

## N O T I C E

THIS DOCUMENT HAS BEEN REPRODUCED FROM  
MICROFICHE. ALTHOUGH IT IS RECOGNIZED THAT  
CERTAIN PORTIONS ARE ILLEGIBLE, IT IS BEING RELEASED  
IN THE INTEREST OF MAKING AVAILABLE AS MUCH  
INFORMATION AS POSSIBLE

FINAL REPORT

NASA GRANT: NSG-1414, Suppl. 2

THE DYNAMICS AND CONTROL OF LARGE  
FLEXIBLE SPACE STRUCTURES-III

PART A: SHAPE AND ORIENTATION CONTROL OF  
A PLATFORM IN ORBIT USING POINT  
ACTUATORS

(NASA-CR-163253) THE DYNAMICS AND CONTROL  
OF LARGE FLEXIBLE SPACE STRUCTURES, 3. PART  
A: SHAPE AND ORIENTATION CONTROL OF A  
PLATFORM IN ORBIT USING POINT ACTUATORS  
Final Report (Howard Univ.) 179 p

N80-27419

HC A09/MFA01

Unclas

G3/18 28005



HOWARD UNIVERSITY  
SCHOOL OF ENGINEERING  
DEPARTMENT OF MECHANICAL ENGINEERING  
WASHINGTON, D.C. 20059

FINAL REPORT

NASA GRANT: NSG-1414, Suppl. 2

THE DYNAMICS AND CONTROL OF LARGE  
FLEXIBLE SPACE STRUCTURES-III

PART A: SHAPE AND ORIENTATION CONTROL OF  
A PLATFORM IN ORBIT USING POINT  
ACTUATORS

by

Peter M. Bainum  
Professor of Aerospace Engineering  
Principal Investigator

and

A.S.S.R. Reddy  
R. Krishna  
Paul K. James  
Graduate Research Assistants

June 1980

## ABSTRACT

The dynamics and attitude and shape control of a large thin flexible square platform in orbit are studied. Attitude and shape control is assumed to result from actuators placed perpendicular to the main surface and one edge and their effect on the rigid body and elastic modes is modelled to first order. The equations of motion are linearized about three different nominal orientations: (1) the platform following the local vertical with its major surface perpendicular to the orbital plane; (2) the platform following the local horizontal with its major surface normal to the local vertical; and (3) the platform following the local vertical with its major surface perpendicular to the orbit normal. The stability of the uncontrolled system is investigated analytically. Once controllability is established for a set of actuator locations, control law development is based on decoupling, pole placement, and linear optimal control theory. Frequencies and elastic modal shape functions are obtained using a finite element computer algorithm and two different approximate analytical methods and the results of the three methods compared.



## Table of Contents

	Page
Abstract	
I. INTRODUCTION	I-1
I.1 References - Introduction	I-6
II. CONTROL OF A LARGE FLEXIBLE PLATE IN ORBIT	II-1
II.1 Introduction - Chapter II	II-1
II.2 Model Development	II-1
II.2A Rotational Equations of Motion	II-1
II.2B Generic Mode Equations	II-2
II.2C Linearization	II-2
II.2D Modelling of Point Actuators	II-2
II.2E Modelling of Distributed Actuators	II-2
II.3 Uncontrolled Motion-Numerical Example	II-3
II.4 Controlled Motion	II-4
II.4A Decoupling of Original State Equations Using State Variable Feedback	II-4
II.4B Stabilizing the System by Pole Clustering	II-5
II.4C Application of the Linear Regulator Theory	II-6
II.5 Conclusions	II-7
II.6 References - Chapter II	II-7
Appendix A - Chapter II	II-8
III. FREQUENCIES AND MODE SHAPES FOR RECTANGULAR PLATES	III-1
III.1 Formulation by Warburton	III-1
III.2 Formulation by Lemke	III-2

III.3	Finite Element Computer Program	III-5
III.4	Discussion of Numerical Results	III-5
III.5	References - Chapter 3	III-9
IV.	GENERAL CONCLUSIONS AND RECOMMENDATIONS	IV-1
	Appendix	

## List of Tables by Chapters

Table No.	Title	Page
Chapter I Table 1	Study Plan 1977-1980	I-3
Chapter II Table 1	Maximum Force Amplitudes ( <b>Newtons</b> ) for Different Combinations of Cases with Actuator Locations	II-6
Chapter III Table 1	Evaluation of Parameters in Frequency Expression (Warburton)	III-10
Table 2	Frequencies and Nodal Patterns Obtained by the Three Methods	III-11
Table 3	Normalized Deflections at the Nodal Points Obtained by the Three Methods- Second Mode Cases 1-3	III-12
Table 4	Normalized Deflections at the Nodal Points Obtained by the Three Methods- Second Mode Case 4	III-13

## List of Figures by Chapters

Fig No.	Caption	Page
Chapter II		
Fig. 1a.	Platform following local vertical with major surface normal to the orbit plane - Case (i)	II-9
Fig. 1b.	Platform along local horizontal - Case (ii)	II-9
Fig. 1c.	Platform following local vertical with major surface in the orbit plane - Case (iii)	II-9
Fig. 2	Roll/yaw motion (uncontrolled) - Case (i)	II-9
Fig. 3	Location of two sets of actuators (I & II)	II-10
Fig. 4a.	Controlled state response for all combinations of orientations and actuator locations	II-11
Fig. 4b.	Control force time history for Case (iii) - II	II-12
Fig. 5	Variation of least damped mode negative real part with $\alpha$ and R	II-13
Fig. 6	Maximum force amplitude as a function of $\alpha$ and R for all actuators - (application of linear regulator theory).	II-14
Chapter III (Figures demonstrate nodal patterns)		
Fig. 1	First mode (1,1)	III-14
Fig. 2	Second mode (2,0)-(0,2)	III-14
Fig. 3a	(2,0) mode	III-14
Fig. 3b	(0,2) mode	III-14
Fig. 4	Third mode (2,0) + (0,2)	III-15
Fig. 5	Fourth mode (2,1)	III-15
Fig. 6	Sixth mode (3,0)	III-15
Fig. 7	Eighth (and Seventh) mode (2,2)	III-15
Fig. 8	Ninth mode (1,3) - (3,1)	III-16
Fig. 9	Tenth mode (1,3) + (3,1)	III-16

## I. INTRODUCTION

The present grant represents a continuation of the effort attempted in the previous grant years (May 1977 - May 1979) and reported in Refs. 1 - 4\*. Attitude control techniques for the pointing and stabilization of very large, inherently flexible spacecraft systems are being investigated in this research. First the attitude dynamics and control of a long, homogeneous flexible beam whose center of mass is assumed to follow a circular orbit have been treated<sup>1,2</sup>. In the initial phase, first-order effects of gravity-gradient were included, whereas external perturbations and related orbital station keeping maneuvers were ignored. Three mathematical models describing the system's rotations and deflections within the orbital plane have been developed--one model, which treats the beam as a number of discretized mass particles connected by massless links<sup>1</sup>, and two continuum-type models.<sup>2,3</sup> The natural (uncontrolled) dynamics of this system have been simulated. The concept of distributed modal control<sup>1</sup>, which provides a means for controlling a particular system mode independently of all other modes, has been examined, along with other types of control laws including an application of optimal control theory and the use of decoupling techniques.<sup>3</sup> The effect of varying the number of modes in our model as well as the number and location of control devices has been examined, analytically, where possible, and numerically for general cases.<sup>3</sup>

---

\*For references cited in this report please see list of references after each chapter.

Towards the end of the second grant year the three dimensional model of a free-free plate in orbit was developed and a limited number of computer simulations of the uncontrolled dynamics in response to initial perturbations about a specific equilibrium orientation were performed.<sup>4</sup> Frequency values associated with the basic structural modes of a square plate were obtained from energy considerations based on approximate expressions developed by Warburton.<sup>5</sup> It was suggested at the final oral grant presentation that a comparison with results obtained using finite element methods and/or other analytical approaches should be examined to guarantee accuracy, particularly for higher order modes.

With this background and in accordance with our proposal to NASA dated January 25, 1979<sup>6</sup>, a plan of study was developed and has been extended to include the current grant year as outlined in Table I. The items indicated by a check mark have been completed by the end of the third grant year while those indicated by "IP" are currently in progress.

In this part of the 1979-80 final report (Part A) the control of an orbiting square shaped platform based on the continuum model of Ref. 2 with point actuators taken at selected locations on the platform surfaces is examined. A paper to be presented at the following conference forms the basis of Chapter II:

1980 AIAA/AAS Astrodynamics Conference, Danvers, Mass.,  
Aug. 11-13, 1980 (only the contributions by A.S.S.R. Reddy,  
P.M. Bainum, and R. Krishna are included here).

In Chapter III the results of two approximate analytical methods for predicting modal frequencies and modal shape functions are compared with the results obtained using a finite element computer algorithm using the homogeneous plate as an example.

TABLE I - STUDY PLAN 1977-1980

1. MODEL DEVELOPMENT

- ✓ A. Development of General Form of 3-Dimensional Equations for A Flexible Structure - Given the Modal Shape Functions
- ✓ B. Development of 3-Dimensional Equations of a Thin Homogenous Free-Free Beam
  - ✓ (1). The Case of No Longitudinal Vibrations: i.e.  $\phi_x^{(n)} = 0$
  - ✓ (2). The Case of No Yaw: i.e.  $\psi = 0$
- C. Determination of Modal Shape Functions and Frequencies for Different Structural Models
  - ✓ (1). Circular Homogenous Membrane
  - ✓ (2). Rectangular Homogenous Membrane
  - ✓ (3). Rectangular Homogenous Plate (and Square Plate)
  - ✓ (4). Circular Homogenous Plate
  - ✓ (5). Shallow Spherical Shell Structure
- D. Implementation of One or More of the Structural Models for Digital Simulation
  - ✓ (1). Rectangular Homogenous Plate
  - ✓ (2). Thin-Homogenous Beam with Stabilizing Dumbbell (Local Horizontal Orientation)
    - (3). Square Plate with Stabilizing Dumbbell (Local Horizontal Orientation)
  - ✓ (4). Shallow Spherical Shell Structure with Stabilizing Dumbbell
  - ✓ (5). Circular Homogenous Plate with Stabilizing Dumbbell
- IP E. Provide Equations in a Form Suitable for Control Implementation
  - ✓ - Items completed
  - IP - Items in progress

SEARCH IS  
QUALITY

## 2. CONTROL CONCEPTS - LARGE FLEXIBLE SPACE STRUCTURES

### A. Model Development

- ✓ (1). Concentrated on continuum model of large flexible beam in orbit (Santini and Howard University Formulation)
- ✓ (2). Modelled control devices as point actuators at specific locations along the beam
- (3). Modelling of control devices as point or distributed actuators for other large flexible systems
  - ✓ (a) Rectangular Homogenous Plate
  - (b) Circular Homogenous Plate
  - (c) Shallow Spherical Shell Structure

### B. Control Concepts:

- ✓ (1). Modal Control - considered with discretized beam model during 1977-78

For independent control of all modes (N) retained in the model, the number of actuators (P) must be equal to N (P=N)
- ✓ (2). Establish relationship between P and N according to controllability requirements (applications of theorems developed by Balas) P can be less than N. (Applied to continuum beam model 1978-79).
- ✓ (3). Selection of control system gains - considers both position and rate feedback. (Applied to continuum beam model 1978-79)
  - ✓ a. Develop criteria for complete decoupling of linearized controlled equations using the fundamental theorem of a system of N linear equations and P unknowns

For unique solution of gains, P=N consistent with modal control; for non unique solution P<N
  - ✓ b. Application of linear regulator problem to the original linearized and/or transformed equations

IP (4) Application of control concepts to more complex structures

IP C. Modelling of Sensors-the Problem of Observability

D. Treatment of Observation and Control Spillover

✓ Items completed

IP Items in progress



References are given separately for each chapter; symbols used in Chapter II are defined either in the text or in Appendix A of Chapter II, while symbols used in Chapter III are defined in the text where used.

Chapter IV describes general conclusions together with recommendations for future work.

Part B of this report, under separate cover, concentrates on the mathematical modelling and analysis of more complex structures such as beams and plates with connected gimbaled dumbbells to provide gravitational stability about the local horizontal orientation, and also the analysis of the dynamics of a shallow shell-type structure in orbit.

## I.1 References - Introduction

1. Bainum P.M. and Sellappan, R., "The Dynamics and Control of Large Flexible Space Structures," Final Report NASA Grant: NSG-1414, Part A: Discrete Model and Modal Control, Howard University, May 1978.
2. Bainum Peter M., Kumar, V.K., and James, Paul K., "The Dynamics and Control of Large Flexible Space Structures," Final Report, NASA Grant: NSG-1414. Part B: Development of Continuum Model and Computer Simulation, Howard University, May 1978.
3. Bainum, P.M. and Reddy, A.S.S.R., "The Dynamics and Control of Large Space Structures II," Final Report, NASA Grant NSG-1414, Suppl. 1, Part A: Shape and Orientation Control Using Point Actuators, Howard University, June 1979.
4. Bainum, P.M., James P.K., Krishna, R., and Kumar, V.K., "The Dynamics and Control of Large Flexible Space Structures II" Final Report, NASA Grant NSG-1414, Suppl. 1., Part B: Model Development and Computer Simulation, Howard University, June 1979.
5. Warburton, G.B., "The Vibration of Rectangular Plates," Proc. Institute of Mechanical Engineers, Vol. 168, No. 12, 1954, pp. 371-394.
6. Bainum, P.M., "Proposal for Research Grant on: The Dynamics and Control of Large Flexible Space Structures-III," Howard University, (Submitted to NASA), Jan. 25, 1979.

## II. CONTROL OF A LARGE FLEXIBLE PLATE IN ORBIT

ORIGINAL PAGE IS  
OF POOR QUALITY

### Abstract

The dynamics and attitude and shape control of a large thin flexible platform in orbit are studied. Attitude and shape control is assumed to result from actuators placed perpendicular to the main surface and one edge and their affect on the rigid body and elastic modes is modelled to first order. The equations of motion are linearized about nominal orientations where the undeformed plate follows either the local vertical or local horizontal. The stability of the uncontrolled system is investigated analytically. Once controllability is established for a set of actuator locations, control law development is based on pole placement, decoupling, and linear optimal control theory.

### 1. Introduction

Large, flexible spacecraft systems have been proposed for future applications in widespread communications, electronic orbitally based mail systems, and as possible collectors of solar energy for transmittal to earth-based receiving stations.<sup>1,2</sup> For such missions the size of the orbiting system may be several times larger than that of the earth-based receiving station(s), and both orientation and shape control of the orbiting system will be required.

In order to gain insight into the dynamics of such a large flexible system the equations of motion of a long, flexible free-free beam in orbit were developed<sup>3</sup> using a slightly modified version of the general formulation of the dynamics of a general flexible orbiting body formulated by Santini.<sup>4</sup> This specific example considered only the inplane rotations and deformations of the uncontrolled beam and demonstrated the possibility of instability for very small values of the ratio of the fundamental flexural frequency to the orbit angular velocity. Two related papers treated the modelling of point actuators located at specific points along the beam with the associated criteria for controllability<sup>5</sup> and also the problem of selecting control law feedback gains based on decoupling techniques and application of the linear regulator problem.<sup>6</sup> Also included were numerical results showing the effects of control spillover on the uncontrolled modes when the number of controllers is less than the number of modes in the modal, and the effects of inaccurate knowl Age of

the control influence coefficients which lead to errors in the calculated feedback gains.<sup>6</sup>

In the present paper the two dimensional model considered in Refs. 3,5, and 6 is extended to three dimensions by developing the equations of motion for a large flexible rectangular plate (platform) in orbit. These equations include three rigid body equations plus the generic mode elastic equations.

### 2. Model Development

In the present paper three different nominal orientations of the platform in orbit are assumed about which attitude and shape control are to be achieved. These are:

- Case (i) the platform following the local vertical with its larger surface perpendicular to the plane of the orbit (Fig. 1a);
- Case (ii) the platform following the local horizontal with its larger surface area normal to the local vertical (Fig. 1b);
- Case (iii) the platform following the local vertical with its larger surface perpendicular to the orbit normal (Fig. 1c).

From the general formulation of Refs. 3 and 4, the equations of motion of the structure are obtained:

#### A. Rotational Equations of Motion:

$$\begin{aligned}\dot{\omega}_x &= \frac{I_y - I_z}{I_x} \omega_y \omega_z + \frac{G_x}{I_x} + \frac{T_x}{I_x} + \frac{C_x}{I_x} \\ \dot{\omega}_y &= \frac{I_z - I_x}{I_y} \omega_x \omega_z + \frac{G_y}{I_y} + \frac{T_y}{I_y} + \frac{C_y}{I_y} \\ \dot{\omega}_z &= \frac{I_x - I_y}{I_z} \omega_x \omega_y + \frac{G_z}{I_z} + \frac{T_z}{I_z} + \frac{C_z}{I_z}\end{aligned}\quad (1)$$

Using Euler angles to represent rigid body orientations relative to the local vertical (horizontal) system, the transformation from Euler angular rates to body rates is given by:

$$\begin{aligned}\omega_x &= \dot{\psi} + (\dot{\theta} + \omega_c) \sin\theta \\ \omega_y &= (\dot{\theta} + \omega_c) \cos\theta \cos\psi + \dot{\phi} \sin\psi \\ \omega_z &= \dot{\phi} \cos\psi - (\dot{\theta} + \omega_c) \sin\theta \cos\psi\end{aligned}\quad (2)$$

(Note: Symbols used are defined in Appendix A.)

$G_{R_x}$ ,  $G_{R_y}$ ,  $G_{R_z}$  represent the gravity-gradient torques about the principal undeformed body axes and can be evaluated as:

$$\begin{aligned} G_{R_x} &= 3\omega_c^2(I_x - I_y)(-c\psi s\theta c\phi + s\psi c\theta)(s\psi s\theta c\phi + c\psi s\theta) \\ G_{R_y} &= 3\omega_c^2(I_x - I_z)(s\psi s\theta c\phi + c\psi s\theta)c\theta c\phi \\ G_{R_z} &= 3\omega_c^2(I_y - I_x)(-c\psi s\theta c\phi + s\psi s\theta)c\theta c\phi \end{aligned} \quad (3)$$

where  $s(\ ) = \sin(\ )$  and  $c(\ ) = \cos(\ )$ .

### B. Generic Mode Equations

The generic modal equations may be obtained for each of the three nominal orientations considered in terms of the modal amplitude  $(A_r)^{3,4}$

For Case (I)

$$\ddot{A}_r + [\omega_r^2 - (\omega_x^2 + \omega_y^2) - M_{zz}] A_r = E_r/M_r \quad (4a)$$

where

$$M_{zz} = \omega_c^2 [3(s^2\psi s^2\theta c^2\phi + c^2\psi s^2\theta + 2s\psi s\theta c\phi c\psi\theta) - 1]$$

For Case (II)

$$\ddot{A}_r + [\omega_r^2 - (\omega_y^2 + \omega_z^2) - M_{xx}] A_r = E_r/M_r \quad (4b)$$

where

$$M_{xx} = \omega_c^2 [3c^2\theta c^2\phi - 1]$$

For Case (III)

$$\ddot{A}_r + [\omega_r^2 - (\omega_x^2 + \omega_z^2) - M_{yy}] A_r = E_r/M_r \quad (4c)$$

where

$$M_{yy} = \omega_c^2 [3(c^2\psi s^2\theta c^2\phi + s^2\psi s^2\theta - 2c\psi s\theta s\psi s\theta) - 1]$$

### C. Linearization

With the assumption of small amplitudes, the rotational equations of motion given by Eq. (1) become:

$$\begin{aligned} \ddot{\psi} &= \omega_c^2 \psi \left[ \frac{I_x - I_z}{I_x} - 1 \right] - \omega_c^2 \left( \frac{I_x - I_z}{I_x} \right) \psi + \frac{T_x}{I_x} + \frac{C_x}{I_x} \\ \ddot{\phi} &= \omega_c^2 \psi \left[ \frac{I_x - I_y}{I_x} + 1 \right] + 4\omega_c^2 \left( \frac{I_x - I_y}{I_x} \right) \psi + \frac{T_z}{I_x} + \frac{C_z}{I_x} \\ \ddot{\theta} &= 3\omega_c^2 \left( \frac{I_x - I_z}{I_y} \right) \theta + \frac{T_y}{I_y} + \frac{C_y}{I_y} \end{aligned} \quad (5)$$

For the present analysis, the platform is assumed to be square, thin and homogeneous, such that the following relationships among the principal moments of inertia are valid:

$$\begin{aligned} \text{Case (I): } & I_x = I_y \text{ and } I_z = 2I_x = 2I_y \\ \text{Case (II): } & I_y = I_z \text{ and } I_x = 2I_y = 2I_z \\ \text{Case (III): } & I_x = I_z \text{ and } I_y = 2I_x = 2I_z \end{aligned} \quad (6)$$

For small amplitude angles the generic mode equations become:

$$\begin{aligned} \text{Case (I): } & \ddot{A}_r + \omega_r^2 A_r = E_r/M_r \\ \text{Case (II): } & \ddot{A}_r + (\omega_r^2 - 3\omega_c^2) A_r = E_r/M_r \end{aligned} \quad (7)$$

$$\text{Case (III): } \ddot{A}_r + (\omega_r^2 - \omega_c^2) A_r = E_r/M_r$$

### D. Modelling of Point Actuators

For an actuator which can generate a force of the type

$$\vec{F} = f_x \hat{i} + f_y \hat{j} + f_z \hat{k} \quad (8)$$

and placed at a location  $(x, y, z)$ , the resultant control torque is given by

$$\vec{T} = \vec{r} \times \vec{F} \quad (9)$$

where  $\vec{r} = x\hat{i} + y\hat{j} + z\hat{k}$  describes the position of the actuator on the surface (or edge) of the plate. Actuators can be placed perpendicular to the XY, YZ or XZ planes of the plate, so for an actuator whose force axis is perpendicular to the XY plane the torque is given by (since  $f_x = f_y = 0$ )

$$\vec{T} = yf_z \hat{i} - zf_z \hat{j} \quad (10)$$

For an actuator whose force axis is perpendicular to the YZ plane, the torque is given by (since  $f_y = f_z = 0$ )

$$\vec{T} = zf_x \hat{j} - yf_x \hat{k} \quad (11)$$

For an actuator perpendicular to the ZX plane, the torque is given by (since  $f_z = f_x = 0$ ).

$$\vec{T} = -zf_y \hat{i} + xf_y \hat{k} \quad (12)$$

The generic force due to the  $i$ th actuator on the  $r$ th mode is given by<sup>3,4</sup>

$$\begin{aligned} E_r &= \iint W_r(x, y) \hat{k} \cdot \delta(x - x_i, y - y_i) f_i(t) \hat{k} dx dy \\ &= W_r(x_i, y_i) f_i(t) \end{aligned} \quad (13)$$

where  $W_r(x, y)$  is the  $r$ th modal (spatial) function of the deformed plate with vibrations assumed to occur along the Z direction, whose amplitudes are assumed to be much smaller than a characteristic plate length.

For  $n$  actuators placed on the XY plane of the plate with force axes normal to that deformed surface, the generic force on  $r$ th mode is given by

$$E_r = \sum_{i=1}^n W_r(x_i, y_i) f_i \quad (14)$$

where  $x_i, y_i$  are the coordinates of the  $i$ th actuator. An actuator placed normal to the X, Y plane won't produce a torque about the Z-axis; in order to obtain a direct torque about the Z-axis, actuators may have to be located on the other surfaces (edges) of the plate.

### E. Modelling of Distributed Actuators

If the force is distributed along the surfaces of the plate, the force can be represented by

$$\begin{aligned} \vec{F} &= f_x(x, y, z, t) \hat{i} + f_y(x, y, z, t) \hat{j} \\ &+ f_z(x, y, z, t) \hat{k} \end{aligned} \quad (15)$$

where the force components are now both spatially and time dependent.

The torque due to such an actuator is given by

$$\bar{T} = \bar{R}x\bar{f} \quad (16)$$

The total torque is given by

$$\bar{T} = \int_V (\bar{R}x\bar{f}) dx dy dz \quad (17)$$

Using series expansions and separation of variables between spatially and time dependent functions, one can very accurately represent (e.g. for the x component),

$$f_x(x, y, z, t) = \sum_{R=1}^L f_{xR}(x, y, z) g_{xR}(t) \quad (18)$$

The integral for the torque is then given by

$$T = \int_V \left[ \sum_{l=1}^N y f_{zl}(x, y, z) g_{zl}(t) - \sum_{m=1}^M z f_{ym}(x, y, z) g_{ym}(t) + \sum_{R=1}^L x f_{xR}(x, y, z) g_{xR}(t) - \sum_{l=1}^N x f_{zl}(x, y, z) g_{zl}(t) + \sum_{m=1}^M x f_{ym}(x, y, z) g_{ym}(t) - \sum_{R=1}^L z f_{xR}(x, y, z) g_{xR}(t) \right] dx dy dz \quad (19)$$

The resulting generic force is then obtained in the same manner as in Eq. (13) with the result,

$$E_x = \int_V [W_x(x, y)] \left[ \sum_{l=1}^N f_{zl}(x, y, z) g_{zl}(t) \right] dx dy dz \quad (20)$$

### 3. Uncontrolled Motion-Numerical Example

The platform is assumed to have the following physical properties:

$$a = 100 \text{ m (side of square plate)}$$

$$M = 276800 \text{ kg}$$

$$\text{Minimum Moment of Inertia} = 2.354 \times 10^7 \text{ kg-m}^2$$

$$\text{Maximum Moment of Inertia} = 4.7088 \times 10^7 \text{ kg-m}^2$$

For an assumed orbital altitude of 250 n.mi. (circular)

$$\omega_c = 1.25 \times 10^{-3} \text{ rad/sec.}$$

The modal frequencies of the elastic modes have been obtained using a finite element computer algorithm.<sup>7</sup> For the first three flexible modes:

$$\omega_1 = 2.0931947 \times 10^{-2} \text{ rad/sec}$$

$$\omega_2 = 3.0404741 \times 10^{-2} \text{ rad/sec}$$

$$\omega_3 = 3.9088122 \times 10^{-2} \text{ rad/sec}$$

The uncontrolled motion of the linear system through small amplitude deviations with respect to each of the three nominal orientations will now be considered.

$$\text{Case (i): } I_x = I_y, I_z = 2I_x = 2I_y$$

The rotational equations of motion and the generic modal equations are non-dimensionalized by the orbital period and the length variable ( $\tau = \omega_c t$ ,  $Z_r = A_r/a$ ,  $\phi' = d\phi/d\tau$ , etc)

$$\psi'' = [(I_y - I_x - I_z)/\omega_c I_x] \phi' - [(I_y - I_x)/I_x] \psi \quad (21)$$

$$\phi'' = [(I_x - I_y + I_z)/\omega_c I_z] \psi' + [(I_x - I_y)/I_z] \phi \quad (22)$$

$$\theta'' = 3\theta \quad (23)$$

The generic mode equations become:

$$Z_r'' = -(\omega_r/\omega_c)^2 Z_r \quad (24)$$

The pitch and the generic mode equations are decoupled from roll and yaw. The pitch and generic modes exhibit simple harmonic motions. After substituting inertia values into the roll and yaw equations,

$$\psi'' = -(2/\omega_c) \phi' + \psi \quad (25)$$

$$\phi'' = \phi'/\omega_c \quad (26)$$

The characteristic equation for the system (25) and (26) is,  $s^2(s^2 - 1 + 2/\omega_c^2) = 0$

It can be seen that the roll and yaw motion has a double pole at the origin and thus the uncontrolled roll/yaw motion is unstable. The analytical solution is obtained using Laplace transform techniques. A typical response for initial perturbations in both roll and yaw rate ( $s$ ) is shown in Fig. 2.

$$\text{Case (ii): } I_y = I_z \text{ and } I_x = 2I_y$$

The rotational equations of motion are

$$\psi'' = -(1/\omega_c) \phi' \quad (27)$$

$$\phi'' = (2/\omega_c) \psi' + \phi \quad (28)$$

$$\theta'' = 3\theta \quad (29)$$

The generic mode equations can be represented by,

$$Z_r'' = -[(\omega_r/\omega_c)^2 - 3] Z_r \quad (30)$$

From Eq. (29) the pitch amplitude increases exponentially in response to an initial displacement, whereas from Eq. (30), for  $\omega_r/\omega_c > \sqrt{3}$  the generic modal amplitudes exhibit simple harmonic motion.

The characteristic equation for the combined roll/yaw motion is:

$$s^2(s^2 - 4 + 2/\omega_c^2) = 0 \quad (31)$$

The roll/yaw motion is characterized by a double pole at the origin and is thus unstable.

$$\text{Case (iii): } I_x = I_z \text{ and } I_y = 2I_x = 2I_z$$

The rotational equations of motion are

$$\psi'' = -\psi; \phi'' = -4\phi; \theta'' = 0 \quad (32)$$

while the generic mode equations can be expressed by,

$$Z_r'' = -[(\omega_r/\omega_c)^2 - 1] Z_r \quad (33)$$

In this case, roll, yaw, pitch and the generic modes are decoupled from each other. The generic modes, roll and yaw exhibit simple harmonic motion, while the pitch amplitude increases linearly with time for a given initial pitch rate.

#### 4. Controlled Motion

The rotational equations of motion are combined with the generic modal equations using the nondimensional orbital time and length variables and then recast into conventional state space form:

$$\dot{X}' = AX + BU \quad (34)$$

where the state vector,  $X$ , is defined as

$$X = (x_1, x_2, x_3, \dots, x_{1+3}, \dots, x_{n+6+1}, \dots, x_{2n+6})^T$$

and  $x_1 = \phi$ ;  $x_2 = \psi$ ;  $x_3 = \theta$ ;  $x_{1+3} = \dot{\phi} = \Delta_1/a$ ;

$i = 1, 2, \dots, n$  generic modes

$$x_{n+4} = \dot{\phi}'; \quad x_{n+5} = \dot{\psi}'; \quad x_{n+6} = \dot{\theta}'$$

$$x_{n+6+i} = \dot{z}_i' = \dot{x}_{i+3}, \quad i = 1, 2, \dots, n$$

For the examples to be considered in this paper it is assumed that the system can be modelled by three rigid body rotational modes and the first three generic (flexible) modes.

The general A matrix

$$[A] = \begin{bmatrix} \text{Diagonal} & \begin{matrix} \text{6x6} \\ \Delta \\ \text{6x6} \end{matrix} & \begin{matrix} [I] \\ \text{6x6} \end{matrix} \\ \hline 0 & \begin{matrix} \Delta_{78} & \dots & 0 \\ \Delta_{78} & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 \end{matrix} & \begin{matrix} \dots & 0 \\ \dots & 0 \\ \dots & 0 \end{matrix} \end{bmatrix} \quad (35)$$

The non-zero and non-unity elements appearing in A are;

$$A_{7,1} = 4(I_x - I_y)/I_z; \quad A_{8,2} = -(I_y - I_x)/I_x;$$

$$A_{9,3} = 3(I_x - I_z)/I_y; \quad A_{10,4} = -(\omega_1/\omega_c)^2;$$

$$A_{11,5} = (\omega_2/\omega_c)^2; \quad A_{12,6} = -(\omega_3/\omega_c)^2;$$

$$A_{8,7} = (I_y - I_z - I_x)/\omega_c I_x; \quad A_{7,8} = (I_x - I_y + I_z)/\omega_c I_z$$

The general B matrix:

$$B = \begin{bmatrix} O_{6 \times 6} \\ B_{6 \times 6} \end{bmatrix}$$

where the lower part of the B matrix depends on actuator locations.

#### Control Law Selection

Control laws are developed using 3 different techniques. They are: (a) decoupling of the original state equations using state variable feedback; (b) stabilizing the system by clustering the poles on a line parallel to the imaginary axis and in the negative s-plane using the control law of the type  $U = -KX$ ; (c) applying the linear regular theory to the original system equations;

#### (a) Decoupling of Original State Equations Using State Variable Feedback

The equations of motion of the platform can be written as

$$\ddot{x} = A\dot{x} + Cx + BU \quad (36)$$

where  $x = (x_1, x_2, \dots, x_{n+3})$

After selecting  $U = K_p \dot{x} + K_p x$  we can rewrite the controlled motion equations as

$$\ddot{x} = (A + BK_p)\dot{x} + (C + BK_p)x \quad (37)$$

$K_p$  and  $K_p$  are evaluated such that  $(A + BK_p)$  and  $(C + BK_p)$  are diagonalized and thus yield required damping and frequency of the controlled modes. The number of modes must be equal to the number of actuators to avoid the use of pseudo-inverse matrices.

Two sets of actuator locations have been assumed for each of the three nominal orientations previously described. For all orientations, (i)-(iii), it is assumed that five actuators are located on the larger surface (with force axis normal to it) and a sixth actuator along an edge. The body coordinates of the six actuators are taken as

Case (i)

First Location  $a = 100m$

$$f_1(-a/6, -a/6, 0); \quad f_2(a/6, -a/6, 0); \quad f_3(-a/6, 0, 0)$$

$$f_4(a/6, 0, 0); \quad f_5(-a/6, a/6, 0); \quad f_6(a/2, a/6, 0)$$

Second Location  $a = 100m$

$$f_1(-a/2, -a/2, 0); \quad f_2(a/2, -a/2, 0); \quad f_3(-a/2, a/2, 0)$$

$$f_4(a/2, a/2, 0); \quad f_5(-a/2, 0, 0); \quad f_6(a/2, a/2, 0)$$

Case (ii)

First Location  $a = 100m$

$$f_1(0, -a/6, -a/6); \quad f_2(0, -a/6, a/6); \quad f_3(0, 0, -a/6)$$

$$f_4(0, 0, a/6); \quad f_5(0, a/6, -a/6); \quad f_6(0, a/6, a/2)$$

Second Location  $a = 100m$

$$f_1(0, -a/2, -a/2); \quad f_2(0, -a/2, a/2); \quad f_3(0, a/2, -a/2)$$

$$f_4(0, a/2, a/2); \quad f_5(0, 0, -a/2); \quad f_6(0, a/2, a/2)$$

Case (iii)

First Location  $a = 100m$

$$f_1(-a/6, 0, -a/6); \quad f_2(-a/6, 0, a/6); \quad f_3(0, 0, -a/6)$$

$$f_4(0, 0, a/6); \quad f_5(a/6, 0, -a/6); \quad f_6(a/6, 0, a/2)$$

Second Location  $a = 100m$

$$f_1(-a/2, 0, -a/2); \quad f_2(-a/2, 0, a/2); \quad f_3(a/2, 0, -a/2)$$

$$f_4(a/2, 0, a/2); \quad f_5(0, 0, -a/2); \quad f_6(a/2, 0, a/2)$$

Actuator positions for the two different sets of locations are illustrated in Fig. 3. The system A and B matrices corresponding to different combinations of the three platform orientations and the two sets of actuator location are listed as follows.

Case (i) Platform Following Local Vertical  
With Major Surface Normal to the Orbit Plane.

The non zero elements of the A matrix are:

$$\begin{aligned} A_{1,1+6} &= -1 \text{ for } i = 1, \dots, 6; A_{7,8} = 800; \\ A_{8,7} &= -1600; A_{8,2} = 1; A_{9,3} = -3; \\ A_{10,4} &= -277.414, A_{11,5} = -588.647; \\ A_{12,6} &= -976.844 \end{aligned}$$

Location I (Lower Part of B matrix)

0.0	0.0	0.0	0.0	0.0	-0.22643
-0.4329	-0.4329	0.0	0.0	0.4329	0.0
0.4329	-0.4329	0.4329	-0.4329	0.4329	0.0
-0.003126	-0.003126	0.0	0.0	-0.003126	0.0
0.0	0.0	-0.003084	-0.003084	0.0	0.0
-0.008786	-0.008786	-0.0115	-0.0115	-0.008786	0.0

Location II (Lower Part of B matrix)

0	0	0	0	0	-0.6796
-1.3592	-1.3592	1.3592	1.3592	0	0.0
1.3592	-1.3592	1.3592	-1.3592	1.3592	0.0
0.023	-0.023	-0.023	0.023	0.0	0.0
0.0	0.0	0.0	0.0	-0.023	0.0
0.023	0.023	0.023	0.023	0.023	0.0

Case (ii) Platform Along Local Horizontal

The elements of the A matrix that are different from Case (i) are:

$$\begin{aligned} A_{7,1} &= 4; A_{7,8} = 1600; A_{8,2} = 0; A_{8,7} = -800; \\ A_{9,3} &= 3. \end{aligned}$$

Location I (Lower Part of B matrix)

0.4329	0.4329	0.0	0.0	-0.4329	0.0
0.0	0.0	0.0	0.0	0.0	0.22643
-0.4329	0.4329	-0.4329	0.4329	-0.4329	0.0
0.003126	-0.003126	0.0	0.0	-0.003126	0.0
0.0	0.0	-0.003084	-0.003084	0.0	0.0
-0.008786	-0.008786	-0.0115	-0.0115	-0.008786	0.0

Location II (Lower Part of B matrix)

1.3592	1.3592	-1.3592	-1.3592	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.6796
-1.3592	1.3592	-1.3592	1.3592	-1.3592	0.0
0.023	-0.023	-0.023	0.023	0.0	0.0
0	0.0	0.0	0.0	-0.023	0.0
0.023	0.023	0.023	0.023	0.023	0.0

Case (iii) Major Surface in Orbit Plane

The elements of the A matrix that are different from Case (i) are:

$$\begin{aligned} A_{7,1} &= -4; A_{7,8} = 0; A_{8,2} = -1; A_{8,7} = 0; \\ A_{9,3} &= 0. \end{aligned}$$

Location I (Lower Part of B matrix)

-0.4329	-0.4329	0.0	0.0	0.4329	0.0
0.4329	-0.4329	-0.4329	-0.4329	0.4329	0.0
0.0	0.0	0.0	0.0	0.0	-0.22643
0.003126	-0.003126	0.0	0.0	-0.003126	0.0
0.0	0.0	-0.003084	-0.003084	0.0	0.0
-0.008786	-0.008786	-0.0115	-0.0115	-0.008786	0.0

Location II (Lower Part of B matrix)

-1.3592	-1.3592	1.3592	1.3592	0.0	0.0
1.3592	-1.3592	1.3592	-1.3592	1.3592	0.0
0.0	0.0	0.0	0.0	0.0	-0.6796
0.023	-0.023	-0.023	0.023	0.0	0.0
0.0	0.0	0.0	0.0	-0.023	0.0
0.023	0.023	0.023	0.023	0.023	0.0

For all combinations considered above the gains are selected so as to produce 20% of critical damping in each of the rigid body modes and the first generic mode, and 10% of critical damping in the second and third generic modes. In order to provide a better transient response in the lower frequency fundamental elastic mode, the percentage of critical damping is selected to be twice that in the remaining flexible modes. The time response of the rigid body modes and the generic modal amplitudes for all combinations considered and for equal initial position displacements in all components of the state is illustrated in Fig. 4a.

As an example of the time history of the required control forces, Fig. 4b. shows such a time response for the exterior (II nd) location of the actuators with the platform nominally following the local vertical and the major surface area of the platform in the orbital plane. A complete summary of the maximum force amplitudes required for all combinations of actuator locations and platform orientations is given in Table I. In interpreting the results of Table I, it should be pointed out that, in the process of achieving both orientation and shape control, the maximum force(s) required of any actuator will vary with both the moment arm about the principal body axes and the value of the modal shape function at the particular actuator location for all modes contained within the mathematical model.

(b) Stabilizing the System by Pole Clustering

The equations of motion of the platform when recast in state space format can be written as

$$\dot{X} = AX + BU \quad (38), \quad X \rightarrow 2(n+3)x_1$$

ORIGINAL PAGE IS  
OF POOR QUALITY

Table 1 Maximum Force Amplitudes (Newtons) for Different Combinations of Cases with Actuator Locations

Force	Location I (interior)			Location II (exterior)			
	Case 1	Case 11	Case 111	Case 1	Case 11	Case 111	
$f_1$	-807.8	270.63	270.6	57.6	62.56	47.4	Indicates maximum level of control force required for each case.
$f_2$	270.0	136.2	-133.3	28.19	26.20	-17.2	
$f_3$	1141.0	387.3	-425.0	-65.89	-49.5	30.0	
$f_4$	527.9	195.6	163.7	17.10	-30.90	25.08	
$f_5$	489.09	-253.3	153.0	-44.6	-44.67	-44.67	
$f_6$	-264.58	-272.0	4.4	88.16	-88.06	1.47	

The control,  $U = -KX$  is selected by using a digital computer algorithm<sup>8</sup> such that  $(A-BK)$  has the required identical negative real part in each of its eigenvalues. Although the number of actuators can be less than the number of modes (one half of the dimensionality of the state vector), a limitation of this algorithm is that the gains are selected such that all of the closed-loop poles lie on a line parallel to the imaginary axis. However this algorithm is useful when it is important that each mode in the system satisfy some minimum damping characteristics.

As an example of this technique we consider the system with four actuators and six modes where control about the first orientation (Case 1) is desired. Three of the actuators are assumed to provide forces perpendicular to the major surfaces with the remaining actuator thrusting normal to an edge. The actuator coordinates in the body system (Fig. 1a) are:  $f_1(-a/6, -a/6, 0)$ ;  $f_2(a/6, -a/6, 0)$ ;  $f_3(-a/6, 0, 0)$ ; and  $f_4(a/2, a/6, 0)$  where  $a = 100m$ . It is assumed that the minimum damping requirement on the system has a time constant of (13.33 min or (1/2 $\pi$ ) dimensional orbital time). The control influence matrix is then calculated based on the assumed coordinates of the four actuators. The control  $U = -KX$  can be calculated by the ORACLS pole clustering algorithm. Based on these gains time histories of the required control forces are then obtained.

The control influence matrix (lower part), closed loop poles, and maximum force amplitudes required are summarized as follows:

B matrix (Lower Part)

0.0	0.0	0.0	-0.22645
-0.4529	-0.4529	0.0	0.0
0.4529	-0.4529	0.0	0.0
0.003126	-0.003126	0.0	0.0
0.0	0.0	-0.0030844	0.0
-0.008786	-0.008786	-0.0115	0.0

Closed Loop Poles (Nondimensionalized)

The real part is -1.0 and the imaginary parts are +0.000485, +0.993, +16.82, +24.26, +31.33 and +1131.37. The Maximum force amplitudes (Newtons) are calculated as

$|f_1| = 78.5$ ,  $|f_2| = 36.4$ ,  $|f_3| = 169.5$ , and  $|f_4| = 35.3$ .

An interesting comparison can now be made between this result and that shown in Table I for case (1) and the first (I) location of the six actuators considered there. It can be seen that by using fewer actuators, appropriately placed, that better transient response characteristics can be obtained with smaller maximum force amplitudes. However a disadvantage of this method is that some of the controlled frequencies may be orders of magnitude greater than the highest frequency of the uncontrolled system (for this example compare 1131 with  $\sqrt{976} = 31.24$ ). Depending on the nature of the expected disturbance forces this result could be very undesirable.

(c) Application of the Linear Regulator Theory

The control law,  $U = -KX$ , is selected such that the following performance index is minimized

$$J = \int_0^{\infty} (X^T Q X + U^T R U) dt \quad (39)$$

where Q and R are positive definite penalty matrices. The steady state solution of the matrix Riccati equation of dimension equal to the state has to be solved in order to evaluate the gain matrix, K.

A computer algorithm within the ORACLS<sup>8</sup> software package is used to obtain the gain matrices K for different combinations of the Q and R penalty matrices. This algorithm utilizes the Newton Raphson method of solving the Riccati equation. In the examples considered here four actuators are assumed with the system represented by three rigid body and three flexible modes. The locations of the four actuators are taken to be the same as in Section (b), and control about the first nominal orientation (1) is considered.

The weighting matrix, Q, is selected based on the following considerations. For the example considered here it can be seen from Eq. (34), (35), and the B matrix that the uncontrolled system dynamics is either described by sets of uncoupled harmonic oscillators, or (in the case of roll/yaw motion) by a coupled two dimensional harmonic oscillator. The latter motion can be represented by

$$\begin{bmatrix} \dot{\omega}_z \\ \dot{\omega}_x \end{bmatrix} = \begin{bmatrix} 0 & a \\ -b & 0 \end{bmatrix} \begin{bmatrix} \omega_z \\ \omega_x \end{bmatrix} \quad (40)$$

where the system oscillates at the frequency  $\Omega = \sqrt{ab}$ . It is desired that the control remove a maximum "transverse" angular rate,  $\omega_{z \max} = \max \sqrt{\omega_z^2 + \omega_x^2} = \sqrt{\omega_z^2(0) + \omega_x^2(0)}$



so that a strategy for selecting the elements of  $Q$  could be<sup>9,10</sup>

$$Q = \begin{bmatrix} f & 0 \\ 0 & f \end{bmatrix} \quad \text{where } f = \frac{\Omega^2}{\omega_T^2} \quad (41)$$

when the control penalty matrix is fixed. The remaining equations for any of the uncoupled oscillators can also be expressed by

$$\begin{bmatrix} \dot{x}_1' \\ \dot{x}_1'' \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -(\omega_1/\omega_c)^2 & 0 \end{bmatrix} \begin{bmatrix} x_1' \\ x_1'' \end{bmatrix} \quad (42)$$

in the same format as Eq. (39), and thus the weights can be obtained in a similar manner.

The  $Q$  matrix for the case considered here (control about nominal orientation (1) with actuator locations as given in Section (b)) is obtained using the relations given by Eq. (41) and is a diagonal matrix,  $Q_0$ , with the following elements:

$$\begin{aligned} Q_{1,1} &= 4.324 \times 10^9, & Q_{2,2} &= 8.539 \times 10^9, \\ Q_{3,3} &= Q_{9,9} = 3.0 \times 10^4, & Q_{4,4} &= Q_{10,10} = \\ &2.77414 \times 10^6, & Q_{5,5} &= Q_{11,11} = 5.88647 \times 10^6, \\ Q_{6,6} &= Q_{12,12} = 9.74844 \times 10^6, & Q_{7,7} &= Q_{8,8} = \\ &2.222 \times 10^3. \end{aligned}$$

The  $R_0$  matrix is chosen as an identity matrix. A parametric study is done using various multiples of the  $Q_0$  ( $Q = \alpha Q_0$ ) and  $R_0$  matrices obtained above which are plotted against the negative real part of the least damped mode of the controlled system in Fig. 5. All the loci of the negative real part of the least damped mode approach unity and no significant improvement is observed by increasing the state penalty,  $Q = \alpha Q_0$ , any further. Thus one wishes to operate on the horizontal line between the points (1) and (2). The maximum amplitude of the forces for  $R = I$  and  $R = 1000 I$  are calculated and plotted in Fig. 6. The closed loop poles of the controlled system at points (1) and (2) are virtually the same and are given as follows (nondimensionalized):

$$\begin{aligned} &-1.0043, -1.8 \pm 116.64, -2.16 \pm 124.20, -17.18 \pm 119.79, \\ &-26.23, -36.22, -137.64 \text{ and } -38.66 \pm 11132.11 \end{aligned}$$

The maximum force amplitudes as shown in Fig. 6 are less than those corresponding to Case (1) - Location I of Table 1 for comparable transient responses, whereas these are high as compared to the forces obtained using the pole clustering technique (Section (b)). This is due to the large negative real parts of the other modes in the linear regulator case when compared to the pole clustering technique where all the poles have an equal negative real part (-1.0). Both the linear regulator and pole clustering technique have the drawback that the controlled frequencies can be quite high compared to the uncontrolled frequencies. On the otherhand, these techniques have the advantage that they can be applied to situations where the number of actuators is less than the number of modes in the mathematical model, in contrast to the decoupling technique of Section (a).

### 3. Conclusions

In this paper the dynamics, stability, and control of an orbiting homogeneous, flexible square platform are considered. Three different nominal orientations of the platform are examined. When the platform is nominally following the local vertical with its larger surface perpendicular to the orbital plane and also when the platform follows the local horizontal with its larger surface normal to the local vertical, it is seen that the uncontrolled roll/yaw motion is unstable. For the case where the platform follows the local vertical with its large surface perpendicular to the orbit normal, the uncontrolled pitch motion is found to be unstable.

Three different control techniques are considered for the selection of the control laws:

- a) The decoupling of the original state equations using state variable feedback eliminates the need of a transformation from the original coordinates to the modal coordinates and provides a method of specifying directly the amount of damping and frequency of the individual components of the state vector. However, with this technique the number of actuators must be equal to the number of coordinates (modes) in the model.
- b) The pole placement algorithm (ORACLS) guarantees the over-all required damping of the system and does not restrict the number of actuators to be equal to the number of modes in the model. However, it is seen that the closed-loop frequencies may be greatly increased when compared to the open-loop values which may cause problems with externally induced periodic excitations.
- c) The linear regulator theory can provide acceptable performance once the state and penalty matrices are properly selected, and the number of actuators can be less than the number of modes in the model. Computer capacity and accuracy limit the number of modes that can be considered. Here, too, an undesirable increase in the closed-loop frequencies may result in order to provide satisfactory responses with maximum allowable force amplitudes.

### References

1. "Outlook for Space," NASA Report SP-386, January 1976.
2. "Industry Workshop on Large Space Structures," NASA Contractor Report CR-2709, Contract No. NAS-1-12436, for NASA Langley Research Center, May 1976.
3. Kumar, V.K. and Bainum, P.M., "Dynamics of a Flexible Body in Orbit," AIAA/AAS Astrodynamics Conference, Palo Alto, Calif., August 7-9, 1978, Paper No. 78-1418; also, Journal of Guidance and Control, Vol. 3, No. 1, January-February 1980, pp. 90-92.

4. Santini, Paolo, "Stability of Flexible Spacecrafts," Acta Astronautica, Vol. 3, 1977, pp. 685-713.
5. Bainum, P.M. and Reddy, A.S.S.R., "On the Controllability of a Long Flexible Beam in Orbit," Proceedings of the Second AIAA Symposium on Dynamics and Control of Large Flexible Spacecraft, June 21-23, 1979, VPI & SU Press, (Ed. L. Mairovitch), 1980, pp. 145-159.
6. Reddy, A.S.S.R., Bainum, P.M., and Hamer, H.A., "Decoupling Control of a Long Flexible Beam in Orbit," Astrodynamics 1979, Advances in the Astronautical Sciences, Vol. 40, Part II, AAS Publication, 1980, pp. 649-673.
7. Bainum, P.M., Reddy, A.S.S.R., Krishna, R., and James, F., "The Dynamics and Control of Large Space Structures III," Final Report, NASA Grant NSG-1414, Supp. 2, Part A: Shape and Orientation Control of a Platform in Orbit using Point Actuators, Howard University, June 1980.
8. Armstrong, E.S., "ORACLS - A System for Linear-Quadratic-Gaussian Control Law Design," NASA Technical Paper 1106, April 1978.
9. Bainum, P.M. and Sellappan, R., "Optimal Control of Spin-Stabilized Spacecraft with Telescoping Appendages," The Journal of the Astronautical Sciences, Vol. XXI, No. 4, October-December 1976, pp. 329-346.
10. Amieux, J.C. and Liegeois, A., "Design and Ground Test of a Pendulum - Type Active Nutation Damper," Journal of Spacecraft and Rockets, Vol. 11, No. 11, Nov. 1974, pp. 790-792.

$M_r$	$r^{\text{th}}$ modal mass
$T$	Torque due to an actuator
$T_x, T_y, T_z$	Torque components
$W_r(x, y)$	$r^{\text{th}}$ modal shape function
$Z_r$	Nondimensionalized $r^{\text{th}}$ modal amplitude function
$\omega_c$	Orbital frequency
$\omega_x, \omega_y, \omega_z$	Angular body rates
$\phi, \psi, \theta$	Roll, yaw, pitch, respectively
$\omega_1, \omega_2, \omega_3$	First three modal frequencies of the plate

#### Appendix A - Nomenclature

$A_r$	$r^{\text{th}}$ modal amplitude function
$B$	Control influence matrix
$C_x, C_y, C_z$	Disturbance torques about the principal undeformed body axes
$E_r$	Generic force on $r^{\text{th}}$ mode
$f$	Force due to an actuator
$f_x, f_y, f_z$	Force components due to an actuator
$G_{R_x}, G_{R_y}, G_{R_z}$	Gravity gradient torques about the principal undeformed body axes
$I_x, I_y, I_z$	Moments of inertia about the principal axes
$K$	Gain matrix
$K_v, K_p$	Rate and position feedback gain matrices

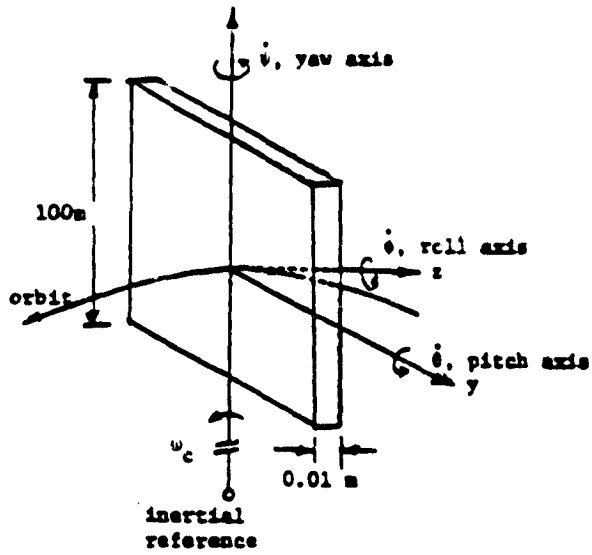


Fig. 1a. Platform following local vertical with major surface normal to the orbit plane - Case (1)

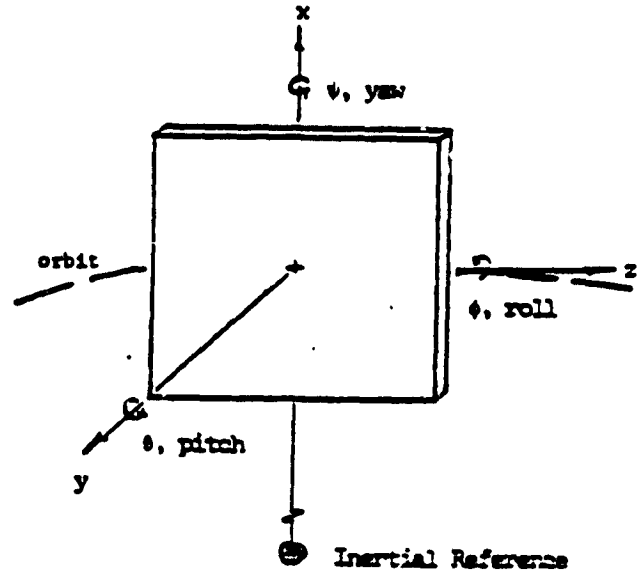


Fig. 1c. Platform following local vertical with major surface in the orbit plane - Case (1c)

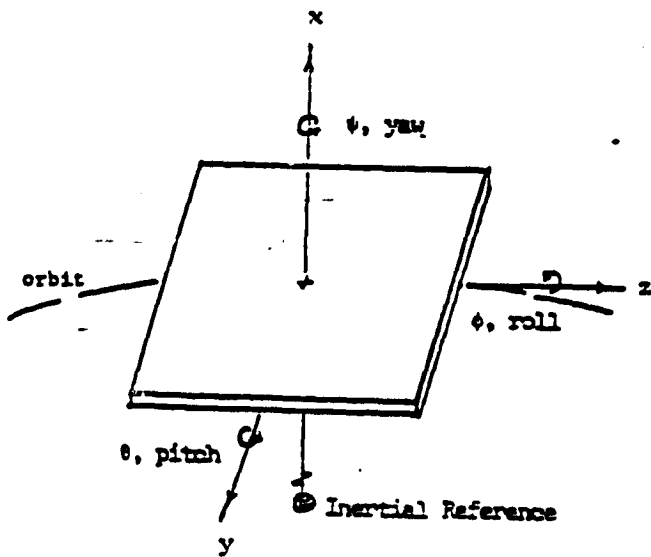


Fig. 1b. Platform along local horizontal - Case (1b)

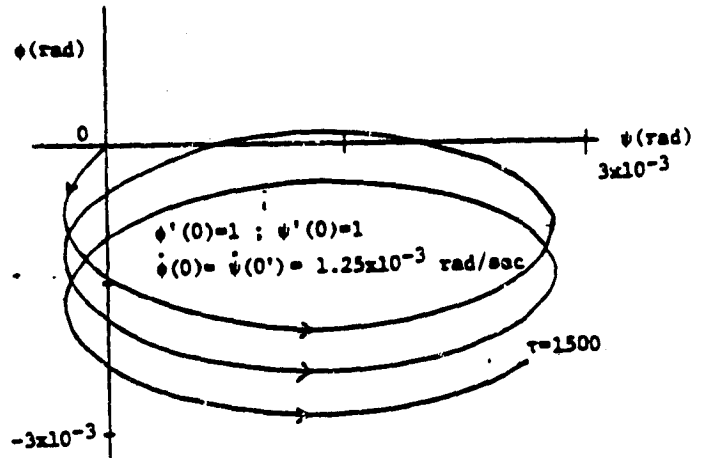


Fig. 2. Roll/yaw motion (uncontrolled) - Case (1)

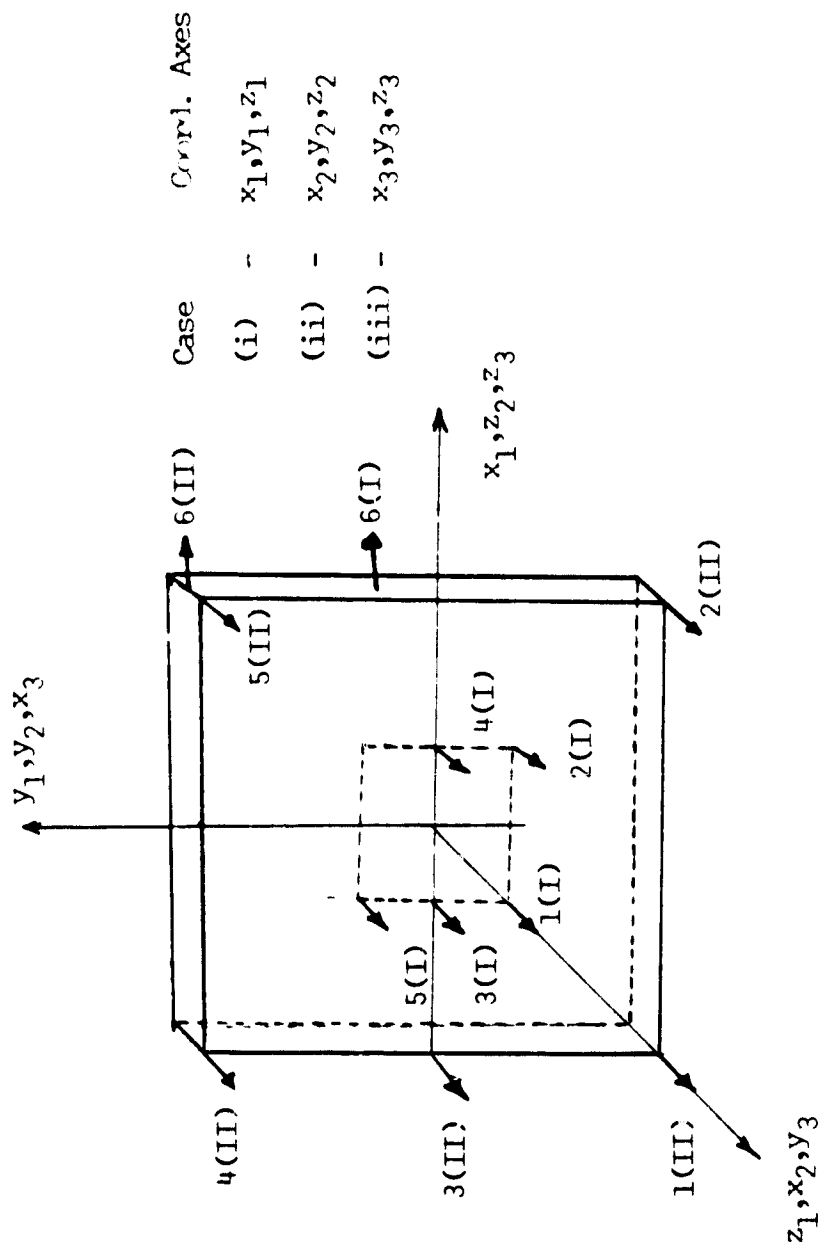


Fig 3. Location of two sets of actuators (I & II)

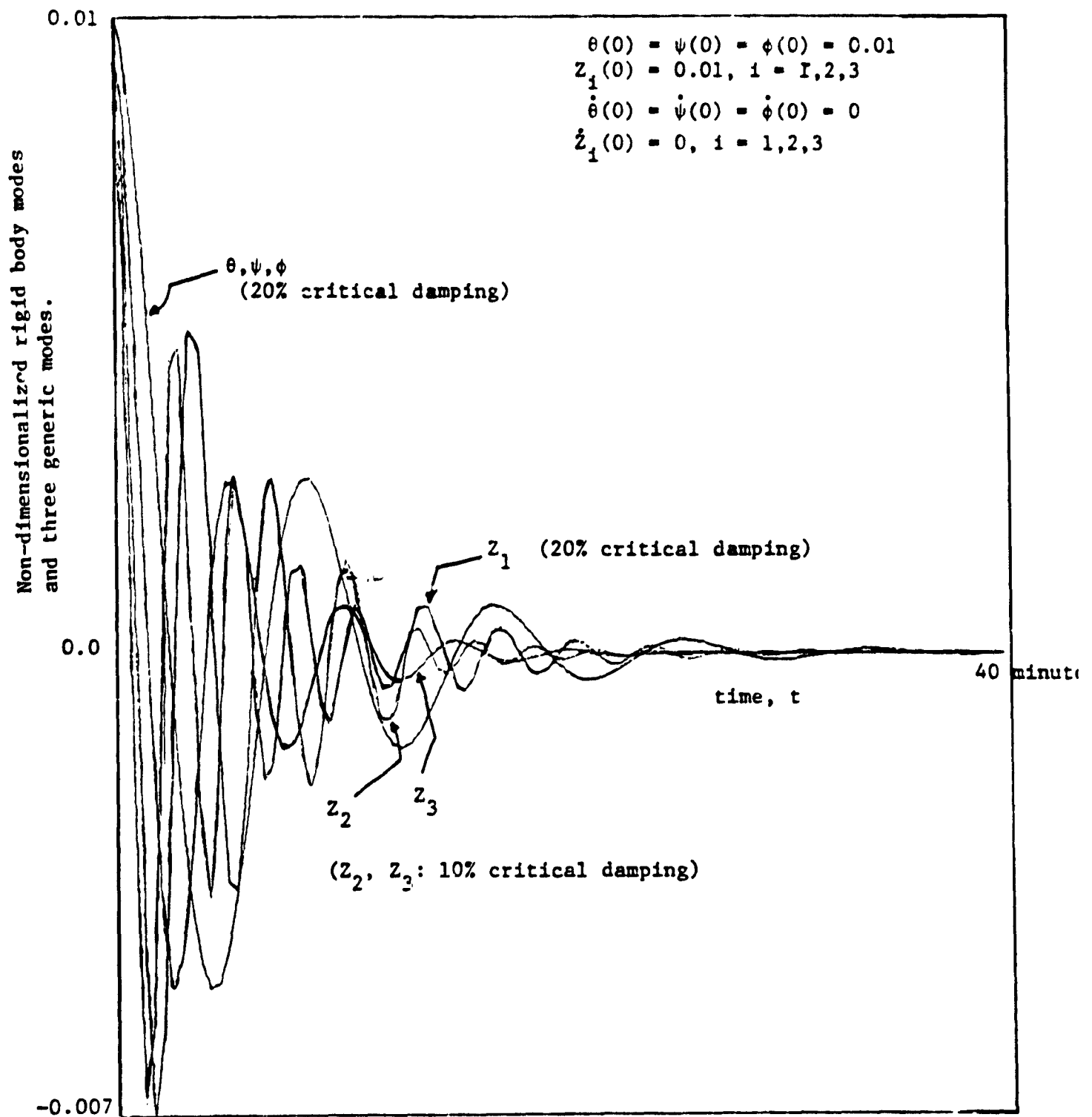


Fig 4a. Controlled state response for all combinations of orientations and actuator locations.

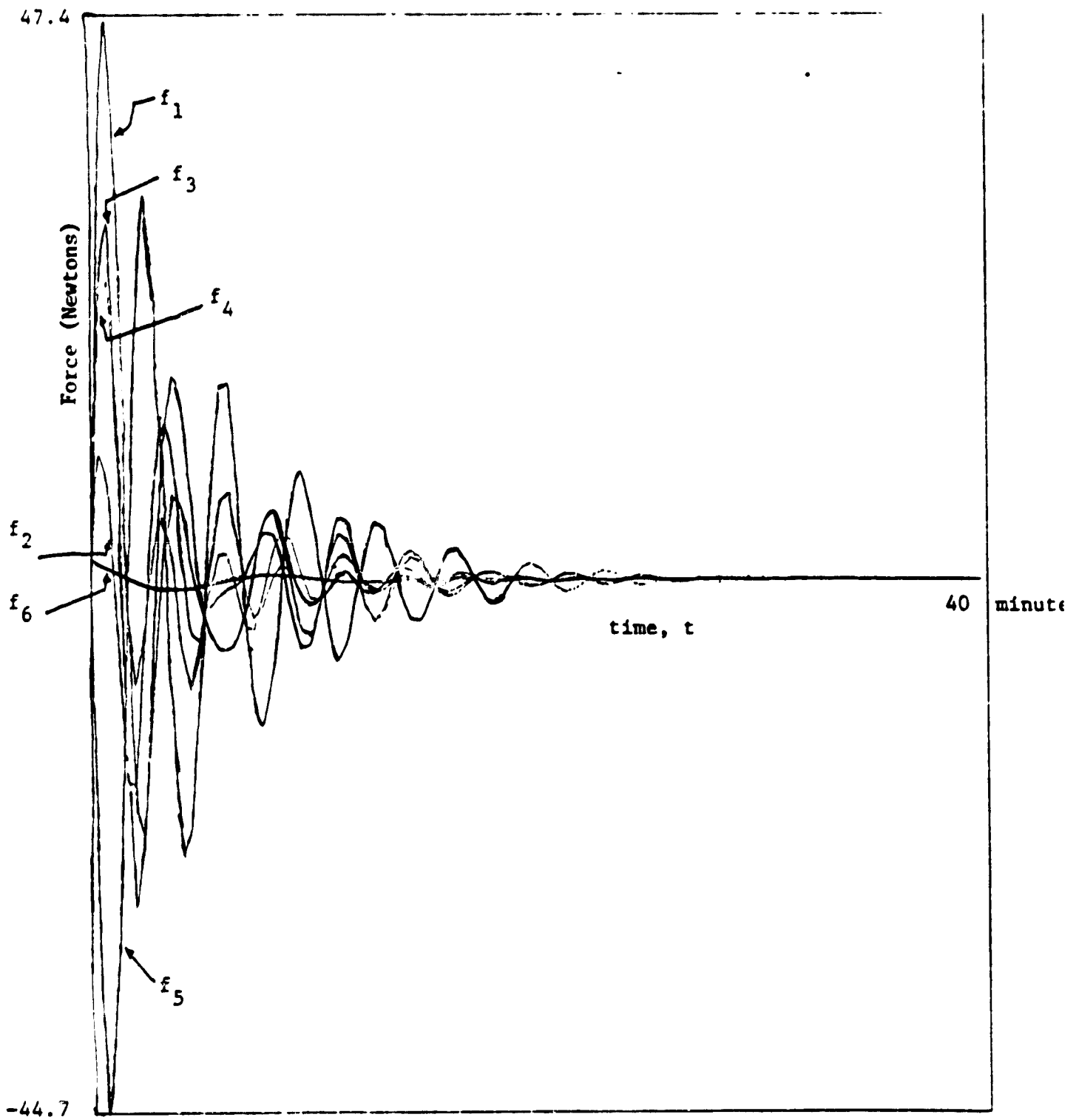


Fig 4b. Control force time history for Case (iii) - II

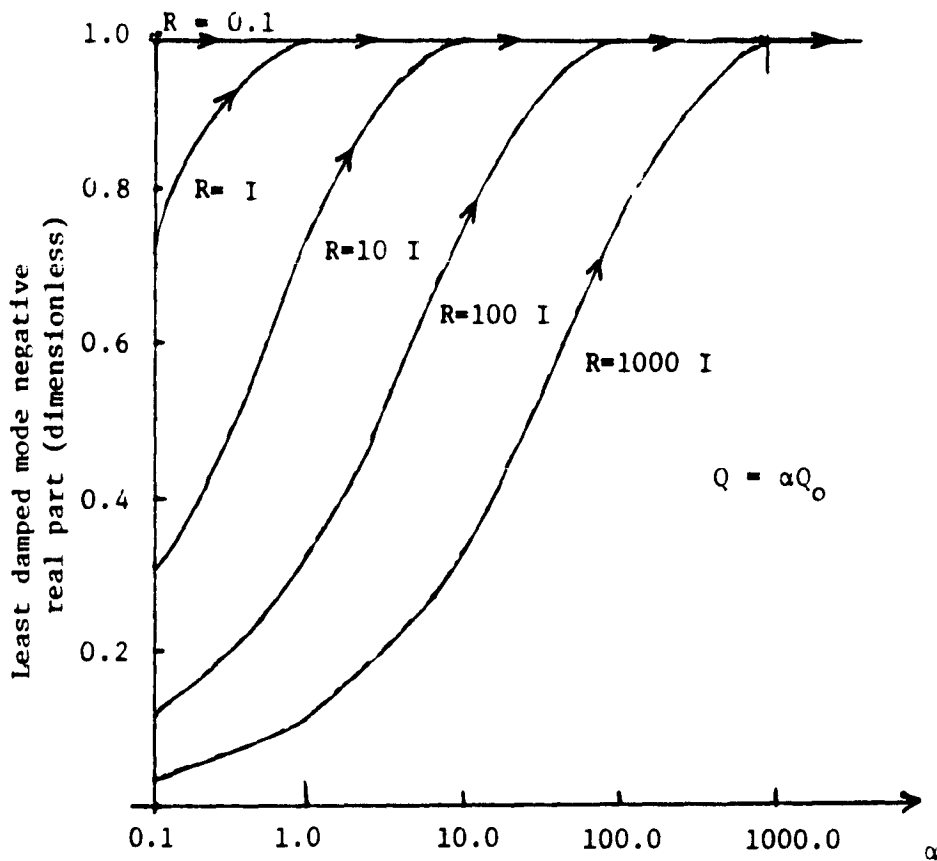


Fig. 5. Variation of least damped mode negative real part with  $\alpha$  and  $R$ .

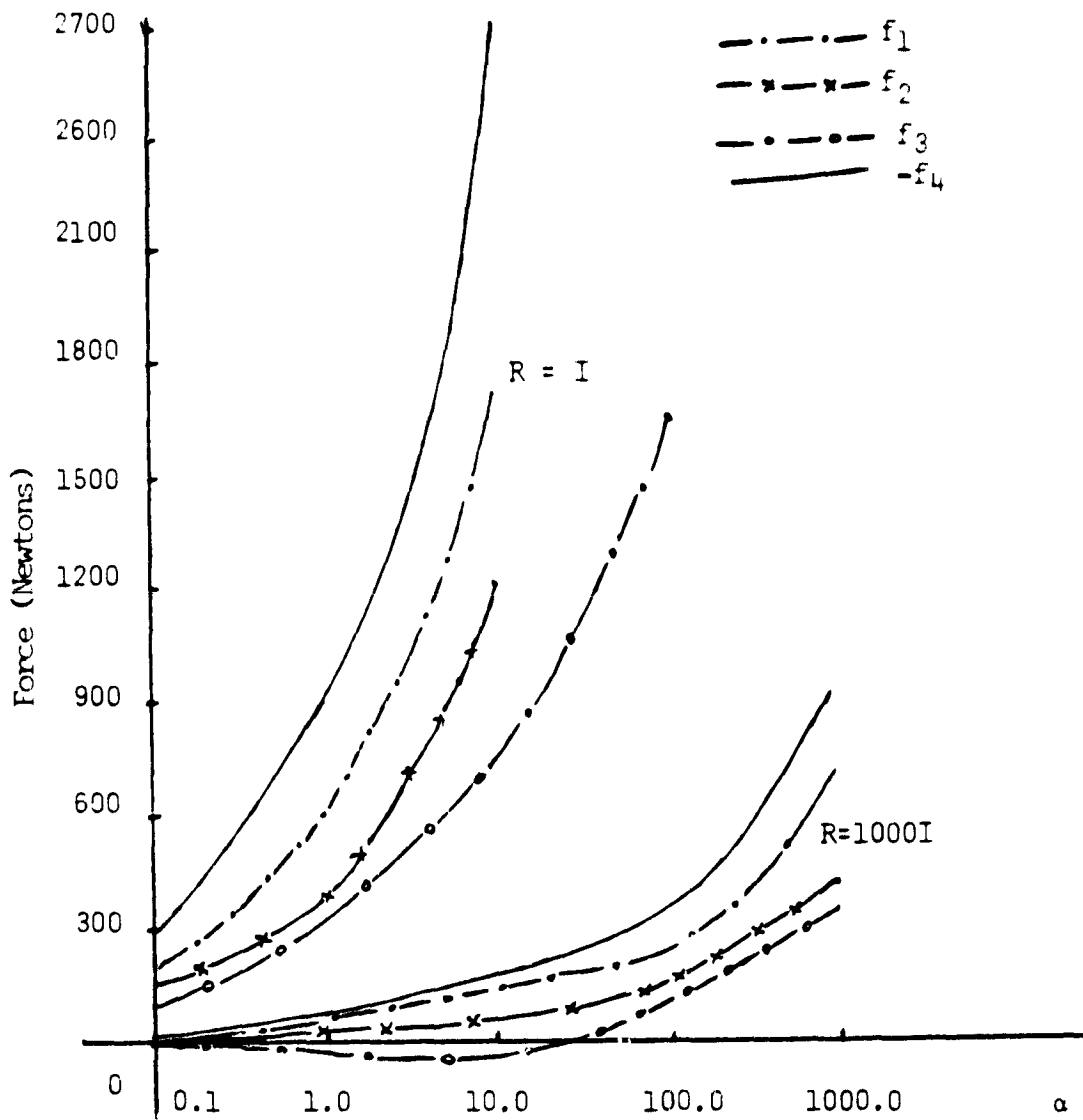


Fig 6. Maximum force amplitudes as a function of  $\alpha$  and  $R$  for all actuators - (application of linear regulator theory).



### III. FREQUENCIES AND MODE SHAPES FOR RECTANGULAR PLATES

The ability to determine accurately the frequencies and mode shapes is essential for the analysis and control of large structures in orbit. A thin rectangular plate, an important basic structure for several space applications, is considered for vibrational analysis. In the following sections the plate is assumed to be large, thin, and homogeneous, and all the edges are assumed to be free to vibrate. First, the approximate frequencies and mode shapes of a rectangular plate obtained by Warburton<sup>1</sup> is discussed. This analysis also includes the special case of a square plate. Next, the analytical results for a square plate using the method of Lemke<sup>2</sup> is considered. For a specific example of a square plate both analytical results are applied to determine the frequencies and mode shapes. An available finite element computer program<sup>3</sup> is also used to obtain the frequencies and mode shapes of this plate. The results of both analytical methods and the computer routine are compared and discussed.

#### 1. Formulation by Warburton

The approximate frequency formula is derived by applying the Raleigh method. The details of this method are given in the earlier contract report.<sup>4</sup> The basic equation used was the plate vibrational equation in the cartesian co-ordinate system (x,y), with the length and width of the plate taken along the x and y directions, respectively, and is given as

$$\frac{\partial^4 W}{\partial x^4} + 2 \frac{\partial^4 W}{\partial x^2 \partial y^2} + \frac{\partial^4 W}{\partial y^4} + \frac{12\rho(1-\sigma^2)}{Egh^2} \frac{\partial^2 W}{\partial t^2} = 0 \quad (\text{III-1})$$

where  $\rho, \sigma$  and  $E$  are the density, Poisson's ratio and Young's modulus of the plate material, respectively,  $h$  is the plate thickness, and  $g$  is the acceleration due to gravity. The displacement,  $w$ , at any point  $(x,y)$  at time  $t$  is given by

$$w = W \sin \omega t = A \theta(x) \phi(y) \sin \omega t \quad (\text{III-2})$$

$\theta(x)$  and  $\phi(y)$  can be taken as the beam functions orthogonal to each other and can be used to approximate plate behavior. After taking the appropriate free-free beam functions for  $\theta(x)$  and  $\phi(y)$ , the frequency expressions for a rectangular plate was derived as<sup>1</sup>

$$\lambda^2 = \frac{\rho a^4 (2\pi f)^2 12(1-\sigma^2)}{\pi^4 E h^2 g} \quad (\text{III-3})$$

$$\text{and } \lambda^2 = G_x^4 + G_y^4 \frac{a^4}{b^4} + 2 \frac{a^2}{b^2} [\sigma H_x H_y + (1-\sigma) J_x J_y] \quad (\text{III-4})$$

where  $\lambda$  is a non-dimensional frequency factor,  $a$  and  $b$  are the length and width of the plate, and  $G_x, H_x, J_x, G_y, H_y,$  and  $J_y$  are functions associated with the number of nodal lines,  $m$  and  $n$ , parallel to  $x$  and  $y$ , respectively, for the beam functions  $\theta(x)$  and  $\phi(y)$ , and are given in Table III-1. From Eq. (III-3), the frequency is obtained as

$$f = \frac{\lambda h \pi}{a^2} \left[ \frac{E g}{48 \rho (1-\sigma^2)} \right]^{\frac{1}{2}} \quad (\text{cps}) \quad (\text{III-5})$$

Eq. (III-5) is valid for thin rectangular plates. However, for square plates,  $(m,n) \pm (n,m)$  types of modes exist, and for these cases  $\lambda$  in Eq. (III-5) must be modified. These cases are discussed in detail in Ref. 1 and a few relevant results are given here.

Modes (m,0) + (0,m), m is even

$$\lambda^2 = (m-\frac{1}{2})^4 + 2 \sigma (m-\frac{1}{2})^2 \frac{8}{\pi^2}$$

Modes (m,1) ± (1,m), for m = 3,5,7,....

(III-6)

$$\lambda^2 = (m-\frac{1}{2})^4 + 2(1-\sigma) (m-\frac{1}{2})^2 \left[ 1 + \frac{6}{(m-\frac{1}{2})\pi} \right] \frac{12}{\pi^2} \\ \pm 2 \sigma (m-\frac{1}{2})^2 \frac{24}{\pi^2} \left[ 1 - \frac{2}{(m-\frac{1}{2})\pi} \right]^2 \pm 2(1-\sigma) \frac{192}{\pi^4}$$

For any mode of vibration the nodal pattern is defined by m and n, the number of nodal lines in the x and y directions, respectively. The mode shapes are obtained by using the corresponding modal frequencies in the beam functions and then evaluating the product,  $\theta(x) \cdot \phi(y)$ , numerically.

## 2. Formulation by Lemke<sup>2</sup>

The frequencies and mode shapes were computed for a square plate using the Raleigh-Ritz method. The results are readily available only for six of the modes obtained by Warburton's method. Lemke uses displacement functions of the type,

$$W = \sum A_{m,n} \theta_m(x) \phi_n(y) \quad (III-7)$$

where  $\theta_m(x)$  and  $\phi_n(y)$  are the free beam functions given as

$$\theta_m(x) = \frac{\cosh k_m \cos k_m x + \cos k_m \cosh k_m \bar{x}}{\sqrt{\cosh^2 k_m + \cos^2 k_m}} \quad (m \text{ even}) \\ = \frac{\sinh k_m \sin k_m x + \sin k_m \sinh k_m x}{\sqrt{\sinh^2 k_m - \sin^2 k_m}} \quad (m \text{ odd}) \quad (III-8)$$

$\phi_n(y)$  is obtained from Eq. (III-8) by replacing x by y and m by n.

The values,  $k_m$ , are the roots of the equations

$$\tan k_m + \tanh k_m = 0 \quad (m \text{ even})$$

$$\tan k_m - \tanh k_m = 0 \quad (m \text{ odd})$$

which result from the spatial boundary conditions. Further, it was shown by an energy principle that <sup>2,4</sup>

$$\omega^2 = \frac{U_{\max}}{\frac{\rho h}{2g} \int_a^b \int_0^b W^2 dx dy} \quad (\text{III-9})$$

where  $U_{\max}$  is the maximum potential energy due to bending. The coefficients,  $A_{mn}$ , in Eq. (III-7) are determined to make  $\omega^2$  in Eq. (III-9) a minimum. Lemke obtained the coefficients,  $A_{mn}$ , by taking six or more terms in the series (III-7) and using four different values of Poisson's ratio. Expressions for six mode shapes and frequencies along with the coefficients,  $A_{mn}$ , are tabulated in Ref. 2. As an example the expression for the first mode is given here.

$$\begin{aligned} W(x,y) = & x_1 y_1 + 0.0325 (x_1 y_3 + x_3 y_1) - .005 x_3 y_3 \\ & - .00257 (x_1 y_5 + x_5 y_1) + .00121 (x_3 y_5 + x_5 y_3) \\ & - .000365 x_5 y_5 + \dots \end{aligned}$$

and 
$$\omega = \frac{13.086}{a^2} \sqrt{\frac{E h^3}{12 \rho (1-\sigma^2)}} \text{ for } \sigma = .343$$

### 3. Finite Element Computer Program

The computer program used is the Structural Design Language (STRUDL) which uses the finite element method to determine the mode shapes and the frequencies of vibration. The input to the computer routine is given by specifying the type of structure and supplying other physical properties and dimensions of the structure. For a rectangular plate, the finite elements can be specified as rectangular elements and the number of elements into which the plate should be divided depends upon the accuracy required. STRUDL gives deflections at each corner of the elements for all the modes from which the mode shapes can be determined. Further, a set of frequencies corresponding to the modes generated is obtained. In general, the accuracies of the frequencies and mode shapes will improve if the plate is modelled with a higher number of elements. However, computational errors due to truncation and round-off errors may predominate as the order of the elements increases beyond a limit. Further, the limitations of the computers will restrict the number of elements into which the plate can be divided to obtain more accurate results.

### 4. Discussion of Numerical Results

A square plate of sides 100 meters each and thickness 0.01 meters is considered to obtain the numerical results. The material of the plate is assumed to be aluminium with the following properties.

$$\text{density} = 2768.0 \text{ kg/m}^3$$

$$\text{Young's modulus} = 0.7441 \times 10^{10} \text{ kg/m}^2$$

$$\text{Poisson's ratio} = 0.33$$

Using Warburton's results, Eq. (III-4), Eq. (III-5), Table (III-1), and expressions for  $\theta(x)$  and  $\phi(y)$ , frequencies and mode shapes are calculated for different combinations of the number of nodal lines,  $m$  and  $n$ , starting with combinations of  $m=0$  and  $n=1$ , through  $m=3$  and  $n=3$ . The first three combinations of nodal line numbers,  $(0,0)$ ,  $(1,0)$  and  $(0,1)$ , represent rigid body motion. The first fundamental flexural frequency is seen to be due to a combination of  $m=1$  and  $n=1$ . The corresponding mode shape for the plate is obtained by multiplying the beam functions,  $\theta(x)$  and  $\phi(y)$ , for (beam) mode numbers 1 and 1, respectively (Fig. 1). Since the plate is approximated by sets of orthogonal beams in the  $x$  and  $y$  directions, the nodal pattern is also obtained by plotting the nodal points of these beams for their first modes. The next two higher frequencies are obtained by combinations of  $m=0$  and  $n=2$ , but the nodal patterns (Figs. 2,4) can not be visualized as before. This is because these frequencies are of a special type resulting from a combination of the  $(2,0)$  and  $(0,2)$  plate modes. It can be seen that when the mode corresponding to  $(2,0)$  (Fig. 3(a)) is superimposed on the mode -  $(0,2)$  (Fig. 3(b)) the mode shape depicted in Fig. 2 results. Similarly by superimposing the  $(2,0)$  and  $(0,2)$  modes the third mode shape (Fig. 4) is obtained. The two combinations of nodal patterns  $m=1$  and  $n=2$ , give identical frequencies for the fourth and fifth mode and the corresponding shapes (Fig. 5) are as expected. The next two higher frequencies are also identical and result from combinations of the  $(3,0)$  and  $(0,3)$  modes. The eighth frequency is obtained from  $m=2$  and  $n=2$  and the mode shape obtained is shown in Fig. 7.

However, the ninth and tenth mode shapes obtained by the (3,1) and (1,3) combinations, are once again of a special type. The ninth mode shape is obtained by superimposing the (1,3) and (3,1) patterns (Fig. 8) and the tenth mode shape is obtained by superimposing (1,3) and (3,1) nodal patterns (Fig. 9). The next higher frequencies are obtained from combinations of the (3,2), (2,3) and (3,3) modes, respectively. The frequencies and nodal patterns obtained for all these modes are shown in Table 2.

Frequencies and mode shapes are also obtained by using the expressions for the six modes given by Lemke.<sup>2</sup> The first three frequencies and mode shapes obtained agree with the frequencies and mode shapes computed from Warburton's formulas (Table 1). However, the next three frequencies obtained by Lemke's method correspond to higher frequencies and mode shapes obtained by Warburton's method. Also the nodal patterns obtained by Lemke's method compare approximately with the nodal patterns obtained by Warburton's method although the frequencies do not correspond in all cases. The results obtained by Lemke do not show the four intermediate frequencies corresponding to the fourth, fifth, sixth and seventh modes obtained by Warburton's method. The frequencies and nodal pattern obtained by Lemke's method are shown in Table 2.

For implementation of the computer program STRUDL, first the plate is divided into four elements. The first six modes (in terms of increasing frequencies) as predicted by STRUDL are also apparent from Warburton's results. The plate is assumed to be divided into 9, 16, 36 and 64 elements, respectively. The results of STRUDL are tabulated in Table 2. It can be seen that STRUDL frequencies approach the frequencies obtained by Warburton's method as the number of plate elements is increased.

However, in the cases of 36 and 64 elements some of the frequencies show a tendency to oscillate about an average value. This probably is due to the computational round off errors which begin to dominate with the increasing computations associated with larger number of elements. Thus, the advantage of taking a large number of elements may not be fully realized due to numerical accuracy limitations. Computation with more elements requires more computation time and a larger computer memory. For the 64 elements case, it was not possible to obtain the mode shapes due to memory limitations. It was also observed that the convergence of the frequencies, with an increase in the number of elements, is faster than the convergence of the mode shapes. It can be seen from Table 2, that the numerical results of STRUDL using 36 elements correlate with the results of Warburton both in frequency and mode shapes.

Table 3 and Table 4 compare non-dimensionalized deflections at the nodes (corners of elements) obtained by the three methods for the second mode (Fig. 2). For locations where deflections exist and do not correspond to maximum amplitude ( $\pm 1.0$ ) in all cases the results predicted by STRUDL lie in between the results obtained by the analytical methods of Lemke and Warburton.

The results of this comparative study give an indication of the types of modelling errors that would be expected in the estimation of the frequencies and mode shapes of the fundamental and lower order flexural modes of a large platform type structure in orbit. As an extension to this study the use of more powerful (and accurate) finite element computer algorithms, not currently available at Howard University, is recommended.



## 5. References

1. Warburton, G.B., "The Vibration of Rectangular Plates," Proc. Inst. Mech. Engrs., Vol. 168, No. 12, 1954, pp. 371-394.
2. Leissa, A.W., "Vibration of Plates", NASA SP-160, NASA, Washington, D.C., 1969, pp. 87-110.
3. ICES-STRUDL-II, Engineering Users Manual, Volume II, 1972.
4. Bainum, P.M. and James, P.K., "The Dynamics and Control of Large Flexible Space Structures - II. PART B: Model Development and Computer Simulation," June 1979.

TABLE III-1. Evaluation of Parameters in Frequency Expression (Warburton)

m	$G_x$	$H_x$	$J_x$	n	$G_y$	$H_y$	$J_y$
0	0	0	0	0	0	0	0
1	0	0	$\frac{12}{\pi^2}$	1	0	0	$\frac{12}{\pi^2}$
2	1.506	1.248	5.017	2	1.506	1.248	5.017
3 4 5 ⋮	$m - \frac{1}{2}$	$(m - \frac{1}{2})^2 P$	$(m - \frac{1}{2})^2 Q$	3 4 5 ⋮	$(n - \frac{1}{2})$	$(n - \frac{1}{2})^2 P$	$(n - \frac{1}{2})^2 Q$

$$P = [1 - 2/(m - 0.5)\pi]$$

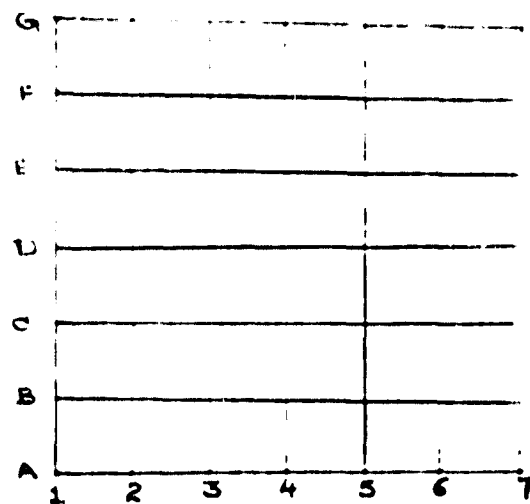
$$Q = [1 + 6/(m - 0.5)\pi]$$

	STRUDL				WARBURTON				LEPKE			
	NUMBER OF ELEMENTS				Nodal Pattern	Freq. (cps)	(m,n)	Nodal Pattern	Freq. (cps)	Nodal Pattern	Freq. (cps)	Nodal Pattern
	4	9	16	36								
1	.003165	.003284	.003315	.003314	.003358		(1,1)		.003471		.003287	
2	.004291	.004664	.004776	.004839	.004856		(2,0)		.004846		.004931	
3	.005549	.006133	.006221	.006221	.006207		(2,0)		.006175		.006175	
4	.007619	.008453	.008632	.008690	.008692		(2,1)		.00400		.015744	
5	.007619	.008453	.008632	.008690	.008692		(1,2)		.00400		.017167	
6	.01323	.01512	.01600	.01596	.01579		(3,0)		.01542		.01944	
7		.01512	.01600	.01596	.01579		(0,3)		.01542			
8		.01584	.01618	.01615	.01607		(2,2)		.01656			
9		.01630	.01727	.01750	.01746		(3,1)		.01715			
10		.01843	.02022	.02031	.02005		(1,3)		.0208			
11		.02465	.02693	.02693	.02700		(3,2)		.02726			
12		.02465	.02673	.02643	.02700		(2,3)		.02726			
13		.03362	.03029	.03162	.03104		(3,3)		.03941			

TABLE-III-2. Frequencies and Nodal Patterns Obtained by the Three Methods.

Case 1: 4 Elements

LOCATION	STRU DL	LEMKE	WARBURTON
A1	0	0	0
A2	1.0	1.0	1.0
B1	-1.0	-1.0	-1.0
B2	0	0	0



grid used for 36 elements

Case 2: 9 Elements

Case 3: 16 Elements

LOCATION	STRU DL	LEMKE	WARBURTON	STRU DL	LEMKE	WARBURTON
A1	0	0	0	0	0	0
A2	1.0	1.0	1.0	.7014	.7327	.6837
A3	1.0	1.0	1.0	1.0	1.0	1.0
B1	-1.0	-1.0	-1.0	-.7014	-.7327	-.6837
B2	0	0	0	0	0	0
B3	0	0	0	.2862	.2673	.3163
C1	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0
C2	0	0	0	-.2862	-.2673	-.3163
C3	0	0	0	0	0	0

TABLE-III-3. Normalized Deflections at the Nodal Points Obtained by the Three Methods - Second Mode.

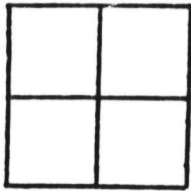
Case 4: 36 Elements

LOCATION	STRUDEL	WARBU- RTON	LEMKE	LOCATION	STRUDEL	WARBU- RTON	LEMKE
A <sub>1</sub>	0	0	0	C <sub>1</sub>	-.8590	-.8524	-.9793
A <sub>2</sub>	.4874	.4733	.5346	C <sub>2</sub>	-.3595	-.3792	-.3447
A <sub>3</sub>	.8590	.8524	.9793	C <sub>3</sub>	0	0	0
A <sub>4</sub>	1.0	1.0	1.0	C <sub>4</sub>	.1334	.1475	.1207
B <sub>1</sub>	-.4873	-.4733	-.5346	D <sub>1</sub>	-1.0	-1.0	-1.0
B <sub>2</sub>	0	0	0	D <sub>2</sub>	-.4938	-.5267	-.4654
B <sub>3</sub>	-.3594	-.3792	-.3447	D <sub>3</sub>	-.1334	-.1475	-.1207
B <sub>4</sub>	-.4933	-.5267	-.4654	D <sub>4</sub>	0	0	0

TABLE III-4. Normalized Deflections at the Nodal Points Obtained by the Three Methods - Second Mode.



Fig. 1 First Mode



(1,1)

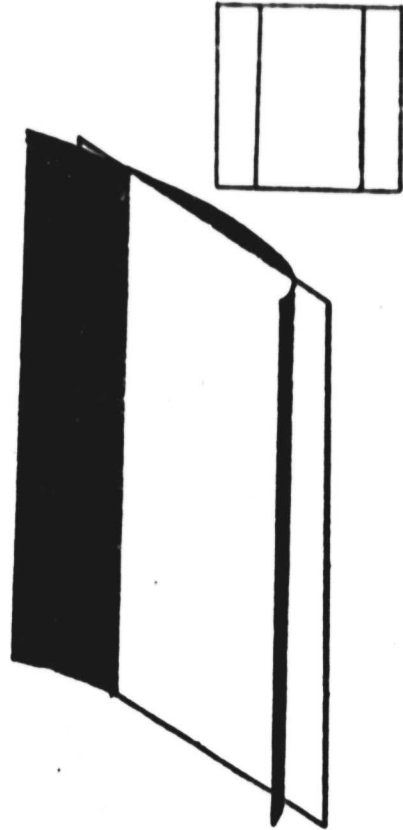
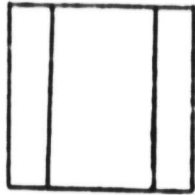


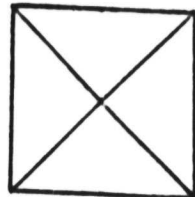
Fig. 3(a) (2,0) Mode



(2,0)



Fig. 2 Second Mode



(2,0)-(0,2)

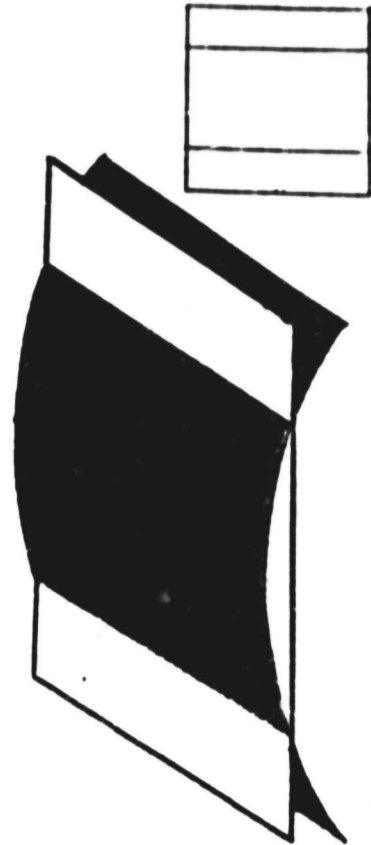


Fig 3(b) (0,2) Mode



(0,2)

ORIGINAL PAGE IS  
OF POOR QUALITY

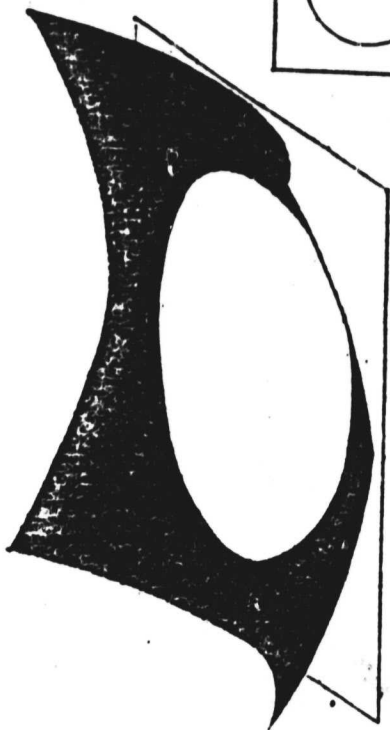
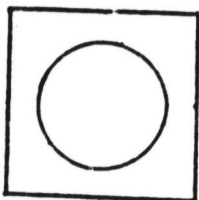


Fig. 4: Third Mode



(2,0) + (0,2)



Fig. 6: Sixth Mode



(3,0)

ORIGINAL PAGE IS  
OF POOR QUALITY

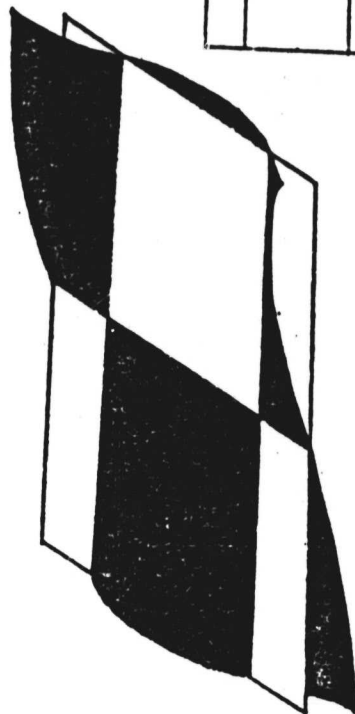
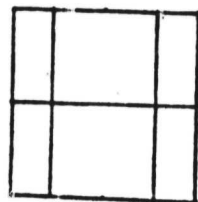


Fig. 5: Fourth Mode



(2,1)

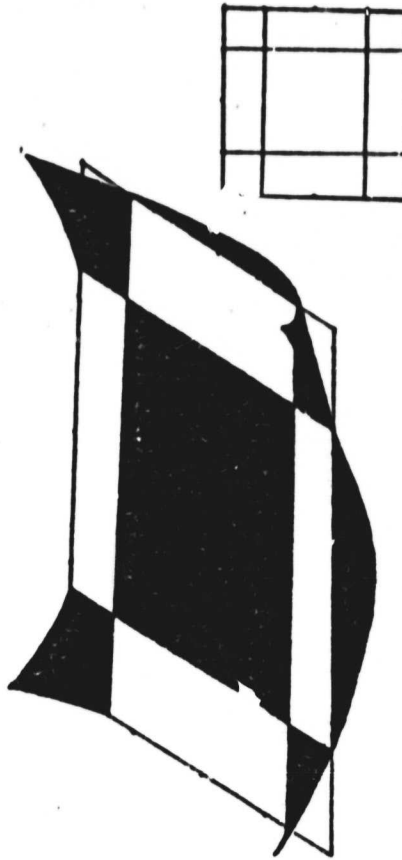
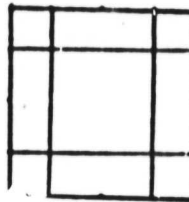
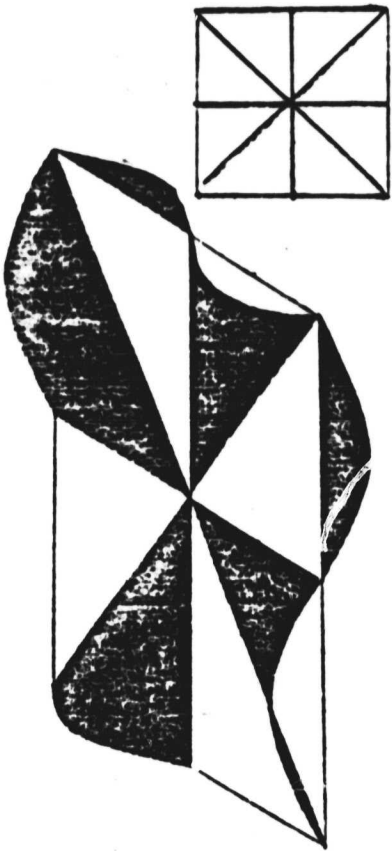


Fig. 7: Eighth Mode

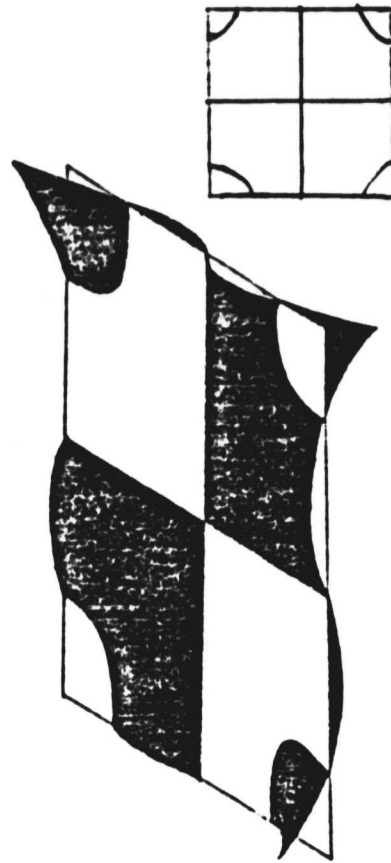


(2,2)



(1,3)-(3,1)

Fig. 8: Ninth Mode



(1,3)+(3,1)

Fig. 9: Tenth Mode



#### IV. GENERAL CONCLUSIONS AND RECOMMENDATIONS

A model is developed for predicting the dynamics of a large flexible free-free thin platform in orbit under the influence of control devices which are considered to be placed at specific locations on the major surface and one of the edges. Control about three different nominal orientations is considered. In the absence of control, for the case of a completely homogeneous platform instability in at least some of the modes is indicated for small amplitude motion about each of the three orientations. Once controllability is established, for a set of actuator locations, three different techniques are employed for the selection of actuator control laws:

- (1) the decoupling of the original state equations using state variable feedback;
- (2) a pole placement algorithm; and
- (3) an application of the linear regulator theory

It is seen that each of the three techniques have certain distinct advantages and also specific limitations), which are discussed in detail in Chapter II. For systems involving multi-degrees of freedom (such as in this application), the implementation of these techniques requires the extensive usage of computer algorithms.

As a logical extension to the present study which assumes perfect instantaneous knowledge of the state, the modelling of the sensor dynamics and related problem of observability should be considered, once specific information on the types of sensors required for monitoring the performance of large flexible systems is available.

The problems caused by both observation and control spillover could also be treated, perhaps by beginning with the simpler model of the control of a long, flexible beam in orbit and then extending this analysis to the three dimensional model of the platform.

A model of the uncontrolled dynamics of a large flexible shallow spherical shell (representative of an antenna dish or large radiometer) in orbit has been developed during the present grant year (see Part B this report). It is suggested that the effect of control devices be included in this model and that control laws could then be developed using different algorithms already in existence.

## APPENDIX

### Modifications to ORACLS Software Package

The ORACLS<sup>8</sup> Software Package that was developed at Langley which operates on the Control Data Cyber Computer System was modified to suit the IBM 370/165 Computer System that is available at Howard University. The major modifications that were done are described below:

- (1) As the single precision accuracy on the CDC is approximately equal to the double precision on the IBM/370 System, the entire package was converted into double precision.
- (2) Some of the machine dependent constants were changed accordingly.
- (3) As the IBM System accepts only six letters for a subroutine/function name all the names that exceeded six letters were changed and the list of those subroutines is given below:

	<u>Old Name</u>	<u>New Name</u>
(1)	TESTSTA	TESTSA
(2)	VARANCE	VARANC
(3)	TRANSIT	TRNSIT
(4)	DISCREG	DISREG
(5)	CNTNREG	CNTREG
(6)	RICTNWT	RICNWT
(7)	ASYMREG	ASMREG
(8)	ASYMFIL	ASMFIL
(9)	EXPMDFL	EXPMDF
(10)	IMPMDFL	IMPMDF

- (4) Some of the additional supporting subroutines/functions required were added and the names of these subroutines are given here:
  - (1) PNCH
  - (2) DIMAG
  - (3) DREAL
  - (4) BLOCK DATA

(5) None of the arguments of the subroutines were changed

The listing of the modified ORACLS package is given in the following pages. These routines have to be used in conjunction with Ref. (8). The numbers that appear in front of the FORTRAN statements are line numbers and have to be omitted.

1	SUBROUTINE RDTITL	RDT00010
2	IMPLICIT REAL*8 (A-H,O-Z)	RDT00020
3	COMMON/LINES/TITLE(10),TIL(3),NLP,LIN	RDT00030
4	COMMON/FORM/FMT1(2),FMT2(2),NEPR	RDT00040
5	COMMON/TOL/EPSSAM,EPSSBM,IACM	RDT00050
6	COMMON/CONV/SUMCV,RICTCV,SERCV,MAXSUM	RDT00060
7	C NLP = NO. LINES/PAGE VARIES WITH THE INSTALLATION	RDT00070
8	READ(5,100,END=90,ERR=91) TITLE	RDT00080
9	100 FORMAT(10A8)	RDT00090
10	CALL LNCNT(100)	RDT00100
11	RETURN	RDT00110
12	90 CONTINUE	RDT00120
13	STOP 1	RDT00130
14	91 CONTINUE	RDT00140
15	STOP 2	RDT00150
16	END	RDT00160

ORIGINAL FILED  
OF FOUR QUALITY

	SUBROUTINE LNCNT (N)	LNC00010
	IMPLICIT REAL*8 (A-H,O-Z)	LNC00020
2	COMMON/LINES/TITLE(10),TIL(3),NLP,LIN	LNC00030
	LIN=LIN+N	LNC00040
	IF (LIN.LE.NLP) GO TO 20	LNC00050
5	WRITE(6,1010) TITLE,TIL	LNC00060
6	1010 FORMAT(1H1,10A8,3A8/)	LNC00070
	LIN=2+N	LNC00080
	IF (N.GT.NLP) LIN=2	LNC00090
9	20 RETURN	LNC00100
	END	LNC00110

1	SUBROUTINE READ(I,A,NA,B,NB,C,NC,D,ND,E,NE)	REA00010
2	IMPLICIT REAL*8 (A-H,O-Z)	REA00020
3	DIMENSION A(1),B(1),C(1),D(1),E(1)	REA00030
4	DIMENSION NA(2),NB(2),NC(2),ND(2),NE(2),NZ(2)	REA00040
5	READ(5,100) LAB, NZ(1), NZ(2)	REA00050
6	CALL READ1(A, NA,NZ, LAB)	REA00060
7	IF(I .EQ. 1) GO TO 999	REA00070
8	READ(5,100) LAB, NZ(1), NZ(2)	REA00080
9	CALL READ1(B, NB,NZ, LAB)	REA00090
10	IF(I .EQ. 2) GO TO 999	REA00100
11	READ(5,100) LAB, NZ(1), NZ(2)	REA00110
12	CALL READ1(C, NC,NZ,LAB)	REA00120
13	IF(I .EQ. 3) GO TO 999	REA00130
14	READ(5,100) LAB, NZ(1), NZ(2)	REA00140
15	CALL READ1(D, ND,NZ, LAB)	REA00150
16	IF(I .EQ. 4) GO TO 999	REA00160
17	READ(5,100) LAB, NZ(1), NZ(2)	REA00170
18	CALL READ1(E, NE,NZ, LAB)	REA00180
19	100 FORMAT(A4,4X,2I4)	REA00190
20	999 RETURN	REA00200
21	END	REA00210

1	SUBROUTINE PRNT(A,NA,NAM,IOP)	PRN00010
2	IMPLICIT REAL*8 (A-H,O-Z)	PRN00020
3	DIMENSION A(1),NA(2)	PRN00030
4	COMMON /FORM/FMT1(2),FMT2(2),NEPR	PRN00040
5	COMMON/LINES/TITLE(10),TIL(3),NLP,LIN	PRN00050
6	C- NOTE NLP NO. LINES/PAGE VARIES WITH THE INSTALLATION.	PRN00060
7	DATA KZ,KW,KB /1H0,1H1,1H /	PRN00070
8	NAME = NAM	PRN00080
9	II = IOP	PRN00090
10	NR = NA(1)	PRN00100
11	NC = NA(2)	PRN00110
12	NLST = NR * NC	PRN00120
13	IF( NLST .LT. 1 .OR. NR .LT. 1 ) GO TO 16	PRN00130
14	IF(NAME .EQ. 0) NAME = KB	PRN00140
15	C- SKIP HEADLINE IF REQUESTED.	PRN00150
16	GO TO (11,10,132,12), II	PRN00160
17	10 CALL LNCNT(100)	PRN00170
18	11 CALL LNCNT(2)	PRN00180
19	3 WRITE(6,177) KZ,NAME,NR,NC	PRN00190
20	177 FORMAT(A1,5X,A4,8H MATRIX,5X,I3,5H ROWS,5X,I3,8H COLUMNS)	PRN00200
21	GO TO 13	PRN00210
22	12 CALL LNCNT(100)	PRN00220
23	GO TO 13	PRN00230
24	132 CALL LNCNT(2)	PRN00240
25	WRITE (6,891)	PRN00250
26	891 FORMAT (1H0)	PRN00260
27	C- BELOW COMPUTE NR OF LINES/ ROW --DECIDE IF 1 EXTRA BLANK LINE	PRN00270
28	13 J=(NC-1)/NEPR+1	PRN00280
29	C- WHY ALWAYS ADD 1 LINE- BECAUSE IF MULTIPLE, USE 1 BLK LINE EXTRA.	PRN00290
30	NLPW=J	
31	JST=1	
32	C COMPUTE LAST ROW POSITION =1 BELOW	
33	NLST = NLST -NR	PRN00330
34	MN=NC	PRN00340
35	IF (NC.GT.NEPR) MN=NEPR	PRN00350
36	KLST=NR*(MN-1)	PRN00360
37	91 CONTINUE	PRN00370
38	DO 912 J = JST, NR	PRN00380
39	CALL LNCNT(NLPW)	PRN00390
40	KLST = KLST +1	PRN00400
41	WRITE (6,FMT1) (A(N), N=J,KLST,NR)	PRN00410
42	IF (NC.LE.NEPR) GO TO 912	PRN00420
43	NLST = NLST +1	PRN00430
44	KNR=KLST+NR	PRN00440
45	WRITE (6,FMT2) (A(N), N=KNR,NLST,NR)	PRN00450
46	912 CONTINUE	PRN00460
47	RETURN	PRN00470
48	16 CALL LNCNT(1)	PRN00480
49	WRITE (6,916) NAM,NA	PRN00490
50	916 FORMAT (' ERROR IN PRNT MATRIX ',A4,' HAS NA=',2I6)	PRN00500
51	RETURN	PRN00510
52	END	PRN00520



	SUBROUTINE EQUATE(A,NA,R,NR)	EQU00010
	IMPLICIT REAL*4 (A-H,O-Z)	EQU00020
2	DIMENSION A(1),B(1),NA(2),NB(2)	EQU00030
	NR(1) = NA(1)	EQU00040
	NR(2) =NA(2)	EQU00050
5	L=NA(1)+NA(2)	EQU00060
6	IF( NA(1) .LT. 1 .OR. L .LT. 1 ) GO TO 999	EQU00070
	DO 300 I=1,L	EQU00080
	300 B(I)=A(I)	EQU00090
9	1000 RETURN	EQU00100
10	999 CALL LNCNT (1)	EQU00110
11	WRITE (6,50) NA	EQU00120
12	50 FORMAT (' DIMENSION ERROR IN EQUATE NA=',2I6)	EQU00130
13	RETURN	EQU00140
14	END	EQU00150

NATIONAL BUREAU OF STANDARDS  
 U.S. GOVERNMENT PRINTING OFFICE: 1967

SUBROUTINE TRANP(A,NA,R,NB)  
IMPLICIT REAL\*8 (A-H,O-Z)  
DIMENSION A(1),B(1),NA(2),NB(2)

NR(1)=NA(2)

NB(2)=NA(1)

NR=NA(1)

NC=NA(2)

L=NR+NC

IF( NR .LT. 1 .OR. L .LT. 1 ) GO TO 999

IR=0

DO 300 I=1, NR

IJ=I-NR

DO 300 J=1, NC

IJ=IJ+NR

IR=IR+1

300 B(IR)=A(IJ)

RETURN

999 CALL LMCNT(1)

WRITE (6,50) NA

50 FORMAT (' DIMENSION ERROR IN TRANP NA=',2I6)

RETURN

END

TRA00010  
TRA00020  
TRA00030  
TRA00040  
TRA00050  
TRA00060  
TRA00070  
TRA00080  
TRA00090  
TRA00100  
TRA00110  
TRA00120  
TRA00130  
TRA00140  
TRA00150  
TRA00160  
TRA00170  
TRA00180  
TRA00190  
TRA00200  
TRA00210  
TRA00220

```
1 SURROUTINE SCALE (A, NA, R, NB, S)
2 IMPLICIT REAL*8 (A-H,O-Z)
3 DIMENSION A(1),R(1),NA(2),NB(2)
4 NB(1) = NA(1)
5 NB(2) = NA(2)
6 L = NA(1)+NA(2)
7 IF( NA(1) .LT. 1 .OR. L .LT. 1 ) GO TO 999
8 DO 300 I=1,L
9 300 B(I)=A(I)*S
10 1000 RETURN
11 999 CALL LNCNT(1)
12 WRITE (6,50) NA
13 50 FORMAT (' DIMENSION ERROR IN SCALE NA=',2I6)
14 RETURN
15 END
```

```
SCA00010
SCA00020
SCA00030
SCA00040
SCA00050
SCA00060
SCA00070
SCA00080
SCA00090
SCA00100
SCA00110
SCA00120
SCA00130
SCA00140
SCA00150
```

0	SUBROUTINE UNITY(A,NA)	UNI00010
1	IMPLICIT REAL*8 (A-H,O-Z)	UNI00020
2	DIMENSION A(1),NA(2)	UNI00030
3	IF(NA(1).NE.NA(2)) GO TO 999	UNI00040
4	L=NA(1)+NA(2)	UNI00050
5	DO 100 IT=1,L	UNI00060
6	100 A(IT)=0.0	UNI00070
7	J = NA(1)	UNI00080
8	NAX = NA(1)	UNI00090
9	DO 300 I=1,NAX	UNI00100
10	J=NAX +J+1	UNI00110
11	300 A(J)=1.	UNI00120
12	GO TO 1000	UNI00130
13	999 CALL LNCNT (1)	UNI00140
14	WRITE(6, 50)(NA(I),I=1,2)	UNI00150
15	50 FORMAT (' DIMENSION ERROR IN UNITY NA=',2I6)	UNI00160
16	1000 RETURN	UNI00170
17	END	UNI00180

0	SUBROUTINE NULL(A,NA)	NUL00010
1	IMPLICIT REAL*8 (A-H,O-Z)	NUL00020
2	DIMENSION A(1)	NUL00030
3	DIMENSION NA(2)	NUL00040
4	N=NA(1)+NA(2)	NUL00050
5	IF( NA(1) .LT. 1 .OR. N .LT. 1 ) GO TO 999	NUL00060
6	DO 10 I=1,N	NUL00070
7	10 A(I) = 0.0	NUL00080
8	RETURN	NUL00090
9	C	NUL00100
10	999 CONTINUE	NUL00110
11	WRITE (6,50) NA	NUL00120
12	50 FORMAT(' DIMENSION ERROR IN NULL NA =',2I6)	NUL00130
13	RETURN	NUL00140
14	END	NUL00150

0	SUBROUTINE TRCE (A,NA,TR)	TRC00010
1	IMPLICIT REAL*8 (A-M,U-Z)	TRC00020
2	DIMENSION A(1)	TRC00030
3	DIMENSION NA(2)	TRC00040
4	IF (NA(1).NE.NA(2)) GO TO 600	TRC00050
5	N=NA(1)	TRC00060
6	TR=0.0	TRC00070
7	IF( N .LT. 1 ) GO TO 600	TRC00080
8	DO 10 I=1,N	TRC00090
9	M=I+N*(I-1)	TRC00100
10	10 TR=TR+A(M)	TRC00110
	RETURN	TRC00120
11	600 CALL LNCNT(1)	TRC00130
12	WRITE (6,1600) NA	TRC00140
13	1600 FORMAT (' TRACE REQUIRES SQUARE MATRIX    NA=',2I6)	TRC00150
14	RETURN	TRC00160
15	END	TRC00170

0	SUBROUTINE ADD (A,NA,B,NB,C,NC)	ADD00010
	IMPLICIT REAL*8 (A-H,O-Z)	ADD00020
	DIMENSION A(1),B(1),C(1),NA(2),NB(2),NC(2)	ADD00030
3	IF( (NA(1) .NE. NB(1)) .OR. (NA(2) .NE. NB(2)) ) GO TO 999	ADD00040
4	NC(1)=NA(1)	ADD00050
	NC(2)=NA(2)	ADD00060
6	L=NA(1)+NA(2)	ADD00070
7	IF( NA(1) .LT. 1 .OR. L .LT. 1 ) GO TO 999	ADD00080
	DO 300 I=1,L	ADD00090
300	C(I)=A(I)+B(I)	ADD00100
10	GO TO 1000	ADD00110
999	CALL LNCNT (1)	ADD00120
	WRITE(6,50) NA,NB	ADD00130
13	50 FORMAT (' DIMENSION ERROR IN ADD      NA=',2I6,5X,'NB=',2I6)	ADD00140
14	1000 RETURN	ADD00150
	END	ADD00160

	SUBROUTINE SUBT(A,NA,B,NB,C,NC)	SUB00010
	IMPLICIT REAL*8 (A-H,O-Z)	SUB00020
2	DIMENSION A(1),B(1),C(1),NA(2),NB(2),NC(2)	SUB00030
3	IF((NA(1).NE.NB(1)),OR.(NA(2).NE.NB(2))) GO TO 999	SUB00040
	NC(1)=NA(1)	SUB00050
	NC(2)=NA(2)	SUB00060
6	L=NA(1)*NA(2)	SUB00070
	IF( NA(1) .LT. 1 .OR. L .LT. 1 ) GO TO 999	SUB00080
	DO 300 I=1,L	SUB00090
9	300 C(I)=A(I)-B(I)	SUB00100
	GO TO 1000	SUB00110
	999 CALL LNCNT (1)	SUB00120
12	WRITE(6,50) NA,NB	SUB00130
13	50 FORMAT (' DIMENSION ERROR IN SUBT    NA=',2I6,5X,'NB=',2I6)	SUB00140
	1000 RETURN	SUB00150
	END	SUB00160



0	SUBROUTINE MULT(A,NA,B,NB,C,NC)	MUL0001
1	IMPLICIT REAL*8 (A-H,O-Z)	MUL0002
2	DIMENSION A(1),B(1),C(1),NA(2),NB(2),NC(2)	MUL0003
3	NC(1) = NA(1)	MUL0004
4	NC(2) = NB(2)	MUL0005
5	IF(NA(2).NE.NB(1)) GO TO 999	MUL0006
6	NAR = NA(1)	MUL0007
7	NAC = NA(2)	MUL0008
8	NRB = NB(1)	MUL0009
9	NAB = NB(2)	MUL0010
10	NBB = NAR*NAC	MUL0011
11	NBB = NAR*NAC	MUL0011
11	IF ( NAR .LT. 1 .OR. NAA .LT. 1 .OR. NBB .LT. 1 ) GO TO 999	MUL0012
12	IR = 0	MUL0013
13	IK = -NAC	MUL0014
14	DO 350 K=1,NBC	MUL0015
15	IK = IK + NAC	MUL0016
16	DO 350 J=1,NAR	MUL0017
17	IR=IR+1	MUL0018
18	IB=IK	MUL0019
19	J1=J-NAR	MUL0020
20	V1=0.0	MUL0021
21	DO 300 I=1,NAC	MUL0022
22	J1 = J1 + NAR	MUL0023
23	IB=IB+1	MUL0024
24	V3=A(J1)	MUL0025
25	V4=B(IB)	MUL0026
26	V2=V3+V4	MUL0027
27	V1=V1+V2	MUL0028
28	300 CONTINUE	MUL0029
29	C(IR)=V1	MUL0030
30	350 CONTINUE	MUL0031
31	GO TO 1000	MUL0032
32	999 CALL LNCNT (1)	MUL0033
33	WRITE(6,500) (NA(I),I=1,2),(NB(I),I=1,2)	MUL0034
34	500 FORMAT (' DIMENSION ERROR IN MULT NA=',2I6,5X,'NB=',2I6)	MUL0035
35	1000 RETURN	MUL0036
36	END	MUL0037

0	SUBROUTINE MAXEL(A,NA,ELMAX)	MAX0001
1	IMPLICIT REAL*8 (A-H,O-Z)	MAX0002
2	DIMENSION A(1),NA(2)	MAX0003
3	C	MAX0004
4	N = NA(1)*NA(2)	MAX0005
5	C	MAX0006
6	ELMAX = DABS( A(1) )	MAX0007
7	DO 100 I = 2,N	MAX0008
8	ELMAXI = DABS( A(1) )	MAX0009
9	IF( ELMAXI.GT.ELMAX) ELMAX=ELMAXI	
10	100 CONTINUE	MAX0011
11	C	MAX0012
12	RETURN	MAX0013
13	END	MAX0014

0	SUBROUTINE NORMS(MAXROW,M,N,A,IOPT,RLNORM)	NOR0001
1	IMPLICIT REAL*4 (A-M,O-Z)	NOR0002
2	DIMENSION A(1)	NOR0003
3	C	NOR0004
4	C INITIALIZATION	NOR0005
5	C	NOR0006
6	RLNORM=0.	NOR0007
7	SUM=0.	NOR0008
8	I=MAXROW	NOR0009
9	C	NOR0010
10	C TRANSFER TO APPROPRIATE LOOP TO COMPUTE THE DESIRED NORM	NOR0011
11	C	NOR0012
12	IF(IOPT-2)5,20,30	NOR0013
13	C	NOR0014
14	C THIS LOOP COMPUTES THE ONE-NORM	NOR0015
15	C	NOR0016
16	5 DO 15 K=1,N	NOR0017
17	I=I+MAXROW	NOR0018
18	DO 10 J=1,M	NOR0019
19	L=I+J	NOR0020
20	10 SUM=0ABS(A(L))+SUM	NOR0021
21	IF(SUM.GT.RLNORM)RLNORM=SUM	NOR0022
22	15 SUM=0.	NOR0023
23	RETURN	NOR0024
24	C	NOR0025
25	C THIS LOOP COMPUTES THE EUCLIDEAN NORM	NOR0026
26	C	NOR0027
27	20 DO 25 K=1,N	NOR0028
28	I=I+MAXROW	NOR0029
29	DO 25 J=1,M	NOR0030
30	L=I+J	NOR0031
31	SUM=A(L)	NOR0032
32	25 RLNORM=SUM*SUM+RLNORM	NOR0033
33	RLNORM=DSQRT(RLNORM)	NOR0034
34	RETURN	NOR0035
35	C	NOR0036
36	C THIS LOOP COMPUTES THE INFINITY-NORM	NOR0037
37	C	NOR0038
38	30 DO 40 J=1,M	NOR0039
39	L=I+J	NOR0040
40	DO 35 K=1,N	NOR0041
41	L=L+MAXROW	NOR0042
42	35 SUM=0ABS(A(L))+SUM	NOR0043
43	IF(SUM.GT.RLNORM)RLNORM=SUM	NOR0044
44	40 SUM=0.0	NOR0045
45	RETURN	NOR0046
46	END	NOR0047

0	SUBROUTINE JUXTC(A,NA,B,NB,C,NC)	JUX0001
1	IMPLICIT REAL*8 (A-H,O-Z)	JUX0002
2	DIMENSION A(1),B(1),C(1),NA(2),NB(2),NC(2)	JUX0003
3	IF (NA(1).NE.NB(1)) GO TO 600	JUX0004
4	NC(1)=NA(1)	JUX0005
5	NC(2)=NA(2)+NB(2)	JUX0006
6	L=NA(1)+NA(2)	JUX0007
7	NNC=NC(1)+NC(2)	JUX0008
8	IF( NA(1) .LT. 1 .OR. L .LT. 1 ) GO TO 600	JUX0009
9	IF( NC(2) .LT. 1 ) GO TO 600	JUX0010
10	MS=NA(1)+NA(2)	JUX0011
11	DO 10 I=1,MS	JUX0012
12	10 C(I)=A(I)	JUX0013
13	MBS=NA(1)+NB(2)	JUX0014
14	DO 20 I=1,MBS	JUX0015
15	J=MS+I	JUX0016
16	20 C(J)=B(I)	JUX0017
17	RETURN	JUX0018
18	600 CALL LNCNT(1)	JUX0019
19	WRITE (6,1600) NA,NB	JUX0020
20	1600 FORMAT (' DIMENSION ERROR IN JUXTC, NA=',2I6,5X,'NB=',2I6)	JUX0021
21	RETURN	JUX0022
22	END	JUX0023

0	SUBROUTINE JUXTR(A,NA,B,NB,C,NC)	JUX0001
1	IMPLICIT REAL*8 (A-H,O-Z)	JUX0002
2	DIMENSION A(1),B(1),C(1),NA(2),NB(2),NC(2)	JUX0003
3	IF(NA(2).NE.NB(2))GO TO 600	JUX0004
4	NC(2)=NA(2)	JUX0005
5	NC(1)=NA(1)+NB(1)	JUX0006
6	L=NA(1)+NA(2)	JUX0007
7	IF( NA(1) .LT. 1 .OR. L .LT. 1 ) GO TO 600	JUX0008
8	IF( NC(2) .LT. 1 ) GO TO 600	JUX0009
9	MCA=NA(2)	JUX0010
10	MRA=NA(1)	JUX0011
11	MRB=NB(1)	JUX0012
12	MRC=NC(1)	JUX0013
13	DO 10 I=1,MCA	JUX0014
14	DO 10 J=1,MRA	JUX0015
15	K=J+MRA*(I-1)	JUX0016
16	L=J+MRC*(I-1)	JUX0017
17	10 C(L)=A(K)	JUX0018
18	DO 20 I=1,MCA	JUX0019
19	DO 20 J=1,MRB	JUX0020
20	K=J+MRB*(I-1)	JUX0021
21	L=MRA+J+MRC*(I-1)	JUX0022
22	20 C(L)=B(K)	JUX0023
23	RETURN	JUX0024
24	600 CALL LNCNT(1)	JUX0025
25	WRITE(6,1600) NA,NB	JUX0026
26	1600 FORMAT(' DIMENSION ERROR IN JUXTR, NA=',2I6,5X,'NB=',2I6)	JUX0027
27	RETURN	JUX0028
28	END	JUX0029

0	SUBROUTINE FACTOR(Q,NQ,D,ND,IOP,IAC,DUMMY)	FAC0001
1	IMPLICIT REAL*8 (A-H,O-Z)	FAC0002
2	DIMENSION Q(1),D(1),DUMMY(1)	FAC0003
3	DIMENSION NQ(2),ND(2),NDUM(2)	FAC0004
4	IOPT = 2	FAC0005
5	N = NQ(1)	FAC0006
6	M = N**2	FAC0007
7	N1 = M + 1	FAC0008
8	N2 = N1 + N	FAC0009
9	C	FAC0010
10	CALL EQUATE(Q,NQ,DUMMY,NQ)	FAC0011
11	CALL SNVDEC(IOPT,N,N,N,N,DUMMY,NOS,B,IAC,ZTEST,DUMMY(N1),D,IRANK,	FAC0012
12	1PLUS,IERR)	FAC0013
13	IF( IERR .EQ. 0 ) GO TO 200	FAC0014
14	CALL LNCNT(5)	FAC0015
15	IF( IERR .GT. 0 ) PRINT 100,IERR	FAC0016
16	IF( IERR .EQ. -1) PRINT 150,ZTEST,IRANK	FAC0017
17	100 FORMAT(//,' IN FACTOR , SNVDEC HAS FAILED TO CONVERGE TO THE ',I4,	FAC0018
18	1' SINGULAR VALUE AFTER 30 ITERATIONS')	FAC0019
19	150 FORMAT(//,' IN FACTOR, THE MATRIX Q SUBMITTED TO SNVDEC IS CLOSE TO	FAC0020
20	10 A MATRIX OF LOWER RANK USING ZTEST = ',D16.8,/,', IF THE ACCURACY	FAC0021
21	2 IS REDUCED THE RANK MAY ALSO BE REDUCED',/,', CURRENT RANK = ',I4)	FAC0022
22	NDUM(1)=N	FAC0023
23	NDUM(2)=1	FAC0024
24	IF(IERR .EQ. -1) CALL PRNT(DUMMY(N1),NDUM,4H\$NVL,1)	FAC0025
25	IF( IERR .GT. 0 ) RETURN	FAC0026
26	C	FAC0027
27	200 CONTINUE	FAC0028
28	NDUM(1) = N	FAC0029
29	C	FAC0030
30	DO 250 J = 1,N	FAC0031
31	M1 = (J-1)*N + 1	FAC0032
32	M2 = J*N	FAC0033
33	DO 250 I = M1,M2	FAC0034
34	K = N2+I-1	FAC0035
35	L = N1+J-1	FAC0036
36	IF( DUMMY(L) .EQ. 0.0) GO TO 300	FAC0037
37	DUMMY(K) = DSQRT(DUMMY(L))*DUMMY(I)	FAC0038
38	250 CONTINUE	FAC0039
39	NDUM(2)=N	FAC0040
40	GO TO 350	FAC0041
41	C	FAC0042
42	300 NDUM(2) = J - 1	FAC0043
43	350 CONTINUE	FAC0044
44	IF( DUMMY(N2) .LT. 0.0 ) CALL SCALE(DUMMY(N2),NDUM,DUMMY(N2),NDUM,	FAC0045
45	1=L.0)	FAC0046
46	CALL TRANP(DUMMY(N2),NDUM,D,ND)	FAC0047
47	C	FAC0048
48	IF( IOP .EQ. 0 ) RETURN	FAC0049
49	CALL LNCNT(4)	FAC0050
50	PRINT 400	FAC0051
51	400 FORMAT(//,' FACTOR Q AS (D TRANSPOSE)XD ',/)	FAC0052
52	CALL PRNT(Q,NQ,4H Q ,1)	FAC0053
53	CALL PRNT(D,ND,4H D ,1)	FAC0054
54	CALL MULT(DUMMY(N2),NDUM,D,ND,DUMMY,NQ)	FAC0055
55	CALL PRNT(DUMMY,NQ,4HD\$XD,1)	FAC0056
56	C	FAC0057
57	RETURN	FAC0058
58	END	FAC0059

0		SUBROUTINE EIGEN(MAX, N, A, ER, EI, ISV, ILV, V, WK, IERR)	EIG0001
1		IMPLICIT REAL*8 (A-H,O-Z)	EIG0002
2		DIMENSION A(MAX,N),ER(N),EI(N),V(MAX,1),WK(N,1)	EIG0003
3		INTEGER INT(20)	EIG0004
4		LOGICAL*1 SELECT(25)	EIG0005
5	C		EIG0006
6	C	PRELIMINARY REDUCTION	EIG0007
7	C		EIG0008
8		CALL BALANC (MAX,N,A,LOW,IGH,WK)	EIG0009
9		CALL ELMHES (MAX,N,LOW,IGH,A,INT(1))	EIG0010
10		IV = ISV + ILV	EIG0011
11		IF (IV .NE. 0) GO TO 10	EIG0012
12	C		EIG0013
13	C	COMPUTE ALL EIGENVALUES AND NO EIGENVECTORS	EIG0014
14	C		EIG0015
15		CALL HQR (MAX,N,LOW,IGH,A,ER,EI,IERR)	EIG0016
16		IF (IERR .NE. 0) GO TO 260	EIG0017
17		DO 5 I=1,N	EIG0018
18		WK(I,1) = ER(I)	EIG0019
19		WK(I,2) = EI(I)	EIG0020
20		WK(I,3) = ER(I)**2 + EI(I)**2	EIG0021
21	5	CONTINUE	EIG0022
22		IC = 0	EIG0023
23		GO TO 190	EIG0024
24	10	CONTINUE	EIG0025
25	C		EIG0026
26	C	SAVE A MATRIX FOR INVERSE ITERATION AND INITIALIZE WK(I,4)	EIG0027
27	C	ARRAY WHICH WILL BE A LOGICAL ARRAY IN CALLED SUBROUTINES	EIG0028
28	C		EIG0029
29		DO 20 I=1,N	EIG0030
30		SELECT(I)=.FALSE.	EIG0031
31		JS = 1	EIG0032
32		IF (I .GE. 3) JS = I-1	EIG0033
33		DO 20 J=JS,N	EIG0034
34		WK(I,J+5) = A(I,J)	EIG0035
35	20	CONTINUE	EIG0036
36	C		EIG0037
37	C	COMPUTE ALL EIGENVALUES (UNORDERED)	EIG0038
38	C		EIG0039
39		CALL HQR (N,N,LOW,IGH,WK(1,6),ER,EI,IERR)	EIG0040
40		IF (IERR .NE. 0) GO TO 260	EIG0041
41		DO 30 I=1,N	EIG0042
42		WK(I,3) = ER(I)**2 + EI(I)**2	EIG0043
43	30	CONTINUE	EIG0044
44		IF (ILV .EQ. 0) GO TO 60	EIG0045
45	C		EIG0046
46	C	FIND LARGEST ILV EIGENVALUES AND FLAG THEM	EIG0047
47	C		EIG0048
48		DO 50 I=1,ILV	EIG0049
49		P = -1.00	EIG0050
50		DO 40 J=1,N	EIG0051
51		IF (WK(J,3) .LE. P) GO TO 40	EIG0052
52		K = J	EIG0053
53		P = WK(J,3)	EIG0054
54	40	CONTINUE	EIG0055
55		SELECT(K)=.TRUE.	EIG0056
56		WK(K,3) = -WK(K,3)	EIG0057
57	50	CONTINUE	EIG0058
58		IF (EI(K) .EQ. 0.) GO TO 60	EIG0059
59		IF (EI(K) .GT. 0.) GO TO 55	EIG0060
60		IF (SELECT(K-1))GO TO 60	EIG0061
61		ILV = ILV+1	EIG0062
62		SELECT(K-1)=.TRUE.	EIG0063

MICROFILM PAGE 15  
 OF 100 QUALITY

53		GO TO 60	EIG0064
54	55	CONTINUE	EIG0065
55		IF (.NOT.SELECT(K+1)) ILV = ILV+1	EIG0066
56	60	CONTINUE	EIG0067
57		IF (ISV .EQ. 0) GO TO 90	EIG0068
58	C		EIG0069
59	C	FIND SMALLEST ISV EIGENVALUES AND FLAG THEM	EIG0070
70	C		EIG0071
71		DO 65 I=1,N	EIG0072
72		WK(I,3) = DARS(WK(I,3))	EIG0073
73	65	CONTINUE	EIG0074
74		DO 80 I=1,ISV	EIG0075
75		P = 1.074	EIG0076
76		DO 70 J=1,N	EIG0077
77		IF (WK(J,3) .GE. P) GO TO 70	EIG0078
78		K = J	EIG0079
79		P = WK(J,3)	EIG0080
80	70	CONTINUE	EIG0081
81		SELECT(K) = .TRUE.	EIG0082
82		WK(K,3) = 1.074	EIG0083
83	80	CONTINUE	EIG0084
84		IF (EI(K) .EQ. 0.) GO TO 90	EIG0085
85		IF (EI(K) .GT. 0.) GO TO 85	EIG0086
86		IF (SELECT(K-1)) GO TO 90	EIG0087
87		ISV = ISV+1	EIG0088
88		SELECT(K-1) = .TRUE.	EIG0089
89		GO TO 90	EIG0090
90	85	CONTINUE	EIG0091
91		IF (.NOT.SELECT(K+1)) ISV = ISV+1	EIG0092
92	90	CONTINUE	EIG0093
93	C		EIG0094
94	C	FIND EIGENVECTORS FOR FLAGGED EIGENVALUES	EIG0095
95	C		EIG0096
96		CALL INVT (MAX,N,A,ER,EI,SELECT,N,M,V,IERR,WK(1,6),WK(1,3),	EIG0097
97	1	WK(1,5))	EIG0098
98	C		EIG0099
99	C	BACK TRANSFORM EIGENVECTORS TO ORIGINAL MATRIX	EIG0100
100	C		EIG0101
101		CALL ELMBAK (MAX,LOW,IGH,A,INT(1),M,V)	EIG0102
102		CALL BALBAK (MAX,N,LOW,IGH,WK,M,V)	EIG0103
103	C		EIG0104
104	C	SEPARATE FLAGGED EIGENVALUES FROM UNFLAGGED EIGENVALUES	EIG0105
105	C		EIG0106
106		IV = ISV + ILV	EIG0107
107		IF (IV .LE. N) GO TO 100	EIG0108
108		ILV = N-IV	EIG0109
109		IV = N	EIG0110
110	100	CONTINUE	EIG0111
111		IC = 0	EIG0112
112		JC = IV	EIG0113
113		DO 150 I=1,N	EIG0114
114		IF (SELECT(I)) GO TO 120	EIG0115
115		IF (EI(I) .GE. 0.) GO TO 110	EIG0116
116		IF (SELECT(I-1)) GO TO 120	EIG0117
117	110	CONTINUE	EIG0118
118		JC = JC+1	EIG0119
119		WK(JC,1) = ER(I)	EIG0120
120		WK(JC,2) = EI(I)	EIG0121
121		KC = JC	EIG0122
122		GO TO 130	EIG0123
123	120	CONTINUE	EIG0124
124		IC = IC+1	EIG0125
125		WK(IC,1) = ER(I)	EIG0126



126		WK(IC,2) = EI(I)	EIG012
127		IC = IC	EIG012
128	130	CONTINUE	EIG012
129		WK(KC,3) = ER(I)**2 + EI(I)**2	EIG013
130	150	CONTINUE	EIG013
131	C		EIG013
132	C	NORMALIZE VECTORS TO UNIT LENGTH AND STORE FOR REORDERING	EIG013
133	C		EIG013
134		J = 0	EIG013
135	151	CONTINUE	EIG013
136		J = J+1	EIG013
137		IF (WK(J,2) .NE. 0.) GO TO 154	EIG013
138		SUM = 0.	EIG013
139		DO 152 I=1,N	EIG014
140		SUM = SUM + V(I,J)**2	EIG014
141	152	CONTINUE	EIG014
142		IF (SUM .EQ. 0.) GO TO 158	EIG014
143		SUM = DSQRT(SUM)	EIG014
144		DO 153 I=1,N	EIG014
145		WK(I,J+4) = V(I,J)/SUM	EIG014
146	153	CONTINUE	EIG014
147		GO TO 158	EIG014
148	154	CONTINUE	EIG014
149		JP1 = J+1	EIG015
150		SUM = 0.	EIG015
151		DO 155 I=1,N	EIG015
152		SUM = SUM + V(I,J)**2 + V(I,JP1)**2	EIG015
153	155	CONTINUE	EIG015
154		IF (SUM .EQ. 0.) GO TO 157	EIG015
155		SUM = DSQRT(SUM)	EIG015
156		DO 156 I=1,N	EIG015
157		WK(I,J+4) = V(I,J)/SUM	EIG015
158		WK(I,J+5) = V(I,JP1)/SUM	EIG015
159	156	CONTINUE	EIG016
160	157	CONTINUE	EIG016
161		J = JP1	EIG016
162	158	CONTINUE	EIG016
163		IF (J .LT. IV) GO TO 151	EIG016
164		IC = 0	EIG016
165		LC = 0	EIG016
166		IF (ISV .EQ. 0) GO TO 190	EIG016
167	C		EIG016
168	C	ORDER SMALLEST ISV EIGENVALUES AND EIGENVECTORS FOR OUTPUT	EIG016
169	C		EIG017
170		DO 190 I=1,ISV	EIG017
171		P = 1.074	EIG017
172		DO 160 J=1,IV	EIG017
173		IF (WK(J,3) .GE. P) GO TO 160	EIG017
174		K = J	EIG017
175		P = WK(J,3)	EIG017
176	160	CONTINUE	EIG017
177		IC = IC+1	EIG017
178		LC = LC+1	EIG017
179		ER(IC) = WK(K,1)	EIG018
180		EI(IC) = WK(K,2)	EIG018
181		DO 170 J=1,N	EIG018
182		V(J,LC) = WK(J,K+4)	EIG018
183	170	CONTINUE	EIG018
184		WK(K,3) = 1.074	EIG018
185	180	CONTINUE	EIG018
186	190	CONTINUE	EIG018
187		IF (IV .EQ. N) GO TO 220	EIG018
188	C		EIG018

89	C	ORDER UNFLAGGED EIGENVALUES FOR OUTPUT	EIG0190
90	C		EIG0191
91		IV1 = IV+1	EIG0192
92		IUF = N - IV	EIG0193
93		DO 210 I=1,IUF	EIG0194
94		P = 1.074	EIG0195
95		DO 200 J=IV1,N	EIG0196
96		IF (WK(J,3) .GE. P) GO TO 200	EIG0197
97		K = J	EIG0198
98		P = WK(J,3)	EIG0199
99	200	CONTINUE	EIG0200
100		IC = IC+1	EIG0201
201		ER(IC) = WK(K,1)	EIG0202
202		EI(IC) = WK(K,2)	EIG0203
203		WK(K,3) = 1.074	EIG0204
204	210	CONTINUE	EIG0205
205	220	CONTINUE	EIG0206
206		IF (ILV .EQ. 0) GO TO 260	EIG0207
207	C		EIG0208
208	C	ORDER LARGEST ILV EIGENVALUES AND EIGENVECTORS FOR OUTPUT	EIG0209
209	C		EIG0210
210		DO 250 I=1,ILV	EIG0211
211		P = 1.074	EIG0212
212		DO 230 J=1,IV	EIG0213
213		IF (WK(J,3) .GE. P) GO TO 230	EIG0214
214		K = J	EIG0215
215		P = WK(J,3)	EIG0216
216	230	CONTINUE	EIG0217
217		IC = IC+1	EIG0218
218		LC = LC+1	EIG0219
219		ER(IC) = WK(K,1)	EIG0220
220		EI(IC) = WK(K,2)	EIG0221
221		DO 240 J=1,N	EIG0222
222		V(J,LC) = WK(J,K+4)	EIG0223
223	240	CONTINUE	EIG0224
224		WK(K,3) = 1.074	EIG0225
225	250	CONTINUE	EIG0226
226	260	CONTINUE	EIG0227
227		RETURN	EIG0228
228		END	EIG0229

0	SUBROUTINE SYMPOD (MAXN,N,A,NRHS,B,IOP,IFAC,DETERM,ISCALE,P,IERR)	SYM000
1	IMPLICIT REAL*8 (A-H,O-Z)	SYM000
2	DIMENSION A(MAXN,N),B(MAXN,NRHS),P(N)	SYM000
3	C	SYM000
4	DATA R1,R2/1.00+75,1.00-75/	SYM000
5	C	SYM000
6	C TEST FOR A SCALAR MATRIX (IF COEFFICIENT MATRIX IS A	SYM000
7	C SCALAR-- SOLVE AND COMPUTE DETERMINANT IF DESIRED)	SYM000
8	IERR = 0	SYM000
9	NM1 = N-1	SYM001
10	IF (NM1 .GT. 0) GO TO 20	SYM001
11	C	SYM001
12	IF( A(1,1) .LE. 0.0 ) IERR = 1	SYM001
13	ISCALE = 0	SYM001
14	DETERM = A(1,1)	SYM001
15	P(1) = 1.0/A(1,1)	SYM001
16	DO 10 J=1,NRHS	SYM001
17	B(1,J) = B(1,J)/DETERM	SYM001
18	10 CONTINUE	SYM001
19	RETURN	SYM002
20	C	SYM002
21	C TEST TO DETERMINE IF CHOLESKY DECOMPOSITION OF COEFFICIENT	SYM002
22	C MATRIX IS DESIRED	SYM002
23	20 IF (IFAC .EQ. 1) GO TO 160	SYM002
24	C	SYM002
25	C INITIALIZE DETERMINANT EVALUATION PARAMETERS	SYM002
26	DETERM=1.0	SYM002
27	ISCALE=0	SYM002
28	C	SYM002
29	C 'LOOP' TO PERFORM CHOLESKY DECOMPOSITION ON THE COEF-	SYM003
30	C FICIENT MATRIX A (I.E. MATRIX A WILL BE DECOMPOSED INTO	SYM003
31	C THE PRODUCT OF A UNIT LOWER TRIANGULAR MATRIX (L), A	SYM003
32	C DIAGONAL MATRIX (D), AND THE TRANSPOSE OF L (LTRANSPOSE.)	SYM003
33	C	SYM003
34	30 DO 150 I=1,N	SYM003
35	IM1 = I-1	SYM003
36	C	SYM003
37	DO 150 J=1,I	SYM003
38	X = A(J,I)	SYM003
39	C	SYM004
40	C DETERMINE IF ELEMENTS ARE ABOVE OR BELOW THE DIAGONAL	SYM004
41	IF (I .GT. J) GO TO 110	SYM004
42	C	SYM004
43	C USING THE DIAGONAL ELEMENTS OF MATRIX A, THIS SECTION	SYM004
44	C COMPUTES DIAGONAL MATRIX AND DETERMINES IF MATRIX A IS	SYM004
45	C SYMMETRIC POSITIVE DEFINITE	SYM004
46	IF (IM1 .EQ. 0) GO TO 50	SYM004
47	DO 40 K=1,IM1	SYM004
48	Y = A(I,K)	SYM004
49	A(I,K) = Y*P(K)	SYM005
50	X = X - Y*A(I,K)	SYM005
51	40 CONTINUE	SYM005
52	C	SYM005
53	C TEST IF COEFFICIENT MATRIX IS POSITIVE DEFINITE	SYM005
54	50 IF( X .LE. 0.0 ) IERR = 1	SYM005
55	C	SYM005
56	C COMPUTE INVERSE OF DIAGONAL MATRIX D**-1 = 1/P	SYM005
57	P(I) = 1.0 / X	SYM005
58	C	SYM005
59	C TEST TO SEE IF DETERMINANT IS TO BE EVALUATED	SYM006
60	IF (IOP, .EQ. 0) GO TO 150	SYM006
61	C	SYM006
62	C	SYM006

6	C	SCALE THE DETERMINANT (COMPUTE THE DETERMINANT EVALUATION	SYM0064
64	C	PARAMETERS DETERM AND ISCALE)	SYM0065
65		PIVOTI=X	SYM0066
6		60 IF(DABS(DETERM),LT,R1) GO TO 70	SYM0067
6		DETERM = DETERM*R2	SYM0068
68		ISCALE = ISCALE+1	SYM0069
6		GO TO 60	SYM0070
7		70 IF(DABS(DETERM),GT,R2) GO TO 80	SYM0071
71		DETERM = DETERM*R1	SYM0072
72		ISCALE = ISCALE-1	SYM0073
7		GO TO 70	SYM0074
7		80 IF(DABS(PIVOTI),LT,R1) GO TO 90	SYM0075
75		PIVOTI = PIVOTI*R2	SYM0076
7		ISCALE = ISCALE+1	SYM0077
7		GO TO 80	SYM0078
78		90 IF(DABS(PIVOTI),GT,R2) GO TO 100	SYM0079
79		PIVOTI = PIVOTI*R1	SYM0080
8		ISCALE = ISCALE-1	SYM0081
81		GO TO 90	SYM0082
82		100 DETERM = DETERM*PIVOTI	SYM0083
8		GO TO 150	SYM0084
8	C		SYM0085
85	C		SYM0086
85	C	USING THE LOWER TRIANGULAR ELEMENTS OF MATRIX A, THIS	SYM0087
8	C	SECTION COMPUTES THE UNIT LOWER TRIANGULAR MATRIX	SYM0088
88		110 JM1 = J-1	SYM0089
89		IF (JM1 .EQ. 0) GO TO 140	SYM0090
9		DO 120 K=1,JM1	SYM0091
9		X = X - A(I,K)*A(J,K)	SYM0092
92		120 CONTINUE	SYM0093
93	C		SYM0094
9		140 A(I,J) = X	SYM0095
95	C		SYM0096
96		150 CONTINUE	SYM0097
9	C		SYM0098
9	C	SECTION TO APPLY BACK SUBSTITUTION TO SOLVE L*Y = B FOR	SYM0099
99	C	UNIT LOWER TRIANGULAR MATRIX AND CONSTANT COLUMN VECTOR B	SYM0100
10	C		SYM0101
10		160 IF( IFAC .EQ. 2 ) RETURN	SYM0102
102		DO 180 I=2,N	SYM0103
103		IM1=I-1	SYM0104
10	C		SYM0105
10		DO 180 J=1,NRHS	SYM0106
106		X = B(I,J)	SYM0107
10	C		SYM0108
10		DO 170 K=1,IM1	SYM0109
109		X = X - A(I,K)*R(K,J)	SYM0110
110		170 CONTINUE	SYM0111
11	C		SYM0112
11		B(I,J) = X	SYM0113
113		180 CONTINUE	SYM0114
11	C		SYM0115
11	C	SECTION TO SOLVE (LTRANSPOSE)*X = (D**=I)*Y FOR TRANSPOSE	SYM0116
116	C	OF UNIT LOWER TRIANGULAR MATRIX AND INVERSE OF DIAGONAL	SYM0117
117	C	MATRIX	SYM0118
11	C		SYM0119
11		Y = P(N)	SYM0120
20		DO 190 J=1,NRHS	SYM0121
2		R(N,J) = B(N,J)*Y	SYM0122
2		190 CONTINUE	SYM0123
23	C		SYM0124
24		200 I = NM1+1	SYM0125
2		Y = P(NM1)	SYM0126

THIS PAGE IS  
PROPERTY

126 C			SYM012
127	DO 220 J=1,NRMS		SYM012
128	X = A(NM1,J)*Y		SYM012
129 C			SYM013
130	DO 210 K=1,N		SYM013
131	X = X - A(K,NM1)*B(K,J)		SYM013
132	210 CONTINUE		SYM013
133 C			SYM013
134	B(NM1,J) = X		SYM013
135	220 CONTINUE		SYM013
136 C			SYM013
137 C			SYM013
138 C	TEST TO DETERMINE IF SOLUTIONS HAVE BEEN DETERMINED FOR		SYM013
139 C	ALL COLUMN VECTORS		SYM014
140	NM1 = NM1-1		SYM014
141	IF (NM1 .GT. 0) GO TO 200		SYM014
142 C			SYM014
143	RETURN		SYM014
144	END		SYM014

0		SUBROUTINE GELIM(NMAX,N,A,NRHS,B,IPIVOT,IFAC,WK,IERR)	GEL000
1		IMPLICIT REAL*4 (A-H,O-Z)	GEL000
2		DIMENSION A(NMAX,1),B(NMAX,1),IPIVOT(1),WK(1)	GEL000
3		IERR=0	GEL000
4	C		GEL000
5	C	TEST FOR L/U FACTORIZATION	GEL000
6	C		GEL000
7		IF(IFAC.EQ.1)GO TO 10	GEL000
8		CALL DEFAC(NMAX,N,A,IPIVOT,IFAC,DETERM,ISCALE,WK,IERR)	GEL000
9		IF(IERR.GT.0)RETURN	GEL001
10		IF (IFAC.EQ.2) DETA=DETERM*(10.**((100*ISCALE)))	
11	10	NM1=N-1	GEL001
12	C		GEL001
13	C	TEST FOR SCALAR A MATRIX	GEL001
14	C		GEL001
15		IF(NM1.GT.0)GO TO 40	GEL001
16		IF(A(1,1).EQ.0.)GO TO 30	GEL001
17		DO 20 I=1,NRHS	GEL001
18	20	B(1,I)=B(1,I)/A(1,1)	GEL001
19		IF (IFAC.EQ.2) WK(1)=DETA	
20		RETURN	GEL001
21	30	IERR=1	GEL002
22		RETURN	GEL002
23	C		GEL002
24	40	DO 100 M=1,NRHS	GEL002
25	C		GEL002
26	C	PIVOT THE M-TH COLUMN OF B MATRIX	GEL002
27	C		GEL002
28		DO 50 I=1,NM1	GEL002
29		KI=IPIVOT(I)	GEL002
30		P=B(KI,M)	GEL002
31		B(KI,M)=B(I,M)	GEL003
32	50	B(I,M)=P	GEL003
33	C		GEL003
34	C	FORWARD SUBSTITUTION	GEL003
35	C		GEL003
36		WK(1)=B(1,M)	GEL003
37	C		GEL003
38		DO 70 I=2,N	GEL003
39		IM1=I-1	GEL003
40		P=0.0	GEL003
41		DO 60 K=1,IM1	GEL004
42	60	P=P+A(I,K)*WK(K)	GEL004
43	70	WK(I)=B(I,M)-P	GEL004
44	C		GEL004
45	C	BACK SUBSTITUTION	GEL004
46	C		GEL004
47		B(N,M)=WK(N)/A(N,N)	GEL004
48	C		GEL004
49		DO 90 J=1,NM1	GEL004
50		I=N-J	GEL004
51		IPI=I+1	GEL005
52		P=WK(I)	GEL005
53		DO 80 K=IPI,N	GEL005
54	80	P=P-A(I,K)*B(K,M)	GEL005
55	90	B(I,M)=P/A(I,I)	GEL005
56	C		GEL005
57	100	CONTINUE	GEL005
58		IF (IFAC.EQ.2) WK(1)=DETA	
59		RETURN	GEL005
60		END	GEL005

0		SUBROUTINE SNVDEC(IOP,MD,ND,M,N,A,NOS,R,IAC,ZTEST,Q,V,IRANK,APLUS,SNV000	SNV000
1		IERR)	SNV000
2		IMPLICIT REAL*8 (A-H,O-Z)	SNV000
3		LOGICAL WITHU,WITHV	SNV000
4		DIMENSION A(MD,N),V(ND,N),Q(N),E(150)	SNV000
5		DIMENSION B(MD,NOS),APLUS(ND,M)	SNV000
6	C		SNV000
7	C		SNV000
8	C	TEST FOR SCALAR OR VECTOR A	SNV000
9	C		SNV001
10		IF( N .GE. 2 ) GO TO 3000	SNV001
11	C		SNV001
12		IERR = 0	SNV001
13		ZTEST = 10.**(-IAC)	SNV001
14		SUM = 0.0	SNV001
15		DO 1000 I=1,M	SNV001
16		SUM = SUM + A(I,1)*A(I,1)	SNV001
17	1000	CONTINUE	SNV001
18		SUM = DSQRT(SUM)	SNV001
19		IRANK = 0	SNV002
20		IF( SUM .GT. ZTEST ) IRANK = 1	SNV002
21		Q(1) = SUM	SNV002
22	C		SNV002
23		IF( IOP .EQ. 1 ) RETURN	SNV002
24		V(1,1) = 1.0	SNV002
25		IF( IRANK .EQ. 0 ) GO TO 1200	SNV002
26		DO 1100 I = 1,M	SNV002
27		A(I,1) = A(I,1)/SUM	SNV002
28	1100	CONTINUE	SNV002
29		GO TO 1300	SNV003
30	1200	CONTINUE	SNV003
31		A(1,1) = 1.0	SNV003
32	1300	CONTINUE	SNV003
33	C		SNV003
34		IF( IOP .EQ. 2 ) RETURN	SNV003
35		IF( IOP .EQ. 4 ) GO TO 1850	SNV003
36		IF( IRANK .EQ. 0 ) GO TO 1600	SNV003
37		DO 1500 J = 1,NOS	SNV003
38		Z = 0	SNV003
39		DO 1400 I = 1,M	SNV004
40		Z = Z + A(I,1)*B(I,J)/SUM	SNV004
41	1400	CONTINUE	SNV004
42		B(1,J) = Z	SNV004
43	1500	CONTINUE	SNV004
44		GO TO 1800	SNV004
45	1600	CONTINUE	SNV004
46		DO 1700 J = 1,NOS	SNV004
47		B(1,J) = 0.0	SNV004
48	1700	CONTINUE	SNV004
49	1800	CONTINUE	SNV005
50	C		SNV005
51		IF( IOP .EQ. 3 ) RETURN	SNV005
52	1850	CONTINUE	SNV005
53		IF( IRANK .EQ. 0 ) GO TO 2000	SNV005
54		DO 1900 I = 1,M	SNV005
55		APLUS(1,I) = A(I,1)/SUM	SNV005
56	1900	CONTINUE	SNV005
57		RETURN	SNV005
58	2000	CONTINUE	SNV005
59		DO 2100 I=1,M	SNV006
60		APLUS(1,I) = 0.0	SNV006
61	2100	CONTINUE	SNV006
62		RETURN	SNV006

63	C		SNV006
64	C		SNV006
65		3000 CONTINUE	SNV006
66	C		SNV006
67	C		SNV006
68	C		SNV006
69		TOL=1.0D-60	SNV007
70		SIZE=0.0	SNV007
71		NPI=N+1	SNV007
72	C		SNV007
73	C	COMPUTE THE E-NORM OF MATRIX A AS ZERO TEST FOR SINGULAR VALUES.	SNV007
74	C		SNV007
75		SUM=0.0	SNV007
76		DO 500 I=1,M	SNV007
77		DO 500 J=1,N	SNV007
78	500	SUM = SUM + A(I,J)**2	SNV007
79		ZTEST = DSQRT(SUM)	SNV008
80		ZTEST = ZTEST*10.**(-IAC)	SNV008
81	C		SNV008
82	510	IF (IOP.NE.1 ) GO TO 515	SNV008
83		WITHU=.FALSE.	SNV008
84		WITHV=.FALSE.	SNV008
85		GO TO 520	SNV008
86	515	WITHU=.TRUE.	SNV008
87		WITHV=.TRUE.	SNV008
88	520	CONTINUE	SNV008
89		G = 0.0	SNV009
90		X = 0.0	SNV009
91		DO 30 I = 1,N	SNV009
92	C		SNV009
93	C	HOUSEHOLDER REDUCTION TO BIDIAGONAL FORM.	SNV009
94	C		SNV009
95		E(I) = G	SNV009
96		S = 0.0	SNV009
97		L = I+1	SNV009
98	C		SNV009
99	C	ANNIHILATE THE I-TH COLUMN BELOW DIAGONAL.	SNV010
100	C		SNV010
101		DO 3 J = I,M	SNV010
102	3	S = S + A(J,I)**2	SNV010
103		G = 0.0	SNV010
104		IF(S .LT. TOL) GO TO 10	SNV010
105		G = DSQRT(S)	SNV010
106		F = A(I,I)	SNV010
107		IF(F .GE. 0.0) G = -G	SNV010
108		H = F*G - S	SNV010
109		A(I,I) = F/G	SNV011
110		IF(I .EQ. N) GO TO 10	SNV011
111		DO 9 J = L,N	SNV011
112		S = 0.0	SNV011
113		DO 7 K = I,M	SNV011
114	7	S = S + A(K,I)*A(K,J)	SNV011
115		F = S/H	SNV011
116		DO 8 K = I,M	SNV011
117	8	A(K,J) = A(K,J) + F*A(K,I)	SNV011
118	9	CONTINUE	SNV011
119	10	G(I) = G	SNV012
120		IF(I .EQ. N) GO TO 20	SNV012
121	C		SNV012
122	C	ANNIHILATE THE I-TH ROW TO RIGHT OF SUPER-DIAG.	SNV012
123	C		SNV012
124		S = 0.0	SNV012
125		DO 11 J = L,N	SNV012



26	11	S = S + A(I,J)**2	SNV012
27		G = 0.0	SNV012
128		IF (S .LT. TOL) GO TO 20	SNV012
29		G = DSQRT(S)	SNV013
30		F = A(I,I+1)	SNV013
31		IF(F .GE. 0.0) G = -G	SNV013
132		H = F*G - S	SNV013
33		A(I,I+1) = F - G	SNV013
34		DO 15 J = L,N	SNV013
135	15	E(J) = A(I,J)/H	SNV013
36		DO 19 J = L,M	SNV013
37		S = 0.0	SNV013
138		DO 16 K = L,N	SNV013
139	16	S = S + A(J,K) * A(I,K)	SNV014
40		DO 17 K = L,N	SNV014
41	17	A(J,K) = A(J,K) + S*E(K)	SNV014
142	19	CONTINUE	SNV014
43	20	Y = DARS(Q(I)) + DARS(E(I))	SNV014
44		IF(Y .GT. SIZE) SIZE = Y	SNV014
145	30	CONTINUE	SNV014
46		IF(.NOT. WITHV) GO TO 41	SNV014
47	C		SNV014
48	C	ACCUMULATION OF RIGHT TRANSFORMATIONS.	SNV014
149	C		SNV015
50		DO 40 II = 1,N	SNV015
51		I = NP1 - II	SNV015
152		IF(I .EQ. N) GO TO 39	SNV015
153		IF(G .EQ. 0.0) GO TO 37	SNV015
54		H = A(I,I+1)*G	SNV015
55		DO 32 J = L,N	SNV015
156	32	V(J,I) = A(I,J)/H	SNV015
57		DO 36 J = L,N	SNV015
58		S = 0.0	SNV015
159		DO 33 K = L,N	SNV016
160	33	S = S + A(I,K)*V(K,J)	SNV016
161		DO 34 K = L,N	SNV016
162	34	V(K,J) = V(K,J) + S*V(K,I)	SNV016
163	36	CONTINUE	SNV016
164	37	DO 38 J = L,N	SNV016
165		V(I,J) = 0.0	SNV016
166	38	V(J,I) = 0.0	SNV016
167	39	V(I,I) = 1.0	SNV016
168		G = E(I)	SNV016
169	40	L = I	SNV017
170	41	CONTINUE	SNV017
171		IF(.NOT. WITHU) GO TO 53	SNV017
172	C		SNV017
173	C	ACCUMULATION OF LEFT TRANSFORMATIONS.	SNV017
174	C		SNV017
175		DO 52 II = 1,N	SNV017
176		I = NP1 - II	SNV017
177		L = I + 1	SNV017
178		G = Q(I)	SNV017
179		IF(I .EQ. N) GO TO 43	SNV017
180		DO 42 J = L,N	SNV018
181	42	A(I,J) = 0.0	SNV018
182	43	CONTINUE	SNV018
183		IF(G .EQ. 0.0) GO TO 49	SNV018
184		IF(I .EQ. N) GO TO 47	SNV018
185		H = A(I,I)*G	SNV018
186		DO 46 J = L,N	SNV018
187		S = 0.0	SNV018
188		DO 44 K = L,M	SNV018

189	44	S = S + A(K,I)*A(K,J)	SNV019
190		F = S/H	SNV019
191		DO 45 K = I,M	SNV019
192	45	A(K,J) = A(K,J) + F*A(K,I)	SNV019
193	46	CONTINUE	SNV019
194	47	DO 48 J = I,M	SNV019
195	48	A(J,I) = A(J,I)/G	SNV019
196		GO TO 51	SNV019
197	49	DO 50 J = I,M	SNV019
198	50	A(J,I) = 0.0	SNV019
199	51	A(I,I) = A(I,I) + 1.0	SNV020
200	52	CONTINUE	SNV020
201	53	CONTINUE	SNV020
202	C		SNV020
203	C	DIAGONALIZATION OF BIDIAGONAL FORM.	SNV020
204	C		SNV020
205		DO 100 KK=1,N	SNV020
206		K=NPI-KK	SNV020
207		ITCNT=0	SNV020
208		KPI=K+1	SNV020
209	C		SNV021
210	C	TEST F SPLITTING.	SNV021
211	C		SNV021
212	59	CONTINUE	SNV021
213		DO 60 LL=1,K	SNV021
214		L=KPI-LL	SNV021
215		IF((SIZE+DABS(E(L))),EQ,SIZE) GO TO 64	SNV021
216		LM1=L-1	SNV021
217		IF((SIZE+DABS(Q(LM1))),EQ,SIZE) GO TO 61	SNV021
218	60	CONTINUE	SNV021
219	C		SNV022
220	C	CANCELLATION OF E(L) IF L .GT. 1.	SNV022
221	C		SNV022
222	61	C=0.0	SNV022
223		S=1.0	SNV022
224		L1=L-1	SNV022
225		DO 63 I=L,K	SNV022
226		F=S+E(I)	SNV022
227		E(I)=C+E(I)	SNV022
228		IF((SIZE+DABS(F)),EQ,SIZE) GO TO 64	SNV022
229		G=Q(I)	SNV023
230		Q(I)=DSQRT(F*F+G*G)	SNV023
231		H=Q(I)	SNV023
232		C=G/H	SNV023
233		S=F/H	SNV023
234		IF(,NOT,WITHU) GO TO 63	SNV023
235		DO 62 J=1,M	SNV023
236		Y=A(J,L1)	SNV023
237		Z=A(J,I)	SNV023
238		A(J,L1)=Y*C+Z*S	SNV023
239		A(J,I) = -Y*S+Z*C	SNV024
240	62	CONTINUE	SNV024
241	C		SNV024
242	63	CONTINUE	SNV024
243	C		SNV024
244	C	TEST F CONVERGENCE.	SNV024
245	C		SNV024
246	64	Z=Q(K)	SNV024
247		IF(L,EQ,K) GO TO 75	SNV024
248		IF(ITCNT .LE. 30) GO TO 65	SNV024
249		IFRR = KK	SNV025
250		RETURN	SNV025
251	65	ITCNT = ITCNT + 1	SNV025

252	C			3NV025
253	C	SHIFT FROM LOWER 2X2.		3NV025
254	C			3NV025
255		X=Q(L)		3NV025
256		Y=Q(K-1)		3NV025
257		G=E(K-1)		3NV025
258		H=E(K)		3NV025
259		F=((Y-Z)*(Y+Z)+(G-H)*(G+H))/(2.0*H*Y)		3NV025
260		G=DSQRT(F*F+1.0)		3NV026
261		IF(F.LT.0.0) G=-G		3NV026
262		F = ((X-Z)*(X+Z)+H*(Y/(F+G)-H))/X		3NV026
263	C			3NV026
264	C			3NV026
265	C	NEXT QR TRANSFORMATION.		3NV026
266	C			3NV026
267		C=1.0		3NV026
268		S=1.0		3NV026
269		LP1=L+1		3NV027
270		DO 73 I=LP1,K		3NV027
271		G=E(I)		3NV027
272		Y=Q(I)		3NV027
273		H=S*G		3NV027
274		G=C*G		3NV027
275		Z=DSQRT(F*F+H*H)		3NV027
276		E(I-1)=Z		3NV027
277		C=F/Z		3NV027
278		S=H/Z		3NV027
279		F=X*C+G*S		3NV02A
280		G=-X*S+G*C		3NV02A
281		H=Y*S		3NV02A
282		Y=Y*C		3NV02A
283		IF(.NOT.WITHV) GO TO 70		3NV02A
284		DO 6A J=1,N		3NV02A
285		X=V(J,I-1)		3NV02A
286		Z=V(J,I)		3NV02A
287		V(J,I-1)=X*C+Z*S		3NV02A
288		V(J,I)=-X*S+Z*C		3NV02A
289	6A	CONTINUE		3NV02A
290	C			3NV02A
291	70	Z=DSQRT(F*F+H*H)		3NV02A
292		Q(I-1)=Z		3NV02A
293		C=F/Z		3NV02A
294		S=H/Z		3NV02A
295		F=C*G+S*Y		3NV02A
296		X=-S*G+C*Y		3NV02A
297		IF(.NOT.WITHU) GO TO 73		3NV02A
298		DO 72 J=1,M		3NV02A
299		Y=A(J,I-1)		3NV030
300		Z=A(J,I)		3NV030
301		A(J,I-1)=Y*C+Z*S		3NV030
302		A(J,I)=-Y*S+Z*C		3NV030
303	72	CONTINUE		3NV030
304	C			3NV030
305	C			3NV030
306	73	E(L) = 0.0		3NV030
307		F(K)=F		3NV030
308		Q(K)=X		3NV030
309		GO TO 59		3NV031
310	C			3NV031
311	C	CONVERGENCE.		3NV031
312	C			3NV031
313	75	CONTINUE	A-31	3NV031
314		IF(Z.GE.0.0) GO TO 100		3NV031

31	Q(K)=Z	SNV03160
31	IF(.NOT.WITHV) GO TO 100	SNV03170
317	DO 76 J=1,N	SNV03180
318	76 V(J,K)=V(J,K)	SNV03190
31	100 CONTINUE	SNV03200
32	C	SNV03210
321	IERR=0	SNV03220
32	DO 240 II=2,N	SNV03230
32	I=II-1	SNV03240
324	K=I	SNV03250
325	P=Q(I)	SNV03260
32	DO 250 J=II,N	SNV03270
327	IF (Q(J).LE.P) GO TO 250	SNV03280
328	K=J	SNV03290
32	P=Q(J)	SNV03300
32	250 CONTINUE	SNV03310
331	C	SNV03320
332	IF (K.EQ.I) GO TO 280	SNV03330
33	Q(K) = Q(I)	SNV03340
334	Q(I) = P	SNV03350
335	C	SNV03360
33	IF(IOP.EQ.1) GO TO 280	SNV03370
33	C	SNV03380
338	DO 260 J=1,N	SNV03390
339	P= V(J,I)	SNV03400
34	V(J,I)= V(J,K)	SNV03410
341	V(J,K)= P	SNV03420
342	260 CONTINUE	SNV03430
34	C	SNV03440
34	DO 270 J=1,M	SNV03450
345	P = A(J,I)	SNV03460
346	A(J,I)= A(J,K)	SNV03470
347	A(J,K)= P	SNV03480
348	270 CONTINUE	SNV03490
349	C	SNV03500
35	280 CONTINUE	SNV03510
35	C	SNV03520
352	J=N	SNV03530
353	290 IF (Q(J).GT.ZTEST) GO TO 300	SNV03540
354	Q(J)=0.0	SNV03550
355	J=J-1	SNV03560
356	GO TO 290	SNV03570
357	300 IRANK =J	SNV03580
358	TEMP = ZTEST/Q(J)	SNV03590
359	IF (TEMP.GT..0625) IERR=-1	SNV03600
36	C	SNV03610
361	IF (IOP.LT. 3) RETURN	SNV03620
362	IF(IOP.GT.3) GO TO 170	SNV03630
363	DO 160 L=1,NOS	SNV03640
364	DO 130 J=1,IRANK	SNV03650
365	SUM=0.0	SNV03660
366	DO 120 I=1,M	SNV03670
367	120 SUM =SUM + A(I,J)*B(I,L)	SNV03680
368	130 E(J)= SUM/Q(J)	SNV03690
369	C	SNV03700
370	DO 150 K=1,N	SNV03710
371	SUM=0.0	SNV03720
372	DO 140 I=1,IRANK	SNV03730
373	140 SUM =SUM + V(K,I)*E(I)	SNV03740
374	150 R(K,L)=SUM	SNV03750
375	160 CONTINUE	SNV03760
376	RETURN	SNV03770
377	170 DO 200 J=1,M	SNV03780

37	DO 100 I=1,N	SNV03790
37	SUM=0.0	SNV03800
380	DO 180 K=1,IRANK	SNV03810
381	180 SUM =SUM + V(I,K)*A(J,K)/Q(K)	SNV03820
381	190 APLUS(I,J)= SUM	SNV03830
383	200 CONTINUE	SNV03840
384	C	SNV03850
381	IF( IOP .EQ.4) RETURN	SNV03860
381	DO 230 K=1,NOS	SNV03870
387	DO 220 I=1,N	SNV03880
387	SUM=0.0	SNV03890
381	DO 210 J=1,M	SNV03900
390	210 SIJM=SUM+ APLUS(I,J)*B(J,K)	SNV03910
391	220 E(I)=SUM	SNV03920
391	DO 225 I=1,N	SNV03930
391	225 B(I,K)=E(I)	SNV03940
394	230 CONTINUE	SNV03950
391	RETURN	SNV03960
391	END	SNV03970

	SUBROUTINE SUM(A,NA,B,NB,C,NC,IOP,SYM,DUMMY)	SUM00010
	IMPLICIT REAL*4 (A-H,O-Z)	SUM00020
2	DIMENSION A(1),B(1),C(1),DUMMY(1)	SUM00030
3	DIMENSION NA(2),NB(2),NC(2)	SUM00040
	LOGICAL SYM	SUM00050
6	COMMON/CONV/SUMCV,RICTCV,SERCV,MAXSUM	SUM00060
7	IF( IOP .EQ. 0 ) GO TO 100	SUM00070
	PRINT 50	SUM00080
50	FORMAT(/, ' LINEAR EQUATION SOLVER    X = AX + B ')	SUM00090
	CALL PRNT(A,NA,4H A ,1)	SUM00100
10	IF( SYM ) GO TO 75	SUM00110
11	CALL PRNT(C,NC,4H C ,1)	SUM00120
11	GO TO 85	SUM00130
13	75 CONTINUE	SUM00140
	PRINT 80	SUM00150
11	80 FORMAT(/, ' C = A TRANSPOSE ',/)	SUM00160
16	85 CONTINUE	SUM00170
17	CALL PRNT(B,NB,4H B ,1)	SUM00180
	C	SUM00190
100	CONTINUE	SUM00200
20	N1 = 1 + NA(1)*NC(1)	SUM00210
	I=1	SUM00220
200	CONTINUE	SUM00230
23	CALL MULT(A,NA,B,NB,DUMMY,NB)	SUM00240
24	CALL MULT(DUMMY,NB,C,NC,DUMMY(N1),NB)	SUM00250
	CALL MAXEL(B,NB,WNS)	SUM00260
	CALL MAXEL(DUMMY(N1),NB,WNDX)	SUM00270
27	IF(WNS .GE. 1.) GO TO 225	SUM00280
	IF( WNDX/WNS .LT. SUMCV ) GO TO 300	SUM00290
	GO TO 235	SUM00300
30	225 IF( WNDX .LT. SUMCV ) GO TO 300	SUM00310
31	235 CONTINUE	SUM00320
	CALL ADD(R,NB,DUMMY(N1),NB,B,NB)	SUM00330
33	CALL MULT(A,NA,A,NA,DUMMY,NA)	SUM00340
34	CALL EQUATE(DUMMY,NA,A,NA)	SUM00350
	IF( SYM ) GO TO 245	SUM00360
	CALL MULT(C,NC,C,NC,DUMMY,NC)	SUM00370
37	CALL EQUATE(DUMMY,NC,C,NC)	SUM00380
38	GO TO 250	SUM00390
	245 CONTINUE	SUM00400
40	CALL TRANP(A,NA,C,NC)	SUM00410
41	250 CONTINUE	SUM00420
	I=I+1	SUM00430
	IF( I .LE. MAXSUM ) GO TO 200	SUM00440
44	CALL LNCNT(3)	SUM00450
45	PRINT 275,MAXSUM	SUM00460
46	275 FORMAT(/, ' IN SUM, THE SEQUENCE OF PARTIAL SUMS HAS EXCEEDED STAG	SUM00470
47	1 E',I5,' WITHOUT CONVERGENCE')	SUM00480
48	300 CONTINUE	SUM00490
	IF(IOP .EQ. 0) RETURN	SUM00500
	CALL PRNT(B,NB,4H X ,1)	SUM00510
51	RETURN	SUM00520
	END	SUM00530

1	SUBROUTINE BILIN(A,NA,B,NB,C,NC,IOP,BETA,SYM,DUMMY)	BIL00010
2	IMPLICIT REAL*8 (A-H,O-Z)	BIL00020
3	DIMENSION A(1),B(1),C(1),DUMMY(1)	BIL00030
4	DIMENSION NA(2),NB(2),NC(2),NDUM(2)	BIL00040
5	DIMENSION IOP(2)	BIL00050
6	LOGICAL SYM	BIL00060
7	IF( IOP(1) .EQ. 0 ) GO TO 300	BIL00070
8	IF(SYM) GO TO 100	BIL00080
9	CALL LNCNT(3)	BIL00090
10	PRINT 50	BIL00100
11	50 FORMAT(//, ' LINEAR EQUATION SOLVER AX + XB = C')	BIL00110
12	CALL PRNT(A,NA,4M A ,1)	BIL00120
13	CALL PRNT(B,NB,4M B ,1)	BIL00130
14	GO TO 200	BIL00140
15	100 CONTINUE	BIL00150
16	CALL LNCNT(3)	BIL00160
17	PRINT 150	BIL00170
18	150 FORMAT(//, ' LINEAR EQUATION SOLVER ( B TRANSPOSE )X + XB = C')	BIL00180
19	CALL TRANSP(A,NA,DUMMY,NDUM)	BIL00190
20	CALL PRNT(DUMMY,NDUM,4M B ,1)	BIL00200
21	200 CONTINUE	BIL00210
22	CALL PRNT(C,NC,4M C ,1)	BIL00220
23	300 CONTINUE	BIL00230
24	C	BIL00240
25	IOP(1) = 0	BIL00250
26	N=NA(1)**2	BIL00260
27	M=NB(1)**2	BIL00270
28	C	BIL00280
29	IF( IOP(2) .EQ. 0 ) GO TO 500	BIL00290
30	C	BIL00300
31	N1 = N + 1	BIL00310
32	CALL EQUATE(A,NA,DUMMY,NA)	BIL00320
33	N2 = N1 + NA(1)	BIL00330
34	N3 = N2 + NA(1)	BIL00340
35	ISV = 0	BIL00350
36	ILV = 0	BIL00360
37	NEVL = NA(1)	BIL00370
38	CALL EIGEN(NA(1),NA(1),DUMMY,DUMMY(N1),DUMMY(N2),ISV,ILV,V,DUMMY(NBIL00380	
39	13),IERR)	BIL00390
40	IF (IERR .EQ. 0) GO TO 350	BIL00400
41	CALL LNCNT(3)	BIL00410
42	PRINT 325,IERR	BIL00420
43	325 FORMAT(//, ' IN BILIN, THE ',I4, ' EIGENVALUE OF A HAS NOT BEEN DETRIL00430	
44	TERMINED AFTER 30 ITERATIONS')	BIL00440
45	IERR=1	BIL00450
46	CALL NORMS(NEVL,NEVL,NEVL,A,IERR,BETA)	BIL00460
47	BETA=2.*BETA	BIL00470
48	GO TO 385	BIL00480
49	350 CONTINUE	BIL00490
50	J = N1 + NEVL - 1	BIL00500
51	K = N2 + NEVL - 1	BIL00510
52	CO = DSQRT(DUMMY(N1)**2 + DUMMY(N2)**2)	BIL00520
53	CN = DSQRT(DUMMY(J)**2 + DUMMY(K)**2)	BIL00530
54	CD = DUMMY(J)-DUMMY(N1)	BIL00540
55	IF(CD .EQ. 0,0) GO TO 365	BIL00550
56	BETA = (DUMMY(N1)*CN-DUMMY(J)*CO)/CO	BIL00560
57	IF(BETA .LE. 0,0) GO TO 365	BIL00570
58	BETA = DSQRT(BETA)	BIL00580
59	GO TO 385	BIL00590
60	C	BIL00600
61	365 CONTINUE	BIL00610
62	C	BIL00620
63	BETA = 0.0	BIL00630

67	DO 375 I = 1, NEVL	BIL00640
68	J = N1 + I - 1	BIL00650
69	K = N2 + I - 1	BIL00660
66	IF(DUMMY(J) .GE. 0.0) GO TO 375	BIL00670
67	BETA = BETA + DSQRT(DUMMY(J)**2 + DUMMY(K)**2)	BIL00680
68	375 CONTINUE	BIL00690
69	BETA = BETA/NEVL	BIL00700
70	C	BIL00710
71	385 CONTINUE	BIL00720
72	C	BIL00730
73	IF( SYM ) GO TO 500	BIL00740
74	CALL EQUATE(B,NB,DUMMY,NB)	BIL00750
75	N1 = M + 1	BIL00760
76	N2 = N1 + NB(1)	BIL00770
77	N3 = N2 + NB(1)	BIL00780
78	NEVL = NB(1)	BIL00790
79	CALL EIGEN(NB(1),NB(1),DUMMY,DUMMY(N1),DUMMY(N2),ISV,ILV,V,DUMMY(NB(1),	BIL00800
80	13),IERR)	BIL00810
81	IF(IERR .EQ. 0) GO TO 450	BIL00820
82	CALL LNCNT(3)	BIL00830
83	PRINT 400,IERR	BIL00840
84	400 FORMAT(/,,' IN BILIN, THE ',I4,' EIGENVALUE OF B HAS NOT BEEN FOUND	BIL00850
85	10 AFTER 30 ITERATIONS')	BIL00860
86	IERR = 1	BIL00870
87	CALL NORMS(NEVL,NEVL,NEVL,B,IERR,BETA1)	BIL00880
88	BETA1 = 2.*BETA1	BIL00890
89	GO TO 485	BIL00900
90	450 CONTINUE	BIL00910
91	J = N1 + NEVL - 1	BIL00920
92	K = N2 + NEVL - 1	BIL00930
93	CO = DSQRT(DUMMY(N1)**2 + DUMMY(N2)**2)	BIL00940
94	CN = DSQRT(DUMMY(J)**2 + DUMMY(K)**2)	BIL00950
95	CD = DUMMY(J) - DUMMY(N1)	BIL00960
96	IF(CD .EQ. 0.0) GO TO 465	BIL00970
97	BETA1 = (DUMMY(N1)*CN - DUMMY(J)*CO)/CD	BIL00980
98	IF(BETA1 .LE. 0.0) GO TO 465	BIL00990
99	BETA1 = DSQRT(BETA1)	BIL01000
100	GO TO 485	BIL01010
101	C	BIL01020
102	465 CONTINUE	BIL01030
103	C	BIL01040
104	BETA1 = 0.0	BIL01050
105	DO 475 I = 1, NEVL	BIL01060
106	J = N1 + I - 1	BIL01070
107	K = N2 + I - 1	BIL01080
108	IF(DUMMY(J) .GE. 0.0) GO TO 475	BIL01090
109	BETA1 = BETA1 + DSQRT(DUMMY(J)**2 + DUMMY(K)**2)	BIL01100
110	475 CONTINUE	BIL01110
111	BETA1 = BETA1/NEVL	BIL01120
112	C	BIL01130
113	485 CONTINUE	BIL01140
114	BETA = (BETA + BETA1)/2.	BIL01150
115	C	BIL01160
116	500 CONTINUE	BIL01170
117	C	BIL01180
118	C	BIL01190
119	IF( IOP(1) .EQ. 0 ) GO TO 520	BIL01200
120	CALL LNCNT(4)	BIL01210
121	PRINT 515,BETA	BIL01220
122	515 FORMAT(/,,' BETA = ',E16.8,')	BIL01230
123	520 CONTINUE	BIL01240
124	C	BIL01250
125	N1 = N + 1	BIL01260



124	CALL EQUATE(A,NA,DUMMY,NA)	BIL01270
125	CALL EQUATE(A,NA,DUMMY(N1),NA)	BIL01280
126	CALL SCALE(DUMMY,NA,DUMMY,NA,-1.)	BIL01290
129	L = -NA(1)	BIL01300
131	NAX = NA(1)	BIL01310
131	DO 525 I=1,NAX	BIL01320
132	L = L + NAX + 1	BIL01330
133	M1 = L + N	BIL01340
134	DUMMY(L) = BETA - A(L)	BIL01350
135	DUMMY(M1) = BETA + A(L)	BIL01360
136	525 CONTINUE	BIL01370
137	N2 = N1 + N	BIL01380
138	CALL EQUATE(C,NC,DUMMY(N2),NDUM)	BIL01390
139	NDUM(2) = NDUM(2) + NA(1)	BIL01400
140	N3 = N2 + NC(1)*NC(2)	BIL01410
141	GAM = -2.*BETA	BIL01420
142	C	BIL01430
143	IF( .NOT. SYM ) GO TO 600	BIL01440
144	C	BIL01450
145	CALL UNITY(DUMMY(N3),NA)	BIL01460
146	N4 = N3 + N	BIL01470
147	NDUM(2) = NDUM(2) + NA(1)	BIL01480
148	N5 = N4 + NA(1)	BIL01490
149	IFAC = 0	BIL01500
150	CALL GELIM(NA(1),NA(1),DUMMY,NDUM(2),DUMMY(N1),DUMMY(N4),IFAC,DUMM	BIL01510
151	Y(N5),IERR)	BIL01520
152	IF( IERR .EQ. 1 ) PRINT 625	BIL01530
153	CALL EQUATE(DUMMY(N1),NA,DUMMY,NA)	BIL01540
154	CALL EQUATE(DUMMY(N2),NC,C,NC)	BIL01550
155	CALL TRANSP(DUMMY,NA,DUMMY(N1),NA)	BIL01560
156	CALL TRANSP(DUMMY(N3),NA,DUMMY(N2),NA)	BIL01570
157	CALL MULT(C,NC,DUMMY(N2),NA,DUMMY(N3),NA)	BIL01580
158	CALL SCALE(DUMMY(N3),NC,C,NC,GAM)	BIL01590
159	C	BIL01600
160	C	BIL01610
161	CALL SUM(DUMMY,NA,C,NC,DUMMY(N1),NA,IOPTT,SYM,DUMMY(N2))	BIL01620
162	GO TO 700	BIL01630
163	600 CONTINUE	BIL01640
164	N4 = N3 + NA(1)	BIL01650
165	IFAC = 0	BIL01660
166	CALL GELIM(NA(1),NA(1),DUMMY,NDUM(2),DUMMY(N1),DUMMY(N3),IFAC,DUMM	BIL01670
167	Y(N4),IERR)	BIL01680
168	IF(IERR .EQ. 1 ) PRINT 625	BIL01690
169	625 FORMAT('/', ' IN BILIN, THE MATRIX (BETA)I - A IS SINGULAR, INCREAS	BIL01700
170	1E BETA')	BIL01710
171	CALL EQUATE(DUMMY(N1),NA,DUMMY,NA)	BIL01720
172	CALL EQUATE(DUMMY(N2),NC,C,NC)	BIL01730
173	N2 = M + N	BIL01740
174	CALL EQUATE(B,NB,DUMMY(N1),NB)	BIL01750
175	CALL EQUATE(B,NB,DUMMY(N2),NB)	BIL01760
176	CALL SCALE(DUMMY(N1),NB,DUMMY(N1),NB,-1.0)	BIL01770
177	L = NB(1)	BIL01780
178	NAX = NB(1)	BIL01790
179	DO 650 I = 1, NAX	BIL01800
180	L = L + NAX + 1	BIL01810
181	L1 = L + N	BIL01820
182	M1 = L + N2 - 1	BIL01830
183	DUMMY(L1) = BETA - B(L)	BIL01840
184	DUMMY(M1) = BETA + B(L)	BIL01850
185	650 CONTINUE	BIL01860
186	C	BIL01870
187	N3 = N2 + M	BIL01880
188	CALL TRANSP(DUMMY(N1),NB,DUMMY(N3),NB)	BIL01890

189	CALL EQUATE(DUMMY(N3),NS,DUMMY(N1),NB)	BIL01900
190	CALL TRANSP(DUMMY(N2),NB,DUMMY(N3),NB)	BIL01910
191	CALL EQUATE(DUMMY(N3),NB,DUMMY(N2),NB)	BIL01920
192	CALL TRANSP(C,NC,DUMMY(N3),NDUM)	BIL01930
193	NSDUM = NDUM(2)	BIL01940
194	NDUM(2) = NDUM(2) + NB(2)	BIL01950
195	IFAC = 0	BIL01960
196	N4 = N3 + NC(1) * NC(2)	BIL01970
197	N5 = N4 + NB(1)	BIL01980
198	CALL GELIM(NB(1),NB(1),DUMMY(N1),NDUM(2),DUMMY(N2),DUMMY(N4),IFAC,	BIL01990
199	DUMMY(N5),IERR)	BIL02000
200	IF(IERR .EQ. 1) PRINT 675	BIL02010
201	675 FORMAT(/, 'IN BILIN, THE MATRIX (BETA)I - B IS SINGULAR, INCREASE	BIL02020
202	1 BETA ')	BIL02030
203	CALL TRANSP(DUMMY(N2),NB,DUMMY(N1),NB)	BIL02040
204	NDUM(2) = NSDUM	BIL02050
205	CALL TRANSP(DUMMY(N3),NDUM,C,NC)	BIL02060
206	CALL SCALE(C,NC,C,NC,GAM)	BIL02070
207	N2 = N + M + 1	BIL02080
208	CALL SUM(DUMMY,NA,C,NC,DUMMY(N1),NB,IOPIT,SYM,DUMMY(N2))	BIL02090
209	C	BIL02100
210	700 CONTINUE	BIL02110
211	IF( IOP(1) .EQ. 0 ) RETURN	BIL02120
212	CALL PRNT(C,NC,4H X ,1)	BIL02130
213	RETURN	BIL02140
214	END	BIL02150

1	SUBROUTINE BARSTW(A,NA,B,NB,C,NC,IOP,SYM,EPSA,EPSE,DUMMY)	BAR00010
2	IMPLICIT REAL*8 (A-H,O-Z)	BAR00020
3	DIMENSION A(1),B(1),C(1),DUMMY(1)	BAR00030
4	DIMENSION NA(2),NB(2),NC(2),NDUM1(2),NDUM2(2),NDUM3(2),NDUM4(2)	BAR00040
5	LOGICAL SYM	BAR00050
6	IF ( IOP .EQ. 0 ) GO TO 250	BAR00060
7	IF(SYM) GO TO 100	BAR00070
8	CALL LNCNT(3)	BAR00080
9	PRINT 50	BAR00090
10	50 FORMAT(/, ' LINEAR EQUATION SOLVER    AX + XB = C ')	BAR00100
11	CALL PRNT(A,NA,4H A ,1)	BAR00110
12	CALL PRNT(B,NB,4H B ,1)	BAR00120
13	GO TO 200	BAR00130
14	100 CONTINUE	BAR00140
15	CALL LNCNT(3)	BAR00150
16	PRINT 150	BAR00160
17	150 FORMAT(/, ' LINEAR EQUATION SOLVER ( B TRANSPOSE )X + XB = C')	BAR00170
18	CALL TRANP(A,NA,DUMMY,NDUM1)	BAR00180
19	CALL PRNT(DUMMY,NDUM1,4H B ,1)	BAR00190
20	200 CONTINUE	BAR00200
21	CALL PRNT(C,NC,4H C ,1)	BAR00210
22	C	BAR00220
23	250 CONTINUE	BAR00230
24	CALL EQUATE(A,NA,DUMMY,NDUM1)	BAR00240
25	N1=(NA(1)**2)+1	BAR00250
26	N2=N1+NA(1)-1	BAR00260
27	DO 300 I=N1,N2	BAR00270
28	DUMMY(I)=0.0	BAR00280
29	300 CONTINUE	BAR00290
30	C	BAR00300
31	NDUM1(2)=NDUM1(2)+1	BAR00310
32	NDUM2(1)=1	BAR00320
33	NDUM2(2)=NDUM1(2)	BAR00330
34	N1=NDUM1(1)*NDUM1(2)+1	BAR00340
35	CALL NULL(DUMMY(N1),NDUM2)	BAR00350
36	LU=(NA(1)+1)**2 + 1	BAR00360
37	CALL JUXTA(DUMMY,NDUM1,DUMMY(N1),NDUM2,DUMMY(LU),NDUM3)	BAR00370
38	CALL EQUATE(DUMMY(LU),NDUM3,DUMMY,NDUM1)	BAR00380
39	N=NA(1)+1	BAR00390
40	C	BAR00400
41	IF(SYM ) GO TO 500	BAR00410
42	C	BAR00420
43	CALL EQUATE(B,NB,DUMMY(LU),NDUM2)	BAR00430
44	M1=LU+NB(1)**2	BAR00440
45	M2=M1+NB(1)-1	BAR00450
46	DO 400 I=M1,M2	BAR00460
47	DUMMY(I)=0.0	BAR00470
48	400 CONTINUE	BAR00480
49	C	BAR00490
50	NDUM2(2)=NDUM2(2)+1	BAR00500
51	NDUM3(1)=1	BAR00510
52	NDUM3(2)=NDUM2(2)	BAR00520
53	M1=NDUM2(1)*NDUM2(2)+LU	BAR00530
54	CALL NULL(DUMMY(M1),NDUM3)	BAR00540
55	M2=LU+(NB(1)+1)**2	BAR00550
56	CALL JUXTA(DUMMY(LU),NDUM2,DUMMY(M1),NDUM3,DUMMY(M2),NDUM4)	BAR00560
57	CALL EQUATE(DUMMY(M2),NDUM4,DUMMY(LU),NDUM2)	BAR00570
58	M=NB(1)+ 1	BAR00580
59	LNB = LU	BAR00590
60	LU = LU + (NB(1)+1)**2	BAR00600
61	LV = LU + NA(1)**2	BAR00610
62	CALL AXPXA(DUMMY,DUMMY(LU),NA(1),N,NA(1),DUMMY(LNB),DUMMY(LV),NB(1)	BAR00620
63	),M,NB(1),C,NC(1),EPSA,EPSE,FAIL)	BAR00630

63	GO TO 600	BAR00640
64	C	BAR00650
65	500 CONTINUE	BAR00660
66	CALL TRANP(DUMMY,NDUM1,DUMMY(LU),NDUM2)	BAR00670
67	CALL EQUATE(DUMMY(LU),NDUM2,DUMMY,NDUM1)	BAR00680
68	CALL ATXPXA(DUMMY,DUMMY(LU),C,NA(1),N,NA(1),NC(1),EPSA,NFAIL)	BAR00690
69	C	BAR00700
70	600 CONTINUE	BAR00710
71	IF(NFAIL .EQ. 0 ) GO TO 700	BAR00720
72	CALL LNCNT(3)	BAR00730
73	PRINT 650	BAR00740
74	650 FORMAT(//,' IN BARSTN, EITHER THE SURROUTINE AXPXB OR ATXPXA	BAR00750
75	IS UNABLE TO REDUCE A OR 4 TO SCHUR FORM')	BAR00760
76	RETURN	BAR00770
77	C	BAR00780
78	700 CONTINUE	BAR00790
79	C	BAR00800
80	IF( IOP .NE. 0 ) CALL PRNT(C,NC,4H X ,1)	BAR00810
81	RETURN	BAR00820
82	END	BAR00830

1	SUBROUTINE TESTSA(A,NA,ALPHA,DISC,STABLE,IOP,DUMMY)	TES00010
2	IMPLICIT REAL*8 (A-H,O-Z)	TES00020
3	DIMENSION A(1),DUMMY(1)	TES00030
4	DIMENSION NA(2),NDUM1(2),NDUM2(2)	TES00040
5	LOGICAL DISC,STABLE	TES00050
6	STABLE = .FALSE.	TES00060
7		TES00070
8	CALL EQUATE(A,NA,DUMMY,NA)	TES00080
9	N1 = NA(1)**2 + 1	TES00090
10	N2 = N1+NA(1)	TES00100
11	N3 = N2+NA(1)	TES00110
12	ISV = 0	TES00120
13	CALL EIGEN(NA(1),NA(1),DUMMY,DUMMY(N1),DUMMY(N2),ISV,ISV,V,DUMMY(N	TES00130
14	13),IERR)	TES00140
15	NEVL = NA(1)	TES00150
16	IF( IERR .EQ. 0 ) GO TO 200	TES00160
17	CALL LNCNT(4)	TES00170
18	PRINT 100,IERR	TES00180
19	100 FORMAT(//,' IN TESTSA, THE ',IS,' EIGENVALUE OF A HAS NOT BEEN FO	TES00190
20	UND AFTER 30 ITERATIONS',/)	TES00200
21	RETURN	TES00210
22	1 C	TES00220
23	200 CONTINUE	TES00230
24	NDUM1(1) = NEVL	TES00240
25	NDUM1(2) = 1	TES00250
26	CALL JUXTC(DUMMY(N1),NDUM1,DUMMY(N2),NDUM1,DUMMY,NDUM2)	TES00260
27	2 C	TES00270
28	IF( DISC ) GO TO 400	TES00280
29	DO 300 I=1,NEVL	TES00290
30	IF( DUMMY(I) .GE. ALPHA ) GO TO 600	TES00300
31	300 CONTINUE	TES00310
32	GO TO 550	TES00320
33	400 CONTINUE	TES00330
34	N = NDUM2(1)*NDUM2(2)+1	TES00340
35	DO 500 I =1,NEVL	TES00350
36	K = I + NEVL	TES00360
37	L=N + I -1	TES00370
38	DUMMY(L) = DSQRT((DUMMY(I)**2)+(DUMMY(K)**2))	TES00380
39	500 CONTINUE	TES00390
40	3 C	TES00400
41	IF( DUMMY(L) .GE. ALPHA ) GO TO 600	TES00410
42	1 C	TES00420
43	550 CONTINUE	TES00430
44	STABLE = .TRUE.	TES00440
45	600 CONTINUE	TES00450
46	IF( IOP .EQ. 0 ) RETURN	TES00460
47	CALL LNCNT(4)	TES00470
48	PRINT 700	TES00480
49	700 FORMAT(//,' PROGRAM TO TEST THE RELATIVE STABILITY OF THE MATRIX A	TES00490
50	1',/)	TES00500
51	CALL PRNT(A,NA,4H A ,1)	TES00510
52	CALL LNCNT(4)	TES00520
53	PRINT 750	TES00530
54	750 FORMAT(//,' EIGENVALUES OF A ',/)	TES00540
55	CALL PRNT(DUMMY,NDUM2,4HEVLA,1)	TES00550
56	IF( .NOT. DISC ) GO TO 850	TES00560
57	CALL LNCNT(4)	TES00570
58	PRINT 800	TES00580
59	800 FORMAT(//,' MODULI OF EIGENVALUES OF A ',/)	TES00590
60	CALL PRNT(DUMMY(N),NDUM1,4HMODA,1)	TES00600
61	1 C	TES00610
62	850 CONTINUE	TES00620
63	CALL LNCNT(4)	TES00630

```
IF(STABLE) PRINT 900,ALPHA
64 IF( .NOT. STABLE) PRINT 950,ALPHA
65 900 FORMAT(//,' MATRIX A IS STABLE RELATIVE TO ',E16.8,/)
66 950 FORMAT(//,' MATRIX A IS UNSTABLE RELATIVE TO ',E16.8,/)
67 C
68 RETURN
69 END
```

```
TES00640
TES00650
TES00660
TES00670
TES00680
TES00690
TES00700
```

```

1) SUBROUTINE EXPSER(A,NA,EXPA,NEXPA,T,IOP,DUMMY) EXP00010
2) IMPLICIT REAL*8 (A-H,O-Z) EXP00020
3) DIMENSION A(1),EXPA(1),DUMMY(1) EXP00030
4) DIMENSION NA(2),NEXPA(2) EXP00040
5) COMMON/CONV/SUMCV,RICTCV,SERCV,MAXSUM EXP00050
6) C EXP00060
7) N = NA(1) EXP00070
8) L = (N**2) + 1 EXP00080
9) TT = T EXP00090
10) NEXPA(1)=NA(1) EXP00100
11) NEXPA(2)=NA(2) EXP00110
12) C EXP00120
13) CALL MAXEL(A,NA,ANAA) EXP00130
14) ANAA = ANAA*TT EXP00140
15) ANAA = DABS(ANAA) EXP00150
16) IF( ANAA .GT. 1.E-15 ) GO TO 100 EXP00160
17) CALL UNITY(EXPA,NEXPA) EXP00170
18) GO TO 800 EXP00180
19) C EXP00190
20) 100 CONTINUE EXP00200
21) IOPT=2 EXP00210
22) CALL NORMS(N,N,N,A,IOPT,ZERO) EXP00220
23) ZERO=ZERO/(2.**47) EXP00230
24) CALL TRCE(A,NA,TR) EXP00240
25) TR = TR/N EXP00250
26) DO 200 I = 1,N EXP00260
27) M = I+N*(I-1) EXP00270
28) A(M) = A(M) - TR EXP00280
29) 200 CONTINUE EXP00290
30) C EXP00300
31) IOPT = 1 EXP00310
32) CALL NORMS(N,N,N,A,IOPT,COL) EXP00320
33) IOPT = 3 EXP00330
34) CALL NORMS(N,N,N,A,IOPT,ROW) EXP00340
35) ANORM = ROW EXP00350
36) IF( ANORM .GT. COL ) ANORM = COL EXP00360
37) TMAX = 1./ANORM EXP00370
38) K = 0 EXP00380
39) 300 CONTINUE EXP00390
40) IF( TMAX = TT ) 325,350,350 EXP00400
41) 325 CONTINUE EXP00410
42) K=K+1 EXP00420
43) TT = T/(2**K) EXP00430
44) IF( K = 1000 ) 300,700,700 EXP00440
45) 350 CONTINUE EXP00450
46) SC = TT EXP00460
47) CALL SCALE(A,NA,A,NA,TT) EXP00470
48) CALL UNITY(EXPA,NEXPA) EXP00480
49) II = 2 EXP00490
50) CALL ADD(A,NA,EXPA,NEXPA,DUMMY,NA) EXP00500
51) CALL EQUATE(A,NA,DUMMY(L),NA) EXP00510
52) 400 CONTINUE EXP00520
53) CALL MULT(A,NA,DUMMY(L),NA,EXPA,NEXPA) EXP00530
54) S = 1./I EXP00540
55) CALL SCALE(EXPA,NEXPA,DUMMY(L),NA,S) EXP00550
56) CALL ADD(DUMMY(L),NA,DUMMY,NA,EXPA,NEXPA) EXP00560
57) CALL MAXEL(DUMMY,NA,TOT) EXP00570
58) CALL MAXEL(DUMMY(L),NA,DELT) EXP00580
59) IF( TOT .GT. 1.0 ) GO TO 500 EXP00590
60) IF( DELT/TOT .LT. SERCV ) GO TO 600 EXP00600
61) GO TO 550 EXP00610
62) 500 CONTINUE EXP00620
63) IF( DELT .LT. SERCV ) GO TO 600 EXP00630

```

5	550	CONTINUE	EXP00640
6		CALL EQUATE(EXPA,NEXPA,DUMMY,NA)	EXP00650
65		II = II + 1	EXP00660
66		GO TO 400	EXP00670
67	C		EXP00680
68	600	CONTINUE	EXP00690
69		IF( K ) 625,675,650	EXP00700
70	625	CONTINUE	EXP00710
71		CALL LNCNT(1)	EXP00720
72		PRINT 635	EXP00730
73	635	FORMAT( ' ERROR IN EXPSER, K IS NEGATIVE ' )	EXP00740
74		RETURN	EXP00750
75	C		EXP00760
76	650	CONTINUE	EXP00770
77		DO 660 I = 1,K	EXP00780
78		TT = 2*TT	EXP00790
79		CALL EQUATE(EXPA,NEXPA,DUMMY,NA)	EXP00800
80		CALL EQUATE(DUMMY,NA,DUMMY(L),NA)	EXP00810
81		CALL MULT(DUMMY(L),NA,DUMMY,NA,EXPA,NEXPA)	EXP00820
82	660	CONTINUE	EXP00830
83		T = TT	EXP00840
84	675	CONTINUE	EXP00850
85		S = 1./SC	EXP00860
86		CALL SCALE(A,NA,A,NA,S)	EXP00870
87		DO 685 I = 1,N	EXP00880
88		M = I + N*(I-1)	EXP00890
89		A(M) = A(M) + TR	EXP00900
90		IF( DABS(A(M)) .LE. ZERO ) A(M) = 0.0	EXP00910
91	685	CONTINUE	EXP00920
92	C		EXP00930
93		TR=TR*T	EXP00940
94		S = DEXP(T)	EXP00950
95		CALL SCALE(EXPA,NEXPA,EXPA,NEXPA,S)	EXP00960
96		GO TO 800	EXP00970
97	C		EXP00980
98	700	CONTINUE	EXP00990
99		CALL LNCNT(1)	EXP01000
100		PRINT 750	EXP01010
101	750	FORMAT( ' ERROR IN EXPSER, K = 1000 ' )	EXP01020
102		RETURN	EXP01030
103	C		EXP01040
104	800	CONTINUE	EXP01050
105		IF( IOP .EQ. 0 ) RETURN	EXP01060
106		CALL LNCNT(4)	EXP01070
107		PRINT 825	EXP01080
108	825	FORMAT(// ' COMPUTATION OF THE MATRIX EXPONENTIAL EXP(A T) BY THE S	EXP01090
109		ERIES METHOD '//)	EXP01100
110		CALL PRNT(A,NA,4H A ,1)	EXP01110
111		CALL LNCNT(3)	EXP01120
112		PRINT A50,T	EXP01130
113	850	FORMAT(// ' T = ',D16.8//)	EXP01140
114		CALL PRNT(EXPA,NEXPA,4HEXPA,1)	EXP01150
115		RETURN	EXP01160
116		END	EXP01170



0	SUBROUTINE EXPADE (MAX, N, A, EA, IDIG, WK, IERR)	EXP00010
	IMPLICIT REAL*8 (A-H,O-Z)	EXP00020
	DIMENSION A(MAX,N),EA(MAX,N),WK(N,1),C(9)	EXP00030
3	REAL*4 SDIGC,ALOG10	EXP00040
	IERR = 0	EXP00050
	C CALCULATE NORM OF A	EXP00060
6	ANORM = 0.	EXP00070
7	DO 10 I=1,N	EXP00080
	S = 0.	EXP00090
4	DO 5 J=1,N	EXP00100
10	S = S + DABS(A(I,J))	EXP00110
	5 CONTINUE	EXP00120
	IF (S .GT. ANORM) ANORM = S	EXP00130
13	10 CONTINUE	EXP00140
	C ****	EXP00150
5	C CALCULATE ACCURACY ESTIMATE	EXP00160
16	C ****	EXP00170
17	DIGC = 24.*DFLOAT(N)	EXP00180
8	IF (ANORM .GT. 1.) DIGC = DIGC*ANORM	EXP00190
19	SDIGC=DIGC	EXP00200
20	IDIG = 15 - IFIX(ALOG10(SDIGC) + .5)	EXP00210
1	C DETERMINE POWER OF TWO AND NORMALIZATION FACTOR	EXP00220
2	C ****	EXP00230
23	M = 0	EXP00240
24	IF (ANORM .LE. 1.) GO TO 27	EXP00250
5	FACTOR = 2.	EXP00260
26	DO 15 M=1,46	EXP00270
27	IF (ANORM .LE. FACTOR) GO TO 20	EXP00280
8	FACTOR = FACTOR*2.	EXP00290
9	15 CONTINUE	EXP00300
30	GO TO 125	EXP00310
1	20 CONTINUE	EXP00320
2	C ****	EXP00330
33	C NORMALIZE MATRIX	EXP00340
24	C ****	EXP00350
5	DO 25 I=1,N	EXP00360
26	DO 25 J=1,N	EXP00370
37	A(I,J) = A(I,J)/FACTOR	EXP00380
8	25 CONTINUE	EXP00390
9	27 CONTINUE	EXP00400
40	C ****	EXP00410
1	C SET COEFFICIENTS FOR (9,9) PADE TABLE ENTRY	EXP00420
2	C ****	EXP00430
43	C(1) = .5	EXP00440
44	C(2) = 1.1764705882352D-01	EXP00450
45	C(3) = 1.7156862745098D-02	EXP00460
46	C(4) = 1.7156862745098D-03	EXP00470
47	C(5) = 1.2254901960784D-04	EXP00480
48	C(6) = 6.2845651080945D-06	EXP00490
49	C(7) = 2.2444875386051D-07	EXP00500
50	C(8) = 5.1011080422845D-09	EXP00510
51	C(9) = 5.6678978247605D-11	EXP00520
52	C ****	EXP00530
53	C CALCULATE PADE NUMERATOR AND DENOMINATOR BY COLUMNS	EXP00540
54	C ****	EXP00550
55	NP1 = N+1	EXP00560
56	NP7 = N+7	EXP00570
57	DO 95 J=1,N	EXP00580
58	C ****	EXP00590
59	C COMPUTE JTH COLUMN OF FIRST NINE POWERS OF A	EXP00600
60	C ****	EXP00610
61	DO 35 I=1,N	EXP00620
62	S = 0.	EXP00630

ORIGINAL PAGE IS  
OF POOR QUALITY

3		DO 30 L=1,N	EXP00640
4		S = S + A(I,L)*A(L,J)	EXP00650
5	30	CONTINUE	EXP00660
6		WK(I, NP1) = S	EXP00670
7	35	CONTINUE	EXP00680
8		DO 45 K=NP1, NP7	EXP00690
9		KP1 = K+1	EXP00700
0		DO 45 I=1,N	EXP00710
1		S = 0.	EXP00720
2		DO 40 L=1,N	EXP00730
3		S = S + A(I,L)*WK(L,K)	EXP00740
4	40	CONTINUE	EXP00750
5		WK(I, KP1) = S	EXP00760
6	45	CONTINUE	EXP00770
7	C ****		EXP00780
8	C	COLLECT TERMS FOR JTH COLUMN OF NUMERATOR AND DENOMINATOR	EXP00790
9	C ****		EXP00800
0		DO 85 I=1,N	EXP00810
1		S = 0.	EXP00820
2		U = 0.	EXP00830
3		DO 65 L=1,8	EXP00840
4		K = N+9-L	EXP00850
5		KN1 = K-N+1	EXP00860
6		P = C(KN1)*WK(I,K)	EXP00870
7		S = S + P	EXP00880
8		IEO = MOD(KN1,2)	EXP00890
9		IF (IEO, EQ, 0) GO TO 55	EXP00900
0		U = U - P	EXP00910
1		GO TO 65	EXP00920
2	55	CONTINUE	EXP00930
3		U = U + P	EXP00940
4	65	CONTINUE	EXP00950
5		P = C(1)*A(I,J)	EXP00960
6		S = S + P	EXP00970
7		U = U - P	EXP00980
8		IF (I .NE. J) GO TO 80	EXP00990
9		S = S + 1.	EXP01000
0		U = U + 1.	EXP01010
1	80	CONTINUE	EXP01020
2		EA(I,J) = S	EXP01030
3		WK(I,J) = U	EXP01040
4	85	CONTINUE	EXP01050
5	95	CONTINUE	EXP01060
6	C ****		EXP01070
7	C	CALCULATE NORMALIZED EXP(A) BY WK * EXP(A) = EA	EXP01080
8	C ****		EXP01090
9		CALL GAUSEL (MAX,N,WK,N,EA,IERR)	EXP01100
0		IF (IERR .NE. 0) GO TO 130	EXP01110
1		IF (M .EQ. 0) GO TO 130	EXP01120
2	C ****		EXP01130
3	C	TAKE OUT EFFECT OF NORMALIZATION ON EXP(A)	EXP01140
4	C ****		EXP01150
5		DO 120 K=1,M	EXP01160
6		DO 110 I=1,N	EXP01170
7		DO 110 J=1,N	EXP01180
8		S = 0.	EXP01190
9		DO 105 L=1,N	EXP01200
0		S = S + EA(I,L)*EA(L,J)	EXP01210
1	105	CONTINUE	EXP01220
2		WK(I,J) = S	EXP01230
3	110	CONTINUE	EXP01240
4		DO 115 I=1,N	EXP01250
5		DO 115 J=1,N	EXP01260

26		EA(I,J) = WK(I,J)	EXP01270
7	115	CONTINUE	EXP01280
24	120	CONTINUE	EXP01290
29	C	****	EXP01300
0	C	UN-NORMALIZE A	EXP01310
1	C	****	EXP01320
32		DO 122 I=1,N	EXP01330
3		DO 122 J=1,N	EXP01340
4		A(I,J) = A(I,J)*FACTOR	EXP01350
35	122	CONTINUE	EXP01360
36		GO TO 130	EXP01370
7	C	****	EXP01380
38	C	NORM OF A IS EXCESSIVE	EXP01390
39	C	****	EXP01400
0	125	CONTINUE	EXP01410
1		IERR = 1	EXP01420
42	C	****	EXP01430
43	C	EXIT ROUTINE	EXP01440
44	C	****	EXP01450
45	130	CONTINUE	EXP01460
46		RETURN	EXP01470
47		END	EXP01480

0		SUBROUTINE EXPINT(A,NA,B,NB,C,NC,T,IOP,DUMMY)	EXP00
1		IMPLICIT REAL*8 (A-H,O-Z)	EXP00
2		DIMENSION A(1),B(1),C(1),DUMMY(1)	EXP00
3		DIMENSION NA(2),NB(2),NC(2)	EXP00
4		COMMON/CONV/SUMCV,RICTCV,SERCV,MAXSUM	EXP00
5		N = NA(1)	EXP00
6		L = (N**2)+1	EXP00
7		NC(1) = NA(1)	EXP00
8		NC(2) = NA(2)	EXP00
9		NB(1) = NA(1)	EXP00
10		NB(2) = NA(2)	EXP00
11		TT = T	EXP00
12	C		EXP00
13		IOPT = 1	EXP00
14		CALL NORMS(N,N,N,A,IOP, COL)	EXP00
15		IOPT = 3	EXP00
16		CALL NORMS(N,N,N,A,IOP, ROW)	EXP00
17		ANAA = COL	EXP00
18		IF( ANAA .GT. ROW ) ANAA = ROW	EXP00
19		TMAX = 1./ANAA	EXP00
20		K = 0	EXP00
21	100	CONTINUE	EXP00
22		IF( TMAX - TT ) 125,150,150	EXP00
23	125	CONTINUE	EXP00
24		K = K + 1	EXP00
25		TT = T/(2**K)	EXP00
26		IF( K = 1000 )100,600,600	EXP00
27	C		EXP00
28	150	CONTINUE	EXP00
29		SC = TT	EXP00
30		CALL SCALE(A,NA,A,NA,TT)	EXP00
31		CALL UNITY(B,NB)	EXP00
32		CALL SCALE(B,NB,DUMMY,NB,TT)	EXP00
33		S = TT/2.	EXP00
34		CALL SCALE(A,NA,DUMMY(L),NA,S)	EXP00
35		II = 2	EXP00
36		CALL ADD(DUMMY,NA,DUMMY(L),NA,DUMMY(L),NA)	EXP00
37		CALL ADD(A,NA,B,NB,DUMMY,NA)	EXP00
38		CALL EQUATE(A,NA,C,NC)	EXP00
39	200	CONTINUE	EXP00
40		CALL MULT(A,NA,C,NC,B,NB)	EXP00
41		S = 1./II	EXP00
42		CALL SCALE(B,NB,C,NC,S)	EXP00
43		CALL MAXEL(DUMMY,NA,TOT)	EXP00
44		CALL MAXEL(C,NC,DELT)	EXP00
45		IF( TOT .GT. 1.0 ) GO TO 300	EXP00
46		IF( DELT/TOT .LT. SERCV ) GO TO 400	EXP00
47		GO TO 350	EXP00
48	300	CONTINUE	EXP00
49		IF( DELT .LT. SERCV ) GO TO 400	EXP00
50	350	CONTINUE	EXP00
51		S = TT/(II + 1)	EXP00
52		CALL SCALE(C,NC,B,NB,S)	EXP00
53		CALL ADD(B,NB,DUMMY(L),NB,DUMMY(L),NB)	EXP00
54		CALL ADD(C,NC,DUMMY,NC,DUMMY,NC)	EXP00
55		II = II + 1	EXP00
56		GO TO 200	EXP00
57	C		EXP00
58	400	CONTINUE	EXP00
59		CALL EQUATE(DUMMY,NB,B,NB)	EXP00
60		IF( K ) 425,500,450	EXP00
61	425	CONTINUE	EXP00
62		CALL LNCNT(1)	EXP00

3	PRINT 435	EXP00640
4	435 FORMAT(' ERROR IN EXPINT, K IS NEGATIVE')	EXP00650
65	RETURN	EXP00660
46	C	EXP00670
7	450 CONTINUE	EXP00680
68	DO 475 J = 1,K	EXP00690
69	TT = 2*TT	EXP00700
0	CALL EQUATE(B,NB,DUMMY,NB)	EXP00710
1	CALL MULT(DUMMY,NA,DUMMY(L),NA,C,NC)	EXP00720
72	CALL ADD(DUMMY(L),NC,C,NC,DUMMY(L),NC)	EXP00730
73	CALL MULT(DUMMY,NB,DUMMY,NB,B,NB)	EXP00740
4	475 CONTINUE	EXP00750
75	T = TT	EXP00760
76	C	EXP00770
7	500 CONTINUE	EXP00780
78	CALL EQUATE(DUMMY(L),NC,C,NC)	EXP00790
79	S = 1./SC	EXP00800
80	CALL SCALE(A,NA,A,NA,S)	EXP00810
81	C	EXP00820
82	IF( IOP .EQ. 0 ) RETURN	EXP00830
93	CALL LNCNT(5)	EXP00840
84	PRINT 550	EXP00850
85	550 FORMAT('/', ' COMPUTATION OF THE MATRIX EXPONENTIAL EXP(A T)', '/', ' A	EXP00860
86	1ND ITS INTEGRAL OVER (0,T) BY THE SERIES METHOD ',/)	EXP00870
87	CALL PRNT(A,NA,4H A ,1)	EXP00880
88	CALL LNCNT(3)	EXP00890
89	PRINT 575, T	EXP00900
90	575 FORMAT('/', ' T = ',D16.8,/,)	EXP00910
91	CALL PRNT(B,NB,4HEXPA,1)	EXP00920
92	CALL PRNT(C,NC,4HINT ,1)	EXP00930
93	RETURN	EXP00940
94	C	EXP00950
95	600 CONTINUE	EXP00960
96	CALL LNCNT(1)	EXP00970
97	PRINT 650	EXP00980
98	650 FORMAT(' ERROR IN EXPINT, K = 1000')	EXP00990
99	RETURN	EXP01000
100	C	EXP01010
101	END	EXP01020

0	SUBROUTINE VARANC(A,NA,G,NG,Q,NQ,W,NW,IDENT,DISC,IOP,DUMMY)	VAR0001
1	IMPLICIT REAL*8 (A-H,O-Z)	VAR0002
2	DIMENSION A(1),G(1),Q(1),W(1),DUMMY(1)	VAR0003
3	DIMENSION NA(2),NG(2),NQ(2),NW(2),NDUM1(2),IOP(3),IOPT(2)	VAR0004
4	LOGICAL IDENT,DISC,SYM	VAR0005
5	COMMON/TOL/EPSAM,EPSBM,IACM	VAR0006
6	IF( IOP(1) .EQ. 0 ) GO TO 100	VAR0007
7	CALL LNCNT(5)	VAR0008
8	IF( DISC ) PRINT 25	VAR0009
9	IF( .NOT. DISC ) PRINT 35	VAR0010
10	25 FORMAT('/', ' PROGRAM TO SOLVE FOR THE STEADY-STATE VARIANCE MATRIX '	VAR0011
11	1,/, ' FOR A LINEAR DISCRETE SYSTEM',/)	VAR0012
12	35 FORMAT('/', ' PROGRAM TO SOLVE FOR THE STEADY-STATE VARIANCE MATRIX '	VAR0013
13	1,/, ' FOR A LINEAR CONTINUOUS SYSTEM',/)	VAR0014
14	CALL PRNT(A,NA,4H A ,1)	VAR0015
15	IF( .NOT. IDENT ) GO TO 55	VAR0016
16	CALL LNCNT(3)	VAR0017
17	PRINT 45	VAR0018
18	45 FORMAT('/', ' G IS AN IDENTITY MATRIX ',/)	VAR0019
19	GO TO 65	VAR0020
20	55 CONTINUE	VAR0021
21	CALL PRNT(G,NG,4H G ,1)	VAR0022
22	65 CONTINUE	VAR0023
23	IF ( .NOT. IDENT ) GO TO 85	VAR0024
24	CALL LNCNT(3)	VAR0025
25	PRINT 75	VAR0026
26	75 FORMAT('/', ' INTENSITY MATRIX FOR COVARIANCE OF PROCESS NOISE ',/)	VAR0027
27	C	VAR0028
28	85 CONTINUE	VAR0029
29	CALL PRNT(Q,NQ,4H Q ,1)	VAR0030
30	C	VAR0031
31	100 CONTINUE	VAR0032
32	IF( IDENT ) GO TO 200	VAR0033
33	CALL MULT(G,NG,Q,NQ,DUMMY,NG)	VAR0034
34	N1 = NG(1)*NG(2) + 1	VAR0035
35	CALL TRANP(G,NG,DUMMY(N1),NDUM1)	VAR0036
36	CALL MULT(DUMMY,NG,DUMMY(N1),NDUM1,Q,NQ)	VAR0037
37	C	VAR0038
38	IF( IOP(1) .EQ. 0 ) GO TO 200	VAR0039
39	CALL LNCNT(3)	VAR0040
40	PRINT 75	VAR0041
41	CALL PRNT(Q,NQ,4H QGT,1)	VAR0042
42	C	VAR0043
43	200 CONTINUE	VAR0044
44	IF(.NOT. DISC) CALL SCALE(W,NW,W,NW,-1.0)	VAR0045
45	IOPT(1) = IOP(2)	VAR0046
46	IOPT(2) = 1	VAR0047
47	SYM = .TRUE.	VAR0048
48	IF( DISC ) GO TO 300	VAR0049
49	IF( IOP(3) .EQ. 0 ) GO TO 250	VAR0050
50	CALL BILIN(A,NA,A,NA,W,NW,IOPT,BETA,SYM,DUMMY)	VAR0051
51	GO TO 400	VAR0052
52	C	VAR0053
53	250 CONTINUE	
54	CALL BARSTW(A,NA,A,NA,W,NW,IOPT,SYM,EPSA,EPSA,DUMMY)	VAR0054
55	GO TO 400	VAR0055
56	C	VAR0056
57	300 CONTINUE	VAR0057
58	CALL EQUATE(A,NA,DUMMY,NA)	VAR0058
59	N = NA(1)*2	VAR0059
60	N1 = N + 1	VAR0060
61	CALL TRANP(A,NA,DUMMY(N1),NA)	VAR0061
62	N2 = N1 + N	VAR0062

63	CALL SUM(DUMMY,NA,W,NW,DUMMY(N1),NA,IOP,SYM,DUMMY(N2))	VAR006'
64	C	VAR006'
65	400 CONTINUE	VAR006'
66	IF( IOP(1) .EQ. 0 ) RETURN	VAR006'
67	CALL LNCNT(3)	VAR006'
68	PPRINT 450	VAR007'
69	450 FORMAT(/, ' VARIANCE MATRIX ',/)	VAR007'
70	CALL PRNT(W,NW,4H W ,1)	VAR007'
71	C	VAR007'
72	RETURN	VAR007'
73	END	VAR007'

0	SUBROUTINE CTROL(A,NA,B,NB,C,NC,IOP,IAC,IRANK,DUMMY)	CTR000
1	IMPLICIT REAL*8 (A-H,O-Z)	CTR000
2	DIMENSION A(1),B(1),C(1),DUMMY(1)	CTR000
3	DIMENSION NA(2),NB(2),NC(2),NV(2),IOP(5)	CTR000
4	N = NA(1)*NB(2)	CTR000
5	N1 = N+1	CTR000
6	N2 = N1+N	CTR000
7	K = NA(1)-1	CTR000
8	J = 1	CTR000
9	C	CTR001
10	CALL EQUATE(B,NB,DUMMY(N2),NV)	CTR001
11	CALL EQUATE(B,NB,DUMMY,NB)	CTR001
12	100 CONTINUE	CTR001
13	CALL MULT(A,NA,DUMMY,NB,DUMMY(N1),NB)	CTR001
14	CALL JUXTC(DUMMY(N2),NV,DUMMY(N1),NB,C,NC)	CTR001
15	C	CTR001
16	IF( J .EQ. K ) GO TO 200	CTR001
17	C	CTR001
18	CALL EQUATE(DUMMY(N1),NB,DUMMY,NB)	CTR001
19	CALL EQUATE(C,NC,DUMMY(N2),NV)	CTR002
20	J = J + 1	CTR002
21	GO TO 100	CTR002
22	C	CTR002
23	200 CONTINUE	CTR002
24	C	CTR002
25	IF(IOP(1) .EQ. 0 ) GO TO 300	CTR002
26	CALL PRNT(A,NA,4H A ,1)	CTR002
27	CALL PRNT(B,NB,4H B ,1)	CTR002
28	CALL LNCNT(4)	CTR002
29	PRINT 250	CTR002
30	250 FORMAT(//, ' THE MATRIX C IS THE CONTROLLABILITY MATRIX FOR THE A/	CTR003
31	1B PAIR',/)	CTR003
32	CALL PRNT(C,NC,4H C ,1)	CTR003
33	C	CTR003
34	300 IF( IOP(2) .EQ. 0 ) RETURN	CTR003
35	NOS = 0	CTR003
36	IOPT = 2	CTR003
37	K = NC(2)	CTR003
38	NC(2) = NB(2)*(NA(2)-NB(2)+1)	CTR003
39	N = NC(1)*NC(2)	CTR004
40	CALL TRANP(C,NC,DUMMY,NV)	CTR004
41	NC(2) = K	CTR004
42	N1 = N + 1	CTR004
43	N2 = N1 + NV(2)	CTR004
44	CALL SNVDEC(IOPT,NV(1),NV(2),NV(1),NV(2),DUMMY,NOS,B,IAC,ZTEST,DUM	CTR004
45	1MY(N1),DUMMY(N2),IRANK,A,IERR)	CTR004
46	IF( IERR .EQ. 0 ) GO TO 340	CTR004
47	CALL LNCNT(5)	CTR004
48	IF( IERR .GT. 0 ) PRINT 310,IERR	CTR004
49	IF( IERR .EQ. -1 ) PRINT 320, ZTEST,IRANK	CTR005
50	310 FORMAT(//, ' IN CTROL, SNVDEC HAS FAILED TO CONVERGE TO THE ',I4, 'CTR005	CTR005
51	1 SINGULAR VALUE AFTER 30 ITERATIONS ')	CTR005
52	320 FORMAT(//, ' IN CTROL, THE MATRIX SUBMITTED TO SNVDEC USING ZTEST =CTR005	CTR005
53	1',D16.8,' IS CLOSE TO A MATRIX WHICH IS OF LOWER RANK',/, ' IF THE CTR005	CTR005
54	2ACCURACACY IS REDUCED THE RANK MAY ALSO BE REDUCED',/, ' CURRENT RACTR005	CTR005
55	3NK =',I4)	CTR005
56	IF( IERR .GT. 0 ) RETURN	CTR005
57	C	CTR005
58	340 CONTINUE	CTR005
59	IF( IOP(3) .EQ. 0 ) GO TO 400	CTR006
60	CALL LNCNT(6)	CTR006
61	PRINT 350,ZTEST,IRANK	CTR006
62	350 FORMAT(//, ' BASED ON THE ZERO-TEST ',D16.8,' THE RANK OF THE CONTRCTR006	CTR006



63	CONTROLLABILITY MATRIX IS ',I4,', THE SINGULAR VALUES ARE ',/')	CTR0
64	IOPT = 0	CTR0
65	NV(1) = NV(2)	CTR0
66	NV(2) = 1	CTR0
67	CALL PRNT(DUMMY(N1),NV,IOPT,3)	CTR0
68	C	CTR0
69	400 IF( IOP(4) .EQ. 0 ) RETURN	CTR0
70	N = NA(1)**2	CTR0
71	CALL EQUATE(DUMMY(N2),NA,DUMMY,NA)	CTR0
72	N1 = N + 1	CTR0
73	N2 = N1 + N	CTR0
74	CALL MULT(A,NA,DUMMY,NA,DUMMY(N1),NA)	CTR0
75	CALL TRANP(DUMMY,NA,DUMMY(N2),NA)	CTR0
76	CALL EQUATE(DUMMY(N2),NA,DUMMY,NA)	CTR0
77	CALL MULT(DUMMY,NA,DUMMY(N1),NA,DUMMY(N2),NA)	CTR0
78	CALL MULT(DUMMY,NA,B,NB,DUMMY(N1),NB)	CTR0
79	C	CTR0
80	IF( IOP(5) .EQ. 0 ) RETURN	CTR0
81	CALL LNCNT(5)	CTR0
82	PRINT 500	CTR0
83	500 FORMAT('/', ' CONTROLLABILITY CANONICAL FORM ',/, ' (V TRANSPOSE) A	CTR0
84	1 V')	CTR0
85	CALL PRNT(DUMMY(N2),NA,IOPT,3)	CTR0
86	CALL LNCNT(2)	CTR0
87	PRINT 510	CTR0
88	510 FORMAT('/', ' (V TRANSPOSE ) B '	CTR0
89	CALL PRNT(DUMMY(N1),NB,IOPT,3)	CTR0
90	CALL LNCNT(2)	CTR0
91	PRINT 520	CTR0
92	520 FORMAT('/', ' V TRANSPOSE')	CTR0
93	CALL PRNT(DUMMY,NA,IOPT,3)	CTR0
94	C	CTR0
95	RETURN	CTR0
96	END	CTR0

ORIGINAL PAGE IS  
OF POOR QUALITY

0	SUBROUTINE TRNSIT(A,NA,B,NB,H,NH,G,NG,F,NF,V,NV,T,X,NX,DISC,STABL	TRN000
1	IE,IOP,DUMMY)	TRN000
2	IMPLICIT REAL*8 (A-H,O-Z)	TRN000
3	DIMENSION A(1),B(1),H(1),G(1),F(1),V(1),X(1),DUMMY(1)	TRN000
4	DIMENSION NA(2),NB(2),NH(2),NG(2),NF(2),NV(2),NX(2),T(2),IOP(4)	TRN000
5	DIMENSION NDUM1(2),NDUM2(2)	TRN000
6	LOGICAL DISC,STABLE	TRN000
7	N = NA(1)*NA(2)	TRN000
8	N1 = N + 1	TRN000
9	N2 = N + N1	TRN001
10	N3 = N + N2	TRN001
11	N4 = N + N3	TRN001
12	N5 = N + N4	TRN001
13	N6 = N + N5	TRN001
14	C	TRN001
15	CALL LNCNT(4)	TRN001
16	IF(DISC) PRINT 100	TRN001
17	IF( .NOT. DISC ) PRINT 120	TRN001
18	100 FORMAT(//, ' COMPUTATION OF TRANSIENT RESPONSE FOR THE DIGITAL SYSTEM	TRN001
19	1EM ',/)	TRN002
20	120 FORMAT(//, ' COMPUTATION OF TRANSIENT RESPONSE FOR THE CONTINUOUS	TRN002
21	1 SYSTEM',/)	TRN002
22	CALL PRNT(A,NA,4H A ,1)	TRN002
23	CALL PRNT(B,NB,4H B ,1)	TRN002
24	IF( (IOP(1) .NE. 1) .AND. (IOP(1) .NE. 0) ) GO TO 180	TRN002
25	CALL LNCNT(3)	TRN002
26	IF( IOP(1) .EQ. 0 ) PRINT 140	TRN002
27	IF( IOP(1) .EQ. 1 ) PRINT 160	TRN002
28	140 FORMAT(//, ' H IS A NULL MATRIX ')	TRN002
29	160 FORMAT(//, ' H IS AN IDENTITY MATRIX ')	TRN003
30	GO TO 200	TRN003
31	180 CONTINUE	TRN003
32	CALL PRNT(H,NH,4H H ,1)	TRN003
33	200 CONTINUE	TRN003
34	IF( (IOP(2) .NE. 1) .AND. (IOP(2) .NE. 0) ) GO TO 260	TRN003
35	CALL LNCNT(3)	TRN003
36	IF( IOP(2) .EQ. 0 ) PRINT 220	TRN003
37	IF( IOP(2) .EQ. 1 ) PRINT 240	TRN003
38	220 FORMAT(//, ' G IS A NULL MATRIX ')	TRN003
39	240 FORMAT(//, ' G IS AN IDENTITY MATRIX ')	TRN004
40	GO TO 280	TRN004
41	260 CONTINUE	TRN004
42	CALL PRNT(G,NG,4H G ,1)	TRN004
43	280 CONTINUE	TRN004
44	CALL PRNT(F,NF,4H F ,1)	TRN004
45	IF( (IOP(3) .NE. 0) .AND. (IOP(3) .NE. 1) ) GO TO 295	TRN004
46	CALL LNCNT(3)	TRN004
47	IF(IOP(3).EQ.0) PRINT 285	TRN004
48	IF(IOP(3).EQ.1) PRINT 290	TRN004
49	285 FORMAT(//, ' V IS A NULL MATRIX ')	TRN005
50	290 FORMAT(//, ' V IS AN IDENTITY MATRIX ')	TRN005
51	GO TO 300	TRN005
52	295 CONTINUE	TRN005
53	CALL PRNT(V,NV,4H V ,1)	TRN005
54	C	TRN005
55	300 CONTINUE	TRN005
56	CALL EQUATE(A,NA,DUMMY(N6),NA)	TRN005
57	CALL MULT(B,NB,F,NF,DUMMY,NA)	TRN005
58	CALL SUBT(A,NA,DUMMY,NA,A,NA)	TRN005
59	C	TRN006
60	IF(DISC) GO TO 350	TRN006
61	NMAX = T(1)/T(2)	TRN006
62	IOPT = 1	TRN006

63	TT = T(2)	TRN00
64	IF( IOP(3) .NE. 0 ) GO TO 315	TRN00
65	CALL EXPSER(A,NA,DUMMY,NA,TT,IOP,DUMMY(N1))	TRN00
66	GO TO 400	TRN00
67	315 CONTINUE	TRN00
68	CALL EXPINT(A,NA,DUMMY,NA,DUMMY(N1),NA,TT,IOP,DUMMY(N2))	TRN00
69	CALL MULT(DUMMY(N1),NA,B,NB,DUMMY(N2),NB)	TRN00
70	IF( IOP(3) .NE. 1 ) GO TO 325	TRN00
71	CALL EQUATE(DUMMY(N2),NB,DUMMY(N1),NX)	TRN00
72	GO TO 400	TRN00
73	325 CONTINUE	TRN00
74	CALL MULT(DUMMY(N2),NB,V,NV,DUMMY(N1),NX)	TRN00
75	GO TO 400	TRN00
76	350 CONTINUE	TRN00
77	NMAX = IOP(4)	TRN00
78	CALL EQUATE(A,NA,DUMMY,NA)	TRN00
79	IF( IOP(3) .EQ. 0 ) GO TO 400	TRN00
80	CALL MULT(B,NB,V,NV,DUMMY(N1),NX)	TRN00
81	C	TRN00
82	400 CONTINUE	TRN00
83	CALL LNCNT(4)	TRN00
84	PRINT 420	TRN00
85	420 FORMAT(//, ' STRUCTURE OF PRINTING TO FOLLOW',//)	TRN00
86	CALL LNCNT(6)	TRN00
87	PRINT 440	TRN00
88	440 FORMAT(' TIME OR STAGE ',//, ' STATE = X TRANSPOSE = FROM DX = AX	TRN00
89	1 + BU',//, ' OUTPUT = Y TRANSPOSE = FROM Y = HX + GU IF DIFFERENT	TRN00
90	2 FROM X',//, ' CONTROL = U TRANSPOSE = FROM U = -FX + V',//)	TRN00
91	C	TRN00
92	K = 0	TRN00
93	L = 0	TRN00
94	CALL SCALE(F,NF,F,NF,-1,0)	TRN00
95	C	TRN00
96	450 CONTINUE	TRN00
97	IF( K .GT. NMAX ) GO TO 800	TRN00
98	CALL MULT(F,NF,X,NX,DUMMY(N2),NV)	TRN00
99	IF( IOP(3) .NE. 0 ) CALL ADD(DUMMY(N2),NV,V,NV,DUMMY(N2),NV)	TRN01
100	CALL MULT(DUMMY,NA,X,NX,DUMMY(N3),NX)	TRN01
101	IF( IOP(3) .EQ. 0 ) GO TO 475	TRN01
102	CALL ADD(DUMMY(N1),NX,DUMMY(N3),NX,DUMMY(N3),NX)	TRN01
103	475 CONTINUE	TRN01
104	IF( IOP(2) .EQ. 0 ) GO TO 525	TRN01
105	IF( IOP(2) .EQ. 1 ) GO TO 500	TRN01
106	CALL MULT(G,NG,DUMMY(N2),NV,DUMMY(N4),NDUM1)	TRN01
107	GO TO 525	TRN01
108	500 CONTINUE	TRN01
109	CALL EQUATE(DUMMY(N2),NV,DUMMY(N4),NDUM1)	TRN01
110	525 CONTINUE	TRN01
111	IF( IOP(1) .EQ. 0 ) GO TO 575	TRN01
112	IF( IOP(1) .EQ. 1 ) GO TO 550	TRN01
113	CALL MULT(H,NH,X,NX,DUMMY(N5),NDUM1)	TRN01
114	GO TO 575	TRN01
115	550 CONTINUE	TRN01
116	CALL EQUATE(X,NX,DUMMY(N5),NDUM1)	TRN01
117	575 CONTINUE	TRN01
118	IF( IOP(2) .EQ. 0 ) GO TO 600	TRN01
119	IF( IOP(1) .EQ. 0 ) GO TO 700	TRN01
120	CALL ADD(DUMMY(N4),NDUM1,DUMMY(N5),NDUM1,DUMMY(N4),NDUM1)	TRN01
121	GO TO 700	TRN01
122	600 CONTINUE	TRN01
123	IF( IOP(1) .NE. 0 ) CALL EQUATE(DUMMY(N5),NDUM1,DUMMY(N4),NDUM1)	TRN01
124	C	TRN01
125	700 CONTINUE	TRN01

126		CALL LNCNT(5)	TRN0
127		IF( .NOT. DISC ) GO TO 720	TRN0
128		PRINT 710,K	TRN0
129	710	FORMAT(////,15)	TRN0
130		GO TO 740	TRN0
131	720	CONTINUE	TRN0
132		TIME=K*T(2)	TRN0
133		PRINT 730,TIME	TRN0
134	730	FORMAT(////,D16.7)	TRN0
135	740	CONTINUE	TRN0
136		CALL TRANP(X,NX,DUMMY(N5),NDUM2)	TRN0
137		CALL PRNT(DUMMY(N5),NDUM2,L,3)	TRN0
138		IF( (IOP(2) .EQ. 0) .AND. ( (IOP(1) .EQ. 0) .OR. (IOP(1) .EQ. 1) )	TRN0
139	1)	GO TO 750	TRN0
140		CALL TRANP(DUMMY(N4),NDUM1,DUMMY(N5),NDUM2)	TRN0
141		CALL PRNT(DUMMY(N5),NDUM2,L,3)	TRN0
142	750	CONTINUE	TRN0
143		CALL TRANP(DUMMY(N2),NV,DUMMY(N5),NDUM2)	TRN0
144		CALL PRNT(DUMMY(N5),NDUM2,L,3)	TRN0
145	C		TRN0
146		CALL EQUATE(DUMMY(N3),NX,X,NX)	TRN0
147		K = K + 1	TRN0
148		GO TO 450	TRN0
149	C		TRN0
150	C		TRN0
151	800	CONTINUE	TRN0
152		CALL SCALE(F,NF,F,NF,-1.0)	TRN0
153		IF( .NOT. STABLE .OR. IOP(3) .EQ. 0 ) GO TO 900	TRN0
154		IF( IOP(3) .EQ. 1 ) GO TO 820	TRN0
155		CALL MULT(B,NB,V,NV,DUMMY,NX)	TRN0
156		GO TO 840	TRN0
157	820	CONTINUE	TRN0
158		CALL EQUATE(B,NB,DUMMY,NX)	TRN0
159	840	CONTINUE	TRN0
160		IF( .NOT. DISC ) GO TO 860	TRN0
161		CALL UNITY(DUMMY(N1),NA)	TRN0
162		CALL SUBT(DUMMY(N1),NA,A,NA,A,NA)	TRN0
163	860	CONTINUE	TRN0
164		IFAC = 0	TRN0
165		CALL GELIM(NA(1),NA(1),A,NX(2),DUMMY,DUMMY(N1),IFAC,DUMMY(N2),IERR	TRN0
166	1)		TRN0
167		IF( IERR .EQ. 0 ) GO TO 880	TRN0
168		CALL LNCNT(3)	TRN0
169		IF( .NOT. DISC ) PRINT 865	TRN0
170		IF( DISC ) PRINT 870	TRN0
171	865	FORMAT(//,' IN TRNSIT, THE MATRIX A-BF SUBMITTED TO GELIM IS SINGU	TRN0
172	1)	LAR')	TRN0
173	870	FORMAT(//,' IN TRNSIT, THE MATRIX I - (A-BF) SUBMITTED TO GELIM I	TRN0
174	1)	S SINGULAR')	TRN0
175		GO TO 900	TRN0
176	880	CONTINUE	TRN0
177		IF( .NOT. DISC ) CALL SCALE(DUMMY,NX,DUMMY,NX,-1.0)	TRN0
178		CALL LNCNT(5)	TRN0
179		PRINT 890	TRN0
180	890	FORMAT(////,' STEADY-STATE VALUE OF X TRANSPOSE')	TRN0
181		CALL TRANP(DUMMY,NX,DUMMY(N5),NDUM2)	TRN0
182		CALL PRNT(DUMMY(N5),NDUM2,L,3)	TRN0
183	C		TRN0
184	900	CONTINUE	TRN0
185		CALL EQUATE(DUMMY(N6),NA,A,NA)	TRN0
186	C		TRN0
187		RETURN	TRN0
188		END	TRN0

0	SUBROUTINE SAMPL(A,NA,B,NB,Q,NQ,R,NR,W,NW,T,IOP,DUMMY)	SAM000
1	IMPLICIT REAL*8 (A-H,O-Z)	SAM000
2	DIMENSION A(1),R(1),Q(1),R(1),W(1),DUMMY(1)	SAM000
3	DIMENSION NA(2),NB(2),NQ(2),NR(2),NW(2),IOP(2),NDUM(2)	SAM000
4	COMMON/CONV/SUMCV,RICTCV,SERCV,MAXSUM	SAM000
5	IF( IOP(1) .EQ. 0 ) GO TO 100	SAM000
6	C	SAM000
7	IF( IOP(2) .EQ. 0 ) GO TO 50	SAM000
8	C	SAM000
9	CALL LNCNT(5)	SAM000
10	PRINT 25	SAM000
11	25 FORMAT(/,,' COMPUTATION OF WEIGHTING MATRICES FOR THE OPTIMAL SAMP	SAM000
12	1LED-DATA REGULATOR PROBLEM',//)	SAM000
13	CALL PRNT(A,NA,4H A ,1)	SAM000
14	CALL PRNT(B,NB,4H B ,1)	SAM000
15	CALL LNCNT(3)	SAM000
16	PRINT 35	SAM000
17	35 FORMAT(/,,' CONTINUOUS PERFORMANCE INDEX WEIGHTING MATRICES',//)	SAM000
18	CALL PRNT(Q,NQ,4H Q ,1)	SAM000
19	CALL PRNT(R,NR,4H R ,1)	SAM002
20	CALL LNCNT(3)	SAM002
21	PRINT 45,T	SAM002
22	45 FORMAT(/,,' SAMPLE TIME = ',D16.8,//)	SAM002
23	C	SAM002
24	GO TO 100	SAM002
25	C	SAM002
26	50 CONTINUE	SAM002
27	CALL LNCNT(8)	SAM002
28	PRINT 75	SAM002
29	75 FORMAT(/,,' COMPUTATION OF THE RECONSTRUCTIBILITY GRAMIAN',//,,' FOR	SAM003
30	1 THE (A,H) SYSTEM OVER THE INTERVAL (0,T) ',//,,' THE MATRIX Q IS (	SAM003
31	2 H TRANSPOSE ) X H',//)	SAM003
32	CALL PRNT(A,NA,4H A ,1)	SAM003
33	CALL PRNT(Q,NQ,4H Q ,1)	SAM003
34	CALL LNCNT(3)	SAM003
35	PRINT 85,T	SAM003
36	85 FORMAT(/,,' T = ',D16.8,//)	SAM003
37	C	SAM003
38	100 CONTINUE	SAM003
39	C	SAM004
40	N = NA(1)	SAM004
41	L = ( N**2)	SAM004
42	N1 = L + 1	SAM004
43	N2 = N1 + L	SAM004
44	TT = T	SAM004
45	C	SAM004
46	IOPT = 1	SAM004
47	CALL NORMS(N,N,N,A,IOPT,ANORM)	SAM004
48	IOPT = 3	SAM004
49	CALL NORMS(N,N,N,A,IOPT,ROWA)	SAM005
50	IF( ANORM .GT. ROWA ) ANORM = ROWA	SAM005
51	C	SAM005
52	IF( ANORM .LE. 1.E-15 ) GO TO 900	SAM005
53	C	SAM005
54	TMAX = 1.0/ANORM	SAM005
55	K = 0	SAM005
56	C	SAM005
57	125 CONTINUE	SAM005
58	IF( TMAX - TT ) 150,150,200	SAM005
59	150 CONTINUE	SAM006
60	K = K + 1	SAM006
61	TT = T/( 2**K)	SAM006
62	IF( K = 1000 ) 125,800,800	SAM006

63	C		SAM00
64		200 CONTINUE	SAM00
65	C		SAM00
66		I = 0	SAM00
67		SC = TT	SAM00
68		CALL SCALE(A,NA,A,NA,TT)	SAM00
69		CALL SCALE(Q,NQ,Q,NQ,TT)	SAM00
70		CALL EQUATE(Q,NQ,DUMMY,NQ)	SAM00
71	C		SAM00
72		IF( IOP(2) .NE. 0 ) GO TO 500	SAM00
73	C		SAM00
74		225 CONTINUE	SAM00
75		II = I + 2	SAM00
76		I = I + 1	SAM00
77		F = 1.0/II	SAM00
78		CALL SCALE(A,NA,DUMMY(N1),NA,F)	SAM00
79		CALL MULT(DUMMY,NA,DUMMY(N1),NA,DUMMY(N2),NA)	SAM00
80		CALL TRANP(DUMMY(N2),NA,DUMMY(N1),NA)	SAM00
81		CALL ADD(DUMMY(N1),NA,DUMMY(N2),NA,DUMMY,NA)	SAM00
82	C		SAM00
83		CALL MAXEL(Q,NQ,TOT)	SAM00
84		CALL MAXEL(DUMMY,NA,DELT)	SAM00
85		IF( TOT .GT. 1.0 ) GO TO 250	SAM00
86		IF( DELT/TOT .LT. SERCV ) GO TO 300	SAM00
87		GO TO 275	SAM00
88		250 CONTINUE	SAM00
89		IF( DELT .LT. SERCV ) GO TO 300	SAM00
90		275 CONTINUE	SAM00
91		CALL ADD(Q,NQ,DUMMY,NA,Q,NQ)	SAM00
92		GO TO 225	SAM00
93	C		SAM00
94		300 CONTINUE	SAM00
95	C		SAM00
96		IF( K .EQ. 0 ) GO TO 400	SAM00
97		N3 = N2 + L	SAM00
98		G = 1.0	SAM00
99		IOPT = 0	SAM01
100		CALL EXPSER(A,NA,DUMMY,NA,G,IOPT,DUMMY(N1))	SAM01
101	C		SAM01
102		350 CONTINUE	SAM01
103		IF( K .EQ. 0 ) GO TO 400	SAM01
104		K = K-1	SAM01
105	C		SAM01
106		CALL TRANP(DUMMY,NA,DUMMY(N1),NA)	SAM01
107		CALL MULT(Q,NQ,DUMMY,NA,DUMMY(N2),NA)	SAM01
108		CALL MULT(DUMMY(N1),NA,DUMMY(N2),NA,DUMMY(N3),NA)	SAM01
109		CALL ADD(Q,NQ,DUMMY(N3),NA,Q,NQ)	SAM01
110		CALL MULT(DUMMY,NA,DUMMY,NA,DUMMY(N1),NA)	SAM01
111		CALL EQUATE(DUMMY(N1),NA,DUMMY,NA)	SAM01
112	C		SAM01
113		GO TO 350	SAM01
114	C		SAM01
115		400 CONTINUE	SAM01
116		S = 1.0/SC	SAM01
117		CALL SCALE(A,NA,A,NA,S)	SAM01
118	C		SAM01
119		IF( IOP(1) .EQ. 0 ) RETURN	SAM012
120		CALL PRNT(Q,NQ,4HGRAM,1)	SAM012
121		RETURN	SAM012
122	C		SAM012
123		500 CONTINUE	SAM012
124		CALL SCALE(B,NB,B,NB,TT)	SAM012
125		N3 = N2 + L	SAM012

ORIGINAL PAGE IS  
OF POOR QUALITY

126		N4 = N3 + L	SAM012
127		N5 = N4 + L	SAM012
128		N6 = N5 + L	SAM012
129	C		SAM013
130	525	CONTINUE	SAM013
131		II = I + 2	SAM013
132		I = I + 1	SAM013
133		F = 1.0/II	SAM013
134		CALL SCALE(A,NA,DUMMY(N1),NA,F)	SAM013
135		CALL TRANP(DUMMY(N1),NA,DUMMY(N2),NA)	SAM013
136		CALL MULT(DUMMY,NA,DUMMY(N1),NA,DUMMY(N3),NA)	SJM013
137		CALL TRANP(DUMMY(N3),NA,DUMMY(N1),NA)	SJM013
138		CALL MULT(DUMMY,NA,B,NB,DUMMY(N5),NW)	SAM013
139		CALL ADD(DUMMY(N1),NA,DUMMY(N3),NA,DUMMY,NA)	SAM014
140		CALL SCALE(DUMMY(N5),NW,DUMMY(N1),NW,F)	SAM014
141		IF( I .NE. 1 ) GO TO 550	SAM014
142		CALL EQUATE(DUMMY(N1),NW,W,NW)	SJM014
143		CALL EQUATE(DUMMY(N1),NW,DUMMY(N6),NW)	SAM014
144		CALL ADD(Q,NQ,DUMMY,NQ,Q,NQ)	SAM014
145		GO TO 525	SAM014
146	C		SAM014
147	550	CONTINUE	SAM014
148		CALL MULT(DUMMY(N2),NA,DUMMY(N6),NW,DUMMY(N5),NW)	SAM014
149		CALL ADD(DUMMY(N5),NW,DUMMY(N1),NW,DUMMY(N1),NW)	SAM015
150		CALL TRANP(B,NB,DUMMY(N2),NDUM)	SAM015
151		CALL SCALE(DUMMY(N2),NDUM,DUMMY(N2),NDUM,F)	SAM015
152		CALL MULT(DUMMY(N2),NDUM,DUMMY(N6),NW,DUMMY(N3),NR)	SAM015
153		CALL TRANP(DUMMY(N3),NR,DUMMY(N5),NR)	SAM015
154		CALL ADD(DUMMY(N3),NR,DUMMY(N5),NR,DUMMY(N3),NR)	SAM015
155		CALL EQUATE(DUMMY(N1),NW,DUMMY(N6),NW)	SAM015
156		IF( I .NE. 2 ) GO TO 575	SAM015
157		CALL ADD(Q,NQ,DUMMY,NQ,Q,NQ)	SAM015
158		CALL ADD(W,NW,DUMMY(N1),NW,W,NW)	SAM015
159		CALL EQUATE(DUMMY(N3),NR,DUMMY(N4),NR)	SAM016
160		GO TO 525	SJM016
161	C		SAM016
162	575	CONTINUE	SAM016
163		CALL MAXEL(Q,NQ,TOT)	SAM016
164		CALL MAXEL(DUMMY,NQ,DELT)	SAM016
165		IF( TOT .GT. 1.0 ) GO TO 580	SAM016
166		IF( DELT/TOT .LT. SERCV ) GO TO 585	SAM016
167		GO TO 595	SAM016
168	C		SAM016
169	580	CONTINUE	SAM017
170		IF( DELT .LT. SERCV ) GO TO 585	SAM017
171		GO TO 595	SAM017
172	C		SAM017
173	585	CONTINUE	SAM017
174		CALL MAXEL(DUMMY(N4),NR,TOT)	SAM017
175		CALL MAXEL(DUMMY(N3),NR,DELT)	SAM017
176		IF( TOT .GT. 1.0 ) GO TO 590	SAM017
177		IF( DELT/TOT .LT. SERCV ) GO TO 600	SAM017
178		GO TO 595	SAM017
179	C		SAM018
180	590	CONTINUE	SAM018
181		IF( DELT .LT. SERCV ) GO TO 600	SAM018
182	C		SAM018
183	595	CONTINUE	SAM018
184		CALL ADD(Q,NQ,DUMMY,NQ,Q,NQ)	SAM018
185		CALL ADD(W,NW,DUMMY(N1),NW,W,NW)	SAM018
186		CALL ADD(DUMMY(N4),NR,DUMMY(N3),NR,DUMMY(N4),NR)	SAM018
187		GO TO 525	SAM018
188	C		SAM018

189	600	CONTINUE	SAM01
190		IF( K .EQ. 0 ) GO TO 700	SAM01
191		G = 1.0	SAM01
192		IOPT = 0	SAM01
193		CALL EXPINT(A,NA,DUMMY,NA,DUMMY(N1),NA,G,IOPT,DUMMY(N2))	SAM01
194		CALL MULT(DUMMY(N1),NA,B,NB,DUMMY(N2),NB)	SAM01
195		CALL EQUATE(DUMMY(N2),NB,DUMMY(N1),NB)	SAM01
196	C		SAM01
197	650	CONTINUE	SAM01
198		IF( K .EQ. 0 ) GO TO 700	SAM01
199		K = K - 1	SAM02
200		CALL MULT(Q,NQ,DUMMY,NA,DUMMY(N2),NA)	SAM02
201		CALL TRANP(DUMMY,NA,DUMMY(N3),NA)	SAM02
202		CALL MULT(DUMMY(N3),NA,DUMMY(N2),NA,DUMMY(N5),NA)	SAM02
203		CALL MULT(Q,NQ,DUMMY(N1),NB,DUMMY(N2),NB)	SAM02
204		CALL ADD(Q,NQ,DUMMY(N5),NA,Q,NQ)	SAM02
205		CALL MULT(DUMMY(N3),NA,DUMMY(N2),NB,DUMMY(N5),NB)	SAM02
206		CALL MULT(DUMMY(N3),NA,W,NW,DUMMY(N6),NW)	SAM02
207		CALL ADD(DUMMY(N5),NW,DUMMY(N6),NW,DUMMY(N5),NW)	SAM02
208		CALL TRANP(DUMMY(N1),NB,DUMMY(N6),NDUM)	SAM02
209		CALL MULT(DUMMY(N6),NDUM,W,NW,DUMMY(N3),NR)	SAM02
210		CALL ADD(W,NW,DUMMY(N5),NW,W,NW)	SAM02
211		CALL MULT(DUMMY(N6),NDUM,DUMMY(N2),NB,DUMMY(N5),NR)	SAM02
212		CALL ADD(DUMMY(N5),NR,DUMMY(N3),NR,DUMMY(N5),NR)	SAM02
213		CALL TRANP(DUMMY(N3),NR,DUMMY(N6),NR)	SAM02
214		CALL ADD(DUMMY(N5),NR,DUMMY(N6),NR,DUMMY(N6),NR)	SAM02
215		CALL SCALE(DUMMY(N4),NR,DUMMY(N4),NR,2.0)	SAM02
216		CALL ADD(DUMMY(N6),NR,DUMMY(N4),NR,DUMMY(N4),NR)	SAM02
217		CALL MULT(DUMMY,NA,DUMMY(N1),NB,DUMMY(N3),NB)	SAM02
218		CALL ADD(DUMMY(N3),NB,DUMMY(N1),NB,DUMMY(N1),NB)	SAM02
219		CALL MULT(DUMMY,NA,DUMMY,NA,DUMMY(N3),NA)	SAM02
220		CALL EQUATE(DUMMY(N3),NA,DUMMY,NA)	SAM02
221		GO TO 650	SAM02
222	C		SAM02
223	700	CONTINUE	SAM02
224		CALL SCALE(R,NR,R,NR,T)	SAM02
225		CALL ADD(R,NR,DUMMY(N4),NR,R,NR)	SAM02
226		CALL SCALE(W,NW,W,NW,2.0)	SAM02
227		S = 1.0/SC	SAM02
228		CALL SCALE(A,NA,A,NA,S)	SAM02
229		CALL SCALE(B,NB,B,NB,S)	SAM02
230		IF( IOP(1) .EQ. 0 ) RETURN	SAM02
231	C		SAM02
232		CALL LNCNT(3)	SAM02
233		PRINT 750	SAM02
234	750	FORMAT(/,' DISCRETE PERFORMANCE INDEX WEIGHTING MATRICES',/)	SAM02
235		CALL PRNT(Q,NQ,4H Q ,1)	SAM02
236		CALL PRNT(W,NW,4H W ,1)	SAM02
237		CALL PRNT(R,NR,4H R ,1)	SAM02
238		RETURN	SAM02
239	C		SAM02
240	800	CONTINUE	SAM02
241		CALL LNCNT(1)	SAM02
242		PRINT 850	SAM02
243	850	FORMAT(' ERROR IN SAMPL , K = 1000')	SAM02
244		RETURN	SAM02
245	C		SAM02
246	900	CONTINUE	SAM02
247		CALL SCALE(Q,NQ,Q,NQ,T)	SAM02
248		IF( IOP(2) .NE. 0 ) GO TO 925	SAM02
249		IF( IOP(1) .NE. 0 ) CALL PRNT(Q,NQ,4HGRAM,1)	SAM02
250		RETURN	SAM02
51	C		SAM02



252	925 CONTINUE	SAM02
253	CALL MULT(Q,NQ,B,NB,W,NW)	SAM02
254	CALL SCALE(W,NW,W,NW,T)	SAM02
255	CALL TRANP(B,NB,DUMMY,NDUM)	SAM02
256	CALL MULT(DUMMY,NDUM,W,NW,DUMMY(N1),NP)	SAM02
257	TT = T/3.	SJM02
258	CALL SCALE(DUMMY(N1),NR,DUMMY,NR,TT)	SAM02
259	CALL SCALE(R,NR,R,NR,T)	SAM02
260	CALL ADD(R,NR,DUMMY,NR,R,NR)	SAM02
261	IF( IOP(1) .EQ. 0 ) RETURN	SAM02
262	CALL LNCNT(3)	SAM02
263	PRINT 750	SAM02
264	CALL PRNT(Q,NQ,4H Q ,1)	SAM02
265	CALL PRNT(W,NW,4H W ,1)	SAM02
266	CALL PRNT(R,NR,4H R ,1)	SAM02
267	RETURN	SAM02
268 C		SAM02
269	END	SAM02

```

0      SUBROUTINE PREFIL(A,NA,B,NB,Q,NQ,W,NW,R,NR,F,NF,IOP,DUMMY)      PRE00
1      IMPLICIT REAL*8 (A-H,O-Z)      PRE00
2      DIMENSION A(1),B(1),Q(1),W(1),R(1),F(1),DUMMY(1)      PRE00
3      DIMENSION NA(2),NB(2),NQ(2),NW(2),NR(2),NF(2),IOP(3)      PRE00
4      IF( IOP(1) .EQ. 0 ) GO TO 100      PRE00
5      CALL LNCNT(5)      PRE00
6      PRINT 25      PRE00
7      25 FORMAT(//, ' PROGRAM TO COMPUTE PREFILTER GAIN F TO ELEMIMATE CRUSPRE00
8      IS-PRODUCT TERM ',/, ' IN QUADRATIC PERFORMANCE INDEX ',/)      PRE00
9      IF( IOP(3) .EQ. 0 ) GO TO 50      PRE00
10     CALL PRNT(A,NA,4H A ,1)      PRE00
11     CALL PRNT(B,NB,4H B ,1)      PRE00
12     50 CONTINUE      PRE00
13     CALL PRNT(Q,NQ,4H Q ,1)      PRE00
14     CALL PRNT(W,NW,4H W ,1)      PRE00
15     CALL PRNT(R,NR,4H R ,1)      PRE00
16 C      PRE00
17     100 CONTINUE      PRE00
18     CALL TRANP(W,NW,F,NF)      PRE00
19     CALL SCALE(F,NF,F,NF,0.5)      PRE00
20     CALL EQUATE(R,NR,DUMMY,NR)      PRE00
21     IOPT=0      PRE00
22     IFAC=0      PRE00
23     N1=NR(1)**2+1      PRE00
24     M = NR(1)      PRE00
25     CALL SYMPDS(M,M,DUMMY,NF(2),F,IOPT,IFAC,DETERM,ISCALE,DUMMY(N1),IEPRE00
26     IRR)      PRE00
27     IF( IERR .EQ. 0 ) GO TO 200      PRE00
28     CALL LNCNT(4)      PRE00
29     PRINT 150      PRE00
30     150 FORMAT(//, ' IN PREFIL, THE MATRIX R IS NOT SYMMETRIC POSITIVE DEF:PRE00
31     INITE',/)      PRE00
32     RETURN      PRE00
33 C      PRE00
34     200 CONTINUE      PRE00
35     IF( IOP(2) .EQ. 0 ) GO TO 300      PRE00
36     CALL MULT(W,NW,F,NF,DUMMY,NQ)      PRE00
37     CALL SCALE(DUMMY,NQ,DUMMY,NQ,0.5)      PRE00
38     CALL SUBT(Q,NQ,DUMMY,NQ,Q,NQ)      PRE00
39 C      PRE00
40     300 CONTINUE      PRE00
41     IF( IOP(3) .EQ. 0 ) GO TO 400      PRE00
42     CALL MULT(B,NB,F,NF,DUMMY,NA)      PRE00
43     CALL SUBT(A,NA,DUMMY,NA,A,NA)      PRE00
44 C      PRE00
45     400 CONTINUE      PRE00
46     IF( IOP(1) .EQ. 0 ) RETURN      PRE00
47     CALL PRNT(F,NF,4H F ,1)      PRE00
48     IF( IOP(2) .EQ. 0 ) GO TO 500      PRE00
49     CALL LNCNT(3)      PRE005
50     PRINT 450      PRE005
51     450 FORMAT(/, ' MATRIX Q = (W/2)F ',/)      PRE005
52     CALL PRNT(Q,NQ,4HNEWQ,1)      PRE005
53 C      PRE005
54     500 CONTINUE      PRE005
55     IF( IOP(3) .EQ. 0 ) RETURN      PRE005
56     CALL PRNT(A,NA,4HNEWA,1)      PRE005
57     RETURN      PRE005
58     END      PRE005

```

0	SUBROUTINE CSTAB(A,NA,B,NB,F,NF,IOP,SCLE,DUMMY)	CST00
1	IMPLICIT REAL*8 (A-H,O-Z)	CST00
2	DIMENSION A(1),B(1),F(1),DUMMY(1)	CST00
3	DIMENSION NA(2),NB(2),NF(2),IOP(3),NDUM(2)	CST00
4	DIMENSION IOPT(2)	CST00
5	LOGICAL SYM	CST00
6	COMMON/TOL/EPSSM,EPSSM,IACM	CST00
7	N = NA(1)**2	CST00
8	N1=N+1	CST00
9	C	CST00
10	IF(IOP(2) .EQ. 0 ) GO TO 100	CST00
11	CALL EQUATE(A,NA,DUMMY,NA)	CST00
12	N2=N1+NA(1)	CST00
13	N3=N2+NA(1)	CST00
14	ISV = 0	CST 0
15	ILV = 0	CST00
16	CALL EIGEN(NA(1),NA(1),DUMMY,DUMMY(N1),DUMMY(N2),ISV,ILV,V,DUMMY(NCST00	
17	13),IERR)	CST00
18	C	CST00
19	M=NA(1)	CST00
20	IF(IERR .EQ. 0) GO TO 50	CST00
21	CALL LNCNT(3)	CST00
22	PRINT 25,IERR	CST00
23	25 FORMAT('/', ' IN CSTAB, THE SUBROUTINE EIGEN FAILED TO DETERMINE THE	CST00
24	1 ',I4,' EIGENVALUE FOR THE MATRIX A AFTER 30 ITERATIONS')	CST00
25	IERR=1	CST00
26	CALL NORMS(M,M,M,A,IERR,BETA)	CST00
27	BETA=2.*BETA	CST00
28	GO TO 200	CST00
29	50 CONTINUE	CST00
30	C	CST00
31	BETA = 0.0	CST00
32	DO 75 I = 1,M	CST00
33	J = N1 + I - 1	CST00
34	BETA1 = DABS(DUMMY(J))	CST00
35	IF(BETA1 .GT. BETA) BETA = BETA1	CST00
36	75 CONTINUE	CST00
37	BETA = SCLE*(BETA + .001)	CST00
38	GO TO 200	CST00
39	C	CST00
40	100 CONTINUE	CST00
41	BETA = SCLE	CST00
42	200 CONTINUE	CST00
43	C	CST00
44	CALL TRANP(B,NB,DUMMY,NDUM)	CST00
45	CALL MULT(B,NB,DUMMY,NDUM,DUMMY(N1),NA)	CST00
46	CALL SCALE(DUMMY(N1),NA,DUMMY,NA,-2.0)	CST00
47	CALL SCALE(A,NA,DUMMY(N1),NA,-1.0)	CST00
48	J = -NA(1)	CST00
49	NAX = NA(1)	CST00
50	DO 225 I=1,NAX	CST00
51	J = J+NAX+1	CST00
52	K = N1+J-1	CST00
53	DUMMY(K)=DUMMY(K)-BETA	CST00
54	225 CONTINUE	CST00
55	N2 = N1 + N	CST00
56	SYM = .TRUE.	CST00
57	IOPT(1)=0	CST00
58	C	CST00
59	IF( IOP(3) .NE. 0 ) GO TO 300	CST00
60	EPSA=EPSSM	CST00
61	CALL BARSTW(DUMMY(N1),NA,A,NA,DUMMY,NA,IOPT,SYM,EPSA,EPSSM,DUMMY(N2CST00	
62	1))	CST00

63	GO TO 350	CST01
64	300 CONTINUE	CST01
65	IOPT(2) = 1	CST01
66	CALL BILIN(DUMMY(N1),NA,A,NA,DUMMY,NA,IOPT,ASCLE,SYM,DUMMY(N2))	CST01
67	350 CONTINUE	CST01
68	C	CST01
69	CALL EQUATE(B,NB,DUMMY(N1),NB)	CST01
70	IOPT(1) = 3	CST01
71	IAC = IACM	CST01
72	N3 = N2 + NA(1)	CST01
73	CALL SNVDEC(IOPT,NA(1),NA(1),NA(1),NA(1),DUMMY,NB(2),DUMMY(N1),IAC	CST01
74	1,ZTEST,DUMMY(N2),DUMMY(N3),IRANK,APLUS,IERR)	CST01
75	IF(IERR .EQ. 0 ) GO TO 400	CST01
76	CALL LNCNT(5)	CST01
77	IF(IERR .GT. 0 ) PRINT 360,IERR	CST01
78	IF(IERR .EQ. -1) PRINT 370,ZTEST,IRANK	CST01
79	360 FORMAT(//,' IN CSTAB, SNVDEC HAS FAILED TO CONVERGE TO THE ',I4,'	CST01
80	1SINGULARVALUE AFTER 30 ITERATIONS',//)	CST01
81	370 FORMAT(//,' IN CSTAB, THE MATRIX SUBMITTED TO SNVDEC USING ZTEST =	CST01
82	1',D16.8,' IS CLOSE TO A MATRIX OF LOWER RANK ',//,' IF THE ACCUR	CST01
83	2ACY IAC IS REDUCED THE RANK MAY ALSO BE REDUCED',//,' CURRENT RANK	CST01
84	3 =',I4)	CST01
85	IF( IERR .GT. 0 ) RETURN	CST01
86	NDUM(1) = NA(1)	CST01
87	NDUM(2) = 1	CST01
88	CALL PRNT(DUMMY(N2),NDUM,4HSGVL,1)	CST01
89	400 CONTINUE	CST01
90	C	CST01
91	CALL TRANP(DUMMY(N1),NB,F,NF)	CST01
92	IF ( IOP(1) .EQ. 0 ) RETURN	CST01
93	CALL LNCNT(4)	CST01
94	PRINT 500	CST01
95	500 FORMAT(//,' COMPUTATION OF F MATRIX SUCH THAT A-BF IS ASYMPTOTICAL	CST01
96	1LY STABLE IN THE CONTINUOUS SENSE ',//)	CST01
97	CALL PRNT(A,NA,4H A ,1)	CST01
98	CALL LNCNT(4)	CST01
99	PRINT 550,BETA	CST01
100	550 FORMAT(//,' BETA = ',D16.8,//)	CST01
101	CALL PRNT(B,NB,4H B ,1)	CST01
102	CALL PRNT(F,NF,4H F ,1)	CST01
103	CALL MULT(B,NB,F,NF,DUMMY,NA)	CST01
104	CALL SUBT(A,NA,DUMMY,NA,DUMMY,NA)	CST01
105	CALL PRNT(DUMMY,NA,4HA-BF,1)	CST01
106	N2 = N1+NA(1)	CST01
107	N3 = N2+NA(1)	CST01
108	ISV = 0	CST01
109	ILV = 0	CST01
110	CALL EIGEN(NA(1),NA(1),DUMMY,DUMMY(N1),DUMMY(N2),ISV,ILV,V,DUMMY(N	CST01
111	13),IERR)	CST01
112	I = NA(1)	CST01
113	IF( IERR .EQ. 0 ) GO TO 600	CST01
114	M = NA(1)-IERR	CST01
115	CALL LNCNT(3)	CST01
116	PRINT 25,IERR	CST01
117	600 CONTINUE	CST01
118	CALL LNCNT(4)	CST01
119	PRINT 650	CST01
120	650 FORMAT(//,' EIGENVALUES OF A-BF',//)	CST01
121	675 FORMAT(10X,2D16.8)	CST01
122	CALL LNCNT(M)	CST01
123	DO 700 I=1,M	CST01
124	I = N1+I-1	CST01
125	K = N2+I-1	CST01

124 PRINT 675,DUMMY(J),DUMMY(K)

127 700 CONTINUE

128 C

129 RETURN

130 END

CS

CS

CS

CS

CS

```

0 SUBROUTINE DSTAB(A,NA,B,NB,F,NF,SING,IOP,SCLE,DUMMY) DST
1 IMPLICIT REAL*8 (A-H,O-Z) DST
2 DIMENSION A(1),B(1),F(1),DUMMY(1) DST
3 DIMENSION NA(2),NB(2),NF(2),NDUM(2),IOP(2),IOPT(3),NDUM1(2) DST
4 LOGICAL SING,SYM DST
5 COMMON/TOL/EPSAM,EPSTM,IACM DST
6 N = NA(1)**2 DST
7 N1 = N + 1 DST
8 N2 = N1 + N DST
9 IF( .NOT. SING ) GO TO 100 DST
10 IOPT(1)=IOP(1) DST
11 IOPT(2) = 1 DST
12 IOPT(3) = 0 DST
13 CSCLE=1.05 DST
14 CALL CSTAB(A,NA,B,NB,F,NF,IOPT,CSCLE,DUMMY) DST
15 CALL MULT(B,NB,F,NF,DUMMY,NA) DST
16 CALL SUBT(A,NA,DUMMY,NA,DUMMY,NA) DST
17 CALL EQUATE(DUMMY,NA,DUMMY(N1),NA) DST
18 GO TO 200 DST
19 C DST
20 100 CONTINUE DST
21 CALL EQUATE(A,NA,DUMMY,NA) DST
22 CALL EQUATE(A,NA,DUMMY(N1),NA) DST
23 C DST
24 200 CONTINUE DST
25 IF( IOP(2) .EQ. 0 ) GO TO 300 DST
26 N3 = N2 + NA(1) DST
27 N4 = N3 + NA(1) DST
28 ISV = 0 DST
29 CALL EIGEN(NA(1),NA(1),DUMMY(N1),DUMMY(N2),DUMMY(N3),ISV,ISV,V,DUM DST
30 1MY(N4),IERR) DST
31 CALL EQUATE(DUMMY,NA,DUMMY(N1),NA) DST
32 M = NA(1) DST
33 IF( IERR .EQ. 0 ) GO TO 250 DST
34 CALL LNCNT(3) DST
35 PRINT 225, IERR DST
36 225 FORMAT(// 'IN DSTAB , THE PROGRAM EIGEN FAILED TO DETERMINE', DST
37 1IS, 'EIGENVALUE FOR THE MATRIX A-BG AFTER 30 ITERATIONS ') DST
38 CALL PRNT(DUMMY,NA,4HA-BG,1) DST
39 IF( SING ) CALL PRNT(F,NF,4H G ,1) DST
40 RETURN DST
41 C DST
42 250 CONTINUE DST
43 ALPHA = 1.0 DST
44 DO 275 I =1,M DST
45 I1 = N2 + I -1 DST
46 I2 = N3 + I -1 DST
47 ALPHA1 = DSQRT(DUMMY(I1)**2 + DUMMY(I2)**2) DST
48 IF( ALPHA1 .LT. ALPHA .AND. ALPHA1 .NE. 0 ) ALPHA = ALPHA1 DST
49 275 CONTINUE DST
50 ALPHA = SCLE*ALPHA DST
51 GO TO 400 DST
52 C DST
53 300 CONTINUE DST
54 ALPHA = SCLE DST
55 C DST
56 400 CONTINUE DST
57 J = -NA(1) DST
58 NAX = NA(1) DST
59 DO 425 I = 1,NAX DST
60 J = J + NAX + 1 DST
61 K = N1 + J -1 DST
62 DUMMY(K) = DUMMY(J) - ALPHA DST

```

63		DUMMY(J) = DUMMY(J) + ALPHA	DST
64	425	CONTINUE	DST
65		CALL EQUATE(B,NB,DUMMY(N2),NB)	DST
66		N3 = N2 + NA(1)*NB(2)	DST
67		NRHS = NA(1)+NB(2)	DST
68		N4 = N3 + NA(1)	DST
69		IFAC = 0	DST
70		CALL GELIM(NA(1),NA(1),DUMMY,NRHS,DUMMY(N1),DUMMY(N3),IFAC,DUMMY(NOST	DST
71		14),IERR)	DST
72		IF( IERR .EQ. 0 ) GO TO 500	DST
73		CALL LNCNT(3)	DST
74		IF( .NOT. SING ) GO TO 445	DST
75		PRINT 435	DST
76	435	FORMAT(//, ' IN DSTAB, GELIM HAS FOUND THE MATRIX ( A-BG) + (ALPHA)	DST
77		1 I SINGULAR	DST
78		CALL PRNT(A,NA,4H A ,1)	DST
79		CALL PRNT(B,NB,4H B ,1)	DST
80		GO TO 465	DST
81	445	CONTINUE	DST
82		CALL LNCNT(3)	DST
83		PRINT 455	DST
84	455	FORMAT(//, ' IN DSTAB, GELIM HAS FOUND THE MATRIX A + (ALPHA)I SING	DST
85		1ULAR '	DST
86		CALL PRNT(A,NA,4H A ,1)	DST
87	465	CONTINUE	DST
88		CALL LNCNT(3)	DST
89		PRINT 475,ALPHA	DST
90	475	FORMAT(//, ' ALPHA = ',D16.8)	DST
91		RETURN	DST
92	C		DST
93	500	CONTINUE	DST
94		CALL EQUATE(DUMMY(N1),NA,DUMMY,NA)	DST
95		CALL TRANSP(DUMMY(N2),NB,DUMMY(N1),NDUM)	DST
96		N3 = N2 + N	DST
97		CALL MULT(DUMMY(N2),NB,DUMMY(N1),NDUM,DUMMY(N3),NA)	DST
98		CALL SCALE(DUMMY(N3),NA,DUMMY(N1),NA,4.0)	DST
99		SYM = .TRUE.	DST
100		IOPT(1) = 0	DST
101		EPSA=EPSAM	DST
102		CALL BARSTW(DUMMY,NA,B,NB,DUMMY(N1),NA,IOPT,SYM,EPSA,EPSA,DUMMY(N2)	DST
103		1))	DST
104		CALL EQUATE(DUMMY(N1),NA,DUMMY,NA)	DST
105		CALL TRANSP(B,NB,DUMMY(N1),NDUM)	DST
106		CALL MULT(B,NB,DUMMY(N1),NDUM,DUMMY(N2),NA)	DST
107		CALL ADD(DUMMY,NA,DUMMY(N2),NA,DUMMY,NA)	DST
108		CALL EQUATE(A,NA,DUMMY(N1),NA)	DST
109		IF( .NOT. SING ) GO TO 600	DST
110		CALL MULT(B,NB,F,NF,DUMMY(N1),NA)	DST
111		CALL SUBT(A,NA,DUMMY(N1),NA,DUMMY(N1),NA)	DST
112	C		DST
113	600	CONTINUE	DST
114		IOPT(1) = 3	DST
115		M = NA(1)	DST
116		IAC=IACM	DST
117		CALL SNVDEC(IOPT,M,M,M,M,DUMMY,M,DUMMY(N1),IAC,ZTEST,DUMMY(N2),DUM	DST
118		1MY(N3),IRANK,APLUS,IERR)	DST
119		IF( IERR .EQ. 0 ) GO TO 700	DST
120		CALL LNCNT(5)	DST
121		IF( IERR .GT. 0 ) PRINT 625,IERR	DST
122		IF( IERR .EQ. -1) PRINT 650,ZTEST,IRANK	DST
123	625	FORMAT(//, ' IN DSTAB, SNVDEC HAS FAILED TO CONVERGE TO THE ',IS,'	DST
124		1SINGULAR VALUE AFTER 30 ITERATIONS')	DST
125	650	FORMAT(//, ' IN DSTAB, THE MATRIX SUBMITTED TO SNVDEC, USING ZTEST	DST

```

126 1= ,D16.8, , IS CLOSE TO A MATRIX OF LOWER RANK,/, , IF THE ACCURST
127 2ACY IAC IS REDUCED THE RANK MAY ALSO BE REDUCED,/, , CURRENT RANK DST
128 3 = ,I4) DST
129 IF( IERR .GT. 0 ) RETURN DST
130 NDUM(1)= NA(1) DST
131 NDUM(2)= 1 DST
132 CALL PRNT(DUMMY(N2),NDUM,4HSGVL,1) DST
133 C DST
134 700 CONTINUE DST
135 CALL TRNPF(B,NB,DUMMY(N2),NDUM) DST
136 CALL MULT(DUMMY(N2),NDUM,DUMMY(N1),NA,DUMMY,NF) DST
137 IF( .NOT. SING ) GO TO 800 DST
138 CALL ADD(F,NF,DUMMY,NF,F,NF) DST
139 GO TO 900 DST
140 C DST
141 800 CONTINUE DST
142 CALL EQUATE(DUMMY,NF,F,NF) DST
143 C DST
144 900 CONTINUE DST
145 IF( IOP(1) .EQ. 0 ) RETURN DST
146 CALL LNCNT(4) DST
147 PRINT 1000 DST
148 1000 FORMAT(/, , COMPUTATION OF F SUCH THAT A-BF IS ASYMPTOTICALLY STABDST
149 1LE IN THE DISCRETE SENSE,/, ) DST
150 CALL PRNT(A,NA,4H A ,1) DST
151 CALL PRNT(B,NB,4H B ,1) DST
152 CALL LNCNT(4) DST
153 PRINT 1100,ALPHA DST
154 1100 FORMAT(/, , ALPHA = ,D16.8,/, ) DST
155 CALL PRNT(F,NF,4H F ,1) DST
156 CALL MULT(B,NB,F,NF,DUMMY,NA) DST
157 CALL SUBT(A,NA,DUMMY,NA,DUMMY,NA) DST
158 CALL PRNT(DUMMY,NA,4HA-BF,1) DST
159 CALL LNCNT(3) DST
160 PRINT 1200 DST
161 1200 FORMAT(/, , EIGENVALUES OF A-BF, ) DST
162 NDUM(1) = NA(1) DST
163 NDUM(2) = 1 DST
164 N2 = N1 + NA(1) DST
165 N3 = N2 + NA(1) DST
166 ISV = 0 DST
167 CALL EIGEN(NA(1),NA(1),DUMMY,DUMMY(N1),DUMMY(N2),ISV,ISV,V,DUMMY(NDST
168 13),IERR) DST
169 IF( IERR .EQ. 0 ) GO TO 1300 DST
170 CALL LNCNT(3) DST
171 PRINT 1250 DST
172 1250 FORMAT(/, , IN DSTAB, THE PROGRAM EIGEN FAILED TO DETERMINE THE ,DST
173 1IS, , EIGENVALUE FOR THE A-BF MATRIX AFTER 30 ITERATIONS, ) DST
174 NDUM(1)=NA(1)-IERR DST
175 C DST
176 1300 CONTINUE DST
177 CALL JUXTC(DUMMY(N1),NDUM,DUMMY(N2),NDUM,DUMMY,NDUM1) DST
178 CALL PRNT(DUMMY,NDUM1,4HEIGN,1) DST
179 CALL LNCNT(4) DST
180 PRINT 1400 DST
181 1400 FORMAT(/, , MODULI OF EIGENVALUES OF A-BF,/, ) DST
182 M =NDUM(1) DST
183 DO 1500 I = 1,M DST
184 J = N1 + I - 1 DST
185 K = N2 + I - 1 DST
186 DUMMY(I)=DSQRT(DUMMY(J)**2 + DUMMY(K)**2) DST
187 1500 CONTINUE DST
188 CALL PRNT(DUMMY,NDUM,4HMOD ,1) DST

```



187 C  
190  
191

RETURN  
END

DST  
DST  
DST

```

0 SUBROUTINE DISREG(A,NA,B,NB,H,NH,Q,NQ,R,NR,F,NF,P,NP,IOP,IDENT,OU DIS
1 IMMY) DIS
2 IMPLICIT REAL*8 (A-H,O-Z) DIS
3 DIMENSION A(1),B(1),Q(1),R(1),F(1),P(1),DUMMY(1) DIS
4 DIMENSION NA(2),NR(2),NQ(2),NR(2),NF(2),NP(2) DIS
5 DIMENSION IOP(3) DIS
6 DIMENSION H(1),NH(2),NDUM(2) DIS
7 LOGICAL IDENT DIS
8 COMMON/TOL/EP3AM,EP3BM,IACM DIS
9 COMMON/CONV/SUMCV,RICTCV,SERCV,MAXSUM DIS
10 N = NA(1)**2 DIS
11 N1 = N + 1 DIS
12 N2 = N1 + N DIS
13 N3 = N2 + N DIS
14 C DIS
15 KSS = 0 DIS
16 I = IOP(3) DIS
17 C DIS
18 IF(IOP(1) .EQ. 0) GO TO 85 DIS
19 CALL LNCNT(5) DIS
20 PRINT 25 DIS
21 25 FORMAT('/',,' PROGRAM TO SOLVE THE TIME-INVARIANT FINITE-DURATION OP DIS
22 1TIMAL',,' DIGITAL REGULATOR PROBLEM WITH NOISE-FREE MEASUREMENTS' DIS
23 2,/) DIS
24 CALL PRNT(A,NA,4H A ,1) DIS
25 CALL PRNT(B,NB,4H B ,1) DIS
26 CALL PRNT(Q,NQ,4H Q ,1) DIS
27 IF(.NOT. IDENT) GO TO 45 DIS
28 CALL LNCNT(3) DIS
29 PRINT 35 DIS
30 35 FORMAT('/',,' H IS AN IDENTITY MATRIX',/) DIS
31 GO TO 65 DIS
32 45 CONTINUE DIS
33 CALL PRNT(H,NH,4H H ,1) DIS
34 CALL MULT(Q,NQ,H,NH,DUMMY,NH) DIS
35 CALL TRANSP(H,NH,DUMMY(N1),NF) DIS
36 CALL MULT(DUMMY(N1),NF,DUMMY,NH,Q,NQ) DIS
37 CALL LNCNT(3) DIS
38 PRINT 55 DIS
39 55 FORMAT('/',,' MATRIX ( H TRANSPOSE )QH',/) DIS
40 CALL PRNT(Q,NQ,4HHTQH,1) DIS
41 65 CONTINUE DIS
42 CALL PRNT(R,NR,4H R ,1) DIS
43 CALL LNCNT(4) DIS
44 PRINT 75 DIS
45 75 FORMAT('/',,' WEIGHTING ON TERMINAL VALUE OF STATE VECTOR',/) DIS
46 CALL PRNT(P,NP,4H P ,1) DIS
47 C DIS
48 85 CONTINUE DIS
49 IF((IOP(1) .NE. 0) .OR. IDENT) GO TO 100 DIS
50 CALL MULT(Q,NQ,H,NH,DUMMY,NH) DIS
51 CALL TRANSP(H,NH,DUMMY(N1),NF) DIS
52 CALL MULT(DUMMY(N1),NF,DUMMY,NH,Q,NQ) DIS
53 C DIS
54 100 CONTINUE DIS
55 I = I - 1 DIS
56 CALL EQUATE(P,NP,DUMMY,NP) DIS
57 CALL MULT(P,NP,A,NA,DUMMY(N1),NA) DIS
58 CALL TRANSP(B,NB,DUMMY(N2),NF) DIS
59 CALL MULT(DUMMY(N2),NF,DUMMY(N1),NA,F,NF) DIS
60 CALL MULT(P,NP,B,NR,DUMMY(N1),NB) DIS
61 CALL MULT(DUMMY(N2),NF,DUMMY(N1),NB,DUMMY(N3),NR) DIS
62 CALL ADD(R,NR,DUMMY(N3),NR,DUMMY(N1),NR) DIS

```

```

63      IOPT = 3
64      IAC=IACM
65      MF = NP(1)
66      CALL SNVDEC(IOPT,MF,MF,MF,MF,DUMMY(N1),NF(2),F,IAC,ZTEST,DUMMY(N2))
67      1,DUMMY(N3),IRANK,APLUS,IERR)
68      IF( IERR .EQ.0) GO TO 300
69      CALL LNCNT(5)
70      IF(IERR .GT. 0) PRINT 200,IERR
71      IF(IERR .EQ. -1) PRINT 250,ZTEST,IRANK
72      200  FORMAT(//,' IN DISREG, SNVDEC HAS FAILED TO CONVERGE TO THE ',I4,
73      1'SINGULARVALUE AFTER 30 ITERATIONS',//)
74      250  FORMAT(//,' IN DISREG, THE MATRIX SUBMITTED TO SNVDEC USING ZTEST
75      1=' ,D16.8,' IS CLOSE TO A MATRIX OF LOWER RANK',//,' IF THE ACCURACY
76      2 IAC IS REDUCED THE RANK MAY ALSO BE REDUCED',//,' CURRENT RANK = '
77      3 ,I4)
78      IF( IERR .GT. 0 ) RETURN
79      NDUM(1) = NA(1)
80      NDUM(2) = 1
81      CALL PRNT(DUMMY(N2),NDUM,4HSGVL,1)
82 C
83      300 CONTINUE
84      CALL MULT(R,NR,F,NF,DUMMY(N1),NF)
85      CALL TRANP(F,NF,DUMMY(N2),NB)
86      CALL MULT(DUMMY(N2),NB,DUMMY(N1),NF,P,NP)
87      CALL ADD(Q,NQ,P,NP,P,NP)
88      CALL MULT(B,NB,F,NF,DUMMY(N1),NA)
89      CALL SUBT(A,NA,DUMMY(N1),NA,DUMMY(N1),NA)
90      CALL MULT(DUMMY,NA,DUMMY(N1),NA,DUMMY(N2),NA)
91      CALL TRANP(DUMMY(N1),NA,DUMMY(N3),NA)
92      CALL MULT(DUMMY(N3),NA,DUMMY(N2),NA,DUMMY(N1),NA)
93      CALL ADD(P,NP,DUMMY(N1),NA,P,NP)
94 C
95      IF( IOP(2) .EQ. 0 ) GO TO 400
96      CALL LNCNT(5)
97      PRINT 350,I
98      350  FORMAT(///,' STAGE ',I5,/)
99      CALL PRNT(F,NF,4H F ,1)
100     CALL PRNT(P,NP,4H P ,1)
101 C
102     400 CONTINUE
103     IF( I .EQ. 0 ) GO TO 600
104     CALL MAXEL(DUMMY,NP,ANORM1)
105     CALL SUBT(DUMMY,NP,P,NP,DUMMY(N2),NP)
106     CALL MAXEL(DUMMY(N2),NP,ANORM2)
107     IF( ANORM1 .NE. 0.0 ) GO TO 500
108     GO TO 100
109 C
110     500 CONTINUE
111     IF(ANORM1 .GT. 1.0 ) GO TO 550
112     IF( ANORM2/ANORM1 .LT. RICTCV ) KSS = 1
113     GO TO 575
114     550 CONTINUE
115     IF( ANORM2 .LT. RICTCV ) KSS=1
116     575 CONTINUE
117     IF( KSS .EQ. 1) GO TO 600
118     GO TO 100
119 C
120     600 CONTINUE
121     K = IOP(1) + IOP(2)
122     IF( K .EQ. 0 ) RETURN
123     IF( KSS .EQ. 0) GO TO 700
124     CALL LNCNT(4)
125     PRINT 650

```

126	650	FORMAT(//, ' STEADY-STATE SOLUTION HAS BEEN REACHED IN DISREG', /)	DIS01
127	C		DIS01
128	700	CONTINUE	DIS01
129		IF( IOP(2) .NE. 0 ) RETURN	DIS01
130		IF( IOP(1) .EQ. 0 ) RETURN	DIS01
131		CALL LNCNT(3)	DIS01
132		I = IOP(3)-I	DIS01
133		PRINT 800, I	DIS01
134	800	FORMAT(/, ' F AND P AFTER ', I5, ' STEPS', /)	DIS01
135		CALL PRNT(F, NF, 4H F , 1)	DIS01
136		CALL PRNT(P, NP, 4H P , 1)	DIS01
137		RETURN	DIS01
138		END	DIS01

```

0      SUBROUTINE CNTREG(A,NA,B,NB,H,NH,Q,NQ,R,NR,Z,W,LAMBDA,S,F,NF,P,NP CNT0
1      I,T,IOP,IDENT,DUMMY) CNT0
2      IMPLICIT REAL*8 (A-H,O-Z) CNT0
3 C CNT0
4      DIMENSION A(1),B(1),H(1),Q(1),R(1),Z(1),W(1),LAMBDA(1),S(1),F(1),P CNT0
5      I(1),T(1),DUMMY(1) CNT0
6      DIMENSION NA(2),NB(2),NH(2),NQ(2),NR(2),NF(2),NP(2),IOP(3),NDUM1(2 CNT0
7      I),NDUM2(2) CNT0
8      LOGICAL IDENT CNT0
9      REAL*8 LAMBDA CNT0
10     COMMON/CONV/SUMCV,RICTCV,SERCV,MAXSUM CNT0
11 C CNT0
12     IF( IOP(1) .EQ. 0 ) GO TO 65 CNT0
13     CALL LNCNT(5) CNT0
14     IF( IOP(3) .EQ. 0 ) PRINT 25 CNT0
15     25 EFORMAT(//' PROGRAM TO SOLVE THE TIME-INVARIANT FINITE-DURATION CONCNT0
16     I TINUOUS OPTIMAL'//' REGULATOR PROBLEM WITH NOISE-FREE MEASUREMENTS' CNT0
17     2) CNT0
18     IF( IOP(3) .NE. 0 ) PRINT 30 CNT0
19     30 FORMAT(//' PROGRAM TO SOLVE THE TIME-INVARIANT INFINITE-DURATION CNT0
20     I CONTINUOUS OPTIMAL'//' REGULATOR PROBLEM WITH NOISE-FREE MEASUREMENTS CNT0
21     2TS') CNT0
22     CALL PRNT(A,NA,4H A ,1) CNT0
23     CALL PRNT(B,NB,4H B ,1) CNT0
24     CALL PRNT(Q,NQ,4H Q ,1) CNT0
25     IF( .NOT. IDENT ) GO TO 45 CNT0
26     CALL LNCNT(3) CNT0
27     PRINT 35 CNT0
28     35 FORMAT(/' H IS AN IDENTITY MATRIX'//) CNT0
29     GO TO 55 CNT0
30 C CNT0
31     45 CONTINUE CNT0
32     CALL PRNT(H,NH,4H H ,1) CNT0
33     CALL MULT(Q,NQ,H,NH,DUMMY,NH) CNT0
34     N1= NH(1)*NH(2)+1 CNT0
35     CALL TRANP(H,NH,DUMMY(N1),NDUM1) CNT0
36     CALL MULT(DUMMY(N1),NDUM1,DUMMY,NH,Q,NQ) CNT0
37     CALL LNCNT(3) CNT0
38     PRINT 50 CNT0
39     50 FORMAT(//' MATRIX (H TRANSPOSE)QH'//) CNT0
40     CALL PRNT(Q,NQ,0,3) CNT0
41     55 CONTINUE CNT0
42     CALL PRNT(R,NR,4H R ,1) CNT0
43 C CNT0
44     IF( IOP(3) .NE. 0 ) GO TO 65 CNT0
45     CALL LNCNT(4) CNT0
46     PRINT 60 CNT0
47     60 FORMAT(//' WEIGHTING ON TERMINAL VALUE OF STATE VECTOR'//) CNT0
48     CALL PRNT(P,NP,4H P ,1) CNT0
49 C CNT0
50     65 CONTINUE CNT0
51     CALL EQUATE(R,NR,DUMMY,NR) CNT0
52     N = NA(1)*2 CNT0
53     N1 = NR(1)+NB(2)+1 CNT0
54     CALL TRANP(B,NB,DUMMY(N1),NDUM1) CNT0
55     N2 = N1 + N CNT0
56     L = NR(1) CNT0
57     IOPT = 0 CNT0
58     IFAC = 0 CNT0
59     CALL SYMPDS(L,L,DUMMY,NB(1),DUMMY(N1),IOPT,IFAC,DET,ISCALE,DUMMY(N CNT0
60     I2),IERR) CNT0
61 C CNT0
62     IF( IERR .EQ. 0 ) GO TO 100 CNT0

```



126	400	CONTINUE	CNT
127		CALL EQUATE(W,NDUM2,DUMMY(N1),NDUM2)	CNT
128		J1 = 1	CNT
129		J2 = 1	CNT
130		M = 2*N	CNT
131		NDUM1(1) = L	CNT
132		NDUM1(2) = 1	CNT
133		K4 = N4	CNT
134	C		CNT
135		I=1	CNT
136	415	CONTINUE	CNT
137		IF( I .GT. L ) GO TO 515	CNT
138		K1 = N2+I-1	CNT
139		K2 = N1+(I-1)*L	CNT
140		K3 = N3+I-1	CNT
141		IF(DUMMY(K1) .GT. 0.0 ) GO TO 425	CNT
142		J = (J1-1)*L+M+1	CNT
143		J1 = J1+1	CNT
144		IF(DUMMY(K3) .NE. 0.0 ) J1=J1+1	CNT
145		GO TO 450	CNT
146	425	CONTINUE	CNT
147		DUMMY(K4)=I	CNT
148		K4 = K4+1	CNT
149		J = (J2-1)*L+1	CNT
150		J2 = J2+1	CNT
151		IF( DUMMY(K3) .NE. 0.0 ) J2 = J2 + 1	CNT
152	450	CONTINUE	CNT
153		CALL EQUATE(DUMMY(K2),NDUM1,W(J),NDUM1)	CNT
154		IF(DUMMY(K3) .EQ. 0.0) GO TO 500	CNT
155		I = I+1	CNT
156		K2 = K2+L	CNT
157		J = J+L	CNT
158		CALL EQUATE(DUMMY(K2),NDUM1,W(J),NDUM1)	CNT
159	500	CONTINUE	CNT
160		I=I+1	CNT
161		GO TO 415	CNT
162	515	CONTINUE	CNT
163	C		CNT
164		CALL NULL(LAMBDA,NA)	CNT
165		K0 = -1	CNT
166		J = -NA(1)	CNT
167		NAX = NA(1)	CNT
168		I=1	CNT
169	520	CONTINUE	CNT
170		IF( I .GT. NAX ) GO TO 530	CNT
171		J = NAX + J + 1	CNT
172		K0 = K0 + 1	CNT
173		K1 = N4 + K0	CNT
174		K2 = DUMMY(K1)	CNT
175		K = N2+K2-1	CNT
176		LAMBDA(J) = DUMMY(K)	CNT
177		K3 = N3+K2-1	CNT
178		IF( DUMMY(K3) .EQ. 0.0 ) GO TO 525	CNT
179		K4 = J+1	CNT
180		LAMBDA(K4) = -DUMMY(K3)	CNT
181		K4 = K4+NAX	CNT
182		LAMBDA(K4) = DUMMY(K)	CNT
183		K4 = K4-1	CNT
184		LAMBDA(K4) = DUMMY(K3)	CNT
185		K5 = M + (I-1)*L + 1	CNT
186		K6 = K5 + L	CNT
187		CALL EQUATE(W(K5),NDUM1,DUMMY(N1),NDUM1)	CNT
188		CALL EQUATE(W(K6),NDUM1,W(K5),NDUM1)	CNT

189		CALL EQUATE(DUMMY(N1),NDUM1,W(K6),NDUM1)	CNT0
190		I = I+1	CNT0
191		J = NAX + J + 1	CNT0
192	525	CONTINUE	CNT0
193		I=I+1	CNT0
194		GO TO 520	CNT0
195	530	CONTINUE	CNT0
196	C		CNT0
197		IF( IOP(1) .EQ. 0 ) GO TO 700	CNT0
198		CALL LNCNT(3)	CNT0
199		PRINT 535	CNT0
200	535	FORMAT(// ' REORDERED EIGENVECTORS ')	CNT0
201		CALL PRNT(W,NDUM2,0,3)	CNT0
202		CALL LNCNT(4)	CNT0
203		PRINT 545	CNT0
204	545	FORMAT(// ' LAMBDA MATRIX OF EIGENVALUES OF Z WITH POSITIVE REAL PA	CNT0
205		RTS ')	CNT0
206		CALL PRNT(LAMBDA,NA,0,3)	CNT0
207	C		CNT0
208		CALL MULT(Z,NDUM2,W,NDUM2,DUMMY(N1),NDUM2)	CNT0
209		L = NDUM2(1)	CNT0
210		M = L**2	CNT0
211		N2 = N1+M	CNT0
212		CALL EQUATE(W,NDUM2,DUMMY(N2),NDUM2)	CNT0
213		N3 = N2+M	CNT0
214		N4 = N3+L	CNT0
215		IFAC = 0	CNT0
216		CALL GELIM(L,L,DUMMY(N2),L,DUMMY(N1),DUMMY(N3),IFAC,DUMMY(N4),IERR	CNT0
217		1)	CNT0
218		IF( IERR .EQ. 0 ) GO TO 600	CNT0
219		CALL LNCNT(4)	CNT0
220		PRINT 550	CNT0
221	550	FORMAT(// ' IN CNTREG, GELIM HAS FOUND THE REORDERED MATRIX W TO B	CNT0
222		E SINGULAR ')	CNT0
223	600	CONTINUE	CNT0
224		CALL PRNT(DUMMY(N1),NDUM2,4HWIZW,1)	CNT0
225	C		CNT0
226	700	CONTINUE	CNT0
227		NDUM1(1) = 2*NA(1)	CNT0
228		NDUM1(2) = NA(1)	CNT0
229		N2 = 2*N + N1	CNT0
230		CALL TRANP(W,NDUM1,DUMMY(N2),NDUM2)	CNT0
231		NW11 = N1	CNT0
232		NDUM1(1) = NA(1)	CNT0
233		CALL TRANP(DUMMY(N2),NDUM1,DUMMY(NW11),NDUM1)	CNT0
234		L = N2+N	CNT0
235		NW21 = NW11+N	CNT0
236		CALL TRANP(DUMMY(L),NDUM1,DUMMY(NW21),NDUM1)	CNT0
237		L = 2*N+1	CNT0
238		NDUM1(1)=2*NA(1)	CNT0
239		N3 = N2 + 2*N	CNT0
240		CALL TRANP(W(L),NDUM1,DUMMY(N3),NDUM2)	CNT0
241		NDUM1(1) = NA(1)	CNT0
242		NW12 = NW21+N	CNT0
243		CALL TRANP(DUMMY(N3),NDUM1,DUMMY(NW12),NDUM1)	CNT0
244		L = N3 + N	CNT0
245		NW22 = NW12 + N	CNT0
246		CALL TRANP(DUMMY(L),NDUM1,DUMMY(NW22),NDUM1)	CNT0
247	C		CNT0
248		IF( IOP(1) .EQ. 0 ) GO TO 800	CNT0
249		CALL PRNT(DUMMY(NW11),NA,4HW11 ,1)	CNT0
250		CALL PRNT(DUMMY(NW21),NA,4HW21 ,1)	CNT0
251		CALL PRNT(DUMMY(NW12),NA,4HW12 ,1)	CNT0



```

- 252 CALL PRNT(DUMMY(NW22),NA,4HW22 ,1) CNT0
253 C CNT0
254 900 CONTINUE CNT0
255 IF( IOP(3) .NE. 0 ) GO TO 900 CNT0
256 N2 = N1+4*N CNT0
257 CALL MULT(P, NP, DUMMY(NW12), NA, S, NA) CNT0
258 CALL MULT(P, NP, DUMMY(NW11), NA, DUMMY(N2), NA) CNT0
259 CALL SUBT(S, NA, DUMMY(NW22), NA, S, NA) CNT0
260 CALL SUBT(DUMMY(NW21), NA, DUMMY(N2), NA, DUMMY(N2), NA) CNT0
261 N3 = N2+N CNT0
262 L = NA(1) CNT0
263 IFAC = 0 CNT0
264 N4 = N3+NA(1) CNT0
265 CALL GELIM(L, L, DUMMY(N2), L, S, DUMMY(N3), IFAC, DUMMY(N4), IERR) CNT0
266 IF( IERR .EQ. 0 ) GO TO 850 CNT0
267 CALL LNCNT(4) CNT0
268 PRINT 825 CNT0
269 825 FORMAT(//' IN CNTREG, GELIM HAS FOUND THE MATRIX W21 = PIXW11 TO CNT0
270 BE SINGULAR'//) CNT0
271 RETURN CNT0
272 C CNT0
273 850 CONTINUE CNT0
274 IF( IOP(1) .EQ. 0 ) GO TO 1000 CNT0
275 CALL PRNT(S, NA, 4H S ,1) CNT0
276 NDUM1(1) = NR(1) CNT0
277 NDUM1(2) = NA(1) CNT0
278 CALL LNCNT(3) CNT0
279 PRINT 875 CNT0
280 875 FORMAT(//' MATRIX (R INVERSE)X(B TRANSPOSE)'//) CNT0
281 CALL PRNT(DUMMY, NDUM1, 0, 3) CNT0
282 GO TO 1000 CNT0
283 C CNT0
284 900 CONTINUE CNT0
285 N2 = N1+4*N CNT0
286 CALL TRANP(DUMMY(NW12), NA, DUMMY(N2), NA) CNT0
287 CALL TRANP(DUMMY(NW22), NA, P, NP) CNT0
288 N3 = N2+N CNT0
289 IFAC = 0 CNT0
290 L = NA(1) CNT0
291 N4 = N3 + NA(1) CNT0
292 CALL GELIM(L, L, DUMMY(N2), L, P, DUMMY(N3), IFAC, DUMMY(N4), IERR) CNT0
293 IF( IERR .EQ. 0 ) GO TO 950 CNT0
294 CALL LNCNT(4) CNT0
295 PRINT 925 CNT0
296 925 FORMAT(//' IN CNTREG, GELIM HAS FOUND THE MATRIX W12 TO BE SINGUL CNT0
297 AR'//) CNT0
298 RETURN CNT0
299 950 CONTINUE CNT0
300 NDUM1(1) = NR(1) CNT0
301 NDUM1(2) = NA(1) CNT0
302 CALL MULT(DUMMY, NDUM1, P, NP, F, NF) CNT0
303 IF( IOP(1) .EQ. 0 ) RETURN CNT0
304 CALL PRNT(P, NP, 4H P ,1) CNT0
305 CALL PRNT(F, NF, 4H F ,1) CNT0
306 RETURN CNT0
307 C CNT0
308 1000 CONTINUE CNT0
309 NMAX = T(1)/T(2) CNT0
310 I = NMAX CNT0
311 CALL EQUATE(LAMBDA, NA, DUMMY(N2), NA) CNT0
312 TT = -T(2) CNT0
313 N4 = N3+N CNT0
314 N5 = N4+N CNT0

```

```

315 N6 = N5+N CNT
316 N7 = N6+NA(1) CNT
317 KSS = 0 CNT
318 NDUM1(1) = NR(1) CNT
319 NDUM1(2) = NA(1) CNT
320 CALL EXPSER(DUMMY(N2),NA,DUMMY(N3),NA,TT,KSS,DUMMY(N4)) CNT
321 CALL EQUATE(DUMMY(N3),NA,DUMMY(N2),NA) CNT
322 IF( IOP(1) .EQ. 0 ) GO TO 1075 CNT
323 CALL LNCNT(3) CNT
324 PRINT 1050,T(2) CNT
325 1050 FORMAT(// ' EXP(-LAMBDA X ',D16.8,')' ) CNT
326 CALL PRNT(DUMMY(N2),NA,0,3) CNT
327 1075 CONTINUE CNT
328 IF( NMAX .LE. 0 ) RETURN CNT
329 CALL EQUATE(S,NA,DUMMY(N3),NA) CNT
330 1100 CONTINUE CNT
331 TIME = I*T(2) CNT
332 IF( I .NE. NMAX ) CALL EQUATE(DUMMY(N5),NA,P,NP) CNT
333 CALL MULT(DUMMY(N3),NA,DUMMY(N2),NA,DUMMY(N4),NA) CNT
334 CALL MULT(DUMMY(N2),NA,DUMMY(N4),NA,DUMMY(N3),NA) CNT
335 CALL MULT(DUMMY(NW11),NA,DUMMY(N3),NA,DUMMY(N4),NA) CNT
336 CALL ADD(DUMMY(NW12),NA,DUMMY(N4),NA,DUMMY(N4),NA) CNT
337 CALL TRANP(DUMMY(N4),NA,DUMMY(N5),NA) CNT
338 CALL EQUATE(DUMMY(N5),NA,DUMMY(N4),NA) CNT
339 CALL MULT(DUMMY(NW21),NA,DUMMY(N3),NA,DUMMY(N5),NA) CNT
340 CALL ADD(DUMMY(NW22),NA,DUMMY(N5),NA,DUMMY(N5),NA) CNT
341 CALL TRANP(DUMMY(N5),NA,DUMMY(N6),NA) CNT
342 CALL EQUATE(DUMMY(N6),NA,DUMMY(N5),NA) CNT
343 L = NA(1) CNT
344 IFAC = 0 CNT
345 CALL GELIM(L,L,DUMMY(N4),L,DUMMY(N5),DUMMY(N6),IFAC,DUMMY(N7),IERR CNT
346 1) CNT
347 IF( IERR .EQ. 0 ) GO TO 1200 CNT
348 CALL LNCNT(3) CNT
349 PRINT 1150,TIME CNT
350 1150 FORMAT(// ' IN CNTREG AT TIME ',D16.8, ' P CANNOT BE COMPUTED DUE T CNT
351 10 MATRIX SINGULARITY IN GELIM' ) CNT
352 RETURN CNT
353 C CNT
354 1200 CONTINUE CNT
355 CALL MAXEL(P,NP,ANORM1) CNT
356 CALL SUBT(DUMMY(N5),NA,P,NP,DUMMY(N4),NA) CNT
357 CALL MAXEL(DUMMY(N4),NA,ANORM2) CNT
358 IF( ANORM1 .NE. 0.0 ) GO TO 1225 CNT
359 GO TO 1300 CNT
360 C CNT
361 1225 CONTINUE CNT
362 IF(ANORM1 .GT. 1.0 ) GO TO 1250 CNT
363 IF( ANORM2/ANORM1 .LT. RICTCV ) KSS=1 CNT
364 GO TO 1300 CNT
365 1250 CONTINUE CNT
366 IF( ANORM2 .LT. RICTCV ) KSS=1 CNT
367 C CNT
368 1300 CONTINUE CNT
369 CALL MULT(DUMMY,NDUM1,P,NP,F,NF) CNT
370 IF( IOP(2) .EQ. 0 ) GO TO 1400 CNT
371 CALL LNCNT(5) CNT
372 PRINT 1350,TIME CNT
373 1350 FORMAT(/// ' TIME = ',D16.8/) CNT
374 CALL PRNT(P,NP,4H P ,1) CNT
375 IF( I .NE. NMAX ) CALL PRNT(F,NF,4H F ,1) CNT
376 C CNT
377 1400 CONTINUE CNT

```

- 378	IF( KSS .EQ. 1 ) GO TO 1500	CNT
379	I = I-1	CNT
380	IF( I .GE. 0 ) GO TO 1100	CNT
381	GO TO 1600	CNT
382	1500 CONTINUE	CNT
383	CALL LNCNT(4)	CNT
384	PRINT 1550	CNT
385	1550 FORMAT(// ' STEADY-STATE SOLUTION HAS BEEN REACHED IN CNTREG'//)	CNT
386	C	CNT
387	1600 CONTINUE	CNT
388	IF( IOP(2) .NE. 0 ) RETURN	CNT
389	IF( IOP(1) .EQ. 0 ) RETURN	CNT
390	CALL LNCNT(5)	CNT
391	PRINT 1350, TIME	CNT
392	CALL PRNT(P, NP, 4H P , 1)	CNT
393	CALL PRNT(F, NF, 4H F , 1)	CNT
394	C	CNT
395	RETURN	LNT
396	END	CNT

```

0 SUBROUTINE RICNWT(A,NA,B,NB,H,NH,Q,NQ,R,NR,F,NF,P,NP,IOP,IDENT,DI RIC00
1 ISC,FNULL,DUMMY) RIC00
2 IMPLICIT REAL*8 (A-H,O-Z) RIC00
3 DIMENSION A(1),B(1),Q(1),R(1),F(1),P(1),DUMMY(1) RIC00
4 DIMENSION NA(2),NB(2),NQ(2),NR(2),NF(2),NP(2),IOP(3) RIC00
5 DIMENSION H(1),NH(2),IOPT(2) RIC00
6 LOGICAL IDENT,DISC,FNULL,SYM RIC00
7 COMMON/TOL/EPSAM,EPSSM,IACM RIC00
8 COMMON/CONV/SUMCV,RICTCV,SERCV,MAXSUM RIC00
9 I=1 RIC00
10 IOPT(1)=0 RIC00
11 SYM = .TRUE. RIC00
12 C RIC00
13 N = NA(1)**2 RIC00
14 N1 = N + 1 RIC00
15 IF( .NOT. DISC) N1 = NA(1)*NR(1) + 1 RIC00
16 N2 = N1+N RIC00
17 N3 = N2+N RIC00
18 N4 = N3+N RIC00
19 C RIC00
20 IF( IOP(1) .EQ. 0 ) GO TO 210 RIC00
21 CALL LNCNT(4) RIC00
22 IF(.NOT. DISC)PRINT 100 RIC00
23 IF( DISC )PRINT 150 RIC00
24 100 FORMAT(//,' PROGRAM TO SOLVE CONTINUOUS STEADY-STATE RICCATI EQUATRIC00
25 ION BY THE NEWTON ALGORITHM',/) RIC00
26 150 FORMAT(//,' PROGRAM TO SOLVE DISCRETE STEADY-STATE RICCATI EQUATIORIC00
27 N BY THE NEWTON ALGORITHM',/) RIC00
28 CALL PRNT(A,NA,4H A ,1) RIC00
29 CALL PRNT(B,NB,4H B ,1) RIC00
30 CALL PRNT(Q,NQ,4H Q ,1) RIC00
31 IF( .NOT. IDENT )GO TO 185 RIC00
32 CALL LNCNT(3) RIC00
33 PRINT 180 RIC00
34 180 FORMAT(/,' H IS AN IDENTITY MATRIX',/) RIC00
35 GO TO 200 RIC00
36 185 CONTINUE RIC00
37 CALL PRNT(H,NH,4H H ,1) RIC00
38 CALL MULT(Q,NQ,H,NH,DUMMY,NH) RIC00
39 CALL TRANP(H,NH,DUMMY(N2),NP) RIC00
40 CALL MULT(DUMMY(N2),NP,DUMMY,NH,Q,NQ) RIC00
41 CALL LNCNT(3) RIC00
42 PRINT 195 RIC00
43 195 FORMAT(/,' MATRIX (H TRANSPOSE)QH ',/) RIC00
44 CALL PRNT(Q,NQ,4HHTQH,1) RIC00
45 200 CONTINUE RIC00
46 CALL PRNT(R,NR,4H R ,1) RIC00
47 IF( FNULL ) GO TO 210 RIC00
48 CALL LNCNT(3) RIC00
49 PRINT 205 RIC00
50 205 FORMAT(/,' INITIAL F MATRIX',/) RIC00
51 CALL PRNT(F,NF,4H F ,1) RIC00
52 C RIC00
53 210 CONTINUE RIC00
54 IF((IOP(1) .NE. 0) .OR. IDENT) GO TO 220 RIC00
55 CALL MULT(Q,NQ,H,NH,DUMMY,NH) RIC00
56 CALL TRANP(H,NH,DUMMY(N2),NP) RIC00
57 CALL MULT(DUMMY(N2),NP,DUMMY,NH,Q,NQ) RIC00
58 220 CONTINUE RIC00
59 C RIC00
60 IF( DISC) GO TO 900 RIC00
61 C RIC00
62 CALL TRANP(B,NB,P,NP) RIC00

```

63		CALL EQUATE(R,NR,DUMMY,NR)	RIC
64		CALL SYMPOS(NR(1),NR(1),DUMMY,NP(2),P,IOPT,IOPT,DET,ISCALE,DUMMY(NR(1)),IERR)	RIC
65		IF(IERR.EQ. 0) GO TO 250	RIC
66		CALL LNCNT(3)	RIC
67		PRINT 225	RIC
68	225	FORMAT(/,' IN RICNWT, A MATRIX WHICH IS NOT SYMMETRIC POSITIVE DERIC	RIC
69		IFINITE HAS BEEN SUBMITTED TO SYMPOS',/)	RIC
70		RETURN	RIC
71			RIC
72	C		RIC
73	250	CONTINUE	RIC
74		CALL EQUATE(P,NP,DUMMY,NF)	RIC
75		CALL MULT(B,NB,DUMMY,NF,DUMMY(N1),NA)	RIC
76		CALL TRANP(DUMMY(N1),NA,DUMMY(N2),NA)	RIC
77		CALL ADD(DUMMY(N1),NA,DUMMY(N2),NA,DUMMY(N1),NA)	RIC
78		CALL SCALE(DUMMY(N1),NA,DUMMY(N1),NA,0.5)	RIC
79	C		RIC
80		IF(FNULL) GO TO 300	RIC
81	C		RIC
82		CALL MULT(B,NB,F,NF,DUMMY(N2),NA)	RIC
83		CALL SUBT(A,NA,DUMMY(N2),NA,DUMMY(N2),NA)	RIC
84		CALL TRANP(DUMMY(N2),NA,DUMMY(N3),NA)	RIC
85		CALL EQUATE(DUMMY(N3),NA,DUMMY(N2),NA)	RIC
86		CALL MULT(R,NR,F,NF,DUMMY(N3),NF)	RIC
87		CALL TRANP(F,NF,P,NP)	RIC
88		CALL MULT(P,NP,DUMMY(N3),NF,DUMMY(N4),NA)	RIC
89		CALL TRANP(DUMMY(N4),NA,DUMMY(N3),NA)	RIC
90		CALL ADD(DUMMY(N4),NA,DUMMY(N3),NA,DUMMY(N3),NA)	RIC
91		CALL SCALE(DUMMY(N3),NA,DUMMY(N3),NA,0.5)	RIC
92		CALL ADD(DUMMY(N3),NA,Q,NQ,P,NP)	RIC
93		CALL SCALE(P,NP,P,NP,-1.0)	RIC
94		GO TO 350	RIC
95	C		RIC
96	300	CONTINUE	RIC
97		CALL TRANP(A,NA,DUMMY(N2),NA)	RIC
98		CALL SCALE(Q,NQ,P,NP,-1.0)	RIC
99	C		RIC
100	350	CONTINUE	RIC
101		IF(IOP(3).NE. 0) GO TO 400	RIC
102		EPSA= EPSAM	RIC
103		CALL BARSTW(DUMMY(N2),NA,B,NB,P,NP,IOPT,SYM,EPSA,EPSA,DUMMY(N3))	RIC
104		GO TO 450	RIC
105	C		RIC
106	400	CONTINUE	RIC
107		IOPT(2)=1	RIC
108		CALL BILIN(DUMMY(N2),NA,B,NB,P,NP,IOPT,SCLE,SYM,DUMMY(N3))	RIC
109	C		RIC
110	450	CONTINUE	RIC
111		CALL EQUATE(P,NP,DUMMY(N2),NP)	RIC
112		IF(IOP(2).EQ. 0) GO TO 550	RIC
113		CALL LNCNT(3)	RIC
114		PRINT 500,I	RIC
115	500	FORMAT(/,' ITERATION ',IS,/)	RIC
116		CALL PRNT(P,NP,4# P ,1)	RIC
117	C		RIC
118	550	CONTINUE	RIC
119		CALL MULT(DUMMY(N1),NA,P,NP,DUMMY(N3),NA)	RIC
120		CALL MULT(P,NP,DUMMY(N3),NA,DUMMY(N4),NA)	RIC
121		CALL TRANP(DUMMY(N4),NA,P,NA)	RIC
122		CALL ADD(P,NP,DUMMY(N4),NA,P,NP)	RIC
123		CALL SCALE(P,NP,P,NP,0.5)	RIC
124		CALL ADD(Q,NQ,P,NP,P,NP)	RIC
125		CALL SCALE(P,NP,P,NP,-1.0)	RIC

126		CALL SUBT(A,NA,DUMMY(N3),NA,DUMMY(N4),NA)	RIC01
127		CALL TRANSP(DUMMY(N4),NA,DUMMY(N3),NA)	RIC01
128	C		RIC01
129		IF(IOP(3) .NE. 0 ) GO TO 650	RIC01
130		CALL BARSTW(DUMMY(N3),NA,B,NB,P,NP,IOPT,SYM,EPSA,EPSA,DUMMY(N4))	RIC01
131		GO TO 675	RIC01
132	C		RIC01
133	650	CONTINUE	RIC01
134		CALL BILIN(DUMMY(N3),NA,B,NB,P,NP,IOPT,SCLE,SYM,DUMMY(N4))	RIC01
135	C		RIC01
136	675	CONTINUE	RIC01
137		I=I+1	RIC01
138		CALL MAXEL(DUMMY(N2),NA,ANORM1)	RIC01
139		CALL SUBT(P,NP,DUMMY(N2),NA,DUMMY(N3),NA)	RIC01
140		CALL MAXEL(DUMMY(N3),NA,ANORM2)	RIC01
141		IF(ANORM1 .GT. 1.0) GO TO 700	RIC01
142		IF( ANORM2/ANORM1 .LT. RICTCV ) GO TO 800	RIC01
143		GO TO 750	RIC01
144	C		RIC01
145	700	CONTINUE	RIC01
146		IF( ANORM2 .LT. RICTCV ) GO TO 800	RIC01
147	C		RIC01
148	750	CONTINUE	RIC01
149		IF( I .LE. 101) GO TO 450	RIC01
150		CALL LNCNT(3)	RIC01
151		PRINT 775	RIC01
152	775	FORMAT(/, ' THE SUBROUTINE RICNWT HAS EXCEEDED 100 ITERATIONS WITHORIC01	RIC01
153		UT CONVERGENCE',/)	RIC01
154		IOP(1) = 1	RIC01
155	C		RIC01
156	800	CONTINUE	RIC01
157		CALL MULT(DUMMY,NF,P,NP,F,NF)	RIC01
158		GO TO 1300	RIC01
159	C		RIC01
160	900	CONTINUE	RIC01
161		IF( .NOT. FNULL ) GO TO 950	RIC01
162	C		RIC01
163		CALL EQUATE(Q,NQ,P,NP)	RIC01
164		CALL EQUATE(A,NA,DUMMY(N1),NA)	RIC01
165		CALL TRANSP(A,NA,DUMMY(N2),NA)	RIC01
166		GO TO 1000	RIC01
167	925	CONTINUE	RIC01
168	C		RIC01
169		I=I+1	RIC01
170		CALL EQUATE(P,NP,DUMMY,NP)	RIC01
171	950	CONTINUE	RIC01
172	C		RIC01
173		CALL MULT(R,NR,F,NF,DUMMY(N1),NF)	RIC01
174		CALL TRANSP(F,NF,P,NP)	RIC01
175		CALL MULT(P,NP,DUMMY(N1),NF,DUMMY(N2),NA)	RIC01
176		CALL TRANSP(DUMMY(N2),NA,DUMMY(N1),NA)	RIC01
177		CALL ADD(DUMMY(N1),NA,DUMMY(N2),NA,DUMMY(N1),NA)	RIC01
178		CALL SCALE(DUMMY(N1),NA,DUMMY(N1),NA,0.5)	RIC01
179		CALL ADD(Q,NQ,DUMMY(N1),NA,P,NP)	RIC01
180		CALL MULT(B,NB,F,NF,DUMMY(N1),NA)	RIC01
181		CALL SUBT(A,NA,DUMMY(N1),NA,DUMMY(N1),NA)	RIC01
182		CALL TRANSP(DUMMY(N1),NA,DUMMY(N2),NA)	RIC01
183	C		RIC01
184	1000	CONTINUE	RIC01
185		CALL SUM(DUMMY(N2),NA,P,NP,DUMMY(N1),NA,IOPT,SYM,DUMMY(N3))	RIC01
186		IF(IOP(2) .EQ. 0) GO TO 1100	RIC01
187		CALL LNCNT(3)	RIC01
188		PRINT 500,I	RIC01

189		CALL PRNT(P, NP, 4H P , 1)	RIC0
190	C		RIC0
191	1100	CONTINUE	RIC0
192		CALL MULT(P, NP, A, NA, DUMMY(N1), NA)	RIC0
193		CALL MULT(P, NP, B, NB, DUMMY(N2), NB)	RIC0
194		CALL TRANP(B, NB, DUMMY(N3), NF)	RIC0
195		CALL MULT(DUMMY(N3), NF, DUMMY(N1), NA, F, NF)	RIC0
196		CALL MULT(DUMMY(N3), NF, DUMMY(N2), NB, DUMMY(N1), NR)	RIC0
197		CALL TRANP(DUMMY(N1), NR, DUMMY(N2), NR)	RIC0
198		CALL ADD(DUMMY(N1), NR, DUMMY(N2), NR, DUMMY(N1), NR)	RIC0
199		CALL SCALE(DUMMY(N1), NR, DUMMY(N1), NR, 0.5)	RIC0
200		CALL ADD(R, NR, DUMMY(N1), NR, DUMMY(N1), NR)	RIC0
201		CALL SYMPDS(NR(1), NR(1), DUMMY(N1), NA(1), F, IOPT, IOPT, DET, ISCALE, DUM	RIC0
202		IMY(N2), IERR)	RIC0
203		IF(IERR .EQ. 0) GO TO 1150	RIC0
204		CALL LNCNT(3)	RIC0
205		PRINT 225	RIC0
206		RETURN	RIC0
207	C		RIC0
208	1150	CONTINUE	RIC0
209		IF( I .EQ. 1) GO TO 925	RIC0
210		CALL MAXEL(DUMMY, NA, ANORM1)	RIC0
211		CALL SUBT(P, NP, DUMMY, NA, DUMMY(N1), NA)	RIC0
212		CALL MAXEL(DUMMY(N1), NA, ANORM2)	RIC0
213		IF( ANORM1 .GT. 1.) GO TO 1200	RIC0
214		IF( ANORM2/ANORM1 .LT. RICTCV ) GO TO 1300	RIC0
215		GO TO 1250	RIC0
216	1200	CONTINUE	RIC0
217		IF( ANORM2 .LT. RICTCV ) GO TO 1300	RIC0
218	C		RIC0
219	1250	CONTINUE	RIC0
220		IF( I .LE. 101) GO TO 925	RIC0
221		CALL LNCNT(3)	RIC0
222		PRINT 775	RIC0
223		IOP(1) = 1	RIC0
224	C		RIC0
225	1300	CONTINUE	RIC0
226		IF(IOP(1) .EQ. 0 ) RETURN	RIC0
227		CALL LNCNT(4)	RIC0
228		PRINT 1350, I	RIC0
229	1350	FORMAT(//, ' FINAL VALUES OF P AND F AFTER', IS, ' ITERATIONS TO CONV	RIC0
230		ERGE', /)	RIC0
231		CALL PRNT(P, NP, 4H P , 1)	RIC0
232		CALL PRNT(F, NF, 4H F , 1)	RIC0
233	C		RIC0
234		RETURN	RIC0
235		END	RIC0

```

0 SUBROUTINE ASMREG(A,NA,B,NB,H,NH,G,NG,R,NP,F,NF,P,NP,IDENT,DISC,N ASM0
1 IEWT,STABLE,FNULL,ALPHA,IOP,DUMMY) ASM0
2 IMPLICIT REAL*8 (A-H,O-Z) ASM0
3 DIMENSION A(1),B(1),H(1),G(1),R(1),F(1),P(1),DUMMY(1) ASM0
4 DIMENSION NA(2),NB(2),NH(2),NG(2),NR(2),NF(2),NP(2),IOP(5),IOPT(3) ASM0
5 1,NDUM1(2),NDUM2(2),NDUM3(2) ASM0
6 LOGICAL IDENT,DISC,NEWT,STABLE,FNULL,SING ASM0
7 N = NA(1)**2 ASM0
8 N1 = N+1 ASM0
9 IOPTT=0 ASM0
10 IF ( .NOT. NEWT ) GO TO 600 ASM0
11 IF( STABLE ) GO TO 500 ASM0
12 IF ( FNULL ) GO TO 100 ASM0
13 CALL MULT(B,NB,F,NF,DUMMY,NA) ASM0
14 CALL SUBT(A,NA,DUMMY,NA,DUMMY,NA) ASM0
15 CALL TESTSA(DUMMY,NA,ALPHA,DISC,STABLE,IOPTT,DUMMY(N1)) ASM0
16 GO TO 200 ASM0
17 100 CONTINUE ASM0
18 CALL TESTSA(A,NA,ALPHA,DISC,STABLE,IOPTT,DUMMY) ASM0
19 C ASM0
20 200 CONTINUE ASM0
21 IF( STABLE ) GO TO 500 ASM0
22 IF( DISC ) GO TO 230 ASM0
23 J = -NA(1) ASM0
24 NAX = NA(1) ASM0
25 DO 210 I =1,NAX ASM0
26 J = J + NAX + 1 ASM0
27 A(J) = A(J)-ALPHA ASM0
28 210 CONTINUE ASM0
29 SCLE = 3. ASM0
30 IOPT(1)=IOP(1) ASM0
31 IOPT(2) = 1 ASM0
32 IOPT(3)=1 ASM0
33 CALL CSTAB(A,NA,B,NB,F,NF,IOPT,SCLE,DUMMY) ASM0
34 J = -NA(1) ASM0
35 DO 220 I=1,NAX ASM0
36 J = J + NAX + 1 ASM0
37 A(J) = A(J) + ALPHA ASM0
38 220 CONTINUE ASM0
39 225 CONTINUE ASM0
40 CALL MULT(B,NB,F,NF,DUMMY,NA) ASM0
41 CALL SUBT(A,NA,DUMMY,NA,DUMMY,NA) ASM0
42 CALL TESTSA(DUMMY,NA,ALPHA,DISC,STABLE,IOPTT,DUMMY(N1)) ASM0
43 GO TO 300 ASM0
44 C ASM0
45 230 CONTINUE ASM0
46 J = 2*NA(1) + 1 ASM0
47 IF( .NOT. FNULL ) J = J + N ASM0
48 SING = .FALSE. ASM0
49 IF( DUMMY(J) .EQ. 0.0 ) SING = .TRUE. ASM0
50 IOPT(1) = IOP(1) ASM0
51 IOPT(2) = 1 ASM0
52 DSCLE = 0.5 ASM0
53 ALPHAT = 1./ALPHA ASM0
54 CALL SCALE(A,NA,A,NA,ALPHAT) ASM0
55 CALL SCALE(B,NB,B,NR,ALPHAT) ASM0
56 CALL DSTAB(A,NA,B,NB,F,NF,SING,IOPT,DSCLE,DUMMY) ASM0
57 CALL SCALE(A,NA,A,NA,ALPHA) ASM0
58 CALL SCALE(B,NB,B,NB,ALPHA) ASM0
59 GO TO 225 ASM0
60 C ASM0
61 300 CONTINUE ASM0
62 IF( STABLE) GO TO 400 ASM0

```



```

63 CALL LNCNT(5) ASMO
64 IF( DISC ) GO TO 330 ASMO
65 PRINT 310,ALPHA ASMO
66 310 FORMAT(// ' IN ASMREG, CSTAB HAS FAILED TO FIND A STABILIZING GAIN ASMO
67 1 MATRIX (F) RELATIVE TO ',/, ' ALPHA = ',D16.8/) ASMO
68 RETURN ASMO
69 330 CONTINUE ASMO
70 PRINT 340,ALPHA ASMO
71 340 FORMAT(// ' IN ASMREG, DSTAB HAS FAILED TO FIND A STABILIZING GAIN ASMO
72 1 MATRIX (F) RELATIVE TO ',/, ' ALPHA = ',D16.8/) ASMO
73 RETURN ASMO
74 C ASMO
75 400 CONTINUE ASMO
76 FNULL = .FALSE. ASMO
77 C ASMO
78 500 CONTINUE ASMO
79 CALL RICNWT(A,NA,B,NB,H,NH,Q,NQ,R,NR,F,NF,P,NP,IOP,IDENT,DISC,FNU ASMO
80 ILL,DUMMY) ASMO
81 GO TO 750 ASMO
82 C ASMO
83 600 CONTINUE ASMO
84 IF( DISC ) GO TO 700 ASMO
85 NW = 4*N + 1 ASMO
86 NLAM = NW + 4*N ASMO
87 NDUM = NLAM + N ASMO
88 IOP(3) = 1 ASMO
89 CALL CNTREG(A,NA,B,NB,H,NH,Q,NQ,R,NR,DUMMY,DUMMY(NW),DUMMY(NLAM), ASMO
90 IS,F,NF,P,NP,I,IOP,IDENT,DUMMY(NDUM)) ASMO
91 GO TO 750 ASMO
92 700 CONTINUE ASMO
93 CALL DISREG(A,NA,B,NB,H,NH,Q,NQ,R,NR,F,NF,P,NP,IOP,IDENT,DUMMY) ASMO
94 C ASMO
95 750 CONTINUE ASMO
96 C ASMO
97 IF( IOP(4) .EQ. 0 ) GO TO 1100 ASMO
98 C ASMO
99 N2= N1 + N ASMO
100 N3= N2 + N ASMO
101 C ASMO
102 IF( DISC ) GO TO 800 ASMO
103 CALL MULT(P,NP,B,NB,DUMMY,NB) ASMO
104 CALL MULT(DUMMY,NB,F,NF,DUMMY(N1),NP) ASMO
105 CALL TRANSP(DUMMY(N1),NP,DUMMY,NP) ASMO
106 CALL ADD(DUMMY,NP,DUMMY(N1),NP,DUMMY,NP) ASMO
107 CALL SCALE(DUMMY,NP,DUMMY,NP,0.5) ASMO
108 CALL SUBT(Q,NQ,DUMMY,NP,DUMMY,NP) ASMO
109 CALL MULT(P,NP,A,NA,DUMMY(N1),NP) ASMO
110 CALL ADD(DUMMY,NP,DUMMY(N1),NP,DUMMY,NP) ASMO
111 CALL TRANSP(DUMMY(N1),NP,DUMMY(N2),NP) ASMO
112 CALL ADD(DUMMY,NP,DUMMY(N2),NP,DUMMY,NP) ASMO
113 GO TO 900 ASMO
114 C ASMO
115 800 CONTINUE ASMO
116 CALL MULT(R,NR,F,NF,DUMMY,NF) ASMO
117 CALL TRANSP(F,NF,DUMMY(N1),NB) ASMO
118 CALL MULT(DUMMY(N1),NB,DUMMY,NF,DUMMY(N2),NA) ASMO
119 CALL ADD(DUMMY(N2),NA,Q,NQ,DUMMY,NA) ASMO
120 CALL MULT(B,NB,F,NF,DUMMY(N1),NA) ASMO
121 CALL SUBT(A,NA,DUMMY(N1),NA,DUMMY(N1),NA) ASMO
122 CALL MULT(P,NP,DUMMY(N1),NA,DUMMY(N2),NA) ASMO
123 CALL TRANSP(DUMMY(N1),NA,DUMMY(N3),NA) ASMO
124 CALL MULT(DUMMY(N3),NA,DUMMY(N2),NA,DUMMY(N1),NA) ASMO
125 CALL ADD(DUMMY,NA,DUMMY(N1),NA,DUMMY,NA) ASMO

```

26	CALL SUBT(P, NP, DUMMY, NA, DUMMY, NA)	ASM01
27	C	ASM01
128	900 CONTINUE	ASM01
29	CALL LNCNT(4)	ASM01
30	PRINT 1000	ASM01
131	1000 FORMAT(// ' RESIDUAL ERROR IN RICCATI EQUATION ' //)	ASM01
132	CALL PRNT(DUMMY, NP, 4HEROR, 1)	ASM01
33	C	ASM01
34	1100 CONTINUE	ASM01
135	N2 = N1 + NA(1)	ASM01
36	N3 = N2 + NA(1)	ASM01
37	ISV = 0	ASM01
138	CALL EQUATE(P, NP, DUMMY, NP)	ASM01
139	CALL EIGEN(NA(1), NA(1), DUMMY, DUMMY(N1), DUMMY(N2), ISV, ISV, V, DUMMY(NAS	ASM01
140	13), IERR)	ASM01
141	NEVL = NA(1)	ASM01
142	IF( IERR .EQ. 0) GO TO 1300	ASM01
143	NEVL = NA(1) - IERR	ASM01
144	CALL LNCNT(4)	ASM01
145	PRINT 1200, IERR	ASM01
146	1200 FORMAT(// ' IN ASMREG, THE ' //, ' IS, ' EIGENVALUE OF P HAS NOT BEEN COAS	ASM01
147	1PUTED AFTER 30 ITERATIONS ' //)	ASM01
148	C	ASM01
149	1300 CONTINUE	ASM01
150	NDUM1(1) = NEVL	ASM01
151	NDUM1(2) = 1	ASM01
152	CALL EQUATE(DUMMY(N1), NDUM1, DUMMY, NDUM1)	ASM01
153	N1 = NDUM1(1) + 1	ASM01
154	CALL MULT(B, NB, F, NF, DUMMY(N1), NA)	ASM01
155	CALL SUBT(A, NA, DUMMY(N1), NA, DUMMY(N1), NA)	ASM01
156	N2 = N1 + N	ASM01
157	CALL EQUATE(DUMMY(N1), NA, DUMMY(N2), NA)	ASM01
158	N3 = N2 + N	ASM01
159	N4 = N3 + NA(1)	ASM01
160	N5 = N4 + NA(1)	ASM01
161	CALL EIGEN(NA(1), NA(1), DUMMY(N2), DUMMY(N3), DUMMY(N4), ISV, ISV, V, DUM	ASM01
162	1MY(N5), IERR)	ASM01
163	NEVL = NA(1)	ASM01
164	IF( IERR .EQ. 0 ) GO TO 1500	ASM01
165	NEVL = NA(1) - IERR	ASM01
166	CALL LNCNT(4)	ASM01
167	PRINT 1400, IERR	ASM01
168	1400 FORMAT(// ' IN ASMREG, THE ' //, ' IS, ' EIGENVALUE OF A-BF HAS NOT BEEN C	ASM01
169	1OMPUTED AFTER 30 ITERATIONS ' //)	ASM01
170	C	ASM01
171	1500 CONTINUE	ASM01
172	NDUM2(1) = NEVL	ASM01
173	NDUM2(2) = 1	ASM01
174	CALL JUXTC(DUMMY(N3), NDUM2, DUMMY(N4), NDUM2, DUMMY(N2), NDUM3)	ASM01
175	C	ASM01
176	IF ( IOP(5) .EQ. 0 ) RETURN	ASM01
177	C	ASM01
178	CALL LNCNT(4)	ASM01
179	PRINT 1600	ASM01
180	1600 FORMAT(// ' EIGENVALUES OF P ' //)	ASM01
181	CALL PRNT(DUMMY, NDUM1, 4HEVLP, 1)	ASM01
182	CALL LNCNT(4)	ASM01
183	PRINT 1700	ASM01
184	1700 FORMAT(// ' CLOSED-LOOP RESPONSE MATRIX A-BF ' //)	ASM01
185	CALL PRNT(DUMMY(N1), NA, 4HA-BF, 1)	ASM01
186	CALL LNCNT(3)	ASM01
187	PRINT 1800	ASM01
188	1800 FORMAT(// ' EIGENVALUES OF A-BF ' //)	ASM01

189 CALL PRNT('DUMMY(N2),NOUM3,0,3)  
190 C  
191 RETURN  
192 END

ASMO  
ASMO  
ASMO  
ASMO

```

0 SUBROUTINE ASMFIL(A,NA,G,NG,H,NH,Q,NQ,R,NR,F,NF,P,NP,IDENT,DISC,N AS
1 NEWT,STABLE,FNULL,ALPHA,IOP,DUMMY) AS
2 IMPLICIT REAL*8 (A-H,O-Z) AS
3 DIMENSION A(1),G(1),H(1),Q(1),R(1),F(1),P(1),DUMMY(1) AS
4 DIMENSION NA(2),NG(2),NH(2),NQ(2),NR(2),NF(2),NP(2),IOPT(5),NDUM1(AS
5 12),IOP(1) AS
6 LOGICAL IDENT,DISC,NEWT,STABLE,FNULL AS
7 IF( IOP(1) .EQ. 0 ) GO TO 100 AS
8 CALL LNCNT(4) AS
9 IF(DISC) PRINT 15 AS
10 IF( .NOT. DISC ) PRINT 25 AS
11 15 FORMAT(/,,' PROGRAM TO SOLVE THE DISCRETE INFINITE-DURATION OPTIMAAS
12 IL FILTER PROBLEM',/) AS
13 25 FORMAT(/,,' PROGRAM TO SOLVE THE CONTINUOUS INFINITE-DURATION OPTIA
14 MAL FILTER PROBLEM',/) AS
15 CALL PRNT(A,NA,4H A ,1) AS
16 IF( .NOT. IDENT ) GO TO 35 AS
17 CALL LNCNT(3) AS
18 PRINT 30 AS
19 30 FORMAT(/,,' G IS AN IDENTITY MATRIX',/) AS
20 GO TO 40 AS
21 35 CONTINUE AS
22 CALL PRNT(G,NG,4H G ,1) AS
23 40 CONTINUE AS
24 CALL PRNT(H,NH,4H H ,1) AS
25 CALL LNCNT(3) AS
26 PRINT 45 AS
27 45 FORMAT(/,,' INTENSITY MATRIX FOR COVARIANCE OF MEASUREMENT NOISE',/) AS
28 CALL PRNT(R,NR,4H R ,1) AS
29 C AS
30 IF( .NOT. IDENT ) GO TO 65 AS
31 CALL LNCNT(3) AS
32 PRINT 55 AS
33 55 FORMAT(/,,' INTENSITY MATRIX FOR COVARIANCE OF PROCESS NOISE',/) AS
34 C AS
35 65 CONTINUE AS
36 CALL PRNT(Q,NQ,4H Q ,1) AS
37 C AS
38 100 CONTINUE AS
39 IOPT(1)=IOP(2) AS
40 IOPT(2)=IOP(3) AS
41 IOPT(3)=IOP(4) AS
42 IOPT(4)=IOP(5) AS
43 IOPT(5)=0 AS
44 K = 0 AS
45 C AS
46 200 CONTINUE AS
47 CALL TRANP(A,NA,DUMMY,NA) AS
48 CALL EQUATE(DUMMY,NA,A,NA) AS
49 CALL TRANP(H,NH,DUMMY,NDUM1) AS
50 CALL EQUATE(DUMMY,NDUM1,H,NH) AS
51 IF( IDENT ) GO TO 250 AS
52 CALL TRANP(G,NG,DUMMY,NDUM1) AS
53 CALL EQUATE(DUMMY,NDUM1,G,NG) AS
54 250 CONTINUE AS
55 IF ( K .EQ. 1 ) RETURN AS
56 C AS
57 K = K+1 AS
58 CALL ASMREG(A,NA,H,NH,G,NG,Q,NQ,R,NR,F,NF,P,NP,IDENT,DISC,NEWT,ST
59 IABLE,FNULL,ALPHA,IOPT,DUMMY) AS
60 C AS
61 N1=(NA(1)**2)+3*NA(1)+1 AS
62 CALL TRANP(F,NF,DUMMY(N1),NDUM1) AS

```

ORIGINAL PAGE IS  
OF POOR QUALITY

63		CALL EQUATE(DUMMY(N1),NDUM1,F,NF)	ASMO
64	C		ASMO
65		IF( IOP(1) .EQ. 0 ) GO TO 200	ASMO
66	C		ASMO
67		IF(IDENT) GO TO 300	ASMO
68		CALL LNCNT(3)	ASMO
69		PRINT 55	ASMO
70		CALL PRNT(Q,NO,4MGQGT,1)	ASMO
71	C		ASMO
72	300	CONTINUE	ASMO
73		CALL LNCNT(3)	ASMO
74		PRINT 325	ASMO
75	325	FORMAT(/, ' FILTER GAIN',/)	ASMO
76		CALL PRNT(F,NF,4MF,1)	ASMO
77		CALL LNCNT(3)	ASMO
78		PRINT 350	ASMO
79	350	FORMAT(/, 'STEADY-STATE VARIANCE MATRIX OF RECONSTRUCTION ERROR',/)	ASMO
80		CALL PRNT(P,NP,4MP,1)	ASMO
81		NDUM1(1)=NP(1)	ASMO
82		NDUM1(2)=1	ASMO
83		CALL LNCNT(3)	ASMO
84		PRINT 375	ASMO
85	375	FORMAT(/, ' EIGENVALUES OF P ',/)	ASMO
86		CALL PRNT(DUMMY,NDUM1,4HEVLP,1)	ASMO
87		N1 = NP(1) + 1	ASMO
88		N = NA(1)**2	ASMO
89		N2 = N1 + N + 2*NA(1)	ASMO
90		CALL TRAMP(DUMMY(N1),NA,DUMMY(N2),NA)	ASMO
91		CALL PRNT(DUMMY(N2),NA,4HA-FH,1)	ASMO
92		N2 = N1 + N	ASMO
93		CALL LNCNT(3)	ASMO
94		PRINT 385	ASMO
95	385	FORMAT(/, ' EIGENVALUES OF A-FH MATRIX',/)	ASMO
96		NDUM1(1) = NA(1)	ASMO
97		NDUM1(2) = 2	ASMO
98		CALL PRNT(DUMMY(N2),NDUM1,0,3)	ASMO
99	C		ASMO
100		GO TO 200	ASMO
101	C		ASMO
102		END	ASMO

```

0 SUBROUTINE EXPMOD (A,NA,B,NB,H,NH,AM,NAM,HM,NHM,Q,NQ,R,NR,F,NF,P, EXP0
1 INP,HIDENT,HMDENT,DISC,NEWT,STABLE,FNULL,ALPHA,IOP,DUMMY) EXP0
2 IMPLICIT REAL*8 (A-H,O-Z) EXP0
3 DIMENSION A(1),B(1),H(1),AM(1),HM(1),Q(1),R(1),F(1),P(1),DUMMY(1) EXP0
4 DIMENSION NA(2),NB(2),NH(2),NAM(2),NHM(2),NQ(2),NR(2),NF(2),NP(2),EXP0
5 IOP(1),IOPT(5),NDUM1(2),NDUM2(2),NDUM3(2) EXP0
6 LOGICAL HIDENT,HMDENT,DISC,NEWT,STABLE,FNULL,SYM EXP0
7 COMMON/TOL/EPHAM,EPBAM,IACM EXP0
8 IF( IOP(1) .EQ. 0 ) GO TO 300 EXP0
9 CALL LNCNT(6) EXP0
10 IF( DISC ) PRINT 25 EXP0
11 IF( .NOT. DISC ) PRINT 50 EXP0
12 25 FORMAT(/,' PROGRAM TO SOLVE ASYMPTOTIC DISCRETE EXPLICIT MODEL-FOLEXP0
13 ILOWING PROBLEM',//,' PLANT DYNAMICS',/) EXP0
14 50 FORMAT(/,' PROGRAM TO SOLVE ASYMPTOTIC CONTINUOUS EXPLICIT MODEL-FEXP0
15 ILOWING PROBLEM',//,' PLANT DYNAMICS',/) EXP0
16 CALL PRNT(A,NA,4H A ,1) EXP0
17 CALL PRNT(B,NB,4H B ,1) EXP0
18 IF( HIDENT ) GO TO 75 EXP0
19 CALL PRNT(H,NH,4H H ,1) EXP0
20 GO TO 100 EXP0
21 75 CONTINUE EXP0
22 CALL LNCNT(3) EXP0
23 PRINT 85 EXP0
24 85 FORMAT(/,' H IS AN IDENTITY MATRIX',/) EXP0
25 C EXP0
26 100 CONTINUE EXP0
27 CALL LNCNT(4) EXP0
28 PRINT 125 EXP0
29 125 FORMAT(/,' MODEL DYNAMICS',/) EXP0
30 CALL PRNT(AM,NAM,4H AM ,1) EXP0
31 IF( HMDENT ) GO TO 175 EXP0
32 CALL PRNT(HM,NHM,4H HM ,1) EXP0
33 GO TO 200 EXP0
34 175 CONTINUE EXP0
35 CALL LNCNT(3) EXP0
36 PRINT 185 EXP0
37 185 FORMAT(/,' HM IS AN IDENTITY MATRIX ',/) EXP0
38 C EXP0
39 200 CONTINUE EXP0
40 CALL LNCNT(4) EXP0
41 PRINT 225 EXP0
42 225 FORMAT(/,' WEIGHTING MATRICES ',/) EXP0
43 CALL PRNT(Q,NQ,4H Q ,1) EXP0
44 CALL PRNT(R,NR,4H R ,1) EXP0
45 C EXP0
46 300 CONTINUE EXP0
47 IF( IOP(2) .EQ. 0 ) GO TO 400 EXP0
48 NF(1) = NB(2) EXP0
49 NF(2) = NA(1) EXP0
50 NP(1) = NA(1) EXP0
51 NP(2) = NA(1) EXP0
52 IOPT(1) = IOP(3) EXP0
53 IOPT(2) = IOP(4) EXP0
54 IOPT(3) = IOP(5) EXP0
55 IOPT(4) = 0 EXP0
56 IOPT(5) = 0 EXP0
57 N1 = NA(1)*NA(2) + 1 EXP0
58 CALL EQUATE(Q,NQ,DUMMY,NQ) EXP0
59 CALL ASMREG(A,NA,B,NB,H,NH,DUMMY,NQ,R,NR,F,NF,P,NP,HIDENT,DISC,NE EXP0
60 IWT,STABLE,FNULL,ALPHA,IOPT,DUMMY(N1)) EXP0
61 C EXP0
62 400 CONTINUE EXP0

```

```

63 IF( IOP(1) .EQ. 0 ) GO TO 600 EXP00
64 CALL LNCNT(4) EXP00
65 PRINT 425 EXP00
66 425 FORMAT(/, ' CONTROL LAW U = -P( COL.(X,XM) ), F = (F11,F12)',/) EXP00
67 CALL LNCNT(3) EXP00
68 PRINT 450 EXP00
69 450 FORMAT(/, ' PART OF F MULTIPLYING X ',/) EXP00
70 CALL PRNT(F,NF,4H F11,1) EXP00
71 IF( .NOT. DISC .AND. IOP(2) .EQ. 0 ) GO TO 600 EXP00
72 CALL PRNT(P,NP,4H P11,1) EXP00
73 IF( IOP(2) .EQ. 0 ) GO TO 600 EXP00
74 CALL LNCNT(2) EXP00
75 PRINT 475 EXP00
76 475 FORMAT(/, ' EIGENVALUES OF P11' ) EXP00
77 NDUM1(1) = NA(1) EXP00
78 NDUM1(2) = 1 EXP00
79 CALL PRNT(DUMMY(N1),NDUM1,0,3) EXP00
80 N1 = N1 + NDUM1(1) EXP00
81 NDUM1(2) = NA(1) EXP00
82 CALL LNCNT(2) EXP00
83 PRINT 500 EXP00
84 500 FORMAT(/, ' PLANT CLOSED-LOOP RESPONSE MATRIX A = BF11' ) EXP00
85 CALL PRNT(DUMMY(N1),NDUM1,0,3) EXP00
86 CALL LNCNT(2) EXP00
87 PRINT 525 EXP00
88 525 FORMAT(/, ' EIGENVALUES OF CLOSED-LOOP RESPONSE MATRIX' ) EXP00
89 N1 = N1 + NDUM1(1)*NDUM1(2) EXP00
90 NDUM1(2) = 2 EXP00
91 CALL PRNT(DUMMY(N1),NDUM1,0,3) EXP00
92 C EXP00
93 600 CONTINUE EXP00
94 NF(1) = NB(2) EXP00
95 NF(2) = NA(1) EXP00
96 CALL MULT(B,NB,F,NF,DUMMY,NA) EXP00
97 CALL SUBT(A,NA,DUMMY,NA,DUMMY,NA) EXP00
98 IF( IOP(1).EQ. 0 .OR. IOP(2) .NE. 0 ) GO TO 700 EXP00
99 CALL LNCNT(2) EXP00
100 PRINT 500 EXP00
101 CALL PRNT(DUMMY,NA,0,3) EXP00
102 C EXP00
103 700 CONTINUE EXP00
104 N1 = NA(1)**2 + 1 EXP00
105 CALL TRANP(DUMMY,NA,DUMMY(N1),NA) EXP00
106 CALL EQUATE(DUMMY(N1),NA,DUMMY,NA) EXP00
107 NF(2) = NA(1) + NAM(1) EXP00
108 NP(2) = NF(2) EXP00
109 IF( .NOT. DISC .AND. IOP(2).EQ. 0 ) NP(2) = NAM(2) EXP00
110 IOPTT=0 EXP00
111 SYM = .FALSE. EXP00
112 CALL EQUATE( Q,NQ,DUMMY(N1),NDUM2) EXP00
113 IF( HMDENT ) GO TO 725 EXP00
114 CALL MULT(Q,NQ,HM,NHM,DUMMY(N1),NDUM2) EXP00
115 725 CONTINUE EXP00
116 IF( HIDENT ) GO TO 750 EXP00
117 N2 = N1 + NQ(1)*NHM(2) EXP00
118 CALL TRANP(H,NH,DUMMY(N2),NDUM1) EXP00
119 N3 = N2 + NH(1)*NH(2) EXP00
120 CALL MULT(DUMMY(N2),NDUM1,DUMMY(N1),NHM,DUMMY(N3),NDUM2) EXP00
121 CALL EQUATE(DUMMY(N3),NDUM2,DUMMY(N1),NDUM2) EXP00
122 750 CONTINUE EXP00
123 N2 = NA(1)**2 + NA(1)*NHM(2) + 1 EXP00
124 N3 = NA(1)**2 + 1 EXP00
125 IF( .NOT. DISC .AND. IOP(2) .EQ. 0 ) N3 = 1 EXP00

```

126	CALL EQUATE(DUMMY(N1),NDUM2,P(N3),NDUM2)	EXP0
127	IF( DISC ) GO TO 800	EXP0
128	EPSA = EPSAM	EXP0
129	CALL HARSTW(DUMMY,NA,AM,NAM,P(N3),NDUM2,IOPTT,SYM ,EPSA,EPSA,DUMMY	EXP0
130	1(N2))	EXP0
131	GO TO 900	EXP0
132	C	EXP0
133	800 CONTINUE	EXP0
134	CALL SCALE(P(N3),NDUM2,P(N3),NDUM2,-1.0)	EXP0
135	N4 = N2 +NAM(1)**2	EXP0
136	CALL EQUATE(AM,NAM,DUMMY(N2),NAM)	EXP0
137	CALL SUM(DUMMY,NA,P(N3),NDUM2,DUMMY(N2),NAM,IOPTT,SYM,DUMMY(N4))	EXP0
138	C	EXP0
139	900 CONTINUE	EXP0
140	N2 = NB(2)*NA(1) + 1	EXP0
141	CALL TRANP(B,NB,DUMMY,NDUM1)	EXP0
142	CALL MULT(DUMMY,NDUM1,P(N3),NDUM2,F(N2),NDUM3)	EXP0
143	IF( .NOT. DISC ) GO TO 1000	EXP0
144	N1 = NB(1)*NB(2) + 1	EXP0
145	CALL MULT(DUMMY,NDUM1,P,NA,DUMMY(N1),NDUM2)	EXP0
146	CALL MULT(DUMMY(N1),NDUM2,B,NB,DUMMY,NR)	EXP0
147	CALL ADD(R,NR,DUMMY,NR,DUMMY,NR)	EXP0
148	GO TO 1100	EXP0
149	C	EXP0
150	1000 CONTINUE	EXP0
151	CALL EQUATE(R,NR,DUMMY,NR)	EXP0
152	C	EXP0
153	1100 CONTINUE	EXP0
154	N1 = NR(1)**2 + 1	EXP0
155	CALL SYMPOS(NR(1),NR(1),DUMMY,NHM(2),F(N2),IOPTT,IOPTT,DETERM,ISCA	EXP0
156	1LE,DUMMY(N1),IERR)	EXP0
157	IF( IERR .EQ. 0 ) GO TO 1200	EXP0
158	CALL LNCNT(3)	EXP0
159	PRINT 1150	EXP0
160	1150 FORMAT(/, ' IN EXPDMF, THE COEFFICIENT MATRIX FOR SYMPOS IS NOT SY	EXP0
161	MMETRIC POSITIVE DEFINITE ',/)	EXP0
162	RETURN	EXP0
163	C	EXP0
164	1200 CONTINUE	EXP0
165	IF( .NOT. DISC ) GO TO 1300	EXP0
166	CALL MULT(F(N2),NDUM3,AM,NAM,DUMMY,NDUM1)	EXP0
167	CALL EQUATE(DUMMY,NDUM1,F(N2),NDUM1)	EXP0
168	1300 CONTINUE	EXP0
169	IF( IOP(1) .EQ. 0 ) RETURN	EXP0
170	CALL LNCNT(3)	EXP0
171	PRINT 1325	EXP0
172	1325 FORMAT(/, ' PART OF F MULTIPLYING XM ',/)	EXP0
173	CALL PRNT(F(N2),NDUM3,4H F12,1)	EXP0
174	NDUM1(1) = NA(1)	EXP0
175	NDUM1(2) = NAM(1)	EXP0
176	CALL PRNT(P(N3),NDUM1,4H P12,1)	EXP0
177	RETURN	EXP0
178	END	EXP0

OF EQUALITY



```

0      SUBROUTINE IMPMOD(A,NA,B,NB,H,NH,AM,NAM,BM,NBM,Q,NQ,R,NR,F,NF,P,N IMP0
1      IP,IDENT,DISC,NEWT,STABLE,FNULL,ALPHA,IOP,DUMMY) IMP0
2      IMPLICIT REAL*8 (A-H,O-Z) IMP0
3      DIMENSION A(1),B(1),H(1),AM(1),BM(1),Q(1),R(1),F(1),P(1),DUMMY(1) IMP0
4      DIMENSION NA(2),NB(2),NH(2),NAM(2),NBM(2),NQ(2),NR(2),NF(2),NP(2), IMP0
5      IOP(1),IOPT(5),NDUM1(2) IMP0
6      LOGICAL IDENT,DISC,NEWT,STABLE,FNULL,HIDENT IMP0
7      IF( IOP(1) .EQ. 0 ) GO TO 200 IMP0
8      CALL LNCNT(6) IMP0
9      IF( DISC ) PRINT 25 IMP0
10     IF( .NOT. DISC ) PRINT 50 IMP0
11     25 FORMAT(/,'PROGRAM TO SOLVE ASYMPTOTIC DISCRETE IMPLICIT MODEL-FOLL IMP0
12     10WING PROBLEM',//,' PLANT DYNAMICS ',/) IMP0
13     50 FORMAT(/,' PROGRAM TO SOLVE ASYMPTOTIC CONTINUOUS IMPLICIT MODEL-FIMP0
14     10LLOWING PROBLEM',//,' PLANT DYNAMICS',/) IMP0
15     CALL PRNT(A,NA,4H A ,1) IMP0
16     CALL PRNT(B,NB,4H B ,1) IMP0
17     IF( IDENT ) GO TO 75 IMP0
18     CALL PRNT(H,NH,4H H ,1) IMP0
19     GO TO 100 IMP0
20     75 CONTINUE IMP0
21     CALL LNCNT(3) IMP0
22     PRINT 55 IMP0
23     85 FORMAT(/,' H IS AN IDENTITY MATRIX',/) IMP0
24 C IMP0
25     100 CONTINUE IMP0
26     CALL LNCNT(4) IMP0
27     PRINT 125 IMP0
28     125 FORMAT(//,' MODEL DYNAMICS',/) IMP0
29     CALL PRNT(AM,NAM,4H AM ,1) IMP0
30     CALL PRNT(BM,NBM,4H BM ,1) IMP0
31     CALL LNCNT(4) IMP0
32     PRINT 150 IMP0
33     150 FORMAT(//,' WEIGHTING MATRICES',/) IMP0
34     CALL PRNT(Q,NQ,4H Q ,1) IMP0
35     CALL PRNT(R,NR,4H R ,1) IMP0
36 C IMP0
37     200 CONTINUE IMP0
38     N = NA(1)**2 IMP0
39     N1 = N + 1 IMP0
40     IF( .NOT. IDENT ) GO TO 300 IMP0
41     CALL SUBT(A,NA,AM,NAM,DUMMY,NA) IMP0
42     CALL SUBT(B,NB,AM,NBM,DUMMY(N1),NB) IMP0
43     GO TO 400 IMP0
44 C IMP0
45     300 CONTINUE IMP0
46     CALL MULT(H,NH,A,NA,DUMMY,NH) IMP0
47     CALL MULT(AM,NAM,H,NH,DUMMY(N1),NH) IMP0
48     CALL SUBT(DUMMY,NH,DUMMY(N1),NH,DUMMY,NH) IMP0
49     CALL MULT(H,NH,B,NB,DUMMY(N1),NBM) IMP0
50     CALL SUBT(DUMMY(N1),NBM,AM,NBM,DUMMY(N1),NBM) IMP0
51 C IMP0
52     400 CONTINUE IMP0
53     IF( IOP(1) .EQ. 0 ) GO TO 500 IMP0
54     CALL LNCNT(3) IMP0
55     PRINT 450 IMP0
56     450 FORMAT(//,' MATRIX HA = AMH') IMP0
57     CALL PRNT(DUMMY,NH,0,3) IMP0
58     CALL LNCNT(3) IMP0
59     PRINT 475 IMP0
60     475 FORMAT(//,' MATRIX HB = BM') IMP0
61     CALL PRNT(DUMMY(N1),NBM,0,3) IMP0
62 C IMP0

```

63	500	CONTINUE	IMP
64		N2 = N1 + N	IMP
65		N3 = N2 + N	IMP
66		N4 = N3 + N	IMP
67		CALL MULT(Q,NQ,DUMMY,NH,DUMMY(N2),NH)	IMP
68		CALL MULT(Q,NQ,DUMMY(N1),NBM,DUMMY(N3),NBM)	IMP
69		CALL TRANP(DUMMY,NH,DUMMY(N4),NDUM1)	IMP
70		CALL MULT(DUMMY(N4),NDUM1,DUMMY(N2),NH,DUMMY,NA)	IMP
71		CALL MULT(DUMMY(N4),NDUM1,DUMMY(N3),NBM,DUMMY(N2),NB)	IMP
72		CALL TRANP(DUMMY(N1),NBM,DUMMY(N4),NDUM1)	IMP
73		CALL SCALE(DUMMY(N2),NB,DUMMY(N1),NB,2.0)	IMP
74		CALL MULT(DUMMY(N4),NDUM1,DUMMY(N3),NBM,DUMMY(N2),NR)	IMP
75		CALL ADD(DUMMY(N2),NR,R,NR,DUMMY(N2),NR)	IMP
76		IF( IOP(1) .EQ. 0 ) GO TO 600	IMP
77		CALL LNCNT(3)	IMP
78		PRINT 525	IMP
79	525	FORMAT(//, ' MATRIX (HA - AMH TRANSPOSE)Q( HA - AMH)'	IMP
80		CALL PRNT(DUMMY,NA,0,3)	IMP
81		CALL LNCNT(3)	IMP
82		PRINT 550	IMP
83	550	FORMAT(//, ' MATRIX 2( HA - AMH TRANSPOSE)Q( HB - BM)'	IMP
84		CALL PRNT(DUMMY(N1),NB,0,3)	IMP
85		CALL LNCNT(3)	IMP
86		PRINT 575	IMP
87	575	FORMAT(//, ' MATRIX ( HB - BM TRANSPOSE)Q( HB - BM ) + R'	IMP
88		CALL PRNT(DUMMY(N2),NR,0,3)	IMP
89	C		IMP
90	600	CONTINUE	IMP
91		IOPT(1)= 0	IMP
92		IOPT(2)= 1	IMP
93		IOPT(3)= 1	IMP
94		N5 = N4 + N	IMP
95		CALL EQUATE(A,NA,DUMMY(N3),NA)	IMP
96		CALL PREFIL(DUMMY(N3),NA,B,NB,DUMMY,NA,DUMMY(N1),NB,DUMMY(N2),NR,D	IMP
97		DUMMY(N4),NF,IOPT,DUMMY(N5))	IMP
98		IF( IOP(1) .EQ. 0 ) GO TO 700	IMP
99		CALL LNCNT(3)	IMP
100		PRINT 625	IMP
101	625	FORMAT(//, ' PREFILTER GAIN'	IMP
102		CALL PRNT(DUMMY(N4),NF,0,3)	IMP
103		CALL LNCNT(3)	IMP
104		PRINT 650	IMP
105	650	FORMAT(//, ' MATRIX A - B(PREFILTER)'	IMP
106		CALL PRNT(DUMMY(N3),NA,0,3)	IMP
107		CALL LNCNT(3)	IMP
108		PRINT 675	IMP
109	675	FORMAT(//, ' MODIFIED STATE VECTOR WEIGHTING MATRIX'	IMP
110		CALL PRNT(DUMMY,NA,0,3)	IMP
111	C		IMP
112	700	CONTINUE	IMP
113		CALL EQUATE(DUMMY(N4),NF,DUMMY(N1),NF)	IMP
114	C		IMP
115		IF( IOP(2) .EQ. -1000 ) RETURN	IMP
116	C		IMP
117		IOPT(1) = IOP(2)	IMP
118		IOPT(2) = IOP(3)	IMP
119		IOPT(3) = IOP(4)	IMP
120		IOPT(4) = 0	IMP
121		IOPT(5) = 0	IMP
122		HIDENT = .TRUE.	IMP
123		CALL ASMREG(DUMMY(N3),NA,B,NB,H,NH,DUMMY,NA,DUMMY(N2),NR,F,NF,P,N	IMP
124		IP,HIDENT,DISC,NEAT,STABLE,FNULL,ALPHA,IOPT,DUMMY(N4))	IMP
125		IF( IOP(1) .EQ. 0 ) GO TO 800	IMP

126	CALL LNCNT(3)	IMPO
127	PRINT 725	IMPO
129	725 FORMAT(/, ' GAIN FROM ASMREG')	IMPO
129	CALL PRNT(F,NF,0,3)	IMPO
130	CALL LNCNT(3)	IMPO
131	PRINT 750	IMPO
132	750 FORMAT(/, ' SOLUTION OF ASSOCIATED STEADY-STATE RICCATI EQUATION')	IMPO
133	CALL PRNT(P,NP,0,3)	IMPO
134	CALL LNCNT(3)	IMPO
135	PRINT 775	IMPO
136	775 FORMAT(/, ' EIGENVALUES OF P')	IMPO
137	NDUM1(1) = NA(1)	IMPO
138	NDUM1(2) = 1	IMPO
139	CALL PRNT(DUMMY(N4),NDUM1,0,3)	IMPO
140	C	IMPO
141	800 CONTINUE	IMPO
142	CALL ADD(F,NF,DUMMY(N1),NF,F,NF)	IMPO
143	IF( IOP(1) .EQ. 0 ) RETURN	IMPO
144	CALL LNCNT(4)	IMPO
145	PRINT 825	IMPO
146	825 FORMAT(/, ' GAIN FOR MODEL-FOLLOWING CONTROL LAW, $U = - F X$ , $F =$	IMPO
147	L(PREFILTER) + (ASMREG)',/)	IMPO
148	CALL PRNT(F,NF,4H F ,1)	IMPO
149	N6 = N4 + NA(1)	IMPO
150	CALL PRNT(DUMMY(N6),NA,4HA-8F,1)	IMPO
151	NDUM1(2) = 2	IMPO
152	N6 = N6 + N	IMPO
153	CALL LNCNT(3)	IMPO
154	PRINT 850	IMPO
155	850 FORMAT(/, ' EIGENVALUES OF A-8F')	IMPO
156	CALL PRNT(DUMMY(N6),NDUM1,0,3)	IMPO
157	C	IMPO
158	RETURN	IMPO
159	END	IMPO

0	SUBROUTINE READ1 (A,NA,NZ,NAM)	REA
1	IMPLICIT REAL*8 (A-H,O-Z)	REA
2	DIMENSION A(1),NA(2),NZ(2)	REA
3	IF (NZ(1).EQ.0) GO TO 410	REA
4	NR=NZ(1)	REA
5	NC=NZ(2)	REA
6	NLST=NR*NC	REA
7	IF( NLST .LT. 1 .OR. NR .LT. 1 ) GO TO 16	REA
8	DO 400 I = 1, NR	REA
9	400 READ (5,101) (A( J), J = I,NLST,NR)	REA
10	NA(1)=NR	REA
11	NA(2)=NC	REA
12	410 CALL PRNT (A,NA,NAM,1)	REA
13	101 FORMAT(8D10.2)	REA
14	RETURN	REA
15	16 CALL LNCNT(1)	REA
16	WRITE (6,916) NAM,NR,NC	REA
17	916 FORMAT (' ERROR IN READ1 MATRIX ',A4,' HAS NA=',2I6)	REA
18	RETURN	REA
19	END	REA

```

0 SUBROUTINE BALANC(NM,N,A,LOW,IGH,SCALE) BAL00
1 IMPLICIT REAL*8 (A-H,O-Z) BAL00
2 INTEGER I,J,K,L,M,N,JJ,NM,IGH,LOW,IEXC BAL00
3 DIMENSION A(NM,N),SCALE(N) BAL00
4 C REAL C,F,G,R,S,B2,RADIX BAL00
5 C REAL DABS BAL00
6 LOGICAL NOCONV BAL00
7 C BAL00
8 C BAL00
9 C ***** RADIX IS A MACHINE DEPENDENT PARAMETER SPECIFYING BAL00
10 C THE BASE OF THE MACHINE FLOATING POINT REPRESENTATION. BAL00
11 C BAL00
12 C BAL00
13 RADIX = 16. BAL00
14 C BAL00
15 B2 = RADIX * RADIX BAL00
16 K = 1 BAL00
17 L = N BAL00
18 GO TO 100 BAL00
19 C ***** IN-LINE PROCEDURE FOR ROW AND BAL00
20 C COLUMN EXCHANGE ***** BAL00
21 20 SCALE(M) = J BAL00
22 IF (J .EQ. M) GO TO 50 BAL00
23 C BAL00
24 DO 30 I = 1, L BAL00
25 F = A(I,J) BAL00
26 A(I,J) = A(I,M) BAL00
27 A(I,M) = F BAL00
28 30 CONTINUE BAL00
29 C BAL00
30 DO 40 I = K, N BAL00
31 F = A(J,I) BAL00
32 A(J,I) = A(M,I) BAL00
33 A(M,I) = F BAL00
34 40 CONTINUE BAL00
35 C BAL00
36 50 GO TO (80,130), IEXC BAL00
37 C ***** SEARCH FOR ROWS ISOLATING AN EIGENVALUE BAL00
38 C AND PUSH THEM DOWN ***** BAL00
39 80 IF (L .EQ. 1) GO TO 280 BAL00
40 L = L - 1 BAL00
41 C ***** FOR J=L STEP -1 UNTIL 1 DO -- ***** BAL00
42 100 DO 120 JJ = 1, L BAL00
43 J = L + 1 - JJ BAL00
44 C BAL00
45 DO 110 I = 1, L BAL00
46 IF (I .EQ. J) GO TO 110 BAL00
47 IF (A(J,I) .NE. 0.000) GO TO 120 BAL00
48 110 CONTINUE BAL00
49 C BAL00
50 M = L BAL00
51 IEXC = 1 BAL00
52 GO TO 20 BAL00
53 120 CONTINUE BAL00
54 C BAL00
55 GO TO 140 BAL00
56 C ***** SEARCH FOR COLUMNS ISOLATING AN EIGENVALUE BAL00
57 C AND PUSH THEM LEFT ***** BAL00
58 130 K = K + 1 BAL00
59 C BAL00
60 140 DO 170 J = K, L BAL00
61 C BAL00
62 DO 150 I = K, L BAL00

```

63		IF (I .EQ. J) GO TO 150	BAL00
64		IF (A(I,J) .NE. 0.000) GO TO 170	BAL00
65	150	CONTINUE	BAL00
66	C		BAL00
67		M = K	BAL00
68		IEYC = 2	BAL00
69		GO TO 20	BAL00
70	170	CONTINUE	BAL00
71	C	***** NOW BALANCE THE SUBMATRIX IN ROWS K TO L *****	BAL00
72		DO 180 I = K, L	BAL00
73	180	SCALE(I) = 1.000	BAL00
74	C	***** ITERATIVE LOOP FOR NORM REDUCTION *****	BAL00
75	190	NOCONV = .FALSE.	BAL00
76	C		BAL00
77		DO 270 I = K, L	BAL00
78		C = 0.000	BAL00
79		R = 0.000	BAL00
80	C		BAL00
81		DO 200 J = K, L	BAL00
82		IF (J .EQ. I) GO TO 200	BAL00
83		C = C + DABS(A(J,I))	BAL00
84		R = R + DABS(A(I,J))	BAL00
85	200	CONTINUE	BAL00
86	C	***** GUARD AGAINST ZERO C OR R DUE TO UNDERFLOW *****	BAL00
87		IF (C .EQ. 0.000 .OR. R .EQ. 0.000) GO TO 270	BAL00
88		G = R / RADIX	BAL00
89		F = 1.000	BAL00
90		S = C + R	BAL00
91	210	IF (C .GE. G) GO TO 220	BAL00
92		F = F * RADIX	BAL00
93		C = C * B2	BAL00
94		GO TO 210	BAL00
95	220	G = R * RADIX	BAL00
96	230	IF (C .LT. G) GO TO 240	BAL00
97		F = F / RADIX	BAL00
98		C = C / B2	BAL00
99		GO TO 230	BAL01
100	C	***** NOW BALANCE *****	BAL01
101	240	IF ((C + R) / F .GE. 0.95 * S) GO TO 270	BAL01
102		G = 1.000 / F	BAL01
103		SCALE(I) = SCALE(I) * F	BAL01
104		NOCONV = .TRUE.	BAL01
105	C		BAL01
106		DO 250 J = K, N	BAL01
107	250	A(I,J) = A(I,J) * G	BAL01
108	C		BAL01
109		DO 260 J = 1, L	BAL01
110	260	A(J,I) = A(J,I) * F	BAL01
111	C		BAL01
112	270	CONTINUE	BAL01
113	C		BAL01
114		IF (NOCONV) GO TO 190	BAL01
115	C		BAL01
116	280	LOW = K	BAL01
117		IGH = L	BAL01
118		RETURN	BAL01
119	C	***** LAST CARD OF BALANC *****	BAL01
120		END	BAL01

0		SUBROUTINE ELMHES(NM,N,LOW,IGH,A,INT)	ELM000
1		IMPLICIT REAL*8 (A-H,O-Z)	ELM000
2		INTEGER I,J,M,N,LA,NM,IGH,KP1,LOW,MM1,MP1	ELM000
3		DIMENSION A(NM,N)	ELM000
4	C	REAL X,Y	ELM000
5	C	REAL DABS	ELM000
6		INTEGER INT(IGH)	ELM000
7	C		ELM000
8		LA = IGH - 1	ELM000
9		KP1 = LOW + 1	ELM001
10		IF (LA .LT. KP1) GO TO 200	ELM001
11	C		ELM001
12		DO 180 M = KP1, LA	ELM001
13		MM1 = M - 1	ELM001
14		X = 0.000	ELM001
15		I = M	ELM001
16	C		ELM001
17		DO 100 J = M, IGH	ELM001
18		IF (DABS(A(J,MM1)) .LE. DABS(X)) GO TO 100	ELM001
19		X = A(J,MM1)	ELM002
20		I = J	ELM002
21	100	CONTINUE	ELM002
22	C		ELM002
23		INT(M) = I	ELM002
24		IF (I .EQ. M) GO TO 130	ELM002
25	C	***** INTERCHANGE ROWS AND COLUMNS OF A *****	ELM002
26		DO 110 J = MM1, N	ELM002
27		Y = A(I,J)	ELM002
28		A(I,J) = A(M,J)	ELM002
29		A(M,J) = Y	ELM003
30	110	CONTINUE	ELM003
31	C		ELM003
32		DO 120 J = 1, IGH	ELM003
33		Y = A(J,I)	ELM003
34		A(J,I) = A(J,M)	ELM003
35		A(J,M) = Y	ELM003
36	120	CONTINUE	ELM003
37	C	***** END INTERCHANGE *****	ELM003
38	130	IF (X .EQ. 0.000) GO TO 180	ELM003
39		MP1 = M + 1	ELM004
40	C		ELM004
41		DO 160 I = MP1, IGH	ELM004
42		Y = A(I,MM1)	ELM004
43		IF (Y .EQ. 0.000) GO TO 160	ELM004
44		Y = Y / X	ELM004
45		A(I,MM1) = Y	ELM004
46	C		ELM004
47		DO 140 J = M, N	ELM004
48	140	A(I,J) = A(I,J) - Y * A(M,J)	ELM005
49	C		ELM005
50		DO 150 J = 1, IGH	ELM005
51	150	A(J,M) = A(J,M) + Y * A(J,I)	ELM005
52	C		ELM005
53	160	CONTINUE	ELM005
54	C		ELM005
55	180	CONTINUE	ELM005
56	C		ELM005
57	200	RETURN	ELM005
58	C	***** LAST CARD OF ELMHES *****	ELM005
59		END	ELM005

```

0      SUBROUTINE HQR(NM,N,LOW,IGH,H,WR,WI,IERR)                                HQR0
1      IMPLICIT REAL*8 (A-H,O-Z)                                             HQR00
2      REAL*8 NORM,MACHEP                                                    HQR0
3      INTEGER I,J,K,L,M,N,EN,LL,MM,NA,NM,IGH,ITS,LOW,MP2,ENM2,IERR          HQR0
4      DIMENSION H(NM,N),WR(N),WI(N)                                         HQR00
5 C    REAL P,Q,R,S,T,W,X,Y,ZZ,NORM,MACHEP                                  HQR00
6      REAL*8 DSQRT,DABS,DSIGN                                               HQR0
7 C    INTEGER MINO                                                            HQR0
8      LOGICAL NOTLAS                                                        HQR00
9 C                                                                            HQR0
10 C                                                                           HQR0
11 C    ***** MACHEP IS A MACHINE DEPENDENT PARAMETER SPECIFYING          HQR00
12 C    THE RELATIVE PRECISION OF FLOATING POINT ARITHMETIC.                HQR00
13 C                                                                           HQR0
14 C                                                                           HQR00
15      MACHEP = 16.**(-13)                                                  HQR00
16 C                                                                           HQR0
17      IERR = 0                                                              HQR0
18      NORM = 0.000                                                         HQR00
19      K = 1                                                                  HQR0
20 C    ***** STORE ROOTS ISOLATED BY BALANC                               HQR0
21 C    AND COMPUTE MATRIX NORM *****                                       HQR00
22      DO 50 I = 1, N                                                        HQR00
23 C                                                                           HQR0
24      DO 40 J = K, N                                                        HQR00
25      40  NORM = NORM + DABS(H(I,J))                                         HQR00
26 C                                                                           HQR0
27      K = I                                                                  HQR0
28      IF (I .GE. LOW .AND. I .LE. IGH) GO TO 50                            HQR00
29      WR(I) = H(I,I)                                                         HQR0
30      WI(I) = 0.000                                                         HQR0
31      50 CONTINUE                                                           HQR00
32 C                                                                           HQR0
33      EN = IGH                                                              HQR0
34      T = 0.000                                                             HQR0
35 C    ***** SEARCH FOR NEXT EIGENVALUES *****                           HQR00
36      60 IF (EN .LT. LOW) GO TO 1001                                         HQR0
37      ITS = 0                                                                HQR0
38      NA = EN - 1                                                            HQR00
39      ENM2 = NA - 1                                                         HQR00
40 C    ***** LOOK FOR SINGLE SMALL SUB-DIAGONAL ELEMENT                  HQR0
41 C    FOR L=EN STEP -1 UNTIL LOW DO == *****                             HQR00
42      70 DO 80 LL = LOW, EN                                                 HQR00
43      L = EN + LOW - LL                                                     HQR0
44      IF (L .EQ. LOW) GO TO 100                                             HQR0
45      S = DABS(H(L-1,L-1)) + DABS(H(L,L))                                    HQR00
46      IF (S .EQ. 0.000) S = NORM                                           HQR0
47      IF (DABS(H(L,L-1)) .LE. MACHEP * S) GO TO 100                        HQR0
48      80 CONTINUE                                                           HQR00
49 C    ***** FORM SHIFT *****                                           HQR00
50      100 X = H(EN,EN)                                                       HQR0
51      IF (L .EQ. EN) GO TO 270                                              HQR0
52      Y = H(NA,NA)                                                           HQR00
53      W = H(EN,NA) * H(NA,EN)                                              HQR0
54      IF (L .EQ. NA) GO TO 280                                              HQR0
55      IF (ITS .EQ. 30) GO TO 1000                                           HQR00
56      IF (ITS .NE. 10 .AND. ITS .NE. 20) GO TO 130                        HQR00
57 C    ***** FORM EXCEPTIONAL SHIFT *****                                 HQR0
58      T = T + X                                                             HQR0
59 C                                                                           HQR00
60      DO 120 I = LOW, EN                                                    HQR0
61      120 H(I,I) = H(I,I) - X                                               HQR0
62 C                                                                           HQR00

```



```

63 S = DABS(H(EN,NA)) + DABS(H(NA,ENM2))
64 X = 0.75 * S
65 Y = X
66 W = -0.4375 * S * S
67 130 ITS = ITS + 1
68 C ***** LOOK FOR TWO CONSECUTIVE SMALL
69 C SUB-DIAGONAL ELEMENTS.
70 C FOR M=EN-2 STEP -1 UNTIL L DO -- *****
71 DO 140 MM = L, ENM2
72 M = ENM2 + L - MM
73 ZZ = H(M,M)
74 R = X - ZZ
75 S = Y - ZZ
76 P = (R * S - W) / H(M+1,M) + H(M,M+1)
77 Q = H(M+1,M+1) - ZZ - R - S
78 R = H(M+2,M+1)
79 S = DABS(P) + DABS(Q) + DABS(R)
80 P = P / S
81 Q = Q / S
82 R = R / S
83 IF (M .EQ. L) GO TO 150
84 IF (DABS(H(M,M-1)) * (DABS(Q) + DABS(R)) .LE. MACHEP * DABS(P)
85 X * (DABS(H(M-1,M-1)) + DABS(ZZ) + DABS(H(M+1,M+1)))) GO TO 150
86 140 CONTINUE
87 C
88 150 MP2 = M + 2
89 C
90 DO 160 I = MP2, EN
91 H(I,I-2) = 0.000
92 IF (I .EQ. MP2) GO TO 160
93 H(I,I-3) = 0.000
94 160 CONTINUE
95 C ***** DOUBLE QR STEP INVOLVING ROWS L TO EN AND
96 C COLUMNS M TO EN *****
97 DO 260 K = M, NA
98 NOTLAS = K .NE. NA
99 IF (K .EQ. M) GO TO 170
100 P = H(K,K-1)
101 Q = H(K+1,K-1)
102 R = 0.000
103 IF (NOTLAS) R = H(K+2,K-1)
104 X = DABS(P) + DABS(Q) + DABS(R)
105 IF (X .EQ. 0.000) GO TO 260
106 P = P / X
107 Q = Q / X
108 R = R / X
109 170 S = DSIGN(DSQRT(P*P+Q*Q+R*R),P)
110 IF (K .EQ. M) GO TO 180
111 H(K,K-1) = -S * X
112 GO TO 190
113 180 IF (L .NE. M) H(K,K-1) = -H(K,K-1)
114 190 P = P * S
115 X = P / S
116 Y = Q / S
117 ZZ = R / S
118 Q = Q / P
119 R = R / P
120 C ***** ROW MODIFICATION *****
121 DO 210 J = K, EN
122 P = H(K,J) + Q * H(K+1,J)
123 IF (.NOT. NOTLAS) GO TO 200
124 P = P + R * H(K+2,J)
125 H(K+2,J) = H(K+2,J) - P * ZZ

```

```

126 200 H(K+1,J) = H(K+1,J) - P * Y HGR
127 H(K,J) = H(K,J) - P * X HGR
128 210 CONTINUE HGR
129 C HGR
130 J = MINO(EN,K+3) HGR
131 C ***** COLUMN MODIFICATION ***** HGR
132 DO 230 I = L, J HGR
133 P = X * H(I,K) + Y * H(I,K+1) HGR
134 IF (.NOT. NOTLAS) GO TO 220 HGR
135 P = P + ZZ * H(I,K+2) HGR
136 H(I,K+2) = H(I,K+2) - P * R HGR
137 220 H(I,K+1) = H(I,K+1) - P * Q HGR
138 H(I,K) = H(I,K) - P HGR
139 230 CONTINUE HGR
140 C HGR
141 260 CONTINUE HGR
142 C HGR
143 GO TO 70 HGR
144 C ***** ONE ROOT FOUND ***** HGR
145 270 WR(EN) = X + T HGR
146 WI(EN) = 0.000 HGR
147 EN = NA HGR
148 GO TO 60 HGR
149 C ***** TWO ROOTS FOUND ***** HGR
150 280 P = (Y - X) / 2.000 HGR
151 Q = P * P + W HGR
152 ZZ = DSQRT(DABS(Q)) HGR
153 X = X + T HGR
154 IF (Q .LT. 0.000) GO TO 320 HGR
155 C ***** REAL PAIR ***** HGR
156 ZZ = P + DSIGN(ZZ,P) HGR
157 WR(NA) = X + ZZ HGR
158 WR(EN) = WR(NA) HGR
159 IF (ZZ .NE. 0.000) WR(EN) = X - W / ZZ HGR
160 WI(NA) = 0.000 HGR
161 WI(EN) = 0.000 HGR
162 GO TO 330 HGR
163 C ***** COMPLEX PAIR ***** HGR
164 320 WR(NA) = X + P HGR
165 WR(EN) = X + P HGR
166 WI(NA) = ZZ HGR
167 WI(EN) = -ZZ HGR
168 330 EN = ENM2 HGR
169 GO TO 60 HGR
170 C ***** SET ERROR == NO CONVERGENCE TO AN HGR
171 C EIGENVALUE AFTER 30 ITERATIONS ***** HGR
172 1000 IERR = EN HGR
173 1001 RETURN HGR
174 C ***** LAST CARD OF HGR ***** HGR
175 END HGR

```

```

0      SUBROUTINE INVT(NM,N,A,WR,WI,SELECT,MM,M,Z,IERR,RM1,RV1,RV2)      INVO
1      IMPLICIT REAL*8 (A-H,O-Z)      INVO
2      REAL*8 NORM,NORMV,ILAMB0,MACHEP      INVO
3      INTEGER I,J,K,L,M,N,S,II,IP,MM,MP,NM,NS,N1,UK,IP1,ITS,KM1,IERR      RNVO
4      DIMENSION A(NM,N),WR(N),WI(N),Z(NM,MM),RM1(N,N),RV1(N),RV2(N)      INVO
5 C     REAL T,W,X,Y,EPS3,NORM,NORMV,GROWTO,ILAMB0,MACHEP,RLAMB0,UKROOT      INVO
6     REAL*8 DSQRT,CDABS,DABS,DFLOAT      INVO
7     INTEGER IABS      RNVO
8     LOGICAL*1 SELECT(N)      INVO
9     COMPLEX*16 Z3,DCMPLX      INVO
10    REAL*8 OREAL,DIRMAG      INVO
11 C     MACHEP = 16.**(-13)      INVO
12    MACHEP = 16.**(-13)      INVO
13 C     IERR = 0      INVO
14    IERR = 0      INVO
15    UK = 0      RNVO
16    S = 1      INVO
17 C     ***** IP = 0, REAL EIGENVALUE      INVO
18 C     1, FIRST OF CONJUGATE COMPLEX PAIR      INVO
19 C     -1, SECOND OF CONJUGATE COMPLEX PAIR *****      INVO
20    IP = 0      INVO
21    N1 = N - 1      RNVO
22 C     DO 980 K = 1, N      INVO
23    DO 980 K = 1, N      INVO
24    IF (WI(K) .EQ. 0.000 .OR. IP .LT. 0) GO TO 100      INVO
25    IP = 1      INVO
26    IF (SELECT(K) .AND. SELECT(K+1)) SELECT(K+1) = .FALSE.      INVO
27 100   IF (.NOT. SELECT(K)) GO TO 960      INVO
28    IF (WI(K) .NE. 0.000) S = S + 1      INVO
29    IF (S .GT. MM) GO TO 1000      INVO
30    IF (UK .GE. K) GO TO 200      INVO
31 C     ***** CHECK FOR POSSIBLE SPLITTING *****      INVO
32    DO 120 UK = K, N      INVO
33    IF (UK .EQ. N) GO TO 140      INVO
34    IF (A(UK+1,UK) .EQ. 0.000) GO TO 140      INVO
35 120   CONTINUE      INVO
36 C     ***** COMPUTE INFINITY NORM OF LEADING UK BY UK      INVO
37 C     (HESSENBERG) MATRIX *****      INVO
38 140   NORM = 0.000      INVO
39    MP = 1      INVO
40 C     DO 180 I = 1, UK      INVO
41    DO 180 I = 1, UK      INVO
42    X = 0.000      INVO
43 C     DO 160 J = MP, UK      INVO
44    DO 160 J = MP, UK      INVO
45 160   X = X + DABS(A(I,J))      INVO
46 C     IF (X .GT. NORM) NORM = X      INVO
47    IF (X .GT. NORM) NORM = X      INVO
48    MP = I      INVO
49 180   CONTINUE      INVO
50 C     ***** EPS3 REPLACES ZERO PIVOT IN DECOMPOSITION      INVO
51 C     AND CLOSE ROOTS ARE MODIFIED BY EPS3-*****      RNVO
52    IF (NORM .EQ. 0.000) NORM = 1.000      INVO
53    EPS3 = MACHEP * NORM      INVO
54 C     ***** GROWTO IS THE CRITERION FOR THE GROWTH *****      INVO
55    UKROOT = DSQRT(DFLOAT(UK))      INVO
56    GROWTO = 1.00-1 / UKROOT      INVO
57 200   RLAMB0 = WR(K)      INVO
58    ILAMB0 = WI(K)      INVO
59    IF (K .EQ. 1) GO TO 280      INVO
60    KM1 = K - 1      INVO
61    GO TO 240      INVO
62 C     ***** PEPTURB EIGENVALUE IF IT IS CLOSE      INVO

```

```

63 C          TO ANY PREVIOUS EIGENVALUE *****
64 220      RLAMB0 = RLAMB0 + EPS3
65 C          ***** FOR IK=1 STEP -1 UNTIL 1 DO -- *****
66 240      DO 260 II = 1, KMI
67          I = K - II
68          IF (SELECT(I) .AND. DABS(WR(I)-RLAMB0) .LT. EPS3 .AND.
69 X          DABS(WI(I)-ILAMB0) .LT. EPS3) GO TO 220
70 260      CONTINUE
71 C
72          WR(K) = RLAMB0
73 C          ***** PERTURB CONJUGATE EIGENVALUE TO MATCH *****
74          IP1 = K + IP
75          WR(IP1) = RLAMB0
76 C          ***** FORM UPPER HESSENBERG A-RLAMB0*I (TRANPOSED)
77 C          AND INITIAL REAL VECTOR *****
78 280      MP = 1
79 C
80          DO 320 I = 1, UK
81 C
82          DO 300 J = MP, UK
83 300      RM1(J,I) = A(I,J)
84 C
85          RM1(I,I) = RM1(I,I) - RLAMB0
86          MP = I
87          RVL(I) = EPS3
88 320      CONTINUE
89 C
90          ITS = 0
91          IF (ILAMB0 .NE. 0.000) GO TO 520
92 C          ***** REAL EIGENVALUE.
93 C          TRIANGULAR DECOMPOSITION WITH INTERCHANGES,
94 C          REPLACING ZERO PIVOTS BY EPS3 *****
95          IF (UK .EQ. 1) GO TO 420
96 C
97          DO 400 I = 2, UK
98          MP = I - 1
99          IF (DABS(RM1(MP,I)) .LE. DABS(RM1(MP,MP))) GO TO 360
100 C
101          DO 340 J = MP, UK
102          Y = RM1(J,I)
103          RM1(J,I) = RM1(J,MP)
104          RM1(J,MP) = Y
105 340      CONTINUE
106 C
107 360      IF (RM1(MP,MP) .EQ. 0.000) RM1(MP,MP) = EPS3
108          X = RM1(MP,I) / RM1(MP,MP)
109          IF (X .EQ. 0.000) GO TO 400
110 C
111          DO 380 J = I, UK
112 380      RM1(J,I) = RM1(J,I) - X * RM1(J,MP)
113 C
114 400      CONTINUE
115 C
116 420      IF (RM1(UK,UK) .EQ. 0.000) RM1(UK,UK) = EPS3
117 C          ***** BACK SUBSTITUTION FOR REAL VECTOR
118 C          FOR I=UK STEP -1 UNTIL 1 DO -- *****
119 440      DO 500 II = 1, UK
120          I = UK + 1 - II
121          Y = RVL(I)
122          IF (I .EQ. UK) GO TO 480
123          IP1 = I + 1
124 C
125          DO 460 J = IP1, UK

```

ORIGINAL PAGE IS  
OF POOR QUALITY

126	460	Y = Y - RM1(J,I) * RV1(J)	INVO
127	C		INVO
128	480	RV1(I) = Y / RM1(I,I)	INVO
129	500	CONTINUE	INVO
130	C		INVO
131		GO TO 740	INVO
132	C	***** COMPLEX EIGENVALUE.	INVO
133	C	TRIANGULAR DECOMPOSITION WITH INTERCHANGES,	RNVO
134	C	REPLACING ZERO PIVOTS BY EPS3. STORE IMAGINARY	INVO
135	C	PARTS IN UPPER TRIANGLE STARTING AT (1,3) *****	INVO
136	520	NS = N - S	RNVO
137		Z(1,S-1) = -ILAMB0	INVO
138		Z(1,S) = 0.000	INVO
139		IF (N .EQ. 2) GO TO 550	INVO
140		RM1(1,3) = -ILAMB0	INVO
141		Z(1,S-1) = 0.000	INVO
142		IF (N .EQ. 3) GO TO 550	INVO
143	C		INVO
144		DO 540 I = 4, N	INVO
145	540	RM1(1,I) = 0.000	INVO
146	C		INVO
147	550	DO 640 I = 2, UK	INVO
148		MP = I - 1	INVO
149		W = RM1(MP,I)	INVO
150		IF (I .LT. N) T = RM1(MP,I+1)	INVO
151		IF (I .EQ. N) T = Z(MP,S-1)	INVO
152		X = WM1(MP,MP) * RM1(MP,MP) + T * T	INVO
153		IF (W * W .LE. X) GO TO 580	INVO
154		X = RM1(MP,MP) / W	INVO
155		Y = T / W	INVO
156		RM1(MP,MP) = W	INVO
157		IF (I .LT. N) RM1(MP,I+1) = 0.000	INVO
158		IF (I .EQ. N) Z(MP,S-1) = 0.000	INVO
159	C		INVO
160		DO 560 J = I, UK	INVO
161		W = RM1(J,I)	INVO
162		RM1(J,I) = RM1(J,MP) - X * W	INVO
163		RM1(J,MP) = W	INVO
164		IF (J .LT. N1) GO TO 555	INVO
165		L = J - NS	INVO
166		Z(I,L) = Z(MP,L) - Y * W	INVO
167		Z(MP,L) = 0.000	INVO
168		GO TO 560	INVO
169	555	RM1(I,J+2) = RM1(MP,J+2) - Y * W	INVO
170		RM1(MP,J+2) = 0.000	INVO
171	560	CONTINUE	INVO
172	C		INVO
173		RM1(I,I) = RM1(I,I) - Y * ILAMB0	INVO
174		IF (I .LT. N1) GO TO 570	INVO
175		L = I - NS	INVO
176		Z(MP,L) = -ILAMB0	INVO
177		Z(I,L) = Z(I,L) + X * ILAMB0	INVO
178		GO TO 640	INVO
179	570	RM1(MP,I+2) = -ILAMB0	RNVO
180		RM1(I,I+2) = RM1(I,I+2) + X * ILAMB0	INVO
181		GO TO 640	INVO
182	580	IF (X .NE. 0.000) GO TO 600	INVO
183		RM1(MP,MP) = EPS3	INVO
184		IF (I .LT. N) RM1(MP,I+1) = 0.000	INVO
185		IF (I .EQ. N) Z(MP,S-1) = 0.000	INVO
186		T = 0.000	INVO
187		Y = EPS3 * EPS3	INVO
188	600	W = W / X	INVO

```

189      X = RM1(MP,MP) * W
190      Y = -T * W
191 C
192      DO 620 J = I, UK
193          IF (J .LT. N1) GO TO 610
194          L = J - NS
195          T = Z(MP,L)
196          Z(I,L) = -X * T - Y * RM1(J,MP)
197          GO TO 615
198      610      T = RM1(MP,J+2)
199          RM1(I,J+2) = -X * T - Y * RM1(J,MP)
200      615      RM1(J,I) = RM1(J,I) - X * RM1(J,MP) + Y * T
201      620      CONTINUE
202 C
203      IF (I .LT. N1) GO TO 630
204      L = I - NS
205      Z(I,L) = Z(I,L) - ILAMBD
206      GO TO 640
207      630      RM1(I,I+2) = RM1(I,I+2) - ILAMBD
208      640      CONTINUE
209 C
210      IF (UK .LT. N1) GO TO 650
211      L = UK - NS
212      T = Z(UK,L)
213      GO TO 655
214      650      T = RM1(UK,UK+2)
215      655      IF (RM1(UK,UK) .EQ. 0.000 .AND. T .EQ. 0.000) RM1(UK,UK) = EPS3
216 C ***** BACK SUBSTITUTION FOR COMPLEX VECTOR
217 C          FOR I=UK STEP -1 UNTIL 1 DO == *****
218      660      DO 720 II = 1, UK
219          I = UK + 1 - II
220          X = RV1(I)
221          Y = 0.000
222          IF (I .EQ. UK) GO TO 700
223          IP1 = I + 1
224 C
225          DO 680 J = IP1, UK
226              IF (J .LT. N1) GO TO 670
227              L = J - NS
228              T = Z(I,L)
229              GO TO 675
230      670      T = RM1(I,J+2)
231      675      X = X - RM1(J,I) * RV1(J) + T * RV2(J)
232              Y = Y - RM1(J,I) * RV2(J) - T * RV1(J)
233      680      CONTINUE
234 C
235      700      IF (I .LT. N1) GO TO 710
236      L = I - NS
237      T = Z(I,L)
238      GO TO 715
239      710      T = RM1(I,I+2)
240      715      Z3 = DCMLPX(X,Y) / DCMLPX(RM1(I,I),T)
241          RV1(I) = DREAL(Z3)
242          RV2(I) = DIMAG(Z3)
243      720      CONTINUE
244 C ***** ACCEPTANCE TEST FOR REAL OR COMPLEX
245 C          EIGENVECTOR AND NORMALIZATION *****
246      740      ITS = ITS + 1
247          NORM = 0.000
248          NORMV = 0.000
249 C
250      DO 780 I = 1, UK
251          IF (ILAMBD .EQ. 0.000) X = DABS(RV1(I))

```

```

252      IF (ILAMBD .NE. 0.000) X = COABS(DCMPLX(RV1(I),RV2(I)))      INVO1
253      IF (NORMV .GE. X) GO TO 760      INV02
254      NORMV = X      INV02
255      J = I      INV01
256 760      NORM = NORM + X      INV02
257 780      CONTINUE      INV02
258 C      INV01
259      IF (NORM .LT. GROWTO) GO TO 840      INV01
260 C      ***** ACCEPT VECTOR *****      INV02
261      X = RV1(J)      INV02
262      IF (ILAMBD .EQ. 0.000) X = 1.000 / X      INV01
263      IF (ILAMBD .NE. 0.000) Y = RV2(J)      RNV02
264 C      INV02
265      DO 820 I = 1, UK      /0
266      IF (ILAMBD .NE. 0.000) GO TO 800      INV01
267      Z(I,S) = RV1(I) * X      7. V02
268      GO TO 820      INV01
269 800      Z3 = DCMPLX(RV1(I),RV2(I)) / DCMPLX(X,Y)      INV01
270      Z(I,S-1) = DREAL(Z3)      RNV02
271      Z(I,S) = DIMAG(Z3)      INV02
272 820      CONTINUE      INV01
273 C      INV02
274      IF (UK .EQ. N) GO TO 940      INV02
275      J = UK + 1      INV01
276      GO TO 900      INV01
277 C      ***** IN-LINE PROCEDURE FOR CHOOSING      INV02
278 C      A NEW STARTING VECTOR *****      INV01
279 840      IF (ITS .GE. UK) GO TO 880      INV01
280      X = UKROOT      INV02
281      Y = EPS3 / (X + 1.000)      RNV02
282      RV1(1) = EPS3      INV01
283 C      INV01
284      DO 860 I = 2, UK      INV02
285 860      RV1(I) = Y      INV01
286 C      INV01
287      J = UK - ITS + 1      INV02
288      RV1(J) = RV1(J) - EPS3 * X      INV02
289      IF (ILAMBD .EQ. 0.000) GO TO 440      INV01
290      GO TO 660      INV02
291 C      ***** SET ERROR == UNACCEPTED EIGENVECTOR *****      INV02
292 880      J = 1      INV01
293      IERR = -K      INV01
294 C      ***** SET REMAINING VECTOR COMPONENTS TO ZERO *****      INV02
295 900      DO 920 I = J, N      INV02
296      Z(I,S) = 0.000      INV01
297      IF (ILAMBD .NE. 0.000) Z(I,S-1) = 0.000      INV02
298 920      CONTINUE      INV02
299 C      INV01
300 940      S = S + 1      INV01
301 960      IF (IP .EQ. (-1)) IP = 0      INV03
302      IF (IP .EQ. 1) IP = -1      INV01
303 980      CONTINUE      INV01
304 C      INV03
305      GO TO 1001      INV01
306 C      ***** SET ERROR == UNDERESTIMATE OF EIGENVECTOR      INV01
307 C      SPACE REQUIRED *****      INV03
308 1000      IF (IERR .NE. 0) IERR = IERR - N      INV03
309      IF (IERR .EQ. 0) IERR = -(2 * N + 1)      INV01
310 1001      M = S - 1 - IABS(IP)      INV01
311      RETURN      INV03
312 C      ***** LAST CARD OF INVIT *****      INV01
313      END      INV01

```

ORIGINAL PAGE IS  
OF POOR QUALITY

0	SUBROUTINE ELMLBAK(NM,LOW,IGH,A,INT,M,Z)	ELM000
1	IMPLICIT REAL*8 (A-H,O-Z)	ELM000
2	INTEGER I,J,M,LA,MM,MP,NM,IGH,KP1,LOW,MP1	ELM000
3	DIMENSION A(NM,IGH),Z(NM,M)	ELM000
4	C REAL X	ELM000
5	INTEGER INT(IGH)	ELM000
6	C	ELM000
7	C	ELM000
8	IF (M .EQ. 0) GO TO 200	ELM000
9	LA = IGH - 1	ELM000
10	KP1 = LOW + 1	ELM000
11	IF (LA .LT. KP1) GO TO 200	ELM001
12	C ***** FOR MP=IGH-1 STEP -1 UNTIL LOW+1 DO -- *****	ELM001
13	DO 140 MM = KP1, LA	ELM001
14	MP = LOW + IGH - MM	ELM001
15	MP1 = MP + 1	ELM001
16	C	ELM001
17	DO 110 I = MP1, IGH	ELM001
18	X = A(I,MP-1)	ELM001
19	IF (X .EQ. 0.0) GO TO 110	ELM002
20	C	ELM002
21	DO 100 J = 1, M	ELM002
22	100 Z(I,J) = Z(I,J) + X * Z(MP,J)	ELM002
23	C	ELM002
24	110 CONTINUE	ELM002
25	C	ELM002
26	I = INT(MP)	ELM002
27	IF (I .EQ. MP) GO TO 140	ELM002
28	C	ELM002
29	DO 130 J = 1, M	ELM003
30	X = Z(I,J)	ELM003
31	Z(I,J) = Z(MP,J)	ELM003
32	Z(MP,J) = X	ELM003
33	130 CONTINUE	ELM003
34	C	ELM003
35	140 CONTINUE	ELM003
36	C	ELM003
37	200 RETURN	ELM003
38	C ***** LAST CARD OF ELMLBAK *****	ELM003
39	END	ELM004



0	SUBROUTINE BALBAK(NM,N,LOW,IGH,SCALE,M,Z)	BAL00
1	IMPLICIT REAL*8 (A-H,O-Z)	BAL00
2	INTEGER I,J,K,M,N,II,NM,IGH,LOW	BAL00
3	DIMENSION SCALE(N),Z(NM,M)	BAL00
4	C REAL S	BAL00
5	IF (M .EQ. 0) GO TO 200	BAL00
6	IF (IGH .EQ. LOW) GO TO 120	BAL00
7	C	BAL00
8	DO 110 I = LOW, IGH	BAL00
9	S = SCALE(I)	BAL00
10	C ***** LEFT HAND EIGENVECTORS ARE BACK TRANSFORMED	BAL00
11	C IF THE FOREGOING STATEMENT IS REPLACED BY	BAL00
12	C S=1.0/SCALE(I). *****	BAL00
13	DO 100 J = 1, M	BAL00
14	100 Z(I,J) = Z(I,J) * S	BAL00
15	C	BAL00
16	110 CONTINUE	BAL00
17	C ***** FOR I=LOW-1 STEP -1 UNTIL 1,	BAL00
18	C IGH+1 STEP 1 UNTIL N DO == *****	BAL00
19	120 DO 140 II = 1, N	BAL00
20	I = II	BAL00
21	IF (I .GE. LOW .AND. I .LE. IGH) GO TO 140	BAL00
22	IF (I .LT. LOW) I = LOW - II	BAL00
23	K = SCALE(I)	BAL00
24	IF (K .EQ. I) GO TO 140	BAL00
25	C	BAL00
26	DO 130 J = 1, M	BAL00
27	S = Z(I,J)	BAL00
28	Z(I,J) = Z(K,J)	BAL00
29	Z(K,J) = S	BAL00
30	130 CONTINUE	BAL00
31	C	BAL00
32	140 CONTINUE	BAL00
33	C	BAL00
34	200 RETURN	BAL00
35	C ***** LAST CARD OF BALBAK *****	BAL00
36	END	BAL00

```

0      SUBROUTINE DETFAC(NMAX,N,A,IPIVOT,IDET,DETERM,ISCALE,WK,IERR)
1      IMPLICIT REAL*8 (A-H,O-Z)
2      DIMENSION A(NMAX,1),IPIVOT(1),WK(1)
3 C
4 C
5      ISCALE=0
6      NM1=N-1
7      IERR=0
8 C
9 C      DETERMINANT CALCULATION TEST
10 C
11      IF(IDET.EQ.1)GO TO 230
12 C
13 C      TEST FOR A SCALAR MATRIX
14 C
15      IF(NM1.GT.0)GO TO 20
16      DETERM=A(1,1)
17      RETURN
18 C
19 C      COMPUTE SCALING FACTORS
20 C
21      20 CONTINUE
22      DO 60 I=1,N
23      P=0.0
24      DO 30 J=1,N
25      Q=DMAX1(P,DABS(A(I,J)))
26      IF(Q.GT.P)P=Q
27      30 CONTINUE
28      IF(P)60,40,60
29      40 DETERM=0.0
30      IERR=1
31      RETURN
32      60 WK(I)=P
33 C
34      DO 210 M=1,NM1
35 C
36 C      PIVOTAL LOGIC SETUP
37 C
38      P=0.0
39      DO 110 I=M,N
40      Q=DABS(A(I,M)/WK(I))
41      IF(Q-P)110,110,100
42      100 P=Q
43      IP=I
44      110 CONTINUE
45 C
46      IPIVOT(M)=IP
47 C
48      IF(P.EQ.0.)GO TO 40
49      IF(M.EQ.IP)GO TO 155
50 C
51 C      PIVOT THE M-TH ROW OF THE A MATRIX
52 C
53      DO 150 I=1,N
54      P=A(IP,I)
55      A(IP,I)=A(M,I)
56      150 A(M,I)=P
57 C
58      P=WK(IP)
59      WK(IP)=WK(M)
60      WK(M)=P
61 C
62      155 MP1=M+1

```

63	C				DETO
64	C		L/U FACTORIZATION LOGIC		DETO
65	C				DETOO
66			P=A(M,M)		DETO
67			DO 180 I=MP1,N		DETO
68			A(I,M)=A(I,M)/P		DETOO
69			Q=A(I,M)		DETOO
70			DO 180 K=MP1,N		DETO
71	180		A(I,K)=A(I,K)-Q*A(M,K)		DETOO
72	C				DETOO
73	210		CONTINUE		DETO
74	C				DETO
75			IPIVOT(N)=N		DETOO
76			IF (A(N,N) .EQ. 0.0) GO TO 40		DETOO
77	C				DETO
78	C		CALCULATION OF THE DETERMINANT OF A		DETOO
79	C				DETOO
80			IF(IDET.EQ.0)RETURN		DETO
81	C				DETO
82	230		SIGN=1.0		DETOO
83			DETERM=1.0		DETO
84	C				DETO
85	C		ADJUST SIGN OF DETERMINANT DUE TO PIVOTAL STRATEGY		DETOO
86	C				DETOO
87			DO 250 I=1,NM1		DETO
88			IF(I-IPIVOT(I))240,250,240		DETOO
89	240		SIGN=-SIGN		DETOO
90	250		CONTINUE		DETO
91	C				DETO
92			DO 340 I=1,N		DETOO
93			P=A(I,I)		DETO
94	C				DETO
95	260		CONTINUE		DETOO
96			IF(R1.GT.DABS(P))GO TO 280		DETOO
97			P=P*R2		DETO
98			ISCALE=ISCALE+1		DETO
99			GO TO 260		DETO1
100	C				DETO
101	280		CONTINUE		DETO
102			IF(R2.LT.DABS(P))GO TO 290		DETO1
103			P=P*R1		DETO
104			ISCALE=ISCALE-1		DETO
105			GO TO 280		DETO1
106	C				DETO1
107	290		DETERM=DETERM*P		DETO
108	C				DETO
109	300		CONTINUE		DETO1
110			IF(R1.GT.DABS(DETERM))GO TO 320		DETO
111			DETERM=DETERM*R2		DETO
112			ISCALE=ISCALE+1		DETO1
113			GO TO 300		DETO1
114	C				DETO
115	320		CONTINUE		DETO
116			IF(R2.LT.DABS(DETERM))GO TO 340		DETO1
117			DETERM=DETERM*R1		DETO
118			ISCALE=ISCALE-1		DETO
119			GO TO 320		DETO1
120	C				DETO
121	340		CONTINUE		DETO
122	C				DETO1
123			DETERM=DETERM*SIGN		DETO1
124	C				DETO
125			RETURN		DETO

QUALITY

0	SUBROUTINE AXPXB(A,U,M,NA,NU,B,V,N,NB,NV,C,NC,EPSA,	AXP0
1	1EPSB,FAIL)	AXP0
2	IMPLICIT REAL*8 (A-H,O-Z)	AXP00
3	DIMENSION	AXP00
4	1A(NA,1),U(NU,1),B(NB,1),V(NV,1),C(NC,1)	AXP0
5	INTEGER	AXP00
6	C	AXP00
7	1 FAIL	AXP0
8	M1 = M+1	AXP0
9	MM1 = M-1	AXP00
10	N1 = N+1	AXP0
11	NM1 = N-1	AXP0
12	C IF REQUIRED, REDUCE A TO UPPER REAL SCHUR FORM.	AXP00
13	C	AXP00
14	IF(EPSA .LT. 0.) GO TO 35	AXP0
15	DO 10 I=1,M	AXP00
16	DO 10 J=I,M	AXP00
17	TEMP = A(I,J)	AXP0
18	A(I,J) = A(J,I)	AXP0
19	A(J,I) = TEMP	AXP00
20	10 CONTINUE	AXP00
21	CALL HSHLDR(A,M,NA)	AXP0
22	CALL BCKMLT(A,U,M,NA,NU)	AXP00
23	IF(MM1 .EQ. 0) GO TO 25	AXP00
24	DO 20 I=1,MM1	AXP0
25	A(I+1,I) = A(I,M1)	AXP0
26	20 CONTINUE	AXP00
27	CALL SCHUR(A,U,M,NA,NU,EPSA,FAIL)	AXP0
28	IF(FAIL .NE. 0) RETURN	AXP0
29	25 DO 30 I=1,M	AXP00
30	DO 30 J=I,M	AXP00
31	TEMP = A(I,J)	AXP0
32	A(I,J) = A(J,I)	AXP0
33	A(J,I) = TEMP	AXP00
34	30 CONTINUE	AXP0
35	C	AXP0
36	C IF REQUIRED, REDUCE B TO UPPER REAL SCHUR FORM.	AXP00
37	C	AXP00
38	35 IF(EPSB .LT. 0.) GO TO 45	AXP0
39	CALL HSHLDR(B,N,NB)	AXP00
40	CALL BCKMLT(B,V,N,NB,NV)	AXP00
41	IF(NM1 .EQ. 0) GO TO 45	AXP0
42	DO 40 I=1,NM1	AXP00
43	B(I+1,I) = B(I,N1)	AXP00
44	40 CONTINUE	AXP00
45	CALL SCHUR(B,V,N,NB,NV,EPSB,FAIL)	AXP00
46	FAIL = -FAIL	AXP00
47	IF(FAIL .NE. 0) RETURN	AXP00
48	C	AXP00
49	C TRANSFORM C.	AXP00
50	C	AXP00
51	45 DO 60 J=1,N	AXP00
52	DO 50 I=1,M	AXP00
53	A(I,M1) = 0.	AXP00
54	DO 50 K=1,M	AXP00
55	A(I,M1) = A(I,M1) + U(K,I)*C(K,J)	AXP00
56	50 CONTINUE	AXP00
57	DO 60 I=1,M	AXP00
58	C(I,J) = A(I,M1)	AXP00
59	60 CONTINUE	AXP00
60	DO 80 I=1,M	AXP00
61	DO 70 J=1,N	AXP00
62	B(N1,J) = 0.	AXP00

63	DO 70 K=1,N	AXPC
64	B(N1,J) = B(N1,J) + C(I,K)*V(K,J)	AXPC
65	70 CONTINUE	AXPC
66	DO 80 J=1,N	AXPC
67	C(I,J) = B(N1,J)	AXPC
68	80 CONTINUE	AXPC
69	C	AXPC
70	C SOLVE THE TRANSFORMED SYSTEM.	AXPC
71	C	AXPC
72	CALL SHRSLV(A,B,C,M,N,NA,NB,NC)	AXPC
73	C	AXPC
74	C TRANSFORM C BACK TO THE SOLUTION.	AXPC
75	C	AXPC
76	DO 100 J=1,N	AXPC
77	DO 90 I=1,M	AXPC
78	A(I,M1) = 0.	AXPC
79	DO 90 K=1,M	AXPC
80	A(I,M1) = A(I,M1) + U(I,K)*C(K,J)	AXPC
81	90 CONTINUE	AXPC
82	DO 100 I=1,M	AXPC
83	C(I,J) = A(I,M1)	AXPC
84	100 CONTINUE	AXPC
85	DO 120 I=1,M	AXPC
86	DO 110 J=1,N	AXPC
87	B(N1,J) = 0.	AXPC
88	DO 110 K=1,N	AXPC
89	B(N1,J) = B(N1,J) + C(I,K)*V(J,K)	AXPC
90	110 CONTINUE	AXPC
91	DO 120 J=1,N	AXPC
92	C(I,J) = B(N1,J)	AXPC
93	120 CONTINUE	AXPC
94	RETURN	AXPC
95	END	AXPC

```

0      SUBROUTINE SHRSLV(A,R,C,M,N,NA,NB,NC)
1      IMPLICIT REAL*8 (A-H,O-Z)
2      DIMENSION
3      1A(NA,1),B(NB,1),C(NC,1)
4      INTEGER
5      1 DK,DL
6      COMMON/SLVBLK/T(5,5),P(5),NSYS
7      L = 1
8      10  LM1 = L-1
9      DL = 1
10     IF(L .EQ. N) GO TO 15
11     IF(B(L+1,L) .NE. 0.) DL = 2
12     15  LL = L+DL-1
13     IF(L .EQ. 1) GO TO 30
14     DO 20 J=L,LL
15     DO 20 I=1,M
16         DO 20 IB=1,LM1
17             C(I,J) = C(I,J) - C(I,IB)*B(IB,J)
18     20  CONTINUE
19     30  K = 1
20     40  KM1 = K-1
21     DK = 1
22     IF(K .EQ. M) GO TO 45
23     IF(A(K,K+1) .NE. 0.) DK = 2
24     45  KK = K+DK-1
25     IF(K .EQ. 1) GO TO 60
26     DO 50 I=K,KK
27         DO 50 J=L,LL
28             DO 50 JA=1,KM1
29                 C(I,J) = C(I,J) - A(I,JA)*C(JA,J)
30     50  CONTINUE
31     60  IF(DL .EQ. 2) GO TO 80
32     IF(DK .EQ. 2) GO TO 70
33     T(1,1) = A(K,K) + B(L,L)
34     IF(T(1,1) .EQ. 0.) STOP
35     C(K,L) = C(K,L)/T(1,1)
36     GO TO 100
37     70  T(1,1) = A(K,K) + B(L,L)
38     T(1,2) = A(K,KK)
39     T(2,1) = A(KK,K)
40     T(2,2) = A(KK,KK) + B(L,L)
41     P(1) = C(K,L)
42     P(2) = C(KK,L)
43     NSYS = 2
44     CALL SYSSLV
45     C(K,L) = P(1)
46     C(KK,L) = P(2)
47     GO TO 100
48     80  IF(DK .EQ. 2) GO TO 90
49     T(1,1) = A(K,K) + B(L,L)
50     T(1,2) = B(LL,L)
51     T(2,1) = B(L,LL)
52     T(2,2) = A(K,K) + B(LL,LL)
53     P(1) = C(K,L)
54     P(2) = C(K,LL)
55     NSYS = 2
56     CALL SYSSLV
57     C(K,L) = P(1)
58     C(K,LL) = P(2)
59     GO TO 100
60     90  T(1,1) = A(K,K) + B(L,L)
61     T(1,2) = A(K,KK)
62     T(1,3) = B(LL,L)

```

63	T(1,4) = 0.	SHR00
64	T(2,1) = A(KK,K)	SHR00
65	T(2,2) = A(KK,KK) + B(L,L)	SHR00
66	T(2,3) = 0.	SHR00
67	T(2,4) = T(1,3)	SHR00
68	T(3,1) = B(L,LL)	SHR00
69	T(3,2) = 0.	SHR00
70	T(3,3) = A(K,K) + B(LL,LL)	SHR00
71	T(3,4) = T(1,2)	SHR00
72	T(4,1) = 0.	SHR00
73	T(4,2) = T(3,1)	SHR00
74	T(4,3) = T(2,1)	SHR00
75	T(4,4) = A(KK,KK) + B(LL,LL)	SHR00
76	P(1) = C(K,L)	SHR00
77	P(2) = C(KK,L)	SHR00
78	P(3) = C(K,LL)	SHR00
79	P(4) = C(KK,LL)	SHR00
80	NSYS = 4	SHR00
81	CALL SYSSLV	SHR00
82	C(K,L) = P(1)	SHR00
83	C(KK,L) = P(2)	SHR00
84	C(K,LL) = P(3)	SHR00
85	C(KK,LL) = P(4)	SHR00
86	100 K = K + DK	SHR00
87	IF(K .LE. M) GO TO 40	SHR00
88	L = L + DL	SHR00
89	IF(L .LE. N) GO TO 10	SHR00
90	RETURN	SHR00
91	END	SHR00





63	90	CONTINUE		ATX00
64		DO 100 J=1,N		ATX00
65		C(I,J) = A(N1,J)		ATX00
66	100	CONTINUE		ATX00
67		DO 120 J=1,N		ATX00
68		DO 110 I=1,N		ATX00
69		A(I,N1) = 0.		ATX00
70		DO 110 K=1,N		ATX00
71		A(I,N1) = A(I,N1) + U(I,K)*C(K,J)		ATX00
72	110	CONTINUE		ATX00
73		DO 120 I=1,N		ATX00
74		C(I,J) = A(I,N1)		ATX00
75	120	CONTINUE		ATX00
76		DO 130 I=1,N		ATX00
77		DO 130 J=I,N		ATX00
78		C(I,J) = C(I,J) + C(J,I)		ATX00
79		C(J,I) = C(I,J)		ATX00
80	130	CONTINUE		ATX00
81		RETURN		ATX00
82		END		ATX00

```

0      SUBROUTINE SYMSLV(A,C,N,NA,NC)
1      IMPLICIT REAL*8 (A-H,O-Z)
2      DIMENSION
3      1 A(NA,1),C(NC,1)
4      INTEGER
5      1 DK,OL
6      COMMON/SLVBLK/T(5,5),P(5),NSYS
7      L = 1
8      10  DL = 1
9      IF(L .EQ. N) GO TO 20
10     IF(A(L+1,L) .NE. 0.) DL = 2
11     20  LL = L+DL-1
12     K = L
13     30  KM1 = K-1
14     DK = 1
15     IF(K .EQ. N) GO TO 35
16     IF(A(K+1,K) .NE. 0.) DK = 2
17     35  KK = K+DK-1
18     IF(K .EQ. L) GO TO 45
19     DO 40 I=K, KK
20     DO 40 J=L, LL
21     DO 40 IA=L, KM1
22     C(I,J) = C(I,J) - A(IA,I)*C(IA,J)
23     40  CONTINUE
24     45  IF(DL .EQ. 2) GO TO 60
25     IF(DK .EQ. 2) GO TO 50
26     T(1,1) = A(K,K) + A(L,L)
27     IF(T(1,1) .EQ. 0.) STOP
28     C(K,L) = C(K,L)/T(1,1)
29     GO TO 90
30     50  T(1,1) = A(K,K) + A(L,L)
31     T(1,2) = A(KK,K)
32     T(2,1) = A(K, KK)
33     T(2,2) = A(KK, KK) + A(L,L)
34     P(1) = C(K,L)
35     P(2) = C(KK,L)
36     NSYS = 2
37     CALL SYSSLV
38     C(K,L) = P(1)
39     C(KK,L) = P(2)
40     GO TO 90
41     60  IF(DK .EQ. 2) GO TO 70
42     T(1,1) = A(K,K) + A(L,L)
43     T(1,2) = A(LL,L)
44     T(2,1) = A(L, LL)
45     T(2,2) = A(K,K) + A(LL,LL)
46     P(1) = C(K,L)
47     P(2) = C(K,LL)
48     NSYS = 2
49     CALL SYSSLV
50     C(K,L) = P(1)
51     C(K,LL) = P(2)
52     GO TO 90
53     70  IF(K .NE. L) GO TO 80
54     T(1,1) = A(L,L)
55     T(1,2) = A(LL,L)
56     T(1,3) = 0.
57     T(2,1) = A(L,LL)
58     T(2,2) = A(L,L) + A(LL,LL)
59     T(2,3) = T(1,2)
60     T(3,1) = 0.
61     T(3,2) = T(2,1)
62     T(3,3) = A(LL,LL)

```

63		P(1) = C(L,L)/2.	SYMO
64		P(2) = C(LL,L)	SYMO
65		P(3) = C(LL,LL)/2.	SYMO
66		NSYS = 3	SYMO
67		CALL SYSSLV	SYMO
68		C(L,L) = P(1)	SYMO
69		C(LL,L) = P(2)	SYMO
70		C(L,LL) = P(2)	SYMO
71		C(LL,LL) = P(3)	SYMO
72		GO TO 90	SYMO
73	80	T(1,1) = A(K,K) + A(L,L)	SYMO
74		T(1,2) = A(KK,K)	SYMO
75		T(1,3) = A(LL,L)	SYMO
76		T(1,4) = 0.	SYMO
77		T(2,1) = A(K,KK)	SYMO
78		T(2,2) = A(KK,KK) + A(L,L)	SYMO
79		T(2,3) = 0.	SYMO
80		T(2,4) = T(1,3)	SYMO
81		T(3,1) = A(L,LL)	SYMO
82		T(3,2) = 0.	SYMO
83		T(3,3) = A(K,K) + A(LL,LL)	SYMO
84		T(3,4) = T(1,2)	SYMO
85		T(4,1) = 0.	SYMO
86		T(4,2) = T(3,1)	SYMO
87		T(4,3) = T(2,1)	SYMO
88		T(4,4) = A(KK,KK) + A(LL,LL)	SYMO
89		P(1) = C(K,L)	SYMO
90		P(2) = C(KK,L)	SYMO
91		P(3) = C(K,LL)	SYMO
92		P(4) = C(KK,LL)	SYMO
93		NSYS = 4	SYMO
94		CALL SYSSLV	SYMO
95		C(K,L) = P(1)	SYMO
96		C(KK,L) = P(2)	SYMO
97		C(K,LL) = P(3)	SYMO
98		C(KK,LL) = P(4)	SYMO
99	90	K = K + DK	SYMO1
100		IF(K .LE. N) GO TO 30	SYMO1
101		LDL = L + DL	SYMO1
102		IF(LDL .GT. N) RETURN	SYMO1
103		DO 120 J=LDL,N	SYMO1
104		DO 100 I=L,LL	SYMO1
105		C(I,J) = C(J,I)	SYMO1
106	100	CONTINUE	SYMO1
107		DO 120 I=J,N	SYMO1
108		DO 110 K=L,LL	SYMO1
109		C(I,J) = C(I,J) - C(I,K)*A(K,J) - A(K,I)*C(K,J)	SYMO1
110	110	CONTINUE	SYMO1
111		C(J,I) = C(I,J)	SYMO1
112	120	CONTINUE	SYMO1
113		L = LDL	SYMO1
114		GO TO 10	SYMO1
115		END	SYMO1

0	SUBROUTINE HSHLDR(A,N,NA)	HSH0
1	IMPLICIT REAL*8 (A-H,O-Z)	HSH0
2	DIMENSION A(NA,1)	HSH0
3	REAL*8 MAX	HSH0
4	C	HSH0
5	NM2 = N-2	HSH0
6	N1 = N+1	HSH0
7	IF(N .EQ. 1) RETURN	HSH0
8	IF(N .GT. 2) GO TO 5	HSH0
9	A(1,N1) = A(2,1)	HSH0
10	RETURN	HSH0
11	5 DO 80 L=1,NM2	HSH0
12	L1 = L+1	HSH0
13	MAX = 0.	HSH0
14	DO 10 I=L1,N	HSH0
15	MAX = DMAX1(MAX,DABS(A(I,L)))	HSH0
16	10 CONTINUE	HSH0
17	IF(MAX .NE. 0.) GO TO 20	HSH0
18	A(L,N1) = 0.	HSH0
19	A(N1,L) = 0.	HSH0
20	GO TO 80	HSH0
21	20 SUM = 0.	HSH0
22	DO 30 I=L1,N	HSH0
23	A(I,L) = A(I,L)/MAX	HSH0
24	SUM = SUM + A(I,L)**2	HSH0
25	30 CONTINUE	HSH0
26	S = DSIGN(DSQRT(SUM),A(L1,L))	HSH0
27	A(L,N1) = -MAX*S	HSH0
28	A(L1,L) = S + A(L1,L)	HSH0
29	A(N1,L) = S*A(L1,L)	HSH0
30	DO 50 J=L1,N	HSH0
31	SUM = 0.	HSH0
32	DO 40 I=L1,N	HSH0
33	SUM = SUM + A(I,L)*A(I,J)	HSH0
34	40 CONTINUE	HSH0
35	P = SUM/A(N1,L)	HSH0
36	DO 50 I=L1,N	HSH0
37	A(I,J) = A(I,J) - A(I,L)*P	HSH0
38	50 CONTINUE	HSH0
39	DO 70 I=1,N	HSH0
40	SUM = 0.	HSH0
41	DO 60 J=L1,N	HSH0
42	SUM = SUM + A(I,J)*A(J,L)	HSH0
43	60 CONTINUE	HSH0
44	P = SUM/A(N1,L)	HSH0
45	DO 70 J=L1,N	HSH0
46	A(I,J) = A(I,J) - P*A(J,L)	HSH0
47	70 CONTINUE	HSH0
48	80 CONTINUE	HSH0
49	A(N-1,N1) = A(N,N-1)	HSH0
50	RETURN	HSH0
51	END	HSH0

ORIGINAL PAGE IS  
OF POOR QUALITY

0	SUBROUTINE BCKMLT(A,U,N,NA,NU)	BCKO
1	IMPLICIT REAL*8 (A-H,O-Z)	BCKO
2	DIMENSION	BCKO
3	IA(NA,1),U(NU,1)	BCKO
4	C	BCKO
5	N1 = N+1	BCKO
6	NM1 = N-1	BCKO
7	NM2 = N-2	BCKO
8	U(N,N) = 1.	BCKO
9	IF(NM1 .EQ. 0) RETURN	BCKO
10	U(NM1,N) = 0.	BCKO
11	U(N,NM1) = 0.	BCKO
12	U(NM1,NM1) = 1.	BCKO
13	IF(NM2 .EQ. 0) RETURN	BCKO
14	DO 40 LL=1,NM2	BCKO
15	L = NM2-LL+1	BCKO
16	L1 = L+1	BCKO
17	IF(A(N1,L) .EQ. 0.) GO TO 25	BCKO
18	DO 20 J=L1,N	BCKO
19	SUM = 0.	BCKO
20	DO 10 I=L1,N	BCKO
21	SUM = SUM + A(I,L)*U(I,J)	BCKO
22	10 CONTINUE	BCKO
23	P = SUM/A(N1,L)	BCKO
24	DO 20 I=L1,N	BCKO
25	U(I,J) = U(I,J) - A(I,L)*P	BCKO
26	20 CONTINUE	BCKO
27	25 DO 30 I=L1,N	BCKO
28	U(I,L) = 0.	BCKO
29	U(L,I) = 0.	BCKO
30	30 CONTINUE	BCKO
31	U(L,L) = 1.	BCKO
32	40 CONTINUE	BCKO
33	RETURN	BCKO
34	END	BCKO

0	SUBROUTINE SCHUR(H,U,NN,NH,NU,EPS,FAIL)	SC
1	IMPLICIT REAL*4 (A-H,O-Z)	SC
2	DIMENSION	SC
3	1H(NH,1),U(NU,1)	SC
4	INTEGER	SC
5	1 FAIL	SC
6	LOGICAL	SC
7	1LAST	SC
8	N = NN	SC
9	HN = 0.	SC
10	DO 20 I=1,N	SC
11	JL = MAX0(1,I-1)	SC
12	RSUM = 0.	SC
13	DO 10 J=JL,N	SC
14	RSUM = RSUM + DABS(H(I,J))	SC
15	10 CONTINUE	SC
16	HN = DMAX1(HN,RSUM)	SC
17	20 CONTINUE	SC
18	TEST = EPS*HN	SC
19	IF(HN .EQ. 0.) GO TO 230	SC
20	30 IF(N .LE. 1) GO TO 230	SC
21	ITS = 0	SC
22	NA = N-1	SC
23	NM2 = N-2	SC
24	40 DO 50 LL=2,N	SC
25	L = N-LL+2	SC
26	IF(DABS(H(L,L-1)) .LE. TEST) GO TO 60	SC
27	50 CONTINUE	SC
28	L = 1	SC
29	GO TO 70	SC
30	60 H(L,L-1) = 0.	SC
31	70 IF(L .LT. NA) GO TO 72	SC
32	N = L-1	SC
33	GO TO 30	SC
34	72 X = H(N,N)/HN	SC
35	Y = H(NA,NA)/HN	SC
36	R = (H(N,NA)/HN)*(H(NA,N)/HN)	SC
37	IF(ITS .LT. 30) GO TO 75	SC
38	FAIL = N	SC
39	RETURN	SC
40	75 IF(ITS.EQ.10 .OR. ITS.EQ.20) GO TO 80	SC
41	S = X + Y	SC
42	Y = X*Y - R	SC
43	GO TO 90	SC
44	80 Y = (DABS(H(N,NA)) + DABS(H(NA,NM2)))/HN	SC
45	S = 1.5*Y	SC
46	Y = Y**2	SC
47	90 ITS = ITS + 1	SC
48	DO 100 MM=L,NM2	SC
49	M = NM2-MM+L	SC
50	X = H(M,M)/HN	SC
51	R = H(M+1,M)/HN	SC
52	Z = H(M+1,M+1)/HN	SC
53	P = X*(X-S) + Y + R*(H(M,M+1)/HN)	SC
54	Q = R*(X+Z-S)	SC
55	R = R*(H(M+2,M+1)/HN)	SC
56	W = DABS(P) + DABS(Q) + DABS(R)	SC
57	P = P/W	SC
58	Q = Q/W	SC
59	R = R/W	SC
60	IF(M .EQ. L) GO TO 110	SC
61	IF(DABS(H(M,M-1))*(DABS(Q)+DABS(R)) .LE. DABS(P)*TEST)	SC
62	1GO TO 110	SC

63	100	CONTINUE	SCH00
64	110	M2 = M+2	SCH00
65		M3 = M+3	SCH00
66		DO 120 I=M2,N	SCH00
67		H(I,I-2) = 0.	SCH00
68	120	CONTINUE	SCH00
69		IF(M3 .GT. N) GO TO 140	SCH00
70		DO 130 I=M3,N	SCH00
71		H(I,I-3) = 0.	SCH00
72	130	CONTINUE	SCH00
73	140	DO 220 K=M,NA	SCH00
74		LAST = K.EQ.NA	SCH00
75		IF(K .EQ. M) GO TO 150	SCH00
76		P = H(K,K-1)	SCH00
77		Q = H(K+1,K-1)	SCH00
78		R = 0.	SCH00
79		IF(.NOT.LAST) R = H(K+2,K-1)	SCH00
80		X = DABS(P) + DABS(Q) + DABS(R)	SCH00
81		IF(X .EQ. 0.) GO TO 220	SCH00
82		P = P/X	SCH00
83		Q = Q/X	SCH00
84		R = R/X	SCH00
85	150	S = DSQRT(P**2 + Q**2 + R**2)	SCH00
86		IF(P .LT. 0.) S = -S	SCH00
87		IF(K .NE. M) H(K,K-1) = -S*X	SCH00
88		IF(K.EQ.M .AND. L.NE.M) H(K,K-1) = -H(K,K-1)	SCH00
89		P = P + S	SCH00
90		X = P/S	SCH00
91		Y = Q/S	SCH00
92		Z = R/S	SCH00
93		Q = Q/P	SCH00
94		R = R/P	SCH00
95		DO 170 J=K,NN	SCH00
96		P = H(K,J) + Q*H(K+1,J)	SCH00
97		IF(LAST) GO TO 160	SCH00
98		P = P + R*H(K+2,J)	SCH00
99		H(K+2,J) = H(K+2,J) - P*Z	SCH01
100	160	H(K+1,J) = H(K+1,J) - P*Y	SCH01
101		H(K,J) = H(K,J) - P*X	SCH01
102	170	CONTINUE	SCH01
103		J = MIN0(K+3,N)	SCH01
104		DO 190 I=1,J	SCH01
105		P = X*H(I,K) + Y*H(I,K+1)	SCH01
106		IF(LAST) GO TO 180	SCH01
107		P = P + Z*H(I,K+2)	SCH01
108		H(I,K+2) = H(I,K+2) - P*R	SCH01
109	180	H(I,K+1) = H(I,K+1) - P*Q	SCH01
110		H(I,K) = H(I,K) - P	SCH01
111	190	CONTINUE	SCH01
112		DO 210 I=1,NN	SCH01
113		P = X*U(I,K) + Y*U(I,K+1)	SCH01
114		IF(LAST) GO TO 200	SCH01
115		P = P + Z*U(I,K+2)	SCH01
116		U(I,K+2) = U(I,K+2) - P*R	SCH01
117	200	U(I,K+1) = U(I,K+1) - P*Q	SCH01
118		U(I,K) = U(I,K) - P	SCH01
119	210	CONTINUE	SCH01
120	220	CONTINUE	SCH01
121		GO TO 40	SCH01
122	230	FAIL = 0	SCH01
123		RETURN	SCH01
124		END	SCH01



```

0      SUBROUTINE SYSSLV                                SYS0
1      IMPLICIT REAL*8 (A-H,O-Z)                       SYS0
2      C                                                SYS0
3      COMMON/SLV8LK/A(5,5),B(5),N                     SYS0
4      REAL*8 MAX                                       SYS0
5      1 NM1 = N - 1                                    SYS0
6      N1 = N+1                                         SYS0
7      C                                                SYS0
8      C COMPUTE THE LU FACTORIZATION OF A.             SYS0
9      DO 80 K=1,N                                       SYS0
10     KM1 = K-1                                         SYS0
11     IF(K.EQ.1) GO TO 20                               SYS0
12     DO 10 I=K,N                                       SYS0
13     DO 10 J=1,KM1                                     SYS0
14     A(I,K) = A(I,K) - A(I,J)*A(J,K)                 SYS0
15     10 CONTINUE                                       SYS0
16     20 IF(K.EQ.N) GO TO 100                           SYS0
17     KP1 = K+1                                         SYS0
18     MAX = DABS(A(K,K))                                SYS0
19     INTR = K                                          SYS0
20     DO 30 I=KP1,N                                     SYS0
21     AA = DABS(A(I,K))                                 SYS0
22     IF(AA .LE. MAX) GO TO 30                         SYS0
23     MAX = AA                                         SYS0
24     INTR = I                                          SYS0
25     30 CONTINUE                                       SYS0
26     IF(MAX .EQ. 0.) STOP                             SYS0
27     A(N1,K) = INTR                                   SYS0
28     IF(INTR .EQ. K) GO TO 50                         SYS0
29     DO 40 J=1,N                                       SYS0
30     TEMP = A(K,J)                                    SYS0
31     A(K,J) = A(INTR,J)                              SYS0
32     A(INTR,J) = TEMP                                 SYS0
33     40 CONTINUE                                       SYS0
34     50 DO 80 J=KP1,N                                   SYS0
35     IF(K.EQ.1) GO TO 70                             SYS0
36     DO 60 I=1,KM1                                    SYS0
37     A(K,J) = A(K,J) - A(K,I)*A(I,J)                 SYS0
38     60 CONTINUE                                       SYS0
39     70 A(K,J) = A(K,J)/A(K,K)                       SYS0
40     80 CONTINUE                                       SYS0
41     C                                                SYS0
42     C INTERCHANGE THE COMPONENTS OF B.              SYS0
43     C                                                SYS0
44     100 DO 110 J=1,NM1                                SYS0
45     INTR = A(N1,J)                                   SYS0
46     IF(INTR .EQ. J) GO TO 110                       SYS0
47     TEMP = B(J)                                      SYS0
48     B(J) = B(INTR)                                  SYS0
49     B(INTR) = TEMP                                  SYS0
50     110 CONTINUE                                       SYS0
51     C                                                SYS0
52     C SOLVE LX = B.                                  SYS0
53     C                                                SYS0
54     200 B(1) = B(1)/A(1,1)                            SYS0
55     DO 220 I=2,N                                     SYS0
56     IM1 = I-1                                        SYS0
57     DO 210 J=1,IM1                                   SYS0
58     B(I) = B(I) - A(I,J)*B(J)                       SYS0
59     210 CONTINUE                                       SYS0
60     B(I) = B(I)/A(I,I)                               SYS0
61     220 CONTINUE                                       SYS0
62     C                                                SYS0

```

```
63 C SOLVE UX = B. SYS00
64 C SYS00
65 300 DO 310 II=1,NM1 SYS00
66 I = NM1-II+1 SYS00
67 II = I+1 SYS00
68 DO 310 J=II,N SYS00
69 B(I) = B(I) - A(I,J)*B(J) SYS00
70 310 CONTINUE SYS00
71 RETURN SYS00
72 END SYS00
```

ORIGINAL PAGE IS  
OF POOR QUALITY

0	SUBROUTINE GAUSEL (MAX, N, A, NR, B, IERR)		GAU
1	IMPLICIT REAL*8 (A-H,O-Z)		GAU
2	C	FUNCTION - COMPUTES SOLUTION TO A SET OF SIMULTANEOUS	GAU
3	C	LINEAR EQUATIONS (DOES NOT GIVE PIVOT OR	GAU
4	C	DETERMINANT DATA)	GAU
5	C	USAGE - CALL GAUSEL (MAX,N,A,NR,B,IERR)	GAU
6	C	PARAMETERS MAX - MAXIMUM ROW DIMENSION OF B	GAU
7	C	N - ORDER OF A	GAU
8	C	A(N,N) - INPUT MATRIX OF COEFFICIENTS (DESTROYED)	GAU
9	C	NR - NUMBER OF COLUMNS IN B	GAU
10	C	B(MAX,NR) - MATRIX OF CONSTANTS (REPLACED BY SOLUTIONS)	GAU
11	C	IERR - INTEGER ERROR CODE	GAU
12	C	= 0 NORMAL RETURN	GAU
13	C	= 2 INPUT MATRIX IS SINGULAR	GAU
14	C	REQUIRED ROUTINES - NONE	GAU
15	C		GAU
16	C	SOURCE	GAU
17	C	NASA, LRC, ANALYSIS AND COMPUTATION DIVISION SUBPROGRAM	GAU
18	C	LIBRARY	GAU
19	C	****	GAU
20		DIMENSION A(N,N),B(MAX,NR)	GAU
21		NM1 = N-1	GAU
22		IF (NM1 .EQ. 0) GO TO 140	GAU
23	C	****	GAU
24	C	FIND LARGEST REMAINING ELEMENT IN I-TH COLUMN FOR PIVOT	GAU
25	C	****	GAU
26		DO 100 I=1,NM1	GAU
27		BIG = 0.	GAU
28		DO 20 K=I,N	GAU
29		TERM = DABS(A(K,I))	GAU
30		IF (TERM - BIG) 20,20,10	GAU
31	10	BIG = TERM	GAU
32		L = K	GAU
33	20	CONTINUE	GAU
34		IF (BIG) 40,30,40	GAU
35	30	IERR = 2	GAU
36		RETURN	GAU
37	40	IF (I=L) 50,80,50	GAU
38	C	****	GAU
39	C	PIVOT ROWS OF A AND B	GAU
40	C	****	GAU
41	50	CONTINUE	GAU
42		DO 60 J=1,N	GAU
43		TEMP = A(I,J)	GAU
44		A(I,J) = A(L,J)	GAU
45		A(L,J) = TEMP	GAU
46	60	CONTINUE	GAU
47		DO 70 J=1,NR	GAU
48		TEMP = B(I,J)	GAU
49		B(I,J) = B(L,J)	GAU
50		B(L,J) = TEMP	GAU
51	70	CONTINUE	GAU
52	80	CONTINUE	GAU
53	C	****	GAU
54	C	STORE PIVOT AND PERFORM COLUMN OPERATIONS ON A AND B	GAU
55	C	****	GAU
56		IP1 = I+1	GAU
57		DO 100 II=IP1,N	GAU
58		A(II,I) = A(II,I)/A(I,I)	GAU
59		X3 = A(II,I)	GAU
60		DO 90 K=IP1,N	GAU
61		A(II,K) = A(II,K) - X3*A(I,K)	GAU
62	90	CONTINUE	GAU

63	DO 100 K=1, NR	GAU0
64	B(II, K) = B(II, K) - X3*B(I, K)	GAU0
65	100 CONTINUE	GAU0
66	C ****	GAU0
67	C PERFORM BACK SUBSTITUTION	GAU0
68	C ****	GAU0
69	DO 110 IC=1, NR	GAU0
70	B(N, IC) = B(N, IC)/A(N, N)	GAU0
71	110 CONTINUE	GAU0
72	DO 130 KK=1, NM1	GAU0
73	I = N-KK	GAU0
74	IP1 = I+1	GAU0
75	DO 130 J=1, NR	GAU0
76	SUM = B(I, J)	GAU0
77	DO 120 K=IP1, N	GAU0
78	SUM = SUM - A(I, K)*B(K, J)	GAU0
79	120 CONTINUE	GAU0
80	B(I, J) = SUM/A(I, I)	GAU0
81	130 CONTINUE	GAU0
82	RETURN	GAU0
83	140 CONTINUE	GAU0
84	IF (A(1, 1) .EQ. 0.) GO TO 300	GAU0
85	DO 150 J=1, NR	GAU0
86	B(1, J) = B(1, J)/A(1, 1)	GAU0
87	150 CONTINUE	GAU0
88	RETURN	GAU0
89	300 IERR = 2	GAU0
90	RETURN	GAU0
91	END	GAU0

```

0      SUBROUTINE PNCH (A,NA,NAM,IOP)
1 C IOP(1)=0, SKIP TITLE; IOP(2)=N, SKIP LINES; IOP(3)=1, TAB 25 SPACES.
2      IMPLICIT REAL*8 (A-H,O-Z)
3      DIMENSION A(1),IOP(4),NA(2)
4      NR=NA(1)
5      NC=NA(2)
6      NMAX=NR*NC
7      NSKIP=IOP(2)
8      IF (IOP(2).EQ.0) GO TO 205
9      DO 200 I=1,NSKIP
10     200 WRITE(7,150)
11     150 FORMAT(2X)
12     205 CONTINUE
13     IF (IOP(1).EQ.0) GO TO 210
14     WRITE(7,151) NAM,NR,NC
15     151 FORMAT(A4,/,2I5)
16     210 CONTINUE
17     DO 250 I=1,NR
18     IF (IOP(3).EQ.0) WRITE(7,152) (A(J),J=I,NMAX,NR)
19     IF (IOP(3).NE.0) WRITE(7,153) (A(J),J=I,NMAX,NR)
20     250 CONTINUE
21     152 FORMAT(6(1PD13.5))
22     153 FORMAT(25X,6(1PD13.5))
23     RETURN
24     END

```

0	FUNCTION DIMAG(Z)		DIMC
1	REAL*8 A(2),DIMAG		DIMC
2	COMPLEX*16 Z,B		DIMC
3	EQUIVALENCE (A,B)		DIMC
4	B=Z		DIMC
5	DIMAG=A(2)		DIMC
6	RETURN		DIMC
7	END		DIMC

0	FUNCTION DREAL(Z)	DRE00
1	REAL*8 A(2),DREAL	DRE00
2	COMPLEX*16 Z,B	DRE00
3	EQUIVALENCE (A,B)	DRE00
4	B*Z	DRE00
5	DREAL=A(1)	DRE00
6	RETURN	DRE00
7	END	DRE00

0	RLOCK DATA	MOD00
1	IMPLICIT REAL*4 (A-H,O-Z)	MOD00
2	COMMON/LINES/TITLE(10),TIL(3),NLP,LIN	MOD00
3	COMMON/FORM/FMT1(2),FMT2(2),NEPR	MOD00
4	COMMON/TOL/EPSAM,EP8M,IACM	MOD00
5	COMMON/CONV/SUMCV,RICTCV,SERCV,MAXSUM	MOD00
6	DATA LIN,NLP/1,58/	MOD00
7	DATA NEPR,FMT1/7,8H(1P7D16.,8H7) /	MOD00
8	DATA TIL/8H 0RA,8HCLS PRO,8HGRAM /	MOD00
9	DATA FMT2/8H(3X,1P7D,8H16.7) /	MOD00
10	DATA EPSAM/1.E-10/	MOD00
11	DATA EP8M/1.E-10/	MOD00
12	DATA IACM/12/	
13	DATA SUMCV/1.E-8/	MOD00
14	DATA RICTCV/1.E-8/	MOD00
15	DATA SERCV/1.E-8/	MOD00
16	DATA MAXSUM/50/	MOD00
17	END	MOD00