frequency domain. The pole-zero plots for the designs, with ϕ_c as the input, are shown in Figures 5.17 to 5.19 . As all pole locations are the same for the three designs, they are indicated only once in each diagram.

In the AlphaTech's design, the zero locations for the various channels clearly indicate the weakness in the design. Most transmissions zeros are located away from the poles locations. The system is therefore strongly coupled to its external environment. A good example is the lightly "damped" pair of zero at (-1.05,±7.31j) for the v-ϕ_c channels. It is in fact these undesired zeros that reduce the overall robustness of the system. The mechanism of robustness improvement in the LQ design can also be seen in the pole-zero plots in Figures 5.18 and 5.19 The built-in robustness in the LQ design causes the zeros at various channels to move to locations where their transmission properties can be canceled by the closed-loop poles. An excellent example is in the v- ϕ_c channel, where the zeros at -2, -7, and the mirror image of +24.0 are fairly close to the closed-loop pole locations at -2.12, -9.26 and -25.12. In the case where the zero from LQ design are not close to the closed-loop poles (r- ϕ_c channel for LQ-A) the zeros are "well damped" and their transmission properties can be neglected.

Robustness properties of the three designs presented here can also be analyzed from the open-loop Bode plots. The open-loop transfer function gain plots for channel 1-1, 1-2, 2-1 and 2-2 for the three designs are shown in Figures 5.20 to 5.24 Cross coupling problem for the AlphaTech design is clearly indicated by the relatively high gain of

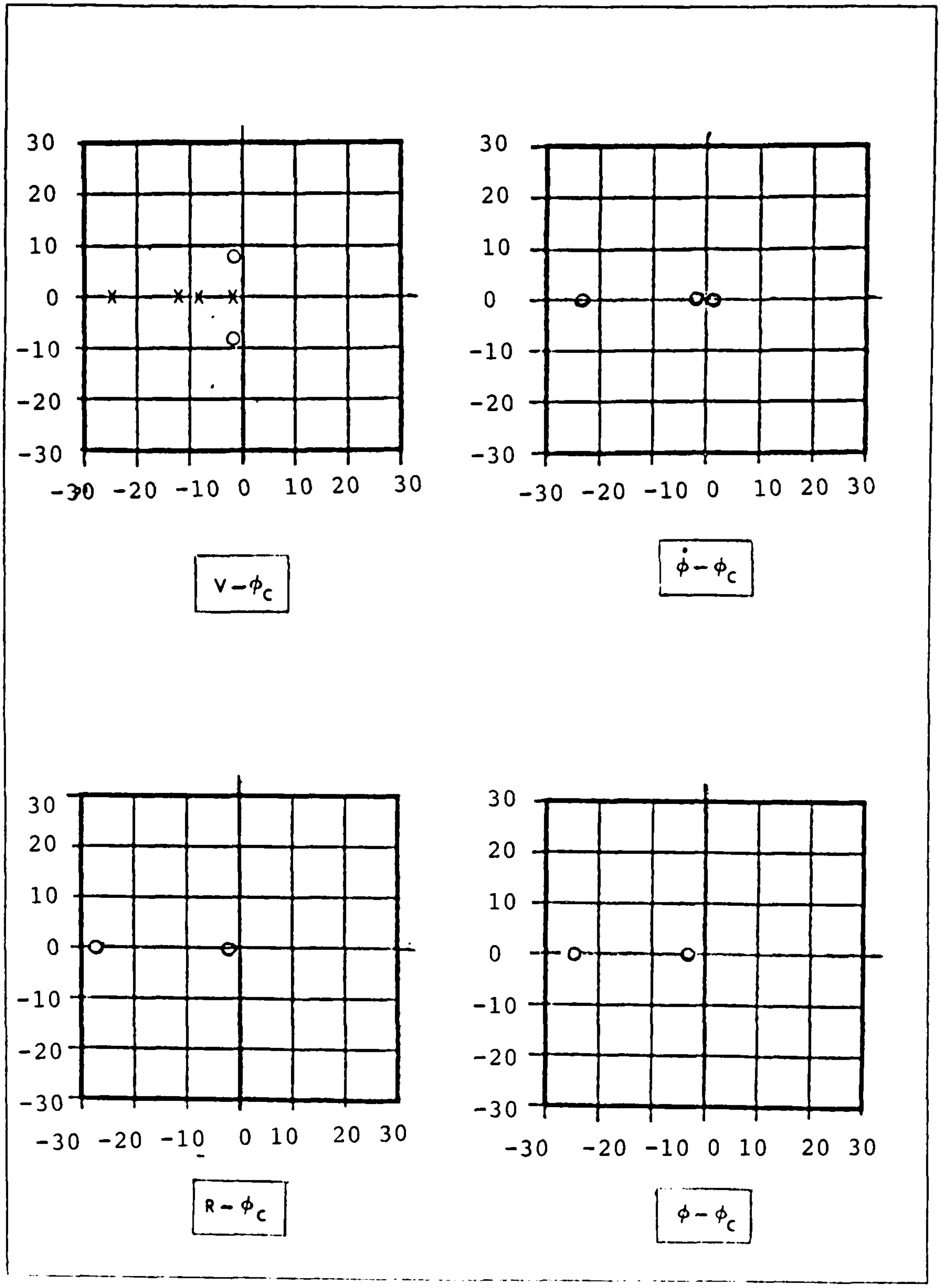


Figure 5.17 Closed-Loop Pole-Zero Plots (AlphaTech 1).

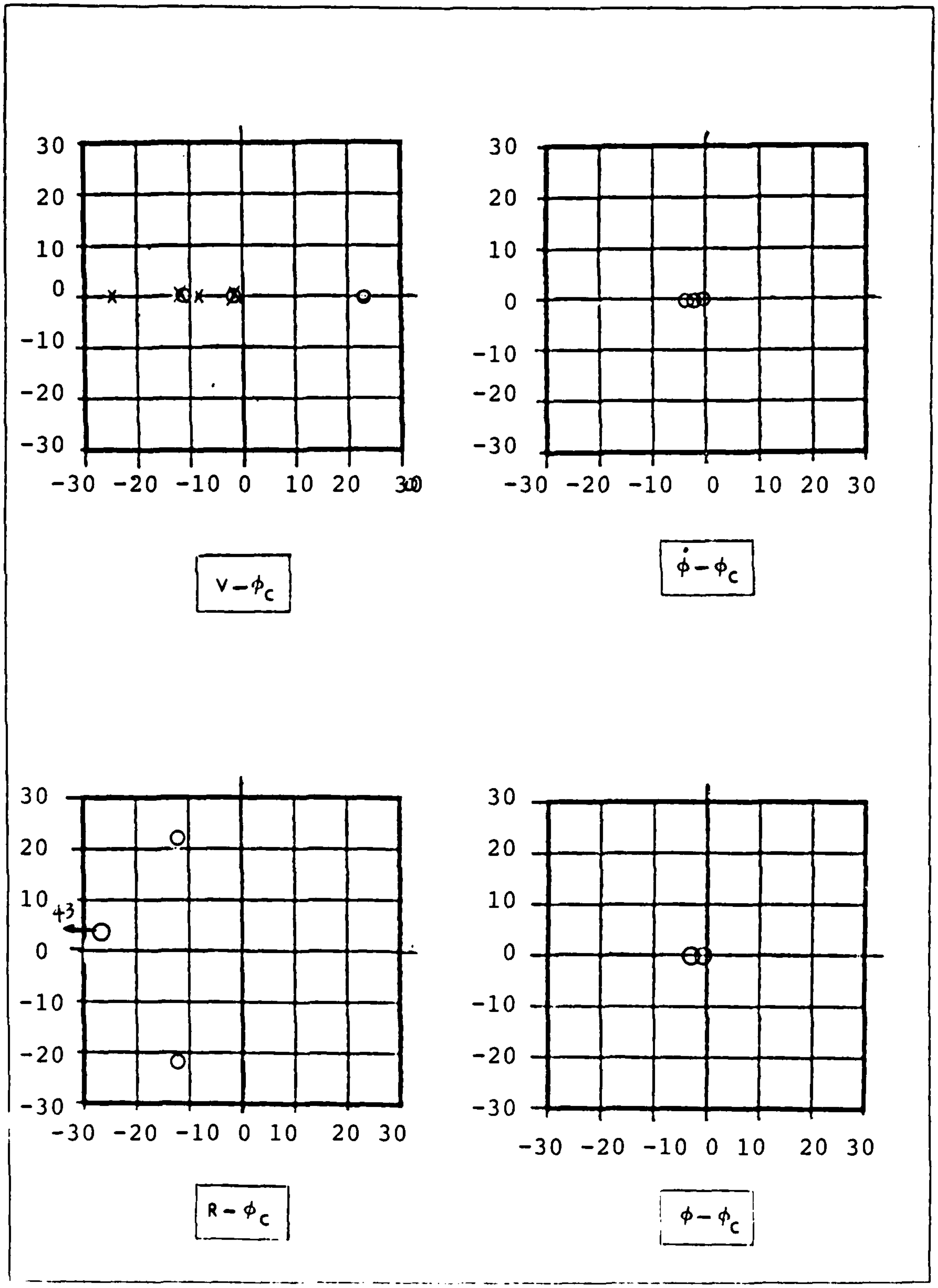


Figure 5.18 Closed-Loop Pole-Zero Plots (LQ-A).

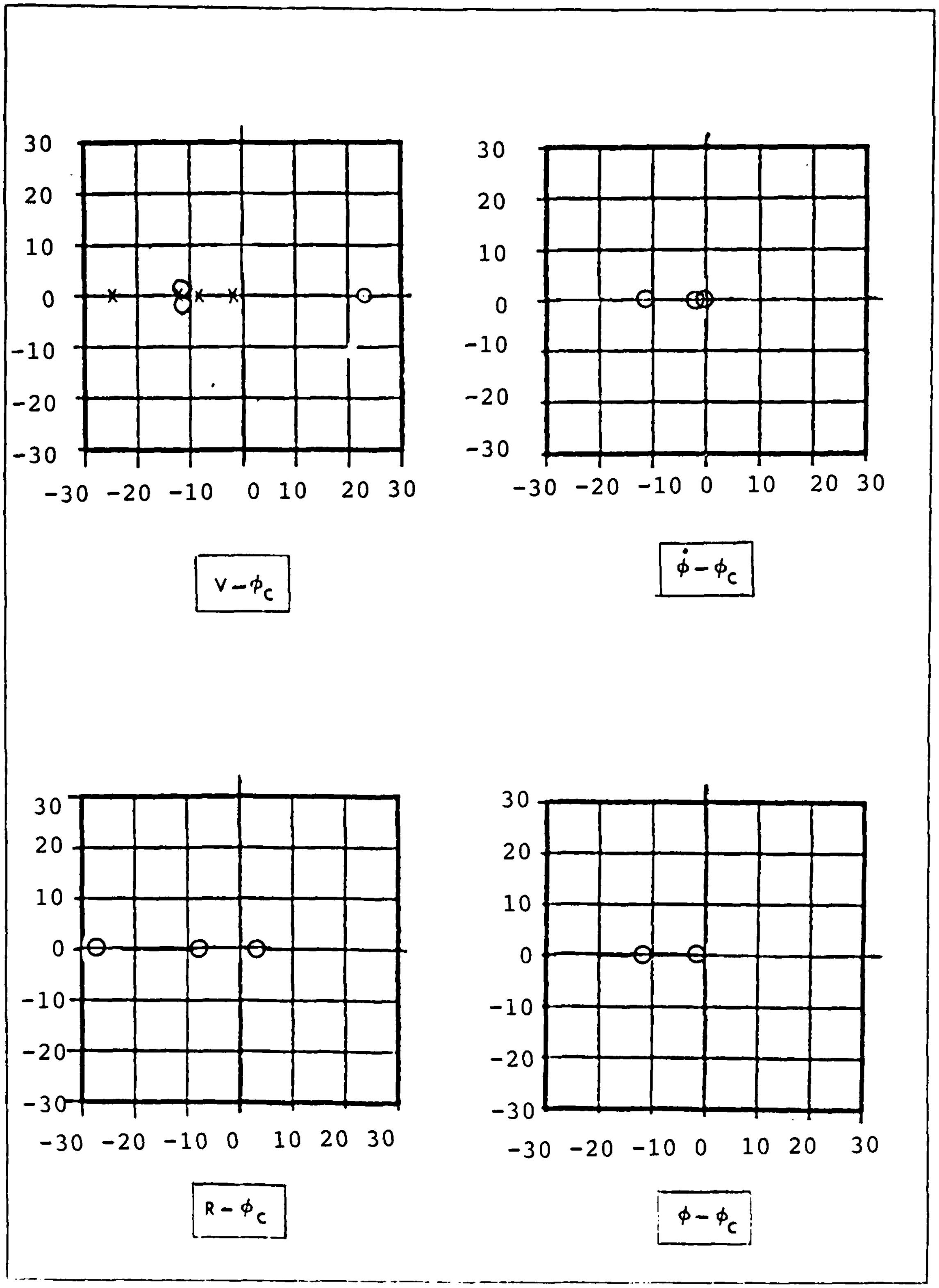


Figure 5.19 Closed-Loop Pole-Zero Plots (LQ-B).

the channel from input 2 to input 1 (2-1). Both LQ designs presented here reduced the gain in this channel by more than 40 dB in the frequencies of interest. The bandwidth is reduced from above 100 rad/s to about 8 rad/s. Unlike the simple (2x2) system where there was little difference between the direct channels of the two designs, gain adjustment is observed in all channels. For example, LQ designs reduce the gain in channel 1-1 but increase the gain in channel 2-2. There is also a slight increase in the coupling channel 1-2. The overall effect is that of gain balancing, gains in channels that are affected by cross-coupling perturbation are lowered together with some adjustments in other channels. Different reassignment sequence results in different adjustments. The designer has to choose a set of gain curve depending on the particular requirement. The relationship between the open-loop Bode plots and the zeros of the system is also evident from these diagrams. The low frequency resonance (or 'peak') in the gain vs frequency plot for the AlphaTech design correspond to the undesirable zero mentioned earlier. In the LQ designs, these low frequency resonances are absent because of the more desirable zero locations. It is also interesting to note that the LQ-B gain curves fall nicely in between the other two designs. This fact, together with its better overall low frequency gain characteristic, may account for the better time response behavior of the LQ-B design.

As a final comparison, the closed-loop frequency response plots of the various channel for the three designs are shown in Figures 5.25 to 5.32 . At high frequency, all the gain plots approach the -20dB/decade slope as all eigenvalues are on the real axis. The AlphaTech design has a near 0db gain for frequency up to about 2 rad/s. It appears

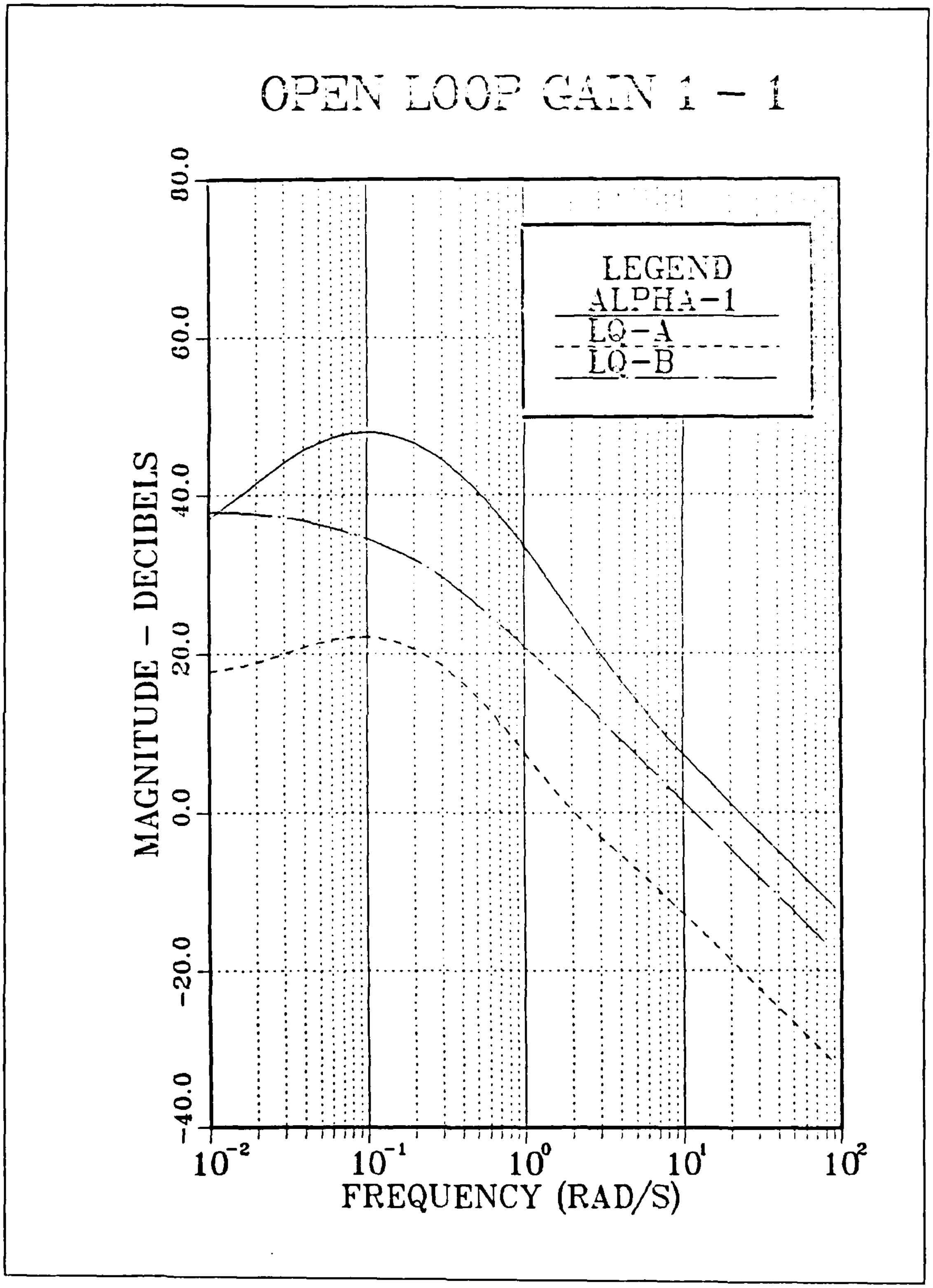


Figure 5.20 Bode Plots Comparison- Input 1-1.

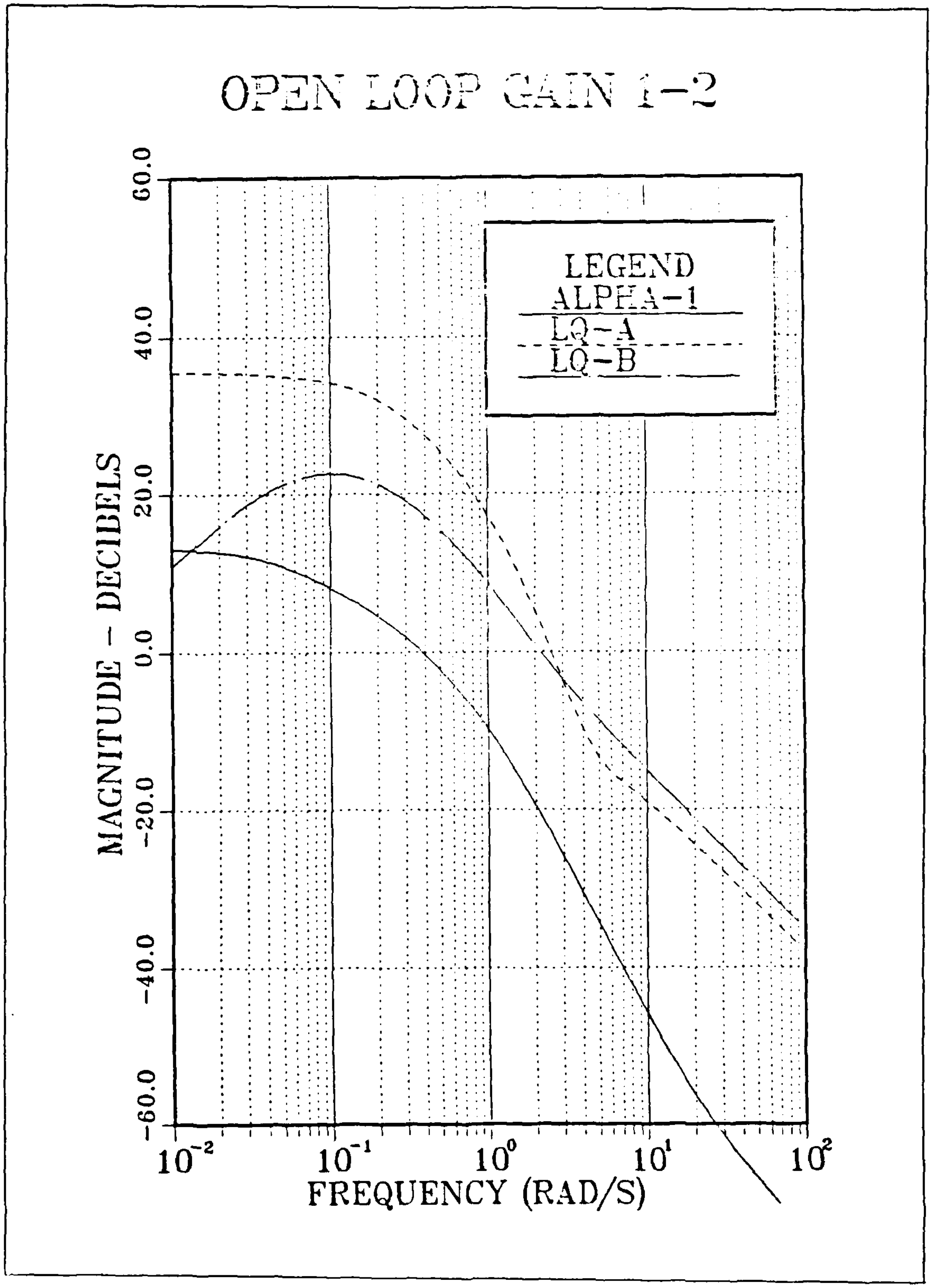


Figure 5.21 Bode Plots Comparison - Input 1-2.

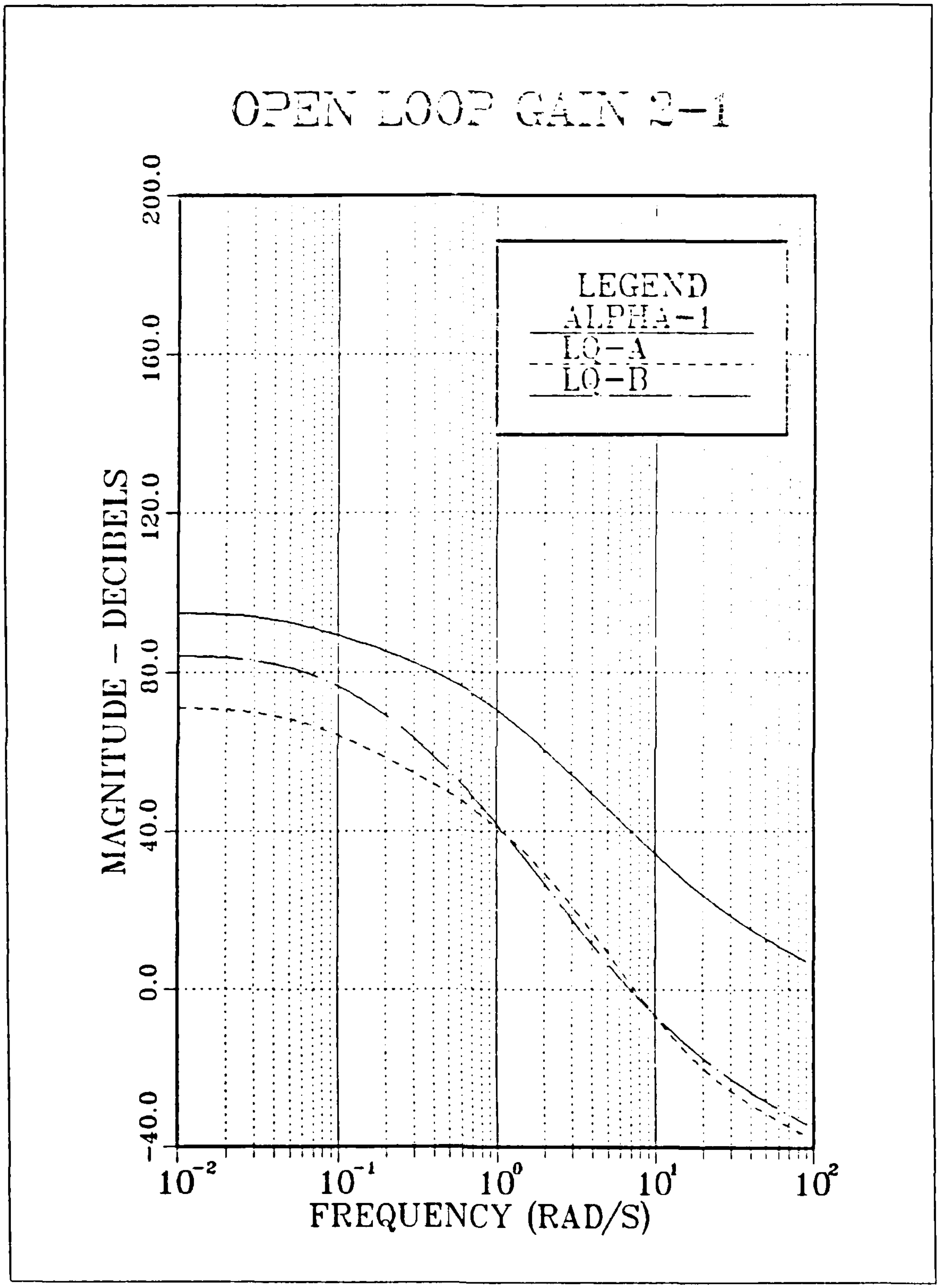


Figure 5.22 Bode Plots Comparison - Input 2-1.

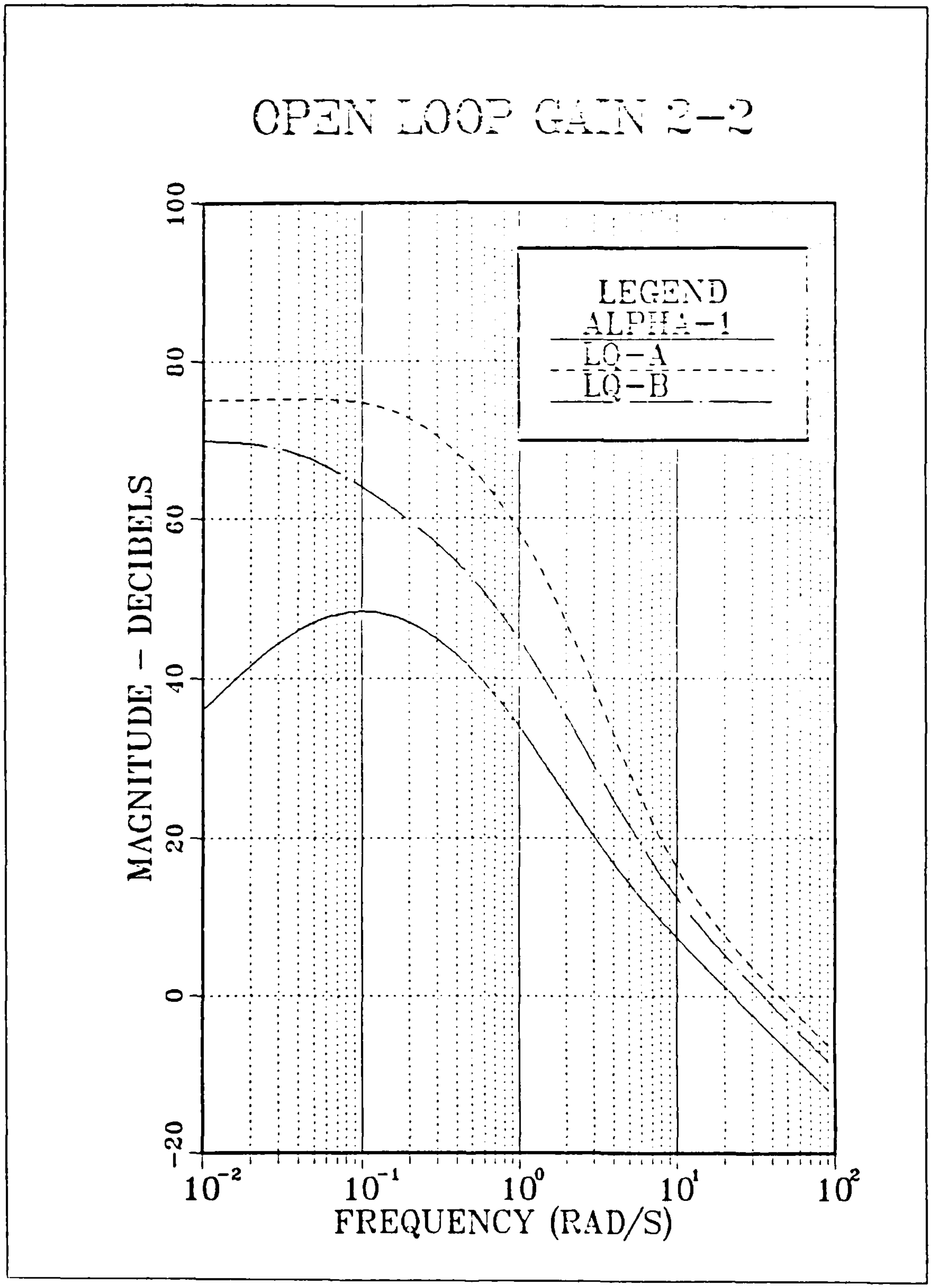


Figure 5.23 Bode Plots Comparison - Input 2-2.

to be an acceptable design but was shown to have undesired transmission zeros in other channels; this can also be seen from the closed-loop plot for the $r\text{-}\phi_c$ channel in Figure 5.31 . Design LQ-A, obtained from the reassignment sequence A, is characterized by very low DC gain (-9dB) and 'peak' in frequencies near 10 rad/s. These have been shown, both in the time response and pole-zero discussions that it will have undesired effect on the pilot's control. Design LQ-B, obtained from the second reassignment sequence appears to be the best compromise. All channels (Figures 5.30 to 5.32) have flat low frequency characteristic and coupling between modes are almost absent.

C. DISCUSSION AND CONCLUDING REMARKS

A new computer aided design procedure for the multivariable linear time-invariant system using Linear Quadratic Pole Placement formulation is presented. The two design examples presented above have served to demonstrate the complexity involved in a multivariable design. The main problem lies in the fact that the solution of a MIMO problem is in general non-unique. This was shown in the helicopter problem where different approaches result in different designs, although the closed-loop eigenvalues for all designs were the same. It was also shown that the extra degrees of freedom in MIMO system design can be accounted for by analyzing the singular value plots, transmission zero movements and closed-loop eigenvectors of the designs. It becomes apparent that the success of any multivariable design methodology hinges on how to make use of these extra degrees of freedom available in MIMO system. It was also shown in the above design examples that the Linear Quadratic Pole Placement procedures possess such quality. Some unique properties of the method presented here are as follows;

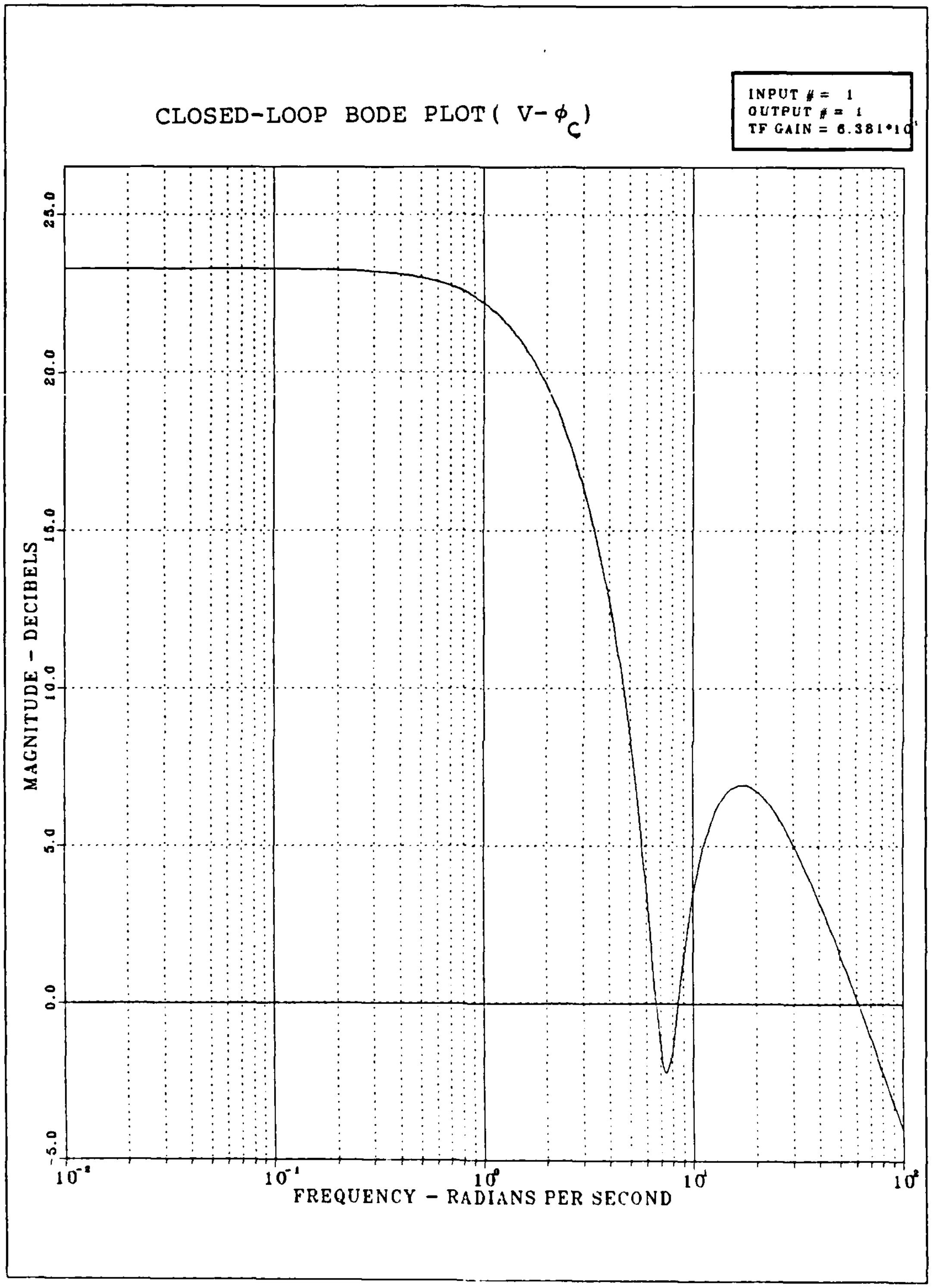


Figure 5.24 Closed-Loop Bode Plot- AlphaTech 1-1.

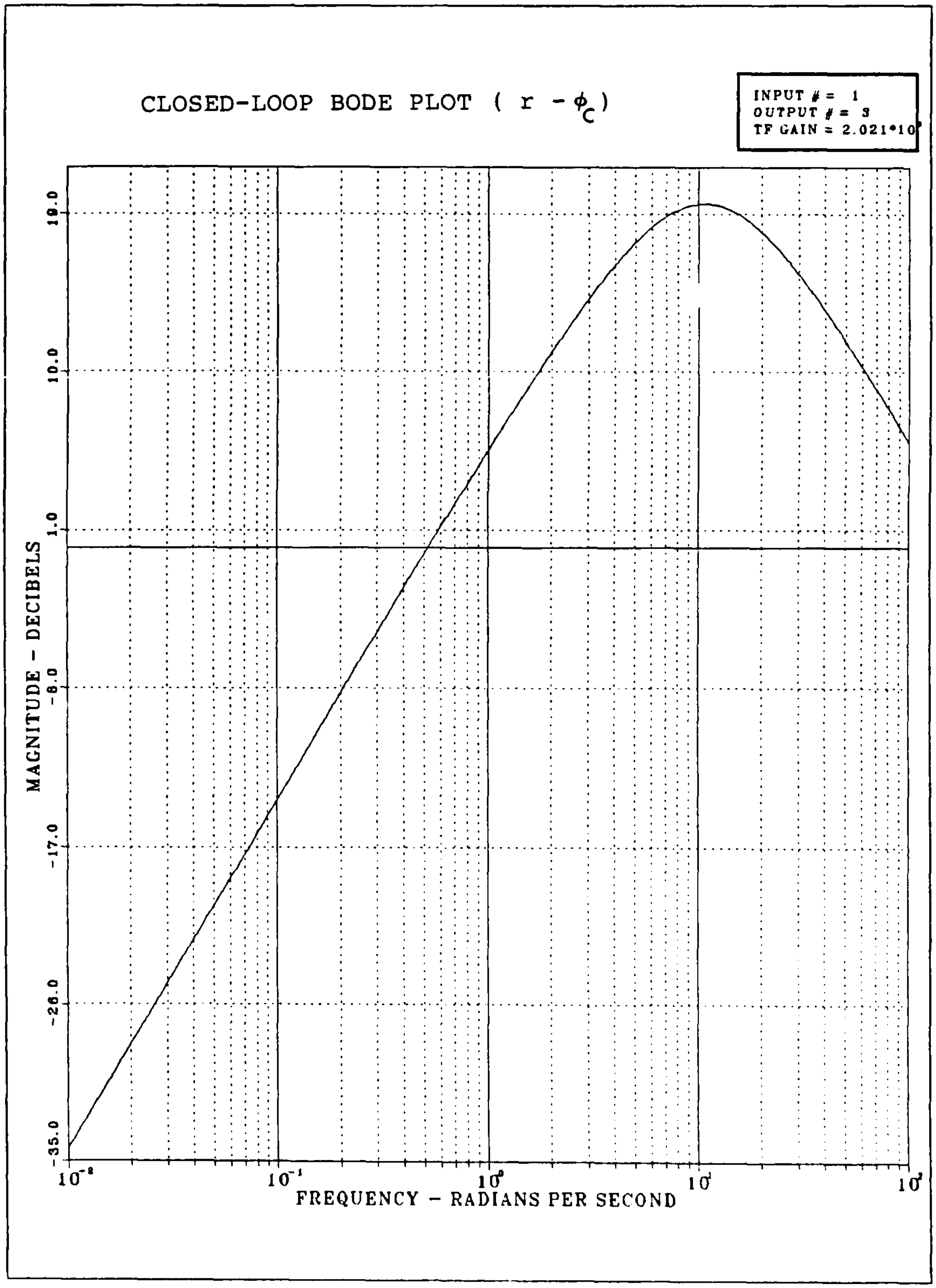


Figure 5.25 Closed-Loop Bode Plot- AlphaTech 1-3.

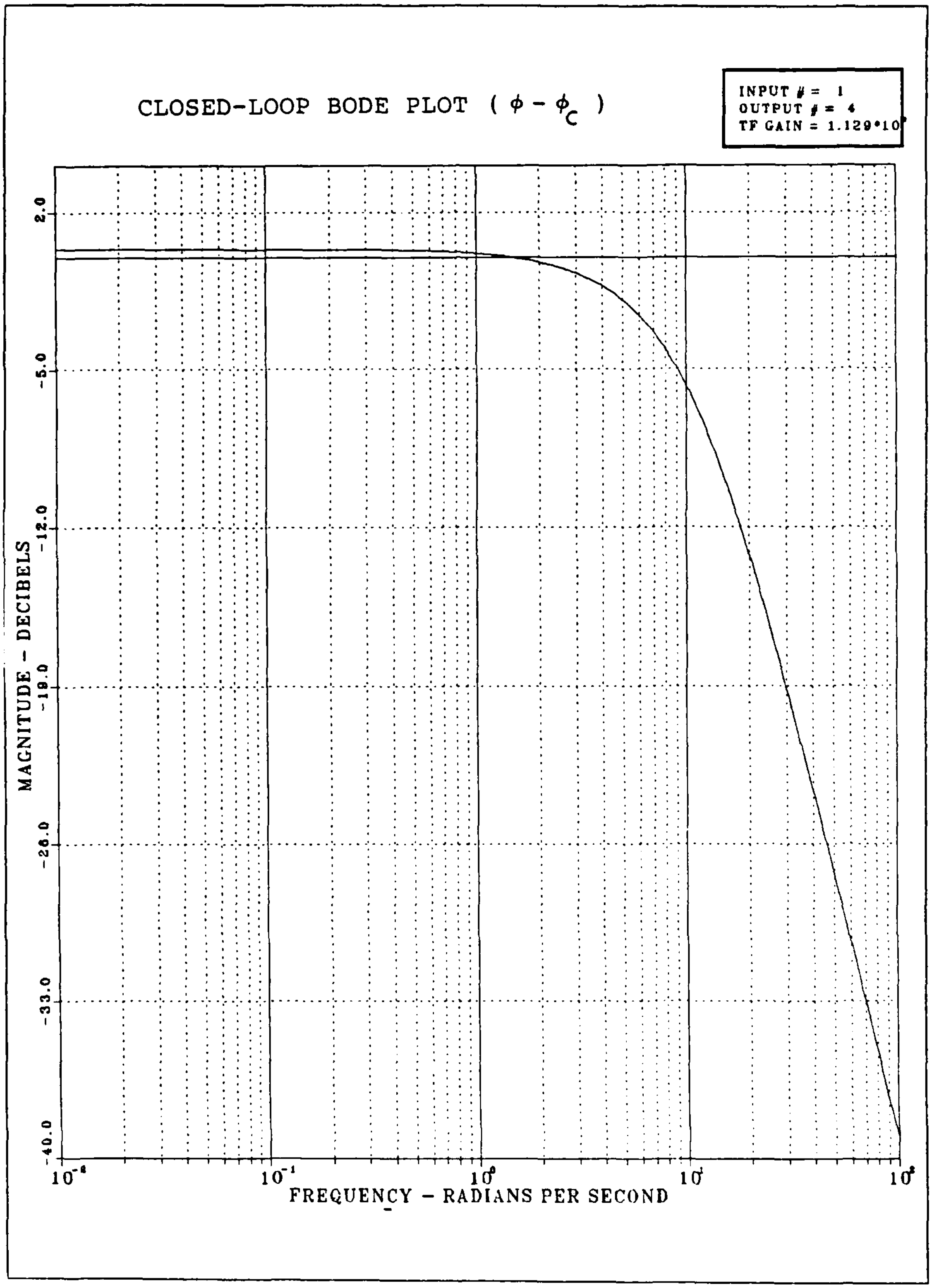


Figure 5.26 Closed-Loop Bode Plot- AlphaTech 1-4.

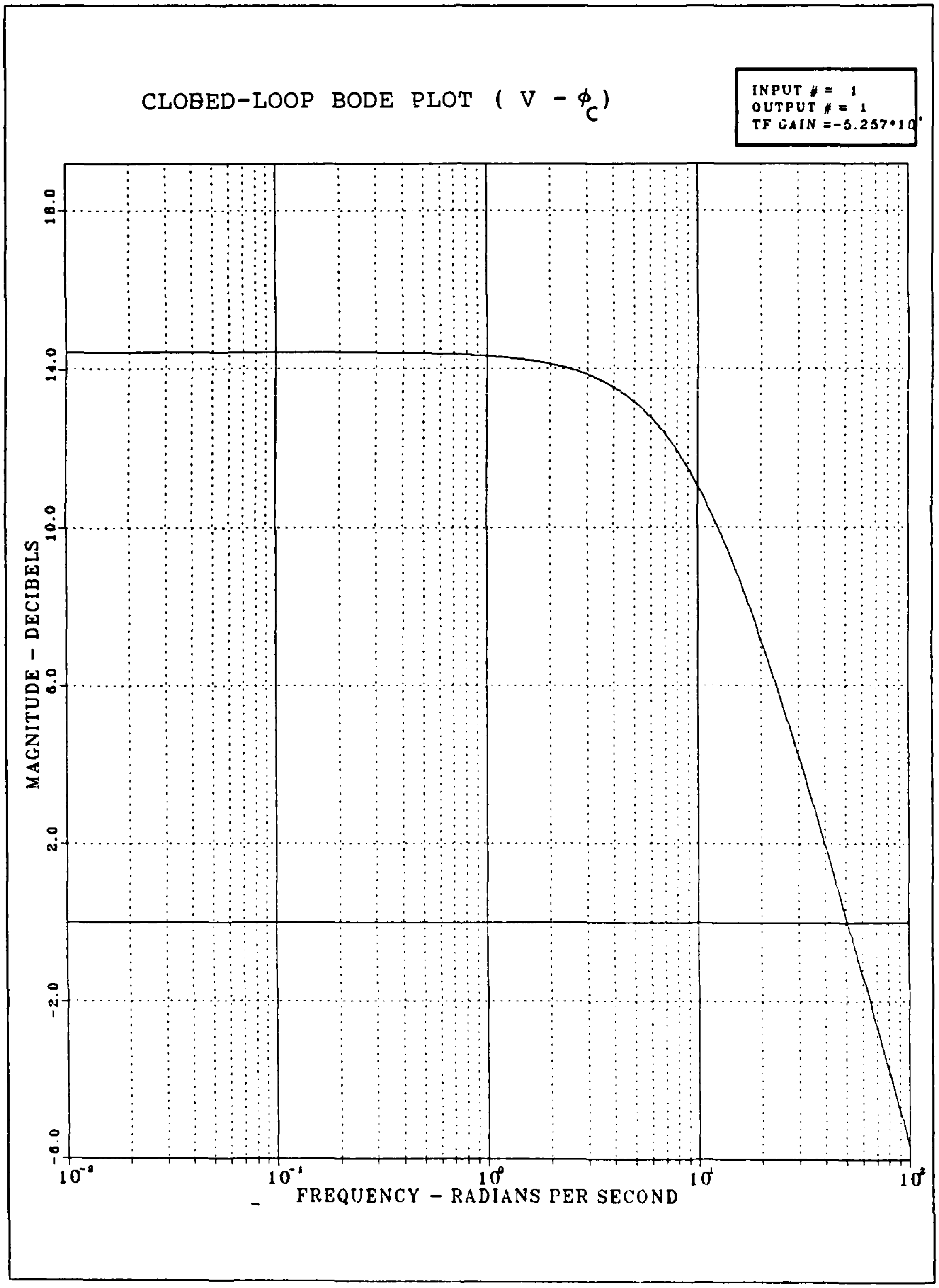


Figure 5.27 Closed-Loop Bode Plot- LQ-A 1-1.

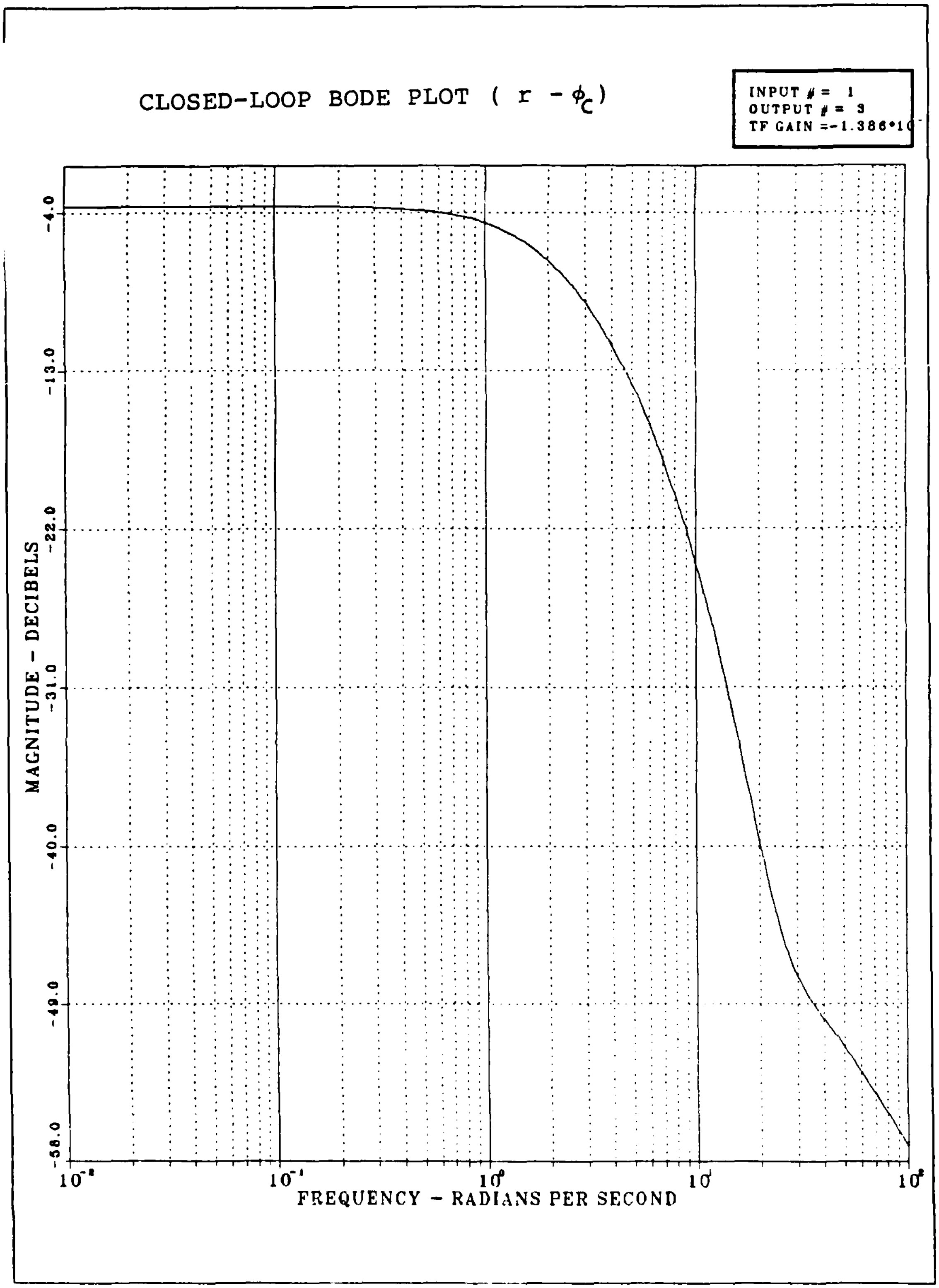


Figure 5.28 Closed-Loop Bode Plot- LQ-A 1-3.

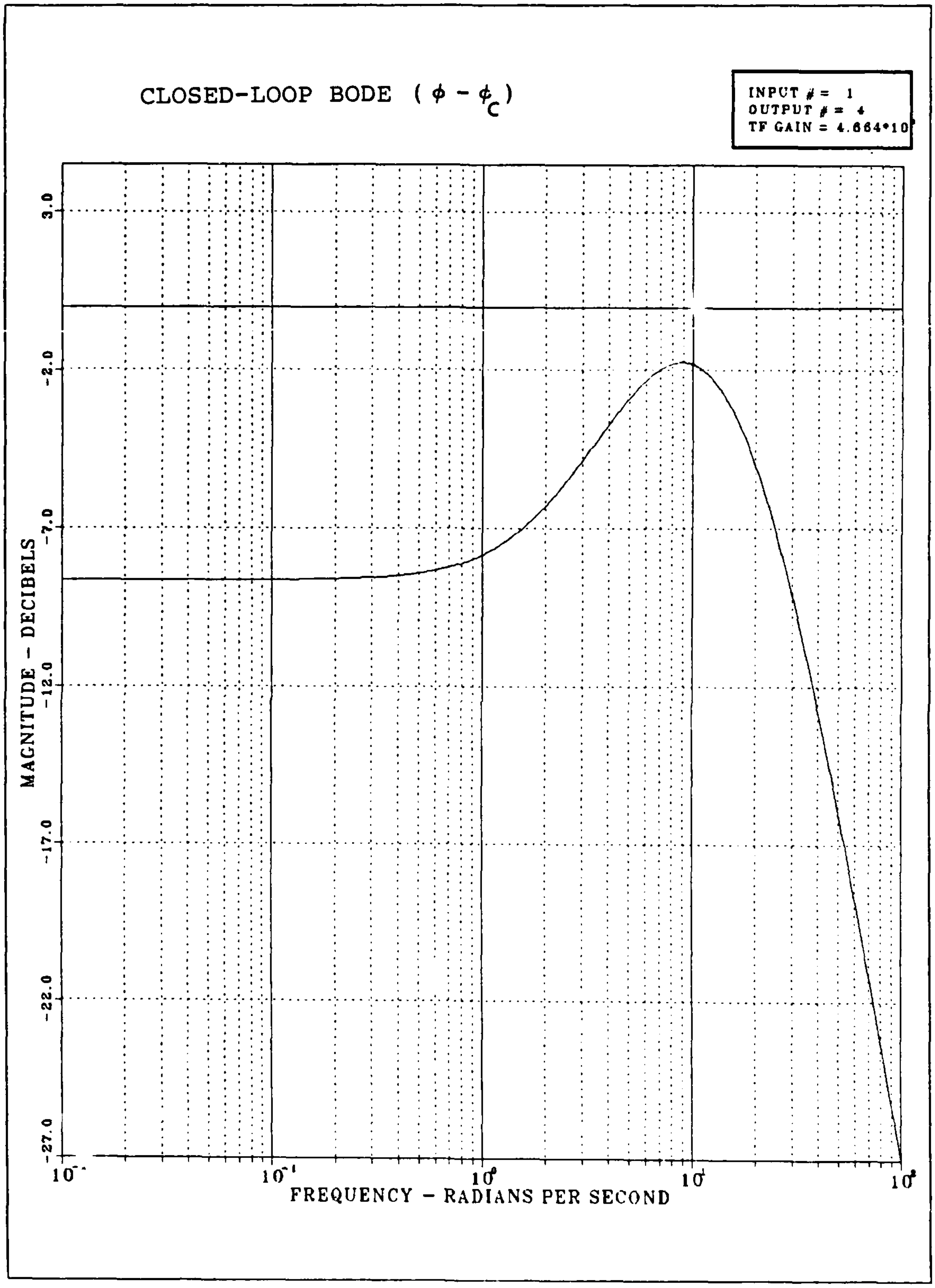


Figure 5.29 Closed-Loop Bode Plot- LQ-A 1-4.

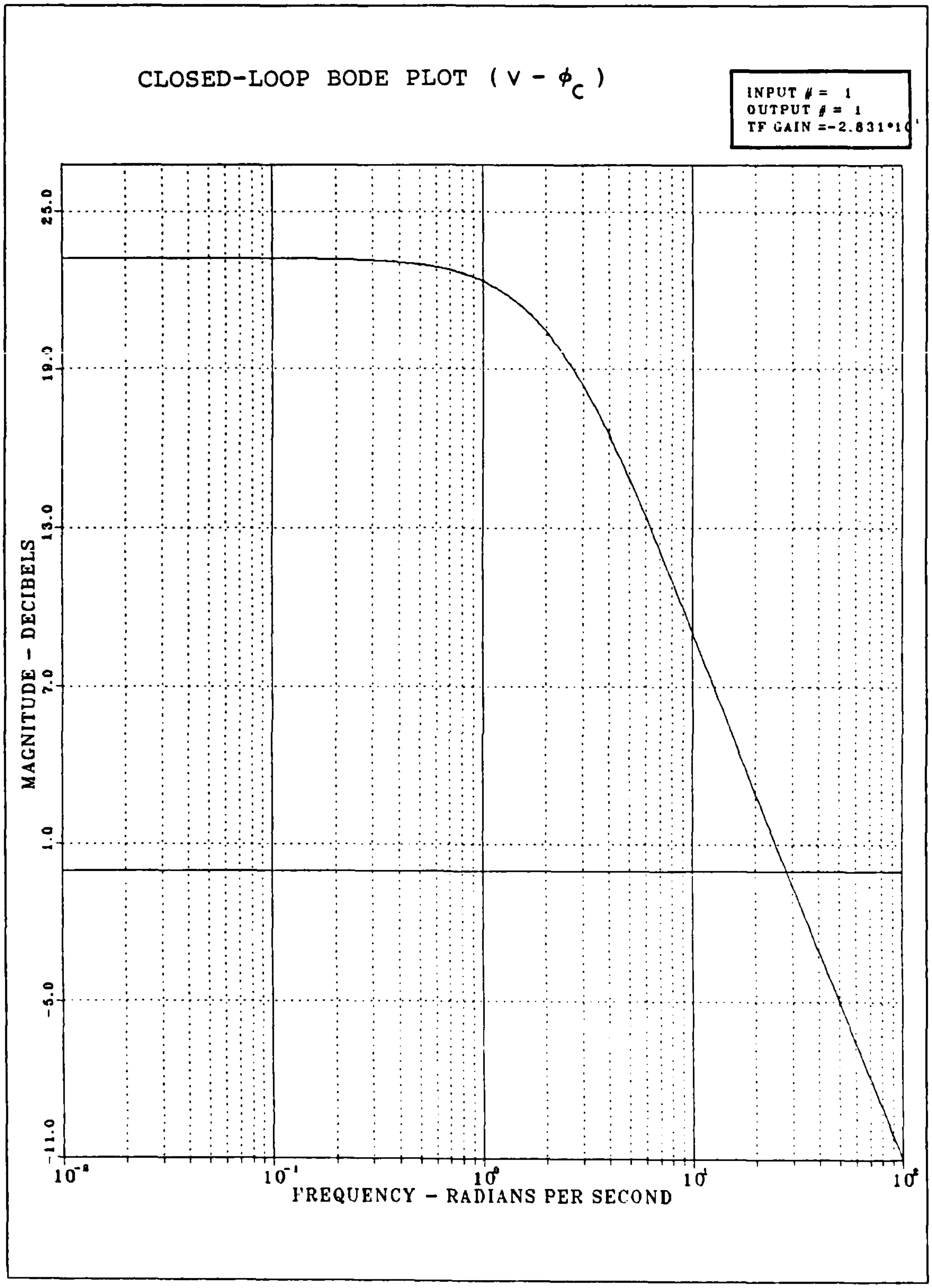


Figure 5.30 Closed-Loop Bode Plot- LQ-B 1-1.

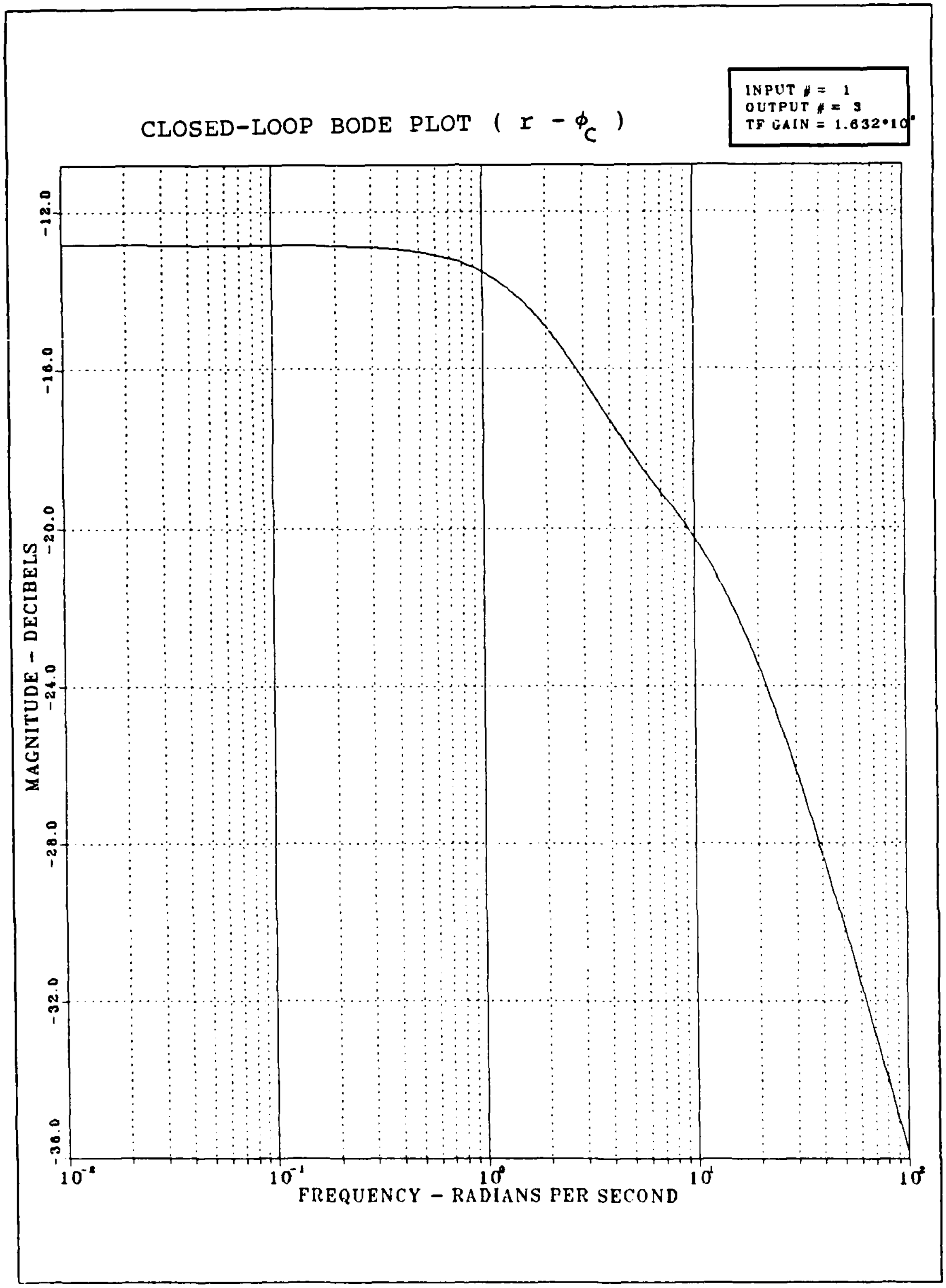


Figure 5.31 Closed-Loop Bode Plot- LQ-B 1-3.

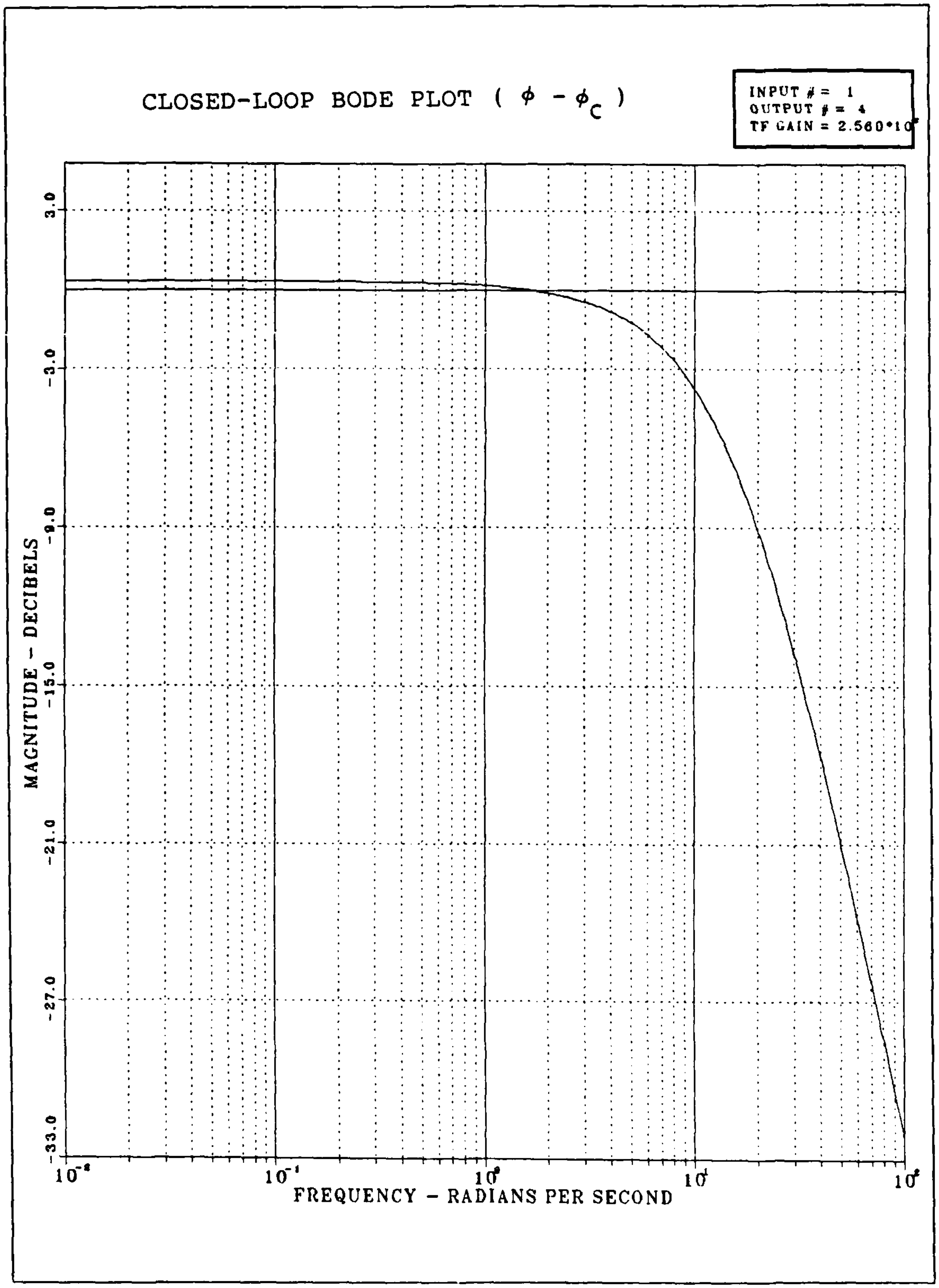


Figure 5.32 Closed-Loop Bode Plot- LQ-B 1-4.

1. It guarantees stability automatically.
2. It has partly overcome the main difficulty in applying optimal control theory to practical problems; i.e. that of selecting suitable performance indices (Q and R matrices). The designer can now choose to compute Q (if he knows somethings about R) or vice versa or compute both Q and R if both of them are unknown.
3. It has built-in robustness to model's error and perturbation.
4. Reduced-order type of problem formulation is possible as the procedure can reassign poles one at a time without affecting others.

On the other hand, the design procedure developed here is not without some shortcomings. Three main problem areas are described below, one of which can be overcome by additional programming efforts while the others are still active research areas pursued by many researchers.

1. Eigenvectors Assignments: It was shown in the design examples presented above that while the overall speed of response of the closed-loop system is determined by its eigenvalues, the 'shape' of the transient response depends on the closed-loop eigenvectors. The problem of eigenvectors assignments was first formulated in [Ref. 16]. Since then many eigenvectors assignment algorithms have been developed [Refs. 17,24] for use in multivariable design. In principle, the eigenvectors assignment routine can be incorporated into the coordinate transformation portion of the OPTPP program (as indicated in Figure 4.1 in Chapter 4). In essence, a new similarity transformation different from M in Equation 4.6 is computed once the eigenvalues and eigenvectors are specified. As most eigenvector algorithms available

at present are iterative in nature and their inclusion requires major programing efforts, it is recommanded for future work.

2. Perturbation or Model's Error Structure: The guaranteed robustness obtained from the LQ formulation given by equations 3.4 and 3.5 ensures that the perturbation or model's error (L(jw)) is sufficiently small so that the closed-loop system remains stable. However, the above only applies to simple model's error structure where both L(jw) and R are diagonal matrices. There may be cases when the above equations do not hold and hence the design becomes very conservative. An example of this nature is shown in [Ref. 19]. An ad hoc solution is to use non-diagonal control weighting matrix as mentioned in Chapter 4. Two designs for the helicopter problem are obtained using non-diagonal R. Their results in terms of singular value plots, Bode plots etc are compared with other designs in Appendix A
3. Reassignment Sequence: As shown in the last section, different reassignment sequences result in different designs. At present there are no known methods for determining the best reassignment sequences.

VI. CONCLUSIONS

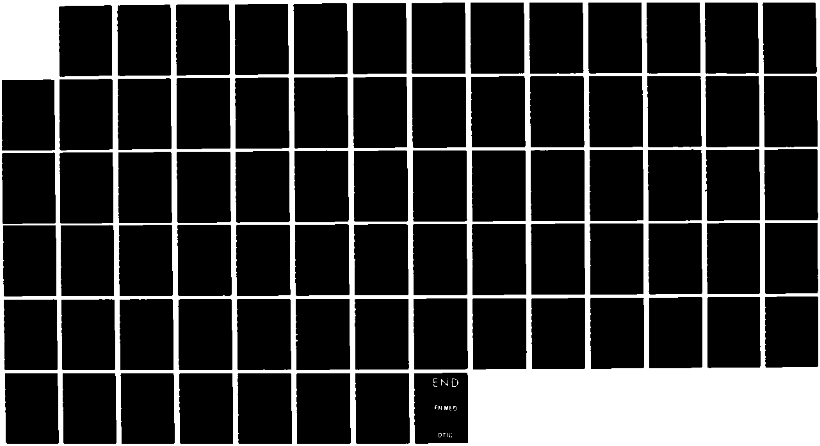
It was demonstrated that the general pole assignment problem in multivariable state feedback control system design can be formulated using the Linear Quadratic Control approach. This method of formulation is effective for two main reasons; First, the extra degrees of freedom available in a multivariable system structure is utilized to produce designs that are robust to perturbations in the system and gain matrices. Secondly, the classical difficulty of selecting suitable performance index in optimal control formulation was partly overcome, as designers now have the flexibility of specifying only Q (the state weighting matrix) or R (the control weighting matrix) or both Q and R as the design parameters to be varied. In other words, knowledge of the performance index which ideally should come from physical argument is used to the best of designer's advantage. In addition, the structure of the present formulation is such that eigenvector assignment can be further incorporated into the procedure. The above properties, when combined with the reduced-order formulation capability, have been shown to be very versatile and have important impact on the performance of the resulting design.

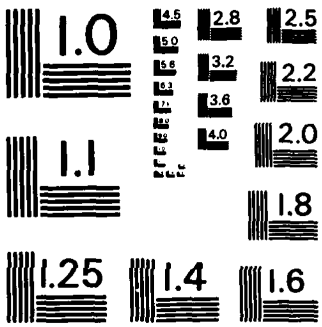
The optimal pole placement (OPTPP) program developed here is combined with other well established routines to form a computer aided design and synthesis package. Together with the design procedure and philosophy presented here, it provides the control system designer an excellent and viable tool to solve complex multivariable problems.

The procedure was applied to practical test examples, and numerical results were presented and discussed. Results indicated that all controllers obtained from the formulation given here were stable and robust. Introducing perturbation in the system matrices leaded only to small errors in the assigned poles. The main shortcoming of the design procedure is the ad hoc nature in which the poles are reassigned. More research is required to develop a systematic way of assigning poles and its eigenvectors thus allowing the designer to optimaly shape the response of the system.

AD-A162 269 AUTOMATED POLE PLACEMENT ALGORITHM FOR MULTIVARIABLE OPTIMAL CONTROL SYNTHESIS(U) NAVAL POSTGRADUATE SCHOOL MONTEREY CA W K CHOW SEP 85 2/2

UNCLASSIFIED F/G 12/1 NL




1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
6.3

APPENDIX A
NON-DIAGONAL R DESIGNS

The two design examples presented in Chapter 5 were based on diagonal control weighting matrix (i.e. R= I). It is now illustrated that design using non-diagonal R will provide yet another degrees of freedom available to the designer. This type of formulation is especially useful when the model's or error structure is known. For multiplicative type of perturbation, the effect of control weighting matrix on the system stability has been explored in [Ref. 12]. In essence, the selection of the R matrix determines the coordinate frame in which the sensitivity assessment is to be made. This can be readily seen from equation 3.1, the general form in which R is non-diagonal is given below;

$$\bar{\sigma}\,[R^{1/2}L^{-1}(s)R^{-1/2} - I] < \underline{\sigma}\,[I + G(s)] \qquad \text{(eqn A.1)}$$

In general, the sensitivity of the system to perturbation in a particular L(s)'s direction can be reduced by making $\bar{\sigma}\,[R^{1/2}L^{-1}(s)R^{-1/2} - I]$ small. This can be done simply by choosing an appropriate R. The main problem with this kind of approach is that L(s) must be known precisly for all frequency for the computation of $\bar{\sigma}\,[R^{1/2}L^{-1}(s)R^{-1/2} - I]$. In addition, the worst case direction of L(s) must be known otherwise the resulting design may be too conservative.

The helicopter problem presented in Chapter 5 is now analyzed using the non-diagonal control weighting matrix. Two designs are obtained as follows;

Design one (LQ-C):

$$R = \begin{bmatrix} 1.0 & 0.0 \\ 0.0 & 2.0 \end{bmatrix}$$

Design two (LQ-D):

$$R = \begin{bmatrix} 1.0 & 0.5 \\ 0.5 & 1.0 \end{bmatrix}$$

Design LQ-C is obtained using an assignment sequence similar to LQ-A with no off-diagonal element in R. Design LQ-D is obtained using placement sequence for design LQ-B but has off-diagonal element in R. Q and F obtained during each move and the final Q_e and F_e are tabulated in Tables VII and VIII. Singular value plots and the open-loop Bode Plots are compared with other designs in Figures A.1 to A.5

Both design has singular value greater than one for all frequency. It is interesting to note that similar reassignment sequence produce similar singular value plots. This can be readily seen by comparing LQ-A with LQ-C and LQ-B with LQ-D in the figures. The effect of using non-diagonal R merely changes the shape of the singular value plots. For the cases presented here, both non-diagonal R designs are slightly less conservative (having lower singular value).

As in the singular value plots, the shape of the open-loop Bode plots (with feedback) are closely related to the reassignment sequence.(compare Bode plots of LQ-A with LQ-C and LQ-B with LQ-D in Figures A.2 to A.5 . Results from pole-zero maps also indicate similar trends.

In summary, it is demonstrated that designs using non-diagonal control weighting matrix (including possibly off-diagonal elements) provide yet another means of 'fine tuning' the design. This capability, such as using the off-diagonal elements directly in design, is unique to the present formulation. Robustness between the upper and lower crossfeed (of L(s)) can be controlled by adjusting the relative weighting of the upper and lower (or off-diagonal) elements of the control weighting matrix. It must be emphasized that this kind of fine tuning is only possible for a class of rather well-defined structure of L. In practices, other constraints, such as the energy of the control input, the conditioning of R etc must be considered over the range of frequencies. The key issue of model's error structure and how it can be used in multivariable control system design is currently being investigated by many researchers.

TABLE VII
RESULTS FROM POLE PLACEMENT SEQUENCE (LQ-C)

Move	Q and F obtained during each reassignment			
Q_1	0.00062 0.10604 0.01180 0.09584	0.10604 18.17288 2.02209 16.42485	0.01180 2.02209 0.22500 1.82758	0.09584 16.42484 1.82758 14.84496
F_1	0.00033 -0.01773	0.05708 -3.03892	0.00635 -0.33814	0.05159 -2.74657
Q_2	4.17258 -3.78394 0.13594 -108.79944	-3.78394 3.43150 -0.12328 98.66574	0.13594 -0.12328 0.00443 -3.54455	-108.79945 98.66574 -3.54455 2836.93335
F_2	0.02701 1.21663	-0.02449 -1.10329	0.00090 0.03967	-0.70429 -31.72316
Q_3	154.23135 36.25874 26.52100 418.56201	36.25874 8.52419 6.23491 98.40114	26.52098 6.23491 4.56044 71.97421	418.56201 98.40111 71.97421 1135.91846
F_3	2.80842 -7.58133	0.66025 -1.78234	0.48291 -1.30355	7.62180 -20.57501
Q_4	0.44288 0.09189 -4.22665 0.82784	0.09189 0.01906 -0.87692 0.17176	-4.22665 -0.87692 40.33727 -7.90052	0.82784 0.17176 -7.90052 1.54741
F_4	-0.63449 -0.09950	-0.13165 -0.02064	6.05549 0.94961	-1.18605 -0.18599
F_e	2.20127 -6.48193	0.56119 -5.94519	6.54565 -0.65241	5.78305 -55.23073
Q_e	158.84741 32.67273 22.44208 310.68604	32.67273 30.14761 7.25680 213.66348	22.44206 7.25680 45.12712 62.35670	310.68604 213.66344 62.35670 3989.24390

$$u(t) = -Fx(t) + h\,\phi_C(t), \qquad h = \begin{bmatrix} 5.7839 \\ -55.23 \end{bmatrix}$$

TABLE VIII

RESULTS FROM POLE PLACEMENT SEQUENCE (LQ-D)

Move	Q and F obtained during each reassignment			
Q_1	0.00023	0.03940	0.00438	0.03561
	0.03940	6.75195	0.75129	6.10249
	0.00438	0.75129	0.08360	0.67902
	0.03561	6.10249	0.67902	5.51550
F_1	0.00895	1.53374	0.17065	1.38619
	-0.01765	-3.02506	-0.33659	-2.73405
Q_2	0.24621	0.11866	14.98355	9.73271
	0.11866	0.05719	7.22105	4.69050
	14.98355	7.22105	911.83813	592.29370
	9.73272	4.69050	592.29370	384.73022
F_2	0.56910	0.27424	34.63293	22.49620
	-0.32115	-0.15476	-19.54404	-12.69504
Q_3	0.17288	-0.43406	7.14036	-7.87012
	-0.43406	1.08987	-17.92834	19.76067
	7.14036	-17.92834	294.92139	-325.06299
	-7.87012	19.76067	-325.06299	358.28516
F_3	0.05761	-0.14465	2.37951	-2.62273
	0.34047	-0.85487	14.06264	-15.49990
F_e	0.63566	1.66333	37.18307	21.25964
	0.00167	-4.03469	-5.81798	-30.92899
Q_e	0.41932	-0.27600	22.12828	1.89820
	-0.27600	7.89901	-9.95601	30.55365
	22.12828	-9.95601	1206.84302	267.90967
	1.89821	30.55365	267.90967	748.53076

$$u(t) = -Fx(t) + h\,\phi_C(t), \qquad h = \begin{bmatrix} 21.159 \\ -30.9289 \end{bmatrix}$$

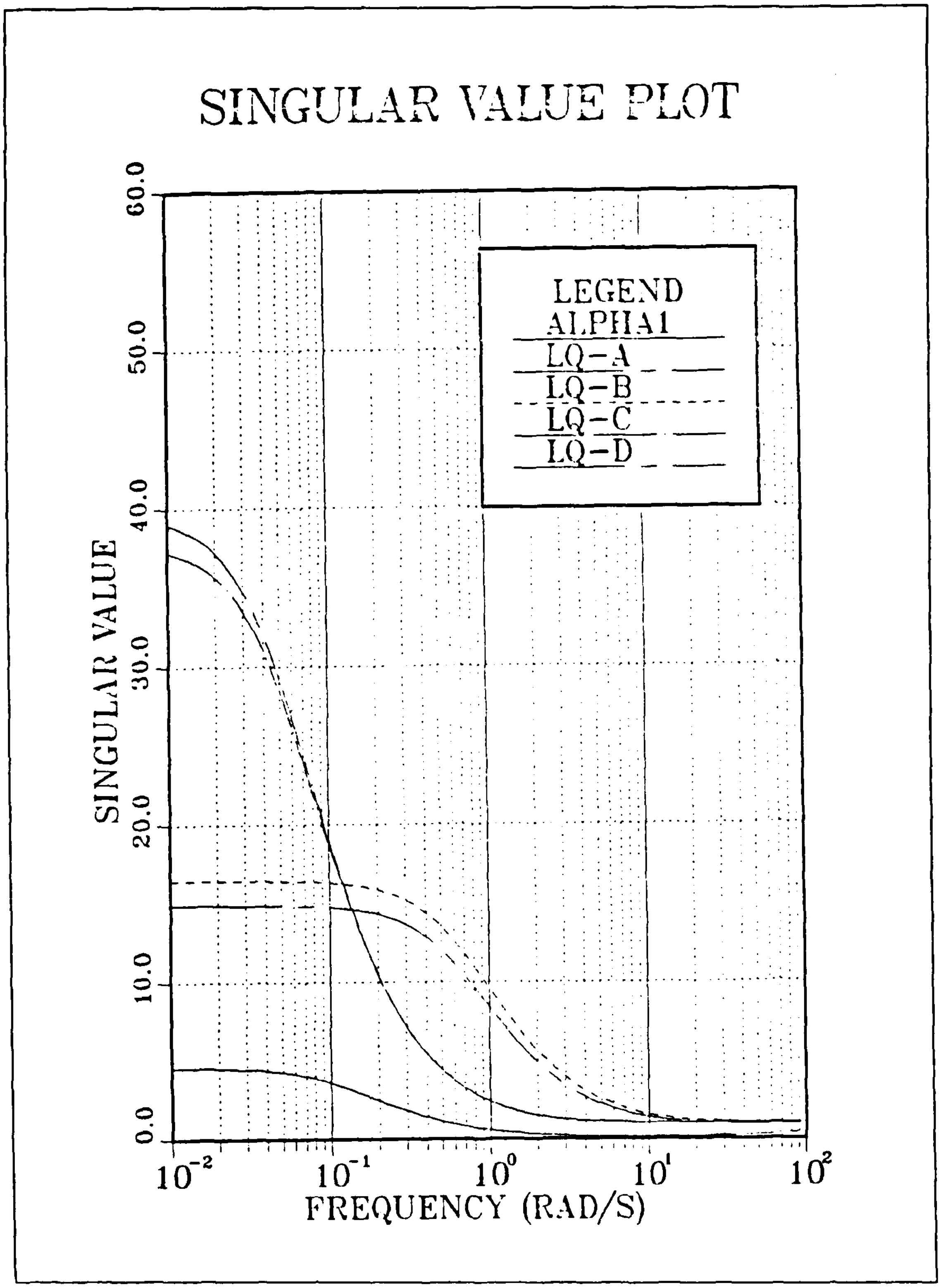


Figure A.1 Singular Value Plots - Comparison.

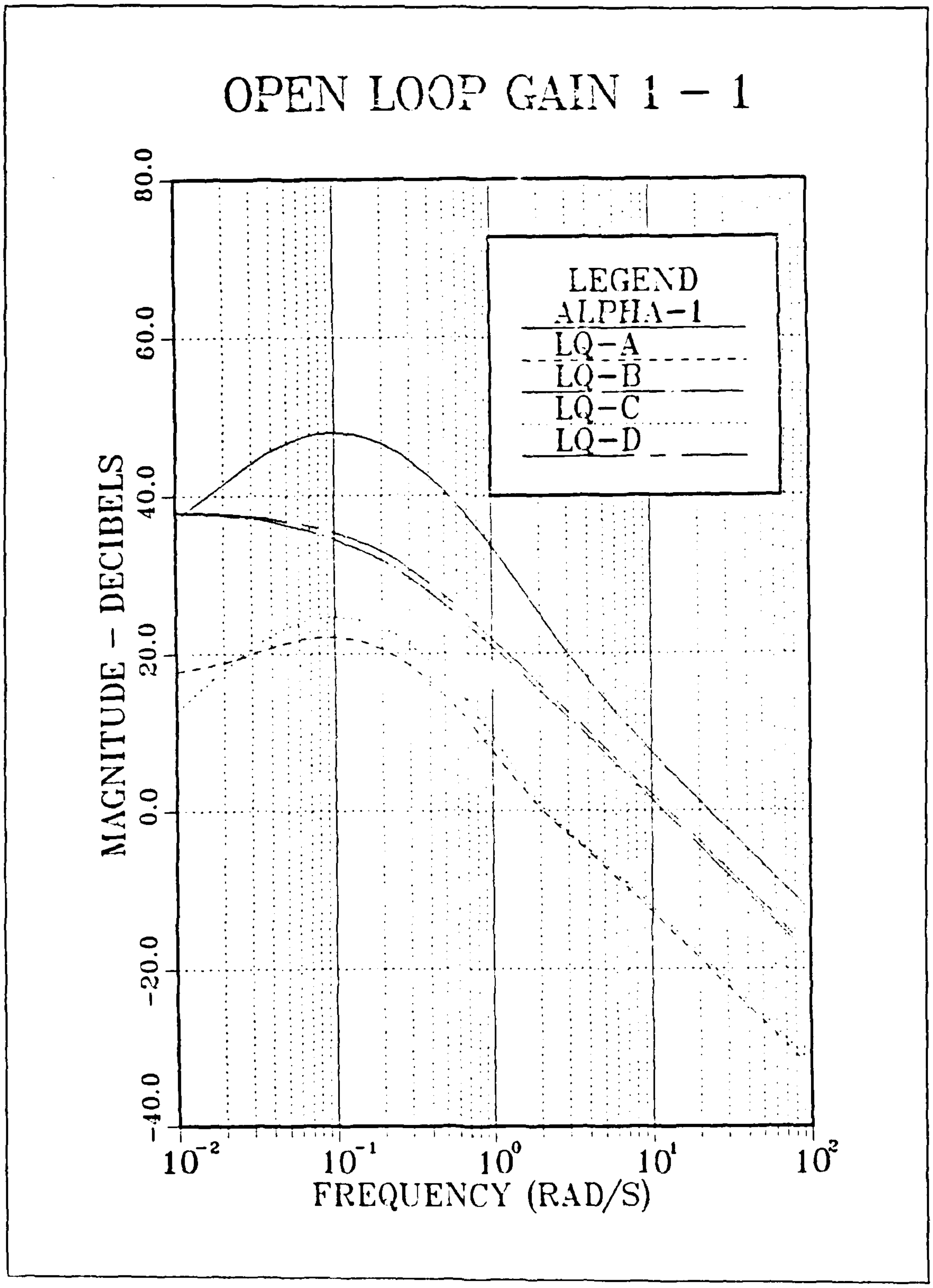


Figure A.2 Bode Plots Comparison- Input 1-1.

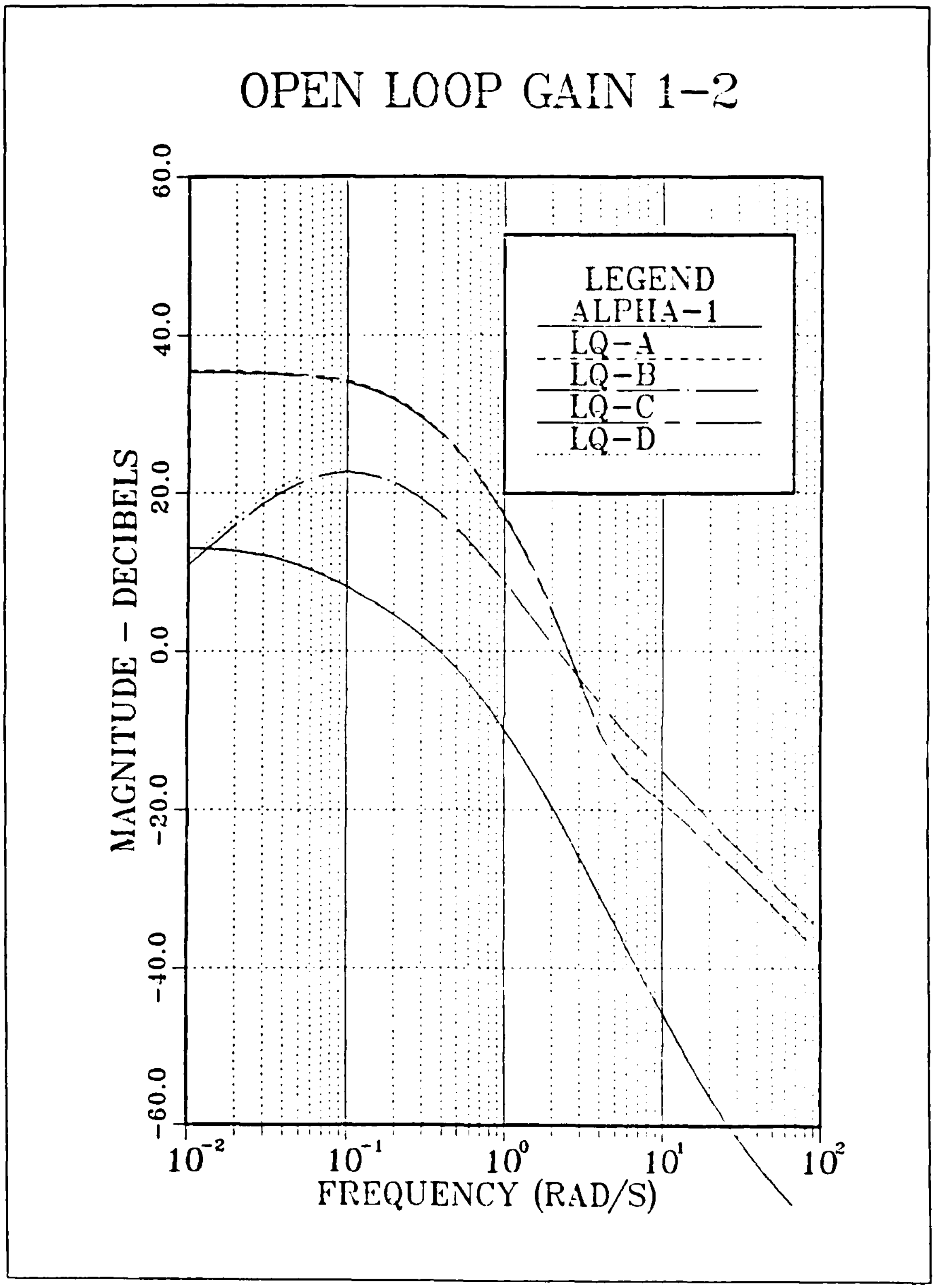


Figure A.3 Bode Plots Comparison - Input 1-2.

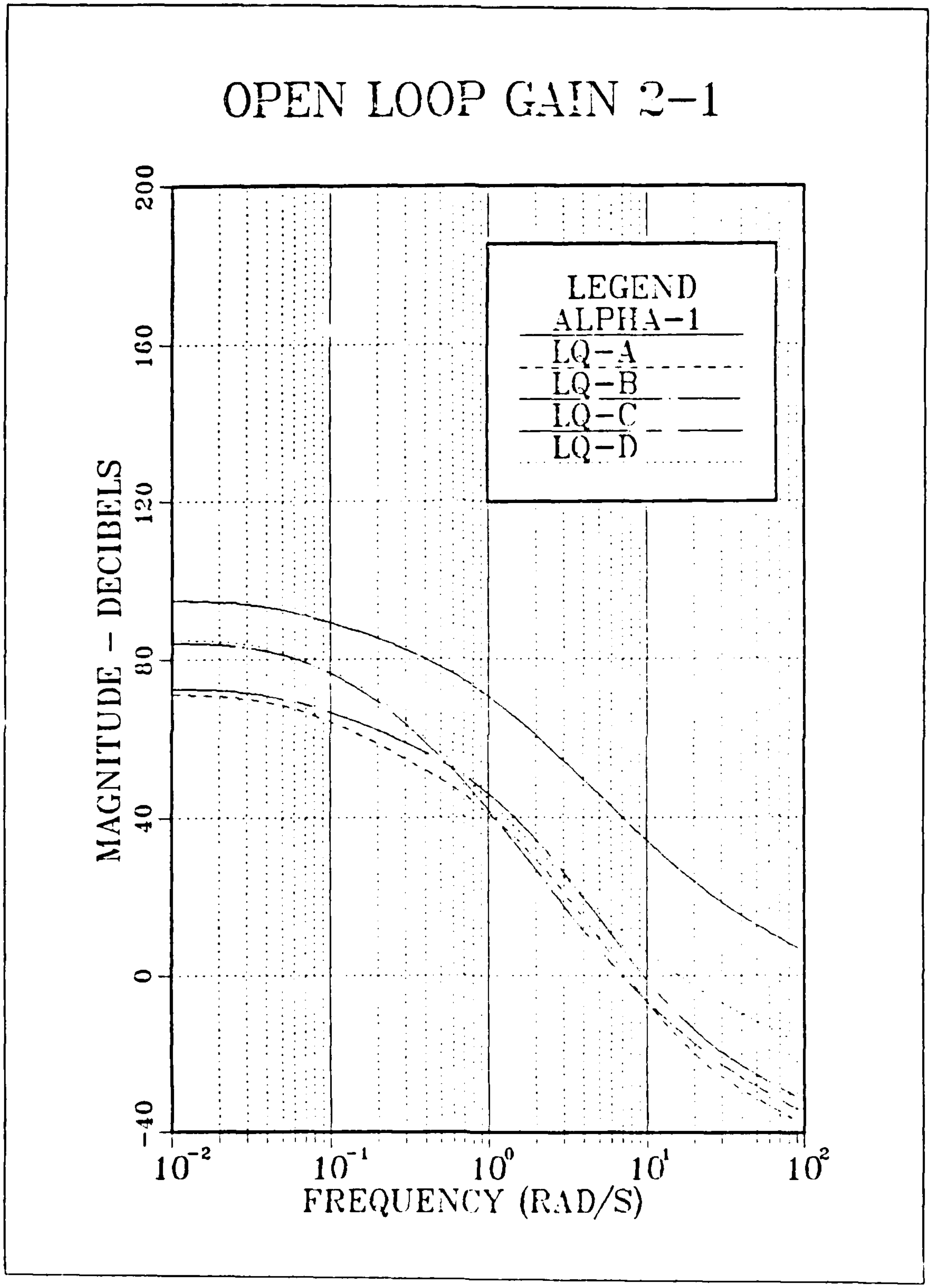


Figure A.4 Bode Plots Comparison - Input 2-1.

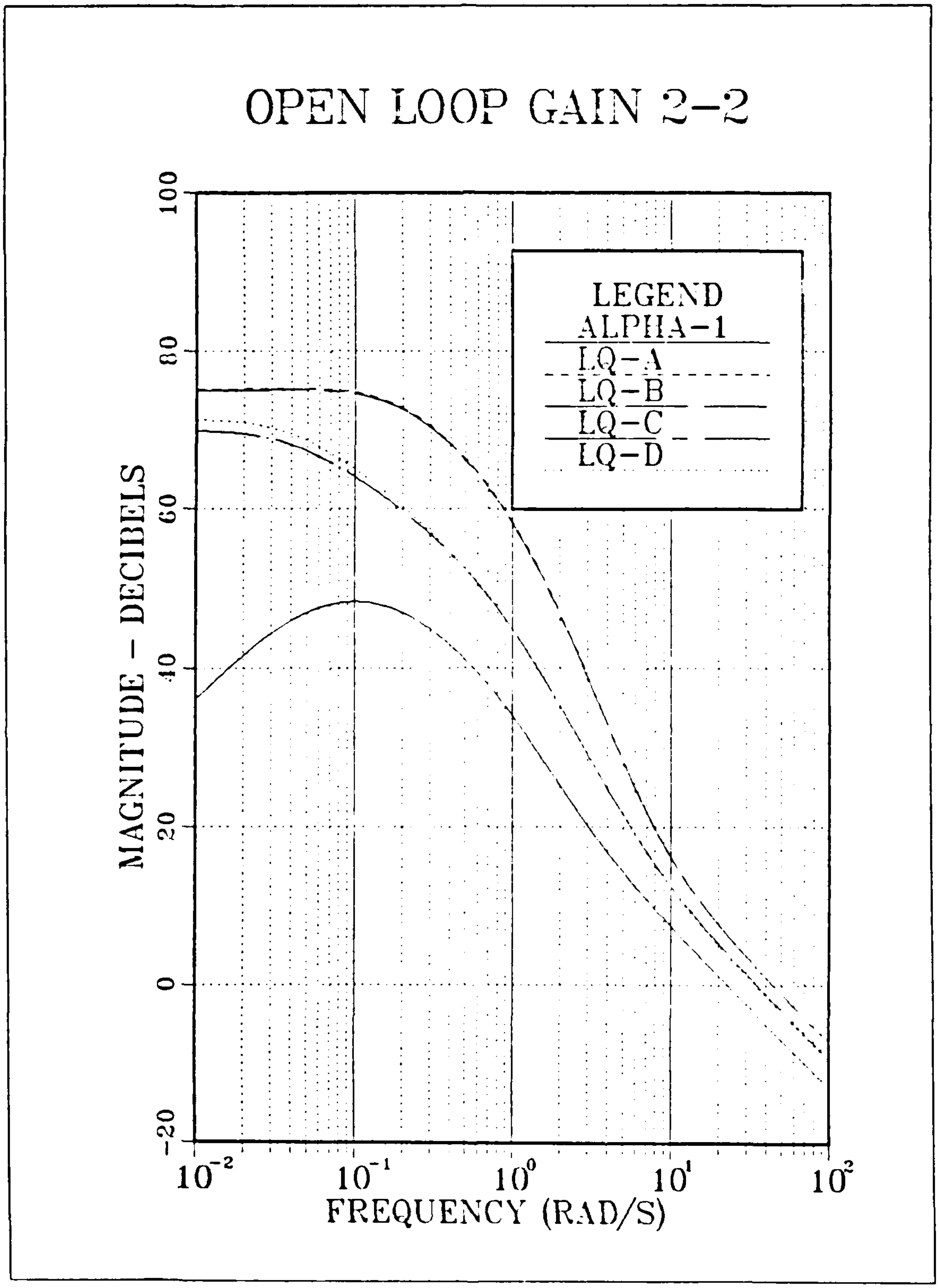


Figure A.5 Bode Plots Comparison - Input 2-2.

APPENDIX B : EXAMPLE DESIGN RUN OUTPUTS

```
***********************************************************************
*                                                                     *
*   THIS EXAMPLE DEMONSTRATES THE POLE REASSIGNMENT WHERE THE OPEN    *
*   LOOP POLE AT -2.126 IS MOVED TO ITS NEW LOCATION AT -25.21        *
*   Q(4,4) IS THE DESIGN VARIABLE. NOTE THAT THE UNSTABLE POLES       *
*   0.2065 IS AUTOMATICALLY MOVED TO ITS MIRROR IMAGE DUE TO LQ       *
*   FORMULATION.                                                      *
*                                                                     *
***********************************************************************

KONTRLO          0

*** THE A PLANT MATRIX ***

   -2.27000      1.42000     -0.15000     31.99001
    0.01000     -0.70000     -0.07000      0.00000
    0.04000     -0.05000     -0.05000      0.00000
    0.00000      1.00000      0.11000      0.00000

*** THE B CONTROL INPUT MATRIX ***

    0.12000      0.95000
    0.04000     -6.37000
    0.34000      0.02000
    0.00000      0.00000

*** THE STARTING Q WEIGHTING MATRIX ***

    0.00000      0.00000      0.00000      0.00000
    0.00000      0.00000      0.00000      0.00000
    0.00000      0.00000      0.00000      0.00000
    0.00000      0.00000      0.00000      0.00000

*** THE DESIGN VARIABLE(Q) X ***

          0            0            0            0
          0            0            0            0
          0            0            0            0
          0            0            0            1
```

```
*** THE R WEIGHTING   MATRIX ***

    1.00000      0.00000
    0.00000      1.00000

*** THE STARTING/FINAL  F  MATRIX ***


  -14.78000      2.16000     77.97000    -55.30000
   -0.00567     -2.55600      0.39400    -15.05000


*** THE ORDERED COMPLEX EIGENVALUES (INPUT)

  -25.21001      0.00000

** ICOMP ****              0

***AUX TRANSFORMATION L ***
    0.00000      0.00000      0.00000      0.00000      0.00000      0.00000      0.0
    0.00000      0.00000      0.00000      0.00000      0.00000      0.00000      0.0
    0.00000      0.00000      0.00000      0.00000      0.00000      0.00000      0.0
    0.00000      0.00000      0.00000      0.00000      0.00000      0.00000      0.0
*** EIGENVECTOR OF A ***
    0.14268      0.98509      0.99767      0.99977
    0.10820     -0.00101     -0.03687     -0.00796
   -0.98384      0.15380     -0.04176     -0.01945
    0.00049      0.07703      0.03949      0.00475
 *** TRANSFORMED A MATRIX ***
   -0.05030      0.00000      0.00000      0.00000
    0.00000      0.20653      0.00000      0.00000
    0.00000     -0.00001     -1.04987     -0.00001
    0.00008      0.00001      0.00002     -2.12635
*** TRANSFORMED G MATRIX ***
    0.00000      0.00000      0.00000      0.00000
    0.00000      0.00000      0.00000      0.00000
    0.00000      0.00000      0.00000      0.00000
    0.00000      0.00000      0.00000      0.00000
*** TRANSFORMED B MATRIX ***
   -0.15647    -20.48637
    0.85855    -91.42015
   -1.80459    190.07515
    1.09721    -95.72333
****SQUARE ROOT INVERSE OF R ***
    1.00000      0.00000
    0.00000      1.00000
```

```
  AAAAA    DDDDDD     SSSSSS
 A     A   D     D   S
 A     A   D     D   S
 AAAAAAA   D     D    SSSSS
 A     A   D     D         S
 A     A   D     D         S
 A     A   DDDDDD    SSSSSS

      F O R T R A N   P R O G R A M
                  F O R
A U T O M A T E D   D E S I G N   S Y N T H E S I S
            V E R S I O N  1.00

1
CONTROL PARAMETERS
ISTRAT =     5    IOPT    =    3    IONED   =    3    IPRINT = 1000
IGRAD  =     0    NDV     =    1    NCON    =    2

              -----------------------
              OPTIMIZATION RESULTS
              -----------------------

OBJECTIVE FUNCTION VALUE    0.56055E-13

DESIGN VARIABLES

                LOWER                           UPPER
VARIABLE        BOUND          VALUE            BOUND
    1       0.00000E+00    0.68858E-01     0.10000E+05

DESIGN CONSTRAINTS

    1) -0.5000E-01 -0.9000E+00

FUNCTION EVALUATIONS =     41
```

```
*** THE OPTIMAL Q MATRIX

    0.00000     0.00000     0.00000     0.00000
    0.00000     0.00000     0.00000     0.00000
    0.00000     0.00000     0.00000     0.00000
    0.00000     0.00000     0.00000     0.06886

 END OF POLE PLACEMENT ROUTINE

 THE Q REQUIRED FOR THIS ASSIGNMENT IS

    0.10027     0.96194     0.11959    -1.50849
    0.96194     9.22847     1.14725   -14.47182
    0.11959     1.14725     0.14262    -1.79908
   -1.50849   -14.47182    -1.79908    22.69429

    EIGENSYSTEM OF OPTIMAL REGULATOR.........

    C-LOOP OPTIMAL REG. E-VALUES...DET(SI-F+G*C)..

-2.52094E+01:-1.04978E+00:-2.06591E-01:-5.02113E-02:

    CONTROL EIGENVECTOR MATRIX..............C*M..

   -3.326553E-02  1.117587E-08 -6.622234E-04 -4.134316E-06
    2.902164E+00 -1.370907E-06 -2.356589E-03 -5.652895E-06

*** THE STARTING/FINAL  F  MATRIX ***

    0.00387     0.03151     0.00399    -0.06416
   -0.24260    -2.83522    -0.34541     3.11642

*** THE AUGMENTED A-B*F*C MATRIX ***

   -2.03999     4.10968     0.17766    29.03711
   -2.02074   -24.43202    -2.96126    26.08698
    0.04354    -0.00401    -0.04445    -0.04051
    0.00000     1.00000     0.11000     0.00000

END OF CPTPP ANALYSIS
```

```
C***************************************************************
C
C       ****** INPUT DATA FILE *********
C
C***************************************************************

0000C     0
  10C.        .C1         .01
    1.0       1.0         1.
    1.0        1.0        .00         0
    0     1     2     5     3     3 1000     0
04040402040402040404020201
 -2.27        1.420       -0.15           31.99
   .01        -0.7          -0.07           0.C
 0.04         -0.05         -0.05             0.
   0.            1.       0.11            0.
 0.12         0.95
 0.04           -8.37
 .34            .020
 0.0            .0
  1.            C.          0.             0.
  0.            1.          0.             0.
  0.            C.          1.             0.
  0.            C.          0.             1.
 - 4.78         2.16        77.97            -55.3
 -0.00567         -2.556    0.394            -15.05
 -0.00          0.          0.             0.0
  0.            0.00        0.             0.
  0.            C.          0.00           C.
  0.0           0.          0.             0.00
  1.000         C.00
  0.C0          1.00
0 0 0 0
0 0 0 0
0 0 0 0
0 0 0 1
0.0          10000.
 -25.21
```

```
******************************************************************
*                                                                *
*   THIS EXAMPLE DEMONSTRATES THE POLE REASSIGNMENT WHERE THE OPEN *
*   LOOP COMPLEX CONJUATE PAIR OF POLE AT (-0.39354, 1.426)      *
*   IS MOVED TO ITS NEW LOCATION AT (-0.7, 1.42). Q(1,1) AND     *
*   Q(2,2) ARE THE DESIGN VARIABLE. THERE ARE NO UNSTABLE POLES  *
*   IN THIS CASE SO THE OTHER POLES/EIGENVECTORS REMAIN UNCHANGE. *
*                                                                *
*                                                                *
******************************************************************

KONTRLO          0

*** THE A PLANT MATRIX ***

   -0.18000    -0.03000     9.57000   -31.87000
   -0.20000    -0.55000   109.42999     2.78000
   -0.01000    -0.01770    -0.10000     0.00000
    0.00000     0.00000     1.00000     0.00000

*** THE B CONTROL INPUT MATRIX ***

   -0.35600     0.52000
    0.00000    -1.00000
    0.33000     0.02100
    0.00000     0.00000

*** THE STARTING Q WEIGHTING MATRIX ***

    0.00000     0.00000     0.00000     0.00000
    0.00000     0.00000     0.00000     0.00000
    0.00000     0.00000     0.00000     0.00000
    0.00000     0.00000     0.00000     0.00000

*** THE DESIGN VARIABLE(Q) X ***

          1           0           0           0
          0           2           0           0
          0           0           0           0
          0           0           0           0
```

```
*** THE R WEIGHTING   MATRIX ***

    2.00000      C.00000
    0.00000      7.00000

*** THE STARTING/FINAL  F  MATRIX ***

    1.00000      1.00000      1.00000      1.00000
    1.00000      1.00000      1.00000      1.00000

*** THE ORDERED COMPLEX EIGENVALUES (INPUT)

   -0.70000      1.42000

** ICOMP ****            1

***AUX TRANSFORMATION L  ***
    0.50000      0.00000      0.00000     -0.50000      0.00000      0.00000      C.0
    0.50000      0.00000      0.00000      C.50000      0.00000      0.00000      C.0
    0.00000      0.00000      0.00000      0.00000      1.00000      0.00000      0.0
    0.00000      0.00000      0.00000      C.00000      0.00000      0.00000      1.0
*** EIGENVECTOR OF A ***
   -0.13554     -0.13554      0.67738      C.87049
    0.14603      C.14603     -0.47971     -C.49218
   -0.01263     -C.01263     -0.00119     -C.00008
    0.00453      0.00453     -0.00865      C.00045
 *** TRANSFORMED A MATRIX ***
   -0.39354     -1.42648     -0.00001      C.00000
    1.42646     -0.39355      0.00000      C.00000
    0.00000      C.00001      0.13768      C.00000
    0.00000     -0.00001      0.00000     -C.18058
*** TRANSFORMED Q MATRIX ***
    0.00000      0.00000      C.00000      0.00000
    0.00000      0.00000      0.00000      0.00000
    0.00000      C.00000      0.00000      C.00000
    0.00000      0.00000      0.00000      C.00000
*** TRANSFORMED B MATRIX ***
  -24.68213     -1.69931
   -1.57555      0.52757
  -11.17528     -1.26164
    6.63291      1.73118
****SQUARE ROOT INVERSE OF R ***
    0.70711      C.00000
    0.00000      0.37796
```

```
      AAAAA      DDDDDD      SSSSSS
     A     A     D     D    S
     A     A     D     D    S
     AAAAAAA     D     D     SSSSS
     A     A     D     D          S
     A     A     D     D          S
     A     A     DDDDDD     SSSSSS

         F O R T R A N   P R O G R A M

                     F O R

   A U T O M A T E D   D E S I G N   S Y N T H E S I S

               V E R S I O N   1.00

1    CONTROL PARAMETERS
     ISTRAT =     5    IOPT    =     3    IONED   =     3    IPRINT = 1000
     IGRAD  =     0    NDV     =     2    NCON    =     3

                    -----------------------
                    OPTIMIZATION RESULTS
                    -----------------------

     OBJECTIVE FUNCTION VALUE    0.28365E-10

     DESIGN VARIABLES

                  LOWER                         UPPER
     VARIABLE     BOUND          VALUE          BOUND
        1      0.00000E+00   0.23088E-02   0.10000E+04
        2      0.00000E+00   0.22134E-02   0.10000E+04

     DESIGN CONSTRAINTS

         1) -0.4999E-01 -0.9000E+00  0.9545E-04

     FUNCTION EVALUATIONS =    48
```

*** THE OPTIMAL Q MATRIX

0.00231	0.00000	0.00000	0.00000
0.00000	0.00221	0.00000	0.00000
0.00000	0.00000	0.00000	0.00000
0.00000	0.00000	0.00000	0.00000

END OF POLE PLACEMENT ROUTINE

THE Q REQUIRED FOR THIS ASSIGNMENT IS

0.00060	0.00106	-0.00908	0.00301
0.00106	0.00189	-0.01970	0.00579
-0.00908	-0.01970	12.94063	-1.60992
0.00301	0.00579	-1.60992	0.20628

EIGENSYSTEM OF OPTIMAL REGULATOR.........

C-LOOP OPTIMAL REG. E-VALUES...DET(SI-F+G*C)..

-6.99887E-01, 1.41996E+00:-1.80579E-01:-1.37682E-01:

CONTROL EIGENVECTOR MATRIX..............C*M..

4.996814E-03	-2.528824E-02	-6.565824E-08	-5.181365E-03
2.956046E-04	-4.560151E-04	-3.259629E-09	-1.896438E-04

*** THE STARTING/FINAL F MATRIX ***

-0.01311	-0.02106	2.67146	2.79805
-0.00052	-0.00084	0.06034	0.09177

*** THE AUGMENTED A-B*F*C MATRIX ***

-0.18440	-0.03706	10.48966	-30.92160
-0.20052	-0.55084	109.49033	2.87177
-0.00566	-0.01073	-0.98285	-0.92528
0.00000	0.00000	1.00000	0.00000

END OF OPTPP ANALYSIS

```
C*****************************************************************
C
C       ****** INPUT DATA FILE *********
C
C*****************************************************************


 0000     0
  100.         .1            .1
    1.0        0.00           0.
    1.0        1.0           .00          0
    0      2    3     5       3    3 1000      0
04040402040402040404020201
 -0.16         -0.03          9.57           -31.87
 -0.2          -0.55          109.43          2.78
 -0.01          -0.0177        -0.10           0.
   0.            0.           1.0           0.
 -.356          0.52
 0.0            -1.00
 .33            .021
 0.0            .0
  1.            0.            0.            0.
  0.            1.            0.            0.
  0.            0.            1.            0.
  0.            0.            0.            1.
  1.            1.            1.            1.
  1.            1.            1.            1.
  0.00          0.            0.            0.
  0.            0.00          0.            0.
  0.            0.            0.            0.
  0.            0.            0.             0.0
  2.            0.
  0.            7.
 1  0  0  0
 0  2  0  0
 0  0  0  0
 0  0  0  0
 0.0          1000.
 0.0          1000.
 -0.70          1.42
```

APPENDIX C : COMPUTER PROGRAM LISTINGS

```
C*****************************************************************  CONO0100
C                                                                 *
C     *** CPTPP --- OPTIMAL POLE PLACEMENT PROGRAM****            *
C                                                                 *
C       THIS PROGRAM IS DESIGNED TO COMPUTE THE STATE WEIGHTING MATRIX  *  CONO0110
C       OF A MIMO OPTIMAL CONTROL PROBLEM GIVEN THE DESIRED CLOSED-LOOP *  CONO0120
C       POLES LOCATIONS.  THE DESIGNER HAS THE CHOICE OF ASSIGNING ONE  *  CONO0130
C       OR MORE POLES DURING ONE RUN.  ON COMPLETION OF POLE ASSIGNMENT. *  CONO0140
C       A MODIFIED VERSION OF THE OPTSYS ROUNTINE IS CALLED TO COMPUTE  *  CONO0150
C       THE FEEDBACK GAIN MATRIX.  SINGULAR VALUE PLOTS ARE THEN COMPUTED *  CONO0160
C       TO DETERMINE THE SYSTEM ROBUSTNESS.                       *  CONO0170
C                                                                 *
C     THE PROGRAM CAN BE MODIFIED TO COMPUTE THE CONTROL WEIGHTING MATRIX*
C     OR BOTH THE STATE AND CONTROL WEIGHTING MATRIX IF BOTH OF THEM ARE *
C     UNKNOWN.                                                    *
C                                                                 *
C     THE EIGENVECTOR TYPE OF ASSIGNMENT CAN BE INCORPORATED IF REQUIRED *
C                                                                 *
C     BY CHOW WAH KEH , REPUBLIC OF SINGAPORE                     *
C          VERSION 1.0 SEPT 1985                                  *
C                                                                 *
C*****************************************************************  CONO0200
C    *                                                            *  CONO0210
C    *                                                            *  CONO0220
C    * THIS FIRST BLOCK SETS DIMENSIONS AND DECLARES REAL AND     *  CONO0230
C    *  COMPLEX FUNCTIONS. THE FOLLOWING IS A LIST OF PROGRAM SYMBOLS: *  CONO0240
C    *                                                            *  CONO0210
C    ************************************************************  CONO0260
C    *                                                            *  CONO0270
C    *           REAL VARIABLES BY NAME AND BRIEF DESCRIPTION     *  CONO0280
C    *     THE SYSTEM UNDER CONSIDERATION IS GIVEN AS:            *  CONO0290
C    *         XDOT = A*X + B*U ; Y= C*X    , FOR STATE FEEDBACK OR *  CONO0300
C    *          OUTPUT FEEDBACK OF THE FORM U=-FX +R OR U=-FY + R *  CONO0310
C    *      A  -  THE PLANT A MATRIX (NSTATE X NSTATE)            *  CONO0320
C    *      B  -  THE CONTROL MATRIX  (NSTATE X NCONTROL)         *  CONO0330
C    *      C  -  THE PLANT OBSERV. OR OUTPUT MATRIX (NOUTPUT X NSTATE; *  CONO0340
C    *              FOR STATE FEEDBACK PROBLEMS)                  *  CONO0350
C    *      F  -  THE FEEDBACK GAIN MATRIX(NCONTROL X NSTATE;     *  CONO0360
C    *              FOR STATE FEEDBACK PROBLEMS)                  *  CONO0370
C    *      QW -  THE STATE WEIGHTING MATRIX   (INPUT/OUTPUT)     *
C    *      RW -  THE CONTROL WEIGHTING MATRIX (INPUT)            *
C    *      MU - THE DESIRED CLOSED-LOOP POLES LOCATIONS          *
C    *      REALMU - THE REAL PART OF THE DESIRED POLE LOCATION   *  CONO0380
C    *      IMAGMU - THE COMPLEX PART OF THE DESIRED POLE LOCATION *  CONO0390
C    *      FC   -  F*C                                           *  CONO0400
C    *      BFC  -  B*F*C                                         *  CONO0410
C    *      AMBFC - A - B*F*C                                     *  CONO0420
```

```
C     *      REIG - REAL COMPUTED EIGENVALUE OF THE SYSTEM                  * CONOC430
C     *--------------------------------------------------------------------*
C     *      THE FOLLOWINGS ARE USED BY THE SINGULAR VALUE ANALYSIS MODE   *
C     *--------------------------------------------------------------------*
C     *      OMEGA - FREQUENCY                                              * CCNOC470
C     *      SV - SINGULAR VALUE OF (I + F * PLANT TRANSFER FUNCTION(G))    * CCNOC480
C     *      SVMI - SINGULAR VALUE OF (I + INV(F* G ))                      * CCNOC490
C     *      SVAC - SINGULAR VALUE OF ( I + G * F)                          * CCNOO500
C     *      SVMC - SINGULAR VALUE CF ( I + INV (G * F))                    * CONOC510
C     *              ONLY OF INTEREST IF NO. OUTPUTS = NO. INPUTS.          * CCNOC520
C     *      SIGNM1 - FIRST SINGULAR VALUE OF (I + F*G)                     * CCNOC530
C     *      SIGNM2 - SECOND SINGULAR VALUE CF (I + F*G)                    * CCNOC540
C     *      SIGNMX - MAXIMUM SINGULAR VALUE OF (I + F*G)                   * CGNOC550
C     *      SIGPRO - SQRT(FIRST SING VAL * SECOND SING VAL)                * CONOC560
C     *      SVADMO - MIN ADDITIVE OUTPUT SINGULAR VALUE (I + G*F)          * CONOC570
C     *      SVADXO - MAX ADDITIVE OUTPUT SINGULAR VALUE                    * CCNOO580
C     *      SVMIM - MIN MULTPLICATIVE INPUT SINGULAR VALUE (I+INV(FG))     * CONOO590
C     *      SVMIX - MAX MULTIPLICATIVE INPUT SINGULAR VALUE                * CONOC600
C     *      SVMMC - MIN MULTIPLICATIVE OUTPUT SINGULAR VALUE (I+INV(GF))   * CCNOO610
C     *      SVMXO - MAX MULTIPLICATIVE OUTPUT SINGULAR VALUE               * CGNOO620
C     *                                                                     * CCNOO630
C     ********************************************************************** CUNOO640
C
      IMPLICIT REAL*4(A-H,O-Z)
      REAL*4 A(10,10),B(10,10),C(10,10),REALMU(50),IMAGMU(50),           CONOC650
     1F(10,10),FC(10,10),BFC(10,10),AMBFC(10,10),WK(1000),MEIG(10),       CONOC660
     2REIG(50),IEIG(50),SV(10),WW(10,10),WA(100),MINEIG(500),             CCNOO670
     3OMEGA(500),SIGNM1(500),SIGNM2(500),SIGNMX(500),SIGPRO(500),         CONOC680
     4SVADMO(500),SVADXO(500),SVMIM(500),SVMIX(500),                      CCNOC690
     7LP11(500),LP12(500),LP21(500),LP22(500),
     5SVMMO(500),SVMXO(500),SVMI(10),SVAC(10),SVMO(10),TEMPR(10,10),      CONOC700
     6QW(10,10),RW(10,10),RWSP(10,10),RWSP1(10,10),RCOND,Z1(10)
C
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     *         QWX - COMPLEX OF QW                                          *
C     *         RWX - COMPLEX OF RW                                          *
C     *       RWSQX - COMPLEX OF R**(-1/2)                                   *
C     *                                                                      *
C     *---------------------------------------------------------------------*
C     *- THE FOLLCWINGS ARE USED IN THE COCRDINATE TRANSFORMATICN BLOCK     *
C     *---------------------------------------------------------------------*
C     *                                                                      *
C     *          Z - EIGENVECTOR OF A ( M IN THE THESIS )                    * CCNOC790
```

```
C     *       CZI - INVERSE OF Z                                          *
C     *       CZTI- INVERSE OF TRANPOSE OF Z                              *
C     *       LX  - AUX TRANSFORMATION MATRIX FOR COMPLEX EIGENVALUE      *
C     *       LXI  - INVERSE OF LX                                        *
C     *      ZLI  - INVERSE OF (Z*LX)                                     *
C     *-----------------------------------------------------------------*
C     *- THE FOLLOWINGS ARE USED TO COMPUTE THE OBJECTIVE FUNCTION       *
C     *-----------------------------------------------------------------*
C     *                                                                   *
C     *      GPS - COMPLEX MATRIX COMPUTE BY PLANT1 G(S)                  *
C     *            WHERE G(S)=C*(SI-A)**(-1) * B                          *
C     *       GMS - SAME AS ABOVE BUT G(-S)                               *
C     *    GPSRSQ - G(S)*R**(-1/2)                                        *
C     *       QGR - Q*GPSRSQ                                              *
C     *      GQGR - GMS*QGR                                               *
C     *     RGQGR - R**(-1/2) * GQGR                                      *
C     *    XIRQR  - I +RGQGR                                              *
C     *   DETERM  - AN ARRAY CONSISTING OF THE DET OF XIRQI               *
C     *  TO BE USED IN THE OBJECTIVE FUNCTION OF THE OPTIMIZER            *
C     *                                                                   *
C     *-----------------------------------------------------------------*
C     *   THE FOLLOWING ARE USED BY THE SINGULAR VALUE ANALYSIS          *
C     *-----------------------------------------------------------------*
C     *          EIG - EIGENVALUE OF A- B*F*C                             * CON00780
C     *          MU - INPUT DESIRED POLE LOCATION                         * CON00800
C     *          OMU - ORDERED VALUE OF MU                                * CON00810
C     *          OEIG - ORDERED VALUE OF EIG                              * CON00820
C     *          OZ - ORDERED EIGENVECTOR                                 * CON00830
C     *          XI - IDENTITY MATRIX                                     * CON00840
C     *          XXI - IDENTITY MATRIX                                    * CON00850
C     *          UI - INPUT ADDITIVE SINGULAR VECTOR ASSOCIATED WITH SV   * CON00860
C     *          VI - INPUT ADDITIVE SINGULAR VECTOR ASSOCIATED WITH SV   * CON00870
C     *          UO - OUTPUT ADDITIVE SINGULAR VECTOR (SVAO)              * CON00880
C     *          VO - OUTPUT ADDITIVE SINGULAR VECTOR (SVAO)              * CON00890
C     *          UMI- INPUT MULTIPLICATIVE SINGULAR VECTOR                * CON00900
C     *          VMI - INPUT MULTIPLICATIVE SINGULAR VECTOR               * CON00910
C     *          UMO - OUTPUT MULTIPLICATIVE SINGULAR VECTOR              * CON00920
C     *          VMO - OUTPUT MULTIPLICATIVE SINGULAR VECTOR              * CON00930
C     *          FXPLT - F * PLANT TRANSFER FUNCTION (G)                  * CON00940
C     *          XIPFXP - I + F*G                                         * CON00950
C     *          PLAN - PLANT TRANSFER FUNCTION (G)                       * CON00960
C     *          PLTFX - G*F                                              * CON00970
C     *          XIPPFX - I + G*F                                         * CON00980
C     *          PLTFXI -  INV(F*G)                                       * CON00990
C     *          FXPLTI -  INV(G*F)                                       * CON01000
C     *          XIFPLI -  I + INV (F*G)                                  * CON01010
C     *          XIPLFI -  I + INV (F*F)                                  * CON01020
C     *                                                                   * CON01030
```

```
C     *                                                                    *  CCNO1C40
C     **********************************************************************  CONO1C50
C                                                                             CCNO1C60
C                                                                             CCNO1C70
       COMPLEX *8EIG(50),Z(10,10),MU(50),ONU(50),DEIG(50),DZ(10,10),AX(10CONO1C80
      1,10),BX(10,10),CX(10,10),FX(10,10),FXPLT(10,10),XI(10,10),XIPFXP(1CONO1C90
      20,10),UI(10,10),VI(10,10),PLAN(10,1C),XXI(10,10),UO(10,10),VO(10,1CONO1100
      30),UMI(1C,1C),VMI(10,10),UMO(10,10),VMO(10,10),PLTFX(10,10),XIPPFXCONO1110
      4(10,10),PLTFXI(10,10),FXPLTI(10,10),XIFPLI(10,10),XIPLFI(10,10),   CCNO1120
      5QWX(10,10),RWX(10,10),GPS(10,10),GMS(10,10),RWSQX(10,10),
      6XIRCR(10,10),DET(2),RGCGR(10,10),GCGR(10,10),TEMPX(10,10),
      7QGR(10,10),WORK(10),DETERM(3),GPSRSQ(10,10),DETQW,OZI(1C,10)
      8,OZTI(1C,10),TEMPX2(10,10),LX(10,10),LXI(10,10),ZLI(10,10)
       COMPLEX ZN,TEMP1,TEMP2                                                 CCNO1170
C
C
C     **********************************************************************  CGNO1140
C     *                                                                    *
C     *----- OPTSYS VARIABLES ---------------------------------------------*
C     *                                                                    *
C     **********************************************************************  CONO1140
       REAL*4 EA(1C,10),G1(10,10),AY(10,10),HO(10,10),B1(10,10),              CPT06E70
      1PRC(10,10),AA(10,10),RM(20,20),X1(20,20)                               CPTC6E80
      2,GN(10,10),B1I(10,10),FBGC(2,4)                                        CPTC6900
C
C      *---- THE FOLLOWING DUMMY VARIABLES MUST BE DECLARED EXACTLY AS    *
C            IN THE PROBLEM SO AS TO PASS VARIABLE CORRECTLY FROM OPTPP  *
C            TC INNER SUBROUTINE  ------------------------------------*
C
C     3,DUMA(2,2),DUMB(2,2),DUMC(2,2),DUMQ(2,2),DUMR(2,2),DUMRI(2,2)
      3,DUMA(4,4),DUMB(4,2),DUMC(4,4),DUMQ(4,4),DUMR(2,2),DUMRI(2,2)
C     **********************************************************************  CONO1140
C     *                                                                    *
C     *----LINPACK VARIABLE S ---------------------------------------------  CONOOC10
C     *                                                                    *
C     **********************************************************************  CONO1140
C
       INTEGER ISAMAX,IDAMAX,ICAMAX,IZAMAX,IPRINT                             CCN00020
       REAL SCCT,SNRM2,SCNRM2,SASSUM,SCASUM                                   CONOOC30
       COMPLEX CDOTC,CDOTU                                                    CONOCC40
       COMPLEX*16 ZDOTC,ZDCTU                                                 CCNOCC50
       DOUBLE PRECISION DDOT,DNRM2,DASUM,DZNRM2,DZASUM                        CONOCC60
       INTEGER N,INCX,INCY,IMAX                                               CCNOOC70
       REAL SX(1),SY(1),SA,SC1,SS,SW                                          CONOCC80
       DOUBLE PRECISION DX(1),DY(1),DA,DC,CS,DW                               CCNOCC90
       COMPLEX CX1(1),CY(1),CA,CW
       COMPLEX*16 ZX(1),ZY(1),ZA,ZW
C
```

```
C     **************************************************************** CON01140
C     *                                                              *
C     *    SET UP DIMENSIONS FOR OPTIMIZER PARAMETERS AND OTHER FUNCTIONS*CON01190
C     *--------------------------------------------------------------* CON01180
C     *                                                              * CON01200
C     *      X - THE DESIGN VARIABLE FOR THE OPTIMIZER               * CON01210
C     *      VLB - DESIGN VARIABLE LOWER BOUND                       * CON01220
C     *      VUB - DESIGN VARIABLE UPPER BOUND                       * CON01230
C     *      G - OPTIMIZER CONSTRAINT VECTOR                         * CON01240
C     *      IDG - OPTIMIZATION CONSTRAINT TYPE                      * CON01250
C     *      IC  - OPTIMIZER PARAMETER                               * CON01260
C     *      DF  - OPTIMIZER GRADIENT VECTOR                         * CON01270
C     *      XA  - OPTIMIZER STORAGE ARRAY                           * CON01280
C     *      WK1 - OPTIMIZER WORK SPACE                              * CON01290
C     *      IWK - OPTIMIZER WORK SPACE                              * CON01300
C     *      IF - ARRAY USED TO CHOOSE DESIGN VARIABLES FROM INPUT DATA  * CON01310
C     *      IR - USED BY SORT ROUTINE                               * CON01320
C     *      KEY - USED BY SORT ROUTINE                              * CON01330
C     *      R - USED BY SORT ROUTINE                                * CON01340
C     *                                                              * CON01350
C     **************************************************************** CON01360
C                                                                      CON01370
      DIMENSION X(50),VLB(50),VUB(50),G(50),IDG(50),IC(50),DF(50),XA(50,CON01380
     150),WK1(10000),IWK(1000),IF(10,10),IR(50),KEY(50),R(50),          CON01390
     2IQW(10,10),IRW(10,10),IPVT(10)
      DATA NRA/50/,NCA/50/,NRWK/10000/,NRIWK/1000/                       CON01400
C     DATA LDA/10/
C ******************************************************************* CON01410
C                                                                    * CON01420
C        INPUT DATA INTO THE PROGRAM TO SET UP THE SYSTEM---------    * CON01430
C           XDOT=AX+BU ; Y=CX                                        * CON01440
C            WHERE U=-FY+R    OR U=-FX+ R   I.E. C=DIA(I)            * CON01450
C******************************************************************** CON01470
C                                                                    *
C     *      DATA FORMAT PREPARATION                                 * CON01480
C                                                                    *
C     * CARD 1 KODE, KONTRL; FORMAT (I5,I5)                          * CON01490
C     * KODE ,KONTRL ARE INPUT NUMBERS USED TO IDENTIFY MODES AS;      CON01500
C     * KODE   KONTRL
C     *  0      0    ENTER OPTIMIER ON FIRST PASS W/O SINGULAR VALUE PLOTS* CON01510
C     *  0      1    COMPUTE THEN OPTIMIZE W/O SINGULAR VALUE         * CON01520
C     *  2      1    ENTER OPTIMIER ON FIRST PASS WITH SINGULAR VALUE PLOT* CON01530
C     *  2      2    NO POLE PLACEMENT, COMPUTE SINGULAR VALUE USING F    * CON01540
C     *                                                              * CON01550
C     * CARD 2 WMAX,WI,DELW; FORMAT(3F10.0)                          * CON01560
C     *        WMAX - MAXIMUM FREQUENCY TO BE USED                   * CON01570
C     *        WI - STARTING FREQUENCY                               * CON01580
C     *        DELW - FREQUENCY STEP FOR FIRST DECADE , THIS STEP IS * CON01590
```

```
C     *               MODIFIED BY PROGRAM AT EACH DECADE SHIFT              * CCNO1600
C     *                                                                      * CCNO1550
C     * CARD 3 WT1,WT2,WT3; FCRMAT(3F10.0)                                   * CCNO1610
C     *        WEIGHTING FACTORS TO BE USED IN THE MULTIPLE POLES            * CCNO1620
C     *        PLACEMENT MCDES ( NOT USED IN THE PRESENT FORMULATION)        * CONO1630
C     *                                                                      * CCNO1550
C     * CARD 4   SVMINI,SVMINO,RJ,NIDG; FORMAT(3F10.0,I5)                    * CCNO1720
C     *          ( ONLY NIDG IS USED IN THIS PROGRAM )                       *
C     *          SVMINI - DESIRED SINGULAR VALUE LEVEL FOR THE S.V. OF       * CCNO1730
C     *                (I+FG).FOUND USING UNIVERSAL GAIN AND PHASE CHART     * CCNO1740
C     *                TO GIVE A DESIRED PHASE AND GAIN MARGIN               * CONO1750
C     *          SVMINO - OUTPUT VERSION OF SVMINI                           * CCNO1760
C     *          RJ - FACTOR TO SET PERCENT POLE PLACEMENT ERROR THE         * CONO1770
C     *               DESIGNER WISHES TO PUT INTO THE OPTIMIZER              * CCNO1780
C     *                                                                      * CCNO1550
C     * ------- NIDG - PARAMETER THAT SETS TYPE CONSTRAINT CONSIDERED        * CCNO1790
C     *                SEE ADS MANUAL FOR USE                                * CONO1800
C     *                                                                      * CCNO1550
C     * CARD 5   IGRAD,NDV,NCON,ISTRAT,IOPT,IONED,IPRINT,INFO                * CONO1810
C     *             FORMAT(8I5)                                              * CCNO1820
C     *          IGRAD - PARAMETER CONCERNING GRADIENT COMP SEE ADS          * CONO1830
C     *          NDV - NUMBER OF DESIGN VARIABLES                            * CONO1840
C     *          NCON - NUMBER OF CONSTRAINTS                                * CCNO1850
C     *          ISTRAT - OPTIMIZER STRATEGY SEE ADS MANUAL                  * CCNO1860
C     *          IOPT - OPTIMIZATION METHOD SEE ADS                          * CONO1870
C     *          IONED - ONE DIMENSIONAL SEARCH TECHNIQUE SEE ADS            * CCNO1880
C     *          IPRINT - ADS PRINT CONTROL SEE ADS                          * CONO1890
C     *          INFO - OPTIMIZER CONTROL PARAMETER SEE ADS                  * CCNO1900
C     *                                                                      * CONO1550
C     *CARD 6 NROWA,NCOLA,NROWB,NCOLB,NROWC,NCOLC,NROWF,NCOLF,NROWQ          * CCNO1910
C     *       NCOLQ,NROWR,NCOLR,NMU,                                         * CONO1920
C     *       FORMAT(12I2)                                                   *
C     *          NUMBER OF ROWS AND COLUMNS OF INPUT MATRICES AND            * CONO1930
C     *          NUMBER OF POLES TO BE INPUT FOR PLACEMENT                   * CONO1940
C     *          EXAMPLE: NMU = 1 MEANS TO PLACE POLE ONE AT A TIME          *
C     *                                                                      * CCNO1550
C     * CARD 7 A MATRIX ; READ BY ROWS AS:                                   * CCNO1950
C     *                A(1,1)     A(1,2)                                     * CONO1960
C     *                A(2,1)     A(2,2)                                     * CCNO1970
C     * THERE WILL BE ONE OR MORE CARDS FOR EACH ROW OF A                    * CCNO1980
C     * AFTER THE PROPER CARDS ARE READ FOR A THE NEXT MATRIX MUST           * CCNO1990
C     * BE READ IN. THE REMAINDER OF THIS COMMENT WILL REFER TO EACH NEW* CONO2000
C     * DATA SET AS A CARD. REMEMBER, THERE MAY BE SEVERAL CARDS PER         * CONO2010
C     * SET SO THE NUMBERS DON'T CORRESPOND DIRECTLY WITH  THE DATA          * CCNO2020
C     * LISTED IN THE INPUT FILE.                                            * CCNO2030
C     *                                                                      * CGNO1550
C     * CARD 8 B MATRIX; READ ROW BY ROW                                     * CCNO2040
C     * CARD 9 C MATRIX; READ ROW BY ROW                                     * CONO2050
```

```
C     * CARD 10 F MATRIX; READ ROW BY ROW                                     * CONO2060
C     *         THE F ENTERED HERE IS USED ONLY FOR SINGULAR VALUE            *
C     *         ANALYSIS MODE, FOR OTHER MODES THE POLE PLACEMENT             *
C     *         ROUTINE WILL GENERATE Q AND THEN F (FROM INNER ROUTINE)       *
C     * CARD 11 Q MATRIX; READ ROW BY ROW ( Q=0 WHEN NO STARTING VALUES       *
C     *                                    IS USED )                          *
C     * CARC 12 R MATRIX; READ ROW BY ROW                                     *
C     * CARD 13 IQW MATRIX ; ROW BY ROW                                       * CONO2070
C     *          FORMAT(8I2) ; THIS MATRIX DEFINES THE DESIGN VARIABLE        * CONO2080
C     *          TO THE OPTIMIZER. THE ENTRY WOULD BE OF THE FORM:            * CONO2090
C     *             1 2                                                       * CONO2100
C     *             0 3                                                       * CONO2110
C     *          THIS WOULD TELL THE OPTIMIZER TO MAKE Q(1,1) THE FIRST       * CONO2120
C     *          DESIGN VARIABLE (X(1)) AND SO FORTH. THE ZERO TELLS THE      * CONO2130
C     *          OPTIMIZER THAT F(2,1) (IN THIS CASE) IS NOT A DESIGN         * CONO2140
C     *          VARIABLE AND IS NOT TO BE CHANGED.                           * CONO2150
C     *          AS Q IS REQUIRED TO BE SYMMETRIC , ONLY ONE OFF              *
C     *          DIAGONAL TERM IS REQUIRED. I.E. ONLY Q(1,2) OR Q(2,1)        *
C     *          IS ASSIGNED. THE PROGRAM AUTOMATICALLY MAKES Q(1,2)          *
C     *          EQUAL Q(2,1) ETC.,                                           *
C     *                                                                       * CONO2210
C     * CARC 14  VLB,VUB; FORMAT(2F10.0)                                      * CONO2160
C     *          THERE WILL BE ONE CARD FOR EACH DESIRED POLES                * CONO2170
C     *                                                                       * CONO2210
C     * CARD 15  REALMU,IMAGMU ; FORMAT (2F10.0)                              * CONO2180
C     *           INPUT REAL AND IMAGINARY DESIRED POLE LOCATION              * CONO2190
C     *           ONE POLE PER CARD                                           * CONO2200
C     *                                                                       * CONO2210
C     ************************************************************************* CONO2220
C
        CALL ERFSET(207,256,-1,1,209)
        CALL ERRSET(215,256,-1,1)
        CALL ERFSET(187,256,-1,1)
C                                                                               CONO2230
C                                                                               CONO2240
C     ************************************************************************* CONO2220
C     *              INPUT MODES                                              * CONO2250
C     ************************************************************************* CONO2220
        READ (1,750) KODE,KONTRL                                                CONO2260
        WRITE (6,*) 'KONTRLO',KONTRL
C                                                                               CONO2270
C                                                                               CONO2280
C ---INPUT MAXN, INITIAL FREQUENCY,DELTA FREQ, OPTIMIZER WEIGHTS-----           CONO2290
        READ (1,710) WMAX,WI9,DELW                                              CONO2300
        READ (1,570) WT1,WT2,WT3                                                CONO2310
C                                                                               CONO2320
C --INPUT DESIRED SINGULAR VALUES (NOT USED) AND TYPE OF CONSTRAINTS---         CONO2330
C                                                                               CONO2340
```

```
      READ (1,580) SVMINI,SVMINO,RJ,NIDG
      W=WI9
      NC=1
C
C ----INPUT OPTIMIZER INPUT DATA (SEE ADS MANUAL)--------------------
C
      READ (1,760) IGRAD,NDV,NCON,ISTRAT,IOPT,IONED,IPRINT,INFO
C
C---INPUT THE PARAMETER VALUES AND THE MATRICES A,B,C,F,QW,RW--------
C
      READ (1,770) NROWA,NCOLA,NROWB,NCOLB,NROWC,NCOLC,NROWF,NCOLF,
     1NROWQ,NCOLQ,NROWR,NCOLR,NMU
      CALL READ (A,NROWA,NCOLA)
      CALL READ (B,NROWB,NCOLB)
      CALL READ (C,NROWC,NCOLC)
      CALL READ (F,NROWF,NCOLF)
      CALL READ (QW,NROWQ,NCOLQ)
      CALL READ (RW,NROWR,NCOLR)
      DO 13 I=1,NROWR
      DO 13 J=1,NCOLR
      RWSP(I,J)=RW(I,J)
13    CONTINUE
C
C---SPECIFY THE DESIGN VARIABLES WITHIN THE Q MATRIX--------
C     IF R OR BOTH Q AND R IS TO BE VARIED, MODIFICATION
C     IS REQUIRED FOR THE FOLLOWING BLOCK
C
      DO 10 J=1,NROWQ
      READ (1,780) (IQW(J,K),K=1,NCOLQ)
10    CONTINUE
C
C -------SET THE DESIGN VARIABLE BOUNDS------------------------------
C
      DO 20 J=1,NDV
      READ (1,790) VLB(J),VUB(J)
20    CONTINUE
C
C-----    READ THE DESIRED EIGENVALUES ------------------------------
C
      DO 60 I=1,NMU
      READ (1,860) REALMU(I),IMAGMU(I)
60    CONTINUE
      DO 70 I=1,NMU
      MU(I)=CMPLX(REALMU(I),IMAGMU(I))
70    CONTINUE
C -------- SORT THE INPUT EIGENVALUES USING IMSL ROUTINE VSRTR-------
C
      DO 80 I=1,NMU
```

```
      IR(I)=I                                                           CON02830
80    CONTINUE
      CALL VSRTR (REALMU,NMU,IR)                                        CON02840
      DO 90 J=1,NMU                                                     CCN02850
      K=IR(J)                                                           CCN02860
      OMU(J)=MU(K)                                                      CON02870
90    CONTINUE
      JJ=NMU-1                                                          CCN02880
      DO 110 J=1,JJ                                                     CON02890
      IF (REAL(OMU(J)).NE.REAL(OMU(J+1))) GO TO 100                     CON02900
      IF (AIMAG(OMU(J)).LT.AIMAG(OMU(J+1))) GO TO 100                   CON02910
      TEMP1=CMU(J)                                                      CCNC2920
      OMU(J)=CMU(J+1)                                                   CON02930
      OMU(J+1)=TEMP1                                                    CONC2940
100   CONTINUE                                                          CCN02950
110   CONTINLE                                                          CCN02960
C    ********************************************************************CON02970
C                                                                       CON02980
C         WRITE THE INPUT DATA FOR CHECKING AND REFERENCE               CONC2990
C                                                                       CON03C00
C    ********************************************************************CON03C10
C     WRITE (6,800) KODE                                                CON03C20
C     WRITE (6,720) WMAX,WI9,DELW                                       CCN03C30
C     WRITE (6,590) WT1,WT2,WT3                                         CON03C40
C     WRITE (6,60C) SVMINI,SVMINO,RJ,NIDG                               CCN03C50
C     WRITE (3,610) NROWA,NMU                                           CON03C60
      WRITE (6,810)                                                     CON03C70
      CALL WRITE (A,NROWA,NCOLA)                                        CCN03C80
      WRITE (6,820)                                                     CON03C90
      CALL WRITE (B,NROWB,NCOLB)                                        CON03100
C     WRITE (6,830)                                                     CCN03110
C     CALL WRITE (C,NROWC,NCOLC)                                        CON03120
      WRITE (6,841)                                                     CON03130
      CALL WRITE (QW,NROWQ,NCOLQ)                                       CON03100
      WRITE (6,832)
      DO 113 J=1,NROWQ                                                  CCN02530
      WRITE (6,*) (IQW(J,K),K=1,NCOLQ)                                  CON02540
113   CONTINUE
C     CALL WRITE (IQW,NROWQ,NCOLQ)                                      CCN03140
      WRITE (6,831)                                                     CON03110
      CALL WRITE (RW,NROWR,NCOLR)                                       CCN03120
      WRITE (6,840)                                                     CON03130
      CALL WRITE (F,NROWF,NCOLF)                                        CON03140
C
C ------SINGULAR VALUE ANALYSIS ONLY --------------------------------
C
      IF (KODE.EQ.2.AND.KONTRL.EQ.2) GO TO 531
C
```

```
      WRITE (6,850)                                                      CCN03150
      CALL CVECWR (MU,NMU)                                               CCN03160
C                                                                        CCN03730
C -----COMPUTE,SORT,NORMALISED  THE EIG-VALUES AND VECTORS-----------     CCN03740
C    OF THE PLANT MATRIX A                                               CCN03750
C
      CALL EIGRF (A,NROWA,10,2,EIG,Z,10,WK,IER)                          CCN03770
      DO 181 I=1,NROWA
      DO 183 J=1,NROWA
      WORK(J)=Z(J,I)
 183  CONTINUE
      SNCRM=SCNRM2(NROWA,WORK,1)
      SNCRM=1./SNCRM
      CALL CSSCAL(NROWA,SNCRM,WORK,1)
      DO 185 K=1,NROWA
      Z(K,I)=WCRK(K)
 185  CONTINUE
 181  CONTINUE
C-----------------------------------------------------------------
C
C      BEGIN CF TRANSFORMATION BLOCK
C
C-----------------------------------------------------------------
C
C  TRANSFORMATION MATRIX: 'OZ' (REAL EIG) & 'OZ*LX' (COMPLEX EIG)
C
C-----------------------------------------------------------------
      DO 190 J=1,NROWA                                                   CCN03780
      REIG(J)=REAL(EIG(J))                                               CCN03790
      IEIG(J)=AIMAG(EIG(J))
C ------- IF A HAS COMPLEX ROOT ICOMP=1 --------------------------
      IF (AIMAG(EIG(J)).NE.0.0) THEN
        ICOMP=1
C ---------INITIALIZE LX MATRIX ---------------------------------
        DO 197 K=1,NROWA
        DO 197 L=1,NCOLA
        LX(K,L)=0.0
        LX(K,K)=1.0
 197    CONTINUE
        CALL CPLEQU ( LX,LXI,NROWA,NCOLA)
        CALL CPLEQU ( LX,ZLI,NROWA,NCOLA)
        LX(1,1)=0.5
        LX(1,2)=(0.0,-0.5)
        LX(2,1)=0.5
        LX(2,2)=(C.0,0.5)
        CALL CPLEQU ( LX,TEMPX,NROWA,NCOLA)
        CALL LEQTIC(TEMPX,NROWA,10,LXI,NROWA,10,0,WA,IER)
C
```

```
      END IF
C
 190  CONTINUE                                                      CCNO3800
C--------END OF DO LOOP ---------------------------------------------
       WRITE (6,*)'** ICOMP **** ',ICOMP
       WRITE (6,*) '           '
       WRITE (6,*) '***AUX TRANSFORMATION L ***'
      CALL CWRITE( LX,NROWA,NCOLA)
      DO 200 J=1,NROWA                                              CCNO3810
 200  KEY(J)=J                                                      CCNO3820
C     CALL VSRTR (REIG,NROWA,KEY)                                   CONO3830
C     **** SORTING BY ALGEBRA OR ABS VALUE ****
      CALL VSRTP (IEIG,NRCWA,KEY)
      DO 220 J=1,NCOLA                                              CCNC3840
      K=KEY(NROWA+1-J)                                              CCNO3850
      OEIG(J)=EIG(K)                                                CCNO3860
      DO 210 KK=1,NROWA                                             CCNO3870
      OZ(KK,J)=Z(KK,K)                                              CCNO3880
      TEMPX(KK,J)=Z(KK,K)
 210  CONTINUE                                                      CCNO3890
 220  CONTINUE                                                      CONC3900
C    ----------------------------------------------------------------
C
C    ANY EIGENVECTCR ASSIGNMENT ROUTINE MAY BE INERTED HERE IN PLACE OF
C    THE EIGENVECTOR MATRIX OZ.
C
C    ----------------------------------------------------------------
      WRITE (6,*) '*** EIGENVECTOR OF A ***'
      CALL CPLREA(OZ,TEMPR,NROWA,NCOLA)
      CALL  WRITE(TEMPR,NROWA,NCOLA)
      DO 222 I=1,NROWA
      DO 222 J=1,NCOLA
      OZI(I,J)=0.C
      OZI(I,I)=1.0
 222  CONTINUE
      CALL CPLEQU (OZI,OZTI,NROWA,NCOLA)
      CALL CPLECU ( OZ,TEMPX2,NROWA,NCOLA)
C     LEQTIC DESTCRY OZ
      CALL LECTIC(TEMPX,NROWA,10,OZI,NROWA,10,0,WA,IER)
      CALL LECTIC(TEMPX2,NROWA,10,OZTI,NROWA,10,0,WA,IER)
      CALL CPLXCV (AX,A,NROWA,NCOLA)                                CONO3590
      CALL CPLXCV (BX,B,NROWB,NCCLB)                                CONO3590
      CALL CPLXCV (QWX,QW,NROWQ,NCOLQ)                              CONO3590
C
C
C ------------ TRANSFORM AX (COMPLEX EIG VAULE ) -----------------------
C
      IF (ICOMP.EC.1) THEN
```

```
C
      CALL CMATML (OZ,LX,NROWA,NCOLA,NROWA,TEMPX2)
      CALL CMATML (AX,TEMPX2,NROWA,NCOLA,NROWA,TEMPX)
      CALL CMATML (OZI,TEMPX,NROWA,NCOLA,NROWA,TEMPX2)
      CALL CMATML (LXI,TEMPX2,NROWA,NCOLA,NROWA,AX)
C
C -------------- TRANSFORM BW --------------------------------
C
      CALL CPLECU (BX,TEMPX,NROWB,NCOLB)
      CALL CMATML (OZ,LX,NROWA,NCOLA,NROWA,TEMPX2)
      CALL LECT1C(TEMPX2,NROWA,10,ZLI,NROWA,10,0,WA,IER)
      CALL CMATML (ZLI,TEMPX,NROWA,NRCWA,NCOLB,BX)
C
C ------------- TRANSFORM CWX -------------------------------------
C
      CALL CMATML (OZ,LX,NROWA,NCOLA,NROWA,TEMPX2)
      CALL CMATML (QWX,TEMPX2,NROWQ,NCOLQ,NROWA,TEMPX)
      CALL CMATMT (TEMPX2,TEMPX,NROWC,NCOLQ,NROWA,QWX)
C
      ELSE
C
C  --------- TRANSFORM AX (REAL EIG VALUE)-----------------------
C
      CALL CMATML (AX,OZ,NROWA,NCOLA,NROWA,TEMPX)
      CALL CMATML (OZI,TEMPX,NROWA,NCOLA,NROWA,AX)
C
C  ---------- TRANSFORM BX ----------------------------------------
C
      CALL CPLECU (BX,TEMPX,NROWB,NCOLB)
      CALL CMATML (OZI,TEMPX,NROWA,NRCWA,NCOLB,BX)
C
C ----------TRANSFORM CWX ------------------------------------------
C
      CALL CMATML (QWX,OZ,NROWQ,NCOLC,NROWA,TEMPX)
      CALL CMATMT (OZ,TEMPX,NROWQ,NCOLQ,NROWA,QWX)
      END IF
C  --------------END OF IF THEN ELSE BLOCK --------------------------
C
C   -------CUTPUT TRANSFCRMED A,B,Q -------------------------------
      WRITE (6,*) ' *** TRANSFORMED A MATRIX *** '
      CALL CPLREA(AX,TEMPR,NROWA,NCOLA)
      CALL  WRITE(TEMPR,NROWA,NCOLA)
      WRITE (6,*) '*** TRANSFORMED Q MATRIX ***'
      CALL CPLREA(QWX,TEMPR,NROWC,NCOLQ)
      CALL  WRITE(TEMPR,NROWQ,NCOLQ)
      WRITE (6,*) '*** TRANSFORMED B MATRIX ***'
      CALL CPLREA(BX,TEMPR,NRCWB,NCOLB)
      CALL  WRITE(TEMPR,NRCWB,NCOLB)
```

```
C
C   ---------------- END OF TRANSFORMATION BLOCK ---------------------
C
C
C  --FORMULATE THE DESIGN VARIABLE X FROM THE STARTING Q---------------  CON02610
C                                                                        CCN02620
      DO 50 J=1,NROWQ                                                    CON02630
      DO 40 K=1,NCOLQ                                                    CONC2640
      IF (IQW(J,K).EQ.0) GO TO 30                                        CON02650
      KK=IQW(J,K)                                                        CON02660
      X(KK)=REAL(QWX(J,K))                                               CONC2670
30    CONTINUE                                                           CONC2680
40    CONTINUE                                                           CCN02690
50    CONTINUE                                                           CCN02700
C------------------------------------------------------------------
      CALL CPLXCV (RWX,RW,NROWR,NCOLR)                                   CCN03600
C
C   ------- CALCULATE R**-1/2 -------------------------------------
C           USING LINPACK ROUTINE AND WURZEL ROUTINE
C
      CALL SPCCO(RW,10,NCCLR,RCOND,Z1,INFC1)
      CALL SPCDI(RW,10,NCOLR,DET,1)
      DO 131 I=1,NROWR
      DO 131 J=1,NROWR
      RW(J,I)=RW(I,J)
 131  CONTINUE
C   --------COMPUTE SQUARE ROOT OF MATRIX R  --------------------------
      CALL WURZEL(RW,TEMPR,NROWR,0.99,1.E-6,10)
      CALL CPLXCV (RWSQX,TEMPR,NROWR,NCOLR)                              CCN03590
      WRITE (6,*) '****SQUARE ROOT INVERSE OF R ***'
      CALL WRITE(TEMPR,NRCWR,NROWR)
C-------------------------------------------------------------------
C
C          *******  ENTER OPTIMIZATION BLOCK HERE*******
C                                                                        CON03170
C    IF KONTRL SET GREATER THAN 0 THE FIRST PASS JUMPS THE OPTIMIZER     CCN03180
C                                                                        CCN03190
      IF (KCNTRL.GT.0) GO TO 130                                         CON03200
C  ------------------------------------------------------------------CON03210
C                                                                        CON03220
C        CALL THE ADS OPTIMIZER TO SOLVE FOR THE BEST Q'S                CCN03230
C                                                                        CON03240
C  ------------------------------------------------------------------CON03210
C                                                                        CCN03260
C    THIS FIRST CALL SETS THE SCALE FLAG SEE ADS MANUAL                  CCN03270
      INFO=-2                                                            CCN03280
      CALL ADS (INFO,ISTRAT,IOPT,IONED,IPRINT,IGRAD,NDV,NCON,X,VLB,VUB,OCCN03290
     1BJ,G,IDG,NGT,IC,DF,XA,NRA,NCA,WK1,NRWK,IWK,NRIWK)                CON03300
```

```
      WK(3)=-C.05
C   -----IWK(2) = 0    IMPLIES NO SCALING -------------------------------
      IWK(2)=0                                                          CCN03310
C ----------------------------------------------------------------------CCN03210
C                                                                       CCN03330
C       THIS CALL STARTS THE OPTIMIZATION PROCESS                       CCN03340
C ----------------------------------------------------------------------CON03210
120   CALL ADS (INFO,ISTRAT,IOPT,IONED,IPRINT,IGRAD,NDV,NCON,X,VLB,VUB,OCCN03360
     1BJ,G,IDG,NGT,IC,DF,XA,NRA,NCA,WK1,NRWK,IWK,NRIWK)                 CON03370
      IF (INFC.EQ.0) GO TO 480                                          CON03380
      IF (INFC.GT.1) GO TO 470                                          CCN03390
C ----------------------------------------------------------------------CCN03210
C                                                                       CCN03410
C     THIS PORTION OF THE PROGRAM DOES THE ACTUAL COMPUTATION           CCN03420
C                                                                       CON03430
C ----------------------------------------------------------------------CCN03210
130   DO 160 J=1,NROWQ                                                  CON03450
      DO 150 K=1,NCOLQ                                                  CCN03460
C                                                                       CCN03470
C----EQUATE DESIGN VECTOR X WITH THE DESIRED SYSTEM MATRIX------------- CON03480
C                                                                       CON03490
      IF (IQW(J,K).EQ.0) GO TO 140                                      CCN03500
      KK=IQW(J,K)                                                       CON03510
      QWX(K,J)= X(KK)
      QWX(J,K)= X(KK)                                                   CCN03520
140   CONTINUE                                                          CON03530
150   CONTINUE                                                          CCN03540
160   CONTINUE                                                          CON03550
C                                                                       CCN03560
      CALL CPLXCV (CX,C,NROWC,NCOLC)                                    CCN03590
      CALL CPLXCV (FX,F,NROWF,NCOLF)                                    CON03590
C ----------------------------------------------------------------------CCN03210
C
C   MAIN LOOP TO COMPUTE THE DET (1+R**(-1/2)G(-S)**T *Q *G(S)R**(-1/2))
C   TO BE USED IN THE OPTIMIZER'S OBJECTIVE FUNCTION
C
C ----------------------------------------------------------------------CON03210
C
      DO 263 I=1,NMU
C
C   --------- PLANT1 COMPUTE G(S) WHERE G=C(SI-A)**(-1)*B-------------
C             WHERE S IS THE DESIRED POLE LOCATION
C
      CALL PLANT1(GPS,AX,BX,CX,MU(I),NROWA,NCOLA,NROWB,NCOLB,NROWC,
     1NCOLC)
C   --------- PLANT1 COMPUTE G(-S) WHERE G=C(SI-A)**(-1)*B-------------
C             WHERE S IS THE DESIRED POLE LOCATION
C
```

```
      CALL PLANT1(GMS,AX,BX,CX,-MU(I),NROWA,NCOLA,NROWB,NCOLB,NROWC,
     1NCOLC)
      CALL CMATML (GPS,RWSQX,NROWA,NCOLA,NCOLR,GPSRSQ)               CCNO4240
      CALL CMATML (QWX,GPSRSQ,NROWQ,NCOLQ,NCOLR,QGR)                 CCNO4240
      CALL CMATMT (GMS,QGR,NROWA,NCOLA,NCOLR,GQGR)                   CCNO4240
      CALL CMATML (RWSQX,GQGR,NROWR,NCOLR,NCOLR,RGQGR)               CCNO4240
C ------------------------------------------------------------------
C       |I+R**-(1/2)|G(-S)**T|Q|G(S)|R**-(1/2)||
C ------------------------------------------------------------------
      DO 911 J=1,NROWR
      DO 913 K=1,NCOLR
      XXI(J,K)=0.0
      XXI(J,J)=1.0
      XIRQR(J,K)=XXI(J,K)+RGQGR(J,K)
 913  CONTINUE
 911  CONTINUE
C  ------- FIND DETERMINANT TO BE USED IN THE OBJECTIVE FUNCTION ---
C          USING LINPACK ROUTINE
C
      CALL CGECO(XIRQR,10,NCOLR,IPVT,RCOND,WORK)
      CALL CGEDI(XIRQR,10,NCOLR,IPVT,DET,WORK,10)
      DETERM(I)=DET(1)*10.**REAL(DET(2))
C
253   CONTINUE
C ---------------END OF DO LOOP ---------------------------------
C                                                                    CCNO5000
C--FORM QUANTITES TO USE IN THE COST CRITERIA OF OPTIMIZER----------  CCNO5010
C EACH REAL POLE REQUIRED ONE COST, EACH COMPLEX PAIR POLE REQUIRED
C TWO COST(COST1 AND COST2)
C
      COST1=     ((REAL(DETERM(1)))**2+(AIMAG(DETERM(1)))**2)
      COST2=     ((REAL(DETERM(2)))**2+(AIMAG(DETERM(2)))**2)
      COST3=     ((REAL(DETERM(3)))**2+(AIMAG(DETERM(3)))**2)
C     WRITE(6,*) 'X',COST1,DETERM(1),X(1)
C
C                                                                    CCNO5130
C---THE OBJECTIVE FUNCTION IS INSERTED HERE------------------------  CCNO5140
C
      OBJ= COST1+COST2+COST1*COST2
C
C     WRITE (6,*) 'OBJ         ',OBJ
C ----- CONSTRAINT EQUATION (IDG(3)=-2 FOR COMPLEX ROOTS)-----------
      IF (NCON.EQ.0) GO TO 460                                       CCNO5180
      DO 450 J=1,2                                                   CCNO5190
      IDG(J)=NIDG                                                    CCNO5200
450   CONTINUE                                                       CCNO5210
      IDG(3)=-2
C -----OTHER CONSTRAINTS CAN BE INTRODUCED HERE--------------------
```

```
      G(1)=SQRT((REAL(DETERM(1)))**2+(AIMAG(DETERM(1)))**2)-0.05         CON0524C
      G(2)= REAL(DETERM(1))-C.9                                          CCN05240
      G(3)=X(1)-X(2)
C ---------------------------------------------------------------
C     WRITE (6,*) 'CONSTRAINT',G(1),G(2),G(3),G(4)
460   IF (KONTRL.GT.0) GC TO 480                                         CCN05260
      GO TO 120                                                          CCN05270
470   CONTINUE                                                           CCN05280
C                                                                        CCN05290
C  THE GRADIENT COMPUTATION ROUTINE GOES HERE IF NEEDED                 CCN05300
C     GO TO 125                                                          CCN05310
480   CONTINUE                                                           CCN05320
C     WRITE (6,*) 'KONTRL3 ',KONTRL
C                                                                        CCN05330
C-----------------------------------------------------------------------
C
C    END OF OPTIMIZATION , OUTPUT THE STATE WEIGHTING MATRIX
C
C-----------------------------------------------------------------------
C***********************************************************************
C
      WRITE (6,880)                                                      CON05510
      CALL CPLREA(QWX,TEMPR,NROWQ,NCOLQ)
      CALL WRITE(TEMPR,NROWQ,NCOLQ)                                      CON05520
C
C ---------INVERSE TRANFORMATION OF Q MATRIX ---------------------------
C
      IF (ICOMP.EQ.1) THEN
         CALL CMATML (QWX,ZLI,NROWQ,NCOLQ,NROWA,TEMPX)
         CALL CMATMT (ZLI,TEMPX,NROWA,NCOLA,NROWA,QWX)
      ELSE
         CALL CMATML (QWX,CZI,NROWQ,NCOLQ,NROWQ,TEMPX)
         CALL CMATMT (CZI,TEMPX,NROWQ,NCOLQ,NROWQ,QWX)
      END IF
C ----------   FIND DETERMINANT OF QWX FOR CHECKING PURPOSE ----------
C                USING LINPACK ROUTINES
C
      CALL CPLEQU (QWX,TEMPX,NROWQ,NCOLQ)
      CALL CGECO(TEMPX,10,NCOLQ,IPVT,RCOND,WORK)
      CALL CGEDI(TEMPX,10,NCOLQ,IPVT,DET,WORK,10)
C     DETQW=DET(1)*10.**REAL(DET(2))
C     WRITE (6,*) ' '
C     WRITE (6,*) 'DETQW',DETQW
C-----------------------------------------------------------------------
      CALL CPLREA(QWX,QW,NROWQ,NCOLQ)
      WRITE (6,*) ' END OF POLE PLACEMENT ROUTINE '
      WRITE (6,*) '    '
      WRITE (6,*) ' THE Q REQUIRED FOR THIS ASSIGNMENT IS '
```

```
      WRITE (6,*) '   '
      CALL WRITE (QW,NROWC,NCOLQ)
C ----------- SINGULAR VALUE ANALYSIS ONLY ---------------------------
 531  IF (KODE.EQ.2.AND.KCNTRL.EQ.2) GO TO 532
      N2=2*NCOLA
C**********************************************************************
C
C   ---- THE REDUCE ROUTINE MATCHES THE VARIABLE FROM THE OPTPP --------
C        WITH THE MODIFIED VERSION OF THE INNER ROUTINE FROM
C        CPTSYS PROGRAM
C
C----------------------------------------------------------------------
C
C    THE SYSTEM MATRIX A,B,C,  AND Q, F(FROM CPTPP) ARE USED
C    BY THE INNER ROUTINE TO COMPUTE THE FEEDBACK GAIN F.
C    F CAN THEN BE USED IN THE SINGULAR VALUE ANALYSIS PORTION OF
C    THIS PROGRAM
C
C----------------------------------------------------------------------
      CALL REDUCE(A,B,C,QW,RWSP,DUMA,DUMB,DUMC,DUMQ,DUMR,DUMRI,
     1NROWA,NCOLB)
C----------------------------------------------------------------------
      CALL INNER (NCOLA,NCOLB,NROWC,N2,DUMA,DUMB,DUMC,DUMQ,DUMR,FBGC
     1,RM,AA,FRC,X1,GN,DUMRI)
      DO 533 I=1,NROWF
      DO 533 J=1,NCOLF
      F(I,J)=-FBGC(I,J)
 533  CONTINUE
 532  CONTINUE
      CALL CPLXCV (AX,A,NROWA,NCOLA)                                    CCN03590
      CALL CPLXCV (BX,B,NROWB,NCOLB)                                    CCN03590
      CALL CPLXCV (CX,C,NROWC,NCOLC)                                    CCN03590
      CALL CPLXCV (FX,F,NROWF,NCOLF)
C
C----- THE FINAL FEEDBACK GAIN MATRIX IS DISPLAY HERE-----------------
C
C
      WRITE (6,840)                                                     CCN03130
      CALL WRITE (F,NROWF,NCOLF)                                        CCN03140
C --------------------------------------------------------------------
C                                                                       CCN03630
C    COMPUTE THE A-B*F*C SYSTEM MATRIX FOR THE NEXT POLE ASSIGNMENT     CCN03640
C                                                                       CCN03650
C --------------------------------------------------------------------
      CALL MMUL (F,C,NROWF,NCOLF,NCOLC,FC)                              CCN03660
      CALL MMUL (B,FC,NROWB,NCOLB,NCOLC,BFC)                            CCN03670
      DO 180 I=1,NROWA                                                  CCN03680
      DO 170 J=1,NCOLA                                                  CCN03690
```

```
170   AMBFC(I,J)=A(I,J)-BFC(I,J)                                          CCN03700
180   CONTINUE                                                            CCN03710
      WRITE (6,890)                                                       CON05530
C ---------OUTPUT (A-BFC)  ---------------------------------------
      CALL WRITE (AMBFC,NROWA,NCOLA)                                      CCN05540
C                                                                         CCN05550
      IF (KODE.EQ.0) GO TO 555
C*********************************************************************
C  ----------------------------------------------------------------      CON04C50
C       SINGULAR VALUE ANALYSIS BEGIN HERE                                CCN04C60
C  ----------------------------------------------------------------      CCN04C50
C                                                                         CCN04C60
C  CALL THE PLANT TRANSFER MATRIX ROUTINE AND FORM THE SYSTEM            CCN04C70
C    RETURN DIFFERENCE MATRICES AS REQUIRED                              CCN04C80
C                                                                         CCN04C90
C ------------------------------------------------------------------     CON04100
      DEL=DELW                                                            CCN04110
      WII=WI9                                                             CCN04120
      NCNT=1                                                              CCN04130
      W=WI9                                                               CCN04140
      SMINAI=C.                                                           CCN04150
      SMINMI=C.                                                           CCN04160
      SMINAO=0.                                                           CCN04170
      SMINMO=C.                                                           CCN04180
      SMAXAI=C.                                                           CCN04190
      SMAXMI=C.                                                           CCN04200
      SMAXAO=C.                                                           CCN04210
      SMAXMO=C.                                                           CCN04220
      CALL CPLXCV (AX,A,NROWA,NCOLA)                                      CCN03590
      CALL CPLXCV (BX,B,NROWB,NCOLB)                                      CCN03590
      CALL CPLXCV (CX,C,NROWC,NCOLC)                                      CON03590
260   CALL PLANT (PLAN,AX,BX,CX,W,NROWA,NCOLA,NROWB,NCOLB,NROWC,NCOLC)   CCN04230
      CALL CMATML (FX,PLAN,NROWF,NCOLF,NCOLB,FXPLT)                       CCN04240
      CALL CMATML (PLAN,FX,NROWC,NCOLB,NCOLF,PLTFX)                       CON04250
      DO 280 J=1,NROWF                                                    CON04260
      DO 270 K=1,NCOLB                                                    CCN04270
270   XI(J,K)=0.0                                                         CCN04280
280   CONTINUE                                                            CCN04290
      DO 290 J=1,NROWF                                                    CCN04300
290   XI(J,J)=1.0                                                         CCN04310
      DO 310 J=1,NROWC                                                    CCN04320
      DO 300 K=1,NCOLF                                                    CON04330
      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                                                    CON04390
```

```
      DO 320 K=1,NCOLB                                                  CCN04400
  320 XIPFXP(J,K)=XI(J,K)+FXPLT(J,K)                                    CCN04410
  330 CONTINUE                                                          CCN04420
      CALL CPLEQU (XIPFXP,TEMPX,NROWF,NRCWF)
      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
C THIS ESTABLISHES THE RETURN DIFFERENCE MATRIX FOR MULT. CASE         CON04530
C
      CALL LEGT1C (FXPLT,NROWF,10,FXPLTI,NROWF,10,0,WA,IER)            CCN04540
C     CALL LEGT1C (PLTFX,NROWC,10,PLTFXI,NROWC,10,0,WA,IER)            CCN04550
      DO 390 J=1,NROWF                                                  CCN04560
      DO 380 K=1,NCOLB                                                  CON04570
      XIFPLI(J,K)=XI(J,K)+FXPLTI(J,K)                                   CCN04580
  380 CONTINUE                                                          CCN04590
  390 CONTINUE                                                          CCN04600
      DO 410 J=1,NROWC                                                  CCN04610
      DO 400 K=1,NROWC                                                  CON04620
      XIPLFI(J,K)=XXI(J,K)+PLTFXI(J,K)                                  CCN04630
  400 CONTINUE                                                          CCN04640
  410 CONTINUE                                                          CCN04650
C ----------------------------------------------------------------------CCN04660
C                                                                       CCN04670
C   DO SINGULAR VALUE DECOMP AND QUANTIFY ALL DESIRED SV'S              CCN04680
C                                                                       CCN04690
C-----------------------------------------------------------------------CON04700
      CALL CSVD (XIPFXP,10,10,NROWF,NROWF,0,NROWF,NROWF,SV,UI,VI)       CCN04710
      CALL CSVD (XIPPFX,10,10,NROWC,NROWC,0,NROWC,NROWC,SVAO,LO,VO)     CON04720
      CALL CSVD (XIFPLI,10,10,NROWF,NROWF,0,NROWF,NROWF,SVMI,UMI,VMI)   CCN04730
      CALL CSVD (XIPLFI,10,10,NROWC,NROWC,0,NROWC,NROWC,SVMO,UMO,VMO)   CCN04740
C
C-------- THIS PORTION  COMPUTES THE MAXIMUM EIG-VALUES OF THE ------
C         RETURN DIFFERENCE MATRIX WHICH GIVE THE UPPER BOUND
C         FOR THE SINGULAR VALUE
C
      CALL EIGCC(TEMPX,NRCWF,10,0,EIG,Z,10,WK,IER)
      DO 411 I=1,NROWF
      MEIG(I)=SQRT((REAL(EIG(I)))**2+(AIMAG(EIG(I)))**2)
      IF (I.EQ.1) THEN
         MEIG(I)=MEIG(I)
```

```
      ELSE
         IF (MEIG(I).LT.MEIG(I-1)) THEN
            MEIG(I)=MEIG(I)
         ELSE
            MEIG(I)=MEIG(I-1)
         END IF
      END IF
411   CONTINUE
C     CALL VSRTM (MEIG,NROWF)                                          CON02840
      MINEIG(NCNT)=MEIG(NROWF)                                         CON02850
C -------------------------------------------------------------------
      OMEGA(NCNT)=W                                                    CON04750
      SIGNM1(NCNT)=SV(NROWF)                                           CON04760
      SIGNM2(NCNT)=SV(NROWF-1)                                         CON04770
      SIGNMX(NCNT)=SV(1)                                               CON04780
      SVADMO(NCNT)=SVAD(NROWC)                                         CON04790
      SVADXO(NCNT)=SVAD(1)                                             CON04800
      SVMIM(NCNT)=SVMI(NROWF)                                          CON04810
      SVMIX(NCNT)=SVMI(1)                                              CON04820
      SVMMO(NCNT)=SVMO(NROWC)                                          CON04830
      SVMXO(NCNT)=SVMO(1)                                              CON04840
      SIGPRO(NCNT)=SQRT(SIGNM1(NCNT)*SIGNM2(NCNT))                     CON04850
      SMINAI=(SMINAI+(AMAX1(0.0,SVMINI-SIGNM1(NCNT)))**2)              CON04860
      SMINAO=(SMINAO+(AMAX1(0.0,SVMINO-SVADMO(NCNT)))**2)              CON04870
C---------------------------------------------------------------------
C   -COMPUTE THE OPEN LOOP TRANSFER FUNCTION OF THE RESULTING DESIGN--
C     FOR TWO INPUT SYSTEM ONLY-NEED MODIFICATION OTHERWISE.
C
      CALL PLANT(TEMPX,AX,BX,FX,W,NROWA,NCOLA,NROWB,NCOLB,NROWF,
     1NCOLF)
      LP11(NCNT)=10.*ALOG10((REAL(TEMPX(1,1)))**2+(AIMAG(TEMPX(1,1)))**2
     1+0.000000)
      LP12(NCNT)=10.*ALOG10((REAL(TEMPX(1,2)))**2+(AIMAG(TEMPX(1,2)))**2
     1+0.000000)
      LP21(NCNT)=10.*ALOG10((REAL(TEMPX(2,1)))**2+(AIMAG(TEMPX(2,1)))**2
     1+0.000000)
      LP22(NCNT)=10.*ALOG10((REAL(TEMPX(2,2)))**2+(AIMAG(TEMPX(2,2)))**2
     1+0.000000)
C---------------------------------------------------------------------
C                                                                      CON04880
C   COMPUTE LOOP FOR ALL DESIRED FREQUENCY BANDWIDTH                   CON04890
C                                                                      CON04900
      W=W+DEL                                                          CON04910
      WD=10.0*WII                                                      CON04920
      IF (W.LT.WD) GO TO 420                                           CON04930
      DEL=10.0*DEL                                                     CON04940
      WII=10.0*WII                                                     CON04950
420   NCNT=NCNT+1                                                      CON04960
```

```
      IF (W.LE.WMAX) GO TO 260
      MAXN=NCNT-1
C
C------- OUTPUT THE OPTIMIZED SINGULAR VALUE DATA--------------
C
      WRITE (6,730)
C     WRITE (3,660) MAXN
      DO 540 J=1,MAXN
C     WRITE (3,670) OMEGA(J),SIGNM1(J),SIGNM2(J),SIGNMX(J),SIGPRO(J)
  540 WRITE (6,740) OMEGA(J),SIGNM1(J),MINEIG(J),SIGNMX(J),SIGPRO(J)
      WRITE (6,680)
      DO 550 J=1,MAXN
C     WRITE (3,690) SVADMO(J),SVADXO(J),SVMIM(J),SVMIX(J),SVMMO(J),SVMXO
C    1(J)
  550 WRITE (3,740) OMEGA(J),SIGNM1(J),LP11(J),LP12(J),LP21(J),LP22(J)
C550  WRITE (6,700) OMEGA(J),SVADMO(J),SVADXO(J),SVMIM(J),SVMIX(J),SVMMO
C    1(J),SVMXO(J)
  555 IF (KONTRL.EQ.0.OR.KONTRL.GT.1) GO TO 560
C     WRITE (6,*) 'KONTRL',KONTRL
      KONTRL=0
      GO TO 120
  560 WRITE (6,*) 'END OF OPTPP ANALYSIS '
      STOP
C*********************************************************************
C ------------------------------------------------------------------
C      FORMAT STATEMENTS
C ------------------------------------------------------------------
  570 FORMAT (3F10.0)
  580 FORMAT (3F10.0,I5)
  590 FORMAT (/,10H WEIGHT 1=,F10.5,9HWEIGHT 2=,F10.5,9HWEIGHT 3=,F10.5,
     1/)
  600 FORMAT (/,9H SVMINI= ,F10.5,8H SVMINO=,F10.5,5H  RJ=,F10.5,6H  IDG
     1=,I5,/)
  610 FORMAT (2I5)
  620 FORMAT (/,28H *** THE INITIAL DATA IS ***,/)
  630 FORMAT (/,41H *** THE OPTIMIZER COMPUTED OUTPUT IS ***,/)
  640 FORMAT (2E12.5)
  650 FORMAT (//,34H  **** THE OBJECTIVE FUNCTION IS =,F10.5,6H  ****,/)
  660 FORMAT (I5)
  670 FORMAT (5E12.5)
  680 FORMAT (/,2X,7H  FREQ ,5X,6HSVADMO,4X,6HSVADXO,4X,5HSVMIM,4X,5HSVM
     1IX,4X,5HSVMMO,4X,5HSVMXO,/)
  690 FORMAT (6E12.5)
  700 FORMAT (7F10.5)
  710 FORMAT (3F10.0)
  720 FORMAT (/,6H WMAX=,F10.5,1X,10HINIT FREQ=,F10.5,1X,10HFREQ STEP=,F
     110.5,//)
  730 FORMAT (//,10H FREQUENCY,5X,13HMIN ADD IN SV,4X,13H2ND ADD IN SV,3
```

```
     1X,13HMAX ADC IN SV,2X,18HMIN PROD ACD IN SV)
740   FORMAT (6(F10.5,1X))
750   FORMAT (I5,I5)
760   FORMAT (8I5)
770   FORMAT (13I2)
780   FORMAT (8I2)
790   FORMAT (2F10.0)
800   FORMAT (1H1,///,12H<<<<< RUN # ,I5,6H >>>>>,////)
810   FORMAT (27H *** THE A PLANT MATRIX ***,////)
820   FORMAT (//,35H ***  THE B CONTROL INPUT MATRIX ***,////)
830   FORMAT (//,33H ***  THE C OBSERVATION MATRIX ***,////)
831   FORMAT (//,33H ***  THE R WEIGHTING   MATRIX ***,////)
832   FORMAT (//,33H ***  THE DESIGN VARIABLE(Q) X ***,////)
840   FORMAT (//,38H ***  THE STARTING/FINAL  F  MATRIX ***,////)
841   FORMAT (//,40H ***  THE STARTING Q WEIGHTING MATRIX ***,////)
850   FORMAT (//,44H ***  THE ORDERED COMPLEX EIGENVALUES (INPUT),///)
860   FORMAT (2F10.0)
870   FORMAT (//,37H ***  THE ORDERED COMPUTED EIGENVALUES,///)
880   FORMAT (//,29H ***  THE OPTIMAL C MATRIX     ,///)
890   FORMAT (//,37H ***  THE AUGMENTED A-B*F*C MATRIX ***,////)
900   FORMAT (8F10.4)
      END
C**********************************************************************
C
C     *********   THE END OF THE MAIN PROGRAM (OPTPP)   ***********
C
C**********************************************************************
```

```
C*****************************************************************
C     BESIDE THE STANDARD IMSL ROUTINES, THE FOLLOWINGS ARE REQUIRED:
C     EISPACK ROUTINES:
C                          BALANC, ORTHES, ORTRAN, HQR2
C
C     LINPACK ROUTINES:
C                          SPOCO, SPODI, CGECO, CGEDI
C
C*****************************************************************
C-----------------------------------------------------------------OPTC3870
C
C     THIS ROUTINE COMPUTE THE FEEDBACK GAIN MATRIX F FOR THE LINEAR     OPTC3830
C     QUADRATIC REGULATOR TYPE OF PROBLEMS GIVEN SYSTEM MATRIX A, B, C   OPTC3840
C     AND THE STATE AND CONTROL WEIGHTING MATRICES                       OPTC3850
C
C-----------------------------------------------------------------OPTC3870
      SUBROUTINE INNER (NS,NC,NO,N2,BA,G1,HO,AY,B1,FBGC,RM,AA,PRC,X1
     1,GN,B1I)
C-----------------------------------------------------------------OPTC5960
      IMPLICIT REAL*4(A-H,O-Z)                                           OPTC6650
C-----------------------------------------------------------------OPTC6660
      INTEGER NS,NC,NO,MH,M,NSQ
      REAL*4 E1(NC,NC),BA(NS,NS),CWI(10),WNORM(10,10),WNORMI(10,10),     OPTC6670
     1CWR(10),FBGC(NC,NS),G1(NS,NS),PRO(NS,NS),B1I(NC,NC)               OPTC6680
     2,WR(20),WI(20),W11(10,10),W21(10,10),X1(N2,N2)                    OPTC6690
     3,SC(10,10),GN(NS,NS),HO(NO,NS),D1(20),D2(20),D3(20),              OPTC6900
     4AA(NS,NS),RM(N2,N2),AY(NO,NO)                                      OPTC6910
C-----------------------------------------------------------------OPTC6940
C-----------------------------------------------------------------OPTC6970
      REAL*4 FMT(20)                                                     OPTC6980
      DATA LCA/10/
C ------THE FOLLOWING ARE USED FOR DEBURGING -------------------------
C     CALL WRITE (BA,NS,NS)
C     CALL WRITE (G1,NS,NS)
C     CALL WRITE (HO,NS,NS)
C     CALL WRITE (AY,NS,NS)
C     CALL WRITE (B1,NS,NS)
C     WRITE (6,*) 'RM',RM
C     WRITE (6,*) 'BA',BA
C     WRITE (6,*) 'G1',G1
C     WRITE (6,*) 'AY',AY
C     WRITE (6,*) 'B1',B1
C     WRITE (6,*)' B1I',B1I
C     WRITE (6,*) 'FBGC',FBGC
C-----------------------------------------------------------------OPTC6970
      NSQ=NS*NS                                                          OPTC7210
      MH=NS                                                              OPTC7220
      DO 220 I=1,NC
```

```
      DO 220 J=1,NC
      BII(I,J)=BI(I,J)
220   CONTINUE
      DO 221 I=1,N2
      DO 221 J=1,N2
      RM(I,J)=0.E0
 221  CONTINUE
      M=N2                                                              CPT07230
         DO 240 I=1,NO                                                  CPT08520
         DO 240 J=1,NS                                                  CPT08530
         DDD=0.E0                                                       CPT08540
         DO 230 K=1,NO                                                  CPT08550
         DDD=(AY(I,K)*HO(K,J))+DDD                                      CPT08560
 230     CONTINUE
         AA(I,J)=DDD                                                    CPT08570
 240     CONTINUE
C                                                                       CPT08600
         DO 260 I=1,NS                                                  CPT08610
         DO 260 J=1,NS                                                  CPT08620
         DO 260 K=1,NO                                                  CPT08630
         RM(I+MH,J)=RM(I+MH,J)+AA(K,I)*HO(K,J)                          CPT08640
C        WRITE (6,*) 'RM(I+MH,J)',RM(I+MH,J)
C
  260    CONTINUE
C -------------------------------------------------------------------
C---CALCULATION OF CONTROL GAINS:FORMATION OF CONTROL HAMILTONIAN-------CPT08710
C -------------------------------------------------------------------
C                                                                       CPT08720
C                                               ***F AND FT ARE THE OPEN LOOP    CPT08730
C    ---                       ---                 DYNAMICS MATRIX AND TRANSPOSE CPT08740
C   |      F        -GM*BI*GMT    |             ***BI IS NCXNC CONTROL WEIGHTING CPT08750
C   |                             |                MATRIX                        CPT08760
C   |                             |             ***A IS THE NSXNS STATE WEIGHTINGCPT08770
C   |                             |                MATRIX                        CPT08780
C   |      A              -FT     |                                              CPT08790
C    ---                       ---              ***GM IS THE NSXNC CONTROL       CPT08800
C                                                  DISTRIBUTION MATRIX           CPT08810
C-----------------------------------------------------------------------CPT08820
C   ---INVERSE OF BI (IMSLSP)OR MINV-----------------------------
C     WRITE(6,*)'AA ',BII
C     K=NC*NC
C     CALL MINV(K,BII,NC,DDD,D1,D2)
      CALL LINV2F(BI,NC,NC,BII,1,D3,IER)
C     WRITE(6,*)'BII',BII
C  -- BI*GT -----------M12 BLOCK ------------------------------------
      DO 370 I=1,NC                                                     CPT08830
      DO 370 J=1,NS                                                     CPT08840
      PRC(I,J)=0.0                                                      CPT08850
```

```
      DO 370 K=1,NC
      PRC(I,J)=PRO(I,J)+B1I(I,K)*G1(J,K)
  370 CONTINUE
      DO 380 I=1,MH                                                     CPT08860
      DO 380 J=1,MH                                                     CPT08870
      RM(I,J+MH)=0.0                                                    CPT08880
      DO 380 K=1,NC                                                     CPT08890
      RM(I,J+MH)= (RM(I,J+MH)-G1(I,K)*PRC(K,J))                         CPT08900
  380 CONTINUE
C-------------------2NX2N HAMILTONIAN MATRIX----------------------------CPT08910
C----------------DIAGONAL BLOCKS---M11 AND M22--------------------------CPT08920
C     WRITE (6,*) 'BA'
C     CALL WRITE (BA,2,2)
      DO 390 I=1,MH                                                     CPT08930
      DO 390 J=1,MH                                                     CPT08940
      RM(I,J)=0.E0
      RM(I,J)=BA(I,J)                                                   CPT08950
      RM(I+MH,J+MH)=-BA(J,I)                                            CPT08960
C---------------------------M21 BLOCK-----------------------------------CPT08970
      RM(I+MH,J)=-RM(I+MH,J)                                            CPT08980
  390 CONTINUE                                                          CPT09160
C-------------M12 BLOCK IS DEFINED IN LINE 430 ABOVE-------------------CPT08990
  400 CONTINUE                                                          CPT09000
C----------CALL EISPACK ROUTINES-----------------------------------------
  410 CALL BALANC (M,M,RM,LOW,IHIGH,D1)                                 CPT09040
      CALL ORTHES (M,M,LOW,IHIGH,RM,D2)                                 CPT09050
      CALL ORTRAN (M,M,LOW,IHIGH,RM,D2,X1)                              CPT09060
      CALL HQR2 (M,M,LOW,IHIGH,RM,WR,WI,X1,IERR)                        CPT09070
C     WRITE (6,*)'IERR',IERR
      IF (IERR .NE. 0) CALL EREXIT (M,RM,IERR)                          CPT09080
      CALL BALBAK (M,M,LOW,IHIGH,D1,M,X1)                               CPT09090
C------------DEBUG DIAGNOSTICS ON EULER-LAGRANGE EQUATIONS--------------CPT09100
  430 CONTINUE                                                          CPT09170
      IF (NOP.EQ.0) WRITE (6,1550)                                      CPT09190
C --------CALL RGAIN TO COMPUTE THE FEEDBACK GAIN --------------------
      CALL RGAIN (M,NS,NC,NCB,WR,WI,X1,GN,W11,RM,W21,D1,CWR,CWI,SC,MHS, CPT09220
     1D2)                                                               CPT09230
  450 CONTINUE                                                          CPT09260
  470 CONTINUE                                                          CPT09340
C-------------------CALCULATION OF FEEDBACK GAIN-------------------------CPT09350
C------------FEEDBACK GAINS---> U = -(BINVERSE)*GT*GN&-----------------CPT09360
C------------------------CALCULATE GT----------------------------------CPT09370
      DO 480 I=1,NS                                                     CPT09380
      DO 480 J=1,NS                                                     CPT09390
      PRO(I,J)=0.0                                                      CPT09400
      DO 480 K=1,NS                                                     CPT09410
  480 PRC(I,J)=PRC(I,J)+G1(K,I)*GN(K,J)                                 CPT09420
      DO 490 I=1,NC                                                     CPT09380
```

```
      DO 490 J=1,NS                                                    CPT09390
      FBGC(I,J)=0.0                                                    CPT09400
      DO 490 K=1,NC                                                    CPT09410
      FBGC(I,J)= FBGC(I,J)-BII(I,K)*PRO(K,J)                           CPT09420
  490 CONTINUE
C---------NORMALIZE AND PRINT OPT. REG. CLOSED LOOP EIGENSYSTEM----------CPT09450
      IWRITE=2                                                         CPT09460
      CALL CNORM (CWR,CWI,SC,NS,IWRITE,NSC,DDC,D1,D2,WNORM,WNCRMI,FBGC, CPT09470
     1AA,NC,NS)                                                        CPT09480
C---------------THE OPTIMUM FEEDBACK CONTROL GAINS----------------------CPT09530
  500 CONTINUE
      RETURN                                                           CPT13050
C-----------------------------------------------------------------------CPT13060
 1380 FORMAT (//,5X,45HOPEN LOOP DYNAMICS MATRIX.................F..,//)CPT13080
 1390 FORMAT (10(2X,0PD11.4))                                          CPT13090
 1400 FORMAT (//,5X,45HTHE CONTROL DISTRIBUTION MATRIX...........G..,//)CPT13100
 1410 FORMAT (//,5X,45HTHE CONTROL COST MATRIX...................B..,//)CPT13110
 1440 FORMAT (//,5X,45HMEASUREMENT SCALING MATRIX................H..,//)CPT13140
 1460 FORMAT (//,5X,45HOUTPUT COST MATRIX........................A..,//)CPT13160
 1480 FORMAT (//,25X,28H....DESTABILIZATION CASE....,//,10X,39HTHE FOLLOCPT13180
     1WING VALUES WILL BE ADDED DOWN,/,10X,49HTHE DIAGONAL OF THE "F" MACPT13190
     2TRIX TO DESTABILIZE IT.,/,10X,41HOPTIMAL GAINS FOR THE DESTABILIZECPT13200
     3D SYSTEM,/,10X,39HARE THEN USED AS FIXED SUBOPTIMAL GAINS,/,10X,28CPT13210
     4HFOR THE SYSTEM CALCULATIONS.,//)                                CPT13220
 1490 FORMAT (///43H PROGRAM TERMINATING DUE TO UNSTABLE SYSTEM)       CPT13230
 1500 FORMAT (//,5X,31HOPEN LOOP TRANSFER FUNCTIONS...)                CPT13240
 1510 FORMAT (//,5X,32H EULER-LAGRANGE SYSTEM MATRIX...,//)            CPT13250
 1520 FORMAT (//,5X,43HEIGENVALUES AND EIGENVECTORS OF THE 2N X 2N,/,5X,CPT13260
     145HEULER-LAGRANGE SYSTEM AFTER HQR2.............,//)             CPT13270
 1530 FORMAT (1X,1P,2E13.6)                                            CPT13280
 1540 FORMAT (1X)                                                      CPT13290
 1550 FORMAT (//,5X,41HEIGENSYSTEM OF OPTIMAL REGULATOR..........,//)  CPT13300
 1560 FORMAT (//,5X,41HEIGENSYSTEM OF OPTIMAL ESTIMATOR..........,//)  CPT13310
 1570 FORMAT (//,5X,39H EIGENVECTORS FROM RGAIN PRIOR TO CNORM,//)     CPT13320
 1580 FORMAT (//,2X,57HTHE OPTIMAL FEEDBACK GAIN CONTROL MATRIX...C=BINVCPT13330
     1*GT*S...,//)                                                     CPT13340
 1590 FORMAT (10(2X,1PE11.4))                                          CPT13350
 1600 FORMAT (//,5X,45HTHE CLOSED LOOP DYNAMICS MATRIX ......F-G*C..,//)CPT13360
 1610 FORMAT (///,60H   PROGRAM TERMINATING DUE TO UNSTABLE CLOSED LOOP CPT13370
     1   SYSTEM)                                                       CPT13380
 1640 FORMAT (1X,2X,1P,6D14.6)                                         CPT13420
 1690 FORMAT ((5(1X,1PD13.6)))                                         CPT13470
 1750 FORMAT (1X,1PD15.7,25X,D15.7)                                    CPT13530
C-------------------------------------------------------                CPT13640
 2000 FORMAT(5X,2E13.6)                                                CPT13650
C-----------------------------------------------------------------------CPT13660
      END
```

```
C*****************************************************************
C----------------------------------------------------------------
C    REDUCE 10*10 MATRIX TO MATCH VARIABLE BET. OPTPP AND INNER
C----------------------------------------------------------------
      SUBROUTINE REDUCE (A,B,C,Q,R,DUMA,DUMB,DUMC,DUMQ,DUMR,DUMRI,NS,
     1NC)
      REAL*4  A(10,10),B(10,10),C(10,10),F(10,10),Q(10,10),R(10,10)
      REAL*4  DUMA(NS,NS),DUMB(NS,NS),DUMC(NS,NS),
     1DUMQ(NS,NS),DUMR(NC,NC),DUMRI(NC,NC)
      INTEGER NS,NO,NC
      DO 10 I=1,NS
      DO 10 J=1,NS
      DUMA(I,J)=A(I,J)
      DUMB(I,J)=B(I,J)
      DUMC(I,J)=C(I,J)
      DUMQ(I,J)=Q(I,J)
 10   CONTINUE
      DO 30 I=1,NC
      DO 30 J=1,NC
      DUMR(I,J)=R(I,J)
      DUMRI(I,J)=R(I,J)
 30   CONTINUE
      RETURN
      END
C******************************************************************   CWROC010
C                                                                     CWROC020
C                                                                     CWROC040
C  *****************************************************************CWROC050
C-----------------------------------------------------------------
C        WRITE THE COMPLEX MATRIX                                     CWROC060
C-----------------------------------------------------------------
      SUBROUTINE CWRITE (A,N,K)                                       CWROC080
      COMPLEX *8A(10,10)                                              CWROC090
      INTEGER N,K,I,J                                                 CWROC100
      DO 10 I=1,N                                                     CWROC110
      WRITE (6,20) (A(I,J),J=1,K)                                     CWROC120
10    CONTINUE                                                        CWROC130
      RETURN                                                          CWROC140
20    FORMAT (8F12.5)                                                 CWROC150
      END                                                             CWROC160
C  ******************************************************************  CWROC170
C
C
C-----------------------------------------------------------------
C    WRITE THE COMPLEX VECTOR                                         CWROC180
C-----------------------------------------------------------------
      SUBROUTINE CVECWR (A,N)                                         CWROC200
      COMPLEX *8A(10)                                                 CWROC210
```

```
      INTEGER N,J
      WRITE (6,10) (A(J),J=1,N)
      RETURN
10    FORMAT (2F12.5)
      END
C    ***************************************************************
C
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
C-----------------------------------------------------------------
C      MATRIX MULTIPLICATION WITH COMPLEX MATRIX
C-----------------------------------------------------------------
      SUBROUTINE CMATML (R,T,M,LL,N,U)
      COMPLEX *8 R(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)=C.0
      DO 10 INDEX=1,LL
      U(I,J)=U(I,J)+R(I,INDEX)*T(INDEX,J)
10    CONTINUE
20    CONTINUE
30    CONTINUE
      RETURN
      END
C    ***************************************************************
C
C-----------------------------------------------------------------
C       REAL MATRIX MULTIPLICATION ROUTINE
C-----------------------------------------------------------------
      SUBROUTINE MMUL (R,T,M,LL,N,U)
      REAL*4 R(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)=C.0
      DO 10 INDEX=1,LL
```

```
      U(I,J)=U(I,J)+R(I,INDEX)*T(INDEX,J)
10    CONTINUE
20    CONTINUE
30    CONTINUE
      RETURN
      END
C   ********************************************************************
C
C-----------------------------------------------------------------------
C   CSVD ALGORITHM FOR COMPLEX MATRICES; ANY OF THE ROUTINES OF IMSL
C   OR LINPACK COULD ALSO BE USED WITH PROPER CAUTIONS
C-----------------------------------------------------------------------
      SUBROUTINE CSVD (A,MMAX,NMAX,M,N,P,NU,NV,S,U,V)
      COMPLEX A(MMAX,1),U(MMAX,1),V(NMAX,1)
      INTEGER M,N,P,NU,NV
      REAL S(1)
      COMPLEX Q,R
      REAL B(100),C(100),T(100)
      DATA ETA,TOL/1.5E-8,1.E-31/
      NP=N+P
      N1=N+1
C
C  HOUSEHOLDER REDUCTION
      C(1)=0.E0
      K=1
10    K1=K+1
C
C   ELIMINATION OF A(I,K),I=K+1,.....,M
      Z=0.E0
      DO 20 I=K,M
20    Z=Z+REAL(A(I,K))**2+AIMAG(A(I,K))**2
      B(K)=0.E0
      IF (Z.LE.TOL) GO TO 70
      Z=SQRT(Z)
      B(K)=Z
      W=CABS(A(K,K))
      Q=(1.E0,0.E0)
      IF (W.NE.0.E0) Q=A(K,K)/W
      A(K,K)=Q*(Z+W)
      IF (K.EQ.NP) GO TO 70
      DO 50 J=K1,NP
      Q=(0.E0,0.E0)
      DO 30 I=K,M
30    Q=Q+CONJG(A(I,K))*A(I,J)
      Q=Q/(Z*(Z+W))
      DO 40 I=K,M
40    A(I,J)=A(I,J)-Q*A(I,K)
50    CONTINUE
```

```
C
C     PHASE TRANSFORMATION
        Q=-CONJG(A(K,K))/CABS(A(K,K))
        DO 60 J=K1,NP
60      A(K,J)=Q*A(K,J)
C
C     ELIMINATION OF A(K,J),J=K+2,...,N
70      IF (K.EQ.N) GO TO 140
        Z=0.E0
        DO 80 J=K1,N
80      Z=Z+REAL(A(K,J))**2+AIMAG(A(K,J))**2
        C(K1)=0.E0
        IF (Z.LE.TOL) GO TO 130
        Z=SQRT(Z)
        C(K1)=Z
        W=CABS(A(K,K1))
        Q=(1.E0,0.E0)
        IF (W.NE.0.E0) Q=A(K,K1)/W
        A(K,K1)=Q*(Z+W)
        DO 110 I=K1,M
        Q=(0.E0,0.E0)
        DO 90 J=K1,N
90      Q=Q+CONJG(A(K,J))*A(I,J)
        Q=Q/(Z*(Z+W))
        DO 100 J=K1,N
100     A(I,J)=A(I,J)-Q*A(K,J)
110     CONTINUE
C
C     PHASE TRANSFORMATION
        Q=-CONJG(A(K,K1))/CABS(A(K,K1))
        DO 120 I=K1,M
120     A(I,K1)=A(I,K1)*Q
130     K=K1
        GO TO 10
C
C   TOLERENCE FOR NEGLIGIBLE ELEMENTS
140     EPS=0.E0
        DO 150 K=1,N
        S(K)=B(K)
        T(K)=C(K)
150     EPS=AMAX1(EPS,S(K)+T(K))
        EPS=EPS*ETA
C
C     INITIALIZATION OF U AND V
        IF (NU.EQ.0) GO TO 180
        DO 170 J=1,NU
        DO 160 I=1,M
160     U(I,J)=(0.E0,0.E0)
```

```
170   U(J,J)=(1.E0,0.E0)
180   IF (NV.EQ.0) GO TO 210
      DO 200 J=1,NV
      DO 190 I=1,N
190   V(I,J)=(0.E0,0.E0)
200   V(J,J)=(1.E0,0.E0)
C
C  QR DIAGONALIZATION
210   DO 380 KK=1,N
      K=N1-KK
C
C   TEST FOR SPLIT
220   DO 230 LL=1,K
      L=K+1-LL
      IF (ABS(T(L)).LE.EPS) GO TO 290
      IF (ABS(S(L-1)).LE.EPS) GO TO 240
230   CONTINUE
C
C   CANCELLATION OF E(L)
240   CS=0.E0
      SN=1.E0
      L1=L-1
      DO 280 I=L,K
      F=SN*T(I)
      T(I)=CS*T(I)
      IF (ABS(F).LE.EPS) GO TO 290
      H=S(I)
      W=SQRT(F*F+H*H)
      S(I)=W
      CS=H/W
      SN=-F/W
      IF (NU.EQ.0) GO TO 260
      DO 250 J=1,N
      X=REAL(U(J,L1))
      Y=REAL(U(J,I))
      U(J,L1)=CMPLX(X*CS+Y*SN,0.E0)
250   U(J,I)=CMPLX(Y*CS-X*SN,0.E0)
260   IF (NP.EQ.N) GO TO 280
      DO 270 J=N1,NP
      Q=A(L1,J)
      R=A(I,J)
      A(L1,J)=Q*CS+R*SN
270   A(I,J)=R*CS-Q*SN
280   CONTINUE
C
C   TEST FOR CONVERGENCE
290   W=S(K)
      IF (L.EQ.K) GO TO 360
```

```
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(H*H+F*F)                                                   CWR02400
      S(I-1)=W                                                          CWR02410
      CS=F/W                                                            CWR02420
      SN=H/W                                                            CWR02430
      F=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)
      R=A(I,J)
      A(I-1,J)=Q*CS+R*SN
340   A(I,J)=R*CS-Q*SN
350   CONTINUE
      T(L)=0.E0
      T(K)=F
      S(K)=X
      GO TO 220
C
C   CONVERGENCE
360   IF (W.GE.0.E0) GO TO 380
      S(K)=-W
      IF (NV.EQ.0) GO TO 380
      DO 370 J=1,N
370   V(J,K)=-V(J,K)
380   CONTINUE
C
C   SORT SINGULAR VALUES
      DO 450 K=1,N
      G=-1.E0
      J=K
      DO 390 I=K,N
      IF (S(I).LE.G) GO TO 390
      G=S(I)
      J=I
390   CONTINUE
      IF (J.EQ.K) GO TO 450
      S(J)=S(K)
      S(K)=G
      IF (NV.EQ.0) GO TO 410
      DO 400 I=1,N
      Q=V(I,J)
      V(I,J)=V(I,K)
400   V(I,K)=Q
410   IF (NU.EQ.0) GO TO 430
      DO 420 I=1,N
      Q=U(I,J)
      U(I,J)=U(I,K)
420   U(I,K)=Q
430   IF (N.EQ.NP) GO TO 450
      DO 440 I=N1,NP
      Q=A(J,I)
      A(J,I)=A(K,I)
440   A(K,I)=Q
450   CONTINUE
C
C   BACK TRANSFORMATION
```

```
      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,M                                                      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,M                                                      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                                                               CWR03370
C    *******************************************************************CWR03380
C
C-----------------------------------------------------------------------
C        READ A REAL MATRIX                                             CWR03390
C-----------------------------------------------------------------------
      SUBROUTINE READ (A,N,K)                                           CWR03410
      REAL*4 A(10,10)                                                   CWR03420
      INTEGER N,K                                                       CWR03430
      DO 10 I=1,N                                                       CWR03440
      READ (1,20) (A(I,J),J=1,K)                                        CWR03450
10    CONTINUE                                                          CWR03460
      RETURN                                                            CWR03470
```

```
20    FORMAT (5F12.0)                                                       CWR03480
      END                                                                   CWR03490
C   *******************************************************************    CWR03500
C
C-------------------------------------------------------------------------
C        WRITE A REAL MATRIX                                                CWR03510
C-------------------------------------------------------------------------
      SUBROUTINE WRITE (A,N,K)                                              CWR03530
      REAL*4 A(10,10)                                                       CWR03540
      INTEGER N,K                                                           CWR03550
      DO 10 I=1,N                                                           CWR03560
      WRITE (6,20) (A(I,J),J=1,K)                                           CWR03570
10    CONTINUE                                                              CWR03580
      RETURN                                                                CWR03590
20    FORMAT (6F12.5)                                                       CWR03600
      END                                                                   CWR03610
C   *******************************************************************    CWR03620
C
C
C-------------------------------------------------------------------------
C     CONVERT REAL MATRIX TO COMPLEX FORM                                   CWR03630
C-------------------------------------------------------------------------
      SUBROUTINE CPLXCV (X,A,N,K)                                           CWR03650
      COMPLEX *8X(10,10)                                                    CWR03660
      REAL*4 A(10,10)                                                       CWR03670
      INTEGER N,K                                                           CWR03680
      DO 20 I=1,N                                                           CWR03690
      DO 10 J=1,K                                                           CWR03700
10    X(I,J)=CMPLX(A(I,J),0.00)                                             CWR03710
20    CONTINUE                                                              CWR03720
      RETURN                                                                CWR03730
      END                                                                   CWR03740
C   *******************************************************************    CWR03620
C
C
C-------------------------------------------------------------------------
C     CONVERT REAL PART OF COMPLEX MATRIX TO REAL MATRIX                    CWR03630
C-------------------------------------------------------------------------
      SUBROUTINE CPLREA (X,A,N,K)                                           CWR03650
      COMPLEX *8X(10,10)                                                    CWR03660
      REAL*4 A(10,10)                                                       CWR03670
      INTEGER N,K                                                           CWR03680
      DO 20 I=1,N                                                           CWR03690
      DO 10 J=1,K                                                           CWR03700
10    A(I,J)=REAL(X(I,J))                                                   CWR03710
20    CONTINUE                                                              CWR03720
      RETURN                                                                CWR03730
      END                                                                   CWR03740
C   *******************************************************************    CWR03620
C
C
C-------------------------------------------------------------------------
```

```
C       EQUATE ONE COMPLEX MATRIX TO ANOTHER                                CWR03630
C------------------------------------------------------------------------
        SUBROUTINE CPLFQU (X,Y,N,K)                                         CWR03650
        COMPLEX *8X(10,10),Y(10,10)                                         CWR03660
C       REAL*4 A(10,10)                                                     CWR03670
        INTEGER N,K                                                         CWR03680
        DO 20 I=1,N                                                         CWR03690
        DO 10 J=1,K                                                         CWR03700
10      Y(I,J)=X(I,J)                                                       CWR03710
20      CONTINUE                                                            CWR03720
        RETURN                                                              CWR03730
        END                                                                 CWR03740
C    ****************************************************************       CWR03750
C
C------------------------------------------------------------------------
C          THIS ROUTINE WILL COMPUTE THE TRANSFER FUNCTION AS A              CWR03760
C           FUNCTION OF FREQUENCY FOR A GIVEN A, B, C SET OF                 CWR03770
C                             MATRICES                                      CWR03780
C                                                                           CWR03790
C------------------------------------------------------------------------
        SUBROUTINE PLANT (GP,AX,BX,CX,W,NRA,NCA,NRB,NCB,NRC,NCC)            CWR03810
        COMPLEX *8CX(10,10),AX(10,10),BX(10,10),GP(10,10),I(10,10),SIA(10,CWR03820
       110),SIAI(10,10),SIAIB(10,10),SI(10,10)                              CWR03830
        COMPLEX S                                                           CWR03840
        REAL*4 WA(100)                                                      CWR03850
        S=CMPLX(0.0,W)                                                      CWR03860
        DO 20 J=1,NRA                                                       CWR03870
        DO 10 K=1,NCA                                                       CWR03880
        SI(J,K)=0.0                                                         CWR03890
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
C------------------------------------------------------------------------
C         THIS ROUTINE IS INPUT FOR DE SOLVER                               CWR04050
C       WHEN DE SOLVER IS ADDED DIRECTLY TO CONXSV                          CWR04060
```

```
C-----------------------------------------------------------------------
      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
   40 AMBFCX(J)=AMBFCX(J)+AMBFC(J,K)*X(K)                                CWR04280
   30 CONTINUE                                                           CWR04290
C     WRITE(6,5501)(AMBFCX(J),J=1,N)                                     CWR04300
C5501 FORMAT(6F10.5)                                                     CWR04310
      DO 50 J=1,N                                                        CWR04320
   50 XDOT(J)=AMBFCX(J)+BPHI(J)                                          CWR04330
C     WRITE(6,5502)(XDOT(J),J=1,N)                                       CWR04340
C5502 FORMAT(6F10.5)                                                     CWR04350
      RETURN                                                             CWR04360
      END                                                                CWR04370
C   ****************************************************************** CWR03750
C
C
C-----------------------------------------------------------------------
C        THIS ROUTINE WILL COMPUTE G(S) OR G(-S)                         CWR03760
C        WHERE G(S) IS GIVEN BY C*(SI-A)**(-1)*B                         CWR03770
C        AND S IS THE DESIRED POLES LOCATIONS                            CWR03780
C-----------------------------------------------------------------------
      SUBROUTINE PLANT1(GP,AX,BX,CX,S,NRA,NCA,NRB,NCB,NRC,NCC)           CWR03810
      COMPLEX *8CX(10,10),AX(10,10),BX(10,10),GP(10,10),I(10,10),SIA(10,CWR03820
     110),SIAI(10,10),SIAIB(10,10),SI(10,10)                             CWR03830
      COMPLEX S                                                          CWR03840
      REAL*4 WA(100)                                                     CWR03850
C     S=CMPLX(W,0.0)                                                     CWR03860
      DO 20 J=1,NRA                                                      CWR03870
      DO 10 K=1,NCA                                                      CWR03880
      SI(J,K)=0.0                                                        CWR03890
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
C     WRITE (6,*) 'SIA'
      CALL CWRITE (SIA,NROWA,NCOLA)
      CALL LECT1C (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   ***********************************************************************  CWR00380
C
C------------------------------------------------------------------------
C       MATRIX MULTIPLICATION WITH COMPLEX MATRIX(A**T*B)               CWR00390
C------------------------------------------------------------------------
      SUBROUTINE CMATMT (R,T,M,LL,N,U)                                  CWR00410
      COMPLEX *8R(10,10),T(10,10),U(10,10)                              CWR00420
      INTEGER M,LL,N                                                    CWR00430
      DO 30 I=1,M                                                       CWR00440
      DO 20 J=1,N                                                       CWR00450
      U(I,J)=0.0                                                        CWR00460
      DO 10 INDEX=1,LL                                                  CWR00470
      U(I,J)=U(I,J)+R(INDEX,I)*T(INDEX,J)                               CWR00480
10    CONTINUE                                                          CWR00490
20    CONTINUE                                                          CWR00500
30    CONTINUE                                                          CWR00510
      RETURN                                                            CWR00520
      END                                                               CWR00530
C=======================================================================CPT13680
      SUBROUTINE RAPRNT (NMAX,M,N,L,A,IDIM,FMT)                         CPT13690
C=======================================================================CPT13680
      REAL*4 A(NMAX,N)                                                  CPT13700
      DIMENSION FMT(IDIM)                                               CPT13710
      NU=L                                                              CPT13720
      DO 20 NL=1,N,L                                                    CPT13730
      IF (NU.GT.N) NU=N                                                 CPT13740
      DO 10 I=1,M                                                       CPT13750
10    WRITE (6,FMT) (A(I,J),J=NL,NU)                                    CPT13760
      WRITE (6,30)                                                      CPT13770
20    NU=NU+L                                                           CPT13780
      RETURN                                                            CPT13790
30    FORMAT (1X)                                                       CPT13800
      END                                                               CPT13810
```

```
C===========================================================================
      SUBROUTINE RGAIN (M,NS,NC,NOB,WR,WI,VF,GN,W11,TCB,W21.LT,C,CI,CT,M
     1HS,MT)
C=============================================================  ===========
      IMPLICIT REAL*4 (A-H,O-Z)
      DIMENSICN WR(M),WI(M),VF(M,M),GN(NS,NS)
      DIMENSICN W11(NS,NS),TCB(M,M),W21(NS,NS),LT(NS),MT(NS)
      DIMENSICN C(NS),CI(NS),CT(NS,MS)
      K=1
      KP=1
      KN=1
      NRZEV=0
      NCPZEV=C
10    IF (K.GT.M) GO TO 210
C---------------------------------------------------------------------------
C CHECK FOR EIGVAL AT OR NEAR J-OMEGA AXIS TO INCLUDE IN E-L EIGSYS
C   TURN FIRST ONE POSITIVE AND SECOND ONE NEGATIVE
C---------------------------------------------------------------------------
C     EIGVR=DABS(WR(K))
      EIGVR=ABS(WR(K))
      IF (EIGVR.GE.1.E-10) GO TO 60
      IF (WI(K)) 40,20,40
20    NRZEV=NRZEV+1
      IF (NRZEV.GT.1) GO TO 30
      WR(K)=EIGVR
      GO TO 8C
30    WR(K)=-EIGVR
      WRITE (6,290)
      GO TO 150
40    NCPZEV=NCPZEV+1
      IF (NCPZEV.GT.1) GO TO 50
      WR(K)=EIGVR
      WR(K+1)=EIGVR
      GO TO 110
50    WR(K)=-EIGVR
      WR(K+1)=-EIGVR
      WRITE (6,300)
      GO TO 180
60    IF (WR(K)) 140,70,70
70    IF (WI(K)) 110,80,110
 80   CONTINUE
C----------------EIGENVECTOR FOR REAL EIGENVALUE,POSITIVE--------------
C80   IF (NOB.EQ.0) GO TO 100
C     DO 90 J=1,M
C90   TCB(J,KP)=VF(J,K)
100   KP=KP+1
      K=K+1
      GO TO 1C
```

```
C-----------EIGENVECTOR FOR COMPLEX EIGENVALUE,POSITIVE REAL PART-------CPT14270
 110  CONTINUE
C110  IF (NCB.EQ.0) GO TO 130                                              CPT14280
C     DO 120 J=1,M                                                         CPT14290
C     FR=VF(J,K)                                                           CPT14300
C     FI=-VF(J,K+1)                                                        CPT14310
C     TCB(J,KP)=FR+FI                                                      CPT14320
C120  TCB(J,KP+1)=FR-FI                                                    CPT14330
130   KP=KP+2                                                              CPT14340
      K=K+2                                                                CPT14350
      GO TO 10                                                             CPT14360
140   IF (WI(K)) 180,150,180                                               CPT14370
C-----------EIGENVECTOR FOR REAL EIGENVALUE,NEGATIVE REAL PART---------CPT14380
150   C(KN)=WR(K)                                                          CPT14390
      CI(KN)=WI(K)                                                         CPT14400
      IF (NCB.NE.0) GO TO 170                                              CPT14410
      KNS=KN+NS                                                            CPT14420
      DO 160 J=1,M                                                         CPT14430
 160  TCB(J,KNS)=VF(J,K)                                                   CPT14440
170   KN=KN+1                                                              CPT14450
      K=K+1                                                                CPT14460
      GO TO 10                                                             CPT14470
C-----------EIGENVECTOR FOR COMPLEX EIGENVALUE,NEGATIVE REAL PART------CPT14480
180   RR=WR(K)                                                             CPT14490
      RI=WI(K)                                                             CPT14500
      C(KN)=RR                                                             CPT14510
      C(KN+1)=RR                                                           CPT14520
      CI(KN)=RI                                                            CPT14530
      CI(KN+1)=-RI                                                         CPT14540
      IF (NCB.NE.0) GO TO 200                                              CPT14550
      KNS=KN+NS                                                            CPT14560
      DO 190 J=1,M                                                         CPT14570
      FR=VF(J,K)                                                           CPT14580
      FI=-VF(J,K+1)                                                        CPT14590
      TCB(J,KNS)=FR+FI                                                     CPT14600
190   TCB(J,KNS+1)=FR-FI                                                   CPT14610
200   KN=KN+2                                                              CPT14620
      K=K+2                                                                CPT14630
      GO TO 10                                                             CPT14640
210   CONTINUE                                                             CPT14650
      IF (NCB.NE.0) GO TO 240                                              CPT14660
C-------------------------FORMATION OF W11-----------------------------CPT14670
      DO 220 I=1,NS                                                        CPT14680
      DO 220 J=1,NS                                                        CPT14690
      W11(I,J)=TCB(I,J+NS)                                                 CPT14700
220   CT(I,J)=W11(I,J)                                                     CPT14710
C-------------------------FORMATION OF W21-----------------------------CPT14720
      DO 230 I=1,NS                                                        CPT14730
```

```
      DO 230 J=1,NS
230   W21(I,J)=TCB(I+NS,J+NS)
 240  CONTINUE
C240  IF (NOB.EQ.0) GO TO 260
C     DO 250 I=1,NS
C     DO 250 J=1,NS
C     W21(I,J)=-TCB(I,J)
C250  W11(I,J)=TCB(I+NS,J)
260   CONTINUE
C-----------------------------INVERT W11------------------------------------
      NSQ=NS*NS
      CALL MINV (NSQ,W11,NS,DETC,LT,MT)
C-------------------CALCULATE THE RGAIN MATRIX-------------------------------
      DO 270 IL=1,NS
      DO 270 JL=1,NS
      GN(IL,JL)=0.E0
      DO 270 KL=1,NS
270   GN(IL,JL)=GN(IL,JL)+W21(IL,KL)*W11(KL,JL)
C     IF (NOB.EQ.0) RETURN
C     DO 280 I=1,NS
C     DO 280 J=1,NS
C280  CT(I,J)=W11(J,I)
      RETURN
C---------------------------------------------------------------------------
290   FORMAT (1X,51H EULER-LAGRANGE EQUATIONS HAVE A REAL EIGENVALUE AT,
     114H OR NEAR ZERO./)
300   FORMAT (1X,49H EULER-LAGRANGE EQUATIONS HAVE A COMPLEX PAIR OF ,40
     1HEIGENVALUES AT OR NEAR THE J-OMEGA AXIS.)
      END
C===========================================================================
      SUBROUTINE MINV (NSQ,A,N,D,L,M)
C===========================================================================
      IMPLICIT REAL*4 (A-H,O-Z)
      DIMENSION A(NSQ),L(N),M(N)
C     DOUBLE PRECISION A,D,BIGA,HOLD
      REAL*4 D,BIGA,HOLD
      NM=N*N
C     D=1.0D0
      D=1.0E0
      NK=-N
      DO 180 K=1,N
      NK=NK+N
      L(K)=K
      M(K)=K
      KK=NK+K
      BIGA=A(KK)
      DO 20 J=K,N
      IZ=N*(J-1)
```

```
      DO 20 I=K,N                                                       CPT15180
      IJ=IZ+I                                                           CPT15190
C     IF (DABS(BIGA)-DABS(A(IJ))) 10,20,20                              CPT15200
      IF (ABS(BIGA)-ABS(A(IJ))) 10,20,20                                CPT15200
10    BIGA=A(IJ)                                                        CPT15210
      L(K)=I                                                            CPT15220
      M(K)=J                                                            CPT15230
20    CONTINUE                                                          CPT15240
C---------------------------INTERCHANGE ROWS---------------------------CPT15250
      J=L(K)                                                            CPT15260
      IF (J-K) 50,50,30                                                 CPT15270
30    KI=K-N                                                            CPT15280
      DO 40 I=1,N                                                       CPT15290
      KI=KI+N                                                           CPT15300
      HOLD=-A(KI)                                                       CPT15310
      JI=KI-K+J                                                         CPT15320
      A(KI)=A(JI)                                                       CPT15330
40    A(JI)=HOLD                                                        CPT15340
C---------------------------INTERCHANGE COLUMNS------------------------CPT15350
50    I=M(K)                                                            CPT15360
      IF (I-K) 80,80,60                                                 CPT15370
60    JP=N*(I-1)                                                        CPT15380
      DO 70 J=1,N                                                       CPT15390
      JK=NK+J                                                           CPT15400
      JI=JP+J                                                           CPT15410
      HOLD=-A(JK)                                                       CPT15420
      A(JK)=A(JI)                                                       CPT15430
70    A(JI)=HOLD                                                        CPT15440
C----------------------DIVIDE COLUMN BY MINUS PIVOT--------------------CPT15450
C------------------(VALUE OF PIVOT ELEMENT IS CONTAINED IN BIGA)-------CPT15460
80    IF (BIGA) 100,90,100                                              CPT15470
C90   D=0.0D0                                                           CPT15480
 90   D=0.0E0                                                           CPT15480
      RETURN                                                            CPT15490
100   DO 120 I=1,N                                                      CPT15500
      IF (I-K) 110,120,110                                              CPT15510
110   IK=NK+I                                                           CPT15520
      A(IK)=A(IK)/(-BIGA)                                               CPT15530
120   CONTINUE                                                          CPT15540
C---------------------------REDUCE MATRIX------------------------------CPT15550
      DO 150 I=1,N                                                      CPT15560
      IK=NK+I                                                           CPT15570
      HOLD=A(IK)                                                        CPT15580
      IJ=I-N                                                            CPT15590
      DO 150 J=1,N                                                      CPT15600
      IJ=IJ+N                                                           CPT15610
      IF (I-K) 130,150,130                                              CPT15620
130   IF (J-K) 140,150,140                                              CPT15630
```

```
140   KJ=IJ-I+K                                                        CPT15640
      A(IJ)=HOLD*A(KJ)+A(IJ)                                           CPT15650
150   CONTINUE                                                         CPT15660
C----------------------------DIVIDE ROW BY PIVOT-----------------------CPT15670
      KJ=K-N                                                           CPT15680
      DO 170 J=1,N                                                     CPT15690
      KJ=KJ+N                                                          CPT15700
      IF (J-K) 160,170,160                                             CPT15710
160   A(KJ)=A(KJ)/BIGA                                                 CPT15720
170   CONTINUE                                                         CPT15730
C-----------------------------PRODUCT OF PIVOTS------------------------CPT15740
      D=D*BIGA                                                         CPT15750
C-------------------------REPLACE PIVOT BY RECIPROCAL-------------------CPT15760
      A(KK)=(1.0E0)/BIGA                                               CPT15770
180   CONTINUE                                                         CPT15780
C-----------------------FINAL ROW AND COLUMN INTERCHANGE----------------CPT15790
      K=N                                                              CPT15800
190   K=(K-1)                                                          CPT15810
      IF (K) 260,260,200                                               CPT15820
200   I=L(K)                                                           CPT15830
      IF (I-K) 230,230,210                                             CPT15840
210   JQ=N*(K-1)                                                       CPT15850
      JR=N*(I-1)                                                       CPT15860
      DO 220 J=1,N                                                     CPT15870
      JK=JQ+J                                                          CPT15880
      HOLD=A(JK)                                                       CPT15890
      JI=JR+J                                                          CPT15900
      A(JK)=-A(JI)                                                     CPT15910
220   A(JI)=HOLD                                                       CPT15920
230   J=M(K)                                                           CPT15930
      IF (J-K) 190,190,240                                             CPT15940
240   KI=K-N                                                           CPT15950
      DO 250 I=1,N                                                     CPT15960
      KI=KI+N                                                          CPT15970
      HOLD=A(KI)                                                       CPT15980
      JI=KI-K+J                                                        CPT15990
      A(KI)=-A(JI)                                                     CPT16000
250   A(JI)=HOLD                                                       CPT16010
      GO TO 190                                                        CPT16020
260   K=0                                                              CPT16030
      RETURN                                                           CPT16040
      END                                                              CPT16050
C======================================================================CPT18980
C===================================================================== CPT30850
      SUBROUTINE EREXIT (N,A,IERR)                                     CPT30860
C     EREXIT RETURNS THE NUMBER OF THE EIGENVALUE WHERE HQR2         = CPT30870
C     FAILS, THEN STOPS THE PROGRAM.                                 = CPT30880
C===================================================================== CPT30890
```

```
      INTEGER IERR                                                      CPT3C900
      DOUBLE PRECISION A                                                CPT3C910
      DIMENSION A(N,N)                                                  CPT3C920
      WRITE (5,10) IERR                                                 CPT3C930
      CALL RAFRNT (N,N,N,9,A,4,'(9(1X,1PD13.6))')                       CPT3C940
      RETURN                                                            CPT3C950
10    FORMAT (35H FAILURE IN HQR2 ON EIGENVALUE NO. ,I3)                CPT3C960
      END                                                               CPT3C970
C=======================================================================CPT3C980
C  SUBROUTINE MATPRT -- DISPLAYS A TWO-DIMENSIONAL ARRAY (16 COLS. MAX)=CPT3C990
C  IN VARIABLE SCREEN FORMAT FOR USER EASE IN ROW IDENTIFICATION.      =CPT31C00
C=======================================================================CPT31C10
      SUBROUTINE MATPRT (PRTT,NROW,NCOL)                                CPT31C20
C=======================================================================CPT31C10
      IMPLICIT REAL*4 (A-H,O-Z)                                         CPT31C30
      DIMENSION PRTT(NROW,NCOL)                                         CPT31C40
C-----------------------------------------------------------------------CPT31C50
      IF (NCOL.EQ.0)  NCOL=1                                            CPT31C60
      IF (NCOL.EQ.1)  WRITE (5,10)  ((PRTT(I,J),J=1,NCOL),I=1,NROW)     CPT31C70
      IF (NCOL.EQ.2)  WRITE (5,20)  ((PRTT(I,J),J=1,NCOL),I=1,NROW)     CPT31C80
      IF (NCOL.EQ.3)  WRITE (5,30)  ((PRTT(I,J),J=1,NCOL),I=1,NROW)     CPT31C90
      IF (NCOL.EQ.4)  WRITE (5,40)  ((PRTT(I,J),J=1,NCOL),I=1,NROW)     CPT31100
      IF (NCOL.EQ.5)  WRITE (5,50)  ((PRTT(I,J),J=1,NCOL),I=1,NROW)     CPT31110
      IF (NCOL.EQ.6)  WRITE (5,60)  ((PRTT(I,J),J=1,NCOL),I=1,NROW)     CPT31120
      IF (NCOL.EQ.7)  WRITE (5,70)  ((PRTT(I,J),J=1,NCOL),I=1,NROW)     CPT31130
      IF (NCOL.EQ.8)  WRITE (5,80)  ((PRTT(I,J),J=1,NCOL),I=1,NROW)     CPT31140
      IF (NCOL.EQ.9)  WRITE (5,90)  ((PRTT(I,J),J=1,NCOL),I=1,NROW)     CPT31150
      IF (NCOL.EQ.10) WRITE (5,100) ((PRTT(I,J),J=1,NCOL),I=1,NROW)     CPT31160
      IF (NCOL.EQ.11) WRITE (5,110) ((PRTT(I,J),J=1,NCOL),I=1,NROW)     CPT31170
      IF (NCOL.EQ.12) WRITE (5,120) ((PRTT(I,J),J=1,NCOL),I=1,NROW)     CPT31180
      IF (NCOL.EQ.13) WRITE (5,130) ((PRTT(I,J),J=1,NCOL),I=1,NROW)     CPT31190
      IF (NCOL.EQ.14) WRITE (5,140) ((PRTT(I,J),J=1,NCOL),I=1,NROW)     CPT31200
      IF (NCOL.EQ.15) WRITE (5,150) ((PRTT(I,J),J=1,NCOL),I=1,NROW)     CPT31210
      IF (NCOL.EQ.16) WRITE (5,160) ((PRTT(I,J),J=1,NCOL),I=1,NROW)     CPT31220
      RETURN                                                            CPT31230
C---------------------------------------------------------------------- CPT31240
10    FORMAT (F12.5)                                                    CPT31250
20    FORMAT (2F12.5)                                                   CPT31260
30    FORMAT (3F12.5)                                                   CPT31270
40    FORMAT (4F12.5)                                                   CPT31280
50    FORMAT (5F12.5)                                                   CPT31290
60    FORMAT (6F12.5)                                                   CPT31300
70    FORMAT (6F12.5,/,F12.5,//)                                        CPT31310
80    FORMAT (6F12.5,/,2F12.5,//)                                       CPT31320
90    FORMAT (6F12.5,/,3F12.5,//)                                       CPT31330
100   FORMAT (6F12.5,/,4F12.5,//)                                       CPT31340
110   FORMAT (6F12.5,/,5F12.5,//)                                       CPT31350
120   FORMAT (6F12.5,/,6F12.5,//)                                       CPT31360
```

```
130   FORMAT (6F12.5,/,6F12.5,/,F12.5,//)                                    CPT31370
140   FORMAT (6F12.5,/,6F12.5,/,2F12.5,//)                                   CPT31380
150   FORMAT (6F12.5,/,6F12.5,/,3F12.5,//)                                   CPT31390
160   FORMAT (6F12.5,/,6F12.5,/,4F12.5,//)                                   CPT31400
      END                                                                    CPT31410
C=========================================================================CPT16800
      SUBROUTINE MODE (WNORM,G,GNORM,NS,N1,N2,ICON)                          CPT16810
C                                                                          =CPT16820
C   WNORM  TRANSFORMATION MATRIX U OR U-INV                                =CPT16830
C   NS     NO. OF STATE                                                    =CPT16840
C   NC     NO.  OF INPUTS OR OUTPUTS                                       =CPT16850
C   ICON   CONTROL FLAG TO INDICATE WHICH TRANSFORMATION                   =CPT16860
C          0 =  MODAL G                                                    =CPT16870
C          1 =  MODAL GAMMA                                                =CPT16880
C          2 =  MODAL H                                                    =CPT16890
C          3 =  MODAL C                                                    =CPT16900
C          4 =  MODAL K                                                    =CPT16910
C          5 =  CONTROL EIGENVECTOR MATRIX                                 =CPT16920
C          6 =  MEASUREMENT EIGENVECTOR MATRIX                             =CPT16930
C=========================================================================CPT16940
      IMPLICIT REAL*4(A-H,O-Z)                                               CPT16950
      DIMENSION WNORM(NS,NS),G(N1,N2),GNORM(N1,N2)                           CPT16960
      DO 10 I=1,N1                                                           CPT16970
      DO 10 J=1,N2                                                           CPT16980
10    GNORM(I,J)=0.                                                          CPT16990
      IPOINT=ICON+1                                                          CPT17000
      GO TO (20,20,90,90,20,90,90), IPOINT                                   CPT17010
20    DO 30 J=1,N2                                                           CPT17020
      DO 30 I=1,NS                                                           CPT17030
      DO 30 K=1,NS                                                           CPT17040
30    GNORM(I,J)=GNORM(I,J)+WNORM(I,K)*G(K,J)                                CPT17050
      GO TO (40,70,90,90,80), IPOINT                                         CPT17060
40    WRITE (6,170)                                                          CPT17070
50    DO 60 I=1,NS                                                           CPT17080
60    WRITE (6,230) (GNORM(I,J),J=1,N2)                                      CPT17090
      RETURN                                                                 CPT17100
70    WRITE (6,180)                                                          CPT17110
      GO TO 50                                                               CPT17120
80    WRITE (6,240)                                                          CPT17130
      GO TO 50                                                               CPT17140
90    DO 100 J=1,NS                                                          CPT17150
      DO 100 I=1,N1                                                          CPT17160
      DO 100 K=1,NS                                                          CPT17170
100   GNORM(I,J)=GNORM(I,J)+G(I,K)*WNORM(K,J)                                CPT17180
      GO TO (110,110,110,120,110,130,140), IPOINT                            CPT17190
110   WRITE (6,190)                                                          CPT17200
      GO TO 150                                                              CPT17210
120   WRITE (6,200)                                                          CPT17220
```

```
      GO TO 150                                                         CPT17230
130   WRITE (6,210)                                                     CPT17240
      GO TO 150                                                         CPT17250
140   WRITE (6,220)                                                     CPT17260
150   DO 160 I=1,N1                                                     CPT17270
160   WRITE (6,230) (GNORM(I,J),J=1,NS)                                 CPT17280
      RETURN                                                            CPT17290
C-----------------------------------------------------------------------CPT17300
170   FORMAT (//,5X,45HMODAL CONTROL DISTRIBUTION MATRIX......TI*G..,//)CPT17310
180   FORMAT (//,5X,50HMODAL PROCESS NOISE DISTRIBUTION MATRIX...TI*GAM.CPT17320
     1.,//)                                                             CPT17330
190   FORMAT (//,5X,45HMODAL MEASUREMENT SCALING MATRIX...H(BAR)*T..,//)CPT17340
200   FORMAT (//,5X,45HTHE MODAL CONTROL GAINS.................C*T..,//)CPT17350
210   FORMAT (//,5X,45HCONTROL EIGENVECTOR MATRIX.............C*M..,//)CPT17360
220   FORMAT (//,5X,45HMEASUREMENT EIGENVECTOR MATRIX.....H(BAR)*M..,//)CPT17370
230   FORMAT (1X,(2X,1P,6E14.6))                                        CPT17380
240   FORMAT (//,5X,45HMODAL FILTER STEADY STATE GAINS........TI*K..,//)CPT17390
      END                                                               CPT17400
C=======================================================================CPT17410
      SUBROUTINE CNORM (WZ,WY,VEC,NS,IWRITE,NSQ,DDD,D1,D2,WNORM,WNORMI,HCPT17420
     10,CM,N1,N2)                                                       CPT17430
C                                                                      =CPT17440
C         WZ(I)       REAL PART OF I-TH EIGENVALUE                     =CPT17450
C                                                                      =CPT17460
C         WY(I)       COMPLEX PART OF I-TH EIGENVALUE                  =CPT17470
C                                                                      =CPT17480
C         VEC         MATRIX OF RIGHT EIGENVECTORS STORED IN REAL FORM =CPT17490
C                       FROM HQR2                                      =CPT17500
C         NS          NO. OF STATES                                    =CPT17510
C                                                                      =CPT17520
C         IWRITE      FLAG TO CONTROL FORMATS FOR DIFFERENT EIGHENSYSTEMS=CPT17530
C                                                                      =CPT17540
C         WNORM       NORMALIZED MATRIX U OF RIGHT EIGENVECTORS STORED =CPT17550
C                       BY COLUMNS IN REAL FORM                        =CPT17560
C         WNORMI      U-INVERSE 2*CONGUGATE OF LEFT EIGENVECTORS       =CPT17570
C                         STORED BY ROW IN REAL FORM                   =CPT17580
C         NSQ,DDD,D1,D2 - ARGUMENTS PASSED TO MINV                     =CPT17590
C=======================================================================CPT17600
      IMPLICIT REAL*4 (A-H,O-Z)                                         CPT17610
      CHARACTER *8 FIELD,COMMA,SEMCOL,RIGHT,FMT,SEMEND                  CPT17620
      DIMENSION WZ(NS),WY(NS),VEC(NS,NS),WNORM(NS,NS),WNORMI(NS,NS),STORCPT17630
     1E(6),D1(NS),D2(NS),FMT(14),HO(N1,N2),CM(N1,N2)                    CPT17640
      DATA FIELD/'E12.5'/,COMMA/',',',',','/,SEMCOL/',':',','/,RIGHT/')'/,CPT17650
     1FMT/'(1X,1P',13*' '/,SEMEND/',':'''/                              CPT17660
C---------NORMALIZE COMPLEX EIGENVECTORS BY LARGEST ELEMENT-------------CPT17670
      KK=0                                                              CPT17680
      LR=0                                                              CPT17690
      LC=0                                                              CPT17700
```

```
      DO 50 K=1,NS                                                      CPT17710
      IF (KK.EQ.1) GO TO 40                                             CPT17720
C     IF (DABS(WY(K)).LT.1.D-10) GO TO 50                               CPT17730
      IF (ABS(WY(K)).LT.1.E-10) GO TO 50                                CPT17730
      LC=LC+1                                                           CPT17740
      EMAX=0.E0                                                         CPT17750
      DO 20 I=1,NS                                                      CPT17760
      CMOD=VEC(I,K)**2+VEC(I,K+1)**2                                    CPT17770
      IF (CMOD-EMAX) 20,10,10                                           CPT17780
   10 EMAX=CMOD                                                         CPT17790
      M=I                                                               CPT17800
   20 CONTINUE                                                          CPT17810
      VMR=VEC(M,K)                                                      CPT17820
      VMI=VEC(M,K+1)                                                    CPT17830
      DO 30 I=1,NS                                                      CPT17840
      VR=VEC(I,K)                                                       CPT17850
      VI=VEC(I,K+1)                                                     CPT17860
      VECRN=(VR*VMR+VI*VMI)/EMAX                                        CPT17870
      VECIN=(-VR*VMI+VI*VMR)/EMAX                                       CPT17880
      WNORM(I,K)=VECRN                                                  CPT17890
      WNORM(I,K+1)=VECIN                                                CPT17900
   30 CONTINUE                                                          CPT17910
      KK=1                                                              CPT17920
      GO TO 50                                                          CPT17930
   40 KK=0                                                              CPT17940
   50 CONTINUE                                                          CPT17950
C------------NORMALIZE REAL EIGENVECTORS BY THE TOTAL LENGTH------------CPT17960
      DO 80 K=1,NS                                                      CPT17970
C     IF (DABS(WY(K)).GE.1.D-10) GO TO 80                               CPT17980
      IF (ABS(WY(K)).GE.1.E-10) GO TO 80                                CPT17980
      LR=LR+1                                                           CPT17990
      REMOD=0.E0                                                        CPT18000
      DO 60 I=1,NS                                                      CPT18010
   60 REMOD=VEC(I,K)**2+REMOD                                           CPT18020
C     RMOD=DSQRT(REMOD)                                                 CPT18030
      RMOD=SQRT(REMOD)                                                  CPT18030
      DO 70 I=1,NS                                                      CPT18040
      RVEC=VEC(I,K)/RMOD                                                CPT18050
      WNORM(I,K)=RVEC                                                   CPT18060
   70 CONTINUE                                                          CPT18070
   80 CONTINUE                                                          CPT18080
      GO TO (90,100,110,120,130), IWRITE                                CPT18090
   90 WRITE (6,320)                                                     CPT18100
      GO TO 140                                                         CPT18110
  100 WRITE (6,330)                                                     CPT18120
      GO TO 140                                                         CPT18130
  110 WRITE (6,340)                                                     CPT18140
      GO TO 140                                                         CPT18150
```

```
120   WRITE (6,350)                                                   CPT18160
      GO TO 140                                                       CPT18170
130   WRITE (6,360)                                                   CPT18180
140   KK=0                                                            CPT18190
      NPRTW=0                                                         CPT18200
      NFMTW=1                                                         CPT18210
      DO 180 I=1,NS                                                   CPT18220
      IF (KK.EQ.1) GO TO 170                                          CPT18230
C     IF (DABS(WY(I)).GT.1.D-10) KK=1                                 CPT18240
      IF (ABS(WY(I)).GT.1.E-10) KK=1                                  CPT18240
C----PRINT OUT NO MORE THAN 6 WORDS, NOT SEPARATING COMPLEX EIGVAL------CPT18250
      IF (NPRTW.LT.5.OR.(NPRTW.EQ.5.AND.KK.EQ.0)) GO TO 150           CPT18260
      FMT(NFMTW+1)=RIGHT                                              CPT18270
      WRITE (6,FMT) (STORE(J),J=1,NPRTW)                              CPT18280
      NPRTW=0                                                         CPT18290
      NFMTW=1                                                         CPT18300
150   NPRTW=NPRTW+1                                                   CPT18310
      NFMTW=NFMTW+1                                                   CPT18320
      IF (KK.EQ.1) GO TO 160                                          CPT18330
      STORE(NPRTW)=WZ(I)                                              CPT18340
      FMT(NFMTW)=FIELD                                                CPT18350
      NFMTW=NFMTW+1                                                   CPT18360
      FMT(NFMTW)=SEMCOL                                               CPT18370
      GO TO 180                                                       CPT18380
160   STORE(NPRTW)=WZ(I)                                              CPT18390
      FMT(NFMTW)=FIELD                                                CPT18400
      FMT(NFMTW+1)=COMMA                                              CPT18410
      STORE(NPRTW+1)=WY(I)                                            CPT18420
      FMT(NFMTW+2)=FIELD                                              CPT18430
      FMT(NFMTW+3)=SEMCOL                                             CPT18440
      NFMTW=NFMTW+3                                                   CPT18450
      NPRTW=NPRTW+1                                                   CPT18460
      GO TO 180                                                       CPT18470
170   KK=0                                                            CPT18480
180   CONTINUE                                                        CPT18490
      FMT(NFMTW)=SEMEND                                               CPT18500
      FMT(NFMTW+1)=RIGHT                                              CPT18510
      WRITE (6,FMT) (STORE(J),J=1,NPRTW)                              CPT18520
      IF (IWRITE.NE.1) GO TO 190                                      CPT18530
      WRITE (6,370)                                                   CPT18540
      GO TO 200                                                       CPT18550
 190  CONTINUE
C     WRITE (6,380)                                                   CPT18560
 200  CONTINUE
C     CALL RAPRNT (NS,NS,NS,6,WNORM,4,'(6(1X,1PE13.6))')              CPT18570
      GO TO (230,210,210,220,220), IWRITE                             CPT18580
210   CALL MODE (WNORM,HO,CM,NS,N1,N2,5)                              CPT18590
      GO TO 230                                                       CPT18600
```

```
220   CALL MODE (WNORM,HC,CM,NS,N1,N2,6)                               CPT18610
230   GO TO (240,250,260,270,280), IWRITE                              CPT18620
240   WRITE (6,390)                                                    CPT18630
      GO TO 290                                                        CPT18640
  250 CONTINUE
C     WRITE (6,400)                                                    CPT18650
      GO TO 290                                                        CPT18660
260   WRITE (6,410)                                                    CPT18670
      GO TO 290                                                        CPT18680
270   WRITE (6,420)                                                    CPT18690
      GO TO 290                                                        CPT18700
280   WRITE (6,430)                                                    CPT18710
C------------------SAVE U-INVERSE OPEN LOOP IN WNORMI------------------CPT18720
290   IF (IWRITE .GT. 1) GO TO 310                                     CPT18730
      DO 300 I=1,NS                                                    CPT18740
      DO 300 J=1,NS                                                    CPT18750
300   WNORMI(I,J)=WNORM(I,J)                                           CPT18760
      CALL MINV (NSQ,WNORMI,NS,DDD,D1,D2)                              CPT18770
C     CALL RAPRNT (NS,NS,NS,6,WNORMI,4,'(6(1X,1PE13.6))')              CPT18780
      RETURN                                                           CPT18790
310   CALL MINV (NSQ,WNORM,NS,DDD,D1,D2)                               CPT18800
C     CALL RAPRNT (NS,NS,NS,6,WNORM,4,'(6(1X,1PE13.6))')               CPT18810
      RETURN                                                           CPT18820
C----------------------------------------------------------------------OPT18830
320   FORMAT (//5X,42HOPEN LOOP EIGENVALUES..........DET(SI-F)..,//)   CPT18840
330   FORMAT (//5X,46HC-LOOP OPTIMAL REG. E-VALUES...DET(SI-F+G*C)..,//)CPT18850
340   FORMAT (//5X,46HC-LOOP SUBOPT. REG. E-VALUES...DET(SI-F+G*C)..,//)CPT18860
350   FORMAT (//5X,46HC-LOOP OPTIMAL EST. E-VALUES...DET(SI-F+K*H)..,//)OPT18870
360   FORMAT (//5X,46HC-LOOP SUBOPT. EST. E-VALUES...DET(SI-F+K*H)..,//)CPT18880
370   FORMAT (//5X,46HOPEN LOOP RIGHT EIGENVECTOR MATRIX.......T....,//)CPT18890
380   FORMAT (//5X,46HC-LOOP RIGHT EIGENVECTOR MATRIX..........M....,//)CPT18900
390   FORMAT (//5X,46HOPEN LOOP LEFT EIGENVECTOR MATRIX......T-INV..,//)CPT18910
400   FORMAT (//5X,46HC-LOOP OPT. REG. LEFT E-VECTOR MATRIX..M-INV..,//)CPT18920
410   FORMAT (//5X,46HC-LOOP SUBOPT-REG. LEFT E-VECTOR MATRIX..M-INV,//)CPT18930
420   FORMAT (//5X,46HC-LOOP OPT. FILTER LEFT E-VECTOR MATRIX..M-INV,//)CPT18940
430   FORMAT (//5X,51HC-LOOP SUBOPT. FILTER LEFT E-VECTOR MATRIX..M-INV.CPT18950
     1.,//)                                                            CPT18960
      END
C                                                                      WURZ0C02
C        ..............................................................WURZ0C03
C                                                                      WURZ0C04
C        SUBROUTINE WURZEL                                             WURZ0C05
C                                                                      WURZ0C06
C        PURPOSE                                                       WURZ0C07
C           TO FIND THE SQUARE ROOT OF A DOUBLE-PRECISION, POSITIVE    WURZ0C08
C           -DEFINITE MATRIX                                           WURZ0C09
C                                                                      WURZ0C10
C        USAGE                                                         WURZ0C11
```

```
C     CALL WURZEL(A,B,N,THETA,EPS,NC)
C
C  DESCRIPTION OF PARAMETERS
C     A      - SYMMETRIC,POSITIVE-DEFINITE,INPUT SQUARE MATRIX.
C     B      - SYMMETRIC,CONTAINS SQUARE ROOT OF A,AFTER EXIT
C              FROM WURZEL.
C     N      - ORDER OF A AND B.
C     THETA - CONSTANT (SEE METHOD AND REMARKS).IF EIGENVALUES
C              OF A ARE UNKNOWN,CHOOSE THETA CLOSE TO UNITY.
C     EPS    - CONSTANT SPECIFYING DESIRED ACCURACY(SEE METHOD).
C              FOR SIX PLACE ACCURACY CHOOSE EPS =1.E-6
C     NC     - ORDER OF A AND B.
C
C  REMARKS
C     PARAMETERS A,B,THETA,AND EPS ABOVE ARE ALL REAL*8
C
C     THE CONVERSION PROCESS IS VERY SLOW IF THE LARGEST AND
C     SMALLEST EIGENVALUES OF A DIFFER BY SEVERAL ORDERS OF
C     MAGNITUDE.
C
C  SUBROUTINES AND FUNCTION SUBPROGRAM REQUIRED
C     NONE
C
C  METHOD
C     IF X IS THE VECTOR ASSOCIATED WITH MATRIX A,THEN MATRIX A
C     IS POSITIVE-DEFINITE IF X'*A*X IS GREATER THAN ZERO.
C     NOW THE SEQUENCE:
C           B(K+1) = B(K) + C*(A-B(K)**2)        ..1..
C     CONVERGES TO THE SQUARE ROOT OF A.
C     IN EXPRESSION ..1..  C = THETA/2*SQRT(NORM(A)), WHERE:
C     THETA IS A CONSTANT GREATER THAN ZERO AND LESS THAN THE
C     SQUARE ROOT OF ALPHA.
C     ALPHA IS THE QUOTIENT OF NORM(A),DIVIDED BY THE MAXIMUM
C     EIGENVALUES OF A, AND IT IS GREATER OR EQUAL TO ONE.
C     NORM(A) IS THE MAXIMUM VALUE OBTAINED BY ADDING EACH ROW
C     OF A ALL THE WAY ACROSS.
C     THE FIRST APPROXIMATION OF B,THE SQUARE ROOT MATRIX, IS
C     OBTAINED AS FOLLOWS; B(0) =2.C *C*A
C
C     THE RATE OF CONVERGENCE OF ..1.. TO THE SQUARE ROOT OF A
C     IS GIVEN BY THE RATE OF CONVERGENCE TO ZERO OF THE
C     SEQUENCE:
C              X(K)=(1 - THETA(SQRT(LAMDAMIN/LAMDAMAX)))**K
C       WHERE LANDAMIN AND LANDAMAX ARE THE MINIMUM AND MAXIMUM
C       EIGENVALUES OF A ,RESPECTIVELY.
C
C     TO ACCELERATE CONVERGENCE THETA SHOULD BE CHOSEN CLOSE TO
C     THE SQUARE ROOT OF ALPHA
```

```
C           IF NOTHING IS KNOWN ABOUT ALPHA,OPTIMUM CHOICE OF THETA IS   WURZ0060
C           A VALUE CLOSE TO ONE.                                        WURZ0061
C                                                                        WURZ0062
C           THE ITERATION FOR COMPUTATION OF SQRT(A) STOPS WHEN:         WURZ0063
C               MAX(ABS(B(I,J)**(K+1) - B(I,J)**K)) LESS THAN EPS        WURZ0064
C               I,J                                                      WURZ0065
C                                                                        WURZ0066
C        SEE 'COMMUNICATIONS OF THE ACM',VOLUME 10,NUMBER 3, MARCH       WURZ0067
C        1967, ALGORITHMS.,ALGORITHM 298, PAGE 182                       WURZ0068
C        ..............................................................WURZ0069
C                                                                        WURZ0070
C                                                                        WURZ0071
C                                                                        WURZ0072
      SUBROUTINE WURZEL(A,B,N,THETA,EPS,ND)                              WURZ0001
C                                                                        WURZ0073
      IMPLICIT REAL*4(A-H,O-Z)                                           WURZ0074
      DIMENSION A(ND,ND),B(ND,ND),BB(100)                                WURZ0075
 1001 FORMAT('C'/(4E17.9))                                               WURZ0076
 1000 FORMAT(' DELTA =' ,E17.9)                                          WURZ0077
C     PART1. DETERMINATION OF C,                                         WURZ0078
C                                                                        WURZ0079
      C=0.                                                               WURZ0080
      DO 20 I=1,N                                                        WURZ0081
      S=0.                                                               WURZ0082
      DO 10 J=1,N                                                        WURZ0083
   10 S=S+ ABS(A(I,J))                                                   WURZ0084
      IF(S-C) 20,20,15                                                   WURZ0085
   15 C=S                                                                WURZ0086
   20 CONTINUE                                                           WURZ0087
C                                                                        WURZ0088
      C= 0.5*THETA/ SQRT(C)                                              WURZ0089
C     SET B(C)                                                           WURZ0090
C                                                                        WURZ0091
   30 DO 40 I=1,N                                                        WURZ0092
      DO 40 J=I,N                                                        WURZ0093
      B(I,J)=2.0*C*A(I,J)                                                WURZ0094
   40 B(J,I)=B(I,J)                                                      WURZ0095
C                                                                        WURZ0096
C     PART2. ITERATION FOR COMPUTATION OF SQRT(A). ITERATION STOPS WHEN:WURZ0097
C           MAX(ABS(B(I,J)**(K+1) - B(I,J)**K)) LESS THAN EPS            WURZ0098
C                                                                        WURZ0099
   50 DELTA = 0.                                                         WURZ0100
      DO 80 I=1,N                                                        WURZ0101
      DO 70 J=I,N                                                        WURZ0102
      S=0.                                                               WURZ0103
      DO 60 K=1,N                                                        WURZ0104
   60 S= S-B(I,K)*B(K,J)                                                 WURZ0105
   70 BB(J)=B(I,J)+C*(A(I,J)+S)                                          WURZ0106
```

```
C     COMPUTE MAXIMUM VALUE AS SHOWN IN PREVIOUS COMMENT.                  WURZC107
C                                                                          WURZC108
      DO 75 J=I,N                                                          WURZC109
      S= ABS(B(I,J)-BB(J))                                                 WURZC110
      IF (S-DELTA) 75,75,74                                                WURZC111
   74 DELTA=S                                                              WURZC112
C     WRITE(6,1000)DELTA                                                   WURZC113
   75 B(I,J) = BB(J)                                                       WURZC114
   80 CONTINUE                                                             WURZC115
C                                                                          WURZC116
C                                                                          WURZC117
C                                                                          WURZC118
C     SET SYMMETRIC TERMS IN MATRIX B.                                     WURZC119
      NN=N-1                                                               WURZC120
      DO 90 I=1,NN                                                         WURZC121
      NJ=I+1                                                               WURZC122
      DO 90 J=NJ,N                                                         WURZC123
   90 B(J,I)=B(I,J)                                                        WURZC124
C     WRITE(6,1001)((B(II,LL),LL=1,N),II=1,N)                              WURZC125
      IF (DELTA-EPS) 95,95,94                                              WURZC126
   94 GO TO 50                                                             WURZC127
   95 RETURN                                                               WURZC128
      END                                                                  WURZC129
C*********************************************************************
C
C      **************   THE END   *************
C
C*********************************************************************
```

LIST OF REFERENCES

1. Kalman, R.E., "When is a Linear Control System Optimal?," J. of Basic Eng, Tran. ASME, Vol 86, pp. 51-60, March 1964.

2. Athans, M., "The Role and Use of the Stochastic Linear Quadratic Gaussian Problem in Control System Design", IEEE Tran. Automat. Control, vol AC-16, pp. 529-552, December 1971.

3. Bryson, A. E. and Y. Ho., Applied Optimal Control, Walthan, MA, Ginn, 1969.

4. Kwakernaak, H., and Sivan, R., Linear Optimal Control System, Wiley International, New York, 1972.

5. Tyler, J. S., "The Use of a Quadratic Performance Index to Design Multivariable Control System," IEEE Trans. Auto. Control, Vol 11, pp. 84-92, 1966

6. Fallside, F. and Seraji, H., "Design of Optimal Systems by a Frequency Domain Technique," Proc. IEE, Vol 117, pp. 2017-2024, 1970.

7. MacFarlane, A. G. J., "Return Difference and Return Ratio Matrices and Their Use in Analysis and Design of Multivariable Feedback System," Proc. IEE, Vol 117, pp. 2037-2049, 1970.

8. Solheim, O. A., "Design of Optimal Control Systems with Prescribed Eigenvalues," Int. J. Controls, Vol. 15, No. 1, pp. 143-160, 1972.

9. Harvey, C. A. and Stein, G., "Quadratic Weights for Asymptotic Regulator Properties," IEEE tran. Auto. Control, Vol. AC-23, June 1978.

10. Stein, G. "Generalized Quadratic Weights for Asymptotic Regulator Properties," IEEE Tran. Auto. Control, Vol AC-24, pp. 559-567, August 1979.

11. Broussard, J. R., "A Quadratic Weight Selection Algorithm," Proc. of the 1981 Joint Auto. Contr. Conference, 1981.

12. Gully, S. W., Sandell, N. R., and Lee, W. H., "A Robustness Synthesis Method," IEEE Decision and Control Proc. 21, 1982.

13. Dolye, J. C., and Stein, G., "Multivariable Feedback Design: Concepts for a Classical/Modern Synthesis," IEEE Trans. Auto. Control, Vol. 26, No. 1 PP. 4-16, Feb 1981.

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

15. Lehtomaki, N.A., N.R. Sandell, Jr. and M. Athans, "Robustness Results in Linear Quadratic Gaussian Based Multivariable Control Designs," IEEE Trans. Auto. Contr., Vol. AC-26, No.1, Feb. 1981

16. Moore, B. C., "On the Flexibility Offered by State Feedback in Multivariable Systems Beyond Closed Loop Eigenvalue Assignment," IEEE Tran. Auto. Contr., AC-20, pp. 689-692, October 1976.

17. Dooren, P. V.,Kautsky, J. and Nichols, N. K., "Robust Pole Assignment in Linear State Feedback," Int. J Contr. Vol 41. No 5. pp. 1129-1155, 1985.

18. Kwakernaak, H., Asymptotic Root Loci for Multivariable Linear Optimal Regulator, IEEE Tran. Auto. Contr., AC-21, pp. 378-381 1975.

19. NASA TM-84524, Application of Matrix Singular Value Properties for Evaluating Gain and Phase Margins of Multiloop Systems, Mukhopadhyay, V. and Newsom, J. R., July 1982.

20. Vanderplatts, G. N., ADS - A Fortran Program for Automated Design Synthesis, Version 1.0, Naval Postgraduate School, Monterey, California, May 1984.

21. Vanderplatts, G. N., Numerical Optimization Techniques for Engineering Design: With Applications, McGraw Hill, 1984.

22. Gordon, V. C., Utilization of Numerical Optimization Techniques in the Design of Robust Multi-input Multi-output Control Systems, Ph.D. Dissertation, Naval Postgraduate School, Monterey, California 1984.

23. Alpatech Inc., TR-121, Multivariable Stability Margins for Vehicle Flight Control System, Sandell, N. R.,Jr., et al, Dec. 1981

24. NASA Report 3889, Design of Multivariable Feedback Control Systems Via Spectral Assignment Using Reduced-Order Models and Reduced-Order Observers, R. R. Mielke, L. J. Tung and P. I. Carraway III, April 1985

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5100	2
3.	Department Chairman, Code 67 Department of Aeronautics Naval Postgraduate School Monterey, California 93943-5100	1
4.	Prof. D. J. Collins, Code 67Co Department of Aeronautics Naval Postgraduate School Monterey, California 93943-5100	2
5.	Mr. Chow Wah Keh #08-5067, Blk 639, Ang Mo Kio Ave. 6 S'pore 2056 Republic of Singapore	3

END

FILMED

1-86

DTIC