

AD-A144 159

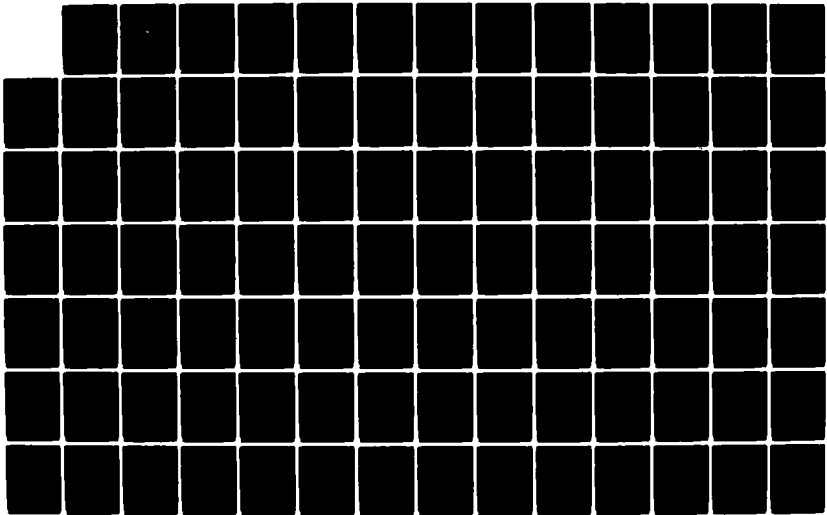
INTERACTIVE IMPLEMENTATION OF THE OPTIMAL SYSTEMS
CONTROL DESIGN PROGRAM (OPTSYSX) ON THE IBM 3033(U)
NAVAL POSTGRADUATE SCHOOL MONTEREY CA J G HODEN MAR 84

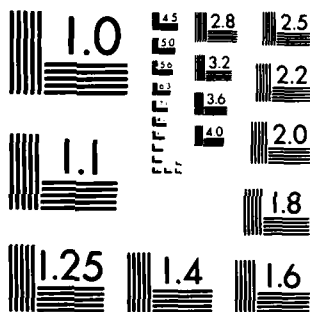
1/2

UNCLASSIFIED

F/G 9/2

NL





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

AD-A144 159

NAVAL POSTGRADUATE SCHOOL
Monterey, California



THESIS

INTERACTIVE IMPLEMENTATION OF THE
OPTIMAL SYSTEMS CONTROL DESIGN PROGRAM (OPTSYSX)
ON THE IBM 3033

by

John Gustav Hoden II

March 1984

Thesis Advisor:

Daniel J. Collins

DTIC
ELECTRONIC
SERIES

DTIC FILE COPY

Approved for public release; distribution unlimited.

84 08 06 065

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO. AD-A144159	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) INTERACTIVE IMPLEMENTATION OF THE OPTIMAL SYSTEMS CONTROL DESIGN PROGRAM (OPTSYSX) ON THE IBM 3033		5. TYPE OF REPORT & PERIOD COVERED Master's Thesis March 1984
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) John G. Hoden II		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 March 1984
		13. NUMBER OF PAGES 145
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Optimal Systems Control Systems Control Control Systems		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) → This thesis discusses the modification of an existing Optimal Systems Control FORTRAN program (OPTSYS) originally obtained from Professor Arthur E. Bryson of Stanford University. The modified FORTRAN program (OPTSYSX) is now designed to run completely interactively under VM/CMS on the IBM 3033 and is considered completely compatible with similar operating systems.		

Program capabilities include: complete eigensystem analysis; the ability to perform computations on very large multi-variable systems; controller, filter, regulator and compensator synthesis; transfer function analysis; steady-state gain determinations; and modal analysis.

The program permits users to rapidly carry out simulation, analysis, and design of all classes of optimal systems control problems in a totally interactive mode. Examples of various types of problems are worked out during individual terminal sessions. *As facts, or program listing is available in the report.*

Accession For	
NSA&I	<input checked="" type="checkbox"/>
TAB	<input type="checkbox"/>
Approved	<input type="checkbox"/>
Classification	
Distribution/	
Availability Codes	
Avail and/or	
Special	
<i>Al</i>	



Approved for public release; distribution unlimited.

Interactive Implementation of the
Optimal Systems Control Program (OPTSYSX)
on the IBM/3033

by

John G. Hoden
Lieutenant Commander, United States Navy
B.A., University of Minnesota, Duluth, 1970

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN AERONAUTICAL ENGINEERING

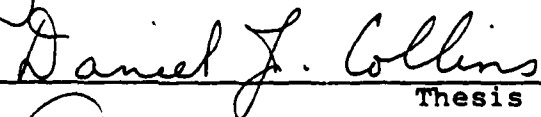
from the

NAVAL POSTGRADUATE SCHOOL
March 1984

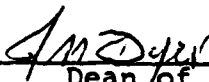
Author:



Approved by:


_____ Thesis Advisor


_____ Chairman, Department of Aeronautics


_____ Dean of Science and Engineering

ABSTRACT

This thesis discusses the modification of an existing Optimal Systems Control FORTRAN Program (OPTSYS) originally obtained from Professor Arthur E. Bryson of Stanford University.

The modified FORTRAN program (OPTSYSX) is now designed to run completely interactively under VM/CMS on the IBM 3633 and is considered completely compatible with similar operating systems.

Program capabilities include: complete eigensystem analysis; the ability to perform computations on very large multivariable systems; controller, filter, regulator and compensator synthesis; transfer function analysis; steady-state gain determination; and modal analysis.

The program permits users to rapidly carry out simulation, analysis, and design of all classes of Optimal Systems Control problems in a totally interactive mode. Examples of various types of problems are worked out during individual terminal sessions.

TABLE OF CONTENTS

I.	INTRODUCTION	9
II.	THE CPTSYSX COMPUTER PROGRAM	11
	A. INTRODUCTION	11
	B. SYSTEM/MODEL DESCRIPTION	11
	C. PROGRAM OUTPUT	13
	1. Open-Loop Eigensystem Calculations	13
	2. Regulator Synthesis Calculations	13
	3. Filter Synthesis Calculations	14
	4. Stationary Closed-Loop Calculations	15
	D. SOLUTION ALGORITHM	15
	E. PROGRAM OVERVIEW	16
	1. Problem Description/Program Flow Control	17
	2. Interactive Data Input	17
	3. Calculation Sequencing	17
	4. QR Algorithm Transformation	17
	5. Riccati Equation Calculations	17
	6. Modal Calculations	18
	7. Transfer Function Calculations	18
	8. Data Output	18
III.	INTERACTIVE PROGRAM OPERATION	19
	A. DESIGN CONSIDERATIONS	19
	B. PROGRAM LANGUAGE	19
	C. GENERAL PROGRAM MODIFICATIONS	20
	D. INTERACTIVE PROGRAM DEVELOPMENT	21
	1. Program Flow Control	21
	2. General Input Sequencing Requirements	22

3.	Interactive Data Input	23
4.	Saving Interactive Input	24
5.	User-Defined Input Files	24
6.	Internal Data Generation	25
7.	Data Entry Correction	26
E.	USER-ERROR PROTECTION FEATURES	27
1.	Data Type Conversion Errors	27
2.	Inconsistent Program Control Flag Errors	28
IV.	PROGRAM USE AND EXAMPLES	29
A.	COPEN-LOOP EIGENSYSTEM ANALYSIS	29
B.	REGULATOR SYNTHESIS	33
C.	FILTER SYNTHESIS	43
D.	EXAMPLE OF PROGRAM FAILURE	53
V.	CONCLUSIONS AND RECOMMENDATIONS	69
A.	CONCLUSIONS	69
B.	RECOMMENDATIONS	70
1.	Program Availability	70
2.	Computer Graphics	70
3.	Further Modifications	70
4.	Program Application	71
	LIST OF REFERENCES	143
	BIBLIOGRAPHY	144
	INITIAL DISTRIBUTION LIST	145

SYMBOLS

A = state (N_s, N_s) or output (N_o, N_o) weighting matrix
B = control (N_c, N_c) weighting matrix
C = control gain matrix (N_c, N_s)
D = control (N_c, N_c) or noise (N_o, N_g) feedforward matrix
E = expected value
F = open-loop dynamics matrix (N_s, N_s)
G = control distribution matrix (N_s, N_c)
GAM = state disturbance distribution matrix (N_s, N_g)
H = measurement scaling matrix (N_o, N_s)
K = estimator gain matrix (N_s, N_o)
Nc = number of controls
Ng = number of process noise sources
Ns = number of states
No = number of observations or measurements
P = covariance matrix of estimate error (N_s, N_s)
Q = white process noise covariance matrix (N_g, N_g)
R = white meas. noise covariance matrix (N_o, N_o)
S = steady-state covariance matrix of control (N_c, N_c)
u = control vector ($N_c, 1$)
v = white measurement noise vector ($N_o, 1$), with zero mean and covariance matrix R
w = white process noise vector ($N_g, 1$), with zero mean and covariance matrix Q
w0 = constant disturbance vector ($N_g, 1$)
x = state vector ($N_s, 1$)
xe = estimate of state vector ($N_s, 1$)
y = output/measurement vector ($N_o, 1$)

ACKNOWLEDGEMENT

I would like to express my sincere appreciation to Professor I.J. Collins, whose assistance and encouragement contributed immeasurably to this research.

I wish to dedicate this thesis to my wife, Brenda. Without her constant love, support, and understanding this work would not have been possible.

I. INTRODUCTION

The purpose of this thesis is to describe the extensive modification and improvement of an existing FORTRAN program (OPTSYS) used in the study, design, and application of Optimal Systems Control theory.

This optimal systems control program was originally developed by Hall [Ref. 1] in 1971 to support his research in rotary-wing VTOL aircraft control systems. The latest program modifications were made by Walker [Ref. 2] and Liu [Ref. 3] of Stanford University, to OPTSYS 4 and OPTSYS 5 respectively. These program versions performed quite satisfactorily in the batch environment, but exhibited varying degrees of user hostility due to data input format requirements and incomplete program documentation.

The original intent of this work was to adapt Walker's modified version of OPTSYS to run interactively under VM/CMS on the IBM 3033; however, the extensive modifications of OPTSYSX now represent a high-level interactive applications software system capable of integrated simulation, analysis, synthesis and design of broad classes of optimal systems control problems. With OPTSYSX users may now evaluate various specialized optimal systems control applications, relieved of the burden of lengthy mathematical program development.

It is assumed that the reader/user is familiar with the basic concepts of Control Theory and Optimal Systems Design. The previous descriptions and program development follow the terminology and symbol/naming conventions of Bryson [Ref. 4].

An explanation of the basic system of equations, the terms and symbolology used, and a program overview including the general methods of solution are presented first.

Interactive program development is then discussed, with an explanation of several alternate options available for data input.

This work concludes with examples of various types of problems demonstrated in the interactive mode, including a copy of each terminal session with the final results. A complete program listing is included in Appendix A.

II. THE OPTSYSX COMPUTER PROGRAM

A. INTRODUCTION

OPTSYSX is a double-precision FORTRAN program employing modern control theory analysis techniques. Although the program was originally written to synthesize controllers for rotary-wing VTOL aircraft [Ref. 5], it has been extensively modified to enable controller, filter, and regulator synthesis as well as transfer function and modal analysis on other types of large, multi-variable systems of equations. The program modifications described in this work now allow rapid numerical computer analysis in a completely interactive mode.

B. SYSTEM/MODEL DESCRIPTION

OPTSYSX treats a linear stationary system model:

$$\dot{x} = [F]x + [G]u + [GAMMA]w \quad (2.1)$$

output equation

$$y = [H]x + [D]u \quad (2.2)$$

measurement equation

$$z = [H]x + [D]w + v \quad (2.3)$$

where

- u = control vector (m X 1)
- v = white measurement noise vector (p X 1)
- w = white process noise vector (q X 1)
- x = state vector (n X 1)

y = output vector ($p \times 1$)

z = measurement vector ($f \times 1$)

[F] is the open-loop dynamics matrix (system matrix or plant matrix); [G] is the control distribution matrix; [GAMMA] is the process noise distribution matrix; [H] is the measurement distribution matrix; and [D] may represent a feed-forward distribution matrix of either the process noise vector (w), or the control vector (u).

The w vector has zero mean value, and a covariance matrix [C], where:

$$E(w) = 0 \quad (2.4)$$

and

$$[C] = E[ww^T] \quad (2.5)$$

The v vector has zero mean value and a covariance matrix [R], where:

$$E(v) = 0 \quad (2.6)$$

and

$$[R] = E[vv^T] \quad (2.7)$$

The quadratic performance (or cost) index for the linear quadratic regulator is the expected value of:

$$J = 1/2 \int \{y^T [A] y + u^T [B] u\} dt \quad (2.8)$$

in the statistical steady-state, where [A] represents an output cost matrix (a weighting on the output variables);

and [E] is a control cost (or control weighting coefficient) matrix.

If full state weighting is desired, [H] is represented by the identity matrix [I].

C. BECGFAM OUTPUT

1. Open-Loop EIGENSYSTEM Calculations

The initial portion of BECGFAM output includes the program flow control flags set by the user for that particular run, the system of equations being modeled, and the open-loop eigenvalue and eigenvector calculations of the [F] matrix.

2. Regulator Synthesis Calculations

In the solution to the optimal regulator problem, full state variable feedback is assumed where:

$$[C] = [B^{-1}][G^T][S] \quad (2.9)$$

and

$$u = -[C]x \quad (2.10)$$

The control gain [C] is a matrix of optimal gains which minimize the cost index expressed in equation (2.8).

For optimal regulator synthesis problems, program output includes the closed-loop eigenvalues and eigenvectors; the control gain [C]; the closed-loop dynamics matrix [F-GC]; and the steady-state gain matrix [S], where [S] is the steady-state solution to the algebraic Riccati equation:

$$S[F] + [F^T]S - S[G][B^{-1}][G^T]S + [H^T][A][H] = 0 \quad (2.11)$$

3. Filter Synthesis Calculations

A Kalman filter or Estimator which describes a continuous time system may be written as:

$$\dot{\hat{x}} = [F]\hat{x} + [K](z - [H]\hat{x}) \quad (2.12)$$

where:

\hat{x} is the state estimate

$[K]$ is a matrix of filter gains,

and the state covariance is described by:

$$E(xx^T) = E(\hat{x}\hat{x}^T) + [P] \quad (2.13)$$

The filter gain matrix $[K]$ of equation (2.12) is obtained from the relationship:

$$[K] = [P][H^T][R^{-1}] \quad (2.14)$$

where $[P]$ is the steady-state solution to the algebraic Riccati equation:

$$[F][P] + [P][F^T] - [P][F^T][R^{-1}][H]P + [G][Q][G^T] = 0 \quad (2.15)$$

representing the error covariance of the estimate \hat{x} . The control covariance is the expected value described by:

$$E(uu^T) = [C]\hat{x}[C^T] \quad (2.16)$$

For the Kalman filter/optimal estimator synthesis problem, CFTSYSX output includes the eigenvalues and eigenvectors of the optimal estimator (Kalman filter); the filter gains $[K]$; the error covariance matrix $[P]$; the covariance

matrix of the estimate \hat{x} ; the state covariance matrix $[X]$ described in equation (2.13), where

$$[X] = E[xx^T] \quad (2.17)$$

and the control covariance matrix $[U]$ described in equation (2.16).

4. Stationary Closed-Loop Calculations

The stationary response of both state and control are presented as root-mean-square values of the state and control covariance matrices $[X]$ and $[U]$ described in equations (2.17) and (2.16) respectively.

E. SOLUTION ALGORITHM

One of the fundamental techniques necessary to quadratic synthesis of optimal control systems is the steady-state solution of the algebraic Riccati equation. This is a non-trivial task due to the iterative nature of the solution.

The steady-state solution by any quadrature method necessitates time increment selection no greater than some fraction of the shortest period of the closed-loop system; imposing a significant computer solution time expenditure on the user as well as the requiring an extensive amount of computer storage capability due to the matrix expansion factor involved. Further, the possible necessity of a time-varying solution of these equations for optimal open loop control or estimation requires the inversion of an $n \times n$ matrix for each time increment where an unsteady solution is desired.

A powerful and efficient alternate method of solution was developed by Bryson and Hall [Ref. 7], based on eigenvector decomposition of the eigensystem of the constant coefficient EULER-LAGRANGE equations.

For the optimal regulator, these equations take the form:

$$\begin{bmatrix} \dot{x} \\ \dot{\lambda} \end{bmatrix} = \begin{bmatrix} F & G^T B^{-1} G \\ A & -F^T \end{bmatrix} \begin{bmatrix} x \\ \lambda \end{bmatrix}$$

For the optimal filter or estimator, the equations are of the form:

$$\begin{bmatrix} \dot{x} \\ \dot{\lambda} \end{bmatrix} = \begin{bmatrix} F & G_{am} * Q * G_{am}^T \\ H^T R^{-1} H & -F^T \end{bmatrix} \begin{bmatrix} x \\ \lambda \end{bmatrix}$$

CFISYSX, and all earlier program versions, use the method of eigenvector decomposition of the EULER-LAGRANGE equations described in [Ref. 7] for quadratic synthesis of control systems. Program calculations are based on the QR algorithm of Francis, modified by Wilkinson [Ref. 8]. Important features of this method of eigensystem analysis are its improved accuracy, and its insensitivity to widely separated eigenvalues.

A more detailed description of the QR algorithm and its numerical applications to eigensystem analysis may be found in [Ref. 8].

E. PROGRAM OVERVIEW

CFISYSX and its 41 subroutines may be divided into three basic categories:

- 1) setup and computation sequencing
- 2) data input
- 3) calculation

A brief and general description of the program modules and subroutines supporting these basic categories concludes this section.

1. Problem Description/Program Flow Control

The MAIN program allows interactive selection of all program flow control flags, and is aided by three subroutines: RDEAL, RDINI, and RDCHAR. A detailed description of these subroutines is provided in Chapter III. Subroutine CHECK verifies the consistency of all requested program options.

2. Interactive Data Input

Interactive data input is performed by the 14 READ__ subroutines. A detailed description of the operation of these subroutines is also included in Chapter III. Internal data generation or external data input is provided by subroutine SETUP.

3. Calculation Sequencing

Subroutine INNER functions as a second MAIN program. It orders the data input/calculation sequences for the type of problem being solved and performs numerous matrix calculations. It is from this subroutine that all input and calculation sequences are ordered and performed.

4. QR Algorithm Transformation

Subroutines MINV, BALANC, ORTHES, ORTRAN, HQF, HQF2, EALBAK, CNCRM, and EREXIT perform the major calculation sequences of the "QR" algorithm.

5. Riccati Equation Calculations

Subroutine FGAIN separates the eigenvalues and eigenvectors of the Euler-Lagrange equations by eigenvector decomposition. RGAIN and subroutine SCOV calculate the steady-state solution of the Riccati equations for the controller or estimator problem. Subroutine SCOV computes

the covariance matrix solution to the algebraic Riccati equation.

6. Modal Calculations

Subroutine MCDE computes the modal transformations required for modal analysis.

7. Transfer Function Calculations

Subroutines TF, POLES, ZEROS, RESID, ACOMP, CCOMP, EQR, and Function SCI perform transfer function computations associated with Modal calculation sequences. Subroutine ESDCAI computes the power spectral density of the outputs or controls of a controlled system.

8. Data Output

Subroutine RAERNT prints all program calculations in object time format. Subroutine MATFRT allows variable format screen viewing of all interactive matrix data input.

III. INTERACTIVE PROGRAM OPERATION

A. DESIGN CONSIDERATIONS

During the development of CPTSYSX, all program modifications and additions were focused primarily toward interactive user operation. Experience has demonstrated that interactive computer communication offers many advantages in the research and problem solving environment. The opportunity for flexible and immediate computer communication, as well as the ability to select alternate solution methods, are significant advantages to the user; advantages which are unavailable in a batch processing environment.

Although previous versions of OPTSYS produced all the desired calculations in the batch environment, the input format and data sequencing and naming conventions were confusing to many users. The user was burdened with the necessity of verifying the correctness of input data format and program flow control flag settings for each program run, in order to ensure the desired calculation sequence was properly performed.

These requirements, combined with incomplete program documentation, prompted a lack of confidence in the results and discouraged many potential users from even attempting to use CPTSYS.

E. PROGRAM LANGUAGE

CPTSYSX is programmed in FORTRAN IV, following the conventions of IBM System/360/370 FORTRAN IV language. Very few program features have been incorporated which are not written in ANSI Standard FORTRAN. These subtle differences notwithstanding, OPTSYSX has been compiled and run

under both FORTRAN IV (G1) and FORTRAN H (Extended) compilers on the IBM 3033. Although the overall program length is in excess of 2800 lines of text, it is considered completely portable from one operating system to another.

On the presumption that most scientific and research personnel are familiar with FORTRAN language, program modifications may be easily undertaken once system operation is understood.

C. GENERAL PROGRAM MODIFICATIONS

All of the previously developed numerical calculation sequences of OPTSYS were retained un-modified in OPTSYSX.

Those program sequences requiring the input of diagonal cost or covariance matrix elements were deleted or modified to improve user flexibility in entering any desired weighting matrix, diagonal or otherwise. This modification streamlined program operation through elimination of several program flow control flags; reduced a measure of user uncertainty; and further decreased the required degree of user program familiarity--promoting uninterrupted operation.

The FEAL subroutines represent variations on the principle of simple, effective methods of interactive input, coupled with error-correction/recovery sequences.

Subroutines RDCHAR, RDINT, and RDREAL were developed by the author and Cdr. P.D. Sullivan to accommodate varying input format requirements; null-string entry protection was developed by Cdr. P.D. Sullivan. These program features are discussed in greater detail later in this chapter.

L. INTERACTIVE PROGRAM DEVELOPMENT

1. Program Flow Control

Initial program development required an understanding of the various problem descriptions as well as program input and calculation sequence. After careful analysis of [Ref. 2], a basic program flow control diagram was established. From this flow control diagram, a logical branching network was formulated whose path could be determined through either binary logic or numerical selection.

Three basic branching categories were established from the various problem description statements:

Logical-----{"Yes" or "No"..}

Integer-----{"1","2",etc...}

Real Number----{"Input the value of...}

From the viewpoint of free-format computer communication, integer and real number input presented no significant problems. FORTRAN compiler language is written such that numerical data input (real number or integer) is expected, thereby requiring only an INTEGER or REAL data type statement within the program. Once the data type has been declared, the desired values may then be input with a free-format READ (5,*) statement.

One note of caution concerning numerical data input in free-format deserves mentioning: Although all FORTRAN compilers treat character string input as an illegal data type conversion, many will automatically convert the inadvertent character entry to a "zero" and continue execution. Protection against inadvertent errors of this type is discussed later in this chapter.

Logical input ("Yes" or "No") poses a unique problem to programmers. The usual method of incorporation is to require the user to convert the logical answer into an integer i.e., "Yes" = 1, "No" = 2. These integers may then be read directly into the program, determining program flow.

Although this method may promote programming ease, it requires the user to adopt an unnatural input pattern--one which increases the possibility of accidental program termination in the event of inadvertent user error.

A more refined (from a programming language standpoint) and ergonomic (from the user viewpoint) method of logical selection involves utilization of the entire character string answer as an input value. This method has been incorporated into OPTSYSX.

The logical strings ("Yes" and "No") are declared as character strings in a data type statement within the program or subprogram. A format statement is also included in the program or subprogram utilizing the "A"-field to specify the desired character field width. A REWIND statement is then incorporated in the specific program or subprogram immediately prior to each logical string input point. This REWIND statement allows the input device (the terminal screen in this case) to read the character string in the same manner as free-format data input. The character field width for this modification was established at A1, allowing streamlined operation with the user typing either "Y" or "Yes" for an affirmative reply; "N" or "No" for a negative reply.

2. General Input Sequencing Requirements

All data input to OPTSYSX is in matrix or vector format. This data input must be correlated in accordance with the problem description and then properly sequenced in order for the program to perform the desired calculations.

The original and modified OPTSYS programs [Ref. 2] and [Ref. 3] required not only problem description knowledge but complete user familiarity with the detailed calculation sequence of the program. The latter point was considered a significant disadvantage. Elimination of this disadvantage

was an area where interactive programming offered the greatest benefits to the potential user; and it was toward this end that the remaining modifications of OPTSYSX were directed.

3. Interactive Data Input

In its calculation sequences, OPTSYSX requires the input of up to 14 unique matrices or vectors. Once the previously described program flow control diagram was constructed, data entry points for each matrix or vector were established. At each of the 16 program data entry points, the required input matrix or vector was determined. Fourteen input subroutines were added to the original program in order to accommodate interactive data entry.

These matrix input subroutines were written such that the user is first informed which specific matrix or vector is required; then prompted for the individual matrix element values. These values are then individually and sequentially entered from the terminal. Once the matrix or vector is filled, it is returned to the terminal screen for user verification and correction if necessary.

Interactive sequential data entry was programmed by means of a two-dimensional DO loop, with a terminal prompt of the matrix name and element position prior to the element value entry. Data element input is via a free-format READ (5,*) program statement.

Once the matrix data entry sequence is complete, that input matrix is returned to the terminal screen in variable format for user ease in row identification. With an arbitrary data field width of 12 characters, the maximum number of matrix elements available on an 80 column terminal screen is six. Provided the matrix column dimension is less than six, this restriction presented no programming format limitations.

For those matrices whose column dimension exceeded six, elements were progressively written on subsequent terminal lines. Once the matrix row is filled the screen is double-spaced, and element display is repeated in the same fashion. This method allows the user to view the matrix much as he would expect to see it, providing the advantage of ease in row and column identification.

Within CPTSYSX, subroutine MATPRT performs this variable-format print sequence. The print sequence was arbitrarily terminated with a matrix size of 16 X 16, presuming that users with larger systems of equations would select alternate forms of data input.

4. Saving Interactive Input

In most control system design problems, the system matrices generally remain relatively unchanged for a desired sequence of design calculations.

In order to relieve the user of the burden of repeated system entry in the interactive mode, several additional program flow control flags were added, allowing the option of saving the entire original system of matrices for subsequent calculations. Separate options for saving each system matrix are automatically offered at the end of each program run.

These matrix saving options provide a further advantage to the user in that the matrices are redisplayed for verification prior to calculation execution. Users may then change individual matrix elements, relieved of the burden of full system re-entry.

5. User-Defined Input Files

Although the basic objective of this work is to provide the user with a totally interactive method of data input, several disadvantages to the method of individual

matrix element input are apparent--input of very large matrices is unwieldy and time-consuming; input of systems of matrices whose elements remain unchanged from run to run is inefficient.

In order to provide an increased measure of user flexibility in data input, subroutine SETUP was modified to include provisions for matrix data input from a data file on the user's disk. The three system matrices, [F],[G], and [GAMMA] may now be input from the user's disk. Minor program modification is required of the user as follows:

- a. FRICMS Filedef commands must be modified or added to reflect the name and location of each data set.
- b. The REAL Format statement (or statements) must be changed to reflect the proper data format of the user's input file.

6. Internal Data Generation

As a further measure of flexibility, the documentation within subroutine SETUP was expanded to include several specific examples of internal matrix data generation. The three system matrices [F], [G], and [GAMMA] may be generated either within user-written two-dimensional DO-loops or by direct assignment statements. These methods may be preferable for the input of very large matrices with few non-zero elements.

A specific example of internal program generation of the output equation [H] matrix is included in subroutine READH. This matrix input method may be preferable for the entry of a large output equation matrix with very few non-zero elements.

Once these modifications have been made to subroutine SETUP or subroutine READH (as desired), the program should be re-compiled and then run in the usual manner. An

interactive program prompt is provided at the beginning of CPISYSX offering the user the option of specifying the desired method of data input.

CPISYSX was further modified to include an option to input the [H] matrix (or other required input matrices) from separate data files. Users with rudimentary programming skills may now modify subroutine READH (or one of the other specific READ subroutines) in the manner previously described for subroutine SETUP or subroutine READH. Detailed examples of the nature and extent of these modifications may be found in Appendix A.

7. Data Entry Correction

In an effort to protect users from errors in data input, an error correction sequence was incorporated into each matrix input subroutine.

Once the entire matrix or vector is displayed on the terminal screen the user is prompted with the question, "Do you wish to change the value of any matrix element? Type 'Yes' or 'No'." If the user types "No", program execution continues.

If the user types "Yes", he is then prompted with three additional statements specifying the row number of the element to be changed, the column number of the element to be changed, and the value to be inserted into that matrix element. After the corrected value is entered, the new matrix values are returned to the screen for re-verification.

This correction sequence continues indefinitely until the user signifies that no additional changes are necessary. Program execution then proceeds normally.

E. USER-ERROR PROTECTION FEATURES

Many interactive computer programs suffer the unkind characteristic of abnormal program termination (without recovery!) should the user inadvertently make an erroneous keyboard entry. Examples of these inadvertent errors include--entry of a keyboard character or character string when the program expects a numerical value; entry of a numerical value when the program is expecting a character string; entry of a null string. In order to preclude abnormal program termination due to these types of inadvertent user errors, several program protection features were incorporated into OPTISYSX.

1. Data Type Conversion Errors

Three subroutines--EDREAL, RDINT, RDCHAR--were added to OPTISYSX in order to ensure that the proper input data type is provided to the program. Subroutine RDREAL is called at any point a real number or zero integer input may be encountered; subroutine RDINT is called at any point a non-zero integer input is required; subroutine RDCHAR is called at any point a character string ("Yes" or "No") input is required.

Within each of these subroutines a null string entry protection loop was incorporated (allowing one recovery); prompting the user for the correct data type input, and returning an error message in the event an incorrect data type is encountered.

Within subroutine RDINT, improper data type entry was further protected by the addition of a three-way IF comparison of entry integer magnitude. This modification precludes illegal (but automatic, with some compilers) data type conversion errors.

These program design features boast the additional advantage of allowing normal program termination at any point in the data input phase by merely pressing the "Enter" key twice.

2. Inconsistent Program Control Flag Errors

Earlier versions of OPTSYS [Ref. 2] and [Ref. 3], did provide user error messages in the event of inconsistent program flow control flags, but terminated the program. This feature was undesirable from the standpoint of smooth interactive program operation, and was improved in OPTSYSX.

Subroutine CHECK was modified to include RETURN statements any time inconsistent program flow control flags are encountered. The user is notified of the type of error encountered; that run termination has occurred; and prompted regarding his desire to return to the beginning of the program or terminate execution completely.

IV. PROGRAM USE AND EXAMPLES

This chapter contains several basic examples of the numerous types of problems which may be solved using CPTSYSX. Included with these examples are copies of each recorded terminal session.

Potential users should examine carefully the example of program failure found in Section D. This example clearly demonstrates that unstable modes or incorrect choice of certain design parameters may cause program failure (and incorrect output!), even with the highly stable "QA" algorithm. It also indicates one possible method of correcting this type of failure by merely making a very small change to one of the design parameters.

A. OPEN-LOOP EIGENSYSTEM ANALYSIS

The following open-loop eigenvalue example was taken from [Ref. 9, p.669].

Examination of the following program output shows open-loop eigenvalues at -1, -2, and -3. Note that the eigenvectors of the left and right eigenvector matrices (pa. 33) correspond in column fashion to the open-loop eigenvalues calculated immediately above them (pa. 32).

The full terminal session is recorded below, with user input in lower case letters following each "?".

```
record on  
BEGIN RECORDING OF TERMINAL SESSION  
R; T=0.01/0.02 21:49:30  
filedef C6 term (recfr fa blksize 133  
global txtlib fortmod2 mod2eeh imslip ncninsl  
load cptsysx (start
```


EXECUTION BEGINS...

OPTSYSX IS A COMPLETELY INTERACTIVE OPTIMAL SYSTEMS CONTROL PROGRAM. IT WILL SOLVE NUMEROUS CONTROL PROBLEMS ON THE FOLLOWING TYPES OF SYSTEMS CONTROL EQUATIONS:

$$\dot{X} = (F) * X + (G) * U + (GAM) * (W + W_0)$$

MEASUREMENT EQUATION--

$$Z = (H) * X + (D) * W$$

REGULATOR PERFORMANCE INDEX--

$$J = 1/2 * \int (Y^t * (A) * Y + U^t * (B) * U) dt$$

STATE FEEDBACK GAIN DEFINITION--

$$U = -(C) * X$$

DO YOU WISH TO CONTINUE? TYPE "YES" OR "NO".

yes

--DATA ENTRY--

ALTHOUGH OPTSYSX IS SPECIFICALLY DESIGNED TO READ ALL MATRIX DATA INTERACTIVELY, SEVERAL ALTERNATE METHODS ARE AVAILABLE TO USERS:

METHOD 1--THE "F", "G", AND "GAMMA" MATRICES

MAY BE READ FROM SEPARATE DATA FILES.

METHOD 2--THE "F", "G", AND "GAMMA" MATRICES MAY BE

EXPLICITLY DEFINED WITHIN SUBROUTINE "SETUP".

(NOTE: IN EITHER CASE, THE USER SHOULD OBTAIN A COPY OF THE PROGRAM LISTING AND EXAMINE THE EXAMPLES CONTAINED IN S/R "SETUP".)

DO YOU WISH TO CONTINUE? TYPE "YES" OR "NO".

yes

DO YOU WISH TO INPUT THE "F", "G", AND "GAMMA" MATRICES FROM SUBROUTINE "SETUP" IN THE METHOD DESCRIBED ON THE PREVIOUS SCREEN? TYPE "YES" OR "NO".

no

GENERAL OPTSYSX OPTIONS:

OPTION 1 -- SYSTEM ANALYSIS WITHOUT

CLOSED-LOOP EIGENSYSTEM CALCULATIONS.

OPTION 2 -- SYSTEM ANALYSIS WITH OPEN-LOOP

EIGENSYSTEM CALCULATIONS.

OPTION 3 -- CREN-LOCF EIGENSYSTEM FOUND
AND PROGRAM TERMINATES.

("F"-MATRIX ENTRY FOLLOWS IMMEDIATELY.)

OPTION 4 -- MODAL DISTRIBUTION MATRICES COMPUTED
WITHOUT FILTER OR REGULATOR SYNTHESIS
OR STEADY-STATE ANALYSIS.

SELECT AN OPTION: 1,2,3, OR 4.

?

3

ENTER THE # OF STATES (NS) OF THE SYSTEM MATRIX
"F"-MATRIX .

?

3

FLAG/PARAMETER SETTINGS FOR THIS RUN ARE AS FOLLOWS:

IOL	IC	IR	ISS	IM	ITF1	ITF2	ITF3	IFDFW	IE	IDRUG
2	0	0	0	C	0	0	0	0	0	0
ISET	IDSTAE	IPSD	IYU	INCRM	IREG	NS	NC	NOB	NG	
0	0	0	C	0	0	3	0	0	0	0

ORDER OF SYSTEM = 3

NUMBER OF CONTROLS = 0

NUMBER OF OBSERVATIONS = 0

NUMBER OF PROCESS NOISE SOURCES = 0

ENTER THE SYSTEM MATRIX "F"-MATRIX

DIMENSION = # STATES (NS) X # STATES (NS)

THE ELEMENT F(1, 1) =

?

0

THE ELEMENT F(1, 2) =

?

1

THE ELEMENT F(1, 3) =

?

0

THE ELEMENT F(2, 1)=

?

C

THE ELEMENT F(2, 2)=

?

C

THE ELEMENT F(2, 3)=

?

1

THE ELEMENT F(3, 1)=

?

-6

THE ELEMENT F(3, 2)=

?

-11

THE ELEMENT F(3, 3)=

?

-6

THE SYSTEM MATRIX "F"-MATRIX ...

C.C	1.00000	0.0
0.0	0.0	1.00000
-6.00000	-11.00000	-6.00000

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?

TYPE "YES" OR "NO".

n

OPEN LCCP DYNAMICS MATRIX.....F..

0.0	0.1000E+01	0.0
0.0	0.0	0.1000D+01
-0.6000E+01	-0.1100E+02	-0.6000D+01

OPEN LCCP EIGENVALUES.....DET(SI-F) ..

-1.0000E+00:-2.0000E+00:-3.0000D+00:

OPEN ICCE RIGHT EIGENVECTOR MATRIX.....T.....

-5.773503D-01 -2.182179D-01 1.048285D-01
5.773503D-01 4.364358D-01 -3.144855D-01
-5.773503D-01 -8.728716D-01 9.434564D-01

OPEN ICCE LEFT EIGENVECTOR MATRIX.....T-INV..

-5.196152D+00 -4.330127D+00 -8.660254D-01
1.374773D+01 1.833030D+01 4.582576D+00
9.539392D+00 1.430909D+01 4.769696D+00

ANALYSIS COMPLETE. DO YOU WANT ANOTHER RUN?
TYPE "YES" OR "NO".

IO

.....CPTISYSX IS NOW TERMINATED.....

R; T=C.2E/0.89 21:52:08

record cff

END RECORDING OF TERMINAL SESSION

E. REGULATOR SYNTHESIS

The following regulator synthesis example was taken from "Lecture Notes on Advanced Control Systems", by Professor D.J. Collins of the Naval Postgraduate School, Monterey, Ca. This example involved determination of the optimal regulator gains based on an arbitrarily chosen quadratic index; with the various system and cost matrices described below.

Examination of the extensive program output indicates that the optimal regulator gains are: -5.0 and $-\sqrt{10.0}$. The algebraic sign of the gains is consistent with the definiteness displayed on the first screen of the program (p. 34).

The full terminal session is recorded below, with user input in lower case letters following each "?".

record cr

BEGIN RECORDING OF TERMINAL SESSION

R; T=C.01/C.02 13:55:26

filedef C6 term (recfr fa blksize 133

global titlib fortmod2 mod2eeh imslap nonimsi

load cptsysx (start

EXECUTION BEGINS...

CPTSYSX IS A COMPLETELY INTERACTIVE OPTIMAL SYSTEMS CONTROL PROGRAM. IT WILL SOLVE NUMEROUS CONTROL PROBLEMS ON THE FOLLOWING TYPES OF SYSTEMS CONTROL EQUATIONS:

$$\dot{X} = (F) * X + (G) * U + (GAM) * (W + W_0)$$

MEASUREMENT EQUATION--

$$Z = (H) * X + (D) * W + V$$

REGULATOR PERFORMANCE INDEX--

$$J = 1/2 * \int (Y^t * (A) * Y + U^t * (B) * U) dt$$

STATE FEEDBACK GAIN DEFINITION--

$$U = -(C) * X$$

DO YOU WISH TO CONTINUE? TYPE "YES" OR "NO".

Y

--DATA ENTRY--

ALTHOUGH OPTSYSX IS SPECIFICALLY DESIGNED TO READ ALL MATRIX DATA INTERACTIVELY, SEVERAL ALTERNATE METHODS ARE AVAILABLE TO USERS:

METHOD 1--THE "F", "G", AND "GAMMA" MATRICES MAY BE READ FROM SEPARATE DATA FILES.

METHOD 2--THE "F", "G", AND "GAMMA" MATRICES MAY BE EXPLICITLY DEFINED WITHIN SUBROUTINE "SETUP".

(NOTE: IN EITHER CASE, THE USER SHOULD OBTAIN A COPY OF THE PROGRAM LISTING AND EXAMINE THE EXAMPLES CONTAINED IN S/R "SETUP".)

DO YOU WISH TO CONTINUE? TYPE "YES" OR "NO".

Y

DO YOU WISH TO INPUT THE "F", "G", AND "GAMMA" MATRICES FROM SUBROUTINE "SETUP" IN THE METHOD DESCRIBED ON THE PREVIOUS SCREEN? TYPE "YES" OR "NO".

R

GENERAL CFTSYSX OPTIONS:

- OPTION 1 -- SYSTEM ANALYSIS WITHOUT
OPEN-LOOP EIGENSYSTEM CALCULATIONS.
- OPTION 2 -- SYSTEM ANALYSIS WITH OPEN-LOOP
EIGENSYSTEM CALCULATIONS.
- OPTION 3 -- OPEN-LOOP EIGENSYSTEM FOUND
AND PROGRAM TERMINATES.
("F"-MATRIX ENTRY FOLLOWS IMMEDIATELY.)
- OPTION 4 -- MODAL DISTRIBUTION MATRICES COMPUTED
WITHOUT FILTER OR REGULATOR SYNTHESIS
OR STEADY-STATE ANALYSIS.
- SELECT AN OPTION: 1, 2, 3, OR 4.

?

2

DO YOU DESIRE RMS VALUES OF STATE AND CONTROL?
TYPE "YES" OR "NO".

EO

CFTSYSX LQR/CLASSICAL OPTIONS:

- OPTION 1 -- OPTIMAL FILTER AND/OR REGULATOR
SYNTHESIS WITH NO EXTERNAL "C" OR "K"
MATRIX INPUT.
- OPTION 2 -- OPTIMAL FILTER AND/OR REGULATOR
SYNTHESIS WITH EXTERNAL "C"
MATRIX INPUT.
- OPTION 3 -- OPTIMAL FILTER AND/OR REGULATOR
SYNTHESIS WITH EXTERNAL "K"
MATRIX INPUT.
- OPTION 4 -- OPTIMAL FILTER AND/OR REGULATOR
SYNTHESIS WITH EXTERNAL "C" AND "K"
MATRIX INPUT.
- SELECT AN OPTION: 1, 2, 3, OR 4.

?

1

DO YOU WISH TO DETERMINE THE STEADY-STATE RESPONSE
FOR A CONSTANT DISTURBANCE?

TYPE "YES" OR "NO".

EO

DO YOU WISH TO DETERMINE THE MODAL DISTRIBUTION
AND GAIN MATRICES?

TYPE "YES" OR "NO".

EO

OPEN-LOOP TRANSFER FUNCTION OPTIONS:

OPTION 1 -- NO OPEN-LOOP TRANSFER FUNCTIONS COMPUTED.

OPTION 2 -- POLES, RESIDUES, AND ZEROS COMPUTED.

OPTION 3 -- ONLY POLES AND ZEROS COMPUTED.

OPTION 4 -- ONLY POLES AND RESIDUES COMPUTED.

SELECT AN OPTION: 1, 2, 3, OR 4.

?

1

NOISE TRANSFER FUNCTION OPTIONS:

OPTION 1 -- NO NOISE TRANSFER FUNCTIONS COMPUTED.

OPTION 2 -- POLES, RESIDUES, AND ZEROS COMPUTED.

OPTION 3 -- ONLY POLES AND ZEROS COMPUTED.

OPTION 4 -- ONLY POLES AND RESIDUES COMPUTED.

SELECT AN OPTION: 1, 2, 3, OR 4.

?

1

COMPENSATOR TRANSFER FUNCTION OPTIONS:

OPTION 1 -- NO COMP. TRANSFER FUNCTIONS COMPUTED.

OPTION 2 -- POLES, RESIDUES, AND ZEROS COMPUTED.

OPTION 3 -- ONLY POLES AND ZEROS COMPUTED.

OPTION 4 -- ONLY POLES AND RESIDUES COMPUTED.

(NOTE: A COMPENSATOR TRANSFER FUNCTION CAN BE
COMPUTED ONLY IF BOTH A REGULATOR AND
FILTER ARE SYNTHESIZED AND/OR INPUT.)

SELECT AN OPTION: 1, 2, 3, OR 4.

?

1

WILL A FEED-FORWARD DISTRIBUTION MATRIX

("D" - MATRIX) BE INPUT ?

TYPE "YES" OR "NO".

NO

THIS OPTION DETERMINES THE CRITERIA FOR DECIDING WHEN A MARKOV PARAMETER IS ZERO-THE MARKOV PARAMETER INDICATES THE ORDER OF THE NUMERATOR POLYNOMIAL OF EACH TRANSFER FUNCTION.

ALL "N" ZEROS OF THIS POLYNOMIAL ARE PRINTED OUT AND THIS TEST TELLS HOW MANY EXTRA ROOTS EXIST AT $Z = 0$. LESS THAN $10.0^{**}(-IE)$ IS CONSIDERED ZERO.

THE DEFAULT VALUE OF THIS PARAMETER (IE) IS 6.

IN OTHER WORDS, $IE = 1.0E-6$.

IF YOU DESIRE A DIFFERENT MARKOV CRITERIA, TYPE THE INTEGER VALUE.

IF YOU DESIRE THE DEFAULT VALUE, TYPE "0" (ZERO)

?

C

DO YOU DESIRE TO SYNTHESIZE A STABLE FILTER (OR REGULATOR) BY DESTABILIZING THE ORIGINAL SYSTEM?

(NOTE:WORKS FOR FILTER OR REGULATOR BUT NOT FOR BOTH IN THE SAME RUN.)

TYPE "YES" OR "NO".

NO

DO YOU DESIRE TO PRINT THE EULER-LAGRANGE EIGENSYSTEM PRIOR TO DECOMPOSITION (FOR CHECKING THE PROGRAM)?

TYPE "YES" OR "NO".

yes

POWER SPECTRAL DENSITY (PSD) OPTION 1 :

OPTION 1 -- COMPUTE THE PSD OF THE OUTPUTS AND/OR THE CORRELATIONS OF THE CONTROLLED SYSTEM WHEN FORCED BY PROCESS AND MEASUREMENT NOISE.

(NOTE: BOTH A REGULATOR AND A FILTER MUST BE RESIDENT IN THE PROGRAM TO USE THIS OPTION. (

OPTION 2 -- SAME AS OPTION 1 ABOVE BUT ONLY PRINT THE RESIDUES OF EACH TRANSFER FUNCTION USED IN THE PSD COMPUTATION.

OPTION 3 -- NOT DESIRED.

SELECT AN OPTION: 1, 2, OR 3.

?

3

DO YOU DESIRE REGULATOR SYNTHESIS ONLY?

TYPE "YES" OR "NO".

yes

ENTER THE # OF STATES (NS) OF THE SYSTEM MATRIX
("F"-MATRIX).

?

2

ENTER THE # OF CONTROLS (NC) OF THE CONTROL SYSTEM MODEL
("G"-MATRIX).

?

1

ENTER THE # OF MEASUREMENTS OR OBSERVATIONS (NO) OF THE
("H"-MATRIX).

?

2

ENTER THE # OF PROCESS NOISE SOURCES (NG) OF THE
("GAMMA"-MATRIX).

?

C

FLAG/PARAMETER SETTINGS FOR THIS RUN ARE AS FOLLOWS:

IOL	IQ	IR	ISS	IM	ITF1	ITF2	ITF3	IFDFW	IE	IDDEBUG
1	0	C	0	C	0	0	0	0	0	1
ISET	IDSTAE	IPSD	IYU	INCEM	IREG	NS	NC	NOB	NG	
0	0	0	C	0	1	2	1	2	0	

ORDER OF SYSTEM = 2

NUMBER OF CONTROLS = 1

NUMBER OF OBSERVATIONS = 2

NUMBER OF PROCESS NOISE SOURCES = 0

ENTER THE SYSTEM MATRIX "F"-MATRIX

DIMENSION = # STATES (NS) X # STATES (NS)

THE ELEMENT F(1, 1) =

?
0
?
1
?
0
?
C

THE ELEMENT F(1, 2)=

THE ELEMENT F(2, 1)=

THE ELEMENT F(2, 2)=

THE SYSTEM MATRIX "F"-MATRIX ...

C.0 1.00000

C.0 0.0

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?

TYPE "YES" OR "NO".

EO

CFEN LCCP DYNAMICS MATRIX.....F..

0.0 0.1000E+01

0.0 0.0

CFEN LCCP EIGENVALUES.....DET(SI-F)..

0.0 : 0.0 :

CFEN LCCP RIGHT EIGENVECTOR MATRIX.....T.....

1.00000E+00 -1.00000E+00

0.0 2.220446D-16

CFEN LCCP LEFT EIGENVECTOR MATRIX.....T-INV..

1.00000E+00 4.50360E+15

0.0 4.50360E+15

ENTER THE MEASUREMENT SCALING MATRIX "H"-MATRIX .

DIMENSION = # OBSERVATIONS (NO) X # STATES (NS)

THE ELEMENT H(1, 1)=

?
1

?
C THE ELEMENT H (1, 2) =

?
C

THE ELEMENT H (2, 1) =

?
O

THE ELEMENT H (2, 2) =

?
1

THE MEASUREMENT SCALING MATRIX "H"-MATRIX ...

1.00000 0.0
0.0 1.00000

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?
TYPE "YES" OR "NO".

EO

MEASUREMENT SCALING MATRIX.....H..

0.1000E+01 0.0
0.0 0.1000E+01

ENTER THE OUTPUT MEASUREMENT COST MATRIX "A"-MATRIX .
DIMENSION = # OBSERVATIONS (NO) X # OBSERVATIONS (NO)
THE ELEMENT A (1, 1) =

?
25

THE ELEMENT A (1, 2) =

?
C

THE ELEMENT A (2, 1) =

?
O

THE ELEMENT A (2, 2) =

?
C

THE OUTPUT MEASUREMENT COST MATRIX "A"-MATRIX ...
25.00000 0.0

0.0 0.0

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?
TYPE "YES" OR "NO".

EO

OUTPUT COST MATRIX.....A..

0.25(0E+02 0.0
0.0 0.0

ENTER THE CONTROL DISTRIBUTION MATRIX "G"-MATRIX .
DIMENSION = # STATES (NS) X # CONTROLS (NC)
THE ELEMENT G(1, 1)=

?

0

THE ELEMENT G(2, 1)=

?

1

THE CONTROL DISTRIBUTION MATRIX "G"-MATRIX ...

0.0
1.00000

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?
TYPE "YES" OR "NO".

EO

ENTER THE CONTROL COST WEIGHTING MATRIX "B"-MATRIX
DIMENSION = # CONTROLS (NC) X # CONTROLS (NC)
THE ELEMENT B(1, 1)=

?

1

THE CONTROL COST MATRIX.....B...

1.00000

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?
TYPE "YES" OR "NO".

EO

THE CONTROL DISTRIBUTION MATRIX.....G..

0.0
0.10(0E+01

THE CONTROL COST MATRIX.....B..
0.10000E+01

EULER-LAGRANGE SYSTEM MATRIX...
0.0 1.000000E+00 0.0 0.0
0.0 0.0 0.0 -1.000000E+00
-2.500000E+01 0.0 0.0 0.0
0.0 0.0 -1.000000E+00 0.0

EIGENVALUES AND EIGENVECTORS OF THE 2N X 2N
EULER-LAGRANGE SYSTEM AFTER HQR2.....
-1.581139E+00 1.581139E+00
-1.581139E+00 -1.581139E+00
1.581139E+00 1.581139E+00
1.581139E+00 -1.581139E+00
-7.430443E-02 7.168812E-02 -1.824925E-01 -1.503482E-01
4.136755E-03 -2.308345E-01 -5.082459E-02 -5.262675E-01
-1.154172E+00 -2.068377E-02 2.631337E+00 -2.541229E-01
-3.584406E-01 -3.715222E-01 -7.517412E-01 9.124627E-01

EIGENSYSTEM OF OPTIMAL REGULATOR.....
EIGENVECTORS FROM RGAIN PRIOR TO CNORM
-1.459925E-01 -2.616313E-03
2.349712E-01 -2.266977E-01

C-LCCF OPTIMAL REG. E-VALUES... DET(SI-F+G*C)..
-1.581141E+00, 1.581141E+00:

C-LCCF RIGHT EIGENVECTOR MATRIX.....M....
-3.162278E-01 -3.162278E-01
1.000000E+00 0.0

CONTROL EIGENVECTOR MATRIX.....C*M..
-1.581139E+00 1.581139E+00

```
C-ICCE CPT. REG. LEFT E-VECTOR MATRIX..M-INV..
0.0          1.000000D+00
-3.162278D+00 -1.000000D+00

THE ORIGINAL FEEDBACK GAIN CONTROL MATRIX...C=BLK*G1*F...
-5.000001+00 -3.16231+00
```

```
THE CLOSED LOOP DYNAMICS MATRIX .....F-G*C..
0.0          1.000000D+00
-5.000000D+00 -3.162278D+00
```

```
ANALYSIS COMPLETE. DO YOU WANT ANOTHER RUN?
TYPE "YES" OR "NO".
```

no

```
.....OPTSYSX IS NOW TERMINATED.....
```

E; T=C.42/1.85 14:03:03

reccid off

END FEEDLING OF TERMINAL SESSION

C. FILTER SYNTHESIS

The following Kalman filter synthesis example was taken from "Lecture Notes on Advanced Control Systems", by Professor L.J. Collins of the Naval Postgraduate School, Monterey, Ca.

This example involved determination of the optimal filter gains of an arbitrary system; modeled nearly identically to the previous regulator problem.

In its present configuration, OPTSYSX program sequencing requires the design of an optimal regulator, prior to performing any optimal estimator synthesis. In order to comply with built-in program sequencing conventions, and circumvent program difficulties which may not be specified in the particular system model, optimal filter synthesis may be accomplished by entering the identity matrix [I] in these

program input sequences requiring the entry of an output cost (weighting) matrix. Although the optimal regulator calculations may differ from those expected, the optimal estimator calculations will be correct for the system model.

Examination of the extensive program output indicates that the optimal filter gains are: -5.0, and -SQRT(2.0).

The full terminal session is recorded below, with user input in lower case letters following each "?".

record of

BEGIN RECORDING OF TERMINAL SESSION

B; T=C.01/0.02 21:49:30

filedef C6 term (recfm fa blksize 133

global txtlib fortmod2 mcd2eeh insldp ncaims1

load cptsysx (start

EXECUTION BEGINS...

OPTSYSX IS A COMPLETELY INTERACTIVE OPTIMAL SYSTEMS CONTROL PROGRAM. IT WILL SOLVE NUMEROUS CONTROL PROBLEMS ON THE FOLLOWING TYPES OF SYSTEMS CONTROL EQUATIONS:

$$\dot{X} = (F)*X + (G)*U + (GAM)*(W+W0)$$

MEASUREMENT EQUATION--

$$Z = (H)*X + (D)*W + V$$

REGULATOR PERFORMANCE INDEX--

$$J = 1/2 * \text{INTEGRAL } (Y^t*(A)*Y + U^t*(B)*U) DT$$

STATE FEEDBACK GAIN DEFINITION--

$$U = -(C)*X$$

DO YOU WISH TO CONTINUE? TYPE "YES" OR "NO".

yes

--DATA ENTRY--

ALTHOUGH OPTSYSX IS SPECIFICALLY DESIGNED TO READ ALL MATRIX DATA INTERACTIVELY, SEVERAL ALTERNATE METHODS ARE AVAILABLE TO USERS:

METHOD 1--THE "F", "G", AND "GAMMA" MATRICES MAY BE READ FROM SEPARATE DATA FILES.

METHOD 2--THE "F", "G", AND "GAMMA" MATRICES MAY BE

EXPLICITLY DEFINED WITHIN SUBROUTINE "SETUP".
(NOTE: IN EITHER CASE, THE USER SHOULD OBTAIN A COPY
OF THE PROGRAM LISTING AND EXAMINE
THE EXAMPLES CONTAINED IN S/R "SETUP".)
DO YOU WISH TO CONTINUE? TYPE "YES" OR "NO".

yes

DO YOU WISH TO INPUT THE "F", "G", AND "GAMMA"
MATRICES FROM SUBROUTINE "SETUP" IN THE
METHOD DESCRIBED ON THE PREVIOUS SCREEN?
TYPE "YES" OR "NO".

no

GENERAL CPTSYSX OPTIONS:

OPTION 1 -- SYSTEM ANALYSIS WITHOUT
OPEN-LOOP EIGENSYSTEM CALCULATIONS.

OPTION 2 -- SYSTEM ANALYSIS WITH OPEN-LOOP
EIGENSYSTEM CALCULATIONS.

OPTION 3 -- OPEN-LOOP EIGENSYSTEM FOUND
AND PROGRAM TERMINATES.

("F"-MATRIX ENTRY FOLLOWS IMMEDIATELY.)

OPTION 4 -- MODAL DISTRIBUTION MATRICES COMPUTED
WITHOUT FILTER OR REGULATOR SYNTHESIS
OR STEADY-STATE ANALYSIS.

SELECT AN OPTION: 1,2,3, OR 4.

?

1

DO YOU DESIRE RMS VALUES OF STATE AND CONTROL?
TYPE "YES" OR "NO".

no

CPTSYSX LQR/CLASSICAL OPTIONS:

OPTION 1 -- OPTIMAL FILTER AND/OR REGULATOR
SYNTHESIS WITH NO EXTERNAL "C" OR "K"
MATRIX INPUT.

OPTION 2 -- OPTIMAL FILTER AND/OR REGULATOR
SYNTHESIS WITH EXTERNAL "C"
MATRIX INPUT.

OPTION 3 -- OPTIMAL FILTER AND/OR REGULATOR
SYNTHESIS WITH EXTERNAL "K"
MATRIX INPUT.

OPTION 4 -- OPTIMAL FILTER AND/OR REGULATOR
SYNTHESIS WITH EXTERNAL "C" AND "K"
MATRIX INPUT.

SELECT AN OPTION: 1, 2, 3, OR 4.

?

1

DO YOU WISH TO DETERMINE THE STEADY-STATE RESPONSE
FOR A CONSTANT DISTURBANCE?

TYPE "YES" OR "NO".

no

DO YOU WISH TO DETERMINE THE MODAL DISTRIBUTION
AND GAIN MATRICES?

TYPE "YES" OR "NO".

no

OPEN-LOOP TRANSFER FUNCTION OPTIONS:

OPTION 1 -- NO OPEN-LOOP TRANSFER FUNCTIONS COMPUTED.

OPTION 2 -- POLES, RESIDUES, AND ZEROS COMPUTED.

OPTION 3 -- ONLY POLES AND ZEROS COMPUTED.

OPTION 4 -- ONLY POLES AND RESIDUES COMPUTED.

SELECT AN OPTION: 1, 2, 3, OR 4.

?

1

NOISE TRANSFER FUNCTION OPTIONS:

OPTION 1 -- NO NOISE TRANSFER FUNCTIONS COMPUTED.

OPTION 2 -- POLES, RESIDUES, AND ZEROS COMPUTED.

OPTION 3 -- ONLY POLES AND ZEROS COMPUTED.

OPTION 4 -- ONLY POLES AND RESIDUES COMPUTED.

SELECT AN OPTION: 1, 2, 3, OR 4.

?

1

COMPENSATOR TRANSFER FUNCTION OPTIONS:

OPTION 1 -- NO COMP. TRANSFER FUNCTIONS COMPUTED.

OPTION 2 -- POLES, RESIDUES, AND ZEROS COMPUTED.

OPTION 3 -- ONLY POLES AND ZEROS COMPUTED.

OPTION 4 -- ONLY POLES AND RESIDUES COMPUTED.

(NOTE: A COMPENSATOR TRANSFER FUNCTION CAN BE
COMPUTED ONLY IF BOTH A REGULATOR
AND FILTER ARE SYNTHESIZED
AND/OR INPUT.)

SELECT AN OPTION: 1, 2, 3, OR 4.

?

1

WILL A FEED-FORWARD DISTRIBUTION MATRIX
"L" - MATRIX BE INPUT ?

TYPE "YES" OR "NO".

DO

THIS OPTION DETERMINES THE CRITERIA FOR DECIDING WHEN
A MARKOV PARAMETER IS ZERO-THE MARKOV PARAMETER
INDICATES THE ORDER OF THE NUMERATOR POLYNOMIAL OF EACH
TRANSFER FUNCTION.

ALL "N" ZEROS OF THIS POLYNOMIAL ARE PRINTED OUT AND
THIS TEST TELLS HOW MANY EXTRA ROOTS EXIST AT $Z = 0$.
LESS THAN $10.0^{**}(-IE)$ IS CONSIDERED ZERO.

THE DEFAULT VALUE OF THIS PARAMETER (IE) IS 6.

IN OTHER WORDS, $IE = 1.0E-6$.

IF YOU DESIRE A DIFFERENT MARKOV CRITERIA,
TYPE THE INTEGER VALUE.

IF YOU DESIRE THE DEFAULT VALUE, TYPE "0" (ZERO)

?

C

DO YOU DESIRE TO SYNTHESIZE A STABLE FILTER (OR REGULATOR)
BY DESTABILIZING THE ORIGINAL SYSTEM?

(NOTE:WORKS FOR FILTER OR REGULATOR BUT NOT FOR BOTH
IN THE SAME RUN.)

TYPE "YES" OR "NO".

DO

DO YOU DESIRE TO PRINT THE EULER-LAGRANGE EIGENSYSTEM
PRICR TO DECCMPOSITION (FOR CHECKING THE PROGRAM)
TYPE "YES" OR "NO".

NO

POWER SPECTRAL DENSITY (PSD) OPTION 1 :

OPTION 1 -- COMPUTE THE PSD OF THE OUTPUTS AND/OR
THE CONTROLS OF THE CONTROLLED SYSTEM
WHEN FORCED BY PROCESS AND MEASUREMENT
NOISE. (NOTE: BOTH A REGULATOR AND A
FILTER MUST BE RESIDENT IN THE PROGRAM
TO USE THIS OPTION.)

OPTION 2 -- SAME AS OPTION 1 ABOVE BUT ONLY PRINT THE
RESIDUES OF EACH TRANSFER FUNCTION
USED IN THE PSD COMPUTATION.

OPTION 3 -- NOT DESIRED.

SELECT AN OPTION: 1, 2, OR 3.

?

3

DO YOU DESIRE REGULATOR SYNTHESIS ONLY?

TYPE "YES" OR "NO".

NO

ENTER THE # OF STATES (NS) OF THE SYSTEM MATRIX
"F"-MATRIX .

?

2

ENTER THE # OF CONTROLS (NC) OF THE SYSTEM MODEL
"G"-MATRIX .

?

1

ENTER THE # OF MEASUREMENTS OR OBSERVATIONS (NO)
"H"-MATRIX .

?

1

ENTER THE # OF PROCESS NOISE SOURCES (NG)
"GAMMA"-MATRIX .

?
0

FLAG/PARAMETER SETTINGS FOR THIS RUN ARE AS FOLLOWS:

IOL	IC	IF	ISS	IM	IIF1	IIF2	ITF3	IFDEFW	IE	IEDEGG
0	0	C	0	0	0	0	0	0	0	0
ISSET	ILSTAE	IPSD	IYU	INCRM	IREG	NS	NC	NOB	NG	
0	C	0	0	0	0	2	1	1	0	

ORDIE OF SYSTEM = 2
NUMBER OF CCNTRCLS = 1
NUMBER OF CBSERVATIONS = 1
NUMBER OF PROCESS NOISE SOURCES = 0

ENTER THE SYSTEM MATRIX "F"-MATRIX
DIMENSION = # STATES (NS) X # STATES (NS)
THE ELEMENT F(1, 1)=

?
0

THE ELEMENT F(1, 2)=

?
1

THE ELEMENT F(2, 1)=

?
0

THE ELEMENT F(2, 2)=

?
C

THE SYSTEM MATRIX "F"-MATRIX ...

C.0	1.00000
0.0	0.0

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?
TYPE "YES" OR "NO".

EO

CFEN LCCP DYNAMICS MATRIX.....F..

0.0	0.1000E+01
-----	------------

0.0 0.0

ENTER THE MEASUREMENT SCALING MATRIX "H"-MATRIX .

DIMENSION = # OBSERVATIONS (NO) X # STATES (NS)

THE ELEMENT H(1, 1)=

?

1

THE ELEMENT H(1, 2)=

?

C

THE MEASUREMENT SCALING MATRIX "H"-MATRIX ...

1.00000 0.0

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?

TYPE "YES" OR "NO".

NO

MEASUREMENT SCALING MATRIX.....H..

0.10001+01 0.0

ENTER THE OUTPUT MEASUREMENT COST MATRIX "A"-MATRIX .

DIMENSION = # OBSERVATIONS (NO) X # OBSERVATIONS (NO)

THE ELEMENT A(1, 1)=

?

1

THE OUTPUT MEASUREMENT COST MATRIX "A"-MATRIX ...

1.00000

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?

TYPE "YES" OR "NO".

NO

OUTPUT COST MATRIX.....A..

0.10001+01

ENTER THE CONTROL DISTRIBUTION MATRIX "G"-MATRIX .

DIMENSION = # STATES (NS) X # CONTROLS (NC)

THE ELEMENT G(1, 1)=

?

0

THE ELEMENT G(2, 1)=
?
0
THE CONTROL DISTRIBUTION MATRIX "G"-MATRIX ...
0.0
0.0
DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?
TYPE "YES" OR "NO".

yes
ENTER THE ROW NUMBER OF THE ELEMENT TO BE CHANGED.
?
2
ENTER THE COLUMN NUMBER OF THE ELEMENT TO BE CHANGED.
?
1

THE ELEMENT G(2, 1)=
?
1
THE CONTROL DISTRIBUTION MATRIX "G"-MATRIX ...
0.0
1.00000
DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?
TYPE "YES" OR "NO".

no
ENTER THE CONTROL COST WEIGHTING MATRIX "B"-MATRIX
DIMENSION = # CONTROLS NC X # CONTROLS NC
THE ELEMENT B(1, 1)=
1
?

THE CONTROL COST MATRIX.....B...
1.00000
DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?
TYPE "YES" OR "NO".

no

```

THE CONTROL DISTRIBUTION MATRIX.....G..
0.0
0.1000E+01

THE CONTROL COST MATRIX.....E..
0.1000E+01

EIGENSYSTEM OF OPTIMAL REGULATOR.....

C-LCCF OPTIMAL REG. E-VALUES...DET(SI-F+G*C)..
-7.07107E-01, 7.07107E-01:

C-LCCF RIGHT EIGENVECTOR MATRIX.....M....
-7.071068E-01 -7.071068D-01
1.000000D+00 0.0

CONTROL EIGENVECTOR MATRIX.....C*M..
-7.071068E-01 7.071068D-01

C-LCCF OPT. REG. LEFT E-VECTOR MATRIX..M-INV..
0.0          1.000000D+00
-1.414214E+00 -1.000000D+00

THE OPTIMAL FEEDBACK GAIN CONTROL MATRIX...C=BI*V*G*T*S...
-1.0000E+00 -1.4142E+00

THE CLOSED LOOP DYNAMICS MATRIX .....F-G*C..
0.0          1.000000D+00
-1.000000D+00 -1.414214D+00

```

ANALYSIS COMPLETE. DO YOU WANT ANOTHER RUN?
TYPE "YES" OR "NO".

no

.....OPTSYSX IS NOW TERMINATED.....

F; T=0.54/2.84 15:36:49
ieccid off
END BECCFDING OF TERMINAL SESSION

I. EXAMPLE OF PROGRAM FAILURE

The following pathological example of program failure during regulator synthesis was taken from the Journal of Guidance and Control, Vol.3, No.2, pp.190-192, March-April 1980.

In this example, the choice of the quadratic index value was the factor promoting program instability; leading to eventual program failure in subroutine HQR2. The calculated regulator gains of -5.1, and -3.1 (pa. 63) are not correct!

With a 'slight' modification of the cost matrix from a previous value of 4.0000 to a new value of 4.0001, the program was run a second time. Failure did not occur on the second run, and the new calculations (pa. 68) indicate filter gains of -2.0, and a "small" residue of 3.19D-14 (essentially zero). These are the correct values.

This example points out one possible method of correcting certain program failure modes, should they occur during execution.

The full terminal session is recorded below, with user input in lower case letters following each "?".

Following the program failure example, that portion of the repeated terminal output was deleted up to the point where program execution of the second run begins.

reccrd on

BEGIN RECORDING OF TERMINAL SESSION

R; T=0.01,0.02 21:49:30

filedef C6 term (reclm fa blksize 133

global txtlib fortmod2 mod2eeh insldp ncnimsl

load cptsysx (start

EXECUTION BEGINS...

OPTSYSX IS A COMPLETELY INTERACTIVE OPTIMAL SYSTEMS CONTROL PROGRAM. IT WILL SOLVE NUMEROUS CONTROL PROBLEMS ON THE FOLLOWING TYPES OF SYSTEMS CONTROL EQUATIONS:

$$\dot{X} = (F) * X + (G) * U + (GAM) * (W + W0)$$

MEASUREMENT EQUATION--

$$Z = (H) * X + (D) * W + V$$

REGULATOR PERFORMANCE INDEX--

$$J = 1/2 * \int (Y * (A) * Y + U * (B) * U) dt$$

STATE FEEDBACK GAIN DEFINITION--

$$U = - (C) * X$$

DO YOU WISH TO CONTINUE? TYPE "YES" OR "NO".

yes

--DATA ENTRY--

ALTHOUGH OPTSYSX IS SPECIFICALLY DESIGNED TO READ ALL MATRIX DATA INTERACTIVELY, SEVERAL ALTERNATE METHODS ARE AVAILABLE TO USERS:

METHOD 1--THE "F", "G", AND "GAMMA" MATRICES MAY BE READ FROM SEPARATE DATA FILES.

METHOD 2--THE "F", "G", AND "GAMMA" MATRICES MAY BE EXPLICITLY DEFINED WITHIN SUBROUTINE "SETUP".

(NOTE: IN EITHER CASE, THE USER SHOULD OBTAIN A COPY OF THE PROGRAM LISTING AND EXAMINE THE EXAMPLES CONTAINED IN S/R "SETUP".)

DO YOU WISH TO CONTINUE? TYPE "YES" OR "NO".

yes

DO YOU WISH TO INPUT THE "F", "G", AND "GAMMA" MATRICES FROM SUBROUTINE "SETUP" IN THE METHOD DESCRIBED ON THE PREVIOUS SCREEN? TYPE "YES" OR "NO".

no

GENERAL OPTSYSX OPTIONS:

OPTION 1 -- SYSTEM ANALYSIS WITHOUT OPEN-LOOP EIGENSYSTEM CALCULATIONS.

OPTION 2 -- SYSTEM ANALYSIS WITH OPEN-LOOP EIGENSYSTEM CALCULATIONS.

OPTION 3 -- OPEN-LOOP EIGENSYSTEM FOUND AND PROGRAM TERMINATES.

("F"-MATRIX ENTRY FOLLOWS IMMEDIATELY.)

OPTION 4 -- MODAL DISTRIBUTION MATRICES COMPUTED
WITHOUT FILTER OR REGULATOR COMPUTED
OF STEADY-STATE ANALYSIS.

SELECT AN OPTION: 1, 2, 3, OR 4.

?

2

DO YOU DESIRE RES VALUES OF STATE AND CONTROL?

TYPE "YES" OR "NO".

yes

CFTSYSX LQR/CLASSICAL OPTIONS:

OPTION 1 -- OPTIMAL FILTER AND/OR REGULATOR
SYNTHESIS WITH NO EXTERNAL "C" OR "K"
MATRIX INPUT.

OPTION 2 -- OPTIMAL FILTER AND/OR REGULATOR
SYNTHESIS WITH EXTERNAL "C"
MATRIX INPUT.

OPTION 3 -- OPTIMAL FILTER AND/OR REGULATOR
SYNTHESIS WITH EXTERNAL "K"
MATRIX INPUT.

OPTION 4 -- OPTIMAL FILTER AND/OR REGULATOR
SYNTHESIS WITH EXTERNAL "C" AND "K"
MATRIX INPUT.

SELECT AN OPTION: 1, 2, 3, OR 4.

?

1

DO YOU WISH TO DETERMINE THE STEADY-STATE RESPONSE
FOR A CONSTANT DISTURBANCE?

TYPE "YES" OR "NO".

no

DO YOU WISH TO DETERMINE THE MODAL DISTRIBUTION
AND GAIN MATRICES?

TYPE "YES" OR "NO".

no

CFEN-LOCP TRANSFER FUNCTION OPTIONS:

OPTION 1 -- NO CFEN-LOCP TRANSFER FUNCTIONS COMPUTED.

OPTION 2 -- POLES, RESIDUES, AND ZEROS COMPUTED.

OPTION 3 -- ONLY POLES AND ZEROS COMPUTED.

OPTION 4 -- ONLY POLES AND RESIDUES COMPUTED.

SELECT AN OPTION: 1, 2, 3, OR 4.

?

2

NOISE TRANSFER FUNCTION OPTIONS:

OPTION 1 -- NO NOISE TRANSFER FUNCTIONS COMPUTED.

OPTION 2 -- POLES, RESIDUES, AND ZEROS COMPUTED.

OPTION 3 -- ONLY POLES AND ZEROS COMPUTED.

OPTION 4 -- ONLY POLES AND RESIDUES COMPUTED.

SELECT AN OPTION: 1, 2, 3, OR 4.

?

1

COMPENSATOR TRANSFER FUNCTION OPTIONS:

OPTION 1 -- NO COMP. TRANSFER FUNCTIONS COMPUTED.

OPTION 2 -- POLES, RESIDUES, AND ZEROS COMPUTED.

OPTION 3 -- ONLY POLES AND ZEROS COMPUTED.

OPTION 4 -- ONLY POLES AND RESIDUES COMPUTED.

(NOTE: A COMPENSATOR TRANSFER FUNCTION CAN BE
COMPUTED ONLY IF BOTH A REGULATOR
AND FILTER ARE SYNTHESIZED
AND/OR INPUT.)

SELECT AN OPTION: 1, 2, 3, OR 4.

?

1

WILL A FEED-FORWARD DISTRIBUTION MATRIX

("D" - MATRIX) BE INPUT ?

TYPE "YES" OR "NO".

DO

DO YOU DESIRE TO SYNTHESIZE A STABLE FILTER (OR REGULATOR)
BY DESTABILIZING THE ORIGINAL SYSTEM?

(NOTE: WORKS FOR FILTER OR REGULATOR BUT NOT FOR BOTH
IN THE SAME RUN.)

TYPE "YES" OR "NO".

DO

DO YOU DESIRE TO PRINT THE EULER-LAGRANGE EIGENSYSTEM
PRICR TO DECCMPOSITION (FOR CHECKING THE PROGRAM)?

TYPE "YES" OR "NO".

DO

POWER SPECTRAL DENSITY (PSD) OPTION 1 :

OPTION 1 -- COMPUTE THE PSD OF THE OUTPUTS AND/OR THE
CONTROLS OF THE CONTROLLED SYSTEM WHEN FORCED BY
PROCESS AND MEASUREMENT NOISE. (NOTE: BOTH A
REGULATOR AND A FILTER MUST BE RESIDENT IN THE
PROGRAM TO USE THIS OPTION.)

OPTION 2 -- SAME AS OPTION 1 ABOVE BUT ONLY PRINT THE
RESIDUES OF EACH TRANSFER FUNCTION
USED IN THE PSD COMPUTATION.

OPTION 3 -- NOT DESIRED.

SELECT AN OPTION: 1, 2, OR 3.

?

3

DO YOU DESIRE REGULATOR SYNTHESIS ONLY?

TYPE "YES" OR "NO".

yes

ENTER THE # OF STATES (NS) OF THE SYSTEM MATRIX
("F"-MATRIX).

?

2

ENTER THE # OF CONTROLS (NC) OF THE SYSTEM MODEL
("G"-MATRIX).

?

1

ENTER THE # OF MEASUREMENTS OR OBSERVATIONS (NO)
("H"-MATRIX).

?

2

ENTER THE # OF PROCESS NOISE SOURCES (NG)

("GAMMA"-MATRIX).

?

C

FLAG/PARAMETER SETTINGS FOR THIS RUN ARE AS FOLLOWS:

IOL	IC	IR	ISS	IM	ITF1	ITF2	ITF3	IFDFW	IE	ILLEBUG
1	1	C	0	0	1	0	0	0	0	0
ISBT	IDSTAE	IPSD	IYU	INCRM	IREG	NS	NC	NOB	NG	
0	C	0	C	0	1	2	1	2	0	

ORDER OF SYSTEM = 2

NUMBER OF CONTROLS = 1

NUMBER OF OBSERVATIONS = 2

NUMBER OF PROCESS NOISE SOURCES = 0

ENTER THE SYSTEM MATRIX ("F"-MATRIX)

DIMENSION = # STATES (NS) X # STATES (NS)

THE ELEMENT F(1, 1)=

?

C

THE ELEMENT F(1, 2)=

?

1

THE ELEMENT F(2, 1)=

?

-1

THE ELEMENT F(2, 2)=

?

0

THE SYSTEM MATRIX ("F"-MATRIX) ...

0.0	1.00000
-1.00000	0.0

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?

TYPE "YES" OR "NO".

NO

```

CFEN ICCP DYNAMICS MATRIX.....F..
0.0          0.1000E+01
-0.1000E+01  0.0

OPEN ICCP EIGENVALUES.....DET(SI-F)..
0.0          , 1.0000E+00:

CFEN ICCP RIGHT EIGENVECTCS MATRIX.....T....
0.0          -1.0000E+00
1.0000E+00  0.0

CFEN ICCP LEFT EIGENVECTOR MATRIX.....T-INV..
0.0          1.0000E+00
-1.0000E+00  0.0

```

```

ENTER THE MEASUREMENT SCALING MATRIX ("H"-MATRIX).
      DIMENSION = # OBSERVATIONS (NO) X # STATES (NS)
THE ELEMENT H( 1, 1)=

```

?

C

```

THE ELEMENT H( 1, 2)=

```

?

0

```

THE ELEMENT H( 2, 1)=

```

?

C

```

THE ELEMENT H( 2, 2)=

```

?

-1

```

      THE MEASUREMENT SCALING MATRIX ("H"-MATRIX)....

```

```

0.0          0.0
0.0          -1.00000

```

```

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?
TYPE "YES" OR "NO".

```

NO

MEASUREMENT SCALING MATRIX.....H..

0.0 0.0
0.0 -0.1000E+01

MEASUREMENT SCALING MATRIX...d(BAR)*T..

0.0 0.0
-1.000000E+00 0.0

ENTER THE OUTPUT MEASUREMENT COST MATRIX ("A"-MATRIX).

DIMENSION = # OBSERVATIONS (NO) X # OBSERVATIONS (NO)

THE ELEMENT A(1, 1)=

?
C

THE ELEMENT A(1, 2)=

?
0

THE ELEMENT A(2, 1)=

?
C

THE ELEMENT A(2, 2)=

?
4

THE OUTPUT MEASUREMENT COST MATRIX ("A"-MATRIX)...

0.0 0.0
0.0 4.00000

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?
TYPE "YES" OR "NO".

NO

OUTPUT COST MATRIX.....A..

0.0 0.0
0.0 0.4000E+01

ENTER THE CONTROL DISTRIBUTION MATRIX ("G"-MATRIX).

DIMENSION = # STATES (NS) X # CONTROLS (NC)

THE ELEMENT G(1, 1)=

?

C

THE ELEMENT G(2, 1)=

?

1

THE CONTROL DISTRIBUTION MATRIX ("G"-MATRIX)...

C.0

1.00000

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?

TYPE "YES" OR "NO".

NO

ENTER THE CONTROL COST WEIGHTING MATRIX ("B"-MATRIX)

DIMENSION = # CONTROLS (NC) X # CONTROLS (NC)

THE ELEMENT B(1, 1)=

?

1

THE CONTROL COST MATRIX.....B...

1.00000

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?

TYPE "YES" OR "NO".

NO

THE CONTROL DISTRIBUTION MATRIX.....G..

0.0

0.1000E+01

SCALE CONTROL DISTRIBUTION MATRIX.....TI*G..

1.00000E+00

0.0

THE CONTROL COST MATRIX.....B..

0.1000E+01

OPEN LOOP TRANSFER FUNCTIONS...

TF FOR INPUT NO. 1 AND OUTPUT NO. 1:

NO FINITE ZEROS. TF GAIN = 0.0

RESIDUES AT THE POLES:

P O L E S		R E S I D U E S	
REAL (A)	IMAG (B)		
(0.0)+J(1.000000)	(0.0)	EXP (A*T) *COS(B*T)	
(0.0)+J(-1.000000)	(0.0)	EXP (A*T) *SIN(B*T)	

IF FOR INPUT NO. 1 AND OUTPUT NO. 2:

CRIES OF NUMERATOR = 1 TF GAIN = -0.1000D+01

NUMERATOR EIGENVALUES (INCLUDING EXTRANEIOUS ZERO VALUES):

(0.0)+J(0.0)	(0.0)
(0.0)+J(0.0)	(0.0)

RESIDUES AT THE POLES:

P O L E S		R E S I D U E S	
REAL (A)	IMAG (B)		
(0.0)+J(1.000000)	(-1.000000)	EXP (A*T) *COS (E*T)	
(0.0)+J(-1.000000)	(0.0)	EXP (A*T) *SIN (E*T)	

FAILURE IN HQE2 ON EIGENVALUE NO. 4

-1.962366D+00	3.464812D-03	-2.499867D+00	1.508857D+00
3.464836D-03	3.762172D-02	-1.491143D+00	2.500102D+00
-4.415041E-15	-3.208843D-13	-1.962366D+00	3.621151D-03
5.281945D-11	-1.267812D-17	3.621125D-03	3.762121D-02

EIGENSYSTEM OF OPTIMAL REGULATOR.....

EULER-LAGRANGE EQUATIONS HAVE A REAL EIGENVALUE
AT OR NEAR ZERO.

C-LCCF OPTIMAL REG. E-VALUES...DET(SI-F+G*C) ..
0.0 : 0.0 , -1.00000D+00:

C-LCCF RIGHT EIGENVECTOR MATRIX.....M.....
-7.058867D-01 6.035185D-01
-7.083307E-01 1.000000D+00

CONTROL EIGENVECTOR MATRIX.....C*M..
-1.411761D+00 -1.504787D-02

C-LOOP CRT. REG. LEFT E-VECTOR MATRIX..M-INV..
-3.592082D+00 2.167888D+00
-2.544382D+00 2.535582D+00

THE OPTIMAL FEEDBACK GAIN CONTROL MATRIX...C=BINV*GT*S...
5.10951+00 -3.0987D+00

THE MODAL CONTROL GAINS.....C*T..
-3.098696D+00 -5.109451D+00

THE CLOSED LOOP DYNAMICS MATRIXF-G*C..
0.0 1.000000D+00
4.109451D+00 -3.098696D+00

ANALYSIS COMPLETE. DO YOU WANT ANOTHER RUN?
TYPE "YES" OR "NO".

yes

DO YOU WISH TO SAVE THE "F"-MATRIX FROM THE LAST
RUN TO BE USED IN THE FOLLOWING RUN?
NOTE: THE MATRIX WILL BE REDISPLAYED AT
THE NEXT INPUT SEQUENCE INTERVAL
AND YOU WILL HAVE THE OPTION OF CHANGING
INDIVIDUAL MATRIX ELEMENTS.
TYPE "YES" OR "NO".

yes

DO YOU WISH TO SAVE THE "H"-MATRIX FROM THE LAST
RUN TO BE USED IN THE FOLLOWING RUN?
NOTE: THE MATRIX WILL BE REDISPLAYED AT
THE NEXT INPUT SEQUENCE INTERVAL
AND YOU WILL HAVE THE OPTION OF CHANGING
INDIVIDUAL MATRIX ELEMENTS.
TYPE "YES" OR "NO".

yes

DO YOU WISH TO SAVE THE "G"-MATRIX FROM THE LAST

RUN TO BE USED IN THE FOLLOWING RUN?
NOTE: THE MATRIX WILL BE REDISPLAYED AT
THE PROPER INPUT SEQUENCE INTERVAL
AND YOU WILL HAVE THE OPTION OF CHANGING
INDIVIDUAL MATRIX ELEMENTS.

TYPE "YES" OR "NO".

yes

Author's note: Since the same program options are to
be run again, with only a change in one
of the cost matrix element values, the
terminal output was deleted up to the
point where program calculations resume
in order to avoid redundancy.

ORDER OF SYSTEM = 2
NUMBER OF CONTROLS = 1
NUMBER OF OBSERVATIONS = 2
NUMBER OF PROCESS NOISE SOURCES = 0

THE SYSTEM MATRIX ("F"-MATRIX) ...

0.0	1.00000
-1.00000	0.0

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?

TYPE "YES" OR "NO".

no

OPEN LOOP DYNAMICS MATRIX.....F..

0.0	0.1000E+01
-0.1000E+01	0.0

OPEN LOOP EIGENVALUES.....DET(SI-F)...

0.0 , 1.0000E+00:

OPEN LOOP RIGHT EIGENVECTOR MATRIX.....T....

0.0	-1.0000E+00
1.0000E+00	0.0

OPEN LOOP LEFT EIGENVECTOR MATRIX.....T-INV..

0.0 1.000000D+00
-1.000000D+00 0.0

THE MEASUREMENT SCALING MATRIX ("H"-MATRIX)...

0.0 0.0
0.0 -1.000000

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?
TYPE "YES" OR "NO".

NO

MEASUREMENT SCALING MATRIX.....H..

0.0 0.0
0.0 -0.1000E+01

SPECIAL MEASUREMENT SCALING MATRIX...H(BAR) *T..

0.0 0.0
-1.000000D+00 0.0

ENTER THE OUTPUT MEASUREMENT COST MATRIX ("A"-MATRIX).

DIMENSION = # OBSERVATIONS (NO) X # OBSERVATIONS (NO)

THE ELEMENT A(1, 1) =

?

C

THE ELEMENT A(1, 2) =

?

0

THE ELEMENT A(2, 1) =

?

C

THE ELEMENT A(2, 2) =

?

4.0001

THE OUTPUT MEASUREMENT COST MATRIX ("A"-MATRIX)...

0.0 0.0
0.0 4.00010

DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?

TYPE "YES" OR "NO".

IC

OUTPUT COST MATRIX.....A..

0.0 0.0
0.0 0.4000E+01

THE CONTROL DISTRIBUTION MATRIX ("G"-MATRIX)...

0.0
1.00000

IC YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?

TYPE "YES" OR "NO".

NO

ENTER THE CONTROL COST WEIGHTING MATRIX ("B"-MATRIX)

DIMENSION = # CONTROLS (NC) X # CONTROLS (NC)

THE ELEMENT B(1, 1) =

?

1

THE CONTROL COST MATRIX.....B...

1.00000

IC YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEMENT?

TYPE "YES" OR "NO".

NO

THE CONTROL DISTRIBUTION MATRIX.....G..

0.0
0.1000E+01

SCALAR CONTROL DISTRIBUTION MATRIX.....TI*G..

1.00000E+00
0.0

THE CONTROL COST MATRIX.....B..

0.1000E+01

OPEN LOOP TRANSFER FUNCTIONS...

TF FOR INPUT NO. 1 AND OUTPUT NO. 1:

NC FINITE ZEROS. TF GAIN = 0.0

RESIDUES AT THE POLES:

P O L E S		R E S I D U E S	
REAL(A)	IMAG(B)		
(0.0)	+J(1.000000)	(0.0)	EXP(A*T)*COS(B*T)
(0.0)	+J(-1.000000)	(0.0)	EXP(A*T)*SIN(B*T)

TF FCF INPUT NO. 1 AND OUTPUT NO. 2:

ORDER OF NUMERATOR = 1 TF GAIN = -0.1000D+01

NUMERATOR EIGENVALUES (INCLUDING EXTRANEOUS ZERO VALUES):

(0.0)	+J(0.0)
(0.0)	+J(0.0)

RESIDUES AT THE POLES:

P O L E S		R E S I D U E S	
REAL(A)	IMAG(E)		
(0.0)	+J(1.000000)	(-1.000000)	EXP(A*T)*COS(B*T)
(0.0)	+J(-1.000000)	(0.0)	EXP(A*T)*SIN(B*T)

EIGENSYSTEM OF OPTIMAL REGULATOR.....

C-LCCE OPTIMAL REG. E-VALUES...DET(SI-F+G*C)..
-1.00501E+00:-9.95012E-01:

C-LCCE RIGHT EIGENVECTOR MATRIX.....M....
7.053368E-01 -7.088723D-01
-7.088723D-01 7.053368D-01

CONTROL EIGENVECTOR MATRIX.....C*M..
1.417762D+00 -1.410691D+00

C-LCCE OPT. REG. LEFT E-VECTOR MATRIX..M-INV..
-1.410691D+02 -1.417762D+02
-1.417762D+02 -1.410691D+02

THE OPTIMAL FEEDBACK GAIN CONTROL MATRIX...C=BI⁻¹GT⁻¹S...
-3.1974E-14 -2.0000E+00

THE MODAL CONTROL GAINS.....C*E...
-2.000025D+00 3.197442D-14

THE CLOSED LOOP DYNAMICS MATRIXF-G*C..
0.0 1.000000D+00
-1.000000E+00 -2.000025D+00

ANALYSIS COMPLETE. DO YOU WANT ANOTHER RUN?
TYPE "YES" OR "NO".

NO

.....CPTSYSX IS NOW TERMINATED.....

R: T=C.63/2.60 23:33:07

record off

END RECORDING OF TERMINAL SESSION

V. CONCLUSIONS AND RECOMMENDATIONS

A. CONCLUSIONS

Although originally developed for the quadratic synthesis of controllers for rotary-wing VTOL aircraft, the extensive modifications and enhancements of Hall's original work, coupled with its efficient and accurate eigensystem solution routine, represent a powerful tool in the design of optimally controlled systems.

In its present interactive form, OPTSYSX has been transferred from the arena of high-level applied mathematics and numerical analysis to the level of control system engineers and students. It now represents an even more powerful educational tool, able to rapidly and effectively unlock many misunderstood linear systems mathematical relationships.

As an ultimate evaluation of the computational abilities of OPTSYSX, the program was tested using an 82 X 82 matrix of aircraft longitudinal motion equations for the VX-29 experimental Fighter aircraft derivative, provided by NASA-Edwards.

For a system of equations of this magnitude, all program arrays were re-dimensioned (as shown in Appendix A), and a 2-Megabyte virtual machine size was required. This system was run through the Modal Analysis option of OPTSYSX, requiring less than 90 seconds to load the system and complete all open-loop and modal analysis calculations!

Program results exhibited perfect eigenvalue correlation with those obtained from the John Edwards Control Program. Additionally, OPTSYSX provided complete longitudinal modal analysis, previously unavailable on a system of this size.

It is hoped that the use of this interactive program version will be encouraged; and that its expanded abilities will stimulate both interest and research on basic systems control problems, as well as more advanced designs.

E. RECOMMENDATIONS

Based on the results of this thesis, four areas emerged as possibilities for further research and study:

1. Program Availability

The use of OPTSYSX and similar design programs should be encouraged in all undergraduate and graduate level courses involved in the analysis and design of control systems. Toward this end, it is recommended that OPTSYSX be placed in the non-IMSL library of subroutines, making it easily available to all potential users.

2. Computer Graphics

The addition of graphical plotting routines to the program in the time and frequency domain would make OPTSYSX an even more powerful tool in the design of many optimally controlled systems.

3. Further Modifications

The present version of the program should be modified to include the OPTSYS 5 derivative input term improvements of Liu [Ref. 3], and program sequencing during optimal filter synthesis should be examined. Various test runs indicate an area of conflict in that the program appears to require the design of an optimal regulator prior to performing any filter calculations.

4. Program Application

CFTSYSX offers attractive possibilities in the area of microcomputer implementation.


```

IDBUG=0
ISET=0
IPED=0
IYU=0
INQAM=0
IREG=0
NS=0
NC=0
NCH=0
NG=0
-----SCRN1-----
20 CALL PRTCHS ('CLRSCEN ')
   WRITE (5,890)
   CALL RDCHAR (IANS)
   IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GO TO 30
   GO TO 40
30 WRITE (5,880)
   GO TO 20
40 CONTINUE
   IF (IANS.EQ.IZ) GO TO 560
-----SCRN2-----
50 CALL PRTCHS ('CLRSCEN ')
   WRITE (5,900)
   CALL RDCHAR (IANS)
   IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GO TO 60
   GO TO 70
60 WRITE (5,890)
   GO TO 50
70 CONTINUE
   IF (IANS.EQ.IZ) GO TO 560
-----ISET-----
80 CALL PRTCHS ('CLRSCEN ')
   WRITE (5,910)
   CALL RDCHAR (IANS)
   IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GO TO 90
   GO TO 100
90 WRITE (5,880)
   GO TO 80
100 CONTINUE
   IF (IANS.EQ.IY) ISET=1
-----IOL-----
   CALL PRTCHS ('CLRSCEN ')
   WRITE (5,570)
   CALL RDINT (IANS)
   ICL=IANS-1
   IF (IOL.EQ.2) GO TO 350
-----IQ-----
110 CALL PRTCHS ('CLRSCEN ')
   WRITE (5,580)
   CALL RDCHAR (IANS)
   IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GO TO 120
   GO TO 130
120 WRITE (5,880)
   GO TO 110
130 CONTINUE
   IF (IANS.EQ.IY) IC=1
   IF (IANS.EQ.IZ) IC=0
   IF (IOL.EQ.3) GO TO 200
-----IR-----
   CALL PRTCHS ('CLRSCEN ')
   WRITE (5,590)
   CALL RDINT (IANS)
   IR=IANS-1
-----ISS-----
140 CALL PRTCHS ('CLRSCEN ')
   WRITE (5,600)
   CALL RDCHAR (IANS)
   IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GO TO 150
   GO TO 160
150 WRITE (5,880)
   GO TO 140
160 CONTINUE
   IF (IANS.EQ.IY) ISS=1
   IF (IANS.EQ.IZ) ISS=0
-----IM-----
170 WRITE (5,610)

```

```

CALL RDCHAR (IANS)
IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GO TO 180
GO TO 190
180 WRITE (5,880)
GO TO 170
190 CONTINUE
IF (IANS.EQ.IY) IM=1
IF (IANS.EQ.IZ) IE=0
200 CONTINUE
IF (IOL.EQ.3) IM=1
C-----ITF1-----
CALL PRTCHS ('CLRSCFN ')
WRITE (5,620)
CALL RDINT (IANS)
ITF1=IANS-1
IF (IOL.EQ.3) GO TO 240
C-----ITF2-----
CALL PRTCHS ('CLRSCFN ')
WRITE (5,630)
CALL RDINT (IANS)
ITF2=IANS-1
IF (IOL.EQ.3) GO TO 240
C-----ITF3-----
CALL PRTCHS ('CLRSCFN ')
WRITE (5,640)
CALL RDINT (IANS)
ITF3=IANS-1
C-----IPDFW-----
210 CALL PRTCHS ('CLRSCFN ')
WRITE (5,650)
CALL RDCHAR (IANS)
IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GC TO 220
GO TO 230
220 WRITE (5,880)
GO TO 210
230 CONTINUE
IF (IANS.EQ.IY) IPDFW=1
IF (IANS.EQ.IZ) IPDFW=0
C-----IE-----
CALL PRTCHS ('CLRSCFN ')
WRITE (5,660)
CALL RDREAL (ANSR)
IE=IDINT (ANSR)
IF (IOL.EQ.3) GO TO 300
C-----IDSTAB-----
240 CALL PRTCHS ('CLRSCFN ')
WRITE (5,670)
CALL RDCHAR (IANS)
IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GC TO 250
GO TO 260
250 WRITE (5,880)
GO TO 240
260 CONTINUE
IF (IANS.EQ.IY) IDSTAB=1
IF (IANS.EQ.IZ) IDSTAB=0
C-----IDEBUG-----
270 WRITE (5,680)
CALL RDCHAR (IANS)
IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GO TO 280
GO TO 290
280 WRITE (5,880)
GO TO 270
290 CONTINUE
IF (IANS.EQ.IY) IDEBUG=1
IF (IANS.EQ.IZ) IDEBUG=0
300 CONTINUE
C-----IPSD-----
CALL PRTCHS ('CLRSCFN ')
WRITE (5,690)
CALL RDINT (IANS)
IPSD=IANS
IF (IPSD.EQ.3) IPSD=C
IF (IPSD.EQ.0) GO TO 310
C-----IYU-----
CALL PRTCHS ('CLRSCFN ')
WRITE (5,700)

```

```

CALL RDINT (IANS)
IYU=IANS-1
C-----INORM-----
CALL PRTCHS ('CLRSCEN ')
WRITE (5,820)
CALL RDRÉAL (ANSR)
INCRN=IDINT(ANSR)
310 IF (IOL.EQ.1) GC TO 350
C-----IREG-----
CALL PRTCHS ('CLRSCEN ')
320 WRITE (5,710)
CALL RDCBAR (IANS)
IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GC TO 330
GO TO 340
330 WRITE (5,880)
GO TO 320
340 CONTINUE
IF (IANS.EQ.IY) IREG=1
IF (IANS.EQ.IZ) IREG=0
C-----NS-----
350 CALL PRTCHS ('CLRSCEN ')
WRITE (5,720)
CALL RDRÉAL (ANSR)
NS=IDINT(ANSR)
IF (IOL.EQ.2) GO TO 360
C-----NC-----
WRITE (5,730)
CALL RDRÉAL (ANSR)
NC=IDINT(ANSR)
C-----NOB-----
WRITE (5,740)
CALL RDRÉAL (ANSR)
NOB=IDINT(ANSR)
C-----NG-----
WRITE (5,750)
CALL RDRÉAL (ANSR)
NG=IDINT(ANSR)
360 CCNTINUE
C-----PLAG SETTINGS-----
CALL PRTCHS ('CLRSCEN ')
WRITE (6,760)
WRITE (6,770)
WRITE (6,780) IOL,IQ,IR,ISS,IE,ITF1,ITF2,ITF3,IPDPW,IE,IDEBUG,ISET
1,IESTAB
WRITE (6,790)
WRITE (6,800) IPSE,IYU,INCRN,IREG,NS,NC,NOB,NG
WRITE (6,810) NS,NC,NOB,NG
C-----BEGIN CALCULATIONS-----
N2=2*NS
CALL IINDEX (NS,NC,NOB,NG,N2,ACL,B,BA,CI,CR,CO,CWI,CWB,D,FBGC,FBGE,
1G,GAN,GM,GN,HC,D1,D2,PRO,SH,RC,C,SC,WR,HI,#11,#21,X,ENCRN,#NOFMI,D
2,ESTAB,AA,BH,CM,JCF,SES,AY,38,CC,CP,GW,GV,RY,HU,DSTORE,ISAF,ISAH,IS
3AG,IGAN,INET,FRIT,NRCW,NCCL)
C-----IRET-----
370 WRITE (5,830)
CALL RDCBAR (IANS)
IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GC TO 380
GO TO 390
380 WRITE (5,880)
GO TO 370
390 CONTINUE
IF (IANS.EQ.IY) GC TO 400
IF (IANS.EQ.IZ) GC TO 560
C-----ISAF-----
400 CONTINUE
IF (INET.EQ.1) GO TO 10
IF (ISET.EQ.1) GO TO 10
CALL PRTCHS ('CLRSCEN ')
410 WRITE (5,840)
CALL RDCBAR (IANS)
IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GO TO 420
GO TO 430
420 WRITE (5,880)
GO TO 410
430 CONTINUE
IF (IANS.EQ.IY) ISAF=1

```

```

IF (IANS.EQ.IZ) ISAF=0
C-----ISAH-----
IF (NOB.EC.0) GC TC 470
CALL PRTCHS ('CLRSCN')
440 WRITE (5,850)
CALL RDCBAR (IANS)
IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GC TC 450
450 GO TO 460
WRITE (5,880)
GO TO 440
460 CONTINUE
IF (IANS.EQ.IY) ISAH=1
IF (IANS.EQ.IZ) ISAH=0
470 CONTINUE
C-----ISAG-----
IF (NG.EC.0) GC TC 510
CALL PRTCHS ('CLRSCN')
480 WRITE (5,860)
CALL RDCBAR (IANS)
IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GC TC 490
490 GO TO 500
WRITE (5,880)
GO TO 480
500 CONTINUE
IF (IANS.EQ.IY) ISAG=1
IF (IANS.EQ.IZ) ISAG=0
510 CONTINUE
C-----IGAN-----
IF (NG.EC.0) GC TC 550
CALL PRTCHS ('CLRSCN')
520 WRITE (5,870)
CALL RDCBAR (IANS)
IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GO TC 530
530 GO TO 540
WRITE (5,880)
GO TO 520
540 CONTINUE
IF (IANS.EQ.IY) IGAN=1
IF (IANS.EQ.IZ) IGAN=0
550 CONTINUE
GC TO 10
C-----TERMINATE-----
560 WRITE (5,920)
STOP
C-----
570 FORMAT (25X,24HGENERAL OPTSYSX OPTIONS:,,,10X,35HOPTION 1 -- SYST
1EN ANALYSIS WITHOUT,,,22X,35HOPEN-LOOP EIGENSYSTEM CALCULATIONS.//
2//,10X,42HOPTION 2 -- SYSTEM ANALYSIS WITH OPEN-LOOP //,22X,25HEIGEN
3SYSTEM CALCULATIONS.//,,10X,39HOPTION 3 -- OPEN-LOOP EIGENSYSTEM P
4OUND,,,22X,23HAND PROGRAM TERMINATES.//,22X,39H "P" MATRIX ENTRY P
5OLLOWS IMMEDIATELY.//,,10X,48HOPTION 4 -- MODAL DISTRIBUTION MATR
6ICES COMPUTED.//,22X,37HWITHOUT FILTER OR REGULATOR SYNTHESIS.//,22X
7,25HOR STEADY-STATE ANALYSIS.//,,15X,30HSELECT AN OPTION: 1,2,3, 0
8R 4.)
580 FORMAT (//,5X,46HDO YOU DESIRE RMS VALUES OF STATE AND CONTROL?//
1,10X,19HTYPE "YES" OR "NO".)
590 FORMAT (//,20X,30HCPTSYSX LQR/CLASSICAL OPTIONS:,,,10X,43HOPTION 1
1 -- OPTIMAL FILTER AND/OR REGULATOR.//,22X,37HSYNTHESIS WITH NC EXT
2ERNAL "C" OR "K".//,22X,13HMATRIX INPUT.//,,10X,43HOPTION 2 -- OPTI
3MAL FILTER AND/OR REGULATOR.//,22X,27HSYNTHESIS WITH EXTERNAL "C"/
4,22X,13HMATRIX INPUT.//,,10X,43HOPTION 3 -- OPTIMAL FILTER AND/OR
5REGULATOR.//,22X,27HSYNTHESIS WITH EXTERNAL "K".//,22X,13HMATRIX INP
6UT.//,,10X,43HOPTION 4 -- OPTIMAL FILTER AND/OR REGULATOR.//,22X,35
7HSYNTHESIS WITH EXTERNAL "C" AND "K".//,22X,13HMATRIX INPUT.//,,10X
8,32HSELECT AN OPTION: 1, 2, 3 OR 4.)
600 FORMAT (//,5X,5CHDO YOU WISH TO DETERMINE THE STEADY-STATE RESPON
1S.//,,9X,27HOR A CONSTANT DISTURBANCE?//,,10X,19HTYPE "YES" OR "NO"
2.)
610 FORMAT (5X,47HDO YOU WISH TO DETERMINE THE MODAL DISTRIBUTION.//,8X
1,18HAND GAIN MATRICES?//,10X,19HTYPE "YES" OR "NO".)
620 FORMAT (//,5X,36HOPEN-LOOP TRANSFER FUNCTION OPTIONS:,,,10X,53HCP
1TION 1 -- NO OPEN-LOOP TRANSFER FUNCTIONS COMPUTED.//,,10X,48HOPTI
2ION 2 -- POLES, RESIDUES AND ZEROS COMPUTED.//,,10X,42HOPTION 3 --
3ONLY POLES AND ZEROS COMPUTED.//,,10X,45HOPTION 4 -- ONLY POLES A
4ND RESIDUES COMPUTED.//,,10X,32HSELECT AN OPTION: 1, 2, 3 OR 4.)
630 FORMAT (//,5X,32HWISE TRANSFER FUNCTION OPTIONS:,,,10X,49HOPTICN

```

```

1 1 -- NO NOISE TRANSFER FUNCTIONS COMPUTED. //, 10X, 48HOPTION 2 --
2 PCLES, RESIDUES, AND ZEROS COMPUTED. //, 10X, 42HOPTION 3 -- ONLY PO
3 LES AND ZEROS COMPUTED. //, 10X, 45HOPTION 4 -- ONLY POLES AND RESID
4 UES COMPUTED. //, 10X, 32HSELECT AN OPTION: 1 -- 2, 3, OR 4.)
640 FORMAT //, 5X, 38HCOMPENSATOR TRANSFER FUNCTION OPTIONS: //, 10X, 49H
1 OPTION 1 -- NO COMP. TRANSFER FUNCTIONS COMPUTED. //, 10X, 38HOPTION
2 2 -- POLES, RESIDUES, AND ZEROS COMPUTED. //, 10X, 42HOPTION 3 -- O
3 NLY POLES AND ZEROS COMPUTED. //, 10X, 45HOPTION 4 -- ONLY POLES AND
4 RESIDUES COMPUTED. //, 15X, 45H NOTE: A COMPENSATOR TRANSFER FUNCTI
5 ON CAN BE //, 22X, 33HCOMPUTED ONLY IF BOTH A REGULATOR //, 22X, 26HANC
6 FILTER ARE SYNTHESIZED. //, 22X, 14HAND/CR INPUT. //, 10X, 32HSELECT AN
7 OPTION: 1, 2, 3, OR 4.)
650 FORMAT //, 5X, 39HWILL A FEED-FORWARD DISTRIBUTION MATRIX //, 5X, 25H
1 "C" - MATRIX BE INPUT? //, 15X, 19HTYPE "YES" OR "NO".)
660 FORMAT //, 5X, 63H THIS OPTION DETERMINES THE CRITERIA FOR DECIDING
1 WHEN A MARKOV //, 8X, 58HPARAMETER IS ZERO-THE MARKOV PARAMETER INDIC
2 ATES THE ORDER //, 8X, 54HOF THE NUMERATOR POLYNOMIAL OF EACH TRANSFE
3 R FUNCTION. //, 8X, 52HALL "N" ZEROS OF THIS POLYNOMIAL ARE PRINTED
4 OUT AND //, 8X, 52HTHIS TEST TELLS HOW MANY EXTRA ROOTS EXIST AT z =
5 0. //, 3X, 4HLESS THAN 10.0** -IF IS CONSIDERED ZERO. //, 8X, 47H THE
6 DEFAULT VALUE OF THIS PARAMETER IS 6 //, 8X, 28HIN OTHER WORDS
7, IT = 1. CR-6. //, 10X, 66HIF YOU DESIRE A DIFFERENT MARKOV CRITERIA
8, TYPE THE INTEGER VALUE. //, 10X, 48HIF YOU DESIRE THE DEFAULT VALU
9E, TYPE "0" ZERO.)
670 FORMAT //, 5X, 61HDO YOU DESIRE TO SYNTHESIZE A STABLE FILTER OR A
1 REGULATOR //, 8X, 8Y, 34HDESTABILIZING THE ORIGINAL SYSTEM? //, 12X, 52H
2 NOTE: WORKS FOR FILTER OR REGULATOR BUT NOT FOR BOTH. //, 20X, 17HIN T
3 HE SAME RUN. //, 10X, 19HTYPE "YES" OR "NO".)
680 FORMAT //, 5X, 53HDO YOU DESIRE TO PRINT THE EULER-LAGRANGE EIGENSYSTEM
1 //, 9X, 50HPRICE TO DECOMPOSITION FOR CHECKING THE PROGRAM? //, 10
2 X, 19HTYPE "YES" OR "NO".)
690 FORMAT //, 5X, 39HPOWER SPECTRAL DENSITY PSD OPTION 1 : //, 10X, 53
1 HOPTION 1 -- COMPUTE THE PSD OF THE CUTENTS AND/OR THE //, 22X, 48HCO
2 NTROLS OF THE CONTROLLED SYSTEM WHEN FORCED BY //, 22X, 45HPROCESS AN
3 D MEASUREMENT NOISE. NOTE: EITHER A //, 22X, 49HREGULATOR AND A FILTE
4 R MUST BE RESIDENT IN THE //, 22X, 38HPROGRAM TO USE THIS OPTION. //
5 10X, 53HOPTION 2 -- SAME AS OPTION 1 ABOVE BUT ONLY PRINT THE //, 22
6 X, 34HRESIDUES OF EACH TRANSFER FUNCTION. //, 22X, 28HUSED IN THE PSD C
7 OMPUTATION. //, 10X, 24HOPTION 3 -- NOT DESIRED. //, 10X, 29HSELECT A
8 N OPTION: 1, 2, OR 3.)
700 FORMAT //, 5X, 39HPOWER SPECTRAL DENSITY PSD OPTION 2 : //, 10X, 35
1 HOPTION 1 -- PSD OUTPUT NOT DESIRED. //, 10X, 38HOPTION 2 -- COMPUTE
2 ONLY OUTPUT PSD'S. //, 10X, 39HOPTION 3 -- COMPUTE ONLY CONTROL PSD
3 S. //, 10X, 50HOPTION 4 -- COMPUTE BOTH OUTPUT AND CONTROL PSD'S. //
4 //, 15X, 32HSELECT AN OPTION: 1, 2, 3, OR 4.)
710 FORMAT //, 5X, 39HDO YOU DESIRE REGULATOR SYNTHESIS ONLY? //, 10X, 19
1 HTYPE "YES" OR "NO".)
720 FORMAT //, 5X, 47HENTER THE # OF STATES NS OF THE SYSTEM MATRIX. //
15X, 13H "N" - MATRIX.)
730 FORMAT //, 5X, 56HENTER THE # OF CONTROLS NC OF THE CONTROL SYSTEM
1 MODEL //, 5X, 13H "N" - MATRIX.)
740 FORMAT //, 5X, 54HENTER THE # OF MEASUREMENTS OR OBSERVATIONS NO O
1 FTHE //, 5X, 13H "N" - MATRIX.)
750 FORMAT //, 5X, 48HENTER THE # OF PROCESS NOISE SOURCES NG OF THE //
1 5X, 17H "N" - MATRIX.)
760 FORMAT (5X, 52HFLAG/PARAMETER SETTINGS FOR THIS RUN ARE AS FOLLOWS:
1 //)
770 FORMAT (1X, 3HIOL, 2X, 2HIQ, 2X, 2HIR, 2X, 3HISS, 2X, 2HIN, 2X, 4HITF1, 2X, 4HI
1TF2, 2X, 4HITF3, 2X, 5HIFDP, 2X, 2HIE, 2X, 6HIDEBUG, 2X, 4HISET, 2X, 6HICSTAB
2 //)
780 FORMAT (1X, 12, 3X, 12, 3X, 12, 2X, 12, 3X, 12, 3X, 12, 4X, 12, 4X, 12, 4X, 1
12, 3X, 12, 6X, 12, 5X, 12 //)
790 FORMAT (1X, 4HIFSD, 2X, 3HIYU, 2X, 5HINORM, 2X, 4HIREG, 2X, 2HNS, 2X, 2HNC, 2X
1 3HNQB, 2X, 2HNG //)
800 FORMAT (2X, 12, 3X, 12, 4X, 12, 5X, 12, 3X, 12, 2X, 12, 3X, 12, 2X, 12, //)
810 FORMAT (2X, 17HCRDR OF SYSTEM =, I3, //, 2X, 20HNUMBER OF CONTROLS =, I
13, //, 2X, 24HNUMBER OF OBSERVATIONS =, I3, //, 2X, 33HNUMBER OF PROCESS
2 NOISE SOURCES =, I3, //)
820 FORMAT (5X, 53HDETERMINE THE NORMALIZATION PARAMETER INORM FOR TH
1 E //, 5X, 55HPOWER SPECTRAL DENSITY PSD OPTION YOU HAVE PREVIOUSLY,
2 //, 5X, 52HCHOSEN. TWO PSD NORMALIZATION METHODS ARE AVAILABLE: //, 10
3 X, 54HMETHOD 1 -- PSD IS NORMALIZED BY THE I-NORM OF THE PROCESS //, 21X
4 29HNOTSE MINUS "0" INORM INORM //, 21X, 49HNOTE: "0" IS AN OPTIMAL
5 STATE WEIGHTING MATRIX. //, 21X, 14HIN THIS METHOD, INORM = 0.1, //
6 NG. //, 10X, 63HMETHOD 2 -- PSD IS NORMALIZED BY THE INORM - NG, //, 21X
7 MEASUREMENT //, 21X, 39HNOISE MINUS "R" INORM - NG, INORM - NG //, 21X

```



```

8,51H NOTE: "R" IS AN OPTIMAL CONTROL WEIGHTING MATRIX. //,21X,44HI
9N THIS METHOD, INCRN = NG + 1,....,NG + NOB. //,10X,51HSELECT AN IN
STEGER PRCE 0 - 16 REPRESENTING YOUR ESD. //,15X,27HNORMALIZATION REQ
UIREMENTS. //,10X,53HIF PSD NORMALIZATION IS NOT DESIRED ENTER "C"
$ ZERO - 1
830 FORMAT (5X,43H ANALYSIS COMPLETE. DO YOU WANT ANOTHER RUN? //,15X,19
1HTYPE "YES" OR "NO".)
840 FORMAT (///,5X,48HDC YOU WISH TO SAVE THE "F"-MATRIX FROM THE LAST
1 //,5X,36H RUN TO BE USED IN THE FOLLOWING RUN? //,5X,39HNOTE: THE M
2ATRIX WILL BE REDISPLAYED AT //,5X,34H THE PROPER INPUT SEQUENCE INT
3EVAL. //,5X,40H AND YOU WILL HAVE THE OPTION OF CHANGING //,5X,27H IND
4IVIDUAL MATRIX ELEMENTS. //,15X,19HTYPE "YES" OR "NO".)
850 FORMAT (///,5X,48HDC YOU WISH TO SAVE THE "G"-MATRIX FROM THE LAST
1 //,5X,36H RUN TO BE USED IN THE FOLLOWING RUN? //,5X,39HNOTE: THE M
2ATRIX WILL BE REDISPLAYED AT //,5X,34H THE PROPER INPUT SEQUENCE INT
3EVAL. //,5X,40H AND YOU WILL HAVE THE OPTION OF CHANGING //,5X,27H IND
4IVIDUAL MATRIX ELEMENTS. //,15X,19HTYPE "YES" OR "NO".)
860 FORMAT (///,5X,48HDC YOU WISH TO SAVE THE "NG"-MATRIX FROM THE LAST
1 //,5X,36H RUN TO BE USED IN THE FOLLOWING RUN? //,5X,39HNOTE: THE M
2ATRIX WILL BE REDISPLAYED AT //,5X,34H THE PROPER INPUT SEQUENCE INT
3EVAL. //,5X,40H AND YOU WILL HAVE THE OPTION OF CHANGING //,5X,27H IND
4IVIDUAL MATRIX ELEMENTS. //,15X,19HTYPE "YES" OR "NO".)
870 FORMAT (///,5X,52HDC YOU WISH TO SAVE THE "GAMMA"-MATRIX FROM THE
1 LAST //,5X,36H RUN TO BE USED IN THE FOLLOWING RUN? //,5X,39HNOTE: T
2HE MATRIX WILL BE REDISPLAYED AT //,5X,34H THE PROPER INPUT SEQUENCE
3 INTERVAL. //,5X,40H AND YOU WILL HAVE THE OPTION OF CHANGING //,5X,27
4H INDIVIDUAL MATRIX ELEMENTS. //,15X,19HTYPE "YES" OR "NO".)
880 FORMAT (1X,51HWARNING: IMPROPER DATA ENTRY! ENTER "YES" OR "NO".)
890 FORMAT (5X,59HOPTSYSX IS A COMPLETELY INTERACTIVE OPTIMAL SYSTEMS
1CONTROL. //,8X,55HPROGRAM. IT WILL SOLVE NUMEROUS CONTROL PROBLEMS O
2N THE //,8X,45HCLICKING TYPES OF SYSTEMS CONTROL EQUATIONS. //,15X
3,35HXDOT = P * X + G * U + GAM * (U + UO) //,10X,22HMEASUREMENT //,50BA
4TION -- //,15X,21HZ = H * X + E * U + V //,20X,29HREGULATOR PERFORM
5ANCE INDEX -- //,15X,42HJ = 1/2 * INTEGRAL (Y * A * Y + U * B * U)DT.
6 //,20X,32HSTATE FEEDBACK GAIN DEFINITION -- //,25X,10HU = - C * X //
7 //,15X,45HDC YOU WISH TO CONTINUE? TYPE "YES" OR "NO".)
900 FORMAT (25X,14H--DATA ENTRY-- //,5X,49HALTHOUGH OPTSYSX IS SPECIFI
1CALLY DESIGNED TO READ //,5X,48HALL MATRIX DATA INTERACTIVELY, SEVE
2RAL ALTERNATE //,5X,31HMETHODS ARE AVAILABLE TO USERS. //,10X,43HME
3THOD 1--THE "F", "G", AND "GAMMA" MATRICES //,13X,17H MAY BE READ FRO
4M SEPARATE DATA FILES. //,10X,50HMETHOD 2--THE "F", "G", AND "GAMMA
5" MATRICES MAY BE //,13X,45HEXPPLICITLY DEFINED WITHIN SUBROUTINE "S
6ETUP". //,10X,52H NOTE: IN EITHER CASE, THE USER SHOULD OBTAIN A C
7OPY //,17X,34HCF THE PROGRAM LISTING AND EXAMINE //,17X,39H THE EXAMP
8LES CONTAINED IN S/R "SETUP". //,10X,45HDC YOU WISH TO CONTINUE?
9 TYPE "YES" OR "NO".)
910 FORMAT (///,5X,46HDC YOU WISH TO INPUT THE "F", "G", AND "GAMMA" //
11C1,40H MATRICES FROM SUBROUTINE "SETUP" IN THE //,10X,40H METHOD DE
2SCRIBED ON THE PREVIOUS SCREEN? //,15X,19HTYPE "YES" OR "NO".)
920 FORMAT (///,41H.....OPTSYSX IS NOW TERMINATED.....//)
ENC

```

```

-----
C SUEROUTINE SETUP (BA,G,GAM,NS,NC,NG)
-----
C IMPLICIT REAL*8 (A-H,O-Z)
C DIMENSION BA(NS,NS),G(NS,NC),GAM(NS,NG),DUM(82,85)
C COMMON /PROG/ IOL,IQ,IR,ISS,IS,ITF1,ITF2,ITF3,ISDFW,IE,IDSTAB,IDEB
C 10G,ISET,IREG,IESD,IYU,INORM
-----
C FILE DEFINITIONS
-----
C CALL FBTCMS ('FILEDEF ','03 ','DISK ','X29A82 ',
C 1 'DATA ','A ')
-----
C THIS IS AN EXAMPLE OF AN 82 X 85 DATA FILE X29A82 DATA A1 READ FROM
C A USER'S DISK AND CONVERTED (FROM A "DUMMY" ARRAY NAMED 'DUM') TO A
C SYMMETRIC ARRAY. THE FORMAT STATEMENT MUST MATCH YOUR DISK DATA
C FORMAT OR THE PROGRAM WILL FAIL! NOTE: ALL PROGRAM DIMENSIONS
C MUST BE ENLARGED ACCORDINGLY FOR A SYSTEM OF THIS SIZE.
-----
C READ (3,50) ((DUM(I,J),J=1,85),I=1,NS)
C DO 20 I=1,NS
C DO 10 J=1,NS
C BA(I,J)=DUM(I,J)
10 CONTINUE
20 CONTINUE
-----
C THESE ARE EXAMPLES OF SEVERAL POSSIBLE METHODS OF ARRAY GENERATION
C WITHIN SUEROUTINE SETUP. THE "GAM" ARRAY WAS SET TO ZERO SINCE NO
C "NCISE" WAS PRESENT, AND THE NON-ZERO ELEMENTS OF THE "G" ARRAY WERE
C EXPLICITLY DEFINED. THEY COULD ALSO BE READ FROM FILES AS ABOVE.
-----
C DO 40 I=1,NS
C DO 30 J=1,NC
C GAM(I,J)=0.5D+00
C G(I,J)=0.5D+00
C G(82,1)=0.1000E+01
30 CONTINUE
40 CONTINUE
C RETURN
-----
C 50 FORMAT (5(E12.4))
C ENR

```

```

C-----
C SUBROUTINE CHECK (EPS,NC,NG,NC,IRET)
C CHECKS THE CONSISTENCY OF REQUESTED OPTIONS.
C-----
DOUBLE PRECISION EPS
COMMON /FREG/ IOL,IQ,IR,ISS,IM,ITP1,ITP2,ITP3,IPDPW,IP,IDSTAB,IDEB
10G,ISET,IREG,IESD,IYU,INORM
C-----SET LOCAL ANALYSIS *REN OL EIGEN SYS OR OL TF REQUESTED-----
IF (IM .EQ. 1 .AND. IOL .EQ. 0) IOL=1
IF (IOL .EQ. 3 .OR. ITP1 .NE. 0) IM=1
C-----CHECK TO SEE IF H MATRIX INPUT-----
IF (NO .NE. 0 .OR. ICL .GE. 2) GO TO 10
WRITE (5,90)
IRET=1
RETURN
CONTINUE
10
C-----TRANSFER FUNCTION CHECKS-----
IF (IE .EQ. 0) IE=6
EPS=10.**(-IE)
C-----OPEN LOOP TF-----
IF (ITP1 .EQ. 0 .OR. NC .NE. 0) GO TO 20
WRITE (5,100)
IRET=1
RETURN
C-----COMPENSATOR TF-----
20 IF (ITP3 .EQ. 0) GO TO 30
IF (IREG .EQ. 0 .AND. (NC .NE. 0 .AND. NG .NE. 0)) GO TO 30
WRITE (5,110)
IRET=1
RETURN
30 CONTINUE
C-----NOISE TF-----
IF (ITP2 .EQ. 0) GO TO 40
IF (NG .NE. 0 .AND. NC .NE. 0) GO TO 40
WRITE (5,120)
IRET=1
RETURN
C-----DESTABILIZATION RESTRICTIONS-----
40 IF (IDSTAB .EQ. 0) GO TO 50
IF (NC .EQ. 0) GO TO 50
IF (NG .NE. 0) IREG=1
WRITE (5,130)
IF (IREG .EQ. 1) GO TO 50
IRET=1
RETURN
50 CONTINUE
C-----PSD INPUT-----
IF (IPSD .EQ. 0) GO TO 30
IF (IPSD .LT. 0 .OR. IPSD .GT. 3) GO TO 60
IF (IYU .LT. 0 .OR. IYU .GT. 3) GO TO 60
IF (INCRH .LT. 0 .OR. INCRH .GT. NG+NO) GO TO 60
GO TO 70
WRITE (5,140)
IRET=1
RETURN
70 IF (IREG .EQ. 0 .AND. NC .NE. 0) GO TO 80
WRITE (5,150)
IRET=1
RETURN
80 CONTINUE
RETURN
90
90 FORMAT (//,5X,49H H - MATRIX MUST BE INPUT, I.E. "NO" MUST BE > 0.
1,/)
100 FORMAT (//,5X,46H(G) MATRIX MUST BE INPUT, I.E. NC MUST BE > 0.,/)
110 26HC COMPUTE OPEN LOOP T. F.//)
110 FORMAT (//,5X,48HREGULATOR AND FILTER SYNTHESIS MUST BE REQUESTED.
1,5X,44HIF THE SAME RUN TO COMPUTE COMPENSATOR T. F.//,5X,47H I.E.
2,IREG MUST = 0. "NC" AND "NG" MUST BE > 0.//)
120 FORMAT (//,5X,51HNOISE T. F. CALCULATED ONLY WHEN REGULATOR DESIGN
1E,5X,47HI.I.E. IREG MUST = 1. "NC" AND "NG" MUST BE > 0.//)
130 FORMAT (//,5X,47HDESTABILIZATION OPTION DESIGNED FOR A REGULATOR,
1,51,38HOR FILTER BUT NOT BOTH SIMULTANEOUSLY.//,5X,55HIF "NG" > 0
2,THE REGULATOR OPTION IS AUTOMATICALLY SET!//)
140 FORMAT (//,5X,49H ***** INCONSISTENT PSD INPUT FLAGS *****
1,/)

```

```
150  FORMAT (//,5X,44"BOTH A REGULATOR AND FILTER MUST BE RESIDENT",//,10
1X,42H"TO COMPUTE THE PSD OF A CONTROLLED SYSTEM!",//,10X,42H".E. IREG
2 MUST BE 0. AND "NC" MUST BE > 0.",//)
END
```

```

-----
C-----
SUBROUTINE INNER (NS,NC,NG,NG,N2,ACL,B,BA,CI,CR,CO,CWI,CWR,D,PBGC,
1 PBGE,G,GAM,GM,GN,HO,D1,D2,PRO,EM,SC,C,SC,AR,AI,A11,A21,X,NORM,NNO
2 RHT,DESTAB,AA,EH,CH,JCF,RES,AY,BB,CC,CP,G,GV,HY,HU,DSTORE,ISAF,IS
3 AH,ISAG,IGAM,IRET,PRTT,NRCW,NCCI)
C-----
IMPLICIT REAL*8 (A-H,C-Z)
C-----
DIMENSION ACL(NS,NS),B(NC,NC),EA(NS,NS),CI(NS),CR(NS),CO(NS,NS),C
1 I(NS),CWR(NS),PBGC(NC,NS),PBGE(NS,NO),G(NS,NS),EM(NS,NS),PRO(NS,NS)
2 ),SC(NC,NC),SC(NS,NS),AR(N2),AI(N2),A11(NS,NS),A21(NS,NS),X(N2,N2)
3 ,GM(NS,NS),HO(NC,NS),D1(N2),D2(N2),EM(N2,N2),J(NG,NG),D(NC,NC),GAM
4 (NS,NG),NORM(NS,NS),NORMI(NS,NS),DESTAB(NS),AA(NS,NS),EM(NS,NC),
5 CM(NC,NS),JCF(N2),RES(N2),AY(NC,NC),BB(N2),CC(N2),C2(NS),GA(N2,NG)
6 ,GV(N2,NC),HY(NC,N2),HU(NC,N2),DSTORE(NS,NS),PRTT(16,16)
C-----
COMMON /PROG/ IOL,IQ,IR,ISS,IH,ITF1,ITF2,ITF3,IFDPW,IE,IDSTAB,IDEB
1UG,ISET,IEEG,IEFD,IYU,I,NORM
C-----
REAL*4 FMT(20)
C-----
OUTPUT OPTIONS-----
C--- IOL=1 IF THE OPEN LOOP EIGENSYSTEM IS DESIRED--OTHERWISE IOL=0
C--- IQ=1 IF THE RMS VALUES OF THE CONTROL AND STATE ARE TO BE FOUND
C--- IR=0 IF OPTIMAL FILTER AND REGULATOR EIGENSYSTEMS ARE TO BE FOUND
C--- IR=1 IF EXTERNAL C MATRIX IS SUPPLIED
C--- IR=2 IF EXTERNAL K IS SUPPLIED
C--- IR=3 IF EXTERNAL C AND K ARE SUPPLIED
C--- ISS=1 IF STEADY STATE VALUES ARE TO BE DETERMINED
C--- IN=1 IF MODAL STATES DESIRED
C-----
NSC=NS*NS
NH=NS
I=N2
CALL CHECK (EPS,NC,NG,NO,IRET)
IF (IRET.EQ.1) RETURN
IF (ISET.EQ.1) GO TO 20
CALL READP (NS,ISA?,EA)
IF (IDSTAB.EQ.0) GO TO 10
WRITE (5,1800)
CALL RDBREAL (ANSR)
DSTAR=ANSE
DO 10 I=1,NS
DESTAB(I)=DSTAR
CONTINUE
10 GO TO 30
20 CALL SETUP (BA,G,GAM,NS,NG,NC)
CONTINUE
30 WRITE (6,1380)
DO 40 I=1,NS
40 WRITE (6,1390) (EA(I,J),J=1,NS)
IF (IDSTAB.EQ.0) GO TO 50
WRITE (6,1480)
WRITE (6,1390) (DESTAB(I),I=1,NS)
50 CONTINUE
C-----
EIGENSYSTEM OF THE OPEN LOOP DYNAMICS-----
IF (IOL.EQ.0.AND.IQ.EQ.0) GO TO 90
IF (IOL.EQ.0.AND.NC.NE.0) GO TO 90
DO 60 I=1,NS
DO 60 J=1,NS
60 GN(I,J)=EA(I,J)
CALL BALANC (NS,NS,GN,LOW,IHIGH,D1)
CALL ORTHES (NS,NS,LOW,IHIGH,GN,D2)
CALL ORTRAN (NS,NS,LOW,IHIGH,GN,D2,SC)
CALL HCR2 (NS,NS,LOW,IHIGH,GN,CWR,CWI,SC,IERR)
IF (IERR.NE.0) CALL IREXIT (NS,GN,IERR)
CALL BALBAK (NS,NS,LOW,IHIGH,D1,NS,SC)
C-----
NORMALIZE AND PRINT OPEN LOOP EIGENSYSTEM-----
IWRITE=1
CALL CNOEN (CWR,CWI,SC,NS,IWRITE,NSQ,DDD,D1,D2,NORM,NORMI,HC,CH,
1 NO,NS)
IF (IOL.EQ.2) RETURN
IF (IQ.EQ.3.OR.(NC.NE.0.OR.IDSTAB.GT.0)) GO TO 90
DO 70 I=1,NS
IF (CWR(I).LT.0.) GO TO 70
WRITE (5,1490)
RETURN

```

```

70  CONTINUE
    IF (IOL.EQ.3) GO TO 130
    DO 80 I=1,NS
    DC 80 J=1,NS
80  W11(I,J)=SC(I,J)
    CALL INVT(NSC,W11,NS,DDD,D1,D2)
90  CONTINUE
    IF (IDSTAB.EQ.0) GO TO 130
----- FORM 0 * DIAG (DESTAB) * J-INV -----
    DC 100 J=1,NS
    DO 100 I=1,NS
100 AA(I,J)=WNORM(I,J)*DESTAB(J)
    DO 120 I=1,NS
    DC 120 J=1,NS
    DDD=0.00
110 DC 110 K=1,NS
    DDD=DDD+AA(I,K)*WFORM(K,J)
    DSTORE(I,J)=DDD
120 BA(I,J)=BA(I,J)+DDD
130 CONTINUE
    CALL REACH (NC,NS,ISAR,HO)
    WRITE (6,1440)
    DC 140 I=1,NO
140 WRITE (6,1390) (HC(I,J),J=1,NS)
    IF (IN.NE.1) GO TO 150
    CALL NODE (WNCRM,HO,CM,NS,NC,NS,2)
150 CONTINUE
    IF (IDFW.EQ.0) GO TO 170
    CALL REACT (NC,NC,D)
    WRITE (6,1470)
    DC 160 I=1,NO
160 WRITE (6,1390) (D(I,J),J=1,NC)
170 CCNTINUE
    NOB=0
    IF (NC.EQ.3) GO TO 590
    IF (IOL.EQ.3) GO TO 270
    IF (IS.NE.1.AND.IS.NE.3) GO TO 210
    IF (ISET.EQ.1) GO TO 190
    CALL REALG (NS,NC,ISAG,G)
180 CONTINUE
    CALL REACFB (NC,NS,FBGC)
    WRITE (6,1400)
    DC 190 I=1,NS
190 WRITE (6,1390) (G(I,J),J=1,NC)
    IF (IN.NE.1) GO TO 200
    CALL NODE (WNCMI,G,EM,NS,NS,NC,0)
200 CCNTINUE
    GO TO 330
210 DO 220 I=1,NS
    DO 220 J=1,NS
220 RM(I+MH,J)=0.0
    CALL REACAY (NO,AY)
    DO 240 I=1,NO
    DO 240 J=1,NS
    DDE=0.00
    DO 230 K=1,NO
230 DDE=DDD+AY(I,K)*EC(K,J)
240 AA(I,J)=DDD
    WRITE (6,1460)
    DO 250 I=1,NO
250 WRITE (6,1390) (AY(I,J),J=1,NC)
    DO 260 I=1,NS
    DO 260 J=1,NS
    DO 260 K=1,NO
    RM(I+MH,J)=RM(I+MH,J)+AA(K,I)*EO(K,J)
260 RM(I+MH,J)=RM(I+MH,J)+AA(K,I)*EO(K,J)
270 IF (ISET.EQ.1) GO TO 280
    CALL REALG (NS,NC,ISAG,G)
280 CONTINUE
    IF (IOL.EQ.3) GO TO 290
    CALL REACT (NC,E)
290 WRITE (6,1400)
    DO 300 I=1,NS
300 WRITE (6,1390) (G(I,J),J=1,NC)
    IF (IN.NE.1) GO TO 310
    CALL NODE (WNORMI,G,EM,NS,NS,NC,0)
310 CONTINUE

```



```

460 BA(I,I)=EA(I,I)-DESTAB(I)
IR=1
470 CONTINUE
C-----CALCULATION OF FEEDBACK GAIN-----
C-----FEEDBACK GAINS--> U = -(B*INVERSE)*GT*GN-----
C-----CALCULATE GT-----
DO 490 I=1,NC
DO 490 J=1,NS
FBG(I,J)=0.00
DO 480 K=1,NH
480 FBG(I,J)=FBG(I,J)+G(K,I)*GN(K,J)
490 FBGC(I,J)=-PRO(I,J)/E(I,I)
IF (IDSTAE.EQ.1) GO TO 500
C-----NORMALIZE AND PRINT OPT. REG. CLOSED LOOP EIGENSYSTEM-----
IWRITE=2
CALL CNOEM (CWR,CWI,SC,NS,IWRITE,NSQ,DDD,D1,D2,*NORM,*NORMI,FBGC,
1AA,NC,NS)
C-----THE OPTIMUM FEEDBACK CONTROL GAINS-----
500 WRITE (6,1580)
DC 510 I=1,NC
510 WRITE (6,1590) (FBGC(I,J),J=1,NS)
C-----COMPUTE MODAL C MATRIX OPEN LOOP U-INVERSE SAVED IN *NORMI-----
IF (IM.NE.1) GO TO 530
C-----IN COMPUTING MODAL C RECCOMPUTE U CFEN LOCP SINCE *NORM USED TO STORE
C U & U-INV FOR CLOSED LOOE SYSTEMS; *NORMI USED TO SAVE U-INV OPEN LCCP
C-----
DC 520 I=1,NS
DO 520 J=1,NS
520 WNCRM(I,J)=*NORMI(I,J)
CALL MINV (NSC,WNCRM,NS,DDD,D1,D2)
CALL MODE (WNCRM,FBGC,AA,NS,NC,NS,3)
530 CONTINUE
C-----THE CLOSED LOOP DYNAMICS MATRIX-----
DO 550 I=1,NS
DO 550 J=1,NS
SUM=0.00
DO 540 K=1,NC
540 SUM=SUM+G(I,K)*FBGC(K,J)
550 ACL(I,J)=EA(I,J)+SUM
WRITE (6,1600)
CALL SAPRNT (NH,MH,HH,5,ACL,4,'(5(1X,1PD13.5))')
IF (IR.NE.1.AND.IR.NE.3) GO TO 590
DO 560 I=1,NS
DO 560 J=1,NS
560 GN(I,J)=ACL(I,J)
CALL BALANC (NS,NS,GN,LOW,IHIGH,D1)
CALL ORTHES (NS,NS,LOW,IHIGH,GN,D2)
CALL ORTRAM (NS,NS,LOW,IHIGH,GN,D2,SC)
CALL HOR2 (NS,NS,LOW,IHIGH,GN,CWR,CWI,SC,IERR)
IF (IERR.NE.0) CALL EREXIT (NS,GN,IERR)
CALL BALBAK (NS,NS,LOW,IHIGH,C1,NS,SC)
C-----NORMALIZE AND PRINT CLOSED LOOP SUBOPT. REG. EIGENSYSTEM-----
IWRITE=3
CALL CNORM (CWR,CWI,SC,NS,IWRITE,NSQ,DDD,D1,D2,*NORM,*NORMI,FBGC,
1AA,NC,NS)
DO 570 I=1,NS
IF (CWR(I).LT.0.0) GO TO 570
WRITE (5,1610)
RETURN
CONTINUE
IF (IO.NE.1) GO TO 590
DO 580 I=1,NS
DO 580 J=1,NS
580 W11(I,J)=SC(I,J)
CALL MINV (NSC,W11,NS,DDD,D1,D2)
590 NOB=NO
IF (NG.EQ.0) RETURN
IF (ISET.EQ.1) GO TO 610
CALL REALG2 (NS,NG,IGAM,GAM)
610 CONTINUE
IF (IOL.EQ.3) GO TO 620
CALL READC (NG,Q)
620 WRITE (6,1420)
DO 630 I=1,NS
630 WRITE (6,1390) (GAM(I,J),J=1,NG)

```



```

1NO,YS)
770 DC 780 I=1,MH
DO 780 J=1,NO
780 FRC(I,J)=+HO(J,I)/RC(J,J)
DO 790 I=1,MH
DO 790 J=1,NO
FBGE(I,J)=0.00
DC 790 K=1,MH
790 FBGE(I,J)=FBGE(I,J)+GN(I,K)*FRC(K,J)
IF (IDSTAR.EQ.1) GO TO 810
WRITE (5,1670)
CALL RAPRNT (MH,MH,MH,5,GN,4,'(5(1X,1PD13.6))')
WRITE (6,1630)
DO 800 I=1,MH
800 X(I,I)=DSQRT(GN(I,I))
WRITE (6,1630) (X(I,I),I=1,MH)
810 WRITE (6,1630)
DO 820 I=1,MH
820 WRITE (6,1640) (FBGE(I,J),J=1,NO)
C-----COMPUTE MODAL K MATRIX OPEN LOOP U-INV SAVED IN WNormI -----
IF (IN.NE.1) GO TO 830
CALL XCOE (WNormI,FBGE,AA,MH,MH,NO,4)
830 CONTINUE
C-----RESET FLAG AND F MATRIX FOR ITERATIVE DESTABILIZATION CASE-----
IF (IDSTAR.EQ.0) GO TO 850
DO 840 I=1,NS
DO 840 J=1,NS
840 BA(I,J)=EA(I,J)-ESTORE(I,J)
IR=2
850 CONTINUE
DO 870 I=1,NS
DC 870 J=1,NS
SUM=0.0
DC 860 K=1,NO
860 SUM=SUM+FBGE(I,K)*MC(K,J)
870 FRC(I,J)=BA(I,J)-SUM
WRITE (6,1650)
CALL RAPRNT (NS,NS,NS,5,PRO,4,'(5(1X,1PD13.5))')
IF (IR.LT.2) GO TO 890
CALL BALANC (NS,NS,PRO,LOW,IHIGH,D1)
CALL ORTHES (NS,NS,LOW,IHIGH,PRO,D2)
CALL ORTRAN (NS,NS,LOW,IHIGH,PRO,D2,GM)
CALL HCR2 (NS,NS,LOW,IHIGH,PRO,CR,CI,GM,IERR)
IF (IERR.NE.0) CALL IERRIT (NS,PRO,IERR)
CALL BALEAK (NS,NS,LOW,IHIGH,D1,NS,GM)
WRITE (6,1560)
C-----NORMALIZE AND PRINT SUBOPT. ESTIMATOR EIGENSYSTEM-----
IWRITE=5
CALL CNOEM (CR,CI,GM,NS,IWRITE,NSQ,DD,D1,D2,WNorm,WNormI,HO,AA,
1NC,NS)
DO 880 I=1,NS
IF (CB(I).LT.0.0) GO TO 880
WRITE (5,1660)
RETURN
880 CONTINUE
GO TO 900
890 IF (ID.EQ.0) GO TO 1260
900 DO 910 I=1,NO
DO 910 J=1,MH
FRC(I,J)=0.00
DO 910 K=1,NO
910 FRC(I,J)=FRC(I,J)+RC(I,K)*FBGE(J,K)
DO 920 I=1,MH
DO 920 J=1,MH
CQ(I,J)=0.00
DC 920 K=1,NO
920 CQ(I,J)=CQ(I,J)-FBGE(I,K)*PRO(K,J)
930 CONTINUE
C-----THE RMS STATE AND CONTROL RESPONSES-----
IR=IR+1
GO TO (1050,1090,940,940), IR
940 DO 950 I=1,NS
DO 950 J=1,NG
X(I,J)=0.0
DO 950 K=1,NG
950 X(I,J)=X(I,J)+GAN(I,K)*2(K,J)

```

```

DO 970 I=1,NS
DO 970 J=I,NS
SUM=0.0
DO 960 K=1,NG
1960 SUB=SUM-X(I,K)*GAM(J,K)
PRO(I,J)=SUM+CC(I,J)
FRC(J,I)=PRO(I,J)
CC(I,J)=SUM
CQ(J,I)=SCM
W21(I,J)=CM(I,J)
1970 W21(J,I)=GM(J,I)
CALL MINV(NS, W21, NS, DDD, D1, E2)
CALL SCOV(NS, GM, W21, CR, CI, NS, GN, W21, CR, CI, PRO, GN)
WRITE(6,1670)
CALL RAPRNT(MH, MH, MH, 5, GN, 4, '(5(1X,1ED13.0))')
WRITE(6,1680)
CC 980 I=1, MH
1980 X(I,I)=DESCAT(GN(I,I))
WRITE(6,1690) (X(I,I), I=1, MH)
IF (NC.EC.0) GC TC 1260
DO 1000 I=1, NC
DO 1000 J=1, NS
SUM=0.0
DO 990 K=1, NS
1990 SUM=SUM+FRC(I,K)*GN(K,J)
1000 X(I,J)=SUM
DC 1020 I=1, NS
DC 1020 J=1, NS
SUM=0.0
IF (NC.EC.0) GC TC 1020
DO 1010 K=1, NC
1010 SUM=SUM+G(I,K)*X(K,J)
1020 FRC(I,J)=CQ(I,J)+SUM
CALL SCOV(NS, SC, W11, CWR, CWI, NS, SC, W11, CWR, CWI, PRO, BA)
IF (NC.EC.0) GC TC 1040
DO 1030 I=1, NC
DC 1030 J=1, NS
W21(I,J)=C.0
DO 1030 K=1, NS
1030 W21(I,J)=W21(I,J)+FBGC(I,K)*BA(J,K)
1040 DO 1060 I=1, NS
DO 1060 J=1, NS
SUM=0.0
IF (NC.EC.0) GC TC 1060
DO 1050 K=1, NC
1050 SUM=SUM+G(I,K)*W21(K,J)
1060 FRC(I,J)=SUM
DO 1070 I=1, NS
DO 1070 J=I, NS
FRC(I,J)=FRC(I,J)+CQ(I,J)+PRO(J,I)
1070 PRO(J,I)=FRC(I,J)
CALL SCOV(NS, SC, W11, CWR, CWI, NS, SC, W11, CWR, CWI, PRO, CQ)
DO 1080 I=1, NS
DC 1080 J=I, NS
GN(I,J)=CQ(I,J)-BA(I,J)-BA(J,I)+GN(I,J)
1080 GN(J,I)=GN(I,J)
GO TO 1100
1090 CALL SCOV(NS, SC, W11, CWR, CWI, NS, SC, W11, CWR, CWI, CQ, GN)
1100 IF (NC.EC.0) GC TC 1150
DO 1120 I=1, NS
DC 1120 J=1, NC
PRO(I,J)=C.00
DO 1110 K=1, NS
1110 PRO(I,J)=PRO(I,J)+GE(I,K)*FBGC(J,K)
1120 CONTINUE
DO 1140 I=1, NC
DO 1140 J=1, NC
SC(I,J)=C.00
DO 1130 K=1, NS
1130 SC(I,J)=SC(I,J)+FRC(I,K)*PRO(K,J)
1140 CONTINUE
1150 IF (IREG.EQ.0) GO TO 1170
DO 1160 I=1, NS
DO 1160 J=1, NS
1160 CC(I,J)=GN(I,J)
GO TO 1190

```

```

117C WRITE (6,1700)
CALL RAPANT (ME,ME,ME,5,GM,4,'(5(1X,1PD13.6))')
IF (IR.GI.2) GC TC 1190
DO 1180 I=1,MH
DC 1180 J=1,MH
1180 CQ(I,J)=GN(I,J)+GE(I,J)
1190 CONTINUE
WRITE (6,1710)
CALL RAPANT (ME,ME,ME,5,CQ,4,'(5(1X,1PD13.6))')
IF (NC.EC.0) GC TC 1210
WRITE (6,1720)
DO 1200 I=1,NC
1200 WRITE (6,1730) (SC(I,J),J=1,NC)
1210 DO 1220 I=1,NS
122C CQ(I,I)=DSORT(CQ(I,I))
IF (NC.EC.0) GC TC 1240
DO 1230 I=1,NC
1230 SC(I,I)=ESORT(SC(I,I))
124C WRITE (6,1740)
DO 1250 I=1,NS
IF (I.LE.NC) WRITE (6,1750) CQ(I,I),SC(I,I)
IF (I.GT.NC) WRITE (6,1750) CQ(I,I)
1250 CONTINUE
1260 IF (ITF3.EQ.0) GO TO 1290
C-----FORM COMPENSATOR FROM MEAS IC INPUT AND COMPUTE TF-----
DO 1280 I=1,NS
DO 1280 J=1,NS
SUM=0.0
DO 1270 K=1,NO
1270 SUM=SUM+FECE(I,K)*HO(K,J)
128C CQ(I,J)=ACL(I,J)-SUM
WRITE (6,1760)
ITFX=3
IZERO=0
CALL TF (NS,NS,NSC,CC,AA,NO,FECE,BM,NC,FBGC,CM,IZERO,D,BB,CC,CP,
1WR,WI,CWR,CWI,SC,JCF,RES,D1,D2,DDD,SES,ITF3,ITFX)
1290 CONTINUE
C-----COMPUTE PSD FUNCTIONS OF THE CONTROLLED SYSTEM-----
IF (IPSD.EQ.0) GO TO 1310
IF (IYU.LT.3) GC TC 1300
CALL PSDCAL (M,NS,RM,X,NC,GW,GV,FBGC,NO,HY,HU,HO,FBGE,NG,
1 GAN,ACL,BA,WR,WI,D1,D2,JCF,RES,C,RC,EE,CC,1,IPSD,INORM)
CALL PSDCAL (M,NS,RM,X,NC,GW,GV,FBGC,NO,HY,HU,HO,FBGE,NG,
1 GAN,ACL,BA,WR,WI,D1,D2,JCF,RES,C,RC,EE,CC,2,IPSD,INORM)
GO TO 1310
1300 CALL PSDCAL (M,NS,RM,X,NC,GW,GV,FBGC,NO,HY,HU,HO,FBGE,NG,
1 GAN,ACL,BA,WR,WI,D1,D2,JCF,RES,C,RC,EE,CC,IYU,IPSD,INORM)
1310 IF (ISS.EQ.0) RETURN
IF (NC.NE.0) GO TO 1330
DO 1320 I=1,NS
DO 1320 J=1,NS
132C ACL(I,J)=BA(I,J)
1330 CONTINUE
CALL INVV (NSQ,ACI,NS,DDD,D1,D2)
CALL REACH (NG,WR)
WRITE (6,1770) (WR(I),I=1,NG)
WRITE (6,1780)
DO 1340 I=1,NS
WI(I)=0.0
DO 1340 J=1,NG
1340 WI(I)=WI(I)+GAN(I,J)*WR(J)
DO 1360 I=1,NS
CR(I)=0.0
DO 1350 J=1,NS
1350 CR(I)=CR(I)-ACL(I,J)*WI(J)
1360 WRITE (6,1390) CR(I)
DO 1370 I=1,NC
CI(I)=0.0
DO 1370 J=1,NS
1370 CI(I)=CI(I)+FEGC(I,J)*CR(J)
WRITE (6,1790) (CI(I),I=1,NC)
RETURN
C-----
C670 FCHMAT (2X,196C14.6,/,2X,6D14.6)
1380 FORNAT (//,5X,45HCEEN LOOE DYNAMICS MATHIX.....P...//)
1390 FORNAT (10(2X,0PC11.4))

```

```

1400 FORMAT (//,5X,45HTHE CONTROL DISTRIBUTION MATRIX.....G...//)
1410 FORMAT (//,5X,45HTHE CONTROL CCST MATRIX.....B...//)
1420 FORMAT (//,5X,45HPROCESS NOISE DISTRIBUTION MATRIX.....GAMMA...//)
1430 FORMAT (//,5X,45HPOWER SPECTRAL DENSITY - PROCESS NOISE.....C...//)
1440 FORMAT (//,5X,45HMEASUREMENT SCALING MATRIX.....H...//)
1450 FORMAT (//,5X,45HPOWER SPECTRAL DENSITY-MEASUREMENT NOISE.....R...//)
1460 FORMAT (//,5X,45HCLUTER CCST MATRIX.....A...//)
1470 FORMAT (//,5X,45HMEASUREMENT FEEDTHROUGH MATRIX.....D...//)
1480 FORMAT (//,25X,23H...DESTABILIZATION CASE.....//,10X,39HTHE FOLLOW
1481 17ING VALUES WILL BE ADDED DOWN, //,10X,49HTHE DIAGONAL OF THE "F" MA
1482 2TRIX TO DESTABILIZE IT, //,10X,41HOPTIMAL GAINS FOR THE DESTABILIZE
1483 3D SYSTEM, //,10X,39HARE THEN USED AS FIXED SUBOPTIMAL GAINS, //,10X,28
1484 4HFOR THE SYSTEM CALCULATIONS.//)
1490 FORMAT (//,43H PROGRAM TERMINATING DUE TO UNSTABLE SYSTEM)
1500 FORMAT (//,2X,31HOPEN LOOP TRANSFER FUNCTIONS...//)
1510 FORMAT (//,5X,32H EDLER-LAGRANGE SYSTEM MATRIX...//)
1520 FORMAT (//,1X,43HEIGENVALUES AND EIGENVECTORS OF THE 2N X 2N, //,2X,
1521 14SHEULER-LAGRANGE SYSTEM AFTER HQB2.....//)
1530 FORMAT (//,1X,1P2F11.6)
1540 FORMAT (//,1X)
1550 FORMAT (//,2X,41HEIGENSYSTEM OF OPTIMAL REGULATOR.....//)
1560 FORMAT (//,2X,41HEIGENSYSTEM OF OPTIMAL ESTIMATOR.....//)
1570 FORMAT (//,5X,39H EIGENVECTORS FROM GAIN PRIOR TO CNORM, //)
1580 FORMAT (//,1X,57HTHE OPTIMAL FEEDBACK GAIN CONTROL MATRIX...C=B*INV
1581 1*GT*S...//)
1590 FORMAT (//,1C(2X,1PDI1.4))
1600 FORMAT (//,2X,45HTHE CLOSED LOOP DYNAMICS MATRIX.....F-G*C...//)
1610 FORMAT (//,60H PROGRAM TERMINATING DUE TO UNSTABLE CLOSED LOOP
1611 1 SYSTEM)
1620 FORMAT (//,2X,61HNOISE TRANSFER FUNCTIONS ,32H THROUGH THE CLOSED L
1621 1 LOOP SYSTEM...//)
1630 FORMAT (//,5X,45H FILTER STEADY STATE GAINS.....K...//)
1640 FORMAT (//,1X,2X,1P6D14.6)
1650 FORMAT (//,1X,43H THE CLOSED LOOP FILTER DYNAMICS MATRIX IS...//)
1660 FORMAT (//,43H PROGRAM TERMINATING DUE TO UNSTABLE FILTER)
1670 FORMAT (//,5X,45H THE COVARIANCE OF THE ESTIMATION ERROR.....P...//)
1680 FORMAT (//,5X,45H RMS VALUES OF THE ESTIMATION ERROR.....//)
1690 FORMAT (//,5(1X,1PDI3.5))
1700 FORMAT (//,5X,45H THE COVARIANCE OF THE ESTIMATE.....XHAT...//)
1710 FORMAT (//,5X,45H THE STATE COVARIANCE MATRIX.....X=XHAT + P...//)
1720 FORMAT (//,5X,45H THE CONTROL COVARIANCE.....J=C*XHAT*CT...//)
1730 FORMAT (//,1P6D14.8)
1740 FORMAT (//,2X,18H STATE RMS RESPONSE, 20X, 20H CONTROL RMS RESPONSE, //)
1750 FORMAT (//,1X,1P15.7,2EX, D15.7)
1760 FORMAT (//,5X,50H COMPENSATOR TRANSFER FUNCTIONS FROM MEAS. TO INEU
1761 1T, //,5X,52H.....J/2 = -C*(SI-F+G*C+K*H) INV*K...//)
1770 FORMAT (//,2X,46H STEADY DISTURBANCE VECTOR.....W...//)
1771 1/10(1X,1ED12.4//)
1780 FORMAT (//,5X,45H STEADY STATE VALUES OF STATE VAR. ARE.....//)
1790 FORMAT (//,5X,47H STEADY STATE CONTROL IS .....//)
1791 1/10(1X,1EC12.4//)
1800 FORMAT (//,5X,49H ENTER THE MAGNITUDE OF THE DESTABILIZATION VECTOR
1801 1, //,8X,47H TO BE ADDED DOWN THE DIAGONAL OF THE "F"-MATRIX, //,8X,18HT
1802 20 DESTABILIZE IT.//)
ENC

```

```

-----
SUBROUTINE RAPENT (NMAX, S, N, L, A, IDIM, FMT)
REAL*8 A (NMAX, N)
DIMENSION FMT (IDIM)
NU=1
DC 20 NL=1, N, L
IF (NU.SI.N) NU=N
DC 10 I=1, N
10 WRITE (6, FMT) (A (I, J), J=NL, NU)
20 WRITE (6, 30)
  NU=NU+L
  RETURN
30 FORMAT (1X)
  ENC

```

```

C-----
SUBROUTINE RGAIN (M,NS,NC,NCB,WR,WI,VF,GN,W11,TCB,W21,LT,C,CI,CT,M
1HS,MT)
IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION WR(M),WI(M),VF(M,M),GN(NS,NS)
DIMENSION W11(NS,NS),TCB(M,M),W21(NS,NS),LT(NS),MT(NS)
DIMENSION C(NS),CI(NS),CT(NS,NS)
K=1
KP=1
KN=1
NRZEV=0
NCFZEV=0
10 IF (K.GT.M) GO TO 210
C-----
C CHECK FOR SIGVAL AT CF NEAR J-OMEGA AXIS TO INCLUDE IN E-L EIGSYS
C TURN FIRST ONE POSITIVE AND SECOND ONE NEGATIVE
C-----
EIGVR=CAES(WR(K))
IF (EIGVR.GE.1D-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 80
30 WR(K)=-EIGVR
WRITE (6,290)
GO TO 150
40 NCFZEV=NCFZEV+1
IF (NCFZEV.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
C-----
80 EIGENVECTOR FOR REAL EIGENVALUE, POSITIVE-----
IF (NOB.EC.0) GO TO 100
DO 90 J=1,M
90 TCB(J,KP)=VF(J,K)
100 KP=KP+1
K=K+1
GO TO 10
C-----
110 EIGENVECTOR FOR COMPLEX EIGENVALUE, POSITIVE REAL PART-----
IF (NOB.EQ.0) GO TO 130
DO 120 J=1,M
FR=VF(J,K)
FI=-VF(J,K+1)
120 TCB(J,KP)=FR+FI
130 TCE(J,KP+1)=FR-FI
KP=KP+2
K=K+2
GO TO 10
140 IF (WI(K)) 180,150,180
C-----
150 EIGENVECTOR FOR REAL EIGENVALUE, NEGATIVE REAL PART-----
C(KN)=WR(K)
CI(KN)=WI(K)
IF (NOB.NE.0) GO TO 170
KNS=KN+NS
DO 160 J=1,M
160 TCB(J,KNS)=VF(J,K)
170 KN=KN+1
K=K+1
GO TO 10
C-----
180 EIGENVECTOR FOR COMPLEX EIGENVALUE, NEGATIVE REAL PART-----
RR=WR(K)
RI=WI(K)
C(KN)=RR
CI(KN+1)=RI
180 C(KN+1)=RR
CI(KN)=RI
CI(KN+1)=-RI
IF (NOB.NE.0) GO TO 200
KNS=KN+NS
DO 190 J=1,M

```

```

      FR=VF(J,K)
      FI=-VF(J,K+1)
      TCB(J,KN)=FR+FI
190   TCB(J,KN+1)=FR-FI
200   RN=KN+2
      K=K+2
      GO TO 10
210   CONTINUE
      IF (NOB.NE.J) GO TO 240
C-----FORMATION OF #11-----
      DO 220 I=1,NS
      DO 220 J=1,NS
      W11(I,J)=TCB(I,J+NS)
220   CT(I,J)=W11(I,J)
C-----FORMATION OF #21-----
      DO 230 I=1,NS
      DO 230 J=1,NS
230   W21(I,J)=TCB(I+NS,J+NS)
240   IF (NOB.EQ.O) GO TO 260
      DO 250 I=1,NS
      DO 250 J=1,NS
      W21(I,J)=-TCB(I,J)
250   W11(I,J)=TCB(I+NS,J)
260   CONTINUE
C-----INVERT W11-----
      NSQ=NS*NS
      CALL INV (NSQ,W11,NS,DETC,LT,MT)
C-----CALCULATE THE GAIN MATRIX-----
      DO 270 IL=1,NS
      DO 270 JL=1,NS
      GN(IL,JL)=0.00
      DO 270 KL=1,NS
270   GN(IL,JL)=GN(IL,JL)+W21(IL,KL)*W11(KL,JL)
      IF (NOB.EQ.O) BETERN
      DO 280 I=1,NS
      DO 280 J=1,NS
280   CT(I,J)=W11(J,I)
      RETURN
C-----
290   FORMAT (1X,51H EULER-LAGRANGE EQUATIONS HAVE A REAL EIGENVALUE AT,
14H OR NEAR ZERO.)
300   FORMAT (1X,49H EULER-LAGRANGE EQUATIONS HAVE A COMPLEX PAIR OF ,40
1HEIGENVALUES AT OR NEAR THE J-COMPA AXIS.)
      END

```



```

-----
C-----
SUBROUTINE MINV (MSC,A,N,C,L,K)
IMPLICIT REAL*8 (A-E,O-Z)
DIMENSION A(MSC),I(N),J(N)
DOUBLE PRECISION A,C,BIGA,HOLD
N=N*N
D=.1D0
MK=-V
DO 180 K=1,N
MK=MK+N
L(K)=K
M(K)=K
KR=MK+K
BIGA=A(KK)
DC 20 J=K,N
IZ=N*(J-1)
DC 20 I=K,N
IJ=IZ+I
10 IF (DABS(BIGA)-CAES(A(IJ))) 10,20,20
BIGA=A(IJ)
L(K)=I
M(K)=J
20 CONTINUE
C-----INTERCHANGE ROWS-----
J=L(K)
IF (J-K) 50,50,30
30 KI=K-N
DO 40 I=1,N
KI=KI+N
HOLD=-A(KI)
JI=KI-K+J
40 A(KI)=A(JI)
A(JI)=HOLD
C-----INTERCHANGE COLUMNS-----
I=M(K)
IF (I-K) 80,80,60
60 JF=N*(I-1)
DO 70 J=1,N
JK=NK+J
JI=JP+J
HOLD=-A(JK)
70 A(JK)=A(JI)
A(JI)=HOLD
C-----DIVIDE COLUMN BY MINUS PIVOT-----
C----- (VALUE OF PIVOT ELEMENT IS CONTAINED IN BIGA) -----
80 IF (BIGA) 100,90,100
90 D=0.0D0
RETURN
DC 120 I=1,N
IF (I-K) 110,120,110
110 IK=NK+I
A(IK)=A(IK)/(-BIGA)
120 CONTINUE
C-----REDUCE MATRIX-----
DO 150 I=1,N
IK=NK+I
HOLD=A(IK)
IJ=I-N
DC 150 J=1,N
IJ=IJ+N
IF (I-K) 130,150,130
130 IF (J-K) 140,150,140
140 KJ=IJ-I+K
A(IJ)=HOLD*A(KJ)+A(IJ)
150 CONTINUE
C-----DIVIDE ROW BY PIVOT-----
KJ=K-N
DO 170 J=1,N
KJ=KJ+N
IF (J-K) 160,170,160
160 A(KJ)=A(KJ)/BIGA
170 CONTINUE
C-----PRODUCT OF PIVOTS-----
D=C*BIGA
C-----REPLACE PIVOT BY RECIPROCAL-----
A(KK)=(1.CD0)/BIGA

```

```

180  CONTINUE
C-----FINAL ROW AND COLUMN INTERCHANGE-----
      K=N
190  K=(K-1)
      IF (K) 260,260,200
200  I=L(K)
      IF (I-K) 230,230,210
210  JQ=I*(K-1)
      JR=N*(I-1)
      DO 220 J=1,N
          JK=JQ+J
          HOLD=A(JK)
          JI=JR+J
220  A(JK)=-A(JI)
230  A(JI)=HOLD
          J=M(K)
240  IF (J-K) 190,190,240
          KI=K-N
          DO 250 I=1,N
              KI=KI+M
              HOLD=A(KI)
              JI=KI-K+J
          A(KI)=-A(JI)
250  A(JI)=HOLD
          GO TO 190
260  R=0
      RETURN
      END

```

AD-A144 159

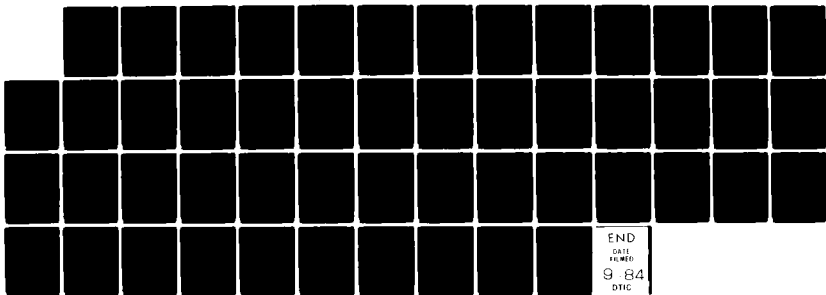
INTERACTIVE IMPLEMENTATION OF THE OPTIMAL SYSTEMS
CONTROL DESIGN PROGRAM (OPTSYSX) ON THE IBM 3033(U)
NAVAL POSTGRADUATE SCHOOL MONTEREY CA J G HODEN MAR 84

22

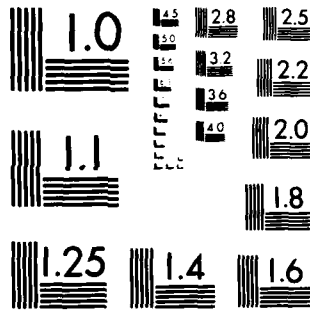
UNCLASSIFIED

F/G 9/2

NL



END
DATE
FILMED
9 84
DTIC



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

```

C=====
SUBROUTINE SCCV (NL,NL,WLI,VL1,VL2,NR,WR,WRI,VR1,VR2,J,X)
REAL*4 VL1(NL),VL2(NL),WL(NL,NL),WLI(NL,NL),X(NL,NR),Q(NL,NR),
1 VR1(NR),VR2(NR),WR(NR,NR),WFI(NR,NR)
REAL*8 A,E,C,D,K1,K2,K3,K4
10 DO 20 I=1,NL
DO 20 J=1,NR
X(I,J)=0
DO 20 II=1,NL
20 X(I,J)=X(I,J)+WLI(I,II)*Q(II,J)
DO 40 I=1,NL
DO 40 J=1,NR
Q(I,J)=0
DO 30 JJ=1,NR
30 Q(I,J)=Q(I,J)+X(I,JJ)*WR(J,JJ)
40 CONTINUE
I=1
50 IF (VL2(I)) 60,110,60
60 J=1
70 IF (VR2(J)) 80,90,80
80 A=VL1(I)+VR1(J)
B=-2.*VL2(I)*VR2(J)
C=A**2+VL2(I)**2+VR2(J)**2
D=C**2-B**2
K1=A*C/D
K2=- (VR2(J)*C+VL2(I)*B)/D
K3=- (VR2(J)*B+VL2(I)*C)/D
K4=-A*B/D
I1=I+1
J1=J+1
X(I,J)=+K1*Q(I,J)+K2*Q(I,J1)+K3*Q(I1,J)+K4*Q(I1,J1)
X(I,J1)=-K2*Q(I,J)+K1*Q(I,J1)-K4*Q(I1,J)+K3*Q(I1,J1)
X(I1,J)=-K3*Q(I,J)-K4*Q(I,J1)+K1*Q(I1,J)+K2*Q(I1,J1)
X(I1,J1)=+K4*Q(I,J)-K3*Q(I,J1)-K2*Q(I1,J)+K1*Q(I1,J1)
J=J+2
GO TO 100
90 A=VR1(J)+VL1(I)
B=A**2+VL2(I)**2
K1=A/B
K2=VL2(I)/B
X(I,J)=K1*Q(I,J)-K2*C(I+1,J)
X(I+1,J)=K2*Q(I,J)+K1*Q(I+1,J)
J=J+1
100 IF (J.LE.NR) GC TC 70
I=I+2
GO TO 160
110 J=1
120 IF (VR2(J)) 130,140,130
130 A=VR1(J)+VL1(I)
E=A**2+VR2(J)**2
K1=A/B
K2=VR2(J)/B
X(I,J)=K1*Q(I,J)-K2*C(I,J+1)
X(I,J+1)=K2*Q(I,J)+K1*Q(I,J+1)
J=J+2
GO TO 150
140 X(I,J)=Q(I,J)/(VR1(J)+VL1(I))
J=J+1
150 IF (J.LE.NR) GO TC 120
I=I+1
160 IF (I.LE.NL) GC TC 50
DO 170 I=1,NL
DO 170 J=1,NR
Q(I,J)=0
DO 170 II=1,NL
170 Q(I,J)=Q(I,J)+WL(I,II)*X(II,J)
DO 190 I=1,NL
DO 190 J=1,NR
Y(I,J)=0
DO 180 JJ=1,NR
180 Y(I,J)=Y(I,J)+Q(I,JJ)*WR(J,JJ)
190 CONTINUE
RETURN
END

```



```

C-----
SUBROUTINE CNORM (WZ,WY,VEC,NS,IWRITE,NSQ,DDD,D1,D2,WNORM,WNORMI,H
10,CM,N1,N2)
C-----
      WZ(I)      REAL PART OF I-TH EIGENVALUE
      WY(I)      COMPLEX PART OF I-TH EIGENVALUE
      VEC        MATRIX OF EIGHT EIGENVECTORS STORED IN REAL FORM
                PFCB HQR2
      NS         NO. OF STATES
      IWRITE     FLAG TO CONTROL FORMATS FOR DIFFERENT EIGENSYSTEMS
      WNORM      NORMALIZED MATRIX U OF RIGHT EIGENVECTORS STORED
                BY COLUMNS IN REAL FORM
      WNORMI     U-INVERSE 2*CONJUGATE OF LEFT EIGENVECTORS
                STORED BY ROW IN REAL FORM
      NSQ,DDD,D1,D2 - ARGUMENTS PASSED TO MINV
C-----
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*8 FIELD,CCMMA,SENCOL,RIGHT,FMT
      DIMENSION WZ(NS),WY(NS),VEC(NS,NS),WNORM(NS,NS),WNORMI(NS,NS),STOR
1E(6),D1(NS),D2(NS),FMT(14),HO(N1,N2),CM(N1,N2)
      DATA FILEL,SHR,2.5,COMMA,SH,',' ,S,SENCOL,SH,':',/,RIGHT,1H,/,FMT/
16H(1X,1F13.1H /,SENCOL/4H,':',/
C-----
      NORMALIZE COMPLEX EIGENVECTORS BY LARGEST ELEMENT-----
      KK=0
      LB=0
      LC=0
      DO 50 K=1,NS
      IF (KK.EQ.1) GC TC 40
      IF (DABS(WY(K)).LT.1.D-10) GO TO 50
      LC=LC+1
      EMAX=0.D0
      DO 20 I=1,NS
      CMOD=VEC(I,K)**2+VEC(I,K+1)**2
      IF (CMOD-EMAX) 20,10,10
10      EMAX=CMOD
      N=I
20      CONTINUE
      VMB=VEC(N,K)
      VMI=VEC(N,K+1)
      DO 30 I=1,NS
      VR=VEC(I,K)
      VI=VEC(I,K+1)
      VECRN=(VR*VMB+VI*VMI)/EMAX
      VECIN=(-VR*VMI+VI*VMB)/EMAX
      WNORM(I,K)=VECRN
      WNORM(I,K+1)=VECI
30      CCONTINUE
      KK=1
      GO TO 50
40      KK=0
50      CONTINUE
C-----
      NORMALIZE REAL EIGENVECTORS BY THE TOTAL LENGTH-----
      DO 80 K=1,NS
      IF (DABS(WY(K)).GE.1.D-10) GO TO 80
      LB=LB+1
      REMOD=0.D0
      DO 60 I=1,NS
      REMOD=VEC(I,K)**2+REMOD
60      REMOD=DSQRT(REMOD)
      DO 70 I=1,NS
      RVEC=VEC(I,K)/REMOD
      RNORM(I,K)=RVEC
70      CONTINUE
80      CONTINUE
      GO TO (90,100,110,120,130), IWRITE
90      WRITE (6,320)
      GO TO 140
100     WRITE (6,330)
      GO TO 140
110     WRITE (6,340)
      GO TO 140
120     WRITE (6,350)

```

```

130 GC TO 140
140 WRITE (6,360)
      KK=0
      NPRTW=0
      NFMTW=1
      DO 180 I=1,NS
      IF (KK.EC.1) GC TO 170
      IF (DABS(WY(I)).GT.1.D-10) KK=1
C-----PRINT OUT NO MORE THAN 6 WORDS, NOT SEPARATING COMPLEX SIGNAL-----
      IF (NPRTW.LT.5.OR.(NFRTW.EQ.5.AND.KK.EC.0)) GO TO 150
      FMT(NFMTW+1)=RIGHT
      WRITE (6,FMT) (STCRE(J),J=1,NFRTW)
      NPRTW=0
      NFMTW=1
150 NPRTW=NPRTW+1
      NFMTW=NFMW+1
      IF (KK.EC.1) GC TO 160
      STCRE(NFRTW)=WZ(I)
      FMT(NFMTW)=FIELD
      NFRTW=NFMW+1
      FMT(NFRTW)=SEMCOL
      GC TO 180
160 STORE(NPRTW)=WZ(I)
      FMT(NFMTW)=FIELD
      FMT(NFMTW+1)=CCMMA
      STORE(NPRTW+1)=WY(I)
      FMT(NFMTW+2)=FIELD
      FMT(NFMTW+3)=SEMCCL
      NFMTW=NFMW+3
      NFRTW=NPRTW+1
      GO TO 180
170 KK=0
180 CONTINUE
      FMT(NFMTW)=SEMENC
      FMT(NFMTW+1)=RIGHT
      WRITE (6,FMT) (STCRE(J),J=1,NFRTW)
      IF (NFRTW.EC.1) GO TO 190
      WRITE (6,370)
      GO TO 200
190 WRITE (6,380)
200 CALL RAPRNT (NS,NS,NS,6,#NORM,4,'(6 (1X,1PD13.6))')
      GC TO (230,210,10,220,220), IWRITE
210 CALL YCDE (#NORM,HO,CM,NS,1,2,3)
      GO TO 230
220 CALL YCDE (WNCEN,HO,CM,YS,1,2,6)
230 GO TO (240,250,260,270,280), IWRITE
240 WRITE (6,390)
      GO TO 290
250 WRITE (6,400)
      GO TO 290
260 WRITE (6,410)
      GO TO 290
270 WRITE (6,420)
      GO TO 290
280 WRITE (6,430)
C-----SAVE U-INVERSE CEEN LOCP IN #NORMI-----
290 IF (IWRITE.GT.1) GC TO 310
      DO 300 I=1,NS
      DO 300 J=1,NS
300 WNMORI(I,J)=WNCEN(I,J)
      CALL MINV (NS,WNCENI,NS,DDD,D1,D2)
      CALL RAPRNT (NS,NS,NS,6,#NORMI,4,'(6 (1X,1PD13.6))')
      RETURN
310 CALL MINV (NS,WNCEN,NS,DDD,D1,D2)
      CALL RAPRNT (NS,NS,NS,6,#NORM,4,'(6 (1X,1PD13.6))')
      RETURN
C-----
320 FORMAT (//1X,42HOPEN LOOP EIGENVALUES.....DET (SI-F).....//)
330 FORMAT (//1X,46HC-LOCP OPTIMAL REG. E-VALUES.....DET (SI-F+G+C).....//)
340 FORMAT (//1X,46HC-LOCP SUBOPT. REG. E-VALUES.....DET (SI-F+G+C).....//)
350 FORMAT (//1X,46HC-LOCP OPTIMAL EST. E-VALUES.....DET (SI-F+G+H).....//)
360 FORMAT (//1X,46HC-LOCP SUBOPT. EST. E-VALUES.....DET (SI-F+G+H).....//)
370 FORMAT (//1X,46HCEN LOOP RIGHT EIGENVECTOR MATRIX.....//)
380 FORMAT (//1X,46HC-LOCP RIGHT EIGENVECTOR MATRIX.....//)
390 FORMAT (//1X,46HCEN LOOP LEFT EIGENVECTOR MATRIX.....//)
400 FORMAT (//1X,46HC-LOCP OPT. REG. LEFT E-VECTOR MATRIX.....//)

```



```
410 FCENAT (//1X,46HC-LOOP SUBOPT-REG. LEFT E-VECTOR MATRIX..N-INV,//)
420 FENAT (//1X,46HC-LCCP OPT. FILTER LEFT E-VECTOR MATRIX..N-INV,//)
430 FENAT (//1X,51HC-LOOP SUBOPT. FILTER LEFT E-VECTOR MATRIX..N-INV.
      ENC)
```

```

-----
SUBROUTINE TP (N, NM, NSQ, A, AA, M, B, BM, L, C, CM, IPDFW, D, BB, CC, CP,
1 EVR, EVI, PR, PI, SC, JCF, RES, D1, D2, DDD, EPS, ITF, ITFX)
IMPLICIT REAL*8(A-Z)
DIMENSION A(N,N), AA(N,N), B(N,M), BM(N,M), C(L,N), CM(L,N), D(L,M), BB(N
1), CC(N), CF(N), EVR(N), EVI(N), PR(N), PI(N), SC(N,N), JCF(N), RES(N), D1(N
2), D2(N)
C--SAVE COMPUTATION ON CL AND CL SYS WITH LOCAL WORK DONE IN OPTSYS-----
IF (ITFX.EQ.1) GO TO 50
IF (ITFX.EQ.2) GO TO 10
CALL POLES (N, NM, A, AA, M, B, L, C, PR, PI, D1, D2, JCF, SC)
-----
10 DO 20 I=1,N
DO 20 J=1,N
20 AA(I,J)=SC(I,J)
DO 30 I=1,L
DO 30 J=1,N
CM(I,J)=C-DO
DO 30 K=1,N
30 CM(I,J)=CM(I,J)+C(I,K)*AA(K,J)
CALL INV (NSQ, AA, N, DDD, D1, D2)
DO 40 I=1,M
DO 40 J=1,M
BM(I,J)=0.00
DO 40 K=1,M
40 BM(I,J)=BM(I,J)+AA(I,K)*B(K,J)
50 CONTINUE
DO 60 I=1,L
DO 60 J=1,N
IF (ITF.NE.3) CALL ZEROS (I,J,IPDFW,N,NM,A,AA,M,B,L,C,D,BB,CC,CP
1, EVR, EVI, D1, D2, EPS)
IF (ITF.NE.2) CALL RESID (I,J,N,JCF,M,BB,L,CM,PR,PI,RES,BB,CC,1)
60 CONTINUE
RETURN
END

```

```

C-----
SUBROUTINE POLES (N, NM, A, AA, E, E, L, C, EVB, EVI, D1, D2, JCF, SC)
IMPLICIT REAL*8(A-H, C-L)
DIMENSION A(N, N), AA(N, N), B(N, N), C(L, N), EVB(N), EVI(N), D1(N), D2(N), J
1CF(N), SC(N, N)
DO 10 I=1, N
DO 10 J=1, N
10 AA(I, J)=A(I, J)
CALL BALANC (NM, N, AA, LOW, IHIGH, D1)
CALL ORTHES (NM, N, LOW, IHIGH, AA, D2)
CALL ORTRAN (NM, N, LOW, IHIGH, AA, D2, SC)
CALL HCR2 (NM, N, LOW, IHIGH, AA, EVB, EVI, SC, IERR)
IF (IERR .NE. C) GO TO 30
CALL BALBAC (NM, N, LOW, IHIGH, D1, N, SC)
WRITE (6, 40)
DO 20 I=1, N
20 WRITE (6, 50) EVB(I), EVI(I)
RETURN
30 WRITE (6, 60)
RETURN
C-----
40 FCENAT (///, 28H TF DENOMINATOR EIGEN VALUES: ,/)
50 FCRHAT (//, 2X, 3H , , P13.6, 4H) + J(, P13.6, 1B)
60 FCRHAT (35H FAILURE IN HCR2, CALCULATING POLES)
END

```

```

-----
SUBROUTINE ZERCS (K1,K2,IPDFW,N,NM,A,AA,M,B,L,C,D,BB,CC,CP,EVR,EVI
1, D1,E2,EPS)
IMPLICIT REAL*8 (A-H,C-Z)
DIMENSION A(N,M),AA(N,M),B(N,M),C(L,M),D(L,M),BB(N),CC(N),CP(N),EV
IR(N),EVI(N),D1(M),D2(M)
DOUBLE PRECISION SCL,DABS
DC 10 I=1,M
BB(I)=B(I,K1)
CC(I)=C(K2,I)
DO 10 J=1,M
AA(I,J)=A(I,J)
10 WRITE (6,50) K1,K2
IF (IPDFW.EQ.0) GO TO 20
H=D(K2,K1)
IF (DABS(H).LE.EPS) GO TO 20
JJ=N
GC TO 50
20 NN=N-1
DC 30 I=1,NN
H=SCL(N,BB,CC)
CALL CCOMP(N,NN,AA,CC,CP)
IF (DABS(H).GT.EPS) GO TO 40
30 CONTINUE
H=SCL(N,BB,CC)
WRITE (6,100) H
GO TO 70
40 JJ=N-I
50 WRITE (6,110) JJ,H
CALL ACOMP(N,NM,AA,EB,CC,H)
CALL BALANC(NM,N,AA,LOW,HIGH,D1)
CALL ORTHES(NM,N,LOW,HIGH,AA,D2)
CALL HOR(NM,N,LOW,HIGH,AA,EVR,EVI,IERR)
IF (IERR.NE.0) GO TO 80
WRITE (6,120)
DC 60 I=1,N
60 WRITE (6,130) EVR(I),EVI(I)
70 RETURN
80 WRITE (5,140)
RETURN
-----
90 FORMAT (///,17H IF FOR INPUT NO.,I3,15H AND OUTPUT NO.,I3,1H:)
100 FORMAT (///,5X,27HNO FINITE ZEICS. IF GAIN =,E12.4)
110 FORMAT (///,3X,20HORDER OF NUMERATOR = I3,3X,9HTZ GAIN = ,E12.4)
120 FORMAT (///,3X,57HNUMERATOR EIGENVALUES (INCLUDING EXTRANEOUS ZERO V
1ALUES):)
130 FORMAT (//,4X,1H(,F13.6,4H)+J(,F13.6,1H))
140 FORMAT (//,5H FAILURE IN HOR CALCULATING TRANSFER FUNCTION ZEROES)
END

```

```
C-----  
SUBROUTINE ACOMP (N, NN, A, B, C, H)  
REAL*8 A, B, C, H  
DIMENSION A(NN, N), B(N), C(N)  
DO 10 I=1, N  
DO 10 J=1, N  
10 A(I, J) = A(I, J) - B(I) * C(J) / H  
RETURN  
END
```

```
C-----  
SUBROUTINE CCCMP (N, NM, A, C, CC)  
REAL*8 A, C, CC  
DIMENSION A(NM, N), C(N), CC(N)  
DO 10 I=1, N  
  CC(I)=0.  
DO 10 J=1, N  
  CC(I)=CC(I)+C(J)*A(J, I)  
DO 20 I=1, N  
  C(I)=CC(I)  
RETURN  
END
```

```
C-----  
FUNCTION SCL (N,J,C)  
REAL*8 B,C,SCL  
DIMENSION B(N),C(N)  
SCL=0.  
DO 10 I=1,N  
10 SCL=SCL+C(I)*B(I)  
RETURN  
END
```

```

C-----
SUBROUTINE RESID (K1,K2,N,JCF,R,BM,L,CM,PS,PI,RES,BB,CC,IT)
IMPLICIT REAL*8(A-H,C-3)
DIMENSION JCF(N),EM(N,N),CM(L,N),PR(N),PI(N),RES(N),BB(N),CC(N),PS
1T(4)
DATA SN/8E*SIN(B*T/R1/8H /*,R2/8HEXP(A*T)/,2D/1H)/
DATA ZERC/0.0D0/,T1/4H*T**/,BLANK/8H /*,CS/8H*COS(2*T/
C-----TEMPORARY MOD TILL JCF IS CALCULATED-----
10 DO 10 I=1,N
JCF(I)=0
C-----TEMPORARY MOD-----
IF (IPT .EQ. 1) WRITE (6,170)
DO 20 I=1,N
20 BB(I)=BM(I,K1)
CC(I)=CM(K2,I)
C-----LOOP THROUGH THE POLES-----
30 I=0
I=I+1
IF (I .GT. N) GO TO 160
IF (JCF(I) .EQ. 1) GO TO 60
IF (DABS(PR(I)) .LT. 1.D-10) GO TO 50
C-----COMPUTE SIMPLE COMPLEX POLE RESIDUES AND PRINT BOTH-----
RES(I)=CC(I)*BB(I)+CC(I+1)*BB(I+1)
RES(I+1)=CC(I)*BB(I+1)+CC(I+1)*BB(I)
IF (IPT .EQ. 0) GO TO 40
PRT(1)=BLANK
PRT(2)=R2
IF (PI(I) .EQ. 0.0D0) PRT(2)=BLANK
PRT(3)=CS
PRT(4)=ED
WRITE (6,180) PR(I),PI(I),RES(I),(PRT(J),J=1,4)
I=I+1
PRT(3)=SN
WRITE (6,130) PR(I),PI(I),RES(I),(PRT(J),J=1,4)
GO TO 30
40 I=I+1
GO TO 30
50 CONTINUE
C-----COMPUTE SIMPLE REAL POLE RESIDUE-----
RES(I)=CC(I)*BB(I)
IF (IPT .EQ. 0) GO TO 30
PRT(1)=R1
PRT(2)=R2
PRT(3)=BLANK
PRT(4)=BLANK
WRITE (6,180) PR(I),PI(I),RES(I),(PRT(J),J=1,4)
GO TO 30
C-----LOOK AHEAD TO DETERMINE SIZE OF THE JORDAN BLOCK-----
60 K=1
KT=N-I
DO 70 J=I,KT
IF (JCF(J) .EQ. 0) GO TO 80
70 K=K+1
80 CONTINUE
IF (DABS(PI(I)) .LT. 1.D-10) GO TO 110
C-----COMPUTE REPEATED COMPLEX POLE AND PRINT OUT ALL FOUR-----
K=1
RES(I)=CC(I)*BB(I)+CC(I+1)*BB(I+1)+CC(I+2)*BB(I+2)+CC(I+3)*BB(I+3)
RES(I+1)=CC(I)*BB(I+1)+CC(I+1)*BB(I+2)+CC(I+2)*BB(I+3)+CC(I+3)*BB(I+4)
RES(I+2)=CC(I)*BB(I+2)+CC(I+1)*BB(I+3)+CC(I+2)*BB(I+4)
RES(I+3)=CC(I)*BB(I+3)+CC(I+1)*BB(I+4)
IF (IPT .EQ. 0) GO TO 100
PRT(1)=R1
PRT(2)=R2
IF (DABS(PR(I)) .GT. 1.D-10) GO TO 90
PRT(1)=BLANK
PRT(2)=BLANK
90 PRT(3)=CS
PRT(4)=ED
WRITE (6,180) PR(I),PI(I),RES(I),(PRT(J),J=1,4)
PRT(3)=SN
I=I+1
WRITE (6,180) PR(I),PI(I),RES(I),(PRT(J),J=1,4)
PRT(1)=T1
PRT(2)=R2

```



```

IF (DABS (PR(I)) .LT. 1.D-10) PRT(2) = BLANK
PRT(3) = CS
I = I + 1
WRITE (6,190) PR(I),PI(I),RES(I),PRT(1),K,(PRT(J),J=2,4)
PRT(3) = SM
I = I + 1
WRITE (6,190) PR(I),PI(I),RES(I),PRT(1),K,(PRT(J),J=2,4)
GC TO 30
100 I = I + 3
GC TO 30
C-----COMPUTE REPEATED REAL POLE RESIDUE AND PRINT OUT ALL K OF THEM-----
110 CONTINUE
KT = I + K - 1
NN = 0
DO 130 J = I,KT
NN = NN + 1
RES(J) = ZERO
DO 120 JJ = J,KT
120 RES(J) = RES(J) + EB(JJ) * CC(JJ - NN + 1)
130 CONTINUE
IF (IPT .EQ. 0) GC TO 150
NN = 0
PRT(1) = T1
PRT(2) = R2
PRT(3) = BLANK
PRT(4) = BLANK
DO 140 J = I,KT
140 WRITE (6,190) PR(J),PI(J),RES(J),PRT(1),NN,(PRT(JJ),JJ=2,4)
NN = NN + 1
GC TO 30
150 I = KT
GC TO 30
160 CONTINUE
RETURN
C-----
170 FORMAT (//,3X,22RESIDUES AT THE POLES: /,T16,3HP O L E S,T41,15HR
1E S I D U S S //,19,7HREAL (A),126,7HIMAG (B))
180 FORMAT (//,4X,1H(,F13.6,4H)+J(,F13.6,1H),4X,1H(,F13.6,1H),3A8,A1)
190 FORMAT (//,4X,1H(,F13.6,4H)+J(,F13.6,1H),4X,1H(,F13.6,1H),A8,12,2X,
12A8,A1)
END

```

```

C-----
SUBROUTINE BALANC (NM,N,A,LOW,IGH,SCALE)
INTEGER I,J,K,L,N,N,JJ,MM,IGH,LCW,IEXC
REAL*8 A(NM,N),SCALE(N)
REAL*8 C,E,G,R,S,E2,RADIX
REAL*8 DABS
LOGICAL NCONV
DATA RADIX/Z4210000000000000/
C-----
B2=RADIX*RADIX
K=1
L=N
GC TO 60
C----- IN-LINE PROCEDURE FOR ROW AND COLUMN EXCHANGE-----
10 SCALE(M)=J
   IF (J .EQ. M) GO TO 40
   DO 20 I=1,L
     F=A(I,J)
     A(I,J)=A(I,M)
     A(I,M)=F
20   CONTINUE
     DC 30 I=K,N
     F=A(J,I)
     A(J,I)=A(N,I)
     A(N,I)=F
30   CONTINUE
40   GO TO (50,90), IEXC
C----- SEARCH FOR ROWS ISOLATING AN EIGENVALUE AND PUSH THEM DOWN-----
50   IF (L .EQ. 1) GO TO 230
     L=L-1
60   DO 80 JJ=1,L
     J=L+1-JJ
     DO 70 I=1,L
     IF (I .EQ. J) GC TO 70
     IF (A(J,I) .NE. 0.000) GO TO 80
70   CONTINUE
     H=L
     IEXC=1
     GO TO 10
80   CONTINUE
     GO TO 100
C----- SEARCH FOR COLUMNS ISOLATING AN EIGENVALUE AND PUSH THEM LEFT---
90   K=K+1
100  DO 120 J=K,L
     DO 110 I=K,L
     IF (I .EQ. J) GC TO 110
     IF (A(I,J) .NE. 0.000) GO TO 120
110  CONTINUE
     H=K
     IEXC=2
     GO TO 10
120  CONTINUE
C----- NOW BALANCE THE SUBMATRIX IN ROWS K TO L-----
DO 130 I=K,L
130  SCALE(I)=1.000
C----- ITERATIVE LOOP FOR NORM REDUCTION-----
140  NCONV=.FALSE.
     DO 220 I=R,L
     C=0.000
     R=0.000
     DO 150 J=K,L
     IF (J .EQ. I) GO TO 150
     C=C+DABS(A(J,I))
     R=R+DABS(A(I,J))
150  CONTINUE
C----- GUARD AGAINST ZERO C OR R DUE TO UNDERFLOW-----
     IF (C .EQ. 0.000 .OR. R .EQ. 0.000) GO TO 220
     G=R/RADIX
     F=1.000
     S=C*R
160  IF (C .GE. G) GO TO 170
     F=F/RADIX
     C=C*B2
     GC TO 160
170  G=R/RADIX
180  IF (C .LT. G) GO TO 190

```

```

F=F/RADIX
C=C/B2
GO TO 180
-----NOW BALANCE-----
190 IF ((C + B) / F .GE. 0.95E0 * S) GO TO 220
G=1.0D0/F
SCALE(I)=SCALE(I)*F
NOCONV=.TRUE.
DC 200 J=K,N
200 A(I,J)=A(I,J)*G
DC 210 J=1,L
210 A(J,I)=A(J,I)*F
220 CCNTINUE
IF (NOCONV) GO TO 140
230 LC=K
IGH=L
RETURN
END

```

```

C-----
SUBROUTINE ORTEES (NM,N,LOW,IGH,A,ORT)
INTEGER I,M,N,II,JJ,LA,MP,NE,IGH,KF1,LOW
REAL*8 A(NM,N),CRT(IGH)
REAL*8 F,G,H,SCALE
REAL*8 DSCRT,DAES,CSIGN
LA=IGH-1
KF1=LOW+1
IF (LA -LT. KF1) GO TO 100
DC 30 M=KF1,LA
H=0.0D0
CET(M)=0.0D0
SCALE=C.CEO
C-----SCALE CCOLUMN (ALGOL TOL THEN NOT NEEDED)-----
DO 10 I=M,IGH
SCALE=SCALE+DAES(A(I,M-1))
IF (SCALE .EQ. 0.0D0) GO TO 90
MF=M+IGH
DO 20 II=M,IGH
I=MP-II
ORT(I)=A(I,M-1)/SCALE
H=H+ORT(I)*ORT(I)
20 CONTINUE
G=-DSIGN(ESQRT(H),ORT(M))
H=H-ORT(M)*G
ORT(M)=ORT(M)-G
C-----FORM (I-(U*UT)/H) * A-----
DO 50 J=M,N
F=0.0D0
DO 30 II=M,IGH
I=MP-II
F=F+ORT(I)*A(I,J)
30 CONTINUE
F=F/H
DC 40 I=M,IGH
A(I,J)=A(I,J)-F*CET(I)
50 CONTINUE
C-----FORM (I-(U*UT)/H)*A*(I-(U*UT)/H)-----
DC 80 I=1,IGH
F=C.0D0
DC 60 JJ=M,IGH
J=MP-JJ
F=F+ORT(J)*A(I,J)
60 CONTINUE
F=F/H
DO 70 J=M,IGH
A(I,J)=A(I,J)-F*ORT(J)
70 CONTINUE
80 CONTINUE
ORT(M)=SCALE*CET(M)
A(M,M-1)=SCALE*G
90 CONTINUE
100 RETURN
ENC

```

```

C-----
SUBROUTINE ORTBAN (NM,N,LCW,IGH,A,ORT,Z)
INTEGER I,J,KL,ME,MP,MN,IGH,IGJ,MP1
REAL*8 A(NM,IGH),ORT(IGH),Z(NM,N)
REAL*8 G
C-----INITIALIZE Z TO IDENTITY MATRIX-----
DO 20 I=1,N
DO 10 J=1,N
10 Z(I,J)=0.0D0
Z(I,I)=1.0D0
20 CONTINUE
KL=IGH-LCW-1
IF (KL.LT.1) GO TO 80
DO 70 MN=1,KL
MP=IGH-MN
IF (A(MP,MP-1).EQ.0.0D0) GO TO 70
MP1=MP+1
DO 30 I=MP1,IGH
30 ORT(I)=A(I,MP-1)
DO 60 J=MP,IGH
G=0.0D0
DO 40 I=MP,IGH
40 G=G+ORT(I)*Z(I,J)
C-----DIVISOR BELOW IS NEGATIVE OF H FORMED IN ORTBAN.-----
C-----DOUBLE DIVISION AVOIDS POSSIBLE UNDERFLOW-----
G=(G / ORT(MP))/A(MP,MP-1)
DO 50 I=MP,IGH
50 Z(I,J)=Z(I,J)+G*ORT(I)
60 CONTINUE
70 CONTINUE
80 RETURN
ENC

```

```

C-----
SUBROUTINE HQB2 (NM,N,LOW,IGH,H,WR,WI,Z,IERR)
INTEGER I,J,K,L,M,N,EN,II,JJ,LL,MM,NA,NN,NN,IGH,ITS,LOW,MP2,ENM2,I
1 EBB
REAL*8 H(NM,N),WR(N),WI(N),Z(NM,N)
REAL*8 P,Q,R,S,T,U,V,Y,RA,SA,VI,VR,ZI,NORM,MACHEP
REAL*8 DSCAT,LABS,DSIGN
INTEGER MIND
LOGICAL NCTLAS
COMPLEX *16ZJ
COMPLEX *16DCSELX
REAL*8 DREAL,DIAG
C----- STATEMENT FUNCTIONS ENABLE EXTRACTION OF REAL AND IMAGINARY--
C----- PARTS OF DOUBLE PRECISION COMPLEX NUMBERS-----
DREAL(ZJ)=ZJ
DIAG(ZJ)=(0.5DC -1.0D0)*ZJ
DATA MACHEP/2.3416CC0C0000CC000/
IERR=0
NORM=0.5DC
K=1
C----- STORE FCOTS ISOLATED BY BALANC AND COMPUTE MATRIX NORM-----
DO 20 I=1,N
DC 10 J=K,N
10 NORM=NORM+DABS(H(I,J))
K=I
IF (T .GE. LOW .AND. I .LE. IGH) GO TO 20
WR(I)=H(I,I)
20 WI(I)=C.CD0
CCONTINUE
EN=IGH
T=0.0D0
C----- SEARCH FOR NEXT EIGENVALUES-----
30 IF (EN .IT. LOW) GO TO 290
ITS=0
NA=EN-1
ENM2=NA-1
C----- LOOK FOR SINGLE SMALL SUB-DIAGONAL ELEMENT-----
40 DO 50 LL=LOW,EN
L=EN+LOW-LL
IF (L .EC. LOW) GC TC 60
S=DABS(H(L-1,L-1))+DABS(H(L,L))
IF (S .EC. 0.5DC) S=NORM
IF (DABS(H(L,L-1)) .LE. MACHEP * S) GO TO 60
50 CONTINUE
C----- FORM SHIFT-----
60 X=H(EN,EN)
IF (L .EC. EN) GO TO 220
Y=H(NA,NA)
W=H(EN,NA)*H(NA,EN)
IF (L .EC. NA) GO TO 230
IF (ITS .EQ. 30) GO TO 500
IF (ITS .NE. 1C .AND. ITS .NE. 20) GC TC 90
C----- FORM EXCEPTIONAL SHIFT-----
T=T+X
70 DO 70 I=LOW,EN
H(I,I)=H(I,I)-X
S=CABS(H(EN,NA))+DABS(H(NA,ENM2))
X=0.75D0*S
Y=X
W=-0.4375D0*S*S
80 ITS=ITS+1
C----- LOOK FOR TWO CONSECUTIVE SMALL SUB-DIAGONAL ELEMENTS.-----
DO 90 MM=L,ENM2
M=ENM2+L-MM
ZZ=H(M,M)
R=X-ZZ
S=Y-ZZ
P=(R * S - W)/H(M+1,M)+H(M,M+1)
Q=H(M+1,M+1)-ZZ-R-S
H=H(M+2,M+1)
S=CABS(P)+DABS(Q)+DABS(R)
P=P/S
Q=Q/S
H=H/S
IF (H .EC. L) GO TO 100
IF (DABS(H(M,M-1)) * (DABS(Q) + DABS(R)) .LE. MACHEP * DABS(P)

```

```

1 * (DABS(H(N-1,N-1)) + DABS(ZZ) + DAES(H(N+1,N+1))) GO TO 100
90  CCNTINUE
100 MP2=M+2
    DC 110 I=MP2,EN
    H(I,I-2)=C.000
    IF (I .EQ. MP2) GO TO 110
    R(I,I-3)=C.000
110  CCNTINUE
C-----DOUBLE OR STEP INVOLVING RCWS L TO EN AND COLUMNS M TO EN-----
    DC 210 K=M,NA
    NOTLAS=K.NE.NA
    IF (K .EQ. M) GO TO 120
    P=H(K,K-1)
    Q=H(K+1,K-1)
    R=C.000
    IF (NOTLAS) R=H(K+2,K-1)
    X=DABS(P)+DABS(Q)+DAES(R)
    IF (X .EQ. 0.000) GO TO 210
    P=P/X
    Q=Q/X
    R=R/X
120  S=CSIGN(ESQRT(F*P+Q*C+R*R),P)
    IF (K .EQ. M) GO TO 130
    H(K,K-1)=-S*X
    GO TO 140
130  IF (L .NE. M) H(K,K-1)=-H(K,K-1)
140  P=P+S
    X=F/S
    Y=Q/S
    ZZ=R/S
    Q=Q/P
    R=R/P
C-----RCW MODIFICATION-----
    DC 160 J=K,N
    P=H(K,J)+C*H(K+1,J)
    IF (.NOT. NOTLAS) GO TO 150
    P=P+R*H(K+2,J)
    H(K+2,J)=H(K+2,J)-P*ZZ
150  H(K+1,J)=H(K+1,J)-P*Y
    H(K,J)=H(K,J)-P*X
160  CONTINUE
    J=MINO(EN,K+3)
C-----COLUMN MODIFICATION-----
    DO 180 I=1,J
    F=X*H(I,K)+Y*H(I,K+1)
    IF (.NOT. NOTLAS) GO TO 170
    P=P+ZZ*H(I,K+2)
    H(I,K+2)=H(I,K+2)-P*R
170  H(I,K+1)=H(I,K+1)-P*Q
    H(I,K)=H(I,K)-F
180  CONTINUE
C-----ACCUMULATE TRANSFORMATIONS-----
    DO 200 I=LOW,IGH
    P=X*Z(I,K)+Y*Z(I,K+1)
    IF (.NOT. NOTLAS) GO TO 190
    P=P+ZZ*Z(I,K+2)
    Z(I,K+2)=Z(I,K+2)-P*R
190  Z(I,K+1)=Z(I,K+1)-P*Q
    Z(I,K)=Z(I,K)-F
200  CONTINUE
210  CONTINUE
    GO TO 40
C-----ONE RCCT FOUND-----
220  H(EN,EN)=X+T
    W(EN)=H(EN,EN)
    H(EN)=C.000
    EN=NA
    GO TO 30
C-----TWO RCCTS FOUND-----
230  P=(Y - X)/2.000
    Q=P*P+W
    ZZ=DSORT(DABS(C))
    H(EN,EN)=X+T
    X=H(EN,EN)
    H(NA,NA)=Y+T
    IF (Q .LT. 0.000) GO TO 270

```

```

C-----REAL PAIR-----
ZZ=P+DSIGN(ZZ,E)
WR(NA)=X+ZZ
WR(EN)=WR(NA)
IF (ZZ .NE. 0.000) WR(EN)=X-W/ZZ
WI(NA)=0.000
WI(EN)=0.000
X=H(EN,NA)
S=DABS(X)+DABS(ZZ)
P=X/S
Q=ZZ/S
R=ESQRT(P*P+Q*Q)
P=P/R
C=C/R
C-----ROW MODIFICATION-----
DC 240 J=NA,N
ZZ=H(NA,J)
H(NA,J)=C*ZZ+P*H(EN,J)
H(EN,J)=C*H(EN,J)-P*ZZ
240 CONTINUE
C-----COLUMN MODIFICATION-----
DO 250 I=1,EN
ZZ=H(I,NA)
H(I,NA)=C*ZZ+P*H(I,EN)
H(I,EN)=C*H(I,EN)-P*ZZ
250 CONTINUE
C-----ACCUMULATE TRANSFORMATIONS-----
DC 260 I=LOW,IGH
ZZ=Z(I,NA)
Z(I,NA)=C*ZZ+P*Z(I,EN)
Z(I,EN)=C*Z(I,EN)-P*ZZ
260 CONTINUE
GC TO 280
C-----COMPLEX PAIR-----
270 WR(NA)=X+P
WR(EN)=X+P
WI(NA)=ZZ
WI(EN)=-ZZ
280 EN=EN+2
GO TO 30
C-----ALL ROOTS FOUND. BACKSUBSTITUTE TO FIND-----
C-----VECTORS OF UPPER TRIANGULAR FORM-----
290 IF (NORM .EQ. 0.010) GO TO 510
DO 450 NN=1,N
EN=N+1-NN
P=WR(EN)
Q=WI(EN)
NA=EN-1
IF (Q) 370,300,450
C-----REAL VECTOR-----
300 H=EN
H(EN,EN)=1.000
IF (NA .EQ. 0) GO TO 450
DO 360 II=1,NA
I=EN-II
W=H(I,I)-P
R=H(I,EN)
IF (W .GT. NA) GO TO 320
DO 310 J=H,NA
310 R=R+H(I,J)*H(J,EN)
320 IF (WI(I) .GE. 0.000) GO TO 330
ZZ=W
S=R
GO TO 360
330 H=I
IF (WI(I) .NE. 0.000) GO TO 340
T=W
IF (W .EQ. 0.000) T=HACHEP*NOEF
H(I,EN)=-R/T
GO TO 360
C-----SOLVE REAL EQUATIONS-----
340 X=H(I,I+1)
Y=H(I+1,I)
Q=(WR(I)-P)*(WR(I)-P)+WI(I)*WI(I)
T=(X*S-ZZ*A)/Q
H(I,EN)=T

```



```

IF (DABS(X) .LE. DABS(ZZ)) GO TO 350
H(I+1, EN) = (-R - W * T) / X
GC TO 360
350 H(I+1, EN) = (-S - Y * T) / ZZ
360 CONTINUE
-----END REAL VECTOR-----
GC TO 450
-----COMPLEX VECTOR-----
370 H=NA
-----LAST VECTOR COMPONENT CHOSEN IMAGINARY SO THAT-----
-----EIGENVECTOR MATRIX IS TRIANGULAR-----
IF (DABS(H(EN, NA)) .LE. DABS(H(NA, EN))) GO TO 380
H(NA, NA) = C / H(EN, NA)
H(NA, EN) = -(H(EN, EN) - P) / H(EN, NA)
GC TO 390
380 Z3 = DCMPLEX(0.000, -H(NA, EN) / DCMPLEX(H(NA, NA) - P, J))
H(NA, NA) = CREAL(Z3)
H(NA, EN) = DIMAG(Z3)
390 H(EN, NA) = C.000
H(EN, EN) = 1.000
ENM2 = NA - 1
IF (ENM2 .EQ. 0) GO TO 450
DO 440 II=1, ENM2
I = NA - II
W = H(I, I) - P
RA = 0.000
SA = H(I, EN)
DO 400 J = H, NA
RA = RA + H(I, J) * H(J, NA)
SA = SA + H(I, J) * H(J, EN)
400 CONTINUE
IF (4I(I) .GE. 0.000) GO TO 410
ZZ = W
H = RA
S = SA
GO TO 440
410 W = I
IF (WI(I) .NE. C.000) GO TO 420
Z3 = DCMPLEX(-RA, -SA) / DCMPLEX(W, Q)
H(I, NA) = CREAL(Z3)
H(I, EN) = CIMAG(Z3)
GO TO 440
-----SOLVE COMPLEX EQUATIONS-----
420 X = H(I, I+1)
Y = H(I+1, I)
VR = (WR(I) - P) * (WR(I) - P) * WI(I) * WI(I) - Q * Q
VI = (WR(I) - P) * 2.000 * Q
IF (VR .EQ. 0.000 .AND. VI .EQ. 0.000) VR = MACHEP * NORM * (DABS(W) + D
1ABS(Q) + DABS(X) + DABS(Y) + DABS(ZZ))
Z3 = DCMPLEX(X * VR - Z2 * RA + C * SA, X * S - Z2 * SA - Q * RA) / DCMPLEX(VR, VI)
H(I, NA) = CREAL(Z3)
H(I, EN) = CIMAG(Z3)
IF (DABS(Y) .LE. DABS(ZZ) + DABS(Q)) GO TO 430
H(I+1, NA) = (-RA - W * H(I, NA) + Q * H(I, EN)) / X
H(I+1, EN) = (-SA - W * H(I, EN) - Q * H(I, NA)) / X
GC TO 440
430 Z3 = DCMPLEX(-R - Y * H(I, NA), -S - Y * H(I, EN)) / DCMPLEX(ZZ, Q)
H(I+1, NA) = CREAL(Z3)
H(I+1, EN) = DIMAG(Z3)
440 CONTINUE
-----END COMPLEX VECTOR-----
450 CONTINUE
-----END EACH SUBSTITUTION. VECTORS OF ISOLATED ROOTS-----
DO 470 I = 1, N
IF (I .GE. LOW .AND. I .LE. IGH) GO TO 470
DO 460 J = I, N
460 Z(I, J) = H(I, J)
470 CONTINUE
-----MULTIPLY BY TRANSFORMATION MATRIX TO GIVE-----
-----VECTORS OF ORIGINAL FULL MATRIX.-----
DO 490 JJ = LOW, N
J = N - LOW - JJ
M = SINQ(J, IGH)
DO 490 I = LOW, IGH
ZZ = 0.000
DO 480 K = LOW, N

```

```
480  ZZ=ZZ+Z(I,K)*H(K,J)
      Z(I,J)=ZZ
490  CONTINUE
      GO TO 510
C-----SET ERROR --->NO CONVERGENCE TO AN-----
C-----EIGENVALUE AFTER 30 ITERATIONS-----
500  IERR=EN
510  RETURN
      END
```

```

C-----
SUBROUTINE BALEAK (N,N,LCW,IGH,SCALE,N,Z)
INTEGER I,J,K,L,N,II,MM,IGH,LCW
REAL*8 SCALE(N),Z(NM,N),S
IF (N.EC. 0) GO TO 60
IF (IGH.EC. LCW) GO TO 30
DO 20 I=LCW,IGH
S=SCALE(I)
C-----LEFT HAND EIGENVECTORS ARE BACK TRANSFORMED-----
C-----IF THE FOREGOING STATEMENT IS REPLACED BY-----
C-----S=1.00C/SCALE(I).-----
DO 10 J=1,M
10 Z(I,J)=Z(I,J)*S
20 CONTINUE
30 DO 50 II=1,N
I=II
IF (I.GE. LOW .AND. I.LE. IGH) GO TO 50
IF (I.LT. LOW) I=LCW-II
K=SCALE(I)
IF (K.EC. 1) GO TO 50
DO 40 J=1,N
S=Z(I,J)
Z(I,J)=Z(K,J)
Z(K,J)=S
40 CONTINUE
50 CONTINUE
60 RETURN
END

```

```

-----
SUBROUTINE HCR (NM,N,LOW,IGH,H,HP,WI,IERR)
INTEGER I,J,K,L,M,N,EN,LL,LM,NA,UM,IGH,ITS,LOW,MP2,ENM2,IERR
REAL*8 H(NM,N),WB(N),WI(N)
REAL*8 P,C,S,T,W,X,Y,Z,NORM,MACHEP
REAL*8 DABS,CAES,DSIGN
INTEGER M1NO
LOGICAL MCTLAS
DATA MACHEP/Z3410C00C00000000/
IERR=0
NORM=0.000
K=1
-----
--STORE ROOTS ISCLATED BY BALANC AND COMPUTE MATRIX NORM-----
DO 20 I=1,N
DC 10 J=K,N
10  NORM=NORM+DABS(H(I,J))
   K=I
   IF (I .GE. LOW .AND. I .LE. IGH) GO TO 20
   WB(I)=H(I,I)
   WI(I)=0.000
20  CCNTINUE
   EN=IGH
   T=0.000
-----
30  --SEARCH FOR NEXT EIGENVALUES-----
   IF (EN .LT. LOW) GO TO 250
   ITS=0
   NA=EN-1
   ENM2=NA-1
-----
40  --LOOK FOR SINGLE SMALL SUB-DIAGONAL ELEMENT-----
   DO 50 LL=LOW,EN
   L=EN+LOW-LL
   IF (L .EQ. LOW) GC TC 60
   S=CABS(H(L-1,L-1))+DABS(H(L,L))
   IF (S .EQ. 0.000) S=NORM
   IF (DABS(H(L,L-1)) .LE. MACHEP * S) GO TO 60
50  CONTINUE
-----
60  --FORM SHIFT-----
   X=H(EN,EN)
   IF (L .EQ. EN) GO TO 200
   Y=H(NA,NA)
   W=H(EN,NA)*H(NA,EN)
   IF (L .EQ. NA) GC TO 210
   IF (ITS .EQ. 30) GO TO 240
   IF (ITS .NE. 10 .AND. ITS .NE. 20) GC TO 90
-----
70  --FORM EXCEPTIONAL SHIFT-----
   T=I+K
   DC 70 I=LOW,EN
   H(I,I)=H(I,I)-I
   S=CABS(H(EN,NA))+CABS(H(NA,ENM2))
   Y=0.7500*S
   V=X
   W=-0.437500*S*S
90  ITS=ITS+1
-----
100 --LOOK FOR TWO CONSECUTIVE SMALL SUB-DIAGONAL ELEMENTS.-----
   DO 90 MM=L,ENM2
   N=ENM2+L-MM
   ZZ=H(H,H)
   R=X-ZZ
   S=Y-ZZ
   P=(R * S - W)/H(M+1,M)+H(M,M+1)
   Q=H(M+1,M+1)-ZZ-R-S
   R=H(M+2,M+1)
   S=CABS(P)+CABS(Q)+CAES(R)
   P=P/S
   Q=Q/S
   R=R/S
   IF (M .EQ. L) GO TO 100
   IF (DABS(H(M,M-1)) * (DABS(Q) + DABS(R)) .LE. MACHEP * DABS(P)
1 * (DABS(H(M-1,M-1)) + DABS(ZZ) + DABS(H(M+1,M+1)))) GO TO 100
90  CONTINUE
100  MP2=M+2
   DO 110 I=MP2,EN
   H(I,I-2)=0.000
   IF (I .EQ. MP2) GC TC 110
   H(I,I-3)=0.000
110  CONTINUE

```

```

C-----DOUBLE OR STEEP INVOLVING ROWS L TO EN AND COLUMNS M TO EN-----
DC 190 K=M,NA
NOTLAS=K,NE,NA
IF (K .EQ. M) GO TO 120
P=H(K,K-1)
Q=H(K+1,K-1)
R=C,ODO
IF (NOTLAS) R=H(K+2,K-1)
X=DABS(P)+DABS(Q)+DABS(R)
IF (X .EQ. 0.010) GO TO 190
P=P/X
Q=Q/X
R=R/X
H(K,K-1)=-S*X
120 S=DSIGN(DSQRT(F*P+C*R),P)
IF (K .EQ. M) GO TO 130
H(K,K-1)=-S*X
GO TO 140
130 IF (L .NE. M) H(K,K-1)=-H(K,K-1)
140 P=P+S
X=E/S
Y=Q/S
ZZ=R/S
Q=O/P
R=R/P
C-----ROW MODIFICATION-----
DO 160 J=K,EN
P=H(K,J)+Q*H(K+1,J)
IF (.NOT. NOTLAS) GO TO 150
P=P+R*H(K+2,J)
150 H(K+2,J)=H(K+2,J)-P*ZZ
H(K+1,J)=H(K+1,J)-P*Y
H(K,J)=H(K,J)-P*X
160 CONTINUE
J=MINO(EN,K+1)
C-----COLUMN MODIFICATION-----
DO 180 I=L,J
P=X*H(I,K)+Y*H(I,K+1)
IF (.NOT. NOTLAS) GO TO 170
P=P+ZZ*H(I,K+2)
170 H(I,K+2)=H(I,K+2)-P*R
H(I,K+1)=H(I,K+1)-P*Q
H(I,K)=H(I,K)-P
180 CONTINUE
190 CONTINUE
GO TO 40
C-----ONE ROOT FOUND-----
200 WR(EN)=X+T
WI(EN)=0.0DO
EN=NA
GC TO 30
C-----TWO ROOTS FOUND-----
210 P=(Y-X)/2.0EO
Q=P*P+W
ZZ=DSQRT(DABS(Q))
X=X+T
IF (Q .LT. 0.010) GO TO 220
C-----REAL PAIR-----
ZZ=P+DSIGN(ZZ,E)
WR(NA)=X+ZZ
WR(EN)=WR(NA)
IF (ZZ .NE. 0.0DO) WR(EN)=X-W/ZZ
WI(NA)=0.0DO
WI(EN)=0.0DO
GO TO 230
C-----COMPLEX PAIR-----
220 WR(NA)=X+F
WR(EN)=X+F
WI(NA)=ZZ
WI(EN)=-ZZ
230 EN=EN+2
GO TO 30
C-----SET ERROR -- NO CONVERGENCE TO AN-----
C-----EIGENVALUE AFTER 30 ITERATIONS-----
240 IERR=24
250 RETURN
ENC

```

```

C-----
SUBROUTINE PSDCAL (N2,JS,FA,X,GC,GF,GV,C,NO,NY,HD,H,
1 FBGE,NG,GAM,ACL,F,WR,WI,D1,D2,JCF,RZS,J,R,BB,CC,IYU,
2 IPSD,INORM)
C-----
C PSDCAL COMPUTES THE PSD OF OUTPUTS OF CONTROLS OF
C A CONTROLLED SYSTEM
C-----
C IYU= 1 OUTPUT PSD
C = 2 CONTROL PSD
C = 3 BOTH OUTPUT AND CONTROL PSD
C-----
C IPSC=1 PSD
C =2 PSD AND TF RESIDUES
C-----
C INCRM= 1,2,... NG NORMALIZED BY ITH PROCESS NOISE
C NG+1,... NG+NC NORMALIZED BY ITH MEAS NOISE
C-----
DOUBLE PRECISION FA,X,GC,GF,GV,C,NO,NY,HD,H,FBGE,GAM,ACL,F,WR,WI,D1,D2,RES,
1 IE,CC,Q,R,PSD,W,LNORM,DN1,EMAX,ELOG,EMOD,DM,ST,ON,RE,AZ,BU,DM1
COMPLEX *16 ZD,ZN,ZZ
DIMENSION FA(N2,N2),X(N2,N2),GC(N2,NG),C(NC,NS),HY(NO,N2),H(NC,NS),
1 FBGE(NS,NO),GAM(NS,NG),ACL(NS,NS),F(NS,NS),WR(N2),WI(N2),D1(N2),D2
22(N2),RES(N2),C(NG,NG),R(NO,NC),PSD(30),W(30),BB(N2),CC(N2),GV(N2,
3NC),HU(NC,N2),IWI(4)
INTEGER JCF(N2)
DATA DM1/1.D0,2.D0,5.D0,10.D0/
IF (IYU .EQ. 0) IYU=1
IF (INORM .EQ. 0) INORM=1
IPT=0
IF (IPSD .GT. 1) IPT=1
IX=LNORM-NS
IF (IX .GT. 0) WRITE (6,330) IX
IF (IX .LE. 0) WRITE (6,340) INCRM
NSC=N2*N2
C----- COMPUTE EIGENSYSTEM OF CONTROLLED SYSTEM: FORM FA-----
DO 10 I=1,NS
DO 10 J=1,NS
FA(I,J)=ACL(I,J)
10 FA(NS+I,J)=0.D0
DO 30 I=1,NS
DO 30 J=1,NS
ST=0.D0
DO 20 K=1,NO
ST=ST+FBGE(I,K)*H(K,J)
20 FA(I,NS+J)=-ST
30 FA(NS+I,NS+J)=F(I,J)-ST
CALL RAPRNT (N2,N2,N2,9,FA,4,'(9(1X,1PD13.6))')
C----- DEBUG ABOVE-----
CALL BALANC (N2,N2,FA,LOW,IHIGH,D1)
CALL ORTHES (N2,N2,ICW,IHIGH,FA,D2)
CALL ORTEAN (N2,N2,LOW,IHIGH,FA,D2,X)
CALL HQR2 (N2,N2,LOW,IHIGH,FA,WR,WI,X,IERR)
IF (IERR .NE. 0) GO TO 320
CALL BALBAR (N2,N2,ICW,IHIGH,I1,N2,X)
CALL RAPRNT (N2,N2,N2,9,X,4,'(9(1X,1PD13.6))')
C----- DEBUG ABOVE: DETERMINE PSDAL MATRICES-----
IF (IYU .EQ. 1) GO TO 60
C----- HSUBU-----
DO 50 I=1,NC
DO 50 J=1,N2
ST=0.D0
DO 40 K=1,NS
ST=ST-C(I,K)*X(K,J)
40 HU(I,J)=ST
50 GO TO 90
C----- HSUBY-----
DO 80 I=1,NO
DO 80 J=1,N2
ST=0.D0
DO 70 K=1,NS
ST=ST+H(I,K)*X(K,J)-H(I,K)*X(NS+K,J)
70 HY(I,J)=ST
80 CALL RAPRNT (NC,NC,N2,9,HY,4,'(9(1X,1PD13.6))')
C----- DEBUG ABOVE-----

```

```

90  CALL MINV (NSC,X,N2,ST,D1,D2)
    CALL RAPRNT (N2,N2,N2,3,X,4,'(9(1X,1E13.6))')
C-----DEBUG ABOVE-----
C-----SUB#-----
    DO 110 I=1,N2
    DO 110 J=1,NG
    ST=0.000
    DO 100 K=1,NS
100  ST=ST-K(I,NS+K)*GAM(K,J)
110  GW(I,J)=ST
    CALL RAPRNT (N2,N2,NG,9,GW,4,'(9(1X,1E13.6))')
C-----DEBUG ABOVE: USE SELECTED NORMALIZATION-----
    IF (INORM .LE. NG) DNORM=1.00/C(INORM,INORM)
    IF (INORM .GT. NG) ENORM=1.00/E(INORM-NG,INORM-NG)
C-----DETERMINE BANDWIDTH OF CONTROLLED SYSTEM-----
    EMAX=0.00
    DO 120 I=1,N2
    EMOD=DABS(4R(I)**2 +WI(I)**2)
120  IF (EMOD .GT. EMAX) EMAX=EMOD
    CONTINUE
    EMOD=DSQRT(EMAX)
    EMOD=2*EMOD
C-----ROUND UP TO NEAREST 2,4,5,8,10-----
    ELOG=DLOG10(EMOD)
    IF (ELOG .LT. 0.00) IPOW=-IDINT(DABS(ELOG) + 1)
    IF (ELOG .GE. 0.00) IPOW=IDINT(ELOG)
    EMAX=EMOD*10**(I-IPOW)
    IF (EMAX .GT. 2.00) EMOD=2.00
    IF (EMAX .GT. 4.00) EMOD=4.00
    IF (EMAX .GT. 5.00) EMOD=5.00
    IF (EMAX .GT. 6.00) EMOD=6.00
    IF (EMAX .GT. 8.00) EMOD=8.00
    IF (EMAX .GT. 10.00) EMOD=10.00
    EMAX=EMOD*10**IFC#
    DW=EMAX/20.00
C-----ADD 10 POINTS 3 DECADES UP-----
    IF (EMOD .LT. 5.0) GC TO 130
    EMAX=1.00#
    IK=3
    GC TO 140
130  EMAX=5.00
    IK=2
140  CONTINUE
C-----STORE 30 FREQUENCIES-----
    DC 150 I=1,20
150  W(I)=DW**(I-1)
    DC 160 I=1,3
    IP=20+3*(I-1)
    DO 160 J=1,3
    IX=MOD(IK+J-1,3)+1
    JJ=0
    IF (IK .EQ. 2 .AND. J .GE. 2) JJ=1
160  W(IP+J)=DW(I)*10**(IPOW+I-1+JJ+IK-2)
    CONTINUE
    IX=MOD(IK,3)+1
    W(30)=DW(I)*10**(IPOW+3+IK-2)
C-----LARGE LOOP THRU OUTPUTS-----
    IF (IYU .EQ. 1) NI=NC
    IF (IYU .EQ. 2) NI=NC
    DO 170 L=1,NL
170  DO 170 I=1,30
    PSD(I)=0.00
C-----LOOP THRU PROCESS NOISE-----
    DO 220 I=1,NG
    DN1=DNORM*Q(I,I)
    IF (IYU .EQ. 1 .AND. IPT .EQ. 1) WRITE (6,350) I,L
    IF (IYU .EQ. 2 .AND. IPT .EQ. 1) WRITE (6,360) I,L
    IF (IYU .EQ. 1) CALL RESID (I,L,N2,JCF,NG,GW,NL,HY,WR,WI,
1  RES,BB,CC,IPT)
    IF (IYU .EQ. 2) CALL RESID (I,L,N2,JCF,NG,GW,NL,HU,WR,WI,
1  RES,BB,CC,IPT)
    DO 210 K=1,20
    ZZ=DCNPLX(0.00,0.00)
    OR=W(K)
    DO 200 II=1,N2
180  ZD=DCNPLX(-WR(II),CB-WI(II))

```

```

ZZ=RES (II) / ZD + ZZ
GC TO 200
190 RE=WR (II)
AI=WI (II)
ZD=DCMPLX (RE**2 + AI**2 - OM**2, -2.00*RE*OM)
ZN=DCMPLX (RES (II+1) * AI - RES (II) * RE, RES (II) * OM)
ZZ=ZZ + ZN / ZD
200 CCNTINUE
210 PSD (K) = PSD (K) + DN1 * (ZZ * DCONJG (ZZ))
220 CONTINUE
C-----GUBV-----
DO 240 I=1, N2
DO 240 J=1, NO
ST=0.00
DC 230 K=1, NS
230 ST=ST + X (I, K) * FEGE (K, J) + X (I, NS + K) * PBGE (K, J)
240 GV (I, J) = ST
CALL RAPENT (N2, N2, NO, 9, GV, 4, ('9(1X, 1PD13.6)'))
C-----DEBUG ABCVE, LOOP THRU MEAS NOISE-----
DO 300 I=1, NO
DN1=DNORR * R (I, I)
IF (IYU .EQ. 1 .AND. IPT .EQ. 1) WRITE (6, 370) I, L
IF (IYU .EQ. 2 .AND. IPT .EQ. 1) WRITE (6, 380) I, L
IF (IYU .EQ. 1) CALL RESID (I, L, N2, JCF, NO, GV, NL, NY, WR, WI, RES,
1 BB, CC, IPT)
IF (IYU .EQ. 2) CALL RESID (I, L, N2, JCF, NO, GV, NL, HU, WR, WI, RES,
1 BB, CC, IPT)
DO 290 K=1, 30
ZZ=DCMPLX (0.00, 0.00)
OM=W (K)
DC 270 II=1, N2
IF (WI (II)) 270, 250, 260
250 ZZ=DCMPLX (-WR (II), OM - WI (II))
ZZ=ZZ + RES (II) / ZD
GC TO 270
260 RE=WR (II)
AI=WI (II)
ZD=DCMPLX (RE**2 + AI**2 - OM**2, -2.00*RE*OM)
ZN=DCMPLX (RES (II+1) * AI - RES (II) * RE, RES (II) * OM)
ZZ=ZZ + ZN / ZD
270 CONTINUE
IF (IYU .EQ. 2 .OR. I .NE. L) GO TO 280
PSC (K) = PSC (K) + EN1
280 PSD (K) = PSD (K) + DN1 * (ZZ * DCONJG (ZZ))
290 CCNTINUE
300 CONTINUE
IF (IYU .EQ. 1) WRITE (6, 390) I
IF (IYU .EQ. 2) WRITE (6, 400) I
WRITE (6, 410) (W (I), PSD (I), I=1, 30)
310 CONTINUE
RETURN
320 CONTINUE
CALL EEXIT (N2, FA, IERR)
RETURN
C-----
330 FORMAT (/, 41H SUBSEQUENT PSD IS NORMALIZED BY MEAS NO., I3)
340 FORMAT (/, 50H SUBSEQUENT PSD IS NORMALIZED BY PROCESS NOISE NO., I3)
1)
350 PCFMT (/, 38H TRANSFER FUNCTION FROM PROCESS NOISE, I2, 3H TO, 13H ME
1) ASUREMENT, I2)
360 FORMAT (/, 38H TRANSFER FUNCTION FROM PROCESS NOISE, I2, 3H TO, 9H CCN
1) TRCL, I2)
370 FORMAT (/, 36H TRANSFER FUNCTION FROM MEASUREMENT, I2, 16H TO MEASURE
1) MENT, I2)
380 FORMAT (/, 36H TRANSFER FUNCTION FROM MEASUREMENT, I2, 12H TO CONTRCL
1) , I2)
390 FORMAT (/, 14H PSD OF CUTPUT, I3, 32H FORCED BY ALL NOISE-(RAD FREQ.,
1) 15H NORMALIZED ESC) (/)
400 FORMAT (/, 15H PSD OF CONTRCL, I3, 32H FORCED BY ALL NOISE-(RAD FREQ.,
1) 15H NORMALIZED PSC) (/)
410 FORMAT (4(1X, 1H(, E11.4, 1H., E11.4, 1H))
END

```



```

C-----
C      SUBROUTINE EREXIT (N,A,IERR)
C      EREXIT RETURNS THE NUMBER OF THE EIGENVALUE WHERE HQR2
C      FAILS, THEN STOPS THE PROGRAM.
C-----
      INTEGER IERR
      DOUBLE PRECISION A
      DIMENSION A(N,N)
      WRITE (5,10) IERR
      CALL SAPENT (N,N,N,9,A,4,'(9(1X,1PD13.6))')
      RETURN
10    FORMAT (35H FAILURE IN HQR2 ON EIGENVALUE NO. ,I3)
      END

```

```

C-----
SUBROUTINE REALE (NS, ISAF, BA)
C INTERACTIVELY ENTERS THE "P" MATRIX ELEMENT BY ELEMENT.
C-----
REAL*8 BA(NS, NS), DUM, ANSR
INTEGER I, J, K, L, IANS, ISAF
DATA IY, IYI, IZ, IZI, IYI, IZI, IYI, IZI
IF (ISAF.EQ.1) GO TO 40
WRITE (5, 130)
DO 20 I=1, NS
DO 10 J=1, NS
WRITE (5, 120) I, J
CALL RDREAL (ANSR)
BA (I, J) = ANSR
10 CONTINUE
20 CONTINUE
C-----
30 CALL PRTCMS ('CLBSC5N ')
40 CONTINUE
WRITE (5, 140)
CALL MATFET (EA, NS, NS)
50 WRITE (5, 150)
CALL RDCLEAR (IANS)
IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GC TO 60
GO TO 70
60 WRITE (5, 160)
GC TO 50
70 CONTINUE
IF (IANS.EQ.IZ) GC TO 110
IF (IANS.EQ.IY) GC TO 90
80 WRITE (5, 170)
CALL RDINT (IANS)
K=IANS
WRITE (5, 180)
CALL RDINT (IANS)
L=IANS
WRITE (5, 120) K, L
CALL RDREAL (ANSR)
DUM=ANSR
DO 100 I=1, NS
DO 90 J=1, NS
IF ((I.EQ.K).AND.(J.EQ.L)) BA (I, J)=DUM
90 CONTINUE
100 CONTINUE
GO TO 30
110 CONTINUE
CALL PRTCMS ('CLBSCRN ')
RETURN
C-----
120 FORMAT (5X, 14H THE ELEMENT P ( I2, 1H, I2, 2H) =)
130 FORMAT (//5X, 36H ENTER THE SYSTEM MATRIX "P"-MATRIX ///, 10X, 41HDIM
1 ENSTION = # STATES NS X # STATES NS )
140 FORMAT (//5X, 33H THE SYSTEM MATRIX "P"-MATRIX ///)
150 FORMAT (//5X, 54HDC YCD WISH TO CHANGE THE VALUE OF ANY MATRIX ELEM
1 ENT? //, 10X, 19H YEE "YES" OR "NO".)
160 FORMAT (1X, 51H WARNING: IMPROPER DATA ENTRY! ENTER "YES" OR "NO".)
170 FORMAT (5X, 50H ENTER THE ROW NUMBER OF THE ELEMENT TO BE CHANGED.)
180 FORMAT (5X, 53H ENTER THE COLUMN NUMBER OF THE ELEMENT TO BE CHANGED
1 )
END

```

```

-----
SUBROUTINE REACH (NO,NS,ISAH,HC)
C INTERACTIVELY ENTERS THE "HM" MATRIX MEASUREMENT SCALING MATRIX. =
-----
REAL*8 HC(NO,NS),DUM,ANSR
INTEGER IANS,I,J,K,L,ISAH
DATA IY,IYI,IYI2,IYI4/

THIS IS AN EXAMPLE OF ONE POSSIBLE METHOD OF ARRAY GENERATION
WITHIN THE PROGRAM ITSELF. FOR VERY LARGE DATA ARRAYS, THIS METHOD
MAY BE PREFERABLE TO SOME USERS OVER INTERACTIVE ENTRY OF EACH
INDIVIDUAL ELEMENT.
-----
      DO 2 I=1,11
      DO 1 J=1,92
        HC(I,J) = 0.0D+00
        HC(1,1) = 3.11520D+00
        HO(2,75) = 0.5730D+02
        HC(3,74) = 0.1000D+01
        HC(4,63) = 0.5730D+02
        HC(5,62) = 0.1000D+01
        HC(6,76) = 0.5730D+02
        HC(7,44) = 0.5730D+02
        HC(8,45) = 0.5730D+02
        HC(9,46) = 0.5730D+02
        HO(10,47) = 0.5730D+02
        HC(11,48) = 0.5730D+02
      CONTINUE
      GO TO 90
      CONTINUE
-----
      IF (ISAH.EQ.1) GO TO 40
      WRITE (5,120)
      DC 20 I=1,NO
      DO 10 J=1,NS
        WRITE (5,110) I,J
        CALL RDREAL (ANSR)
        HO(I,J)=ANSR
      CONTINUE
      CCNTINUE
-----
      CALL PRTCMS ('CL3SCRN ')
      CONTINUE
      WRITE (5,130)
      CALL MATPRT (HC,NC,NS)
      WRITE (5,140)
      CALL RDCHAR (IANS)
      IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GO TO 60
      GO TO 70
      WRITE (5,150)
      GO TO 50
      CCNTINUE
      IF (IANS.EQ.IZ) GC TO 100
      WRITE (5,160)
      CALL RDINT (IANS)
      K=IANS
      WRITE (5,170)
      CALL RDINT (IANS)
      L=IANS
      WRITE (5,110) K,L
      CALL RDREAL (ANSR)
      DUM=ANSR
      DO 90 I=1,NO
      DC 80 J=1,NS
      IF ((I.EQ.K).AND.(J.EQ.L)) HO(I,J)=DUM
      CONTINUE
      GO TO 30
      CONTINUE
      CALL PRTCMS ('CL3SCRN ')
      RETURN
-----
110 FORMAT (5X,14H THE ELEMENT H(,I2,1H,,I2,2H)=)
120 FORMAT (//5X,50ENTER THE MEASUREMENT SCALING MATRIX "HM"-MATRIX .
1,/,/,10X,47HDIMENSION = # CBSERVATIONS NO X # STATES NS )
-----

```

```
133  FORMAT (//,10X,46HTHE MEASUREMENT SCALING MATRIX "H"-MATRIX ..../
140  1/)
140  FORMAT (/5X,54HDC YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEM
141  ENT?//,10X,19HTYPE "YES" OR "NO".)
150  FORMAT (1X,51HWARNING: IMPROPER DATA ENTRY! ENTER "YES" OR "NO".)
160  FORMAT (5X,50HENTER THE ROW NUMBER OF THE ELEMENT TO BE CHANGED.)
170  FORMAT (5X,52HENTER THE COLUMN NUMBER OF THE ELEMENT TO BE CHANGED
171  1)
171  END
```

```

-----
C-----
SUBROUTINE READD (NC, NC, D)
C-----
C ENTERS THE "C" MATRIX MEASUREMENT FEED-FORWARD DIST. MATRIX . =
C-----
REAL*4 C (NO, NC), CUM, ANSR
INTEGER IANS, I, J, K, L
DATA IY, IZ, NI / 1, 2, 3 /
WRITE (5, 110)
DO 20 I=1, NO
DO 10 J=1, NC
WRITE (5, 100) I, J
CALL RDRREAL (ANSR)
D(I, J) = ANSR
10 CONTINUE
20 CONTINUE
C-----
30 CALL PRTCHS ('CLRSCRN ')
WRITE (5, 120)
CALL MATFMT (D, NC, NC)
40 WRITE (5, 130)
CALL RDCHAR (IANS)
IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GC TO 50
GC TO 60
50 WRITE (5, 140)
GC TO 40
60 CONTINUE
IF (IANS.EQ.IZ) GC TO 90
WRITE (5, 150)
CALL RDINT (IANS)
K=IANS
WRITE (5, 160)
CALL RDINT (IANS)
L=IANS
WRITE (5, 100) K, L
CALL RDRREAL (ANSR)
DUM=ANSR
DO 80 I=1, NO
DO 70 J=1, NC
IF ((I.EQ.K).AND.(J.EQ.L)) D(I, J) = DUM
70 CONTINUE
80 CONTINUE
GC TO 30
90 CONTINUE
CALL PRTCHS ('CLRSCRN ')
RETURN
C-----
100 FOMAT (5X, 14)WRITE ELEMENT D ( I2, 1H, I2, 2H) =)
110 FOMAT (//, 5X, 54)ENTER THE MEASUREMENT FEEDTHROUGH MATRIX / FEEDFOR
1WARD, //, 5X, 34)DISTRIBUTION MATRIX "D"-MATRIX //, 8X, 49)DIMENSION
2 * # OBSERVATIONS NC * # CONTROLS NC)
120 FOMAT (//, 5X, 54)THE FEEDFORWARD DISTRIBUTION MATRIX "D"-MATRIX .
1 //)
130 FOMAT (//, 5X, 54)DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEM
1ENT? //, 10X, 19)TYPE "YES" OR "NO".)
140 FOMAT (1X, 51)WARNING: IMPROPER DATA ENTRY! ENTER "YES" OR "NO".)
150 FOMAT (5X, 5)ENTER THE ROW NUMBER OF THE ELEMENT TO BE CHANGED.)
160 FOMAT (5X, 5)ENTER THE COLUMN NUMBER OF THE ELEMENT TO BE CHANGED
1.)
END

```

```

C-----
C      SUBROUTINE READG (NS,NC,ISAG,G)
C      INTERACTIVELY ENTERS THE "G" MATRIX CONTROL DISTRIBUTION MATRIX =
C-----
      REAL*8  G(NS,NC),DUM,ANSR
      INTEGER IANS,I,J,K,L,ISAG
      DATA IY,'Y',IZ,'N'
      IF (ISAG.EQ.1) GO TO 40
      WRITE (5,12J)
      DO 20 I=1,NS
      DO 10 J=1,NC
      WRITE (5,110) I,J
      CALL RDREAL (ANSR)
      G(I,J)=ANSR
10     CONTINUE
20     CONTINUE
C-----
30     CALL PRTCMS ('CLRSCRN ')
40     CONTINUE
      WRITE (5,130)
      CALL MATFET (G,NS,NC)
50     WRITE (5,140)
      CALL RDCHAR (IANS)
      IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GO TO 60
      GO TO 70
60     WRITE (5,150)
      GO TO 50
70     CONTINUE
      IF (IANS.EQ.IZ) GO TO 100
      WRITE (5,160)
      CALL RDINT (IANS)
      K=IANS
      WRITE (5,170)
      CALL RDINT (IANS)
      L=IANS
      WRITE (5,110) K,L
      CALL RDREAL (ANSR)
      DUM=ANSR
      DO 90 I=1,NS
      DO 80 J=1,NC
      IF ((I.EQ.K).AND.(J.EQ.L)) G(I,J)=DUM
80     CONTINUE
90     CONTINUE
      GO TO 30
100    CONTINUE
      CALL PRTCMS ('CLRSCRN ')
      RETURN
C-----
110   FORMAT (5X,14H THE ELEMENT G(I2,1H,I2,2H)=)
120   FORMAT (//5X,51H ENTER THE CONTROL DISTRIBUTION MATRIX "G"-MATRIX
1.//10X,43H DIMENSION = # STATES NS X # CONTROLS NC )
130   FORMAT (//,10X,47H THE CONTROL DISTRIBUTION MATRIX "G"-MATRIX ....
1//)
140   FORMAT (//5X,54H DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEM
1ENT?//10X,19H TYPE "YES" OR "NO".)
150   FORMAT (1X,51H WARNING: IMPROPER DATA ENTRY! ENTER "YES" OR "NO".)
160   FORMAT (5X,51H ENTER THE ROW NUMBER OF THE ELEMENT TO BE CHANGED.)
170   FORMAT (5X,51H ENTER THE COLUMN NUMBER OF THE ELEMENT TO BE CHANGED
1.)
      END

```

```

C-----
C      SUBROUTINE READFB (NC,NS,FBGC)
C      ENTERS THE "C" MATRIX. FEEDBACK GAIN CONTROL MATRIX.
C-----
      REAL*4 FBGC(NC,NS),DUM,ANSR
      INTEGER IANS,I,J,K
      DATA IY,'Y',IZ,'Z'
      WRITE (5,110)
      DO 20 I=1,NC
      DO 30 J=1,NS
      WRITE (5,100) I,J
      CALL RDRREAL (ANSR)
      FBGC(I,J)=ANSR
10      CONTINUE
20      CONTINUE
C-----
30      CALL PRTCHS ('CIRSCEN ')
      WRITE (5,120)
      CALL MATPRT (FBGC,NC,NS)
40      WRITE (5,130)
      CALL RDCHAR (IANS)
      IF ((IANS.EQ.IY).AND.(IANS.NE.IZ)) GC TC 50
      GC TO 60
50      WRITE (5,140)
      GC TO 40
60      CONTINUE
      IF (IANS.EQ.IZ) GC TC 30
      WRITE (5,150)
      CALL RDINT (IANS)
      K=IANS
      WRITE (5,160)
      CALL RDINT (IANS)
      L=IANS
      WRITE (5,170) K,L
      CALL RDRREAL (ANSR)
      DUM=ANSR
      DO 30 I=1,NC
      DO 70 J=1,NS
      IF ((I.EQ.K).AND.(J.EQ.L)) FBGC(I,J)=DUM
70      CONTINUE
80      CONTINUE
      GO TO 30
90      CONTINUE
      CALL PRTCHS ('CLRESCRN ')
      RETURN
C-----
100  PCBMAT (5X,14)THE ELEMENT C(, I2, 1H,, I2, 2H)=)
110  FORMAT (//,5X,52)ENTER THE FEEDBACK GAIN CONTROL MATRIX "C"-MATRIX
120  1 //,10X,44)DIMENSION = # CONTROLS NC X # STATES NS.)
120  1 //,10X,45)THE FEEDBACK GAIN CONTROL MATRIX "C"-MATRIX //
130  1)
130  1)
130  1)
140  PCBMAT (1X,51)WARNING: IMPROPER DATA ENTRY! ENTER "YES" OR "NO".)
150  1)
150  1)
160  PCBMAT (5X,53)ENTER THE COLUMN NUMBER OF THE ELEMENT TO BE CHANGED
160  1)
      END

```

```

C-----
C      SUBROUTINE REALAY (NC,AY)
C      ENTERS THE "A" MATRIX      DIAGONAL OUTPUT COST MATRIX .
C-----
      REAL*8  AY(NC,NC) ,DUM,ANSR
      INTEGER IANS,I,J,K,L
      DATA IY/'Y',IZ/'Z',I3/'/'
      WRITE (5,110)
      DO 20 I=1,NC
      DO 10 J=1,NC
      WRITE (5,100) I,J
      CALL RDREAL (ANSR)
      AY(I,J)=ANSR
10     CONTINUE
20     CONTINUE
C-----
30     CALL PRTCMS ('CLRSCN ')
      WRITE (5,120)
      CALL MATPRN (AY,NC,NC)
40     WRITE (5,130)
      CALL RDCHAR (IANS)
      IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GC TO 50
      GC TO 60
50     WRITE (5,140)
      GC TO 40
60     CONTINUE
      IF (IANS.EQ.IZ) GC TO 90
      WRITE (5,150)
      CALL RDINT (IANS)
      K=IANS
      WRITE (5,160)
      CALL RDINT (IANS)
      L=IANS
      WRITE (5,100) K,L
      CALL RDREAL (ANSR)
      DUM=ANSR
      DO 80 I=1,NC
      DO 70 J=1,NC
      IF ((I.EQ.K).AND.(J.EQ.L)) AY(I,J)=DUM
70     CONTINUE
80     CONTINUE
      GO TO 30
90     CONTINUE
      CALL PRTCMS ('CLRSCN ')
      RETURN
C-----
100    FCENAT (5X,14HT'ENTER ELEMENT A ( I2,1H, I2,2H)=)
110    FOMAT (///5X,54HT'ENTER THE OUTPUT MEASUREMENT COST MATRIX "A"-MAT
110    RIX ///5X,53HT'DIMENSION = # CESERVATIONS "NO X # OBSERVATIONS "NO
120    2)
120    FOMAT (///5X,50HT'ENTER THE OUTPUT MEASUREMENT COST MATRIX "A"-MATRIX ..
130    1)
130    FCENAT (///5X,54HT'DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEM
130    ENT? ///1X,19HT'YES" OR "NO".)
140    FOMAT (1X,51HT'WARNING: IMPROPER DATA ENTRY! ENTER "YES" OR "NO".)
150    FOMAT (5X,50HT'ENTER THE ROW NUMBER OF THE ELEMENT TO BE CHANGED.)
160    FOMAT (5X,53HT'ENTER THE COLUMN NUMBER OF THE ELEMENT TO BE CHANGED
160    1.)
      END

```



```

-----
C-----
SUBROUTINE READB (NC,3)
C-----
ENTERS THE "B" MATRIX CONTROL COST WEIGHTING MATRIX .
-----
REAL*8 B(NC,NC),DUM,ANSR
INTEGER IANS,I,J,K,L
DATA IY,'Y',IZ,'N'
WRITE (5,90)
DO 10 I=1,NC
DO 10 J=1,NC
WRITE (5,80) I,J
CALL RDREAL (ANSR)
10 B(I,J)=ANSR
-----
20 CALL PRTCMS ('CLBSCRN ')
WRITE (5,100)
CALL MATFRT (B,NC,NC)
30 WRITE (5,110)
CALL RCHAR (IANS)
IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GC TC 40
GC TO 50
40 WRITE (5,120)
GC TO 30
50 CONTINUE
IF (IANS.EQ.IZ) GC TC 70
WRITE (5,130)
CALL RDINT (IANS)
K=IANS
WRITE (5,140)
CALL RDINT (IANS)
L=IANS
WRITE (5,80) K,L
CALL RDREAL (ANSR)
DUM=ANSR
DO 60 I=1,NC
DO 60 J=1,NC
IF ((I.EQ.K).AND.(J.EQ.L)) B(I,J)=DUM
60 CONTINUE
GO TO 20
70 CONTINUE
CALL PRTCMS ('CLBSCRN ')
RETURN
-----
80 FORMAT (5X,14H THE ELEMENT B ( I2,1H, I2,2H) =)
90 FORMAT (/,5X,5H ENTER THE CONTROL COST WEIGHTING MATRIX "B"-MATRIX
1X,/,10X,45HDIMENSION = # CONTROLS NC K # CONTROLS NC )
100 FORMAT (/,10X,37H THE CONTROL COST MATRIX.....OF "B"//)
110 FORMAT (/,5X,5H DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEM
1ENT?/,10X,19HTYPE "YES" OR "NO")
120 FORMAT (1X,51HWARNING: IMPROPER DATA ENTRY! ENTER "YES" OR "NO".)
130 FORMAT (5X,5H ENTER THE ROW NUMBER OF THE ELEMENT TO BE CHANGED.)
140 FORMAT (5X,5H ENTER THE COLUMN NUMBER OF THE ELEMENT TO BE CHANGED
)
END

```

```

-----
C-----
SUBROUTINE READG2 (NS,NG,IGAM,GAM)
C-----
C ENTERS THE "GAM" MATRIX PROCESS NOISE DISTRIBUTION MATRIX . =
C-----
REAL*8 GAM(NS,NG),DUM,ANSR
INTEGER IANS,I,J,K,L,IGAM
DATA IY,'Y',IZ,'N'
IF (IGAM.EQ.1) GO TO 40
WRITE (5,120)
DO 20 I=1,NS
DO 10 J=1,NG
WRITE (5,110) I,J
CALL RDREAL (ANSR)
GLE(I,J)=ANSR
10 CONTINUE
20 CONTINUE
C-----
30 CALL FRTCMS ('CLRSCRN ')
40 CONTINUE
WRITE (5,130)
CALL MATFRT (GAM,NS,NG)
50 WRITE (5,140)
CALL RDCHAR (IANS)
IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GO TO 60
GO TO 70
60 WRITE (5,150)
GO TO 50
70 CONTINUE
IF (IANS.EQ.IZ) GO TO 100
WRITE (5,160)
CALL RDINT (IANS)
K=IANS
WRITE (5,170)
CALL RDINT (IANS)
L=IANS
WRITE (5,110) K,L
CALL RDREAL (ANSR)
DUM=ANSR
DO 90 I=1,NS
DO 80 J=1,NG
IF ((I.EQ.K).AND.(J.EQ.L)) GAM(I,J)=DUM
80 CONTINUE
90 CONTINUE
GO TO 30
100 CONTINUE
CALL FRTCMS ('CLRSCRN ')
RETURN
C-----
110 FORMAT (5X,16H THE ELEMENT GAM(I2,1H,I2,2H) =)
120 FORMAT (/5X,36H ENTER THE PROCESS NOISE DISTRIBUTION,/,5X,24H MARI
1X "GAMMA"-MATRIX,/,2X,56H DIMENSION = * STATES NS X * PROCESS
2X NOISE SOURCES NG)
130 FORMAT (//10X,37H THE PROCESS NOISE DISTRIBUTION MATRIX,/,10X,19H
1 "GAMMA"-MATRIX : : ://)
140 FORMAT (//5X,54H DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEM
1 ENT?/,10X,19H TYPE "YES" OR "NO".)
150 FORMAT (1X,51H WARNING: IMPROPER DATA ENTRY! ENTER "YES" OR "NO".)
160 FORMAT (5X,50H ENTER THE ROW NUMBER OF THE ELEMENT TO BE CHANGED.)
170 FORMAT (5X,53H ENTER THE COLUMN NUMBER OF THE ELEMENT TO BE CHANGED
1.)
END

```

```

C-----
C SUBROUTINE READQ (NG,Q)
C INTERACTIVELY ENTERS THE "Q" MATRIX NOISE WEIGHTING MATRIX =
C-----
      REAL*8 Q(NG,NG),DUM,ANSR
      INTEGER IANS,I,J,K,L
      DATA IY,'Y',IZ,'N'
      WRITE (5,110)
      DO 20 I=1,NG
      DC 10 J=1,NG
      WRITE (5,100) I,J
      CALL RDREAL (ANSR)
      Q(I,J)=ANSR
10  CONTINUE
20  CONTINUE
C-----
30  CALL FRTCMS ('CIRCEN ')
      WRITE (5,120)
      CALL MATFRT (Q,NG,NG)
40  WRITE (5,130)
      CALL RDCHAR (IANS)
      IF ((IANS.NE.IY).AND.(IANS.NE.IZ)) GC TO 50
      GC TO 60
50  WRITE (5,140)
      GC TO 40
60  CONTINUE
      IF (IANS.EQ.IZ) GC TO 90
      WRITE (5,150)
      CALL RDINT (IANS)
      K=IANS
      WRITE (5,160)
      CALL RDINT (IANS)
      L=IANS
      WRITE (5,100) K,L
      CALL RDREAL (ANSR)
      DUM=ANSR
      DO 80 I=1,NG
      DO 70 J=1,NG
      IF ((I.EC.K).AND.(J.EQ.L)) Q(I,J)=DUM
70  CONTINUE
80  CONTINUE
      GO TO 30
90  CONTINUE
      CALL FRTCMS ('CIRCEN ')
      RETURN
C-----
100  FORMAT (5X,14H THE ELEMENT Q(, IZ, JH, IZ, 2H)=)
110  FORMAT (///,5X,44H ENTER THE PROCESS NOISE PSD WEIGHTING MATRIX.,5X
1,12H "Q" MATRIX.,///,5X,42HDIMENSION = * PROCESS NOISE SOURCES NG
2,1X,17H PROCESS NOISE SOURCES NG)
120  FORMAT (///,5X,42H THE PROCESS NOISE WEIGHTING MATRIX.....Q.///)
130  FORMAT (///,5X,54H DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEM
1ENT?.,10X,19H TYPE "YES" OR "NO".)
140  FORMAT (1X,51H WARNING: IMPROPER DATA ENTRY! ENTER "YES" OR "NO".)
150  FORMAT (5X,50H ENTER THE ROW NUMBER OF THE ELEMENT TO BE CHANGED.)
160  FORMAT (5X,53H ENTER THE COLUMN NUMBER OF THE ELEMENT TO BE CHANGED
1.)
      END

```

```

C-----
C      SUBROUTINE REACR (NO,RC)
C      ENTERS THE "R" MATRIX MEASUREMENT NOISE DISTRIBUTION MATRIX . =
C-----
      REAL*4  RC(NO,NO),DUM,ANSR
      INTEGER IANS,J,K,L
      DATA IY,'Y',IZ,'N' /
      WRITE (5,90)
      DO 10 I=1,NO
      DO 10 J=1,NO
      WRITE (5,80) I,J
      CALL RDREAL (IANS)
10      RC(I,J)=ANSR
C-----
20      CALL PRTCMS ('CIRSCEN ')
      WRITE (5,100)
      CALL MATDET (RC,NC,NC)
30      WRITE (5,110)
      CALL RDCHAR (IANS)
      IF ((IANS.EQ.IY).AND.(IANS.NE.IZ)) GC TC 40
      GC TO 50
40      WRITE (5,120)
      GC TO 30
50      CONTINUE
      IF (IANS.EQ.IZ) GC TC 70
      WRITE (5,130)
      CALL RDIST (IANS)
      K=IANS
      WRITE (5,140)
      CALL RDINT (IANS)
      L=IANS
      WRITE (5,80) K,L
      CALL RDREAL (AKSR)
      DUM=ANSR
      DO 60 I=1,NO
      DO 60 J=1,NO
60      IF ((I.EQ.K).AND.(J.EQ.L)) RC(I,J)=DUM
      GO TO 20
70      CONTINUE
      CALL PRTCMS ('CIRSCEN ')
      RETURN
C-----
80      FCBHAT (5X,14WRITE ELEMENT R(I2,1H,,I2,2H)=)
90      FORMAT (//,5X,6CENTER THE MEASUREMENT NOISE DISTRIBUTION MATRIX "
1R"MATRIX //,5X,53HDIMENSION = * OBSERVATIONS NO X * OBSERVATIO
2NS NO )
100     FORMAT (//,15X,5GRTHE MEASUREMENT NOISE DISTRIBUTION MATRIX.....R.
1.....//)
110     FORMAT (//,5X,54HDC YOU WISH TC CHANGE THE VALUE OF ANY MATRIX ELEM
1ENT? //,10X,19HTYPE "YES" OR "NO".)
120     FORMAT (1X,51HWARNING: IMPROPER DATA ENTRY! ENTER "YES" OR "NO".)
130     FORMAT (5X,50HENTER THE ROW NUMBER OF THE ELEMENT TO BE CHANGED.)
140     FORMAT (5X,52HENTER THE CCOLUMN NUMBEE OF THE ELEMENT TO BE CHANGED
1)
      END

```

```

-----
C  SUBROUTINE REAPER (NS,NO,FBGE)
C  INTERACTIVELY ENTERS THE "K" FEEDBACK GAIN ESTIMATOR MATRIX
-----
      REAL*4 FBGE(NS,NC),DUM,ANSR
      INTEGER IANS,I,I2,I2H,I2L
      DATA IV,IV1,I2,I2H,I2L / 5,110,12,120,12 /
      WRITE (5,110)
      DO 20 J=1,NS
      DO 30 I=1,NO
      WRITE (5,100) I,J
      CALL RDRFAL (ANSR)
      FBGE(I,J)=ANSR
10  CONTINUE
20  CONTINUE
-----
30  CALL PRTCMS ('CLBSCFN ')
      WRITE (5,120)
      CALL MATPRT (FBGE,NS,NO)
40  WRITE (5,130)
      CALL RDCLEAR (IANS)
      IF ((IANS.NE.I1).AND.(IANS.NE.I2)) GC TO 50
      GO TO 60
50  WRITE (5,140)
      GC TO 40
60  CONTINUE
      IF (IANS.EQ.I2) GC TO 90
      WRITE (5,150)
      CALL RDRINT (IANS)
      K=IANS
      WRITE (5,160)
      CALL RDRINT (IANS)
      L=IANS
      WRITE (5,100) K,L
      CALL RDRFAL (ANSR)
      DUM=ANSR
      DO 80 I=1,NS
      DO 70 J=1,NO
      IF ((I.EQ.K).AND.(J.EQ.L)) FBGE(I,J)=DUM
70  CONTINUE
80  CONTINUE
      GO TO 30
90  CONTINUE
      CALL PRTCMS ('CLBSCFN ')
      RETURN
-----
C 100 FCBMAT (5X,14)ENTER ELEMENT K (I2,1H,I2,2H)=
110 FCBMAT (//5X,54)ENTER THE FEEDBACK GAIN ESTIMATOR MATRIX "K"-MATR
11X //,1CX,48)DIMENSION = # STATES NS X # OBSERVATIONS NO.)
120 FCBMAT (//,15X,47)THE FEEDBACK GAIN ESTIMATOR MATRIX "K"-MATRIX ,
1//)
130 FCBMAT (//,5X,54)DO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELE
1MENT? //,10X,19)TYPE "YES" OR "NO".)
140 FCBMAT (//,5X,54)WARNING: IMPROPER DATA ENTRY! ENTER "YES" OR "NO".)
150 FCBMAT (5X,53)ENTER THE ROW NUMBER OF THE ELEMENT TO BE CHANGED.)
160 FCBMAT (5X,52)ENTER THE COLUMN NUMBER OF THE ELEMENT TO BE CHANGED
1//)
      ENC

```

```

-----
C-----
SUBROUTINE REACH (NG,WR)
C INTERACTIVELY ENTERS "NO" STEADY DISTURBANCE VECTOR
C-----
REAL*8 W(NG),DUM,ANSR
INTEGER IANS,I,K
DATA IY,IY',I2,'N' /
WRITE (5,100)
DO 10 I=1,NG
WRITE (5,80) I
CALL RDREAL (ANSR)
WB(I)=ANSR
10 CONTINUE
-----
2C CALL PRTCMS ('CLSCSN ')
WRITE (5,110)
WRITE (5,90) (WB(I),I=1,NG)
30 WRITE (5,120)
CALL RDREAL (IANS)
IF ((IANS.NE.IY).AND.(IANS.NE.I2)) GC TO 40
GC TO 50
40 WRITE (5,130)
GC TO 30
50 CONTINUE
IF (IANS.EQ.I2) GC TO 70
WRITE (5,140)
CALL RDINT (IANS)
K=IANS
WRITE (5,80) K
CALL RDREAL (ANSR)
DUM=ANSR
DO 60 I=1,NG
IF (I.EQ.K) WB(I)=DUM
60 CONTINUE
GC TO 20
70 CONTINUE
CALL PRTCMS ('CLSCSN ')
RETURN
-----
80 FORMAT (5X,15HTHE ELEMENT NO(,I2,2H) =)
90 FORMAT (F12.5)
100 FORMAT (//,5X,57HENTER THE STEADY DISTURBANCE VECTOR MATRIX "WO"-M
MATRIX //,10X,44HDIMENSION = # PROCESS NOISE SOURCES (NG X 1)
110 FORMAT (//,15X,53HTHE STEADY DISTURBANCE VECTOR MATRIX "WO"-MATRI
1X,57H//)
120 FORMAT (//,5X,54HDO YOU WISH TO CHANGE THE VALUE OF ANY MATRIX ELEM
ENT? //,10X,19HTYPE "YES" OR "NO".)
130 FORMAT (1X,51HWARNING: IMPROPER DATA ENTRY! ENTER "YES" OR "NO".)
140 FORMAT (5X,50HENTER THE ROW NUMBER OF THE ELEMENT TO BE CHANGED.)
END

```

```

C-----
C SUBROUTINE RDREAL -- INTERACTIVELY READS A REAL NUMBER REPLY      =
C INTO A FORTRAN PROGRAM. IF THE USER INADVERTENTLY ENTERS A NULL  =
C STRING THE S/R ISSUES A WARNING AND ALLOWS A RECOVERY.          =
C-----
SUBROUTINE RDREAL (ANSR)
REAL *4 ANSR
INTEGER CCUNT
C-----
COUNT=0
10 CONTINUE
COUNT=COUNT+1
IF (COUNT.LT.3) GC TO 20
WRITE (5,60)
GO TO 40
20 CONTINUE
READ (5,*,END=30,ERR=30) ANSR
RETURN
30 REWIND 5
WRITE (5,50)
GC TO 10
40 CONTINUE
STOP
C-----
50 FORMAT (1X,64HWARNING: NULL STRINGS ARE NOT ALLOWED, ENTER A NUME
TRICAL VALUE.)
60 FORMAT (////,2X,42HPROGRAM KILLED - TWO NULL STRINGS ENTERED!,,)
END

```



```

C-----
C SUBROUTINE RDCHAR -- INTERACTIVELY READS A CHARACTER STRING REPLY =
C ('YES' OR 'NO') INTO A FORTRAN PROGRAM. IF THE USER INADVERTENLY =
C ENTERS A NULL STRING THE S/R ISSUES A WARNING AND ALLOWS A RECOVERY=
C-----
SUBROUTINE RDCHAR (IANS)
INTEGER COUNT, IANS
DATA IY, 'Y', IZ, 'N' /
C-----
COUNT=0
CONTINUE
10 COUNT=COUNT+1
IF (COUNT.LT.3) GC TO 20
WRITE (5,60)
GC TO 40
20 CONTINUE
REWIND 5
READ (5,70,END=30,ERR=30) IANS
RETURN
30 REWIND 5
WRITE (5,50)
GO TO 10
40 CONTINUE
STOP
C-----
50 FORMAT (1X,60HWARNING: NULL STRINGS ARE NOT ALLOWED, ENTER "YES"
1CB "NO".)
60 FORMAT (///,5X,47HPROGRAM TERMINATION - TWO NULL STRINGS ENTERED!)
70 FCBMAT (A1)
END

```

```

-----
C SUBROUTINE MATPRT -- DISPLAYS A TWO-DIMENSIONAL ARRAY (16 COLS. MAX) =
C IN VARIABLE SCREEN FORMAT FOR USER EASE IN ROW IDENTIFICATION.
-----

```

```

SUBROUTINE MATPRT (PRTT, NROW, NCOL)
IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION PRTT (NROW, NCOL)

```

```

C
IF (NCOL.EQ.0) NCOL=1
IF (NCOL.EQ.1) WRITE (5, 10) ((PRTT (I,J), J=1, NCOL), I=1, NROW)
IF (NCOL.EQ.2) WRITE (5, 20) ((PRTT (I,J), J=1, NCOL), I=1, NROW)
IF (NCOL.EQ.3) WRITE (5, 30) ((PRTT (I,J), J=1, NCOL), I=1, NROW)
IF (NCOL.EQ.4) WRITE (5, 40) ((PRTT (I,J), J=1, NCOL), I=1, NROW)
IF (NCOL.EQ.5) WRITE (5, 50) ((PRTT (I,J), J=1, NCOL), I=1, NROW)
IF (NCOL.EQ.6) WRITE (5, 60) ((PRTT (I,J), J=1, NCOL), I=1, NROW)
IF (NCOL.EQ.7) WRITE (5, 70) ((PRTT (I,J), J=1, NCOL), I=1, NROW)
IF (NCOL.EQ.8) WRITE (5, 80) ((PRTT (I,J), J=1, NCOL), I=1, NROW)
IF (NCOL.EQ.9) WRITE (5, 90) ((PRTT (I,J), J=1, NCOL), I=1, NROW)
IF (NCOL.EQ.10) WRITE (5, 100) ((PRTT (I,J), J=1, NCOL), I=1, NROW)
IF (NCOL.EQ.11) WRITE (5, 110) ((PRTT (I,J), J=1, NCOL), I=1, NROW)
IF (NCOL.EQ.12) WRITE (5, 120) ((PRTT (I,J), J=1, NCOL), I=1, NROW)
IF (NCOL.EQ.13) WRITE (5, 130) ((PRTT (I,J), J=1, NCOL), I=1, NROW)
IF (NCOL.EQ.14) WRITE (5, 140) ((PRTT (I,J), J=1, NCOL), I=1, NROW)
IF (NCOL.EQ.15) WRITE (5, 150) ((PRTT (I,J), J=1, NCOL), I=1, NROW)
IF (NCOL.EQ.16) WRITE (5, 160) ((PRTT (I,J), J=1, NCOL), I=1, NROW)

```

```

RETURN
C
10 FORMAT (F12.5)
20 FORMAT (2F12.5)
30 FORMAT (3F12.5)
40 FORMAT (4F12.5)
50 FORMAT (5F12.5)
60 FORMAT (6F12.5)
70 FORMAT (6F12.5 // F12.5 //)
80 FORMAT (6F12.5 // 2F12.5 //)
90 FORMAT (6F12.5 // 3F12.5 //)
100 FORMAT (6F12.5 // 4F12.5 //)
110 FORMAT (6F12.5 // 5F12.5 //)
120 FORMAT (6F12.5 // 6F12.5 //)
130 FORMAT (6F12.5 // 6F12.5 // F12.5 //)
140 FORMAT (6F12.5 // 6F12.5 // 2F12.5 //)
150 FORMAT (6F12.5 // 6F12.5 // 3F12.5 //)
160 FORMAT (6F12.5 // 6F12.5 // 4F12.5 //)
ENC

```

```

C-----
C  SUBROUTINE MATPRT -- DISPLAYS A TWO-DIMENSIONAL ARRAY (16 COLS. MAX) =
C  IN VARIABLE SCREEN FORMAT FOR USER EASE IN ROW IDENTIFICATION.
C-----

```

```

SUBROUTINE MATPRT (FBT, NROW, NCOL)
IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION PRTT (NROW, NCOL)

```

```

C-----
IF (NCOL.EQ.0) NCOL=1
IF (NCOL.EQ.1) WRITE(5,20) ((PRTT(I,J),J=1,NCOL),I=1,NROW)
IF (NCOL.EQ.2) WRITE(5,30) ((PRTT(I,J),J=1,NCOL),I=1,NROW)
IF (NCOL.EQ.3) WRITE(5,40) ((PRTT(I,J),J=1,NCOL),I=1,NROW)
IF (NCOL.EQ.4) WRITE(5,50) ((PRTT(I,J),J=1,NCOL),I=1,NROW)
IF (NCOL.EQ.5) WRITE(5,60) ((PRTT(I,J),J=1,NCOL),I=1,NROW)
IF (NCOL.EQ.6) WRITE(5,70) ((PRTT(I,J),J=1,NCOL),I=1,NROW)
IF (NCOL.EQ.7) WRITE(5,80) ((PRTT(I,J),J=1,NCOL),I=1,NROW)
IF (NCOL.EQ.8) WRITE(5,90) ((PRTT(I,J),J=1,NCOL),I=1,NROW)
IF (NCOL.EQ.9) WRITE(5,100) ((PRTT(I,J),J=1,NCOL),I=1,NROW)
IF (NCOL.EQ.10) WRITE(5,110) ((PRTT(I,J),J=1,NCOL),I=1,NROW)
IF (NCOL.EQ.11) WRITE(5,120) ((PRTT(I,J),J=1,NCOL),I=1,NROW)
IF (NCOL.EQ.12) WRITE(5,130) ((PRTT(I,J),J=1,NCOL),I=1,NROW)
IF (NCOL.EQ.13) WRITE(5,140) ((PRTT(I,J),J=1,NCOL),I=1,NROW)
IF (NCOL.EQ.14) WRITE(5,150) ((PRTT(I,J),J=1,NCOL),I=1,NROW)
IF (NCOL.EQ.15) WRITE(5,160) ((PRTT(I,J),J=1,NCOL),I=1,NROW)
IF (NCOL.EQ.16) WRITE(5,160) ((PRTT(I,J),J=1,NCOL),I=1,NROW)
RETURN

```

```

C-----
10  FORMAT (F12.5)
20  FORMAT (2F12.5)
30  FORMAT (3F12.5)
40  FORMAT (4F12.5)
50  FORMAT (5F12.5)
60  FORMAT (6F12.5)
70  FORMAT (6F12.5 // F12.5 //)
80  FORMAT (6F12.5 // 2F12.5 //)
90  FORMAT (6F12.5 // 3F12.5 //)
100 FORMAT (6F12.5 // 4F12.5 //)
110 FORMAT (6F12.5 // 5F12.5 //)
120 FORMAT (6F12.5 // 6F12.5 //)
130 FORMAT (6F12.5 // 6F12.5 // F12.5 //)
140 FORMAT (6F12.5 // 6F12.5 // 2F12.5 //)
150 FORMAT (6F12.5 // 6F12.5 // 3F12.5 //)
160 FORMAT (6F12.5 // 6F12.5 // 4F12.5 //)
END

```

LIST OF REFERENCES

1. Hall, W.E., Computational Methods for the Synthesis of Rotary-Wing VTCI Aircraft Control Systems, Ph.D. Dissertation, Stanford Univ., Aug. 1971.
2. Walker, R.A., User's Manual for CPTSYS 4 at SCIF, Stanford Univ., Aero/Astro Dept., Dec. 1979.
3. Liu, G., User's Manual for CPTSYS 5 at CIM, Stanford Univ., Aero/Astro Dept., Aug. 1982.
4. Bryson, A.E. and Ho, Y.C., Applied Optimal Control, Hemisphere Pub. Co., 1969, (2nd Printing, 1975).
5. Bryson, A.E., "Kalman Filter Divergence and Aircraft Motion Estimators", Jour. Guidance and Control, Vol 1, No. 1, Jan.-Feb., 1978, pp. 77-79.
6. Bryson, A.E. and Hall, W.E., Controller Synthesis for a Rotary-Wing VTCI Aircraft Near Hover, Final Report under NASA Contract NAS2-5143, SUDAAR 419, Stanford Univ., Mar. 1971.
7. Bryson, A.E. and Hall, W.E., Optimal Control and Filter Synthesis by Eigenvector Decomposition, SUDAAR 436, Stanford Univ., Dec. 1971.
8. Wilkinson, J.H. and Reinsch, C., Linear Algebra, Springer-Verlag, 1971.
9. Ogata, K., Modern Control Engineering, Prentice-Hall, 1970.

BIBLIOGRAPHY

Geib, A. and others, Applied Optimal Estimation, M.I.T. Press, 1974.

Ketter, F.I. and Prawel, S.F., Modern Methods of Engineering Computation, McGraw-Hill, 1978.

Kwakernaak, H. and Sivan, R., Linear Optimal Control Systems, Wiley-Interscience, 1972.

Lipschitz, S. and Poe, A., PROGRAMMING with FORTRAN, Schaum's Outline Series, McGraw-Hill, 1978.

Melsa, J.I. and Jones, S.K., Computer Programs for Computational Assistance in the Study of Linear Control Theory, McGraw-Hill, 1973.

Ogata, K., Modern Control Engineering, Prentice-Hall, 1970.

Research and Educational Association, Problem Solver in Automatic Control Systems/Robotics, 1982.

Sage, A.F., Optimum Control Systems, Prentice-Hall, 1968.

Wilkinson, J.H., The Algebraic Eigenvalue Problem, Clarendon Press, 1965.

INITIAL DISTRIBUTION LIST

	No. Copies
1. Defense Technical Information Center Cameron Station Alexandria Va 22314	2
2. Chairman, Code 67 Department of Aeronautics Naval Postgraduate School Monterey, Ca 93943	1
3. Library, Code 0142 Naval Postgraduate School Monterey Ca 93943	2
4. Professor D.J. Collins Code 67Co Department of Aeronautics Naval Postgraduate School Monterey, Ca 93943	5
5. CIB V.C. Gordon, Code 67 Department of Aeronautics Naval Postgraduate School Monterey, Ca 93943	1
6. Professor A.E. Bryson Department of Aeronautics and Astronautics Stanford University Stanford, Ca 94305	1
7. ICIE J.G. Haden 12979 Via Del Valador San Diego, Ca 92129	2