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/MF A01

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

List of Tables by Chapters

List of Figures by Chapters

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 [1,2]. 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[1], and two continuum-type models.[2,3] The natural (uncontrolled) dynamics of this system have been simulated. The concept of distributed modal control[1], 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.[3] 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.[3]

---

*For references cited in this report please see list of references ater 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.[4] Frequency values associated with the basic structural modes of a square plate were obtained from energy considerations based on approximate expressions developed by Warburton.[5] 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[6], 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

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 gimballed 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

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.[1,2] 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[3] using a slightly modified version of the general formulation of the dynamics of a general flexible orbiting body formulated by Santini.[4] 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[5] and also the problem of selecting control law feedback gains based on decoupling techniques and application of the linear regulator problem.[6] 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 model, and the effects of inaccurate knowledge of the control influence coefficients which lead to errors in the calculated feedback gains.[6]

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:

$$\dot{\omega}_x = \frac{I_y-I_z}{I_x}\,\omega_y\omega_z + \frac{G_{R_x}}{I_x} + \frac{T_x}{I_x} + \frac{C_x}{I_x}$$

$$\dot{\omega}_y = \frac{I_z-I_y}{I_z}\,\omega_x\omega_z + \frac{G_{R_y}}{I_y} + \frac{T_y}{I_y} + \frac{C_y}{I_y} \qquad (1)$$

$$\dot{\omega}_z = \frac{I_x-I_y}{I_z}\,\omega_x\omega_y + \frac{G_{R_z}}{I_z} + \frac{T_z}{I_z} + \frac{C_z}{I_z}$$

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:

$$\omega_x = \dot{\psi} + (\dot{\theta}+\omega_c)\sin\phi$$

$$\omega_y = (\dot{\theta}+\omega_c)\cos\phi\cos\psi + \dot{\phi}\sin\psi \qquad (2)$$

$$\omega_z = \dot{\phi}\cos\psi - (\dot{\theta}+\omega_c)\sin\psi\cos\phi$$

(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:

$$G_{R_x} = 3\omega_c^2(I_z - I_y)(-c\psi s\phi c\theta + s\psi c\theta)(s\psi s\phi c\theta + c\psi s\theta)$$

$$G_{R_y} = 3\omega_c^2(I_x - I_z)(s\psi s\phi c\theta + c\psi s\theta)c\theta c\phi \quad (3)$$

$$G_{R_z} = 3\omega_c^2(I_y - I_x)(-c\psi s\phi c\theta + s\psi s\theta)c\theta c\phi$$

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]

<u>For Case (i)</u>

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

where

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

<u>For Case (ii)</u>

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

where

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

<u>For Case (iii)</u>

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

where

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

C. Linearization

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

$$\ddot{\psi} = \omega_c\dot{\phi}\, [\frac{I_y - I_z}{I_x} - 1] - \omega_c^2\, (\frac{I_y - I_z}{I_x})\psi + \frac{T_x}{I_x} + \frac{C_x}{I_x}$$

$$\ddot{\phi} = \omega_c\dot{\psi}\, [\frac{I_x - I_y}{I_z} + 1] + 4\omega_c^2\, (\frac{I_x - I_y}{I_z})\phi + \frac{T_z}{T_z} + \frac{C_z}{I_z} \quad (5)$$

$$\ddot{\theta} = 3\omega_c^2\, (\frac{I_x - I_z}{I_y})\theta + \frac{T_y}{I_y} + \frac{C_y}{I_y}$$

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:

Case (i): $I_x = I_y$ and $I_z = 2I_x = 2I_y$

Case (ii): $I_y = I_z$ and $I_x = 2I_y = 2I_z$ (6)

Case (iii): $I_x = I_z$ and $I_y = 2I_x = 2I_z$

For small amplitude angles the generic mode equations become:

Case (i): $\ddot{A}_r + \omega_r^2 A_r = \Xi_r / M_r$

Case (ii): $\ddot{A}_r + (\omega_r^2 - 3\omega_c^2)A_r = \Xi_r / M_r$ (7)

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

D. Modelling of Point Actuators

For an actuator which can generate a force of the type

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

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

$$\bar{T} = \bar{X}\times\bar{f} \quad (9)$$

where $\bar{X} = x\hat{i} + y\hat{j} + z\hat{k}$ 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 $f_x = f_y = 0$)

$$\bar{T} = yf_z\hat{i} - xf_z\hat{j} \quad (10)$$

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

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

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

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

The generic force due to the ℓ<u>th</u> actuator on the r<u>th</u> mode is given by[3,4]

$$\Xi_r = \iint W_r(x,y)\hat{k}\cdot\delta(x - x_\ell, y - y_\ell)f_\ell(t)\hat{k}\,dx\,dy$$

$$= W_r(x_\ell, y_\ell)f_\ell(t) \quad (13)$$

where $W_r(x,y)$ is the r<u>th</u> 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<u>th</u> mode is given by

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

where x_i, y_i are the coordinates of the i<u>th</u> 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

$$\bar{f} = f_x(x,y,z,t)\,\hat{i} + f_y(x,y,z,t)\hat{j} + f_z(x,y,z,t)\hat{k} \quad (15)$$

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

The torque due to such an actuator is given by

$$\bar{T} = \bar{R} \times \bar{f} \tag{16}$$

The total torque is given by

$$\bar{T} = \int_{V} (\bar{R} \times \bar{f}) \, dxdydz \tag{17}$$

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

$$f_x(x,y,z,t) = \sum_{k=1}^{L} f_{x_k}(x,y,z) g_{x_k}(t) \tag{18}$$

The integral for the torque is then given by

$$T = \int \Big[\Big(\sum_{\ell=1}^{N} y f_{z_\ell}(x,y,z) g_{z_\ell}(t) - \sum_{m=1}^{M} f_{y_m}(x,y,z) g_{y_m}(t) \Big) \hat{i} + \Big(\sum_{k=1}^{L} z f_{x_k}(x,y,z) g_{x_k}(t) - \sum_{\ell=1}^{N} x f_{z_\ell}(x,y,z) g_{z_\ell}(t) \Big) \hat{j} + \Big(\sum_{m=1}^{N} x f_{y_m}(x,y,z) g_{y_m}(t) - \sum_{k=1}^{L} f_{x_k}(x,y,z) g_{x_k}(t) \Big) \hat{k} \Big] \times dxdydz \tag{19}$$

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

$$F_r = \int [W_r(x,y)] \Big[\sum_{\ell=1} f_{z_\ell}(x,y,z) g_{z_\ell}(t) \Big] dxdydz \tag{20}$$

3. Uncontrolled Motion-Numerical Example

The platform is assumed to have the following physical properties:

$a = 100$ m (side of square plate)

$M = 276800$ kg

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

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

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

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

The modal frequencies of the elastic modes have been obtained using a finite element computer algorithm.[7] For the first three flexible modes:

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

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

$\omega_3 = 3.9088122 \times 10^{-2}$ 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.

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)

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

$$\phi'' = [(I_x - I_y + I_z)/\omega_c I_z]\psi' + 4[(I_x - I_y)/I_z]\phi \tag{22}$$

$$\theta'' = 3\theta \tag{23}$$

The generic mode equations become:

$$Z_r'' = -(\omega_r/\omega_c)^2 Z_r \tag{24}$$

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,

$$\psi'' = -(2/\omega_c)\phi' + \psi \tag{25}$$

$$\phi'' = \psi'/\omega_c \tag{26}$$

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.

Case (ii): $I_y = I_z$ and $I_x = 2I_y$

The rotational equations of motion are

$$\psi'' = -(1/\omega_c)\phi' \tag{27}$$

$$\phi'' = (2/\omega_c)\psi' + 4\phi \tag{28}$$

$$\theta'' = 3\theta \tag{29}$$

The generic mode equations can be represented by,

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

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:

$$s^2(s^2 - 4 + 2/\omega_c^2) = 0 \tag{31}$$

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

Case (iii): $I_x = I_z$ and $I_y = 2I_x = 2I_z$

The rotational equations of motion are

$$\psi'' = -\psi; \quad \phi'' = -4\phi; \quad \theta'' = 0 \tag{32}$$

while the generic mode equations can be expressed by,

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

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 :

$$X' = AX + BU \tag{34}$$

where the state vector, X, is defined as

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

and

$$x_1 = \phi;\ x_2 = \psi;\ x_3 = \theta;\ x_{i+3} = Z_i = \Delta_i/a;$$

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

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

$$x_{n+6+i} = Z_i' = x_{i+3}',\ i = 1,2,\ldots n$$

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

$$[A] = \left[\begin{array}{c|cccc} \bigcirc_{6x6} & & [I]_{6x6} & & \\ \hline \text{diagonal} & 0 & A_{78} & \ldots & 0 \\ A_{6x6} & A_{78} & 0 & \ldots & 0 \\ & 0 & 0 & \ldots & 0 \end{array}\right] \tag{35}$$

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

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

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

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

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

The general B matrix:

$$B = \begin{bmatrix} O_{6x6} \\ \hline B_{6x6} \end{bmatrix}$$

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

$$\ddot{\chi} = A\dot{\chi} + C\chi + BU \tag{36}$$

where $\chi = (x_1, x_2 \ldots x_{n+3})$

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

$$\ddot{\chi} = (A + BK_r)\dot{\chi} + (C + BK_p)\chi \tag{37}$$

K_r and K_p are evaluated such that $(A+BK_r)$ 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

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

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

Second Location a = 100m

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

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

Case (ii)

First Location a = 100m

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

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

Second Location a = 100m

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

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

Case (iii)

First Location a = 100m

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

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

Second Location a = 100m

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

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

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 listedas follows.

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

The non zero elements of the A matrix are:

$A_{i,i+6}=1$ for $i = 1,\ldots 6$; $A_{7,8} = 800$;

$A_{8,7} = -1600$; $A_{8,2} = 1$; $A_{9,3} = -3$;

$A_{10,4} = -277.414$, $A_{11,5} = -588.647$;

$A_{12,6} = -976.844$

Location I (Lower Part of B matrix)

0.0	0.0	0.0	0.0	0.0	-0.22643
-0.4329	-0.4529	0.0	0.0	0.4520	0.0
0.4529	-0.4529	0.4529	-0.4529	0.4529	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 (ii) are:

$A_{7,1} = 4$; $A_{7,8} = 1600$; $A_{8,2} = 0$; $A_{8,7} = -800$;

$A_{9,3} = 3.$

Location I (Lower Part of B matrix)

0.4592	0.4529	0.0	£.0	-0.4529	0.0
0.0	0.0	0.0	0.0	0.0	0.22645
-0.4529	0.4529	-0.4529	0.4529	-0.4529	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 (ii) are:

$A_{7,1} = -4$; $A_{7,8} = 0$; $A_{8,2} = -1$; $A_{8,7} = 0$;

$A_{9,3} = 0.$

Location I (Lower Part of B matrix)

-0.4529	-0.4529	0.0	0.0	0.4529	0.0
0.4529	-0.4529	-0.4529	-0.4529	0.4529	0.0
0.0	0.0	0.0	0.0	0.0	-0.22645
0.003126	-0.003126	0.0	0.0	-0.003126	0.0
0.0	0.0	-0.0030844	-0.0030844	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

$$\dot{X} = AX+BU \quad (38), \quad X \rightarrow 2(n+3)x1$$

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

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

The control, $U = -KX$ is selected by using a digital computer algorithm[8] 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 i) is desired. Three of the actuators are assumed to provide forces perpendicular to the major surface 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 $a = 100$m. 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)

$$\begin{bmatrix} 0.0 & 0.0 & 0.0 & -0.22645 \\ -0.4529 & -0.4529 & 0.0 & 0.0 \\ 0.4529 & -0.4529 & 0.0 & 0.0 \\ 0.003126 & -0.003126 & 0.0 & 0.0 \\ 0.0 & 0.0 & -0.0030844 & 0.0 \\ -0.008786 & -0.008786 & -0.0115 & 0.0 \end{bmatrix}$$

<u>Closed Loop Poles</u> (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 (i) 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) <u>Application of the Linear Regulator Theory</u>

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

$$J = \int_0^\infty (X^T QX + U^T RU)dt \tag{39}$$

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[8] 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 Ricatti 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 (i) 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

$$\begin{bmatrix} \omega_z' \\ \omega_x' \end{bmatrix} = \begin{bmatrix} 0 & a \\ -b & 0 \end{bmatrix} \begin{bmatrix} \omega_z \\ \omega_x \end{bmatrix} \tag{40}$$

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

so that a strategy for selecting the elements of Q could be[9,10]

$$Q = \begin{bmatrix} f & 0 \\ 0 & f \end{bmatrix} \quad \text{where } f = \Omega^2/\omega^2_{T_{max}} \quad . \tag{41}$$

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

$$\begin{bmatrix} z_i' \\ z_i'' \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -(\omega_i/\omega_c)^2 & 0 \end{bmatrix} \begin{bmatrix} z_i \\ z_i' \end{bmatrix} \tag{42}$$

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 (i) with actuator locations as given in Section (b)) is obtained using the relations given by Eq. (41) and is a diagonal matrix, Q_o, with the following elements:

$Q_{1,1} = 4.324x10^9$, $Q_{2,2} = 8.539x10^9$,

$Q_{3,3} = Q_{9,9} = 3.0x10^4$, $Q_{4,4} = Q_{10,10} =$

$2.77414x10^6$, $Q_{5,5} = Q_{11,11} = 5.88647x10^6$,

$Q_{6,6} = Q_{12,12} = 9.74844x10^6$, $Q_{7,7} = Q_{8,8} =$

$2.222x10^3$.

The R_o matrix is chosen as an identity matrix. A parametric study is done using various multiples of the Q_o ($Q=\alpha Q_o$) and R_o 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, $Q=\alpha Q_o$, 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):

-1.0043, $-1.8\pm i16.64$, $-2.16\pm i24.20$, $-17.18\pm i19.79$, -26.23, -36.22, -137.64 and $-38.66\pm i1132.11$

The maximum force amplitudes as shown in Fig. 6 are less than those corresponding to Case (i) - 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 draw back 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).

5. 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. Meirovitch), 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. XXIV, 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.

Appendix A - Nomenclature

A_r	r^{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^{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_r, K_p	Rate and position feedback gain matrices
M_r	r^{th} modal mass
T	Torque due to an actuator
T_x, T_y, T_z	Torque components
$W_r(x,y)$	r^{th} modal shape function
Z_r	Nondimensionalized r^{th} modal amplitude function
ω_c	Orbital frequency
$\omega_x, \omega_y, \omega_z$	Angular body rates
ϕ, ψ, θ	Roll, yaw, pitch, respectively
$\omega_1, \omega_2, \omega_3$	First three modal frequencies of the plate

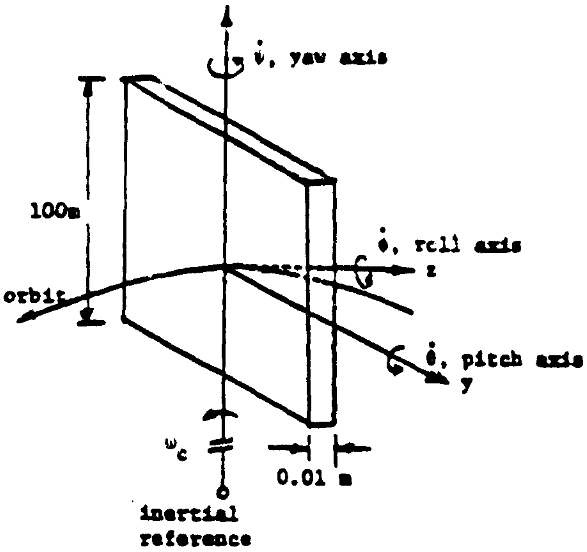


Fig. 1a. Platform following local vertical with major surface normal to the orbit plane- Case (i)

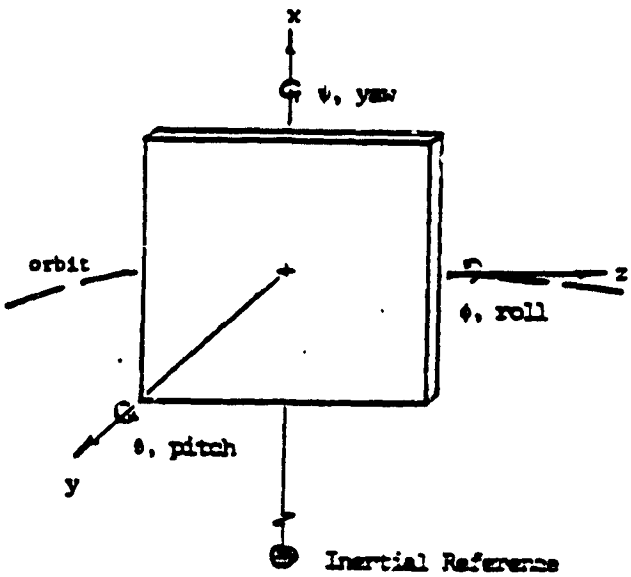


Fig. 1c. Platform following local vertical with major surface in the orbit plane - Case (iii)

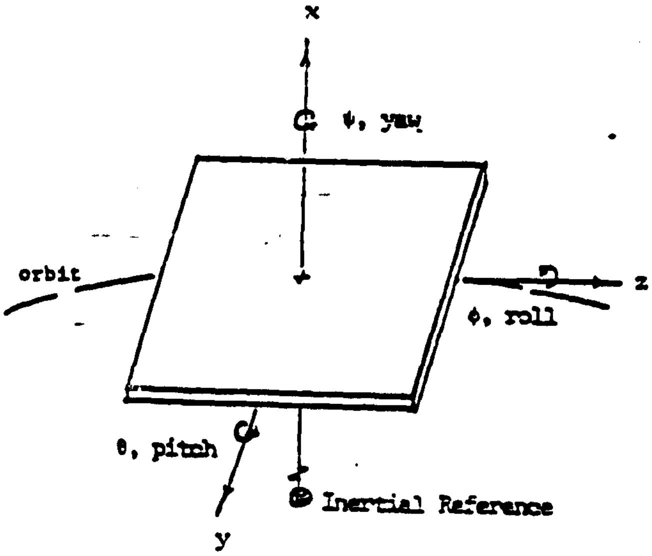


Fig. 1b. Platform along local horizontal - Case (ii)

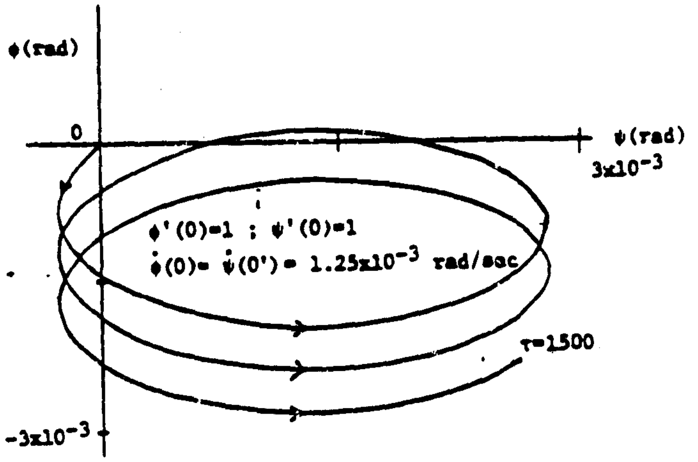
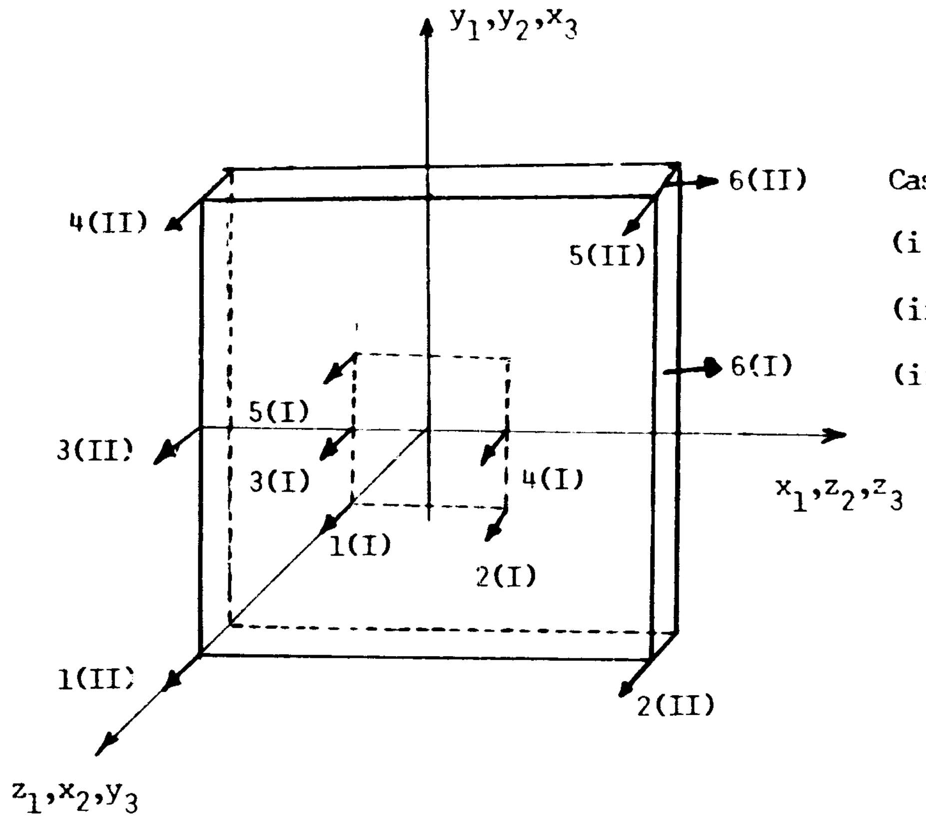


Fig. 2. Roll/yaw motion (uncontrolled) - Case (i)



Case		Coord. Axes
(i)	-	x_1, y_1, z_1
(ii)	-	x_2, y_2, z_2
(iii)	-	x_3, y_3, z_3

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

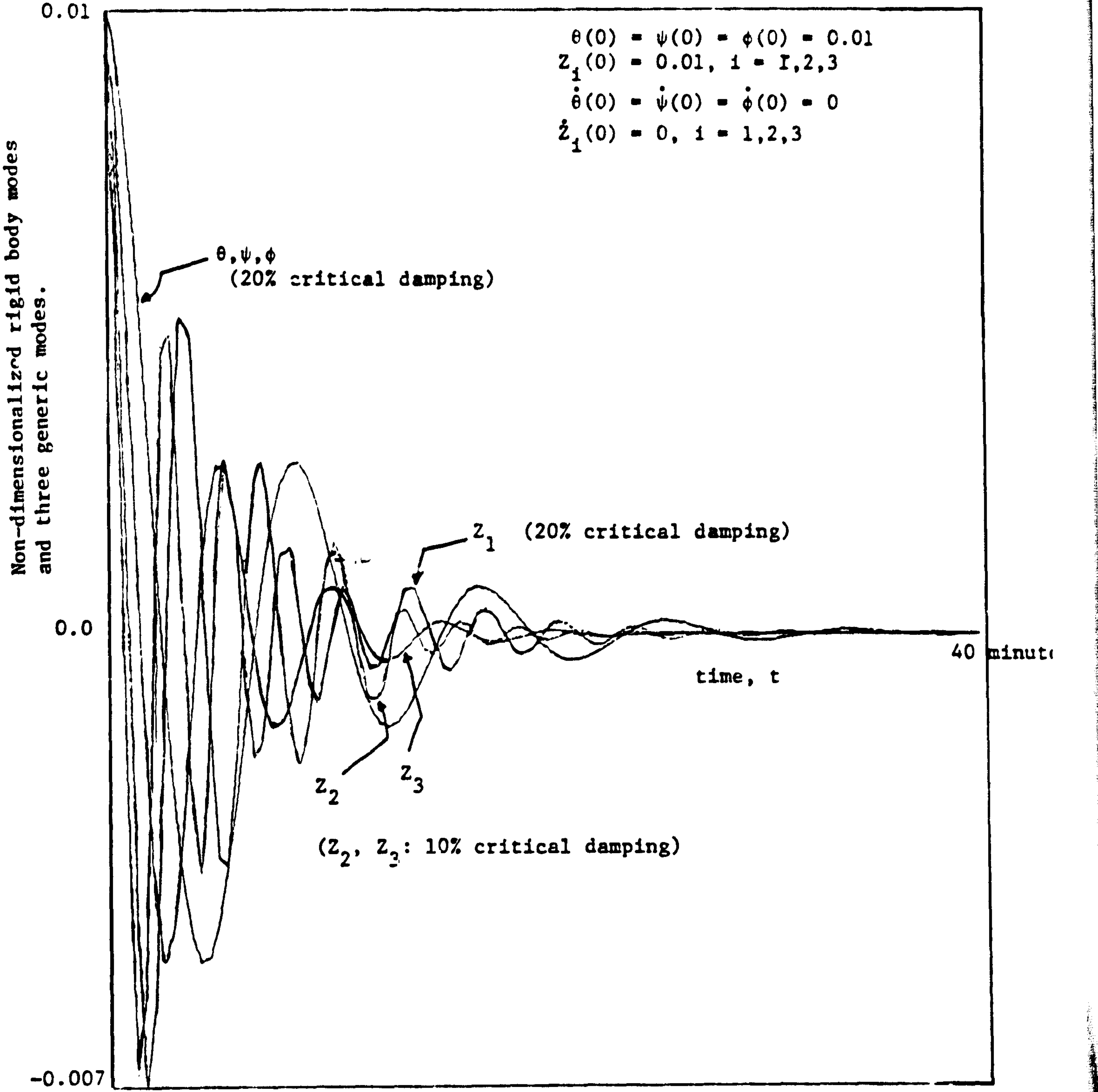


Fig 4a. Controlled state response for all combinations of orientations and actuator locations.

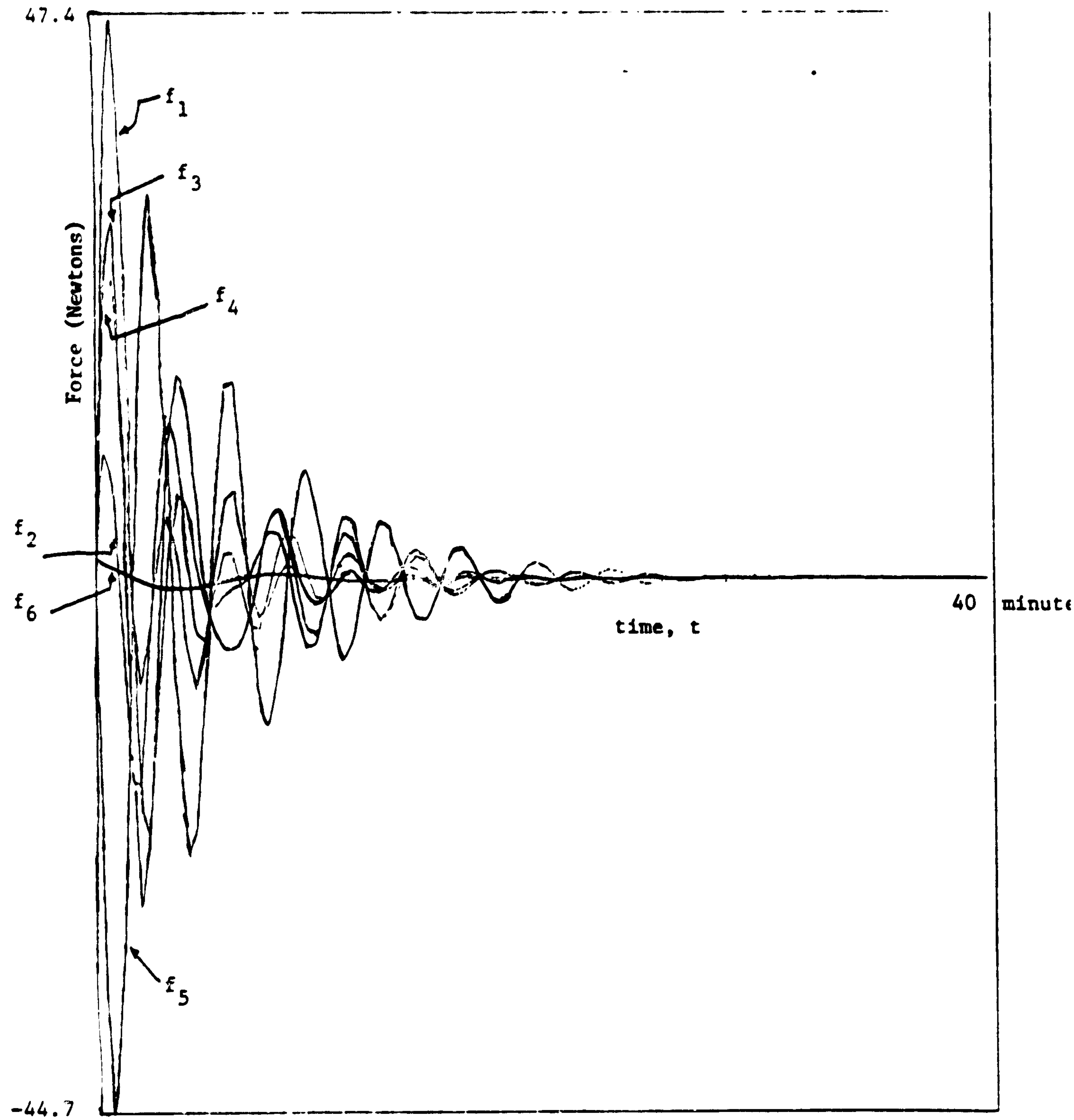


Fig 4b. Control force time history for Case (iii) - II

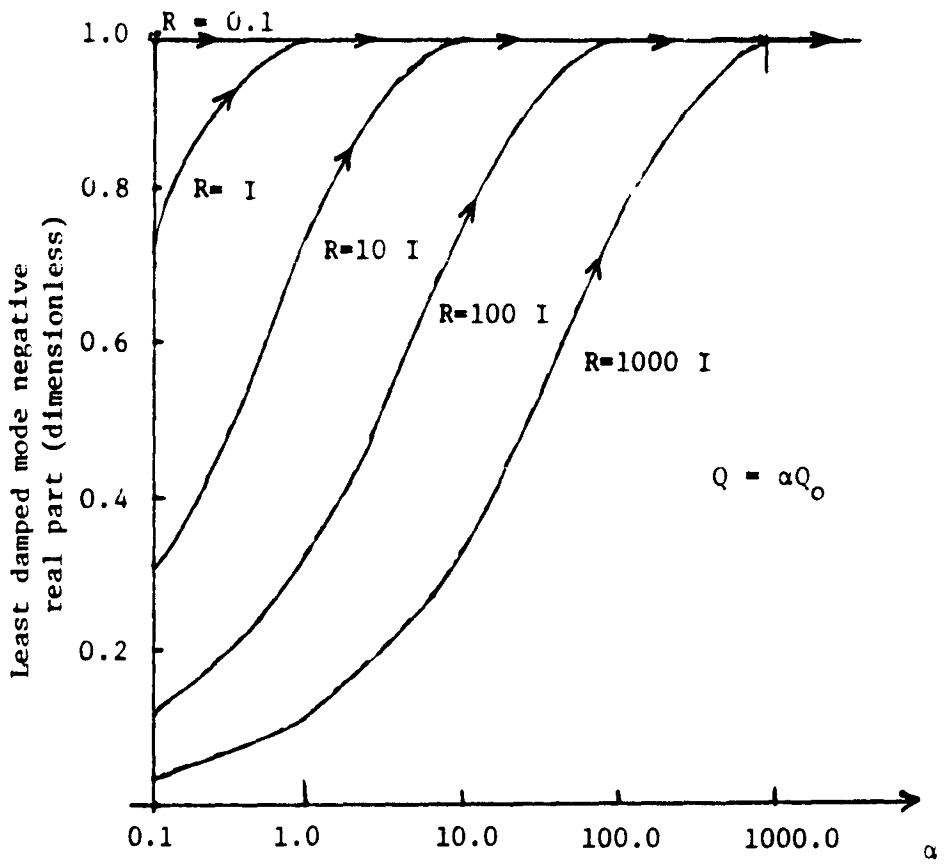


Fig. 5. Variation of least damped mode negative real part with α and R.

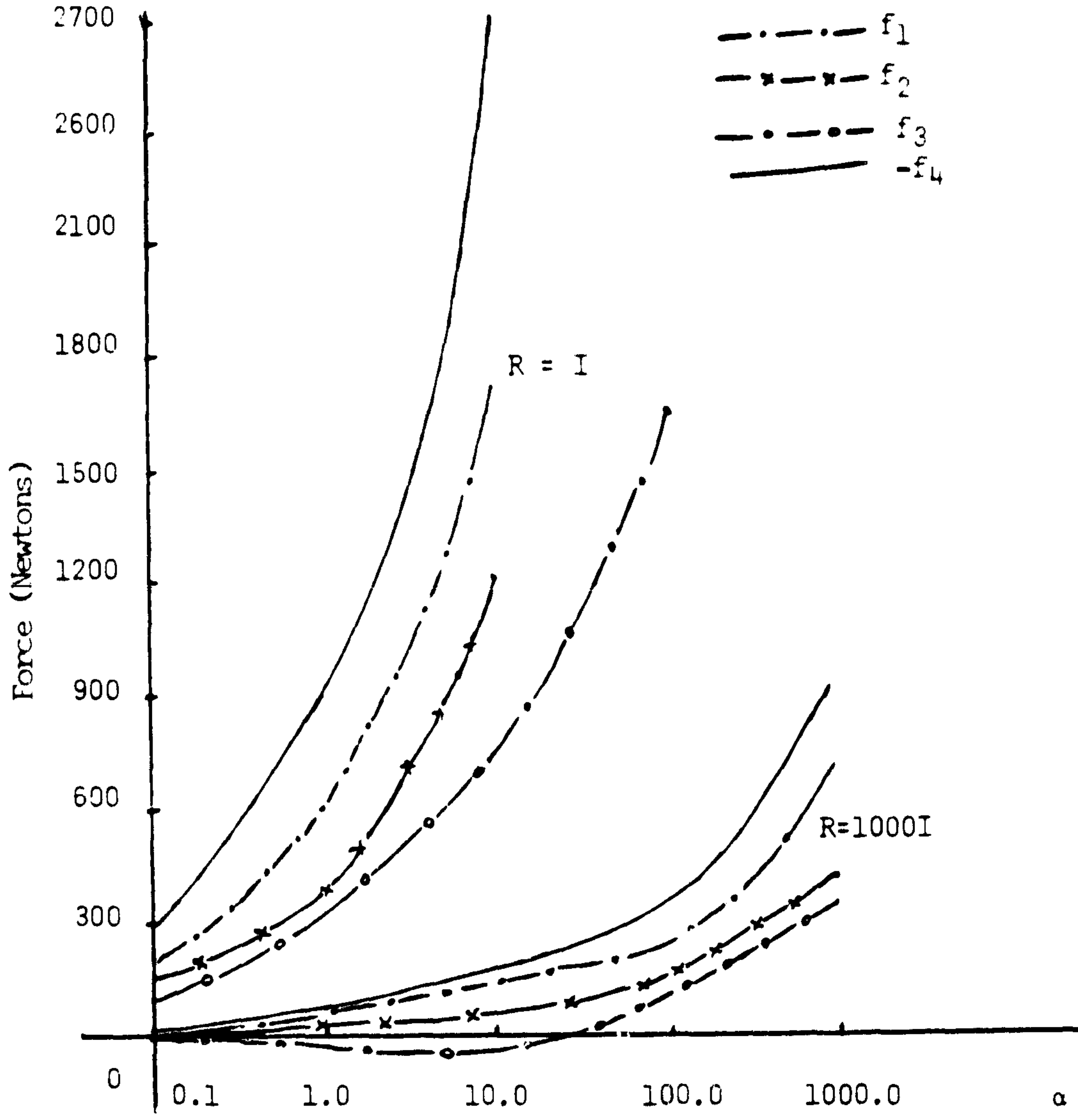


Fig 6. Maximum force amplitudes as a function of α and R for all actuators - (application of linear regulator theory).

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[1] 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[2] 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[3] 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.[4] 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

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

where ρ, σ 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

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

$\theta(x)$ 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[1]

$$\lambda^2 = \frac{\rho a^4 (2\pi f)^2\ 12(1-\sigma^2)}{\pi^4\ Eh^2 g} \tag{III-3}$$

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

where λ 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

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

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 λ 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,o) + (o,m), m is even

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

Modes (m,1) ± (1,m), for m = 3,5,7,.... (III-6)

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

$$\pm\, 2\,\sigma(m-\tfrac{1}{2})^2 \frac{24}{\pi^2} \left[1 - \frac{2}{(m-\tfrac{1}{2})\pi}\right]^2 \pm 2(1-\sigma) \frac{192}{\pi^4}$$

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[2]

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,

$$W = \Sigma\, A_{m,n}\;\; \theta_m(x)\; \phi_n(y) \qquad \text{(III-7)}$$

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

$$\theta_m(x) = \frac{\cosh k_m \cos k_m x + \cos k_m \cosh k_m \bar{x}}{\sqrt{\cosh^2 k_m + \cos^2 k_m}} \quad \text{(m even)}$$

(III-8)

$$= \frac{\sinh k_m \sin k_m x + \sin k_m \sinh k_m x}{\sqrt{\sinh^2 k_m - \sin^2 k_m}} \quad \text{(m odd)}$$

$\phi_n(y)$ 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

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

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

which result from the spatial boundary conditions. Further, it was shown by an energy principle that [2,4]

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

where U_{max} is the maximum potential energy due to bending. The co-efficients, A_{mn}, in Eq. (III-7) are determined to make ω^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 co-efficents, A_{mn}, are tabulated in Ref. 2. As an example the expression for the first mode is given here.

$$W(x,y) = x_1y_1 + 0.0325\,(x_1y_3 + x_3y_1) - .005\,x_3y_3$$
$$- .00257\,(x_1y_5 + x_5y_1) + .00121\,(x_3y_5 + x_5y_3)$$
$$- .000365\,x_5y_5 + \ldots.$$

and

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

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 palte 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.

density = 2768.0 kg/m^3

Young's modulus = 0.7441×10^{10} kg/m^2

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.[2] 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 Lemeke'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 numer 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 shpaes 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 ⋮	$(n-\frac{1}{2})$	$(n-\frac{1}{2})^2 P$	$(n-\frac{1}{2})^2 Q$

$P = [1-2/(m-0.5)\pi]$ $\qquad$ $Q = [1+6/(m-0.5)\pi]$

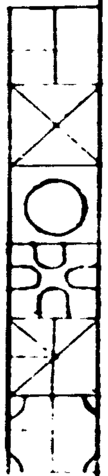
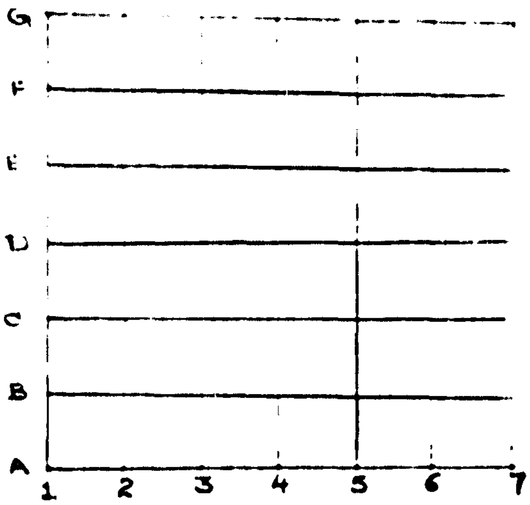
	STRUDL: NUMBER OF ELEMENTS 4	9	16	36	64	Nodal Pattern	WARBURTON Freq. (cps)	(m,n)	Nodal Pattern	LEMKE Freq. (cps)	Nodal Pattern
1	.003165	.003284	.003315	.003314	.003358		.003471	(1,1)		.003287	
2	.004291	.004664	.004776	.004839	.004856		.004846	(2,0)-(0,2)		.004831	
3	.005549	.006133	.006221	.006221	.006207		.006175	(2,0)+(0,2)		.006175	
4	.007619	.008453	.008632	.008690	.008692		.00900	(2,1)		.015744	
5	.007619	.008453	.008632	.008690	.008692		.00900	(1,2)		.017167	
6	.01323	.01512	.01600	.01596	.01579		.01542	(3,0)		.01944	
7		.01512	.01600	.01596	.01579		.01542	(0,3)			
8		.01584	.01618	.01615	.01607		.01656	(2,2)			
9		.01630	.01727	.01750	.01746		.01715	(3,1)-(1,3)			
10		.01843	.02022	.02031	.02005		.0208	(1,3)+(3,1)			
11		.02465	.02693	.02693	.02700		.02726	(3,2)			
12		.02465	.02673	.02693	.02700		.02726	(2,3)			
13		.03362	.03029	.03162	.03104		0.03941	(3,3)			

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

Case 1: 4 Elements

LOCATION	STRUDL	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	STRUDL	LEMKE	WARBURTON	STRUDL	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	[illegible]

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

Case 4: 36 Elements

LOCATION	STRUDL	WARBURTON	LEMKE	LOCATION	STRUDL	WARBURTON	LEMKE
A1	0	0	0	C_1	-.8590	-.8524	-.9793
A_2	.4874	.4733	.5346	C_2	-.3595	-.3792	-.3447
A3	.8590	.8524	.9793	C_3	0	0	0
A4	1.0	1.0	1.0	C_4	.1334	.1475	.1207
B1	-.4573	-.4733	-.5346	D_1	-1.0	-1.0	-1.0
B2	0	0	0	D_2	-.4938	-.5267	-.4654
B3	.3594	.3792	.3447	D_3	-.1334	-.1475	-.1207
B4	.4933	.5267	.4654	D_4	0	0	0

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

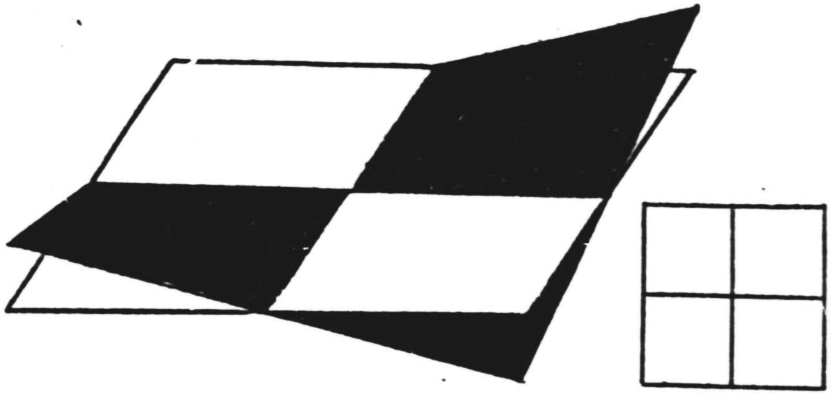


Fig. 1 First Mode

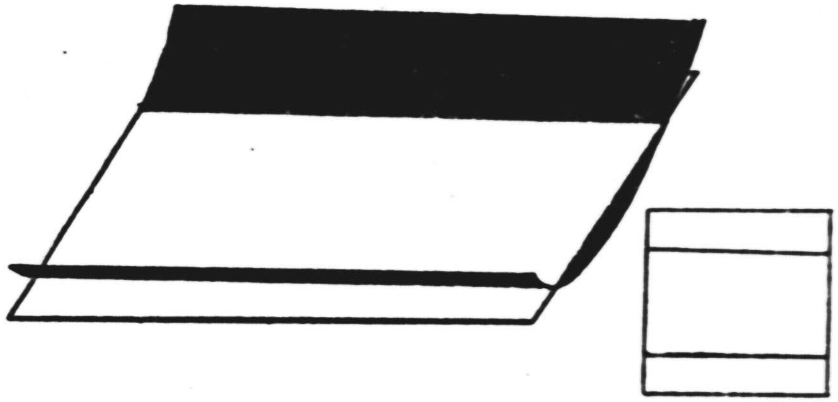


Fig. 3(a) (2,0) Mode

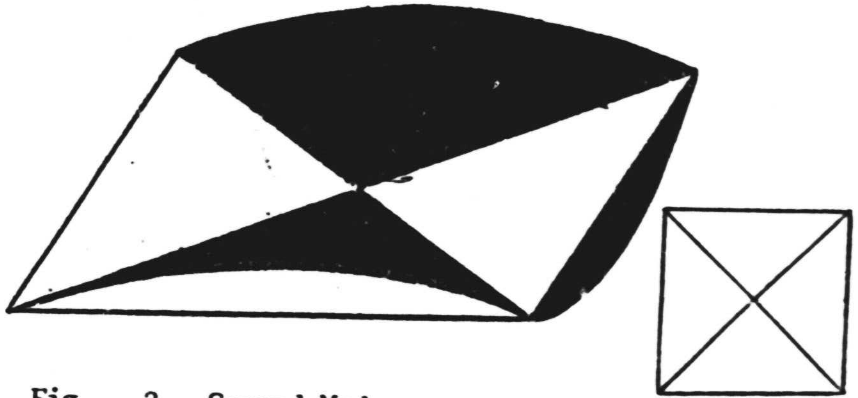


Fig. 2 Second Mode

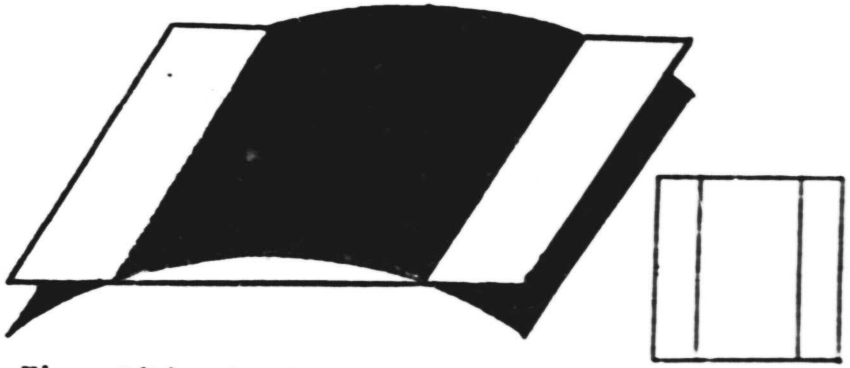


Fig 3(b) (0,2) Mode

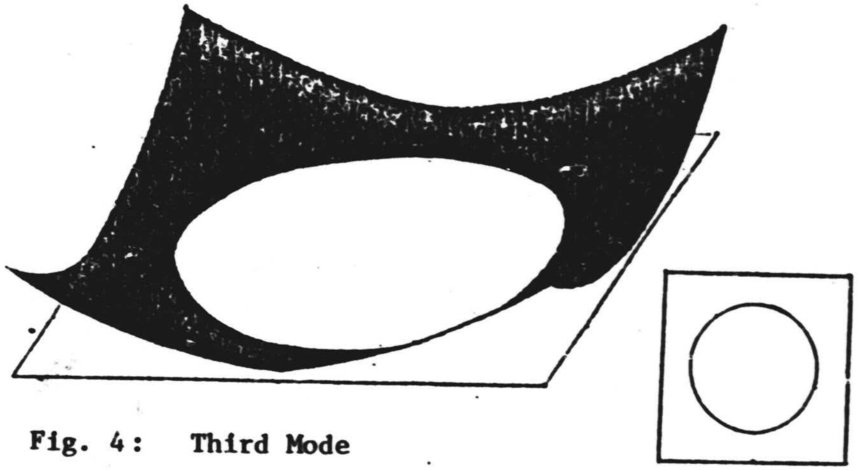


Fig. 4: Third Mode

(2,0) + (0,2)

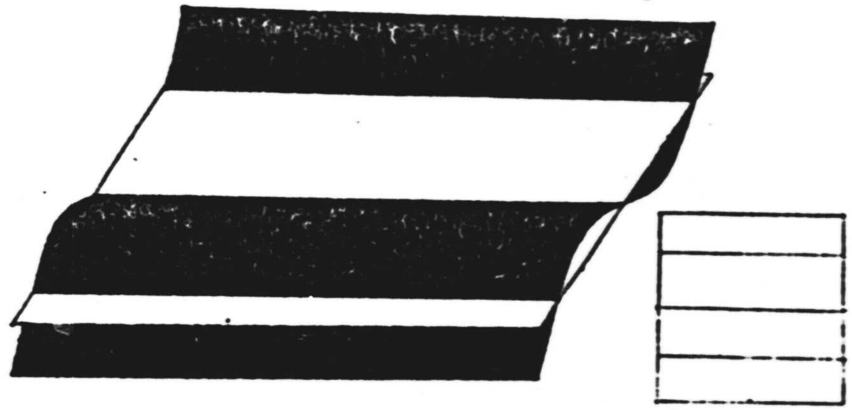


Fig. 6: Sixth Mode

(3,0)

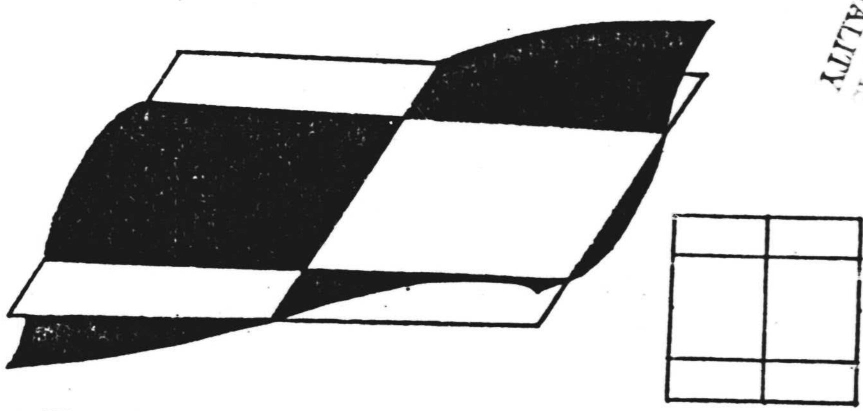


Fig. 5: Fourth Mode

(2,1)

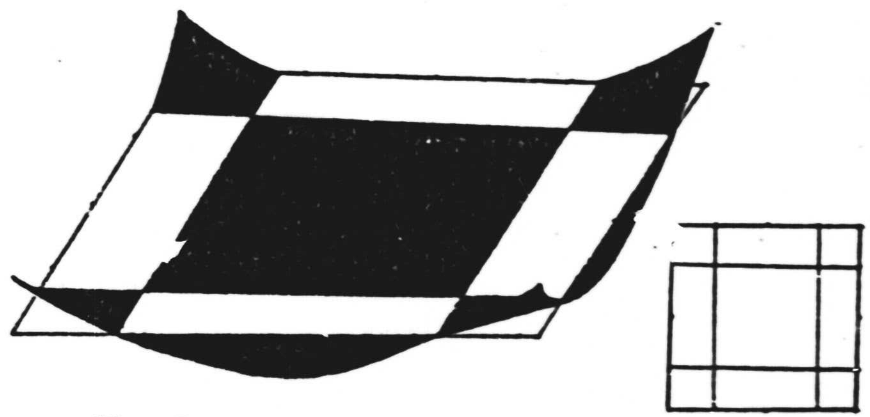


Fig. 7: Eighth Mode

(2,2)

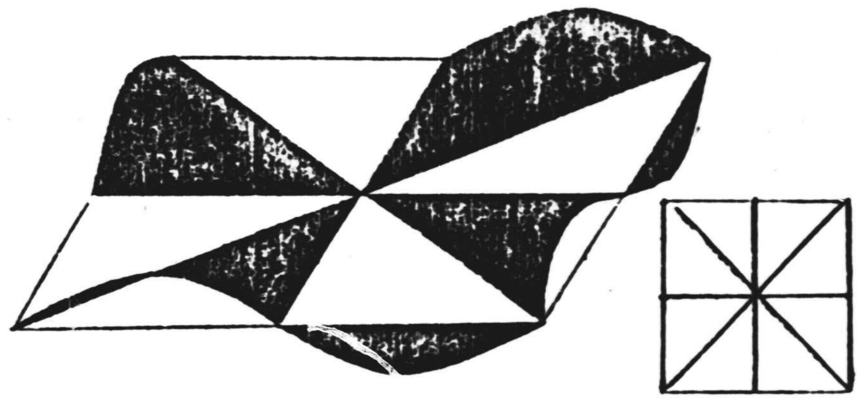


Fig. 8: Ninth Mode (1,3)-(3,1)

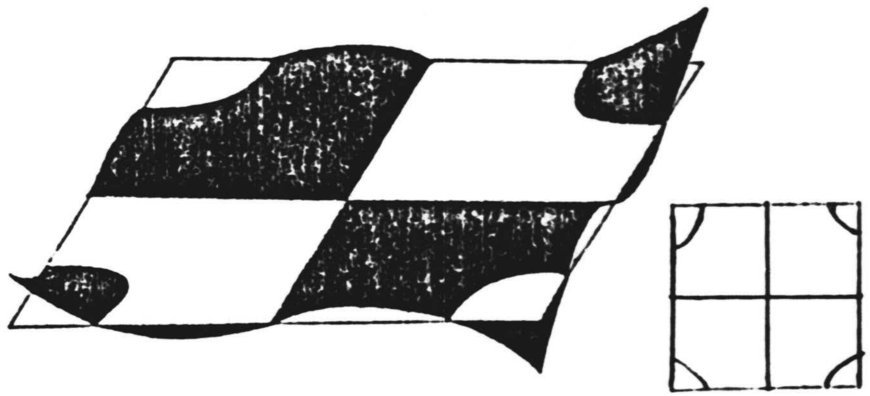


Fig. 9: Tenth Mode (1,3)+(3,1)

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[8] 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:

	Old Name	New Name
(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 ommitted.

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

```
      SUBROUTINE LNCNT  (N)
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/LINES/TITLE(10),TIL(3),NLP,LIN
      LIN=LIN+N
      IF  (LIN.LE.NLP)   GO TO 20
      WRITE(6,1010) TITLE,TIL
 1010 FORMAT(1H1,10A8,3A8/)
      LIN=2+N
      IF  (N.GT.NLP)  LIN=2
   20 RETURN
      END
```

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

```
      SUBROUTINE  PRNT(A,NA,NAM,IOP)                                    PRN00010
      IMPLICIT REAL*8 (A-H,O-Z)                                         PRN00020
      DIMENSION A(1),NA(2)                                              PRN00030
      COMMON  /FORM/FMT1(2),FMT2(2),NEPR                                PRN00040
      COMMON/LINES/TITLE(10),TIL(3),NLP,LIN                             PRN00050
C- NOTE NLP NO. LINES/PAGE VARIES WITH THE INSTALLATION.                PRN00060
      DATA KZ,KW,KB /1H0,1H1,1H /                                       PRN00070
      NAME = NAM                                                        PRN00080
      II = IOP                                                          PRN00090
      NR = NA(1)                                                        PRN00100
      NC = NA(2)                                                        PRN00110
      NLST = NR * NC                                                    PRN00120
      IF( NLST .LT. 1 .OR. NR .LT. 1 ) GO TO 16                         PRN00130
      IF(NAME  .EQ. 0) NAME = KB                                        PRN00140
C- SKIP HEADLINE IF REQUESTED.                                          PRN00150
      GO TO (11,10,132,12),      II                                     PRN00160
   10 CALL LNCNT(100)                                                   PRN00170
   11 CALL LNCNT(2)                                                     PRN00180
    3 WRITE(6,177) KZ,NAME,NR,NC                                        PRN00190
  177 FORMAT(A1,5X,A4,8H  MATRIX,5X,I3,5H ROWS,5X,I3,8H COLUMNS)        PRN00200
      GO TO 13                                                          PRN00210
   12 CALL LNCNT(100)                                                   PRN00220
      GO TO 13                                                          PRN00230
  132 CALL LNCNT(2)                                                     PRN00240
      WRITE (6,891)                                                     PRN00250
  891 FORMAT (1H0)                                                      PRN00260
C- BELOW COMPUTE NR OF LINES/ ROW --DECIDE IF 1 EXTRA BLANK LINE        PRN00270
   13 J=(NC-1)/NEPR+1                                                   PRN00280
C- WHY ALWAYS ADD 1 LINE- BECAUSE IF MULTIPLE, USE 1 BLK LINE EXTRA.    PRN00290
      NLPW=J
      JST=1
C     COMPUTE LAST ROW POSITION -1 BELOW
      NLST = NLST -NR                                                   PRN00330
      MN=NC                                                             PRN00340
      IF  (NC.GT.NEPR)   MN=NEPR                                        PRN00350
      KLST=NR*(MN-1)                                                    PRN00360
   91 CONTINUE                                                          PRN00370
      DO 912 J = JST, NR                                                PRN00380
      CALL LNCNT(NLPW)                                                  PRN00390
      KLST = KLST +1                                                    PRN00400
      WRITE (6,FMT1) (A(N), N=J,KLST,NR)                                PRN00410
      IF  (NC.LE.NEPR)   GO TO 912                                      PRN00420
      NLST = NLST +1                                                    PRN00430
      KNR=KLST+NR                                                       PRN00440
      WRITE (6,FMT2) (A(N), N=KNR,NLST,NR)                              PRN00450
  912 CONTINUE                                                          PRN00460
      RETURN                                                            PRN00470
   16 CALL  LNCNT(1)
      WRITE (6,916) NAM,NA                                              PRN00490
  916 FORMAT  (' ERROR IN PRNT  MATRIX ',A4,' HAS NA=',2I6)             PRN00500
      RETURN                                                            PRN0051
      END                                                               PRN0052
```


ORIGINAL PAGE IS
OF POOR QUALITY

```
      SUBROUTINE EQUATE(A,NA,B,NB)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(1),B(1),NA(2),NB(2)
      NB(1) = NA(1)
      NB(2) =NA(2)
      L=NA(1)*NA(2)
      IF( NA(1) .LT. 1 .OR. L .LT. 1 ) GO TO 999
      DO 300 I=1,L
  300 B(I)=A(I)
 1000 RETURN
  999 CALL LNCNT (1)
      WRITE  (6,50)  NA
   50 FORMAT  (' DIMENSION ERROR IN EQUATE  NA=',2I6)
      RETURN
      END
```

```
      SUBROUTINE TRANP(A,NA,B,NB)                                       TRA00010
      IMPLICIT REAL*8 (A-H,O-Z)                                         TRA00020
      DIMENSION A(1),B(1),NA(2),NB(2)                                   TRA00030
      NB(1)=NA(2)                                                       TRA00040
      NB(2)=NA(1)                                                       TRA00050
      NR=NA(1)                                                          TRA00060
      NC=NA(2)                                                          TRA00070
      L=NR*NC                                                           TRA00080
      IF( NR .LT. 1 .OR. L .LT. 1  )  GO TO 999                         TRA00090
      IR=0                                                              TRA00100
      DO 300 I=1,NR                                                     TRA00110
      IJ=I-NR                                                           TRA00120
      DO 300 J=1,NC                                                     TRA00130
      IJ=IJ+NR                                                          TRA00140
      IR=IR+1                                                           TRA00150
  300 B(IR)=A(IJ)                                                       TRA00160
      RETURN                                                            TRA00170
  999 CALL LNCNT(1)                                                     TRA00180
      WRITE  (6,50)  NA                                                 TRA00190
   50 FORMAT  (' DIMENSION ERROR IN TRANP   NA=',2I6)                   TRA00200
      RETURN                                                            TRA00210
      END                                                               TRA00220
```

```
      SUBROUTINE SCALE (A, NA, B, NB, S)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(1),B(1),NA(2),NB(2)
      NB(1) = NA(1)
      NB(2) =NA(2)
      L = NA(1)*NA(2)
      IF( NA(1) .LT. 1 .OR. L .LT. 1 ) GO TO 999
      DO 300 I=1,L
  300 B(I)=A(I)*S
 1000 RETURN
  999 CALL LNCNT(1)
      WRITE  (6,50) NA
   50 FORMAT  (' DIMENSION ERROR IN SCALE   NA=',2I6)
      RETURN
      END
```

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

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

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

```
      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
      IF( (NA(1) .NE. NB(1)) .OR. (NA(2) .NE. NB(2)) )  GO TO 999       ADD00040
      NC(1)=NA(1)                                                       ADD00050
      NC(2)=NA(2)                                                       ADD00060
      L=NA(1)*NA(2)                                                     ADD00070
      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
      GO TO 1000                                                        ADD00110
  999 CALL LNCNT (1)                                                    ADD00120
      WRITE(6,50) NA,NB                                                 ADD00130
   50 FORMAT  (' DIMENSION ERROR IN ADD     NA=',2I6,5X,'NB=',2I6)      ADD00140
 1000 RETURN                                                            ADD00150
      END                                                               ADD00160
```

```
      SUBROUTINE SUBT(A,NA,B,NB,C,NC)                                   SUB00010
      IMPLICIT REAL*8 (A-H,O-Z)                                         SUB00020
      DIMENSION A(1),B(1),C(1),NA(2),NB(2),NC(2)                        SUB00030
      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
      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
  300 C(I)=A(I)-B(I)                                                    SUB00100
      GO TO 1000                                                        SUB00110
  999 CALL LNCNT (1)                                                    SUB00120
      WRITE(6,50) NA,NB                                                 SUB00130
   50 FORMAT (' DIMENSION ERROR IN SUBT    NA=',2I6,5X,'NB=',2I6)       SUB00140
 1000 RETURN                                                            SUB00150
      END                                                               SUB00160
```

```
      SUBROUTINE MULT(A,NA,B,NB,C,NC)                                   MUL0001
      IMPLICIT REAL*8 (A-H,O-Z)                                         MUL0002
      DIMENSION A(1),B(1),C(1),NA(2),NB(2),NC(2)                        MUL0003
      NC(1) = NA(1)                                                     MUL0004
      NC(2) = NB(2)                                                     MUL0005
      IF(NA(2).NE.NB(1)) GO TO 999                                      MUL0006
      NAR = NA(1)                                                       MUL0007
      NAC = NA(2)                                                       MUL0008
      NBC = NB(2)                                                       MUL0009
      NAA=NAR*NAC                                                       MUL0010
      NBB=NAR*NBC                                                       MUL0011
      IF ( NAR .LT. 1 .OR. NAA .LT. 1 .OR. NBB .LT. 1 ) GO TO 999       MUL0012
      IR = 0                                                            MUL0013
      IK=-NAC                                                           MUL0014
      DO 350 K=1,NBC                                                    MUL0015
      IK = IK + NAC                                                     MUL0016
      DO 350 J=1,NAR                                                    MUL0017
      IR=IR+1                                                           MUL0018
      IB=IK                                                             MUL0019
      JI=J-NAR                                                          MUL0020
      V1=0.0                                                            MUL0021
      DO 300 I=1,NAC                                                    MUL0022
      JI = JI + NAR                                                     MUL0023
      IB=IB+1                                                           MUL0024
      V3=A(JI)                                                          MUL0025
      V4=B(IB)                                                          MUL0026
      V2=V3*V4                                                          MUL0027
      V1=V1+V2                                                          MUL0028
  300 CONTINUE                                                          MUL0029
      C(IR)=V1                                                          MUL0030
  350 CONTINUE                                                          MUL0031
      GO TO 1000                                                        MUL0032
  999 CALL LNCNT (1)                                                    MUL003
      WRITE(6,500) (NA(I),I=1,2),(NB(I),I=1,2)                          MUL0034
  500 FORMAT  (' DIMENSION ERROR IN MULT    NA=',2I6,5X,'NB=',2I6)      MUL0035
 1000 RETURN                                                            MUL0036
      END                                                               MUL003
```

```
      SUBROUTINE MAXEL(A,NA,ELMAX)                                      MAX0001
      IMPLICIT REAL*8 (A-H,O-Z)                                         MAX0002
      DIMENSION A(1),NA(2)                                              MAX0003
C                                                                       MAX0004
      N = NA(1)*NA(2)                                                   MAX0005
C                                                                       MAX0006
      ELMAX = DABS( A(1))                                               MAX0007
      DO 100 I = 2,N                                                    MAX0008
      ELMAXI = DABS( A(I) )                                             MAX0009
      IF( ELMAXI.GT.ELMAX)  ELMAX=ELMAXI
  100 CONTINUE                                                          MAX0011
C                                                                       MAX0012
      RETURN                                                            MAX0013
      END                                                               MAX0014
```

```
      SUBROUTINE NORMS(MAXROW,M,N,A,IOPT,RLNORM)                        NOR0001
      IMPLICIT REAL*8 (A-H,O-Z)                                         NOR0002
      DIMENSION A(1)                                                    NOR0003
C                                                                       NOR0004
C  INITIALIZATION                                                       NOR0005
C                                                                       NOR0006
      RLNORM=0.                                                         NOR0007
      SUM=0.                                                            NOR0008
      I=-MAXROW                                                         NOR0009
C                                                                       NOR0010
C  TRANSFER TO APPROPRIATE LOOP TO COMPUTE THE DESIRED NORM             NOR0011
C                                                                       NOR0012
      IF(IOPT-2)5,20,30                                                 NOR0013
C                                                                       NOR0014
C  THIS LOOP COMPUTES THE ONE-NORM                                      NOR0015
C                                                                       NOR0016
    5 DO 15 K=1,N                                                       NOR0017
      I=I+MAXROW                                                        NOR0018
      DO 10 J=1,M                                                       NOR0019
      L=I+J                                                             NOR0020
   10 SUM=DABS(A(L))+SUM                                                NOR0021
      IF(SUM.GT.RLNORM)RLNORM=SUM                                       NOR0022
   15 SUM=0.                                                            NOR0023
      RETURN                                                            NOR0024
C                                                                       NOR0025
C  THIS LOOP COMPUTES THE EUCLIDEAN NORM                                NOR0026
C                                                                       NOR0027
   20 DO 25 K=1,N                                                       NOR0028
      I=I+MAXROW                                                        NOR0029
      DO 25 J=1,M                                                       NOR003
      L=I+J                                                             NOR003
      SUM=A(L)                                                          NOR003
   25 RLNORM=SUM*SUM+RLNORM                                             NOR003
      RLNORM=DSQRT(RLNORM)                                              NOR003
      RETURN                                                            NOR003
C                                                                       NOR003
C  THIS LOOP COMPUTES THE INFINITY-NORM                                 NOR003
C                                                                       NOR003
   30 DO 40 J=1,M                                                       NOR003
      L=I+J                                                             NOR004
      DO 35 K=1,N                                                       NOR004
      L=L+MAXROW                                                        NOR004
   35 SUM=DABS(A(L))+SUM                                                NOR004
      IF(SUM.GT.RLNORM)RLNORM=SUM                                       NOR004
   40 SUM=0.0
      RETURN                                                            NOR004
      END                                                               NOR004
```

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

```
      SUBROUTINE JUXTR(A,NA,B,NB,C,NC)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(1),B(1),C(1),NA(2),NB(2),NC(2)
      IF(NA(2).NE.NB(2))GO TO 600
      NC(2)=NA(2)
      NC(1)=NA(1)+NB(1)
      L=NA(1)*NA(2)
      IF( NA(1) .LT. 1 .OR. L .LT. 1 ) GO TO 600
      IF( NC(2) .LT. 1 ) GO TO 600
      MCA=NA(2)
      MRA=NA(1)
      MRB=NB(1)
      MRC=NC(1)
      DO 10 I=1,MCA
      DO 10 J=1,MRA
      K=J+MRA*(I-1)
      L=J+MRC*(I-1)
   10 C(L)=A(K)
      DO 20 I=1,MCA
      DO 20 J=1,MRB
      K=J+MRB*(I-1)
      L=MRA+J+MRC*(I-1)
   20 C(L)=B(K)
      RETURN
  600 CALL LNCNT(1)
      WRITE(6,1600) NA,NB
 1600 FORMAT(' DIMENSION ERROR IN JUXTR,  NA=',2I6,5X,'NB=',2I6)
      RETURN
      END
```

```
      SUBROUTINE FACTOR(Q,NQ,D,ND,IOP,IAC,DUMMY)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION Q(1),D(1),DUMMY(1)
      DIMENSION NQ(2),ND(2),NDUM(2)
      IOPT = 2
      N = NQ(1)
      M = N**2
      N1 = M + 1
      N2 = N1 + N
C
      CALL EQUATE(Q,NQ,DUMMY,NQ)
      CALL SNVDEC(IOPT,N,N,N,N,DUMMY,NOS,B,IAC,ZTEST,DUMMY(N1),D,IRANK,A
     1PLUS,IERR)
      IF( IERR .EQ. 0 ) GO TO 200
      CALL LNCNT(5)
      IF( IERR .GT. 0 ) PRINT 100,IERR
      IF( IERR .EQ. -1) PRINT 150,ZTEST,IRANK
  100 FORMAT(//,' IN FACTOR , SNVDEC HAS FAILED TO CONVERGE TO THE ',I4,
     1' SINGULAR VALUE AFTER 30 ITERATIONS')
  150 FORMAT(//,' IN FACTOR, THE MATRIX Q SUBMITTED TO SNVDEC IS CLOSE T
     1O A MATRIX OF LOWER RANK USING ZTEST = ',D16.8,/,' IF THE ACCURACY
     2 IS REDUCED  THE RANK MAY ALSO BE REDUCED',/,' CURRENT RANK =',I4)
      NDUM(1)=N
      NDUM(2)=1
      IF(IERR .EQ. -1)  CALL PRNT(DUMMY(N1),NDUM,4HSNVL,1)
      IF( IERR .GT. 0 ) RETURN
C
  200 CONTINUE
      NDUM(1) = N
C
      DO 250 J =1,N
      M1 = (J-1)*N + 1
      M2 = J*N
      DO 250 I =M1,M2
      K = N2+I-1
      L = N1+J-1
      IF( DUMMY(L) .EQ. 0.0) GO TO 300
      DUMMY(K) = DSQRT(DUMMY(L))*DUMMY(I)
  250 CONTINUE
      NDUM(2)=N
      GO TO 350
C
  300 NDUM(2) = J - 1
  350 CONTINUE
      IF( DUMMY(N2) .LT. 0.0 ) CALL SCALE(DUMMY(N2),NDUM,DUMMY(N2),NDUM,
     1-1.0)
      CALL TRANP(DUMMY(N2),NDUM,D,ND)
C
      IF( IOP .EQ. 0 ) RETURN
      CALL LNCNT(4)
      PRINT 400
  400 FORMAT(//,' FACTOR Q AS (D TRANSPOSE)XD ',/)
      CALL PRNT(Q,NQ,4H Q  ,1)
      CALL PRNT(D,ND,4H D  ,1)
      CALL MULT(DUMMY(N2),NDUM,D,ND,DUMMY,NQ)
      CALL PRNT(DUMMY,NQ,4HDTXD,1)
C
      RETURN
      END
```

```
      SUBROUTINE EIGEN(MAX, N, A, ER, EI, ISV, ILV, V, WK, IERR)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(MAX,N),ER(N),EI(N),V(MAX,1),WK(N,1)
      INTEGER INT(20)
      LOGICAL*1 SELECT(25)
C
C     PRELIMINARY REDUCTION
C
      CALL BALANC (MAX,N,A,LOW,IGH,WK)
      CALL ELMHES (MAX,N,LOW,IGH,A,INT(1))
      IV = ISV + ILV
      IF (IV .NE. 0) GO TO 10
C
C     COMPUTE ALL EIGENVALUES AND NO EIGENVECTORS
C
      CALL HQR (MAX,N,LOW,IGH,A,ER,EI,IERR)
      IF (IERR .NE. 0) GO TO 260
      DO 5 I=1,N
         WK(I,1) = ER(I)
         WK(I,2) = EI(I)
         WK(I,3) = ER(I)**2 + EI(I)**2
    5 CONTINUE
      IC = 0
      GO TO 190
   10 CONTINUE
C
C     SAVE A MATRIX FOR INVERSE ITERATION AND INITIALIZE WK(I,4)
C       ARRAY WHICH WILL BE A LOGICAL ARRAY IN CALLED SUBROUTINES
C
      DO 20 I=1,N
      SELECT(I)=.FALSE.
         JS = 1
         IF (I .GE. 3) JS = I-1
         DO 20 J=JS,N
            WK(I,J+5) = A(I,J)
   20 CONTINUE
C
C     COMPUTE ALL EIGENVALUES (UNORDERED)
C
      CALL HQR (N,N,LOW,IGH,WK(1,6),ER,EI,IERR)
      IF (IERR .NE. 0) GO TO 260
      DO 30 I=1,N
         WK(I,3) = ER(I)**2 + EI(I)**2
   30 CONTINUE
      IF (ILV .EQ. 0) GO TO 60
C
C     FIND LARGEST ILV EIGENVALUES AND FLAG THEM
C
      DO 50 I=1,ILV
         P = -1.D0
         DO 40 J=1,N
            IF (WK(J,3) .LE. P) GO TO 40
            K = J
            P = WK(J,3)
   40    CONTINUE
      SELECT(K)=.TRUE.
         WK(K,3) = -WK(K,3)
   50 CONTINUE
      IF (EI(K) .EQ. 0.) GO TO 60
      IF (EI(K) .GT. 0.) GO TO 55
      IF (SELECT(K-1))GO TO 60
      ILV = ILV+1
      SELECT(K-1)=.TRUE.
```

```
      GO TO 60
   55 CONTINUE
      IF (.NOT.SELECT(K+1)) ILV = ILV+1
   60 CONTINUE
      IF (ISV .EQ. 0) GO TO 90
C
C     FIND SMALLEST ISV EIGENVALUES AND FLAG THEM
C
      DO 65 I=1,N
         WK(I,3) = DABS(WK(I,3))
   65 CONTINUE
      DO 80 I=1,ISV
         P = 1.D74
         DO 70 J=1,N
            IF (WK(J,3) .GE. P) GO TO 70
            K = J
            P = WK(J,3)
   70    CONTINUE
      SELECT(K)=.TRUE.
         WK(K,3) = 1.D74
   80 CONTINUE
      IF (EI(K) .EQ. 0.) GO TO 90
      IF (EI(K) .GT. 0.) GO TO 85
      IF (SELECT(K-1)) GO TO 90
      ISV = ISV+1
      SELECT(K-1)=.TRUE.
      GO TO 90
   85 CONTINUE
      IF (.NOT.SELECT(K+1)) ISV = ISV+1
   90 CONTINUE
C
C     FIND EIGENVECTORS FOR FLAGGED EIGENVALUES
C
      CALL INVIT (MAX,N,A,ER,EI,SELECT,N,M,V,IERR,WK(1,6),WK(1,3),
     1                 WK(1,5))
C
C     BACK TRANSFORM EIGENVECTORS TO ORIGINAL MATRIX
C
      CALL ELMBAK (MAX,LOW,IGH,A,INT(1) ,M,V)
      CALL BALBAK (MAX,N,LOW,IGH,WK,M,V)
C
C     SEPARATE FLAGGED EIGENVALUES FROM UNFLAGGED EIGENVALUES
C
      IV = ISV + ILV
      IF (IV .LE. N) GO TO 100
      ILV = N-ISV
      IV = N
  100 CONTINUE
      IC = 0
      JC = IV
      DO 150 I=1,N
         IF (SELECT(I)) GO TO 120
         IF (EI(I) .GE. 0.) GO TO 110
         IF (SELECT(I-1)) GO TO 120
  110    CONTINUE
         JC = JC+1
         WK(JC,1) = ER(I)
         WK(JC,2) = EI(I)
         KC = JC
         GO TO 130
  120    CONTINUE
         IC = IC+1
         WK(IC,1) = ER(I)
```

```
          WK(IC,2) = EI(I)
          KC = IC
  130     CONTINUE
          WK(KC,3) = ER(I)**2 + EI(I)**2
  150 CONTINUE
C
C     NORMALIZE VECTORS TO UNIT LENGTH AND STORE FOR REORDERING
C
      J = 0
  151 CONTINUE
      J = J+1
      IF (WK(J,2) .NE. 0.) GO TO 154
      SUM = 0.
      DO 152 I=1,N
          SUM = SUM + V(I,J)**2
  152 CONTINUE
      IF (SUM .EQ. 0.) GO TO 158
      SUM = DSQRT(SUM)
      DO 153 I=1,N
          WK(I,J+4) = V(I,J)/SUM
  153 CONTINUE
      GO TO 158
  154 CONTINUE
      JP1 = J+1
      SUM = 0.
      DO 155 I=1,N
          SUM = SUM + V(I,J)**2 + V(I,JP1)**2
  155 CONTINUE
      IF (SUM .EQ. 0.) GO TO 157
      SUM = DSQRT(SUM)
      DO 156 I=1,N
          WK(I,J+4) = V(I,J)/SUM
          WK(I,J+5) = V(I,JP1)/SUM
  156 CONTINUE
  157 CONTINUE
      J = JP1
  158 CONTINUE
      IF (J .LT. IV) GO TO 151
      IC = 0
      LC = 0
      IF (ISV .EQ. 0) GO TO 190
C
C     ORDER SMALLEST ISV EIGENVALUES AND EIGENVECTORS FOR OUTPUT
C
      DO 190 I=1,ISV
         P = 1.D74
         DO 160 J=1,IV
            IF (WK(J,3) .GE. P) GO TO 160
            K = J
            P = WK(J,3)
  160    CONTINUE
         IC = IC+1
         LC = LC+1
         ER(IC) = WK(K,1)
         EI(IC) = WK(K,2)
         DO 170 J=1,N
            V(J,LC) = WK(J,K+4)
  170    CONTINUE
         WK(K,3) = 1.D74
  180 CONTINUE
  190 CONTINUE
      IF (IV .EQ. N) GO TO 220
C
```

```
C
C     ORDER UNFLAGGED EIGENVALUES FOR OUTPUT
C
      IV1 = IV+1
      IUF = N - IV
      DO 210 I=1,IUF
         P = 1.074
         DO 200 J=IV1,N
            IF (WK(J,3) .GE. P) GO TO 200
            K = J
            P = WK(J,3)
  200    CONTINUE
         IC = IC+1
         ER(IC) = WK(K,1)
         EI(IC) = WK(K,2)
         WK(K,3) = 1.074
  210 CONTINUE
  220 CONTINUE
      IF (ILV .EQ. 0) GO TO 260
C
C     ORDER LARGEST ILV EIGENVALUES AND EIGENVECTORS FOR OUTPUT
C
      DO 250 I=1,ILV
         P = 1.074
         DO 230 J=1,IV
            IF (WK(J,3) .GE. P) GO TO 230
            K = J
            P = WK(J,3)
  230    CONTINUE
         IC = IC+1
         LC = LC+1
         ER(IC) = WK(K,1)
         EI(IC) = WK(K,2)
         DO 240 J=1,N
            V(J,LC) = WK(J,K+4)
  240    CONTINUE
         WK(K,3) = 1.074
  250 CONTINUE
  260 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE SYMPOS (MAXN,N,A,NRHS,B,IOPT,IFAC,DETERM,ISCALE,P,IERR)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(MAXN,N),B(MAXN,NRHS),P(N)
C
      DATA R1,R2/1.0D+75,1.0D-75/
C
C              TEST FOR A SCALAR MATRIX (IF COEFFICIENT MATRIX IS A
C              SCALAR-- SOLVE  AND COMPUTE DETERMINANT IF DESIRED)
      IERR = 0
      NM1 = N-1
      IF (NM1 .GT. 0) GO TO 20
C
      IF( A(1,1) .LE. 0.0 )  IERR = 1
      ISCALE = 0
      DETERM = A(1,1)
      P(1) = 1.0/A(1,1)
      DO 10 J=1,NRHS
      B(1,J) = B(1,J)/DETERM
   10 CONTINUE
      RETURN
C
C              TEST TO DETERMINE IF CHOLESKY DECOMPOSITION OF COEFFICIENT
C              MATRIX IS DESIRED
   20 IF (IFAC .EQ. 1) GO TO 160
C
C              INITIALIZE DETERMINANT EVALUATION PARAMETERS
      DETERM=1.0
      ISCALE=0
C
C               'LOOP' TO PERFORM CHOLESKY DECOMPOSTION ON THE COEF-
C              FICIENT MATRIX A (I.E. MATRIX A WILL BE DECOMPOSED INTO
C              THE PRODUCT OF A UNIT LOWER TRIANGULAR MATRIX (L), A
C              DIAGONAL MATRIX (D), AND THE TRANSPOSE OF L (LTRANSPOSE).)
C
   30 DO 150 I=1,N
      IM1 = I-1
C
      DO 150 J=1,I
      X = A(J,I)
C
C              DETERMINE IF ELEMENTS ARE ABOVE OR BELOW THE DIAGONAL
      IF (I .GT. J) GO TO 110
C
C              USING THE DIAGONAL ELEMENTS OF MATRIX A, THIS SECTION
C              COMPUTES DIAGONAL MATRIX AND DETERMINES IF MATRIX A IS
C              SYMMETRIC POSITIVE DEFINITE
      IF (IM1 .EQ. 0) GO TO 50
      DO 40 K=1,IM1
      Y = A(I,K)
      A(I,K) = Y*P(K)
      X = X - Y*A(I,K)
   40 CONTINUE
C
C              TEST IF COEFFICIENT MATRIX IS POSITIVE DEFINITE
   50 IF( X .LE. 0.0 )  IERR = 1
C
C              COMPUTE INVERSE OF DIAGONAL MATRIX D**-1 = 1/P
      P(I) = 1.0 / X
C
C              TEST TO SEE IF DETERMINANT IS TO BE EVALUATED
      IF (IOPT .EQ. 0) GO TO 150
C
C
```

```
C                   SCALE THE DETERMINANT (COMPUTE THE DETERMINANT EVALUATION
C                   PARAMETERS DETERM AND ISCALE)
      PIVOTI=X
   60 IF(DABS(DETERM).LT.R1) GO TO 70
      DETERM = DETERM*R2
      ISCALE = ISCALE+1
      GO TO 60
   70 IF(DABS(DETERM).GT.R2) GO TO 80
      DETERM = DETERM*R1
      ISCALE = ISCALE-1
      GO TO 70
   80 IF(DABS(PIVOTI).LT.R1) GO TO 90
      PIVOTI = PIVOTI*R2
      ISCALE = ISCALE+1
      GO TO 80
   90 IF(DABS(PIVOTI).GT.R2) GO TO 100
      PIVOTI = PIVOTI*R1
      ISCALE = ISCALE-1
      GO TO 90
  100 DETERM = DETERM*PIVOTI
      GO TO 150
C
C
C                   USING THE LOWER TRIANGULAR ELEMENTS OF MATRIX A, THIS
C                   SECTION COMPUTES THE UNIT LOWER TRIANGULAR MATRIX
  110 JM1 = J-1
      IF (JM1 .EQ. 0) GO TO 140
      DO 120 K=1,JM1
      X = X - A(I,K)*A(J,K)
  120 CONTINUE
C
  140 A(I,J) = X
C
  150 CONTINUE
C
C                   SECTION TO APPLY BACK SUBSTITUTION TO SOLVE L*Y = B FOR
C                   UNIT LOWER TRIANGULAR MATRIX AND CONSTANT COLUMN VECTOR B
C
  160 IF( IFAC .EQ. 2 )  RETURN
      DO 180 I=2,N
      IM1=I-1
C
      DO 180 J=1,NRHS
      X = B(I,J)
C
      DO 170 K=1,IM1
      X = X - A(I,K)*B(K,J)
  170 CONTINUE
C
      B(I,J) = X
  180 CONTINUE
C
C                   SECTION TO SOLVE (LTRANSPOSE)*X = (D**-1)*Y FOR TRANSPOSE
C                   OF UNIT LOWER TRIANGULAR MATRIX AND INVERSE OF DIAGONAL
C                   MATRIX
C
      Y = P(N)
      DO 190 J=1,NRHS
      B(N,J) = B(N,J)*Y
  190 CONTINUE
C
  200 I = NM1+1
      Y = P(NM1)
```

```
C
      DO 220 J=1,NRHS
      X = B(NM1,J)*Y
C
      DO 210 K=I,N
      X = X - A(K,NM1)*B(K,J)
  210 CONTINUE
C
      B(NM1,J) = X
  220 CONTINUE
C
C
C             TEST TO DETERMINE IF SOLUTIONS HAVE BEEN DETERMINED FOR
C             ALL COLUMN VECTORS
      NM1 = NM1-1
      IF (NM1 .GT. 0) GO TO 200
C
      RETURN
      END
```

```
      SUBROUTINE GELIM(NMAX,N,A,NRHS,B,IPIVOT,IFAC,WK,IERR)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(NMAX,1),B(NMAX,1),IPIVOT(1),WK(1)
      IERR=0
C
C     TEST FOR L/U FACTORIZATION
C
      IF(IFAC.EQ.1)GO TO 10
      CALL DETFAC(NMAX,N,A,IPIVOT,IFAC,DETERM,ISCALE,WK,IERR)
      IF(IERR.GT.0)RETURN
      IF (IFAC.EQ.2) DETA=DETERM*(10.**(100*ISCALE))
   10 NM1=N-1
C
C     TEST FOR SCALAR A MATRIX
C
      IF(NM1.GT.0)GO TO 40
      IF(A(1,1).EQ.0.)GO TO 30
      DO 20 I=1,NRHS
   20 B(1,I)=B(1,I)/A(1,1)
      IF (IFAC.EQ.2) WK(1)=DETA
      RETURN
   30 IERR=1
      RETURN
C
   40 DO 100 M=1,NRHS
C
C     PIVOT THE M-TH COLUMN OF B MATRIX
C
      DO 50 I=1,NM1
      KI=IPIVOT(I)
      P=B(KI,M)
      B(KI,M)=B(I,M)
   50 B(I,M)=P
C
C     FORWARD SUBSTITUTION
C
      WK(1)=B(1,M)
C
      DO 70 I=2,N
      IM1=I-1
      P=0.0
      DO 60 K=1,IM1
   60 P=P+A(I,K)*WK(K)
   70 WK(I)=B(I,M)-P
C
C     BACK SUBSTITUTION
C
      B(N,M)=WK(N)/A(N,N)
C
      DO 90 J=1,NM1
      I=N-J
      IP1=I+1
      P=WK(I)
      DO 80 K=IP1,N
   80 P=P-A(I,K)*B(K,M)
   90 B(I,M)=P/A(I,I)
C
  100 CONTINUE
      IF (IFAC.EQ.2) WK(1)=DETA
      RETURN
      END
```

```
      SUBROUTINE SNVDEC(IOP,MD,ND,M,N,A,NOS,B,IAC,ZTEST,Q,V,IRANK,APLUS,SNV000
     1IERR)                                                             SNV000
      IMPLICIT REAL*8 (A-H,O-Z)                                         SNV000
      LOGICAL WITHU,WITHV                                               SNV000
      DIMENSION A(MD,N),V(ND,N),Q(N),E(150)                             SNV000
      DIMENSION B(MD,NOS),APLUS(ND,M)                                   SNV000
C                                                                       SNV000
C                                                                       SNV000
C     TEST FOR  SCALAR OR VECTOR A                                      SNV000
C                                                                       SNV001
      IF( N .GE. 2 ) GO TO 3000                                         SNV001
C                                                                       SNV001
      IERR = 0                                                          SNV001
      ZTEST = 10.**(-IAC)                                               SNV001
      SUM = 0.0                                                         SNV001
      DO 1000 I=1,M                                                     SNV001
      SUM = SUM + A(I,1)*A(I,1)                                         SNV001
 1000 CONTINUE                                                          SNV001
      SUM = DSQRT(SUM)                                                  SNV001
      IRANK = 0                                                         SNV002
      IF( SUM .GT. ZTEST ) IRANK = 1                                    SNV002
      Q(1) = SUM                                                        SNV002
C                                                                       SNV002
      IF( IOP .EQ. 1) RETURN                                            SNV002
      V(1,1) = 1.0                                                      SNV002
      IF( IRANK .EQ. 0 ) GO TO 1200                                     SNV002
      DO 1100 I =1,M                                                    SNV002
      A(I,1) = A(I,1)/SUM                                               SNV002
 1100 CONTINUE                                                          SNV002
      GO TO 1300                                                        SNV003
 1200 CONTINUE                                                          SNV003
      A(1,1) = 1.0                                                      SNV003
 1300 CONTINUE                                                          SNV003
C                                                                       SNV003
      IF( IOP .EQ. 2 ) RETURN                                           SNV003
      IF( IOP .EQ. 4 ) GO TO 1850                                       SNV003
      IF( IRANK .EQ. 0 ) GO TO 1600                                     SNV003
      DO 1500 J = 1,NOS                                                 SNV003
      Z = 0                                                             SNV003
      DO 1400 I = 1,M                                                   SNV004
      Z = Z + A(I,1)*B(I,J)/SUM                                         SNV004
 1400 CONTINUE                                                          SNV004
      B(1,J) = Z                                                        SNV004
 1500 CONTINUE                                                          SNV004
      GO TO 1800                                                        SNV004
 1600 CONTINUE                                                          SNV004
      DO 1700 J =1,NOS                                                  SNV004
      B(1,J) = 0.0                                                      SNV004
 1700 CONTINUE                                                          SNV004
 1800 CONTINUE                                                          SNV005
C                                                                       SNV005
      IF( IOP .EQ. 3 ) RETURN                                           SNV005
 1850 CONTINUE                                                          SNV005
      IF( IRANK .EQ. 0 ) GO TO 2000                                     SNV005
      DO 1900 I =1,M                                                    SNV005
      APLUS(1,I) = A(I,1)/SUM                                           SNV005
 1900 CONTINUE                                                          SNV005
      RETURN                                                            SNV005
 2000 CONTINUE                                                          SNV005
      DO 2100 I=1,M                                                     SNV006
      APLUS(1,I) = 0.0                                                  SNV006
 2100 CONTINUE                                                          SNV006
      RETURN                                                            SNV006
```

```
      C                                                                  SNV006
      C                                                                  SNV006
 3000 CONTINUE                                                           SNV006
      C                                                                  SNV006
      C                                                                  SNV006
      C                                                                  SNV006
      TOL=1.0D-60                                                        SNV007
      SIZE=0.0                                                           SNV007
      NP1=N+1                                                            SNV007
      C                                                                  SNV007
      C     COMPUTE THE E-NORM OF MATRIX A AS ZERO TEST FOR SINGULAR VALUES. SNV007
      C                                                                  SNV007
      SUM=0.0                                                            SNV007
      DO 500 I=1,M                                                       SNV007
      DO 500 J=1,N                                                       SNV007
  500 SUM = SUM + A(I,J)**2                                              SNV007
      ZTEST = DSQRT(SUM)                                                 SNV008
      ZTEST = ZTEST*10.**(-IAC)                                          SNV008
      C                                                                  SNV008
  510 IF (IOP.NE.1 ) GO TO 515                                           SNV008
      WITHU=.FALSE.                                                      SNV008
      WITHV=.FALSE.                                                      SNV008
      GO TO 520                                                          SNV008
  515 WITHU=.TRUE.                                                       SNV008
      WITHV=.TRUE.                                                       SNV008
  520 CONTINUE                                                           SNV008
      G = 0.0                                                            SNV009
      X = 0.0                                                            SNV009
      DO 30 I = 1,N                                                      SNV009
      C                                                                  SNV009
      C     HOUSEHOLDER REDUCTION TO BIDIAGONAL FORM.                    SNV009
      C                                                                  SNV009
      E(I) = G                                                           SNV009
      S = 0.0                                                            SNV009
      L = I+1                                                            SNV009
      C                                                                  SNV009
      C     ANNIHILATE THE I-TH COLUMN BELOW DIAGONAL.                   SNV010
      C                                                                  SNV010
      DO 3 J = I,M                                                       SNV010
    3 S = S + A(J,I)**2                                                  SNV010
      G = 0.0                                                            SNV010
      IF(S .LT. TOL)    GO TO 10                                         SNV010
      G = DSQRT(S)                                                       SNV010
      F = A(I,I)                                                         SNV010
      IF(F .GE. 0.0)   G = -G                                            SNV010
      H = F*G -S                                                         SNV010
      A(I,I) = F-G                                                       SNV011
      IF(I .EQ. N)   GO TO 10                                            SNV011
        DO 9 J = L,N                                                     SNV011
        S = 0.0                                                          SNV011
        DO 7 K = I,M                                                     SNV011
    7   S = S +A(K,I)*A(K,J)                                             SNV011
        F = S/H                                                          SNV011
        DO 8 K = I,M                                                     SNV011
    8   A(K,J) =A(K,J) + F*A(K,I)                                        SNV011
    9   CONTINUE                                                         SNV011
   10 Q(I) = G                                                           SNV012
      IF(I .EQ. N)   GO TO 20                                            SNV012
      C                                                                  SNV012
      C     ANNIHILATE THE I-TH ROW TO RIGHT OF SUPER-DIAG.              SNV012
      C                                                                  SNV012
      S = 0.0                                                            SNV012
      DO 11 J = L,N                                                      SNV012
```

```
   11 S = S + A(I,J)**2
      G = 0.0
      IF (S .LT. TOL)    GO TO 20
        G = DSQRT(S)
        F = A(I,I+1)
        IF(F .GE. 0.0)   G = -G
        H = F*G -S
        A(I,I+1) = F - G
        DO 15 J = L,N
   15   E(J) = A(I,J)/H
        DO 19 J = L,M
        S = 0.0
        DO 16 K = L,N
   16   S = S + A(J,K) * A(I,K)
        DO 17 K = L,N
   17   A(J,K) = A(J,K) + S*E(K)
   19   CONTINUE
   20 Y = DABS(Q(I)) + DABS(E(I))
      IF(Y .GT. SIZE)    SIZE = Y
   30 CONTINUE
      IF(.NOT. WITHV)   GO TO 41
C
C     ACCUMULATION OF RIGHT TRANSFORMATIONS.
C
      DO 40 II = 1,N
      I = NP1 - II
      IF(I .EQ. N)   GO TO 39
      IF(G .EQ. 0.0)   GO TO 37
      H = A(I,I+1)*G
      DO 32 J = L,N
   32 V(J,I) = A(I,J)/H
      DO 36 J = L,N
      S = 0.0
      DO 33 K = L,N
   33 S = S + A(I,K)*V(K,J)
      DO 34 K = L,N
   34 V(K,J) = V(K,J) + S*V(K,I)
   36 CONTINUE
   37 DO 38 J = L,N
      V(I,J) = 0.0
   38 V(J,I) = 0.0
   39 V(I,I) = 1.0
      G = E(I)
   40 L = I
   41 CONTINUE
      IF(.NOT. WITHU)   GO TO 53
C
C     ACCUMULATION OF LEFT TRANSFORMATIONS.
C
      DO 52 II = 1,N
      I = NP1 -II
      L = I + 1
      G = Q(I)
      IF(I .EQ. N)   GO TO 43
      DO 42 J = L,N
   42 A(I,J) = 0.0
   43 CONTINUE
      IF(G .EQ. 0.0)   GO TO 49
      IF(I .EQ. N)   GO TO 47
        H = A(I,I)*G
        DO 46 J = L,N
        S = 0.0
        DO 44 K = L,M
```

```
   44   S = S + A(K,I)*A(K,J)
        F = S/H
        DO 45 K = I,M
   45   A(K,J) = A(K,J) + F*A(K,I)
   46   CONTINUE
   47 DO 48 J = I,M
   48 A(J,I) = A(J,I)/G
      GO TO 51
   49 DO 50 J = I,M
   50 A(J,I) = 0.0
   51 A(I,I) = A(I,I) + 1.0
   52 CONTINUE
   53 CONTINUE
C
C     DIAGONALIZATION OF BIDIAGONAL FORM.
C
      DO 100 KK=1,N
        K=NP1-KK
        ITCNT=0
        KP1=K+1
C
C      TEST F SPLITTING.
C
   59   CONTINUE
        DO 60 LL=1,K
          L=KP1-LL
          IF((SIZE+DABS(E(L))).EQ.SIZE)     GO TO 64
          LM1=L-1
          IF((SIZE+DABS(Q(LM1))).EQ.SIZE)     GO TO 61
   60   CONTINUE
C
C      CANCELLATION OF E(L) IF L .GT. 1.
C
   61   C=0.0
        S=1.0
        L1=L-1
        DO 63 I=L,K
          F=S*E(I)
          E(I)=C*E(I)
          IF((SIZE+DABS(F)).EQ.SIZE)   GO TO 64
          G=Q(I)
          Q(I)=DSQRT(F*F+G*G)
          H=Q(I)
          C=G/H
          S=-F/H
          IF(.NOT.WITHU)   GO TO 63
            DO 62 J=1,M
              Y=A(J,L1)
              Z=A(J,I)
              A(J,L1)=Y*C+Z*S
              A(J,I)= -Y*S+Z*C
   62       CONTINUE
C
   63   CONTINUE
C
C      TEST F CONVERGENCE.
C
   64 Z=Q(K)
      IF(L.EQ.K)   GO TO 75
      IF(ITCNT .LE. 30)   GO TO 65
      IERR = KK
      RETURN
   65 ITCNT = ITCNT + 1
```

```
C
C     SHIFT FROM LOWER 2X2.
C
         X=Q(L)
         Y=Q(K-1)
      G=E(K-1)
      H=E(K)
      F=((Y-Z)*(Y+Z)+(G-H)*(G+H))/(2.0*H*Y)
      G=DSQRT(F*F+1.0)
      IF(F.LT.0.0)  G=-G
      F = ((X-Z)*(X+Z)+H*(Y/(F+G)-H))/X
C
C
C     NEXT QR TRANSFORMATION.
C
      C=1.0
      S=1.0
      LP1=L+1
      DO 73 I=LP1,K
         G=E(I)
         Y=Q(I)
         H=S*G
         G=C*G
         Z=DSQRT(F*F+H*H)
         E(I-1)=Z
         C=F/Z
         S=H/Z
         F=X*C+G*S
         G=-X*S+G*C
         H=Y*S
         Y=Y*C
         IF(.NOT.WITHV)    GO TO 70
            DO 68 J=1,N
               X=V(J,I-1)
               Z=V(J,I)
               V(J,I-1)=X*C+Z*S
               V(J,I)=-X*S+Z*C
   68       CONTINUE
C
   70    Z=DSQRT(F*F+H*H)
         Q(I-1)=Z
         C=F/Z
         S=H/Z
         F=C*G+S*Y
         X=-S*G+C*Y
         IF(.NOT.WITHU)    GO TO 73
            DO 72 J=1,M
               Y=A(J,I-1)
               Z=A(J,I)
               A(J,I-1)=Y*C+Z*S
               A(J,I)=-Y*S+Z*C
   72       CONTINUE
C
C
   73    E(L) = 0.0
         E(K)=F
         Q(K)=X
         GO TO 59
C
C        CONVERGENCE.
C
   75    CONTINUE
         IF(Z.GE.0.0)   GO TO 100
```

```
      Q(K)=-Z                                                           SNV03160
      IF(.NOT.WITHV) GO TO 100                                          SNV03170
      DO 76 J=1,N                                                       SNV03180
   76 V(J,K)=-V(J,K)                                                    SNV03190
  100 CONTINUE                                                          SNV03200
C                                                                       SNV03210
      IERR=0                                                            SNV03220
      DO 280 II=2,N                                                     SNV03230
      I=II-1                                                            SNV03240
      K=I                                                               SNV03250
      P=Q(I)                                                            SNV03260
      DO 250  J=II,N                                                    SNV03270
      IF (Q(J).LE.P) GO TO 250                                          SNV03280
      K=J                                                               SNV03290
      P=Q(J)                                                            SNV03300
  250 CONTINUE                                                          SNV03310
C                                                                       SNV03320
      IF (K.EQ.I) GO TO 280                                             SNV03330
      Q(K) = Q(I)                                                       SNV03340
      Q(I) = P                                                          SNV03350
C                                                                       SNV03360
      IF(IOP.EQ.1) GO TO 280                                            SNV03370
C                                                                       SNV03380
      DO 260  J=1,N                                                     SNV03390
      P= V(J,I)                                                         SNV03400
      V(J,I)= V(J,K)                                                    SNV03410
      V(J,K)= P                                                         SNV03420
  260 CONTINUE                                                          SNV03430
C                                                                       SNV03440
      DO 270  J=1,M                                                     SNV03450
      P = A(J,I)                                                        SNV03460
      A(J,I)= A(J,K)                                                    SNV03470
      A(J,K)= P                                                         SNV03480
  270 CONTINUE                                                          SNV03490
C                                                                       SNV03500
  280 CONTINUE                                                          SNV03510
C                                                                       SNV03520
      J=N                                                               SNV03530
  290 IF (Q(J).GT.ZTEST) GO TO 300                                      SNV03540
      Q(J)=0.0                                                          SNV03550
      J=J-1                                                             SNV03560
      GO TO 290                                                         SNV03570
  300 IRANK =J                                                          SNV03580
      TEMP = ZTEST/Q(J)                                                 SNV03590
      IF (TEMP.GT..0625)     IERR=-1                                    SNV03600
C                                                                       SNV03610
      IF (IOP.LT. 3)  RETURN                                            SNV03620
      IF(IOP.GT.3) GO TO 170                                            SNV03630
      DO 160  L=1,NOS                                                   SNV03640
      DO 130  J=1,IRANK                                                 SNV03650
      SUM=0.0                                                           SNV03660
      DO 120  I=1,M                                                     SNV03670
  120 SUM =SUM + A(I,J)*B(I,L)                                          SNV03680
  130 E(J)= SUM/Q(J)                                                    SNV03690
C                                                                       SNV03700
      DO 150  K=1,N                                                     SNV03710
      SUM=0.0                                                           SNV03720
      DO 140  I=1,IRANK                                                 SNV03730
  140 SUM =SUM  + V(K,I)*E(I)                                           SNV03740
  150 B(K,L)=SUM                                                        SNV03750
  160 CONTINUE                                                          SNV03760
      RETURN                                                            SNV03770
  170 DO 200  J=1,M                                                     SNV03780
```

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

```
      SUBROUTINE SUM(A,NA,B,NB,C,NC,IOP,SYM,DUMMY)                      SUM00010
      IMPLICIT REAL*8 (A-H,O-Z)                                         SUM00020
      DIMENSION A(1),B(1),C(1),DUMMY(1)                                 SUM00030
      DIMENSION NA(2),NB(2),NC(2)                                       SUM00040
      LOGICAL SYM                                                       SUM00050
      COMMON/CONV/SUMCV,RICTCV,SERCV,MAXSUM                             SUM00060
      IF( IOP  .EQ. 0  ) GO TO 100                                      SUM00070
      PRINT 50                                                          SUM00080
   50 FORMAT(//,' LINEAR EQUATION SOLVER    X = AXC + B ')              SUM00090
      CALL PRNT(A,NA,4H A  ,1)                                          SUM00100
      IF( SYM ) GO TO 75                                                SUM00110
      CALL PRNT(C,NC,4H C  ,1)                                          SUM00120
      GO TO 85                                                          SUM00130
   75 CONTINUE                                                          SUM00140
      PRINT 80                                                          SUM00150
   80 FORMAT(/, ' C = A TRANSPOSE ',/)                                  SUM00160
   85 CONTINUE                                                          SUM00170
      CALL PRNT(B,NB,4H B  ,1)                                          SUM00180
C                                                                       SUM00190
  100 CONTINUE                                                          SUM00200
      N1 = 1 + NA(1)*NC(1)                                              SUM00210
      I=1                                                               SUM00220
  200 CONTINUE                                                          SUM00230
      CALL MULT(A,NA,B,NB,DUMMY,NB)                                     SUM00240
      CALL MULT(DUMMY,NB,C,NC,DUMMY(N1),NB)                             SUM00250
      CALL MAXEL(B,NB,WNS)                                              SUM00260
      CALL MAXEL(DUMMY(N1),NB,WNDX)                                     SUM00270
      IF(WNS .GE. 1.) GO TO 225                                         SUM00280
      IF( WNDX/WNS .LT. SUMCV ) GO TO 300                               SUM00290
      GO TO 235                                                         SUM00300
  225 IF( WNDX .LT. SUMCV ) GO TO 300                                   SUM00310
  235 CONTINUE                                                          SUM00320
      CALL ADD(B,NB,DUMMY(N1),NB,B,NB)                                  SUM00330
      CALL MULT(A,NA,A,NA,DUMMY,NA)                                     SUM00340
      CALL EQUATE(DUMMY,NA,A,NA)                                        SUM00350
      IF( SYM ) GO TO 245                                               SUM00360
      CALL MULT(C,NC,C,NC,DUMMY,NC)                                     SUM00370
      CALL EQUATE(DUMMY,NC,C,NC)                                        SUM00380
      GO TO  250                                                        SUM00390
  245 CONTINUE                                                          SUM00400
      CALL TRANP(A,NA,C,NC)                                             SUM00410
  250 CONTINUE                                                          SUM00420
      I=I+1                                                             SUM00430
      IF( I .LE. MAXSUM ) GO TO 200                                     SUM00440
      CALL LNCNT(3)                                                     SUM00450
      PRINT 275,MAXSUM                                                  SUM00460
  275 FORMAT(//,' IN SUM, THE SEQUENCE OF PARTIAL SUMS HAS EXCEEDED STAGSUM00470
     1 E',I5,' WITHOUT CONVERGENCE')                                    SUM00480
  300 CONTINUE                                                          SUM00490
      IF(IOP .EQ. 0) RETURN                                             SUM00500
      CALL PRNT(B,NB,4H X  ,1)                                          SUM00510
      RETURN                                                            SUM00520
      END                                                               SUM00530
```

```
      SUBROUTINE BILIN(A,NA,B,NB,C,NC,IOP,BETA,SYM,DUMMY)              BIL00010
      IMPLICIT REAL*8 (A-H,O-Z)                                         BIL00020
      DIMENSION A(1),B(1),C(1),DUMMY(1)                                 BIL00030
      DIMENSION NA(2),NB(2),NC(2),NDUM(2)                               BIL00040
      DIMENSION IOP(2)                                                  BIL00050
      LOGICAL SYM                                                       BIL00060
      IF( IOP(1) .EQ. 0 )  GO TO 300                                    BIL00070
      IF(SYM) GO TO 100                                                 BIL00080
      CALL LNCNT(3)                                                     BIL00090
      PRINT 50                                                          BIL00100
   50 FORMAT(//,' LINEAR EQUATION SOLVER  AX + XB = C ')                BIL00110
      CALL PRNT(A,NA,4H A   ,1)                                         BIL00120
      CALL PRNT(B,NB,4H B   ,1)                                         BIL00130
      GO TO 200                                                         BIL00140
  100 CONTINUE                                                          BIL00150
      CALL LNCNT(3)                                                     BIL00160
      PRINT 150                                                         BIL00170
  150 FORMAT(//,' LINEAR EQUATION SOLVER  ( B TRANSPOSE )X + XB = C ')  BIL00180
      CALL TRANP(A,NA,DUMMY,NDUM)                                       BIL00190
      CALL PRNT(DUMMY,NDUM,4H B   ,1)                                   BIL00200
  200 CONTINUE                                                          BIL00210
      CALL PRNT(C,NC,4H C   ,1)                                         BIL00220
  300 CONTINUE                                                          BIL00230
C                                                                       BIL00240
      IOPTT = 0                                                         BIL00250
      N=NA(1)**2                                                        BIL00260
      M=NB(1)**2                                                        BIL00270
C                                                                       BIL00280
      IF( IOP(2) .EQ. 0 )  GO TO 500                                    BIL00290
C                                                                       BIL00300
      N1 = N + 1                                                        BIL00310
      CALL EQUATE(A,NA,DUMMY,NA)                                        BIL00320
      N2 = N1 + NA(1)                                                   BIL00330
      N3 = N2 + NA(1)                                                   BIL00340
      ISV = 0                                                           BIL00350
      ILV = 0                                                           BIL00360
      NEVL = NA(1)                                                      BIL00370
      CALL EIGEN(NA(1),NA(1),DUMMY,DUMMY(N1),DUMMY(N2),ISV,ILV,V,DUMMY(NBIL00380
     13),IERR)                                                          BIL00390
      IF (IERR .EQ. 0) GO TO 350                                        BIL00400
      CALL LNCNT(3)                                                     BIL00410
      PRINT 325,IERR                                                    BIL00420
  325 FORMAT(//,' IN BILIN, THE ',I4,' EIGENVALUE OF A HAS NOT BEEN  DETBIL00430
     1ERMINED AFTER 30 ITERATIONS')                                     BIL00440
      IERR=1                                                            BIL00450
      CALL NORMS(NEVL,NEVL,NEVL,A,IERR,BETA)                            BIL00460
      BETA=2.*BETA                                                      BIL00470
      GO TO 385                                                         BIL00480
  350 CONTINUE                                                          BIL00490
      J= N1 + NEVL -1                                                   BIL00500
      K = N2 + NEVL -1                                                  BIL00510
      CO = DSQRT(DUMMY(N1)**2 + DUMMY(N2)**2)                           BIL00520
      CN = DSQRT(DUMMY(J)**2 + DUMMY(K)**2)                             BIL00530
      CD = DUMMY(J)-DUMMY(N1)                                           BIL00540
      IF(CD .EQ. 0.0)  GO TO 365                                        BIL00550
      BETA = (DUMMY(N1)*CN-DUMMY(J)*CO)/CD                              BIL00560
      IF(BETA .LE. 0.0)  GO TO 365                                      BIL00570
      BETA = DSQRT(BETA)                                                BIL00580
      GO TO 385                                                         BIL00590
C                                                                       BIL00600
  365 CONTINUE                                                          BIL00610
C                                                                       BIL00620
      BETA = 0.0                                                        BIL00630
```

```
      DO 375 I = 1,NEVL
      J = N1 + I -1
      K = N2 + I -1
      IF(DUMMY(J) .GE. 0.0)  GO TO 375
      BETA = BETA + DSQRT(DUMMY(J)**2 + DUMMY(K)**2)
  375 CONTINUE
      BETA = BETA/NEVL
C
  385 CONTINUE
C
      IF( SYM ) GO TO 500
      CALL EQUATE(B,NB,DUMMY,NB)
      N1=M+1
      N2 = N1 +NB(1)
      N3 = N2 +NB(1)
      NEVL = NB(1)
      CALL EIGEN(NB(1),NB(1),DUMMY,DUMMY(N1),DUMMY(N2),ISV,ILV,V,DUMMY(N
     13),IERR)
      IF(IERR .EQ. 0) GO TO 450
      CALL LNCNT(3)
      PRINT 400,IERR
  400 FORMAT(//,' IN BILIN, THE ',I4,' EIGENVALUE OF B HAS NOT BEEN FOUN
     1D AFTER 30 ITERATIONS')
      IERR=1
      CALL NORMS(NEVL,NEVL,NEVL,B,IERR,BETA1)
      BETA1=2.*BETA1
      GO TO 485
  450 CONTINUE
      J = N1 + NEVL -1
      K = N2 + NEVL -1
      CO = DSQRT(DUMMY(N1)**2 + DUMMY(N2)**2)
      CN = DSQRT(DUMMY(J)**2 + DUMMY(K)**2)
      CD = DUMMY(J)-DUMMY(N1)
      IF(CD .EQ. 0.0)  GO TO 465
      BETA1 = (DUMMY(N1)*CN - DUMMY(J)*CO)/CD
      IF(BETA1 .LE. 0.0)  GO TO 465
      BETA1 = DSQRT(BETA1)
      GO TO 485
C
  465 CONTINUE
C
      BETA1 = 0.0
      DO 475 I= 1,NEVL
      J = N1 + I -1
      K = N2 + I -1
      IF(DUMMY(J) .GE. 0.0)  GO TO 475
      BETA1 = BETA1 + DSQRT(DUMMY(J)**2 + DUMMY(K)**2)
  475 CONTINUE
      BETA1 = BETA1/NEVL
C
  485 CONTINUE
      BETA = (BETA + BETA1)/2.
C
  500 CONTINUE
C
C
      IF( IOP(1) .EQ. 0 )  GO TO 520
      CALL LNCNT(4)
      PRINT 515,BETA
  515 FORMAT(//,' BETA = ',E16.8,/)
  520 CONTINUE
C
      N1 = N+1
```

```
      CALL EQUATE(A,NA,DUMMY,NA)                                        BIL01270
      CALL EQUATE(A,NA,DUMMY(N1),NA)                                    BIL01280
      CALL SCALE(DUMMY,NA,DUMMY,NA,-1.)                                 BIL01290
      L = -NA(1)                                                        BIL01300
      NAX = NA(1)                                                       BIL01310
      DO 525 I=1,NAX                                                    BIL01320
      L = L + NAX +1                                                    BIL01330
      M1 = L + N                                                        BIL01340
      DUMMY(L) = BETA - A(L)                                            BIL01350
      DUMMY(M1)= BETA + A(L)                                            BIL01360
  525 CONTINUE                                                          BIL01370
      N2 = N1 + N                                                       BIL01380
      CALL EQUATE(C,NC,DUMMY(N2),NDUM)                                  BIL01390
      NDUM(2)= NDUM(2) + NA(1)                                          BIL01400
      N3 = N2 + NC(1)*NC(2)                                             BIL01410
      GAM = -2.*BETA                                                    BIL01420
C                                                                       BIL01430
      IF( .NOT. SYM ) GO  TO 600                                        BIL01440
C                                                                       BIL01450
      CALL UNITY(DUMMY(N3),NA)                                          BIL01460
      N4 = N3 + N                                                       BIL01470
      NDUM(2) = NDUM(2) + NA(1)                                         BIL01480
      N5 = N4 + NA(1)                                                   BIL01490
      IFAC = 0                                                          BIL01500
      CALL GELIM(NA(1),NA(1),DUMMY,NDUM(2),DUMMY(N1),DUMMY(N4),IFAC,DUMMBIL01510
     1Y(N5),IERR)                                                       BIL01520
      IF( IERR .EQ. 1 )  PRINT 625                                      BIL01530
      CALL EQUATE(DUMMY(N1),NA,DUMMY,NA)                                BIL01540
      CALL EQUATE(DUMMY(N2),NC,C,NC)                                    BIL01550
      CALL TRANP(DUMMY,NA,DUMMY(N1),NA)                                 BIL01560
      CALL TRANP(DUMMY(N3),NA,DUMMY(N2),NA)                             BIL01570
      CALL MULT(C,NC,DUMMY(N2),NA,DUMMY(N3),NA)                         BIL01580
      CALL SCALE(DUMMY(N3),NC,C,NC,GAM)                                 BIL01590
C                                                                       BIL01600
C                                                                       BIL01610
      CALL SUM(DUMMY,NA,C,NC,DUMMY(N1),NA,IOPTT,SYM,DUMMY(N2))          BIL01620
      GO TO 700                                                         BIL01630
  600 CONTINUE                                                          BIL01640
      N4 = N3 +NA(1)                                                    BIL01650
      IFAC = 0                                                          BIL01660
      CALL GELIM(NA(1),NA(1),DUMMY,NDUM(2),DUMMY(N1),DUMMY(N3),IFAC,DUMMBIL01670
     1Y(N4),IERR)                                                       BIL01680
      IF(IERR .EQ. 1 ) PRINT 625                                        BIL01690
  625 FORMAT(//,' IN BILIN, THE MATRIX  (BETA)I - A IS SINGULAR, INCREASBIL01700
     1E BETA ')                                                         BIL01710
      CALL EQUATE(DUMMY(N1),NA,DUMMY,NA)                                BIL01720
      CALL EQUATE(DUMMY(N2),NC,C,NC)                                    BIL01730
      N2 = M + N                                                        BIL01740
      CALL EQUATE(B,NB,DUMMY(N1),NB)                                    BIL01750
      CALL EQUATE(B,NB,DUMMY(N2),NB)                                    BIL01760
      CALL SCALE(DUMMY(N1),NB,DUMMY(N1),NB,-1.0)                        BIL01770
      L=-NB(1)                                                          BIL01780
      NAX=NB(1)                                                         BIL01790
      DO650I =1,NAX                                                     BIL01800
      L=L + NAX +1                                                      BIL01810
      L1 = L + N                                                        BIL01820
      M1 = L + N2-1                                                     BIL01830
      DUMMY(L1)= BETA- B(L)                                             BIL01840
      DUMMY(M1)= BETA + B(L)                                            BIL01850
  650 CONTINUE                                                          BIL01860
C                                                                       BIL01870
      N3 = N2 + M                                                       BIL01880
      CALL TRANP(DUMMY(N1),NB,DUMMY(N3),NB)                             BIL01890
```

```
      CALL EQUATE(DUMMY(N3),NB,DUMMY(N1),NB)                           BIL01900
      CALL TRANP(DUMMY(N2),NB,DUMMY(N3),NB)                            BIL01910
      CALL EQUATE(DUMMY(N3),NB,DUMMY(N2),NB)                           BIL01920
      CALL TRANP(C,NC,DUMMY(N3),NDUM)                                  BIL01930
      NSDUM = NDUM(2)                                                  BIL01940
      NDUM(2)= NDUM(2) + NB(2)                                         BIL01950
      IFAC = 0                                                         BIL01960
      N4=N3+NC(1)*NC(2)                                                BIL01970
      N5=N4+NB(1)                                                      BIL01980
      CALL GELIM(NB(1),NB(1),DUMMY(N1),NDUM(2),DUMMY(N2),DUMMY(N4),IFAC,BIL01990
     1DUMMY(N5),IERR)                                                  BIL02000
      IF(IERR .EQ. 1 )  PRINT 675                                      BIL02010
  675 FORMAT(//,'IN BILIN, THE MATRIX (BETA)I - B  IS SINGULAR, INCREASEBIL02020
     1 BETA ')                                                         BIL02030
      CALL TRANP(DUMMY(N2),NB,DUMMY(N1),NB)                            BIL02040
      NDUM(2)= NSDUM                                                   BIL02050
      CALL TRANP(DUMMY(N3),NDUM,C,NC)                                  BIL02060
      CALL SCALE(C,NC,C,NC,GAM)                                        BIL02070
      N2 = N + M + 1                                                   BIL02080
      CALL SUM(DUMMY,NA,C,NC,DUMMY(N1),NB,IOPTT,SYM,DUMMY(N2))         BIL02090
C                                                                      BIL02100
  700 CONTINUE                                                         BIL02110
      IF( IOP(1) .EQ. 0 ) RETURN                                       BIL02120
      CALL PRNT(C,NC,4H X   ,1)                                        BIL02130
      RETURN                                                           BIL02140
      END                                                              BIL02150
```

```
      SUBROUTINE BARSTW(A,NA,B,NB,C,NC,IOP,SYM,EPSA,EPSB,DUMMY)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(1),B(1),C(1),DUMMY(1)
      DIMENSION NA(2),NB(2),NC(2),NDUM1(2),NDUM2(2),NDUM3(2),NDUM4(2)
      LOGICAL  SYM
      IF ( IOP .EQ. 0 )  GO TO 250
      IF(SYM) GO TO 100
      CALL LNCNT(3)
      PRINT 50
   50 FORMAT(//,' LINEAR EQUATION SOLVER       AX + XB = C ')
      CALL PRNT(A,NA,4H A   ,1)
      CALL PRNT(B,NB,4H B   ,1)
      GO TO 200
  100 CONTINUE
      CALL LNCNT(3)
      PRINT 150
  150 FORMAT(//,' LINEAR EQUATION SOLVER   ( B TRANSPOSE )X + XB = C')
      CALL TRANP(A,NA,DUMMY,NDUM1)
      CALL PRNT(DUMMY,NDUM1,4H B   ,1)
  200 CONTINUE
      CALL PRNT(C,NC,4H C   ,1)
C
  250 CONTINUE
      CALL EQUATE(A,NA,DUMMY,NDUM1)
      N1=(NA(1)**2)+1
      N2=N1+NA(1)-1
      DO 300I=N1,N2
      DUMMY(I)=0.0
  300 CONTINUE
C
      NDUM1(2)=NDUM1(2)+1
      NDUM2(1)=1
      NDUM2(2)=NDUM1(2)
      N1=NDUM1(1)*NDUM1(2)+1
      CALL NULL(DUMMY(N1),NDUM2)
      LU=(NA(1)+1)**2 + 1
      CALL JUXTR(DUMMY,NDUM1,DUMMY(N1),NDUM2,DUMMY(LU),NDUM3)
      CALL EQUATE(DUMMY(LU),NDUM3,DUMMY,NDUM1)
      N=NA(1)+1
C
      IF(SYM ) GO TO 500
C
      CALL EQUATE(B,NB,DUMMY(LU),NDUM2)
      M1=LU+NB(1)**2
      M2=M1+NB(1)-1
      DO400I=M1,M2
      DUMMY(I)=0.0
  400 CONTINUE
C
      NDUM2(2)=NDUM2(2)+1
      NDUM3(1)=1
      NDUM3(2)=NDUM2(2)
      M1=NDUM2(1)*NDUM2(2)+LU
      CALL NULL(DUMMY(M1),NDUM3)
      M2=LU+(NB(1)+1)**2
      CALL JUXTR(DUMMY(LU),NDUM2,DUMMY(M1),NDUM3,DUMMY(M2),NDUM4)
      CALL EQUATE(DUMMY(M2),NDUM4,DUMMY(LU),NDUM2)
      M=NB(1)+ 1
      LNB = LU
      LU = LU + (NB(1)+1)**2
      LV = LU +  NA(1)**2
      CALL AXPXB(DUMMY,DUMMY(LU),NA(1),N,NA(1),DUMMY(LNB),DUMMY(LV),NB(1
     1),M,NB(1),C,NC(1),EPSA,EPSB,NFAIL)
```

```
      GO TO 600                                                         BAR00640
C                                                                       BAR00650
  500 CONTINUE                                                          BAR00660
      CALL TRANP(DUMMY,NDUM1,DUMMY(LU),NDUM2)                           BAR00670
      CALL EQUATE(DUMMY(LU),NDUM2,DUMMY,NDUM1)                          BAR00680
      CALL ATXPXA(DUMMY,DUMMY(LU),C,NA(1),N,NA(1),NC(1),EPSA,NFAIL)     BAR00690
C                                                                       BAR00700
  600 CONTINUE                                                          BAR00710
      IF(NFAIL .EQ. 0 ) GO TO 700                                       BAR00720
      CALL LNCNT(3)                                                     BAR00730
      PRINT 650                                                         BAR00740
  650 FORMAT(//,' IN BARSTW, EITHER THE SUBROUTINE AXPXB  OR  ATXPXA  WABAR00750
     1S UNABLE TO REDUCE A OR B TO SCHUR FORM ')                        BAR00760
      RETURN                                                            BAR00770
C                                                                       BAR00780
  700 CONTINUE                                                          BAR00790
C                                                                       BAR00800
      IF( IOP .NE. 0 )  CALL PRNT(C,NC,4H  X ,1)                        BAR00810
      RETURN                                                            BAR00820
      END                                                               BAR00830
```

```
      SUBROUTINE TESTSA(A,NA,ALPHA,DISC,STABLE,IOP,DUMMY)              TES00010
      IMPLICIT REAL*8 (A-H,O-Z)                                         TES00020
      DIMENSION A(1),DUMMY(1)                                           TES00030
      DIMENSION NA(2),NDUM1(2),NDUM2(2)                                 TES00040
      LOGICAL  DISC,STABLE                                              TES00050
      STABLE = .FALSE.                                                  TES00060
C                                                                       TES00070
      CALL EQUATE(A,NA,DUMMY,NA)                                        TES00080
      N1= NA(1)**2 + 1                                                  TES00090
      N2= N1+NA(1)                                                      TES00100
      N3= N2+NA(1)                                                      TES00110
      ISV = 0                                                           TES00120
      CALL EIGEN(NA(1),NA(1),DUMMY,DUMMY(N1),DUMMY(N2),ISV,ISV,V,DUMMY(NTES00130
     13),IERR)                                                          TES00140
      NEVL = NA(1)                                                      TES00150
      IF( IERR .EQ. 0 ) GO TO 200                                       TES00160
      CALL LNCNT(4)                                                     TES00170
      PRINT 100,IERR                                                    TES00180
  100 FORMAT(//,' IN TESTSA, THE ',I5,' EIGENVALUE OF A HAS NOT BEEN FO TES00190
     1UND AFTER 30 ITERATIONS',/)                                       TES00200
      RETURN                                                            TES00210
C                                                                       TES00220
  200 CONTINUE                                                          TES00230
      NDUM1(1) = NEVL                                                   TES00240
      NDUM1(2) = 1                                                      TES00250
      CALL JUXTC(DUMMY(N1),NDUM1,DUMMY(N2),NDUM1,DUMMY,NDUM2)           TES00260
C                                                                       TES00270
      IF( DISC ) GO TO 400                                              TES00280
      DO 300 I=1,NEVL                                                   TES00290
      IF( DUMMY(I) .GE. ALPHA ) GO TO 600                               TES00300
  300 CONTINUE                                                          TES00310
      GO TO 550                                                         TES00320
  400 CONTINUE                                                          TES00330
      N = NDUM2(1)*NDUM2(2)+1                                           TES00340
      DO 500 I =1,NEVL                                                  TES00350
      K = I + NEVL                                                      TES00360
      L=N +I -1                                                         TES00370
      DUMMY(L) = DSQRT((DUMMY(I)**2)+(DUMMY(K)**2))                     TES00380
  500 CONTINUE                                                          TES00390
C                                                                       TES00400
      IF( DUMMY(L) .GE. ALPHA ) GO TO 600                               TES00410
C                                                                       TES00420
  550 CONTINUE                                                          TES00430
      STABLE =.TRUE.                                                    TES00440
  600 CONTINUE                                                          TES00450
      IF( IOP .EQ. 0 ) RETURN                                           TES00460
      CALL LNCNT(4)                                                     TES00470
      PRINT 700                                                         TES00480
  700 FORMAT(//,' PROGRAM TO TEST THE RELATIVE STABILITY OF THE MATRIX ATES00490
     1',/)                                                              TES00500
      CALL PRNT(A,NA,4H A   ,1)                                         TES00510
      CALL LNCNT(4)                                                     TES00520
      PRINT 750                                                         TES00530
  750 FORMAT(//,' EIGENVALUES OF A ',/)                                 TES00540
      CALL PRNT(DUMMY,NDUM2,4HEVLA,1)                                   TES00550
      IF(  .NOT. DISC ) GO TO 850                                       TES00560
      CALL LNCNT(4)                                                     TES00570
      PRINT 800                                                         TES00580
  800 FORMAT(//,' MODULI OF EIGENVALUES OF A',/)                        TES00590
      CALL PRNT(DUMMY(N),NDUM1,4HMODA,1)                                TES00600
C                                                                       TES00610
  850 CONTINUE                                                          TES00620
      CALL LNCNT(4)                                                     TES00630
```

```
      IF(STABLE) PRINT 900,ALPHA                                        TES00640
      IF( .NOT. STABLE) PRINT 950,ALPHA                                 TES00650
  900 FORMAT(//,' MATRIX A  IS STABLE RELATIVE TO ',E16.8,/)           TES00660
  950 FORMAT(//,' MATRIX A  IS UNSTABLE RELATIVE TO ',E16.8,/)         TES00670
C                                                                       TES00680
      RETURN                                                            TES00690
      END                                                               TES00700
```

```
      SUBROUTINE EXPSER(A,NA,EXPA,NEXPA,T,IOP,DUMMY)                    EXP00010
      IMPLICIT REAL*8 (A-H,O-Z)                                         EXP00020
      DIMENSION A(1),EXPA(1),DUMMY(1)                                   EXP00030
      DIMENSION NA(2),NEXPA(2)                                          EXP00040
      COMMON/CONV/SUMCV,RICTCV,SERCV,MAXSUM                             EXP00050
C                                                                       EXP00060
      N = NA(1)                                                         EXP00070
      L = (N**2) + 1                                                    EXP00080
      TT = T                                                            EXP00090
      NEXPA(1)=NA(1)                                                    EXP00100
      NEXPA(2)=NA(2)                                                    EXP00110
C                                                                       EXP00120
      CALL MAXEL(A,NA,ANAA)                                             EXP00130
      ANAA = ANAA*TT                                                    EXP00140
      ANAA = DABS(ANAA)                                                 EXP00150
      IF( ANAA .GT. 1.E-15 ) GO TO 100                                  EXP00160
      CALL UNITY(EXPA,NEXPA)                                            EXP00170
      GO TO 800                                                         EXP00180
C                                                                       EXP00190
  100 CONTINUE                                                          EXP00200
      IOPT=2                                                            EXP00210
      CALL NORMS(N,N,N,A,IOPT,ZERO)                                     EXP00220
      ZERO=ZERO/(2.**47)                                                EXP00230
      CALL TRCE(A,NA,TR)                                                EXP00240
      TR = TR/N                                                         EXP00250
      DO 200 I =1,N                                                     EXP00260
      M =I+N*(I-1)                                                      EXP00270
      A(M) = A(M) - TR                                                  EXP00280
  200 CONTINUE                                                          EXP00290
C                                                                       EXP00300
      IOPT = 1                                                          EXP00310
      CALL NORMS(N,N,N,A,IOPT,COL)                                      EXP00320
      IOPT = 3                                                          EXP00330
      CALL NORMS(N,N,N,A,IOPT,ROW)                                      EXP00340
      ANORM = ROW                                                       EXP00350
      IF( ANORM .GT. COL )  ANORM = COL                                 EXP00360
      TMAX = 1./ANORM                                                   EXP00370
      K= 0                                                              EXP00380
  300 CONTINUE                                                          EXP00390
      IF( TMAX - TT ) 325,350,350                                       EXP00400
  325 CONTINUE                                                          EXP00410
      K=K+1                                                             EXP00420
      TT = T/(2**K)                                                     EXP00430
      IF( K - 1000 ) 300,700,700                                        EXP00440
  350 CONTINUE                                                          EXP00450
      SC = TT                                                           EXP00460
      CALL SCALE(A,NA,A,NA,TT)                                          EXP00470
      CALL UNITY(EXPA,NEXPA)                                            EXP00480
      II = 2                                                            EXP00490
      CALL ADD(A,NA,EXPA,NEXPA,DUMMY,NA)                                EXP00500
      CALL EQUATE(A,NA,DUMMY(L),NA)                                     EXP00510
  400 CONTINUE                                                          EXP00520
      CALL MULT(A,NA,DUMMY(L),NA,EXPA,NEXPA)                            EXP00530
      S = 1./II                                                         EXP00540
      CALL SCALE(EXPA,NEXPA,DUMMY(L),NA,S)                              EXP00550
      CALL ADD(DUMMY(L),NA,DUMMY,NA,EXPA,NEXPA)                         EXP00560
      CALL MAXEL(DUMMY,NA,TOT)                                          EXP00570
      CALL MAXEL(DUMMY(L),NA,DELT)                                      EXP00580
      IF( TOT .GT. 1.0 ) GO TO 500                                      EXP00590
      IF( DELT/TOT .LT. SERCV )  GO TO 600                              EXP00600
      GO TO 550                                                         EXP00610
  500 CONTINUE                                                          EXP00620
      IF( DELT .LT. SERCV )  GO TO 600                                  EXP00630
```

```
  550 CONTINUE                                                          EXP00640
      CALL EQUATE(EXPA,NEXPA,DUMMY,NA)                                  EXP00650
      II = II + 1                                                       EXP00660
      GO TO 400                                                         EXP00670
C                                                                       EXP00680
  600 CONTINUE                                                          EXP00690
      IF( K ) 625,675,650                                               EXP00700
  625 CONTINUE                                                          EXP00710
      CALL LNCNT(1)                                                     EXP00720
      PRINT 635                                                         EXP00730
  635 FORMAT( '    ERROR IN EXPSER,   K IS NEGATIVE ' )                 EXP00740
      RETURN                                                            EXP00750
C                                                                       EXP00760
  650 CONTINUE                                                          EXP00770
      DO 660 I =1,K                                                     EXP00780
      TT = 2*TT                                                         EXP00790
      CALL EQUATE(EXPA,NEXPA,DUMMY,NA)                                  EXP00800
      CALL EQUATE(DUMMY,NA,DUMMY(L),NA)                                 EXP00810
      CALL MULT(DUMMY(L),NA,DUMMY,NA,EXPA,NEXPA)                        EXP00820
  660 CONTINUE                                                          EXP00830
      T = TT                                                            EXP00840
  675 CONTINUE                                                          EXP00850
      S = 1./SC                                                         EXP00860
      CALL SCALE(A,NA,A,NA,S)                                           EXP00870
      DO 685 I = 1,N                                                    EXP00880
      M = I + N*(I-1)                                                   EXP00890
      A(M) = A(M) + TR                                                  EXP00900
      IF( DABS(A(M)) .LE. ZERO ) A(M) = 0.0                             EXP00910
  685 CONTINUE                                                          EXP00920
C                                                                       EXP00930
      TR=TR*T                                                           EXP00940
      S =  DEXP(T)                                                      EXP00950
      CALL SCALE(EXPA,NEXPA,EXPA,NEXPA,S)                               EXP00960
      GO TO 800                                                         EXP00970
C                                                                       EXP00980
  700 CONTINUE                                                          EXP00990
      CALL LNCNT(1)                                                     EXP01000
      PRINT 750                                                         EXP01010
  750 FORMAT('  ERROR IN EXPSER,   K = 1000 ')                          EXP01020
      RETURN                                                            EXP01030
C                                                                       EXP01040
  800 CONTINUE                                                          EXP01050
      IF( IOP .EQ. 0 ) RETURN                                           EXP01060
      CALL LNCNT(4)                                                     EXP01070
      PRINT 825                                                         EXP01080
  825 FORMAT(//' COMPUTATION OF THE MATRIX EXPONENTIAL EXP(A T) BY THE SEXP01090
     1ERIES METHOD '/)                                                  EXP01100
      CALL PRNT(A,NA,4H A   ,1)                                         EXP01110
      CALL LNCNT(3)                                                     EXP01120
      PRINT 850,T                                                       EXP01130
  850 FORMAT(/' T = ',D16.8/)                                           EXP01140
      CALL PRNT(EXPA,NEXPA,4HEXPA,1)                                    EXP01150
      RETURN                                                            EXP01160
      END                                                               EXP01170
```

```
      SUBROUTINE EXPADE (MAX, N, A, EA, IDIG, WK, IERR)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(MAX,N),EA(MAX,N),WK(N,1),C(9)
      REAL*4 SDIGC,ALOG10
      IERR = 0
C     CALCULATE NORM OF A
      ANORM = 0.
      DO 10 I=1,N
         S = 0.
         DO 5 J=1,N
            S = S + DABS(A(I,J))
    5    CONTINUE
         IF (S .GT. ANORM)  ANORM = S
   10 CONTINUE
C ****
C     CALCULATE ACCURACY ESTIMATE
C ****
      DIGC = 24.*DFLOAT(N)
      IF (ANORM .GT. 1.) DIGC = DIGC*ANORM
      SDIGC=DIGC
      IDIG = 15 - IFIX(ALOG10(SDIGC) + .5)
C     DETERMINE POWER OF TWO AND NORMALIZATION FACTOR
C ****
      M = 0
      IF (ANORM .LE. 1.)  GO TO 27
      FACTOR =2.
      DO 15 M=1,46
         IF (ANORM .LE. FACTOR) GO TO 20
         FACTOR = FACTOR*2.
   15 CONTINUE
      GO TO 125
   20 CONTINUE
C ****
C     NORMALIZE MATRIX
C ****
      DO 25 I=1,N
         DO 25 J=1,N
            A(I,J) = A(I,J)/FACTOR
   25 CONTINUE
   27 CONTINUE
C ****
C     SET COEFFICIENTS FOR (9,9) PADE TABLE ENTRY
C ****
      C(1) = .5
      C(2) = 1.1764705882352D-01
      C(3) = 1.7156862745098D-02
      C(4) = 1.7156862745098D-03
      C(5) = 1.2254901960784D-04
      C(6) = 6.2845651080945D-06
      C(7) = 2.2444875386051D-07
      C(8) = 5.1011080422845D-09
      C(9) = 5.6678978247605D-11
C ****
C     CALCULATE PADE NUMERATOR AND DENOMINATOR BY COLUMNS
C ****
      NP1 = N+1
      NP7 = N+7
      DO 95 J=1,N
C ****
C        COMPUTE JTH COLUMN OF FIRST NINE POWERS OF A
C ****
         DO 35 I=1,N
            S = 0.
```

```
            DO 30 L=1,N                                                  EXP00640
               S = S + A(I,L)*A(L,J)                                     EXP00650
   30       CONTINUE                                                     EXP00660
            WK(I,NP1) = S                                                EXP00670
   35    CONTINUE                                                        EXP00680
         DO 45 K=NP1,NP7                                                 EXP00690
            KP1 = K+1                                                    EXP00700
            DO 45 I=1,N                                                  EXP00710
               S = 0.                                                    EXP00720
               DO 40 L=1,N                                               EXP00730
                  S = S + A(I,L)*WK(L,K)                                 EXP00740
   40          CONTINUE                                                  EXP00750
               WK(I,KP1) = S                                             EXP00760
   45    CONTINUE                                                        EXP00770
C ****                                                                   EXP00780
C        COLLECT TERMS FOR JTH COLUMN OF NUMERATOR AND DENOMINATOR       EXP00790
C ****                                                                   EXP00800
         DO 85 I=1,N                                                     EXP00810
            S = 0.                                                       EXP00820
            U = 0.                                                       EXP00830
            DO 65 L=1,8                                                  EXP00840
               K = N+9-L                                                 EXP00850
               KN1 = K-N+1                                               EXP00860
               P = C(KN1)*WK(I,K)                                        EXP00870
               S = S + P                                                 EXP00880
              IEO = MOD(KN1,2)                                           EXP00890
              IF (IEO.EQ.0) GO TO 55                                     EXP00900
               U = U - P                                                 EXP0091
               GO TO 65                                                  EXP0092
   55          CONTINUE                                                  EXP0093
               U = U + P                                                 EXP0094
   65       CONTINUE                                                     EXP0095
            P = C(1)*A(I,J)                                              EXP0096
            S = S + P                                                    EXP0097
            U = U - P                                                    EXP0098
            IF (I .NE. J) GO TO 80                                       EXP0099
            S = S + 1.                                                   EXP0100
            U = U + 1.                                                   EXP0101
   80       CONTINUE                                                     EXP0102
            EA(I,J) = S                                                  EXP0103
            WK(I,J) = U                                                  EXP0104
   85    CONTINUE                                                        EXP0105
   95 CONTINUE                                                           EXP0106
C ****                                                                   EXP0107
C     CALCULATE NORMALIZED EXP(A) BY  WK * EXP(A) = EA                   EXP0108
C ****                                                                   EXP0109
      CALL GAUSEL (MAX,N,WK,N,EA,IERR)                                   EXP0110
      IF (IERR .NE. 0) GO TO 130                                         EXP0111
      IF (M .EQ. 0)  GO TO 130                                           EXP0112
C ****                                                                   EXP0113
C     TAKE OUT EFFECT OF NORMALIZATION ON EXP(A)                         EXP0114
C ****                                                                   EXP0115
      DO 120 K=1,M                                                       EXP0116
         DO 110 I=1,N                                                    EXP0117
            DO 110 J=1,N                                                 EXP0118
               S = 0.                                                    EXP0119
               DO 105 L=1,N                                              EXP0120
                  S = S + EA(I,L)*EA(L,J)                                EXP0121
  105          CONTINUE                                                  EXP0122
               WK(I,J) = S                                               EXP0123
  110    CONTINUE                                                        EXP0124
         DO 115 I=1,N                                                    EXP0125
            DO 115 J=1,N                                                 EXP0126
```

```
                 EA(I,J) = WK(I,J)                                    EXP01270
  115        CONTINUE                                                  EXP01280
  120   CONTINUE                                                       EXP01290
C ****                                                                 EXP01300
C       UN-NORMALIZE A                                                 EXP01310
C ****                                                                 EXP01320
        DO 122 I=1,N                                                   EXP01330
           DO 122 J=1,N                                                EXP01340
              A(I,J) = A(I,J)*FACTOR                                   EXP01350
  122   CONTINUE                                                       EXP01360
        GO TO 130                                                      EXP01370
C ****                                                                 EXP01380
C       NORM OF A IS EXCESSIVE                                         EXP01390
C ****                                                                 EXP01400
  125   CONTINUE                                                       EXP01410
        IERR = 1                                                       EXP01420
C ****                                                                 EXP01430
C       EXIT ROUTINE                                                   EXP01440
C ****                                                                 EXP01450
  130   CONTINUE                                                       EXP01460
        RETURN                                                         EXP01470
        END                                                            EXP01480
```

```
      SUBROUTINE EXPINT(A,NA,B,NB,C,NC,T,IOP,DUMMY)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(1),B(1),C(1),DUMMY(1)
      DIMENSION NA(2),NB(2),NC(2)
      COMMON/CONV/SUMCV,RICTCV,SERCV,MAXSUM
      N = NA(1)
      L = (N**2)+1
      NC(1) = NA(1)
      NC(2) = NA(2)
      NB(1) = NA(1)
      NB(2) = NA(2)
      TT = T
C
      IOPT = 1
      CALL NORMS(N,N,N,A,IOPT,COL)
      IOPT = 3
      CALL NORMS(N,N,N,A,IOPT,ROW)
      ANAA = COL
      IF( ANAA .GT. ROW ) ANAA = ROW
      TMAX = 1./ANAA
      K = 0
  100 CONTINUE
      IF( TMAX - TT ) 125,150,150
  125 CONTINUE
      K = K + 1
      TT = T/(2**K)
      IF( K - 1000 )100,600,600
C
  150 CONTINUE
      SC = TT
      CALL SCALE(A,NA,A,NA,TT)
      CALL UNITY(B,NB)
      CALL SCALE(B,NB,DUMMY,NB,TT)
      S =  TT/2.
      CALL SCALE(A,NA,DUMMY(L),NA,S)
      II = 2
      CALL ADD(DUMMY,NA,DUMMY(L),NA,DUMMY(L),NA)
      CALL ADD(A,NA,B,NB,DUMMY,NA)
      CALL EQUATE(A,NA,C,NC)
  200 CONTINUE
      CALL MULT(A,NA,C,NC,B,NB)
      S = 1./II
      CALL SCALE(B,NB,C,NC,S)
      CALL MAXEL(DUMMY,NA,TOT)
      CALL MAXEL(C,NC,DELT)
      IF( TOT .GT. 1.0 ) GO TO 300
      IF( DELT/TOT .LT. SERCV )  GO TO 400
      GO TO 350
  300 CONTINUE
      IF( DELT .LT. SERCV )  GO TO 400
  350 CONTINUE
      S = TT/(II + 1)
      CALL SCALE(C,NC,B,NB,S)
      CALL ADD(B,NB,DUMMY(L),NB,DUMMY(L),NB)
      CALL ADD(C,NC,DUMMY,NC,DUMMY,NC)
      II = II + 1
      GO TO 200
C
  400 CONTINUE
      CALL EQUATE(DUMMY,NB,B,NB)
      IF( K ) 425,500,450
  425 CONTINUE
      CALL LNCNT(1)
```

```
      PRINT 435
  435 FORMAT('  ERROR IN EXPINT, K IS NEGATIVE')
      RETURN
C
  450 CONTINUE
      DO 475 J = 1,K
      TT = 2*TT
      CALL EQUATE(B,NB,DUMMY,NB)
      CALL MULT(DUMMY,NA,DUMMY(L),NA,C,NC)
      CALL ADD(DUMMY(L),NC,C,NC,DUMMY(L),NC)
      CALL MULT(DUMMY,NB,DUMMY,NB,B,NB)
  475 CONTINUE
      T = TT
C
  500 CONTINUE
      CALL EQUATE(DUMMY(L),NC,C,NC)
      S = 1./SC
      CALL SCALE(A,NA,A,NA,S)
C
      IF( IOP .EQ. 0 ) RETURN
      CALL LNCNT(5)
      PRINT 550
  550 FORMAT(//,' COMPUTATION OF THE MATRIX EXPONENTIAL  EXP(A T)',/,' A
     1ND ITS INTEGRAL OVER  (0,T) BY THE SERIES METHOD ',/)
      CALL PRNT(A,NA,4H A  ,1)
      CALL LNCNT(3)
      PRINT 575, T
  575 FORMAT(/,'  T = ',D16.8,/)
      CALL  PRNT(B,NB,4HEXPA,1)
      CALL PRNT(C,NC,4HINT ,1)
      RETURN
C
  600 CONTINUE
      CALL LNCNT(1)
      PRINT 650
  650 FORMAT( ' ERROR IN EXPINT, K = 1000 ')
      RETURN
C
      END
```

```
      SUBROUTINE VARANC(A,NA,G,NG,Q,NQ,W,NW,IDENT,DISC,IOP,DUMMY)       VAR00010
      IMPLICIT REAL*8 (A-H,O-Z)                                         VAR00020
      DIMENSION A(1),G(1),Q(1),W(1),DUMMY(1)                            VAR00030
      DIMENSION NA(2),NG(2),NQ(2),NW(2),NDUM1(2),IOP(3),IOPT(2)         VAR0004
      LOGICAL IDENT,DISC,SYM                                            VAR0005
      COMMON/TOL/EPSAM,EPSBM,IACM                                       VAR0006
      IF( IOP(1) .EQ. 0 ) GO TO 100                                     VAR0007
      CALL LNCNT(5)                                                     VAR0008
      IF( DISC ) PRINT 25                                               VAR0009
      IF( .NOT. DISC ) PRINT 35                                         VAR0010
   25 FORMAT(//,' PROGRAM TO SOLVE FOR THE STEADY-STATE VARIANCE MATRIX'VAR0011
     1,/,' FOR A LINEAR DISCRETE SYSTEM',/)                             VAR0012
   35 FORMAT(//,' PROGRAM TO SOLVE FOR THE STEADY-STATE VARIANCE MATRIX'VAR0013
     1,/,' FOR A LINEAR CONTINUOUS SYSTEM',/)                           VAR0014
      CALL PRNT(A,NA,4H A   ,1)                                         VAR0015
      IF( .NOT. IDENT ) GO TO 55                                        VAR0016
      CALL LNCNT(3)                                                     VAR0017
      PRINT 45                                                          VAR0018
   45 FORMAT(/,' G IS AN IDENTITY MATRIX ',/)                           VAR0019
      GO TO 65                                                          VAR0020
   55 CONTINUE                                                          VAR0021
      CALL PRNT(G,NG,4H G   ,1)                                         VAR0022
   65 CONTINUE                                                          VAR0023
      IF ( .NOT. IDENT ) GO TO 85                                       VAR0024
      CALL LNCNT(3)                                                     VAR0025
      PRINT 75                                                          VAR0026
   75 FORMAT(/,' INTENSITY MATRIX FOR COVARIANCE OF PROCESS NOISE ',/)  VAR0027
C                                                                       VAR0028
   85 CONTINUE                                                          VAR0029
      CALL PRNT(Q,NQ,4H Q   ,1)                                         VAR0030
C                                                                       VAR0031
  100 CONTINUE                                                          VAR0032
      IF( IDENT ) GO TO 200                                             VAR0033
      CALL MULT(G,NG,Q,NQ,DUMMY,NG)                                     VAR0034
      N1 = NG(1)*NG(2) + 1                                              VAR0035
      CALL TRANP(G,NG,DUMMY(N1),NDUM1)                                  VAR0036
      CALL MULT(DUMMY,NG,DUMMY(N1),NDUM1,Q,NQ)                          VAR0037
C                                                                       VAR0038
      IF( IOP(1) .EQ. 0 ) GO TO 200                                     VAR0039
      CALL LNCNT(3)                                                     VAR0040
      PRINT 75                                                          VAR0041
      CALL PRNT(Q,NQ,4HGQGT,1)                                          VAR0042
C                                                                       VAR0043
  200 CONTINUE                                                          VAR0044
      IF(.NOT. DISC) CALL SCALE(W,NW,W,NW,-1.0)                         VAR0045
      IOPT(1) = IOP(2)                                                  VAR0046
      IOPT(2) = 1                                                       VAR0047
      SYM = .TRUE.                                                      VAR0048
      IF( DISC ) GO TO 300                                              VAR0049
      IF( IOP(3) .EQ. 0 ) GO TO 250                                     VAR0050
      CALL BILIN(A,NA,A,NA,W,NW,IOPT,BETA,SYM,DUMMY)                    VAR0051
      GO TO 400                                                         VAR0052
C                                                                       VAR0053
  250 CONTINUE
      CALL BARSTW(A,NA,A,NA,W,NW,IOPT,SYM,EPSA,EPSA,DUMMY)              VAR0056
      GO TO 400                                                         VAR0057
C                                                                       VAR0058
  300 CONTINUE                                                          VAR0059
      CALL EQUATE(A,NA,DUMMY,NA)                                        VAR0060
      N = NA(1)**2                                                      VAR0061
      N1 = N + 1                                                        VAR0062
      CALL TRANP(A,NA,DUMMY(N1),NA)                                     VAR0063
      N2 = N1 + N                                                       VAR0064
```

```
      CALL SUM(DUMMY,NA,W,NW,DUMMY(N1),NA,IOPT,SYM,DUMMY(N2))        VAR0065
C                                                                    VAR0066
  400 CONTINUE                                                       VAR0067
      IF( IOP(1) .EQ. 0 ) RETURN                                     VAR0068
      CALL LNCNT(3)                                                  VAR0069
      PRINT 450                                                      VAR0070
  450 FORMAT(/, ' VARIANCE MATRIX ',/)                               VAR0071
      CALL PRNT(W,NW,4H W   ,1)                                      VAR0072
C                                                                    VAR0073
      RETURN                                                         VAR0074
      END                                                            VAR0075
```

```
      SUBROUTINE CTROL(A,NA,B,NB,C,NC,IOP,IAC,IRANK,DUMMY)              CTR000
      IMPLICIT REAL*8 (A-H,O-Z)                                         CTR000
      DIMENSION A(1),B(1),C(1),DUMMY(1)                                 CTR000
      DIMENSION NA(2),NB(2),NC(2),NV(2),IOP(5)                          CTR000
      N = NA(1)*NB(2)                                                   CTR000
      N1 = N+1                                                          CTR000
      N2 = N1+N                                                         CTR000
      K = NA(1)-1                                                       CTR000
      J = 1                                                             CTR000
C                                                                       CTR001
      CALL EQUATE(B,NB,DUMMY(N2),NV)                                    CTR001
      CALL EQUATE(B,NB,DUMMY,NB)                                        CTR001
  100 CONTINUE                                                          CTR001
      CALL MULT(A,NA,DUMMY,NB,DUMMY(N1),NB)                             CTR001
      CALL JUXTC(DUMMY(N2),NV,DUMMY(N1),NB,C,NC)                        CTR001
C                                                                       CTR001
      IF( J .EQ. K ) GO TO 200                                          CTR001
C                                                                       CTR001
      CALL EQUATE(DUMMY(N1),NB,DUMMY,NB)                                CTR001
      CALL EQUATE(C,NC,DUMMY(N2),NV)                                    CTR002
      J = J + 1                                                         CTR002
      GO TO 100                                                         CTR002
C                                                                       CTR002
  200 CONTINUE                                                          CTR002
C                                                                       CTR002
      IF(IOP(1) .EQ. 0 ) GO TO 300                                      CTR002
      CALL PRNT(A,NA,4H A   ,1)                                         CTR002
      CALL PRNT(B,NB,4H B   ,1)                                         CTR002
      CALL LNCNT(4)                                                     CTR002
      PRINT 250                                                         CTR003
  250 FORMAT(//,' THE MATRIX C IS THE CONTROLLABILITY MATRIX FOR THE  A/CTR003
     1B PAIR',/)                                                        CTR003
      CALL PRNT(C,NC,4H C   ,1)                                         CTR003
C                                                                       CTR003
  300  IF( IOP(2) .EQ. 0 ) RETURN                                       CTR003
      NOS = 0                                                           CTR003
      IOPT = 2                                                          CTR003
      K = NC(2)                                                         CTR003
      NC(2) = NB(2)*(NA(2)-NB(2)+1)                                     CTR003
      N = NC(1)*NC(2)                                                   CTR004
      CALL TRANP(C,NC,DUMMY,NV)                                         CTR004
      NC(2) = K                                                         CTR004
      N1 = N + 1                                                        CTR004
      N2 = N1 + NV(2)                                                   CTR004
      CALL SNVDEC(IOPT,NV(1),NV(2),NV(1),NV(2),DUMMY,NOS,B,IAC,ZTEST,DUMCTR004
     1MY(N1),DUMMY(N2),IRANK,A,IERR)                                    CTR004
      IF( IERR .EQ. 0 ) GO TO 340                                       CTR004
      CALL LNCNT(5)                                                     CTR004
      IF( IERR .GT. 0 ) PRINT 310,IERR                                  CTR004
      IF( IERR .EQ. -1 ) PRINT 320, ZTEST,IRANK                         CTR005
  310 FORMAT(//,' IN CTROL, SNVDEC HAS FAILED TO CONVERGE TO THE ',I4, 'CTR005
     1 SINGULAR VALUE AFTER 30 ITERATIONS ')                            CTR005
  320 FORMAT(//,' IN CTROL, THE MATRIX SUBMITTED TO SNVDEC USING ZTEST =CTR005
     1',D16.8,' IS CLOSE TO A MATRIX WHICH IS OF LOWER RANK',/,' IF THE CTR005
     2ACCURACACY IS REDUCED THE RANK MAY ALSO BE REDUCED',/,' CURRENT RACTR005
     3NK =',I4)                                                         CTR005
       IF( IERR .GT. 0 ) RETURN                                         CTR005
C                                                                       CTR005
  340 CONTINUE                                                          CTR005
      IF( IOP(3) .EQ. 0 ) GO TO 400                                     CTR006
      CALL LNCNT(6)                                                     CTR006
      PRINT 350,ZTEST,IRANK                                             CTR006
  350 FORMAT(//,' BASED ON THE ZERO-TEST ',D16.8,' THE RANK OF THE CONTRCTR006
```

```
     1OLLABILITY MATRIX IS ',I4,/,' THE SINGULAR VALUES ARE ',/)
      IOPT = 0
      NV(1)= NV(2)
      NV(2)= 1
      CALL PRNT(DUMMY(N1),NV,IOPT,3)
C
  400 IF( IOP(4) .EQ. 0 ) RETURN
      N = NA(1)**2
      CALL EQUATE(DUMMY(N2),NA,DUMMY,NA)
      N1 = N + 1
      N2 = N1 + N
      CALL MULT(A,NA,DUMMY,NA,DUMMY(N1),NA)
      CALL TRANP(DUMMY,NA,DUMMY(N2),NA)
      CALL EQUATE(DUMMY(N2),NA,DUMMY,NA)
      CALL MULT(DUMMY,NA,DUMMY(N1),NA,DUMMY(N2),NA)
      CALL MULT(DUMMY,NA,B,NB,DUMMY(N1),NB)
C
      IF( IOP(5) .EQ. 0 ) RETURN
      CALL LNCNT(5)
      PRINT 500
  500 FORMAT(//,' CONTROLLABILITY CANONICAL FORM ',//,' (V TRANSPOSE) A
     1 V')
      CALL PRNT(DUMMY(N2),NA,IOPT,3)
      CALL LNCNT(2)
      PRINT 510
  510 FORMAT(/,' (V TRANSPOSE ) B ')
      CALL PRNT(DUMMY(N1),NB,IOPT,3)
      CALL LNCNT(2)
      PRINT 520
  520 FORMAT(/,' V TRANSPOSE')
      CALL PRNT(DUMMY,NA,IOPT,3)
C
      RETURN
      END
```

```
      SUBROUTINE TRNSIT(A,NA,B,NB,H,NH,G,NG,F,NF,V,NV,T,X,NX,DISC,STABL
     1E,IOP,DUMMY)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(1),B(1),H(1),G(1),F(1),V(1),X(1),DUMMY(1)
      DIMENSION NA(2),NB(2),NH(2),NG(2),NF(2),NV(2),NX(2),T(2),IOP(4)
      DIMENSION NDUM1(2),NDUM2(2)
      LOGICAL  DISC,STABLE
      N = NA(1)*NA(2)
      N1 = N + 1
      N2 = N + N1
      N3 = N + N2
      N4 = N + N3
      N5 = N + N4
      N6 = N + N5
C
      CALL LNCNT(4)
      IF(DISC) PRINT 100
      IF( .NOT. DISC ) PRINT 120
  100 FORMAT(//,' COMPUTATION OF TRANSIENT RESPONSE FOR THE DIGITAL SYST
     1EM ',/)
  120 FORMAT(//,' COMPUTATION OF TRANSIENT RESPONSE FOR THE CONTINUOUS
     1 SYSTEM',/)
      CALL PRNT(A,NA,4H A   ,1)
      CALL PRNT(B,NB,4H B   ,1)
      IF( (IOP(1) .NE. 1) .AND. (IOP(1) .NE. 0) )  GO TO 180
      CALL LNCNT(3)
      IF( IOP(1) .EQ. 0 ) PRINT 140
      IF( IOP(1) .EQ. 1 ) PRINT 160
  140 FORMAT(//,' H IS A NULL MATRIX ')
  160 FORMAT(//,' H IS AN IDENTITY MATRIX ')
      GO TO 200
  180 CONTINUE
      CALL PRNT(H,NH,4H H   ,1)
  200 CONTINUE
      IF( (IOP(2) .NE. 1) .AND. (IOP(2) .NE. 0) )  GO TO 260
      CALL LNCNT(3)
      IF( IOP(2) .EQ. 0 ) PRINT 220
      IF( IOP(2) .EQ. 1 ) PRINT 240
  220 FORMAT(//,' G IS A NULL MATRIX')
  240 FORMAT(//,' G IS AN IDENTITY MATRIX')
      GO TO 280
  260 CONTINUE
      CALL PRNT(G,NG,4H G   ,1)
  280 CONTINUE
      CALL PRNT(F,NF,4H F   ,1)
      IF( (IOP(3) .NE. 0) .AND. (IOP(3) .NE. 1) )  GO TO 295
      CALL LNCNT(3)
      IF(IOP(3).EQ.0) PRINT 285
      IF(IOP(3).EQ.1) PRINT 290
  285 FORMAT(//,' V IS A NULL MATRIX')
  290 FORMAT(//,' V IS AN IDENTITY MATRIX')
      GO TO 300
  295 CONTINUE
      CALL PRNT(V,NV,4H V   ,1)
C
  300 CONTINUE
      CALL EQUATE(A,NA,DUMMY(N6),NA)
      CALL MULT(B,NB,F,NF,DUMMY,NA)
      CALL SUBT(A,NA,DUMMY,NA,A,NA)
C
      IF(DISC) GO TO 350
      NMAX = T(1)/T(2)
      IOPT = 1
```

```
      TT = T(2)
      IF( IOP(3) .NE. 0 )  GO TO 315
      CALL EXPSER(A,NA,DUMMY,NA,TT,IOPT,DUMMY(N1))
      GO TO 400
  315 CONTINUE
      CALL EXPINT(A,NA,DUMMY,NA,DUMMY(N1),NA,TT,IOPT,DUMMY(N2))
      CALL MULT(DUMMY(N1),NA,B,NB,DUMMY(N2),NB)
      IF( IOP(3) .NE. 1 ) GO TO 325
      CALL EQUATE(DUMMY(N2),NB,DUMMY(N1),NX)
      GO TO 400
  325 CONTINUE
      CALL MULT(DUMMY(N2),NB,V,NV,DUMMY(N1),NX)
      GO TO 400
  350 CONTINUE
      NMAX = IOP(4)
      CALL EQUATE(A,NA,DUMMY,NA)
      IF( IOP(3) .EQ. 0 ) GO TO 400
      CALL MULT(B,NB,V,NV,DUMMY(N1),NX)
C
  400 CONTINUE
      CALL LNCNT(4)
      PRINT 420
  420 FORMAT(//,' STRUCTURE OF PRINTING TO FOLLOW',/)
      CALL LNCNT(6)
      PRINT 440
  440 FORMAT('   TIME OR STAGE ',/,'  STATE - X TRANSPOSE - FROM DX = AX
     1 + BU',/,'  OUTPUT - Y TRANSPOSE - FROM Y = HX + GU   IF DIFFERENT
     2 FROM X',/,'  CONTROL - U TRANSPOSE - FROM U = -FX + V',//)
C
      K = 0
      L = 0
      CALL SCALE(F,NF,F,NF,-1,0)
C
  450 CONTINUE
      IF( K .GT. NMAX ) GO TO 800
      CALL MULT(F,NF,X,NX,DUMMY(N2),NV)
      IF( IOP(3) .NE. 0 ) CALL ADD(DUMMY(N2),NV,V,NV,DUMMY(N2),NV)
      CALL MULT(DUMMY,NA,X,NX,DUMMY(N3),NX)
      IF( IOP(3) .EQ. 0 ) GO TO 475
      CALL ADD(DUMMY(N1),NX,DUMMY(N3),NX,DUMMY(N3),NX)
  475 CONTINUE
      IF( IOP(2) .EQ. 0 ) GO TO 525
      IF( IOP(2) .EQ. 1 ) GO TO 500
      CALL MULT(G,NG,DUMMY(N2),NV,DUMMY(N4),NDUM1)
      GO TO 525
  500 CONTINUE
      CALL EQUATE(DUMMY(N2),NV,DUMMY(N4),NDUM1)
  525 CONTINUE
      IF( IOP(1) .EQ. 0 ) GO TO 575
      IF( IOP(1) .EQ. 1 ) GO TO 550
      CALL MULT(H,NH,X,NX,DUMMY(N5),NDUM1)
      GO TO 575
  550 CONTINUE
      CALL EQUATE(X,NX,DUMMY(N5),NDUM1)
  575 CONTINUE
      IF( IOP(2) .EQ. 0 ) GO TO 600
      IF( IOP(1) .EQ. 0 ) GO TO 700
      CALL ADD(DUMMY(N4),NDUM1,DUMMY(N5),NDUM1,DUMMY(N4),NDUM1)
      GO TO 700
  600 CONTINUE
      IF( IOP(1) .NE. 0 ) CALL EQUATE(DUMMY(N5),NDUM1,DUMMY(N4),NDUM1)
C
  700 CONTINUE
```

```
      CALL LNCNT(5)
      IF( .NOT. DISC ) GO TO 720
      PRINT 710,K
  710 FORMAT(////,I5)
      GO TO 740
  720 CONTINUE
      TIME=K*T(2)
      PRINT 730,TIME
  730 FORMAT(////,D16.7)
  740 CONTINUE
      CALL TRANP(X,NX,DUMMY(N5),NDUM2)
      CALL PRNT(DUMMY(N5),NDUM2,L,3)
      IF( (IOP(2) .EQ. 0) .AND. ( (IOP(1) .EQ. 0) .OR. (IOP(1) .EQ. 1) )
     1) GO TO 750
      CALL TRANP(DUMMY(N4),NDUM1,DUMMY(N5),NDUM2)
      CALL PRNT(DUMMY(N5),NDUM2,L,3)
  750 CONTINUE
      CALL TRANP(DUMMY(N2),NV,DUMMY(N5),NDUM2)
      CALL PRNT(DUMMY(N5),NDUM2,L,3)
C
      CALL EQUATE(DUMMY(N3),NX,X,NX)
      K = K + 1
      GO TO 450
C
C
  800 CONTINUE
      CALL SCALE(F,NF,F,NF,-1.0)
      IF( .NOT. STABLE  .OR.  IOP(3) .EQ. 0  )  GO TO 900
      IF( IOP(3) .EQ. 1 )  GO TO 820
      CALL MULT(B,NB,V,NV,DUMMY,NX)
      GO TO 840
  820 CONTINUE
      CALL EQUATE(B,NB,DUMMY,NX)
  840 CONTINUE
      IF( .NOT. DISC )  GO TO 860
      CALL UNITY(DUMMY(N1),NA)
      CALL SUBT(DUMMY(N1),NA,A,NA,A,NA)
  860 CONTINUE
      IFAC = 0
      CALL GELIM(NA(1),NA(1),A,NX(2),DUMMY,DUMMY(N1),IFAC,DUMMY(N2),IERR
     1)
      IF( IERR .EQ. 0 )  GO TO 880
      CALL LNCNT(3)
      IF( .NOT. DISC )  PRINT 865
      IF( DISC )  PRINT 870
  865 FORMAT(//,' IN TRNSIT, THE MATRIX A-BF SUBMITTED TO GELIM IS SINGU
     1LAR')
  870 FORMAT(//,' IN TRNSIT, THE MATRIX  I - (A-BF) SUBMITTED TO GELIM I
     1S SINGULAR')
      GO TO 900
  880 CONTINUE
      IF( .NOT. DISC )  CALL SCALE(DUMMY,NX,DUMMY,NX,-1.0)
      CALL LNCNT(5)
      PRINT 890
  890 FORMAT(////,' STEADY-STATE VALUE OF  X TRANSPOSE')
      CALL TRANP(DUMMY,NX,DUMMY(N5),NDUM2)
      CALL PRNT(DUMMY(N5),NDUM2,L,3)
C
  900 CONTINUE
      CALL EQUATE(DUMMY(N6),NA,A,NA)
C
      RETURN
      END
```

```
      SUBROUTINE SAMPL(A,NA,B,NB,Q,NQ,R,NR,W,NW,T,IOP,DUMMY)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(1),B(1),Q(1),R(1),W(1),DUMMY(1)
      DIMENSION NA(2),NB(2),NQ(2),NR(2),NW(2),IOP(2),NDUM(2)
      COMMON/CONV/SUMCV,RICTCV,SERCV,MAXSUM
      IF(  IOP(1) .EQ. 0 ) GO TO 100
C
      IF(  IOP(2) .EQ. 0 ) GO TO 50
C
      CALL LNCNT(5)
      PRINT 25
   25 FORMAT(//,' COMPUTATION OF WEIGHTING MATRICES FOR THE OPTIMAL SAMP
     1LED-DATA REGULATOR PROBLEM',//)
      CALL PRNT(A,NA,4H A   ,1)
      CALL PRNT(B,NB,4H B   ,1)
      CALL LNCNT(3)
      PRINT 35
   35 FORMAT(/,' CONTINUOUS PERFORMANCE INDEX WEIGHTING MATRICES',/)
      CALL PRNT(Q,NQ,4H Q   ,1)
      CALL PRNT(R,NR,4H R   ,1)
      CALL LNCNT(3)
      PRINT 45,T
   45 FORMAT(/,' SAMPLE TIME = ',D16.8,/)
C
      GO TO 100
C
   50 CONTINUE
      CALL LNCNT(8)
      PRINT 75
   75 FORMAT(//,' COMPUTATION OF THE RECONSTRUCTIBILITY GRAMIAN',/,' FOR
     1 THE (A,H) SYSTEM OVER THE INTERVAL (0,T) ',/,' THE MATRIX Q IS  (
     2 H TRANSPOSE ) X H',//)
      CALL PRNT(A,NA,4H A   ,1)
      CALL PRNT(Q,NQ,4H Q   ,1)
      CALL LNCNT(3)
      PRINT 85,T
   85 FORMAT(/,' T = ',D16.8,/)
C
  100 CONTINUE
C
      N = NA(1)
      L = ( N**2)
      N1 = L + 1
      N2 = N1 + L
      TT  = T
C
      IOPT = 1
      CALL NORMS(N,N,N,A,IOPT,ANORM)
      IOPT = 3
      CALL NORMS(N,N,N,A,IOPT,ROWA)
      IF( ANORM .GT. ROWA ) ANORM = ROWA
C
      IF( ANORM .LE. 1.E-15 ) GO TO 900
C
      TMAX = 1.0/ANORM
      K = 0
C
  125 CONTINUE
      IF( TMAX - TT ) 150,150,200
  150 CONTINUE
      K = K + 1
      TT = T/( 2**K)
      IF( K - 1000 ) 125,800,800
```

```
C
  200 CONTINUE
C
      I = 0
      SC = TT
      CALL SCALE(A,NA,A,NA,TT)
      CALL SCALE(Q,NQ,Q,NQ,TT)
      CALL EQUATE(Q,NQ,DUMMY,NQ)
C
      IF( IOP(2) .NE. 0 ) GO TO 500
C
  225 CONTINUE
      II = I + 2
      I = I + 1
      F = 1.0/II
      CALL SCALE(A,NA,DUMMY(N1),NA,F)
      CALL MULT(DUMMY,NA,DUMMY(N1),NA,DUMMY(N2),NA)
      CALL TRANP(DUMMY(N2),NA,DUMMY(N1),NA)
      CALL ADD(DUMMY(N1),NA,DUMMY(N2),NA,DUMMY,NA)
C
      CALL MAXEL(Q,NQ,TOT)
      CALL MAXEL(DUMMY,NA,DELT)
      IF( TOT .GT. 1.0 ) GO TO 250
      IF( DELT/TOT .LT. SERCV )  GO TO 300
      GO TO 275
  250 CONTINUE
      IF( DELT .LT. SERCV )  GO TO 300
  275 CONTINUE
      CALL ADD(Q,NQ,DUMMY,NA,Q,NQ)
      GO TO 225
C
  300 CONTINUE
C
      IF( K .EQ. 0 ) GO TO 400
      N3 = N2 + L
      G = 1.0
      IOPT = 0
      CALL EXPSER(A,NA,DUMMY,NA,G,IOPT,DUMMY(N1))
C
  350 CONTINUE
      IF( K .EQ. 0 ) GO TO 400
      K = K-1
C
      CALL TRANP(DUMMY,NA,DUMMY(N1),NA)
      CALL MULT(Q,NQ,DUMMY,NA,DUMMY(N2),NA)
      CALL MULT(DUMMY(N1),NA,DUMMY(N2),NA,DUMMY(N3),NA)
      CALL ADD(Q,NQ,DUMMY(N3),NA,Q,NQ)
      CALL MULT(DUMMY,NA,DUMMY,NA,DUMMY(N1),NA)
      CALL EQUATE(DUMMY(N1),NA,DUMMY,NA)
C
      GO TO 350
C
  400 CONTINUE
      S =  1.0/SC
      CALL SCALE(A,NA,A,NA,S)
C
      IF( IOP(1) .EQ. 0 ) RETURN
      CALL PRNT(Q,NQ,4HGRAM,1)
      RETURN
C
  500 CONTINUE
      CALL SCALE(B,NB,B,NB,TT)
      N3 = N2 + L
```

```
      N4 = N3 + L                                                       SAM012
      N5 = N4 + L                                                       SAM012
      N6 = N5 + L                                                       SAM012
C                                                                       SAM013
  525 CONTINUE                                                          SAM013
      II = I + 2                                                        SAM013
      I = I + 1                                                         SAM013
      F = 1.0/II                                                        SAM013
      CALL SCALE(A,NA,DUMMY(N1),NA,F)                                   SAM013
      CALL TRANP(DUMMY(N1),NA,DUMMY(N2),NA)                             SAM013
      CALL MULT(DUMMY,NA,DUMMY(N1),NA,DUMMY(N3),NA)                     SJM013
      CALL TRANP(DUMMY(N3),NA,DUMMY(N1),NA)                             SJM013
      CALL MULT(DUMMY,NA,B,NB,DUMMY(N5),NW)                             SAM013
      CALL ADD(DUMMY(N1),NA,DUMMY(N3),NA,DUMMY,NA)                      SAM014
      CALL SCALE(DUMMY(N5),NW,DUMMY(N1),NW,F)                           SAM014
      IF( I .NE. 1 ) GO TO 550                                          SAM014
      CALL EQUATE(DUMMY(N1),NW,W,NW)                                    SJM014
      CALL EQUATE(DUMMY(N1),NW,DUMMY(N6),NW)                            SAM014
      CALL ADD(Q,NQ,DUMMY,NQ,Q,NQ)                                      SAM014
      GO TO 525                                                         SAM014
C                                                                       SAM014
  550 CONTINUE                                                          SAM014
      CALL MULT(DUMMY(N2),NA,DUMMY(N6),NW,DUMMY(N5),NW)                 SAM014
      CALL ADD(DUMMY(N5),NW,DUMMY(N1),NW,DUMMY(N1),NW)                  SAM015
      CALL TRANP(B,NB,DUMMY(N2),NDUM)                                   SAM015
      CALL SCALE(DUMMY(N2),NDUM,DUMMY(N2),NDUM,F)                       SAM015
      CALL MULT(DUMMY(N2),NDUM,DUMMY(N6),NW,DUMMY(N3),NR)               SAM015
      CALL TRANP(DUMMY(N3),NR,DUMMY(N5),NR)                             SAM015
      CALL ADD(DUMMY(N3),NR,DUMMY(N5),NR,DUMMY(N3),NR)                  SAM015
      CALL EQUATE(DUMMY(N1),NW,DUMMY(N6),NW)                            SAM015
      IF(  I .NE. 2 ) GO TO 575                                         SAM015
      CALL ADD(Q,NQ,DUMMY,NQ,Q,NQ)                                      SAM015
      CALL ADD(W,NW,DUMMY(N1),NW,W,NW)                                  SAM015
      CALL EQUATE(DUMMY(N3),NR,DUMMY(N4),NR)                            SAM016
      GO TO 525                                                         SJM016
C                                                                       SAM016
  575 CONTINUE                                                          SAM016
      CALL MAXEL(Q,NQ,TOT)                                              SAM016
      CALL MAXEL(DUMMY,NQ,DELT)                                         SAM016
      IF( TOT .GT. 1.0 )  GO TO 580                                     SAM016
      IF( DELT/TOT .LT. SEPCV )  GO TO 585                              SAM016
      GO TO 595                                                         SAM016
C                                                                       SAM016
  580 CONTINUE                                                          SAM017
      IF( DELT .LT. SERCV )  GO TO 585                                  SAM017
      GO TO 595                                                         SAM017
C                                                                       SAM017
  585 CONTINUE                                                          SAM017
      CALL MAXEL(DUMMY(N4),NR,TOT)                                      SAM017
      CALL MAXEL(DUMMY(N3),NR,DELT)                                     SAM017
      IF( TOT .GT. 1.0 )  GO TO 590                                     SAM017
      IF( DELT/TOT .LT. SERCV )  GO TO 600                              SAM017
      GO TO 595                                                         SAM017
C                                                                       SAM018
  590 CONTINUE                                                          SAM018
      IF( DELT .LT. SERCV )  GO TO 600                                  SAM018
C                                                                       SAM018
  595 CONTINUE                                                          SAM018
      CALL ADD (Q,NQ,DUMMY,NQ,Q,NQ)                                     SAM018
      CALL ADD(W,NW,DUMMY(N1),NW,W,NW)                                  SAM018
      CALL ADD(DUMMY(N4),NR,DUMMY(N3),NR,DUMMY(N4),NR)                  SAM018
      GO TO 525                                                         SAM018
C                                                                       SAM018
```

```
  600 CONTINUE
      IF( K .EQ. 0 ) GO TO 700
      G = 1.0
      IOPT = 0
      CALL EXPINT(A,NA,DUMMY,NA,DUMMY(N1),NA,G,IOPT,DUMMY(N2))
      CALL MULT(DUMMY(N1),NA,B,NB,DUMMY(N2),NB)
      CALL EQUATE(DUMMY(N2),NB,DUMMY(N1),NB)
C
  650 CONTINUE
      IF( K .EQ. 0 ) GO TO 700
      K = K - 1
      CALL MULT(Q,NQ,DUMMY,NA,DUMMY(N2),NA)
      CALL TRANP(DUMMY,NA,DUMMY(N3),NA)
      CALL MULT(DUMMY(N3),NA,DUMMY(N2),NA,DUMMY(N5),NA)
      CALL MULT(Q,NQ,DUMMY(N1),NB,DUMMY(N2),NB)
      CALL ADD(Q,NQ,DUMMY(N5),NA,Q,NQ)
      CALL MULT(DUMMY(N3),NA,DUMMY(N2),NB,DUMMY(N5),NB)
      CALL MULT(DUMMY(N3),NA,W,NW,DUMMY(N6),NW)
      CALL ADD(DUMMY(N5),NW,DUMMY(N6),NW,DUMMY(N5),NW)
      CALL TRANP(DUMMY(N1),NB,DUMMY(N6),NDUM)
      CALL MULT(DUMMY(N6),NDUM,W,NW,DUMMY(N3),NR)
      CALL ADD(W,NW,DUMMY(N5),NW,W,NW)
      CALL MULT(DUMMY(N6),NDUM,DUMMY(N2),NB,DUMMY(N5),NR)
      CALL ADD(DUMMY(N5),NR,DUMMY(N3),NR,DUMMY(N5),NR)
      CALL TRANP(DUMMY(N3),NR,DUMMY(N6),NR)
      CALL ADD(DUMMY(N5),NR,DUMMY(N6),NR,DUMMY(N6),NR)
      CALL SCALE(DUMMY(N4),NR,DUMMY(N4),NR,2.0)
      CALL ADD(DUMMY(N6),NR,DUMMY(N4),NR,DUMMY(N4),NR)
      CALL MULT(DUMMY,NA,DUMMY(N1),NB,DUMMY(N3),NB)
      CALL ADD(DUMMY(N3),NB,DUMMY(N1),NB,DUMMY(N1),NB)
      CALL MULT(DUMMY,NA,DUMMY,NA,DUMMY(N3),NA)
      CALL EQUATE(DUMMY(N3),NA,DUMMY,NA)
      GO TO 650
C
  700 CONTINUE
      CALL SCALE(R,NR,R,NR,T)
      CALL ADD(R,NR,DUMMY(N4),NR,R,NR)
      CALL SCALE(W,NW,W,NW,2.0)
      S = 1.0/SC
      CALL SCALE(A,NA,A,NA,S)
      CALL SCALE(B,NB,B,NB,S)
      IF( IOP(1) .EQ. 0 ) RETURN
C
      CALL LNCNT(3)
      PRINT 750
  750 FORMAT(/,' DISCRETE PERFORMANCE INDEX WEIGHTING MATRICES',/)
      CALL PRNT(Q,NQ,4H Q   ,1)
      CALL PRNT(W,NW,4H W   ,1)
      CALL PRNT(R,NR,4H R   ,1)
      RETURN
C
  800 CONTINUE
      CALL LNCNT(1)
      PRINT 850
  850 FORMAT(' ERROR IN SAMPL , K = 1000')
      RETURN
C
  900 CONTINUE
      CALL SCALE(Q,NQ,Q,NQ,T)
      IF( IOP(2) .NE. 0 ) GO TO 925
      IF( IOP(1) .NE. 0 ) CALL PRNT(Q,NQ,4HGRAM,1)
      RETURN
C
```

```
  925 CONTINUE                                                    SAM02
      CALL MULT(Q,NQ,B,NB,W,NW)                                   SAM02
      CALL SCALE(W,NW,W,NW,T)                                     SAM02
      CALL TRANP(B,NB,DUMMY,NDUM)                                 SAM02
      CALL MULT(DUMMY,NDUM,W,NW,DUMMY(N1),NR)                     SAM02
      TT = T/3.                                                   SJM02
      CALL SCALE(DUMMY(N1),NR,DUMMY,NR,TT)                        SAM02
      CALL SCALE(R,NR,R,NR,T)                                     SAM02
      CALL ADD(R,NR,DUMMY,NR,R,NR)                                SAM02
      IF( IOP(1) .EQ. 0 ) RETURN                                  SAM02
      CALL LNCNT(3)                                               SAM02
      PRINT 750                                                   SAM02
      CALL PRNT(Q,NQ,4H Q  ,1)                                    SAM02
      CALL PRNT(W,NW,4H W  ,1)                                    SAM02
      CALL PRNT(R,NR,4H R  ,1)                                    SAM02
      RETURN                                                      SAM02
C                                                                 SAM02
      END                                                         SAM02
```

```
      SUBROUTINE PREFIL(A,NA,B,NB,Q,NQ,W,NW,R,NR,F,NF,IOP,DUMMY)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(1),B(1),Q(1),W(1),R(1),F(1),DUMMY(1)
      DIMENSION NA(2),NB(2),NQ(2),NW(2),NR(2),NF(2),IOP(3)
      IF( IOP(1) .EQ. 0 ) GO TO 100
      CALL LNCNT(5)
      PRINT 25
   25 FORMAT(//,' PROGRAM TO COMPUTE PREFILTER GAIN F TO ELEMINATE  CROS
     1S-PRODUCT TERM ',/,' IN QUADRATIC PERFORMANCE INDEX ',/)
      IF( IOP(3) .EQ. 0 ) GO TO 50
      CALL PRNT(A,NA,4H A   ,1)
      CALL  PRNT(B,NB,4H B   ,1)
   50 CONTINUE
      CALL PRNT(Q,NQ,4H Q   ,1)
      CALL PRNT(W,NW,4H W   ,1)
      CALL PRNT(R,NR,4H R   ,1)
C
  100 CONTINUE
      CALL TRANP(W,NW,F,NF)
      CALL SCALE(F,NF,F,NF,0.5)
      CALL EQUATE(R,NR,DUMMY,NR)
      IOPT=0
      IFAC=0
      N1=NR(1)**2+1
      M = NR(1)
      CALL SYMPDS(M,M,DUMMY,NF(2),F,IOPT,IFAC,DETERM,ISCALE,DUMMY(N1),IE
     1RR)
      IF( IERR .EQ. 0 ) GO TO 200
      CALL LNCNT(4)
      PRINT 150
  150 FORMAT(//,' IN PREFIL, THE MATRIX R IS NOT SYMMETRIC POSITIVE DEFI
     1NITE',/)
      RETURN
C
  200 CONTINUE
      IF( IOP(2) .EQ. 0 ) GO TO 300
      CALL MULT(W,NW,F,NF,DUMMY,NQ)
      CALL SCALE(DUMMY,NQ,DUMMY,NQ,0.5)
      CALL SUBT(Q,NQ,DUMMY,NQ,Q,NQ)
C
  300 CONTINUE
      IF( IOP(3) .EQ. 0 ) GO TO 400
      CALL MULT(B,NB,F,NF,DUMMY,NA)
      CALL SUBT(A,NA,DUMMY,NA,A,NA)
C
  400 CONTINUE
      IF( IOP(1) .EQ. 0 ) RETURN
      CALL PRNT(F,NF,4H F   ,1)
      IF( IOP(2) .EQ. 0 ) GO TO 500
      CALL LNCNT(3)
      PRINT 450
  450 FORMAT(/, ' MATRIX  Q - (W/2)F ',/)
      CALL PRNT(Q,NQ,4HNEWQ,1)
C
  500 CONTINUE
      IF( IOP(3) .EQ. 0 ) RETURN
      CALL PRNT(A,NA,4HNEWA,1)
      RETURN
      END
```

```
      SUBROUTINE CSTAB(A,NA,B,NB,F,NF,IOP,SCLE,DUMMY)                   CST0
      IMPLICIT REAL*8 (A-H,O-Z)                                         CST0
      DIMENSION A(1),B(1),F(1),DUMMY(1)                                 CST0
      DIMENSION NA(2),NB(2),NF(2),IOP(3),NDUM(2)                        CST0
      DIMENSION IOPT(2)                                                 CST0
      LOGICAL SYM                                                       CST0
      COMMON/TOL/EPSAM,EPSBM,IACM                                       CST0
      N = NA(1)**2                                                      CST0
      N1=N+1                                                            CST0
C                                                                       CST0
      IF(IOP(2) .EQ. 0 ) GO TO 100                                      CST0
      CALL EQUATE(A,NA,DUMMY,NA)                                        CST0
      N2=N1+NA(1)                                                       CST0
      N3=N2+NA(1)                                                       CST0
      ISV =0                                                            CST 0
      ILV =0                                                            CST00
      CALL EIGEN(NA(1),NA(1),DUMMY,DUMMY(N1),DUMMY(N2),ISV,ILV,V,DUMMY(NCST00
     13),IERR)                                                          CST00
C                                                                       CST00
      M=NA(1)                                                           CST00
      IF(IERR .EQ. 0) GO TO 50                                          CST00
      CALL LNCNT(3)                                                     CST00
      PRINT 25,IERR                                                     CST00
   25 FORMAT(//,' IN CSTAB, THE SUBROUTINE EIGEN FAILED TO DETERMINE THECST00
     1 ',I4,' EIGENVALUE FOR THE MATRIX A  AFTER 30 ITERATIONS')        CST00
      IERR=1                                                            CST00
      CALL NORMS(M,M,M,A,IERR,BETA)                                     CST00
      BETA=2.*BETA                                                      CST00
      GO TO 200                                                         CST00
   50 CONTINUE                                                          CST00
C                                                                       CST00
      BETA = 0.0                                                        CST00
      DO 75 I = 1,M                                                     CST00
      J = N1 + I - 1                                                    CST00
      BETA1 = DABS(DUMMY(J))                                            CST00
      IF(BETA1 .GT. BETA)  BETA = BETA1                                 CST00
   75 CONTINUE                                                          CST00
      BETA = SCLE*(BETA + .001)                                         CST00
      GO TO 200                                                         CST00
C                                                                       CST004
  100 CONTINUE                                                          CST004
      BETA = SCLE                                                       CST004
  200 CONTINUE                                                          CST004
C                                                                       CST004
      CALL TRANP(B,NB,DUMMY,NDUM)                                       CST004
      CALL MULT(B,NB,DUMMY,NDUM,DUMMY(N1),NA)                           CST004
      CALL SCALE(DUMMY(N1),NA,DUMMY,NA,-2.0)                            CST004
      CALL SCALE(A,NA,DUMMY(N1),NA,-1.0)                                CST004
      J = -NA(1)                                                        CST004
      NAX = NA(1)                                                       CST005
      DO 225 I=1,NAX                                                    CST005
      J = J+NAX+1                                                       CST005
      K = N1+J-1                                                        CST005
      DUMMY(K)=DUMMY(K)-BETA                                            CST005
  225 CONTINUE                                                          CST005
      N2 = N1 + N                                                       CST005
      SYM = .TRUE.                                                      CST005
      IOPT(1)=0                                                         CST005
C                                                                       CST005
      IF( IOP(3) .NE. 0 )  GO TO 300                                    CST006
      EPSA=EPSAM                                                        CST006
      CALL BARSTW(DUMMY(N1),NA,A,NA,DUMMY,NA,IOPT,SYM,EPSA,EPSA,DUMMY(N2CST006
     1))                                                                CST006
```

```
      GO TO 350                                                         CST0
  300 CONTINUE                                                          CST0
      IOPT(2) = 1                                                       CST0
      CALL BILIN(DUMMY(N1),NA,A,NA,DUMMY,NA,IOPT,ASCLE,SYM,DUMMY(N2))   CST0
  350 CONTINUE                                                          CST0
C                                                                       CST0
      CALL EQUATE(B,NB,DUMMY(N1),NB)                                    CST0
      IOPT(1) = 3                                                       CST0
      IAC =IACM                                                         CST0
      N3 = N2 + NA(1)                                                   CST0
      CALL SNVDEC(IOPT,NA(1),NA(1),NA(1),NA(1),DUMMY,NB(2),DUMMY(N1),IACCST0
     1,ZTEST,DUMMY(N2),DUMMY(N3),IRANK,APLUS,IERR)                      CST0
      IF(IERR .EQ. 0 ) GO TO 400                                        CST0
      CALL LNCNT(5)                                                     CST0
      IF(IERR .GT. 0 ) PRINT 360,IERR                                   CST00
      IF(IERR .EQ. -1) PRINT 370,ZTEST,IRANK                            CST00
  360 FORMAT(//,' IN CSTAB, SNVDEC HAS FAILED TO CONVERGE TO THE ',I4,' CST00
     1SINGULARVALUE AFTER 30 ITERATIONS',//)                            CST00
  370 FORMAT(//,' IN CSTAB, THE MATRIX SUBMITTED TO SNVDEC USING ZTEST =CST00
     1 ',D16.8,' IS CLOSE TO A  MATRIX OF LOWER  RANK ',/,' IF THE ACCURCST00
     2ACY IAC IS REDUCED THE RANK MAY ALSO BE REDUCED',/,' CURRENT RANK CST00
     3 =',I4)                                                           CST00
      IF( IERR .GT. 0 ) RETURN                                          CST00
      NDUM(1) = NA(1)                                                   CST00
      NDUM(2) =1                                                        CST00
      CALL PRNT(DUMMY(N2),NDUM,4HSGVL,1)                                CST00
  400 CONTINUE                                                          CST00
C                                                                       CST00
      CALL TRANP(DUMMY(N1),NB,F,NF)                                     CST00
      IF ( IOP(1) .EQ. 0 )   RETURN                                     CST00
      CALL LNCNT(4)                                                     CST00
      PRINT 500                                                         CST00
  500 FORMAT(//,' COMPUTATION OF F MATRIX SUCH THAT A-BF IS ASYMPTOTICALCST00
     1LY STABLE IN THE CONTINUOUS SENSE ',/)                            CST00
      CALL PRNT(A,NA,4H A   ,1)                                         CST00
      CALL LNCNT(4)                                                     CST00
      PRINT 550,BETA                                                    CST01
  550 FORMAT(//,' BETA = ',D16.8,/)                                     CST01
      CALL PRNT(B,NB,4H B   ,1)                                         CST01
      CALL PRNT(F,NF,4H F   ,1)                                         CST01
      CALL MULT(B,NB,F,NF,DUMMY,NA)                                     CST01
      CALL SUBT(A,NA,DUMMY,NA,DUMMY,NA)                                 CST01
      CALL PRNT(DUMMY,NA,4HA-BF,1)                                      CST01
      N2 = N1+NA(1)                                                     CST01
      N3 = N2+NA(1)                                                     CST01
      ISV = 0                                                           CST01
      ILV = 0                                                           CST01
      CALL EIGEN(NA(1),NA(1),DUMMY,DUMMY(N1),DUMMY(N2),ISV,ILV,V,DUMMY(NCST011
     13),IERR)                                                          CST011
      M = NA(1)                                                         CST011
      IF( IERR .EQ. 0 ) GO TO 600                                       CST011
      M = NA(1)-IERR                                                    CST011
      CALL LNCNT(3)                                                     CST011
      PRINT 25,IERR                                                     CST011
  600 CONTINUE                                                          CST011
      CALL LNCNT(4)                                                     CST011
      PRINT 650                                                         CST012
  650 FORMAT(//,' EIGENVALUES OF A-BF',/)                               CST012
  675 FORMAT(10X,2D16.8)                                                CST012
      CALL LNCNT(M)                                                     CST012
      DO 700 I=1,M                                                      CST012
      J = N1+I-1                                                        CST012
      K = N2+I-1                                                        CST012
```

```
      PRINT 675,DUMMY(J),DUMMY(K)                                    CS
  700 CONTINUE                                                        CS
C                                                                     CS
      RETURN                                                          CST
      END                                                             CST
```

```
      SUBROUTINE DSTAB(A,NA,B,NB,F,NF,SING,IOP,SCLE,DUMMY)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(1),B(1),F(1),DUMMY(1)
      DIMENSION NA(2),NB(2),NF(2),NDUM(2),IOP(2),IOPT(3),NDUM1(2)
      LOGICAL SING,SYM
      COMMON/TOL/EPSAM,EPSBM,IACM
      N = NA(1)**2
      N1 = N + 1
      N2 = N1 + N
      IF( .NOT. SING ) GO TO 100
      IOPT(1)=IOP(1)
      IOPT(2) = 1
      IOPT(3) = 0
      CSCLE=1.05
      CALL CSTAB(A,NA,B,NB,F,NF,IOPT,CSCLE,DUMMY)
      CALL MULT(B,NB,F,NF,DUMMY,NA)
      CALL SUBT(A,NA,DUMMY,NA,DUMMY,NA)
      CALL EQUATE(DUMMY,NA,DUMMY(N1),NA)
      GO TO 200
C
  100 CONTINUE
      CALL EQUATE(A,NA,DUMMY,NA)
      CALL EQUATE(A,NA,DUMMY(N1),NA)
C
  200 CONTINUE
      IF( IOP(2) .EQ. 0 ) GO TO 300
      N3 = N2 + NA(1)
      N4 = N3 + NA(1)
      ISV = 0
      CALL EIGEN(NA(1),NA(1),DUMMY(N1),DUMMY(N2),DUMMY(N3),ISV,ISV,V,DUM
     1MY(N4),IERR)
      CALL EQUATE(DUMMY,NA,DUMMY(N1),NA)
      M = NA(1)
      IF( IERR .EQ. 0 ) GO TO 250
      CALL LNCNT(3)
      PRINT  225, IERR
  225 FORMAT(//'IN DSTAB , THE PROGRAM EIGEN FAILED TO DETERMINE',
     1I5, 'EIGENVALUE FOR THE MATRIX A-BG AFTER 30 ITERATIONS ')
      CALL PRNT(DUMMY,NA,4HA-BG,1)
      IF( SING ) CALL PRNT(F,NF,4H G   ,1)
      RETURN
C
  250 CONTINUE
      ALPHA = 1.0
      DO 275 I =1,M
      I1 = N2 + I -1
      I2 = N3 + I -1
      ALPHA1 = DSQRT(DUMMY(I1)**2 + DUMMY(I2)**2)
      IF( ALPHA1 .LT. ALPHA .AND. ALPHA1 .NE. 0 ) ALPHA = ALPHA1
  275 CONTINUE
      ALPHA = SCLE*ALPHA
      GO TO 400
C
  300 CONTINUE
      ALPHA = SCLE
C
  400 CONTINUE
      J = -NA(1)
      NAX = NA(1)
      DO 425 I = 1,NAX
      J = J + NAX + 1
      K = N1 + J -1
      DUMMY(K) = DUMMY(J) - ALPHA
```

```
      DUMMY(J) = DUMMY(J) + ALPHA
  425 CONTINUE
      CALL EQUATE(B,NB,DUMMY(N2),NB)
      N3 = N2 + NA(1)*NB(2)
      NRHS = NA(1)+NB(2)
      N4 = N3 + NA(1)
      IFAC = 0
      CALL GELIM(NA(1),NA(1),DUMMY,NRHS,DUMMY(N1),DUMMY(N3),IFAC,DUMMY(N
     14),IERR)
      IF( IERR .EQ. 0 ) GO TO 500
      CALL LNCNT(3)
      IF( .NOT. SING ) GO TO 445
      PRINT 435
  435 FORMAT(//,' IN DSTAB, GELIM HAS FOUND THE MATRIX ( A-BG) + (ALPHA)
     1I SINGULA[illegible]')
      CALL PRNT([illegible],4H A   ,1)
      CALL PRNT([illegible]F,4H G   ,1)
      GO TO 465
  445 CONTINUE
      CALL LNCNT(3)
      PRINT 455
  455 FORMAT(//,' IN DSTAB, GELIM HAS FOUND THE MATRIX A + (ALPHA)I SING
     1ULAR ')
      CALL PRNT(A,NA,4H A   ,1)
  465 CONTINUE
      CALL LNCNT(3)
      PRINT 475,ALPHA
  475 FORMAT(//,' ALPHA = ',D16.8)
      RETURN
C
  500 CONTINUE
      CALL EQUATE(DUMMY(N1),NA,DUMMY,NA)
      CALL TRANP(DUMMY(N2),NB,DUMMY(N1),NDUM)
      N3 = N2 + N
      CALL MULT(DUMMY(N2),NB,DUMMY(N1),NDUM,DUMMY(N3),NA)
      CALL SCALE(DUMMY(N3),NA,DUMMY(N1),NA,4.0)
      SYM = .TRUE.
      IOPT(1) = 0
      EPSA=EPSAM
      CALL BARSTW(DUMMY,NA,B,NB,DUMMY(N1),NA,IOPT,SYM,EPSA,EPSA,DUMMY(N2
     1))
      CALL EQUATE(DUMMY(N1),NA,DUMMY,NA)
      CALL TRANP(B,NB,DUMMY(N1),NDUM)
      CALL MULT(B,NB,DUMMY(N1),NDUM,DUMMY(N2),NA)
      CALL ADD(DUMMY,NA,DUMMY(N2),NA,DUMMY,NA)
      CALL EQUATE(A,NA,DUMMY(N1),NA)
      IF( .NOT. SING ) GO TO 600
      CALL MULT(B,NB,F,NF,DUMMY(N1),NA)
      CALL SUBT(A,NA,DUMMY(N1),NA,DUMMY(N1),NA)
C
  600 CONTINUE
      IOPT(1) = 3
      M = NA(1)
      IAC=IACM
      CALL SNVDEC(IOPT,M,M,M,M,DUMMY,M,DUMMY(N1),IAC,ZTEST,DUMMY(N2),DUM
     1MY(N3),IRANK,APLUS,IERR)
      IF( IERR  .EQ. 0 ) GO TO 700
      CALL LNCNT(5)
      IF( IERR .GT. 0 ) PRINT 625,IERR
      IF( IERR .EQ. -1) PRINT 650,ZTEST,IRANK
  625 FORMAT(//,' IN DSTAB, SNVDEC HAS FAILED TO CONVERGE TO THE ',I5,'
     1SINGULAR VALUE AFTER 30 ITERATIONS')
  650 FORMAT(//,' IN DSTAB, THE MATRIX SUBMITTED TO SNVDEC, USING ZTEST
```

```
     1= ',D16.8,' , IS CLOSE TO A MATRIX OF LOWER RANK',/,' IF THE ACCUR
     2ACY IAC IS REDUCED THE RANK MAY ALSO BE REDUCED',/,' CURRENT RANK
     3 =',I4)
      IF( IERR  .GT. 0 ) RETURN
      NDUM(1)= NA(1)
      NDUM(2)= 1
      CALL PRNT(DUMMY(N2),NDUM,4HSGVL,1)
C
  700 CONTINUE
      CALL TRANP(B,NB,DUMMY(N2),NDUM)
      CALL MULT(DUMMY(N2),NDUM,DUMMY(N1),NA,DUMMY,NF)
      IF( .NOT. SING ) GO TO 800
      CALL ADD(F,NF,DUMMY,NF,F,NF)
      GO TO 900
C
  800 CONTINUE
      CALL EQUATE(DUMMY,NF,F,NF)
C
  900 CONTINUE
      IF( IOP(1) .EQ. 0 ) RETURN
      CALL LNCNT(4)
      PRINT 1000
 1000 FORMAT(//,' COMPUTATION OF F SUCH THAT A-BF IS ASYMPTOTICALLY STAB
     1LE IN THE DISCRETE SENSE',/)
      CALL PRNT(A,NA,4H A  ,1)
      CALL PRNT(B,NB,4H B  ,1)
      CALL LNCNT(4)
      PRINT 1100,ALPHA
 1100 FORMAT(//,' ALPHA = ',D16.8,/)
      CALL PRNT(F,NF,4H F  ,1)
      CALL MULT(B,NB,F,NF,DUMMY,NA)
      CALL SUBT(A,NA,DUMMY,NA,DUMMY,NA)
      CALL PRNT(DUMMY,NA,4HA-BF,1)
      CALL LNCNT(3)
      PRINT 1200
 1200 FORMAT(//,' EIGENVALUES OF A-BF')
      NDUM(1) = NA(1)
      NDUM(2) = 1
      N2 = N1 + NA(1)
      N3 = N2 + NA(1)
      ISV = 0
      CALL EIGEN(NA(1),NA(1),DUMMY,DUMMY(N1),DUMMY(N2),ISV,ISV,V,DUMMY(N
     13),IERR)
      IF( IERR .EQ. 0 ) GO TO 1300
      CALL LNCNT(3)
      PRINT 1250
 1250 FORMAT(//,' IN DSTAB, THE PROGRAM EIGEN FAILED TO DETERMINE THE ',
     1I5,' EIGENVALUE FOR THE A-BF MATRIX AFTER 30 ITERATIONS')
      NDUM(1)=NA(1)-IERR
C
 1300 CONTINUE
      CALL JUXTC(DUMMY(N1),NDUM,DUMMY(N2),NDUM,DUMMY,NDUM1)
      CALL PRNT(DUMMY,NDUM1,4HEIGN,1)
      CALL LNCNT(4)
      PRINT 1400
 1400 FORMAT(//,' MODULI OF EIGENVALUES OF A-BF',/)
      M =NDUM(1)
      DO 1500 I = 1,M
      J = N1 + I - 1
      K = N2 + I - 1
      DUMMY(I)=DSQRT(DUMMY(J)**2 + DUMMY(K)**2)
 1500 CONTINUE
      CALL PRNT(DUMMY,NDUM,4HMOD ,1)
```

```
      C                                                                 DST
            RETURN                                                      DST
            END                                                         DST
```

```
      SUBROUTINE DISREG(A,NA,B,NB,H,NH,Q,NQ,R,NR,F,NF,P,NP,IOP,IDENT,DU
     1MMY)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(1),B(1),Q(1),R(1),F(1),P(1),DUMMY(1)
      DIMENSION NA(2),NB(2),NQ(2),NR(2),NF(2),NP(2)
      DIMENSION IOP(3)
      DIMENSION H(1),NH(2),NDUM(2)
      LOGICAL  IDENT
      COMMON/TOL/EPSAM,EPSBM,IACM
      COMMON/CONV/SUMCV,RICTCV,SERCV,MAXSUM
      N = NA(1)**2
      N1= N +1
      N2= N1+N
      N3= N2+N
C
      KSS = 0
      I=IOP(3)
C
      IF(IOP(1) .EQ. 0)  GO TO 85
      CALL LNCNT(5)
      PRINT 25
   25 FORMAT(//,' PROGRAM TO SOLVE THE TIME-INVARIANT FINITE-DURATION OP
     1TIMAL',/,' DIGITAL REGULATOR PROBLEM WITH NOISE-FREE MEASUREMENTS'
     2,/)
      CALL PRNT(A,NA,4H A  ,1)
      CALL PRNT(B,NB,4H B  ,1)
      CALL PRNT(Q,NQ,4H Q  ,1)
      IF( .NOT. IDENT )  GO TO 45
      CALL LNCNT(3)
      PRINT 35
   35 FORMAT(/,' H IS AN IDENTITY MATRIX',/)
      GO TO 65
   45 CONTINUE
      CALL PRNT(H,NH,4H H  ,1)
      CALL MULT(Q,NQ,H,NH,DUMMY,NH)
      CALL TRANP(H,NH,DUMMY(N1),NF)
      CALL MULT(DUMMY(N1),NF,DUMMY,NH,Q,NQ)
      CALL LNCNT(3)
      PRINT 55
   55 FORMAT(/,' MATRIX ( H TRANSPOSE )QH',/)
      CALL PRNT(Q,NQ,4HHTQH,1)
   65 CONTINUE
      CALL PRNT(R,NR,4H R  ,1)
      CALL LNCNT(4)
      PRINT 75
   75 FORMAT(//,' WEIGHTING ON TERMINAL VALUE OF STATE VECTOR',/)
      CALL PRNT(P,NP,4H P  ,1)
C
   85 CONTINUE
      IF((IOP(1) .NE. 0)  .OR. IDENT)  GO TO 100
      CALL MULT(Q,NQ,H,NH,DUMMY,NH)
      CALL TRANP(H,NH,DUMMY(N1),NF)
      CALL MULT(DUMMY(N1),NF,DUMMY,NH,Q,NQ)
C
  100 CONTINUE
      I=I-1
      CALL EQUATE(P,NP,DUMMY,NP)
      CALL MULT(P,NP,A,NA,DUMMY(N1),NA)
      CALL TRANP(B,NB,DUMMY(N2),NF)
      CALL MULT(DUMMY(N2),NF,DUMMY(N1),NA,F,NF)
      CALL MULT(P,NP,B,NB,DUMMY(N1),NB)
      CALL MULT(DUMMY(N2),NF,DUMMY(N1),NB,DUMMY(N3),NR)
      CALL ADD(R,NR,DUMMY(N3),NR,DUMMY(N1),NR)
```

```
      IOPT = 3
      IAC=IACM
      MF = NP(1)
      CALL SNVDEC(IOPT,MF,MF,MF,MF,DUMMY(N1),NF(2),F,IAC,ZTEST,DUMMY(N2)
     1,DUMMY(N3),IRANK,APLUS,IERR)
      IF( IERR .EQ.0) GO TO 300
      CALL LNCNT(5)
      IF(IERR .GT. 0) PRINT 200,IERR
      IF(IERR  .EQ. -1) PRINT 250,ZTEST,IRANK
  200 FORMAT(//,' IN DISREG, SNVDEC HAS FAILED TO CONVERGE TO THE ',I4,
     1'SINGULARVALUE AFTER 30 ITERATIONS',//)
  250 FORMAT(//,' IN DISREG, THE MATRIX SUBMITTED TO SNVDEC USING ZTEST
     1=',D16.8,' IS CLOSE TO A MATRIX OF LOWER RANK',/,'IF  THE ACCURACY
     2 IAC IS REDUCED THE RANK MAY ALSO BE REDUCED',/,' CURRENT RANK = '
     3 ,I4)
      IF( IERR .GT. 0 ) RETURN
      NDUM(1) = NA(1)
      NDUM(2) =  1
      CALL PRNT(DUMMY(N2),NDUM,4HSGVL,1)
C
  300 CONTINUE
      CALL MULT(R,NR,F,NF,DUMMY(N1),NF)
      CALL TRANP(F,NF,DUMMY(N2),NB)
      CALL MULT(DUMMY(N2),NB,DUMMY(N1),NF,P,NP)
      CALL ADD(Q,NQ,P,NP,P,NP)
      CALL MULT(B,NB,F,NF,DUMMY(N1),NA)
      CALL SUBT(A,NA,DUMMY(N1),NA,DUMMY(N1),NA)
      CALL MULT(DUMMY,NA,DUMMY(N1),NA,DUMMY(N2),NA)
      CALL TRANP(DUMMY(N1),NA,DUMMY(N3),NA)
      CALL MULT(DUMMY(N3),NA,DUMMY(N2),NA,DUMMY(N1),NA)
      CALL ADD(P,NP,DUMMY(N1),NA,P,NP)
C
      IF( IOP(2) .EQ. 0 )  GO TO 400
      CALL LNCNT(5)
      PRINT 350,I
  350 FORMAT(///,' STAGE ',I5,/)
      CALL PRNT(F,NF,4H F   ,1)
      CALL PRNT(P,NP,4H P   ,1)
C
  400 CONTINUE
      IF( I .EQ. 0 )  GO TO 600
      CALL MAXEL(DUMMY,NP,ANORM1)
      CALL SUBT(DUMMY,NP,P,NP,DUMMY(N2),NP)
      CALL MAXEL(DUMMY(N2),NP,ANORM2)
      IF( ANORM1 .NE. 0.0 )  GO TO 500
      GO TO 100
C
  500 CONTINUE
      IF(ANORM1 .GT. 1.0 ) GO TO 550
      IF( ANORM2/ANORM1 .LT. RICTCV ) KSS = 1
      GO TO 575
  550 CONTINUE
      IF( ANORM2 .LT. RICTCV ) KSS=1
  575 CONTINUE
      IF( KSS .EQ. 1) GO TO 600
      GO TO 100
C
  600 CONTINUE
      K = IOP(1) + IOP(2)
      IF( K .EQ. 0 ) RETURN
      IF( KSS .EQ. 0) GO TO 700
      CALL LNCNT(4)
      PRINT 650
```

```
  650 FORMAT(//,' STEADY-STATE SOLUTION HAS BEEN REACHED IN  DISREG',/) DIS01
C                                                                      DIS01
  700 CONTINUE                                                         DIS01
      IF( IOP(2) .NE. 0 )  RETURN                                      DIS01
      IF( IOP(1) .EQ. 0 )  RETURN                                      DIS01
      CALL LNCNT(3)                                                    DIS01
      I = IOP(3)-I                                                     DIS01
      PRINT 800, I                                                     DIS01
  800 FORMAT(/,' F AND P AFTER ',I5, ' STEPS',/)                       DIS01
      CALL PRNT(F,NF,4H F  ,1)                                         DIS01
      CALL PRNT(P,NP,4H P  ,1)                                         DIS01
      RETURN                                                           DIS01
      END                                                              DIS01
```

```
      SUBROUTINE CNTREG(A,NA,B,NB,H,NH,Q,NQ,R,NR,Z,W,LAMBDA,S,F,NF,P,NP
     1,T,IOP,IDENT,DUMMY)
      IMPLICIT REAL*8 (A-H,O-Z)
C
      DIMENSION A(1),B(1),H(1),Q(1),R(1),Z(1),W(1),LAMBDA(1),S(1),F(1),P
     1(1),T(1),DUMMY(1)
      DIMENSION NA(2),NB(2),NH(2),NQ(2),NR(2),NF(2),NP(2),IOP(3),NDUM1(2
     1),NDUM2(2)
      LOGICAL IDENT
      REAL*8 LAMBDA
      COMMON/CONV/SUMCV,RICTCV,SERCV,MAXSUM
C
      IF( IOP(1) .EQ. 0 ) GO TO 65
      CALL LNCNT(5)
      IF( IOP(3) .EQ. 0 ) PRINT 25
   25 FORMAT(//' PROGRAM TO SOLVE THE TIME-INVARIANT FINITE-DURATION CON
     1TINUOUS OPTIMAL'/' REGULATOR PROBLEM WITH NOISE-FREE MEASUREMENTS'
     2)
      IF( IOP(3) .NE. 0 ) PRINT 30
   30 FORMAT(//' PROGRAM TO SOLVE THE TIME-INVARIANT INFINITE-DURATION C
     1ONTINUOUS OPTIMAL'/' REGULATOR PROBLEM  WITH NOISE-FREE MEASUREMEN
     2TS')
      CALL PRNT(A,NA,4H A   ,1)
      CALL PRNT(B,NB,4H B   ,1)
      CALL PRNT(Q,NQ,4H Q   ,1)
      IF( .NOT. IDENT ) GO TO 45
      CALL LNCNT(3)
      PRINT 35
   35 FORMAT(/' H IS AN IDENTITY MATRIX'/)
      GO TO 55
C
   45 CONTINUE
      CALL PRNT(H,NH,4H H   ,1)
      CALL MULT(Q,NQ,H,NH,DUMMY,NH)
      N1= NH(1)*NH(2)+1
      CALL TRANP(H,NH,DUMMY(N1),NDUM1)
      CALL MULT(DUMMY(N1),NDUM1,DUMMY,NH,Q,NQ)
      CALL LNCNT(3)
      PRINT 50
   50 FORMAT(//' MATRIX (H TRANSPOSE)QH')
      CALL PRNT(Q,NQ,0,3)
   55 CONTINUE
      CALL PRNT(R,NR,4H R   ,1)
C
      IF( IOP(3) .NE. 0 ) GO TO 65
      CALL LNCNT(4)
      PRINT 60
   60 FORMAT(//' WEIGHTING ON TERMINAL VALUE OF STATE VECTOR'/)
      CALL PRNT(P,NP,4H P   ,1)
C
   65 CONTINUE
      CALL EQUATE(R,NR,DUMMY,NR)
      N = NA(1)**2
      N1 = NB(1)*NB(2)+1
      CALL TRANP(B,NB,DUMMY(N1),NDUM1)
      N2 = N1 + N
      L = NR(1)
      IOPT = 0
      IFAC = 0
      CALL SYMPDS(L,L,DUMMY,NB(1),DUMMY(N1),IOPT,IFAC,DET,ISCALE,DUMMY(N
     12),IERR)
C
      IF( IERR .EQ. 0 ) GO TO 100
```

```
      CALL LNCNT(4)
      PRINT 75
   75 FORMAT(//' IN CNTREG, THE SUBROUTINE  SYMPOS HAS FOUND THE MATRIX
     1 R NOT SYMMETRIC POSITIVE DEFINITE'/)
      RETURN
C
  100 CONTINUE
      CALL EQUATE(DUMMY(N1),NDUM1,DUMMY,NDUM1)
      CALL MULT(B,NB,DUMMY(N1),NDUM1,DUMMY(N2),NA)
      CALL SCALE(DUMMY(N2),NA,DUMMY(N1),NA,-1.0)
      N3 = N2 + N
      IF( IDENT .OR. (IOP(1) .NE. 0) ) GO TO 200
      CALL MULT(Q,NQ,H,NH,DUMMY(N2),NH)
      CALL TRANP(H,NH,DUMMY(N3),NDUM1)
      CALL MULT(DUMMY(N3),NDUM1,DUMMY(N2),NH,Q,NQ)
C
  200 CONTINUE
      CALL SCALE(Q,NQ,Q,NQ,-1.0)
      CALL JUXTR(A,NA,Q,NQ,Z,NDUM1)
      CALL TRANP(A,NA,DUMMY(N2),NA)
      CALL SCALE( DUMMY(N2),NA,DUMMY(N2),NA,-1.0)
      L = 2*N + 1
      CALL JUXTR(DUMMY(N1),NA,DUMMY(N2),NA,Z(L),NDUM1)
      CALL SCALE(Q,NQ,Q,NQ,-1.0)
      NDUM2(1) = 2*NA(1)
      NDUM2(2) = NDUM2(1)
      IF( IOP(1) .NE. 0 )  CALL PRNT(Z,NDUM2,4H Z  ,1)
      CALL EQUATE(Z,NDUM2,DUMMY(N1),NDUM2)
      M = 4*N
      N2 = M + N1
      L = 2*NA(1)
      N3 = N2 + L
      N4 = N3 + L
      ISV = L
      ILV = 0
      CALL EIGEN(L,L,DUMMY(N1),DUMMY(N2),DUMMY(N3),ISV,ILV,W,DUMMY(N4),I
     1ERR)
      IF( IERR .EQ. 0 ) GO TO 300
      CALL LNCNT(4)
      IF( IERR .GT. 0 ) GO TO 250
      PRINT 225,IERR
  225 FORMAT(//' IN CNTREG, EIGEN FAILED TO COMPUTE THE ',I6,' EIGENVEC
     1TOR OF Z '/)
      RETURN
  250 CONTINUE
      PRINT 275,IERR
  275 FORMAT(//' IN CNTREG, THE ',I6,' EIGENVALUE OF Z HAS NOT BEEN FO
     1UND AFTER 30 ITERATIONS IN EIGEN'/)
      RETURN
C
  300 CONTINUE
      IF( IOP(1) .EQ. 0 ) GO TO 400
      CALL LNCNT(3)
      PRINT 325
  325 FORMAT(//' EIGENVALUES OF Z')
      NDUM1(1) = L
      NDUM1(2) = 2
      CALL PRNT(DUMMY(N2),NDUM1,0,3)
      CALL LNCNT(3)
      PRINT 350
  350 FORMAT(//' CORRESPONDING EIGENVECTORS')
      CALL PRNT(W,NDUM2,0,3)
C
```

```
  400 CONTINUE
      CALL EQUATE(W,NDUM2,DUMMY(N1),NDUM2)
      J1 = 1
      J2 = 1
      M = 2*N
      NDUM1(1) = L
      NDUM1(2) = 1
      K4 = N4
C
      I=1
  415 CONTINUE
      IF( I .GT. L )  GO TO 515
      K1 = N2+I-1
      K2 = N1+(I-1)*L
      K3 = N3+I-1
      IF(DUMMY(K1) .GT. 0.0 ) GO TO 425
      J = (J1-1)*L+M+1
      J1 = J1+1
      IF(DUMMY(K3).NE. 0.0) J1=J1+1
      GO TO 450
  425 CONTINUE
      DUMMY(K4)=I
      K4 = K4+1
      J = (J2-1)*L+1
      J2 = J2+1
      IF( DUMMY(K3) .NE. 0.0 )  J2 = J2 + 1
  450 CONTINUE
      CALL EQUATE(DUMMY(K2),NDUM1,W(J),NDUM1)
      IF(DUMMY(K3) .EQ. 0.0) GO TO 500
      I = I+1
      K2 = K2+L
      J = J+L
      CALL EQUATE(DUMMY(K2),NDUM1,W(J),NDUM1)
  500 CONTINUE
      I=I+1
      GO TO 415
  515 CONTINUE
C
      CALL NULL(LAMBDA,NA)
      K0 = -1
      J = -NA(1)
      NAX = NA(1)
      I=1
  520 CONTINUE
      IF( I .GT. NAX )  GO TO 530
      J = NAX + J + 1
      K0 = K0 + 1
      K1 = N4 + K0
      K2 = DUMMY(K1)
      K = N2+K2-1
      LAMBDA(J) = DUMMY(K)
      K3 = N3+K2-1
      IF( DUMMY(K3) .EQ. 0.0 ) GO TO 525
      K4 = J+1
      LAMBDA(K4) = -DUMMY(K3)
      K4 = K4+NAX
      LAMBDA(K4) = DUMMY(K)
      K4 = K4-1
      LAMBDA(K4) = DUMMY(K3)
      K5 = M + (I-1)*L + 1
      K6 = K5 + L
      CALL EQUATE(W(K5),NDUM1,DUMMY(N1),NDUM1)
      CALL EQUATE(W(K6),NDUM1,W(K5),NDUM1)
```

```
      CALL EQUATE(DUMMY(N1),NDUM1,W(K6),NDUM1)                          CNTO
      I = I+1                                                           CNTO
      J = NAX + J +1                                                    CNTO
  525 CONTINUE                                                          CNTO
      I=I+1                                                             CNTO
      GO TO 520                                                         CNTO
  530 CONTINUE                                                          CNTO
C                                                                       CNTO
      IF( IOP(1) .EQ. 0 ) GO TO 700                                     CNTO
      CALL LNCNT(3)                                                     CNTO
      PRINT 535                                                         CNTO
  535 FORMAT(//' REORDERED EIGENVECTORS')                               CNTO
      CALL PRNT(W,NDUM2,0,3)                                            CNTO
      CALL LNCNT(4)                                                     CNTO
      PRINT 545                                                         CNTO
  545 FORMAT(//' LAMBDA MATRIX OF EIGENVALUES OF Z WITH POSITIVE REAL PACNTO
     1RTS'/)                                                            CNTO
      CALL PRNT(LAMBDA,NA,0,3)                                          CNTO
C                                                                       CNTO
      CALL MULT(Z,NDUM2,W,NDUM2,DUMMY(N1),NDUM2)                        CNTO
      L = NDUM2(1)                                                      CNTO
      M = L**2                                                          CNTO
      N2 = N1+M                                                         CNTO
      CALL EQUATE(W,NDUM2,DUMMY(N2),NDUM2)                              CNTO
      N3 = N2+M                                                         CNTO
      N4 = N3+L                                                         CNTO
      IFAC = 0                                                          CNTO
      CALL GELIM(L,L,DUMMY(N2),L,DUMMY(N1),DUMMY(N3),IFAC,DUMMY(N4),IERRCNTO
     1)                                                                 CNTO
      IF( IERR .EQ. 0 ) GO TO 600                                       CNTO
      CALL LNCNT(4)                                                     CNTO
      PRINT 550                                                         CNTO
  550 FORMAT(//' IN CNTREG, GELIM HAS FOUND THE REORDERED MATRIX W TO B CNTO
     1E SINGULAR '/)                                                    CNTO
  600 CONTINUE                                                          CNTO
      CALL PRNT(DUMMY(N1),NDUM2,4HWIZW,1)                               CNTO
C                                                                       CNTO
  700 CONTINUE                                                          CNTO
      NDUM1(1) = 2*NA(1)                                                CNTO
      NDUM1(2) = NA(1)                                                  CNTO
      N2 = 2*N + N1                                                     CNTO
      CALL TRANP(W,NDUM1,DUMMY(N2),NDUM2)                               CNTO
      NW11 = N1                                                         CNTO
      NDUM1(1) = NA(1)                                                  CNTO
      CALL TRANP(DUMMY(N2),NDUM1,DUMMY(NW11),NDUM1)                     CNTO
      L = N2+N                                                          CNTO
      NW21 = NW11+N                                                     CNTO
      CALL TRANP(DUMMY(L),NDUM1,DUMMY(NW21),NDUM1)                      CNTO
      L = 2*N+1                                                         CNTO
      NDUM1(1)=2*NA(1)                                                  CNTO
      N3 = N2 + 2*N                                                     CNTO
      CALL TRANP(W(L),NDUM1,DUMMY(N3),NDUM2)                            CNTO
      NDUM1(1) = NA(1)                                                  CNTO
      NW12 = NW21+N                                                     CNTO
      CALL TRANP(DUMMY(N3),NDUM1,DUMMY(NW12),NDUM1)                     CNTO
      L = N3 + N                                                        CNTO
      NW22 = NW12 + N                                                   CNTO
      CALL TRANP(DUMMY(L),NDUM1,DUMMY(NW22),NDUM1)                      CNTO
C                                                                       CNTO
      IF( IOP(1) .EQ. 0 ) GO TO 800                                     CNTO
      CALL PRNT(DUMMY(NW11),NA,4HW11 ,1)                                CNTO
      CALL PRNT(DUMMY(NW21),NA,4HW21 ,1)                                CNTO
      CALL PRNT(DUMMY(NW12),NA,4HW12 ,1)                                CNTO
```

```
      CALL PRNT(DUMMY(NW22),NA,4HW22 ,1)
C
  800 CONTINUE
      IF( IOP(3) .NE. 0 ) GO TO 900
      N2 = N1+4*N
      CALL MULT(P,NP,DUMMY(NW12),NA,S,NA)
      CALL MULT(P,NP,DUMMY(NW11),NA,DUMMY(N2),NA)
      CALL SUBT(S,NA,DUMMY(NW22),NA,S,NA)
      CALL SUBT(DUMMY(NW21),NA,DUMMY(N2),NA,DUMMY(N2),NA)
      N3 = N2+N
      L = NA(1)
      IFAC = 0
      N4 = N3+NA(1)
      CALL GELIM(L,L,DUMMY(N2),L,S,DUMMY(N3),IFAC,DUMMY(N4),IERR)
      IF( IERR .EQ. 0 ) GO TO 850
      CALL LNCNT(4)
      PRINT 825
  825 FORMAT(//' IN CNTREG, GELIM HAS FOUND THE MATRIX  W21 - P1XW11 TO
     1 BE SINGULAR'/)
      RETURN
C
  850 CONTINUE
      IF( IOP(1) .EQ. 0 ) GO TO 1000
      CALL PRNT(S,NA,4H S   ,1)
      NDUM1(1) = NR(1)
      NDUM1(2) = NA(1)
      CALL LNCNT(3)
      PRINT 875
  875 FORMAT(//' MATRIX (R INVERSE)X(B TRANSPOSE)')
      CALL PRNT(DUMMY,NDUM1,0,3)
      GO TO 1000
C
  900 CONTINUE
      N2 = N1+4*N
      CALL TRANP(DUMMY(NW12),NA,DUMMY(N2),NA)
      CALL TRANP(DUMMY(NW22),NA,P,NP)
      N3 = N2+N
      IFAC = 0
      L = NA(1)
      N4 = N3 + NA(1)
      CALL GELIM(L,L,DUMMY(N2),L,P,DUMMY(N3),IFAC,DUMMY(N4),IERR)
      IF( IERR .EQ. 0 ) GO TO 950
      CALL LNCNT(4)
      PRINT 925
  925 FORMAT(//' IN CNTREG, GELIM HAS FOUND THE MATRIX W12 TO BE SINGUL
     1AR'/)
      RETURN
  950 CONTINUE
      NDUM1(1) = NR(1)
      NDUM1(2) = NA(1)
      CALL MULT(DUMMY,NDUM1,P,NP,F,NF)
      IF( IOP(1) .EQ. 0 ) RETURN
      CALL PRNT(P,NP,4H P   ,1)
      CALL PRNT(F,NF,4H F   ,1)
      RETURN
C
 1000 CONTINUE
      NMAX = T(1)/T(2)
      I = NMAX
      CALL EQUATE(LAMBDA,NA,DUMMY(N2),NA)
      TT = -T(2)
      N4 = N3+N
      N5 = N4+N
```

```
      N6 = N5+N
      N7 = N6+NA(1)
      KSS = 0
      NDUM1(1) = NR(1)
      NDUM1(2) = NA(1)
      CALL EXPSER(DUMMY(N2),NA,DUMMY(N3),NA,TT,KSS,DUMMY(N4))
      CALL EQUATE(DUMMY(N3),NA,DUMMY(N2),NA)
      IF( IOP(1) .EQ. 0 )  GO TO 1075
      CALL LNCNT(3)
      PRINT 1050,T(2)
 1050 FORMAT(//' EXP(-LAMBDA X ',D16.8,')')
      CALL PRNT(DUMMY(N2),NA,0,3)
 1075 CONTINUE
      IF( NMAX .LE. 0 )  RETURN
      CALL EQUATE(S,NA,DUMMY(N3),NA)
 1100 CONTINUE
      TIME = I*T(2)
      IF( I .NE. NMAX )  CALL EQUATE(DUMMY(N5),NA,P,NP)
      CALL MULT(DUMMY(N3),NA,DUMMY(N2),NA,DUMMY(N4),NA)
      CALL MULT(DUMMY(N2),NA,DUMMY(N4),NA,DUMMY(N3),NA)
      CALL MULT(DUMMY(NW11),NA,DUMMY(N3),NA,DUMMY(N4),NA)
      CALL ADD(DUMMY(NW12),NA,DUMMY(N4),NA,DUMMY(N4),NA)
      CALL TRANP(DUMMY(N4),NA,DUMMY(N5),NA)
      CALL EQUATE(DUMMY(N5),NA,DUMMY(N4),NA)
      CALL MULT(DUMMY(NW21),NA,DUMMY(N3),NA,DUMMY(N5),NA)
      CALL ADD(DUMMY(NW22),NA,DUMMY(N5),NA,DUMMY(N5),NA)
      CALL TRANP(DUMMY(N5),NA,DUMMY(N6),NA)
      CALL EQUATE(DUMMY(N6),NA,DUMMY(N5),NA)
      L = NA(1)
      IFAC = 0
      CALL GELIM(L,L,DUMMY(N4),L,DUMMY(N5),DUMMY(N6),IFAC,DUMMY(N7),IERR
     1)
      IF( IERR .EQ. 0 ) GO TO 1200
      CALL LNCNT(3)
      PRINT 1150,TIME
 1150 FORMAT(//' IN CNTREG AT TIME ',D16.8,' P CANNOT BE COMPUTED DUE T
     1O MATRIX SINGULARITY IN GELIM')
      RETURN
C
 1200 CONTINUE
      CALL MAXEL(P,NP,ANORM1)
      CALL SUBT(DUMMY(N5),NA,P,NP,DUMMY(N4),NA)
      CALL MAXEL(DUMMY(N4),NA,ANORM2)
      IF( ANORM1 .NE. 0.0 ) GO TO 1225
      GO TO 1300
C
 1225 CONTINUE
      IF(ANORM1 .GT. 1.0 ) GO TO 1250
      IF( ANORM2/ANORM1 .LT. RICTCV ) KSS=1
      GO TO 1300
 1250 CONTINUE
      IF( ANORM2 .LT. RICTCV ) KSS=1
C
 1300 CONTINUE
      CALL MULT(DUMMY,NDUM1,P,NP,F,NF)
      IF( IOP(2) .EQ. 0 ) GO TO 1400
      CALL LNCNT(5)
      PRINT 1350,TIME
 1350 FORMAT(///' TIME = ',D16.8/)
      CALL PRNT(P,NP,4H P   ,1)
      IF( I .NE. NMAX )  CALL PRNT(F,NF,4H F   ,1)
C
 1400 CONTINUE
```

```
      IF( KSS .EQ. 1 ) GO TO 1500                                              CNT
      I = I-1                                                                  CNT
      IF( I .GE. 0 ) GO TO 1100                                                CNT
      GO TO 1600                                                               CNT
 1500 CONTINUE                                                                 CNT
      CALL LNCNT(4)                                                            CNT
      PRINT 1550                                                               CNT
 1550 FORMAT(//' STEADY-STATE SOLUTION HAS BEEN REACHED IN CNTREG'/)           CNT
C                                                                              CNT
 1600 CONTINUE                                                                 CNT
      IF( IOP(2) .NE. 0 ) RETURN                                               CNT
      IF( IOP(1) .EQ. 0 ) RETURN                                               CNT
      CALL LNCNT(5)                                                            CNT
      PRINT 1350,TIME                                                          CNT
      CALL PRNT(P,NP,4H P   ,1)                                                CNT
      CALL PRNT(F,NF,4H F   ,1)                                                CNT
C                                                                              CNT
      RETURN                                                                   CNT
      END                                                                      CNT
```

```
      SUBROUTINE RICNWT(A,NA,B,NB,H,NH,Q,NQ,R,NR,F,NF,P,NP,IOP,IDENT,DI
     1SC,FNULL,DUMMY)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(1),B(1),Q(1),R(1),F(1),P(1),DUMMY(1)
      DIMENSION NA(2),NB(2),NQ(2),NR(2),NF(2),NP(2),IOP(3)
      DIMENSION H(1),NH(2),IOPT(2)
      LOGICAL  IDENT,DISC,FNULL,SYM
      COMMON/TOL/EPSAM,EPSBM,IACM
      COMMON/CONV/SUMCV,RICTCV,SERCV,MAXSUM
      I=1
      IOPT(1)=0
      SYM = .TRUE.
C
      N = NA(1)**2
      N1 = N +1
      IF( .NOT. DISC) N1 = NA(1)*NR(1) + 1
      N2= N1+N
      N3= N2+N
      N4 = N3+N
C
      IF( IOP(1) .EQ. 0 )  GO TO 210
      CALL LNCNT(4)
      IF(.NOT. DISC)PRINT 100
      IF( DISC )PRINT 150
  100 FORMAT(//,' PROGRAM TO SOLVE CONTINUOUS STEADY-STATE RICCATI EQUAT
     1ION BY THE NEWTON ALGORITHM',/)
  150 FORMAT(//,' PROGRAM TO SOLVE DISCRETE STEADY-STATE RICCATI EQUATIO
     1N BY THE NEWTON ALGORITHM',/)
      CALL PRNT(A,NA,4H A  ,1)
      CALL PRNT(B,NB,4H B  ,1)
      CALL PRNT(Q,NQ,4H Q  ,1)
      IF( .NOT. IDENT )GO TO 185
      CALL LNCNT(3)
      PRINT 180
  180 FORMAT(/,' H IS AN IDENTITY MATRIX',/)
      GO TO 200
  185 CONTINUE
      CALL PRNT(H,NH,4H H  ,1)
      CALL MULT(Q,NQ,H,NH,DUMMY,NH)
      CALL TRANP(H,NH,DUMMY(N2),NP)
      CALL MULT(DUMMY(N2),NP,DUMMY,NH,Q,NQ)
      CALL LNCNT(3)
      PRINT 195
  195 FORMAT(/,' MATRIX (H TRANSPOSE)QH ',/)
      CALL PRNT(Q,NQ,4HHTQH,1)
  200 CONTINUE
      CALL PRNT(R,NR,4H R  ,1)
      IF( FNULL )  GO TO 210
      CALL LNCNT(3)
      PRINT 205
  205 FORMAT(/,' INITIAL F MATRIX',/)
      CALL PRNT(F,NF,4H F  ,1)
C
  210 CONTINUE
      IF((IOP(1) .NE. 0)  .OR. IDENT)  GO TO 220
      CALL MULT(Q,NQ,H,NH,DUMMY,NH)
      CALL TRANP(H,NH,DUMMY(N2),NP)
      CALL MULT(DUMMY(N2),NP,DUMMY,NH,Q,NQ)
  220 CONTINUE
C
      IF (DISC) GO TO 900
C
      CALL TRANP(B,NB,P,NP)
```

```
      CALL EQUATE(R,NR,DUMMY,NR)
      CALL SYMPOS(NR(1),NR(1),DUMMY,NP(2),P,IOPT,IOPT,DET,ISCALE,DUMMY(N
     11),IERR)
      IF(IERR .EQ. 0) GO TO 250
      CALL LNCNT(3)
      PRINT 225
  225 FORMAT(/,' IN RICNWT, A MATRIX WHICH IS  NOT SYMMETRIC POSITIVE DE
     1FINITE HAS BEEN SUBMITTED TO  SYMPOS',/)
      RETURN
C
  250 CONTINUE
      CALL EQUATE(P,NP,DUMMY,NF)
      CALL MULT(B,NB,DUMMY,NF,DUMMY(N1),NA)
      CALL TRANP(DUMMY(N1),NA,DUMMY(N2),NA)
      CALL ADD(DUMMY(N1),NA,DUMMY(N2),NA,DUMMY(N1),NA)
      CALL SCALE(DUMMY(N1),NA,DUMMY(N1),NA,0.5)
C
      IF(FNULL) GO TO 300
C
      CALL MULT(B,NB,F,NF,DUMMY(N2),NA)
      CALL SUBT(A,NA,DUMMY(N2),NA,DUMMY(N2),NA)
      CALL TRANP(DUMMY(N2),NA,DUMMY(N3),NA)
      CALL EQUATE(DUMMY(N3),NA,DUMMY(N2),NA)
      CALL MULT(R,NR,F,NF,DUMMY(N3),NF)
      CALL TRANP(F,NF,P,NP)
      CALL MULT(P,NP,DUMMY(N3),NF,DUMMY(N4),NA)
      CALL TRANP(DUMMY(N4),NA,DUMMY(N3),NA)
      CALL ADD(DUMMY(N4),NA,DUMMY(N3),NA,DUMMY(N3),NA)
      CALL SCALE(DUMMY(N3),NA,DUMMY(N3),NA,0.5)
      CALL ADD(DUMMY(N3),NA,Q,NQ,P,NP)
      CALL SCALE(P,NP,P,NP,-1.0)
      GO TO 350
C
  300 CONTINUE
      CALL TRANP(A,NA,DUMMY(N2),NA)
      CALL SCALE(Q,NQ,P,NP,-1.0)
C
  350 CONTINUE
      IF(IOP(3) .NE. 0) GO TO 400
      EPSA= EPSAM
      CALL BARSTW(DUMMY(N2),NA,B,NB,P,NP,IOPT,SYM,EPSA,EPSA,DUMMY(N3))
      GO TO 450
C
  400 CONTINUE
      IOPT(2)=1
      CALL BILIN(DUMMY(N2),NA,B,NB,P,NP,IOPT,SCLE,SYM,DUMMY(N3))
C
  450 CONTINUE
      CALL EQUATE(P,NP,DUMMY(N2),NP)
      IF(IOP(2).EQ. 0) GO TO 550
      CALL LNCNT(3)
      PRINT 500,I
  500 FORMAT(/,' ITERATION  ',I5,/)
      CALL PRNT(P,NP,4H P  ,1)
C
  550 CONTINUE
      CALL MULT(DUMMY(N1),NA,P,NP,DUMMY(N3),NA)
      CALL MULT(P,NP,DUMMY(N3),NA,DUMMY(N4),NA)
      CALL TRANP(DUMMY(N4),NA,P,NA)
      CALL ADD(P,NP,DUMMY(N4),NA,P,NP)
      CALL SCALE(P,NP,P,NP,0.5)
      CALL ADD(Q,NQ,P,NP,P,NP)
      CALL SCALE(P,NP,P,NP,-1.0)
```

```
      CALL SUBT(A,NA,DUMMY(N3),NA,DUMMY(N4),NA)                              RIC01
      CALL TRANP(DUMMY(N4),NA,DUMMY(N3),NA)                                  RIC01
C                                                                            RIC01
      IF(IOP(3) .NE. 0 ) GO TO 650                                           RIC01
      CALL BARSTW(DUMMY(N3),NA,B,NB,P,NP,IOPT,SYM,EPSA,EPSA,DUMMY(N4))       RIC01
      GO TO 675                                                              RIC01
C                                                                            RIC01
  650 CONTINUE                                                               RIC01
      CALL BILIN(DUMMY(N3),NA,B,NB,P,NP,IOPT,SCLE,SYM,DUMMY(N4))             RIC01
C                                                                            RIC01
  675 CONTINUE                                                               RIC01
      I=I+1                                                                  RIC01
      CALL MAXEL(DUMMY(N2),NA,ANORM1)                                        RIC01
      CALL SUBT(P,NP,DUMMY(N2),NA,DUMMY(N3),NA)                              RIC01
      CALL MAXEL(DUMMY(N3),NA,ANORM2)                                        RIC01
      IF(ANORM1 .GT. 1.0) GO TO 700                                          RIC01
      IF( ANORM2/ANORM1 .LT. RICTCV ) GO TO 800                              RIC01
      GO TO 750                                                              RIC01
C                                                                            RIC01
  700 CONTINUE                                                               RIC01
      IF( ANORM2 .LT. RICTCV ) GO TO 800                                     RIC01
C                                                                            RIC01
  750 CONTINUE                                                               RIC01
      IF( I .LE. 101) GO TO 450                                              RIC01
      CALL LNCNT(3)                                                          RIC01
      PRINT 775                                                              RIC01
  775 FORMAT(/,' THE SUBROUTINE RICNWT HAS EXCEEDED 100 ITERATIONS WITHORIC01
     1UT CONVERGENCE',/)                                                     RIC01
      IOP(1) = 1                                                             RIC01
C                                                                            RIC01
  800 CONTINUE                                                               RIC01
      CALL MULT(DUMMY,NF,P,NP,F,NF)                                          RIC01
      GO TO 1300                                                             RIC01
C                                                                            RIC01
900   CONTINUE
      IF( .NOT. FNULL ) GO TO 950                                            RIC01
C                                                                            RIC01
      CALL EQUATE(Q,NQ,P,NP)                                                 RIC01
      CALL EQUATE(A,NA,DUMMY(N1),NA)                                         RIC01
      CALL TRANP(A,NA,DUMMY(N2),NA)                                          RIC01
      GO TO 1000                                                             RIC01
  925 CONTINUE                                                               RIC01
C                                                                            RIC01
      I=I+1                                                                  RIC01
      CALL EQUATE(P,NP,DUMMY,NP)                                             RIC01
  950 CONTINUE                                                               RIC01
C                                                                            RIC01
      CALL MULT(R,NR,F,NF,DUMMY(N1),NF)                                      RIC01
      CALL TRANP(F,NF,P,NP)                                                  RIC01
      CALL MULT(P,NP,DUMMY(N1),NF,DUMMY(N2),NA)                              RIC01
      CALL TRANP(DUMMY(N2),NA,DUMMY(N1),NA)                                  RIC01
      CALL ADD(DUMMY(N1),NA,DUMMY(N2),NA,DUMMY(N1),NA)                       RIC01
      CALL SCALE(DUMMY(N1),NA,DUMMY(N   ,NA,0.5)                             RIC01
      CALL ADD(Q,NQ,DUMMY(N1),NA,P,NP)                                       RIC01
      CALL MULT(B,NB,F,NF,DUMMY(N1),NA)                                      RIC01
      CALL SUBT(A,NA,DUMMY(N1),NA,DUMMY(N1),NA)                              RIC01
      CALL TRANP(DUMMY(N1),NA,DUMMY(N2),NA)                                  RIC01
C                                                                            RIC01
 1000 CONTINUE                                                               RIC01
      CALL SUM(DUMMY(N2),NA,P,NP,DUMMY(N1),NA,IOPT,SYM,DUMMY(N3))            RIC01
      IF(IOP(2) .EQ. 0) GO TO 1100                                           RIC01
      CALL LNCNT(3)                                                          RIC01
      PRINT 500,I                                                            RIC01
```

```
      CALL PRNT(P,NP,4H P  ,1)
C
 1100 CONTINUE
      CALL MULT(P,NP,A,NA,DUMMY(N1),NA)
      CALL MULT(P,NP,B,NB,DUMMY(N2),NB)
      CALL TRANP(B,NB,DUMMY(N3),NF)
      CALL MULT(DUMMY(N3),NF,DUMMY(N1),NA,F,NF)
      CALL MULT(DUMMY(N3),NF,DUMMY(N2),NB,DUMMY(N1),NR)
      CALL TRANP(DUMMY(N1),NR,DUMMY(N2),NR)
      CALL ADD(DUMMY(N1),NR,DUMMY(N2),NR,DUMMY(N1),NR)
      CALL SCALE(DUMMY(N1),NR,DUMMY(N1),NR,0.5)
      CALL ADD(R,NR,DUMMY(N1),NR,DUMMY(N1),NR)
      CALL SYMPDS(NR(1),NR(1),DUMMY(N1),NA(1),F,IOPT,IOPT,DET,ISCALE,DUM
     1MY(N2),IERR)
      IF(IERR .EQ. 0) GO TO 1150
      CALL LNCNT(3)
      PRINT 225
      RETURN
C
 1150 CONTINUE
      IF( I .EQ. 1) GO TO 925
      CALL MAXEL(DUMMY,NA,ANORM1)
      CALL SUBT(P,NP,DUMMY,NA,DUMMY(N1),NA)
      CALL MAXEL(DUMMY(N1),NA,ANORM2)
      IF( ANORM1 .GT. 1.) GO TO 1200
      IF( ANORM2/ANORM1 .LT. RICTCV ) GO TO 1300
      GO TO 1250
 1200 CONTINUE
      IF( ANORM2 .LT. RICTCV ) GO TO 1300
C
 1250 CONTINUE
      IF( I .LE. 101) GO TO  925
      CALL LNCNT(3)
      PRINT 775
      IOP(1) = 1
C
 1300 CONTINUE
      IF(IOP(1) .EQ. 0 ) RETURN
      CALL LNCNT(4)
      PRINT 1350,I
 1350 FORMAT(//,' FINAL VALUES OF P AND F AFTER',I5,' ITERATIONS TO CONV
     1ERGE',/)
      CALL PRNT(P,NP,4H P  ,1)
      CALL PRNT(F,NF,4H F  ,1)
C
      RETURN
      END
```

```
      SUBROUTINE ASMREG(A,NA,B,NB,H,NH,Q,NQ,R,NR,F,NF,P,NP,IDENT,DISC,N
     1EWT,STABLE,FNULL,ALPHA,IOP,DUMMY)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(1),B(1),H(1),Q(1),R(1),F(1),P(1),DUMMY(1)
      DIMENSION NA(2),NB(2),NH(2),NQ(2),NR(2),NF(2),NP(2),IOP(5),IOPT(3)
     1,NDUM1(2),NDUM2(2),NDUM3(2)
      LOGICAL IDENT,DISC,NEWT,STABLE,FNULL,SING
      N = NA(1)**2
      N1= N+1
      IOPTT=0
      IF ( .NOT. NEWT ) GO TO 600
      IF( STABLE )  GO TO 500
      IF ( FNULL ) GO TO 100
      CALL MULT(B,NB,F,NF,DUMMY,NA)
      CALL SUBT(A,NA,DUMMY,NA,DUMMY,NA)
      CALL TESTSA(DUMMY,NA,ALPHA,DISC,STABLE,IOPTT,DUMMY(N1))
      GO TO 200
  100 CONTINUE
      CALL TESTSA(A,NA,ALPHA,DISC,STABLE,IOPTT,DUMMY)
C
  200 CONTINUE
      IF( STABLE ) GO TO 500
      IF( DISC ) GO TO 230
      J = -NA(1)
      NAX = NA(1)
      DO 210 I =1,NAX
      J = J + NAX +1
      A(J) = A(J)-ALPHA
  210 CONTINUE
      SCLE = 3.
      IOPT(1)=IOP(1)
      IOPT(2) = 1
      IOPT(3)=1
      CALL CSTAB(A,NA,B,NB,F,NF,IOPT,SCLE,DUMMY)
      J = -NA(1)
      DO 220 I=1,NAX
      J = J + NAX + 1
      A(J) = A(J) + ALPHA
  220 CONTINUE
  225 CONTINUE
      CALL MULT(B,NB,F,NF,DUMMY,NA)
      CALL SUBT(A,NA,DUMMY,NA,DUMMY,NA)
      CALL TESTSA(DUMMY,NA,ALPHA,DISC,STABLE,IOPTT,DUMMY(N1))
      GO TO 300
C
  230 CONTINUE
      J = 2*NA(1) + 1
      IF( .NOT. FNULL )  J = J + N
      SING = .FALSE.
      IF( DUMMY(J) .EQ. 0.0 )  SING = .TRUE.
      IOPT(1) = IOP(1)
      IOPT(2) = 1
      DSCLE = 0.5
      ALPHAT = 1./ALPHA
      CALL SCALE(A,NA,A,NA,ALPHAT)
      CALL SCALE(B,NB,B,NB,ALPHAT)
      CALL DSTAB(A,NA,B,NB,F,NF,SING,IOPT,DSCLE,DUMMY)
      CALL SCALE(A,NA,A,NA,ALPHA)
      CALL SCALE(B,NB,B,NB,ALPHA)
      GO TO 225
C
  300 CONTINUE
      IF( STABLE) GO TO 400
```

```
      CALL LNCNT(5)
      IF( DISC ) GO TO 330
      PRINT 310,ALPHA
  310 FORMAT(//' IN ASMREG, CSTAB HAS FAILED TO FIND A STABILIZING GAIN
     1 MATRIX (F) RELATIVE TO ',/,' ALPHA = ',D16.8/)
      RETURN
  330 CONTINUE
      PRINT 340,ALPHA
  340 FORMAT(//' IN ASMREG, DSTAB HAS FAILED TO FIND A STABILIZING GAIN
     1 MATRIX (F) RELATIVE TO ',/,' ALPHA = ',D16.8/)
      RETURN
C
  400 CONTINUE
      FNULL = .FALSE.
C
  500 CONTINUE
      CALL RICNWT(A,NA,B,NB,H,NH,Q,NQ,R,NR,F,NF,P,NP,IOP,IDENT,DISC,FNU
     1LL,DUMMY)
      GO TO 750
C
  600 CONTINUE
      IF( DISC ) GO TO 700
      NW = 4*N + 1
      NLAM = NW + 4*N
      NDUM = NLAM + N
      IOP(3) = 1
      CALL CNTREG(A,NA,B,NB,H,NH,Q,NQ,R,NR,DUMMY,DUMMY(NW),DUMMY(NLAM),
     1S,F,NF,P,NP,T,IOP,IDENT,DUMMY(NDUM))
      GO TO 750
  700 CONTINUE
      CALL DISREG(A,NA,B,NB,H,NH,Q,NQ,R,NR,F,NF,P,NP,IOP,IDENT,DUMMY)
C
  750 CONTINUE
C
      IF( IOP(4) .EQ. 0 ) GO TO 1100
C
      N2= N1 + N
      N3= N2 + N
C
      IF( DISC ) GO TO 800
      CALL MULT(P,NP,B,NB,DUMMY,NB)
      CALL MULT(DUMMY,NB,F,NF,DUMMY(N1),NP)
      CALL TRANP(DUMMY(N1),NP,DUMMY,NP)
      CALL ADD(DUMMY,NP,DUMMY(N1),NP,DUMMY,NP)
      CALL SCALE(DUMMY,NP,DUMMY,NP,0.5)
      CALL SUBT(Q,NQ,DUMMY,NP,DUMMY,NP)
      CALL MULT(P,NP,A,NA,DUMMY(N1),NP)
      CALL ADD(DUMMY,NP,DUMMY(N1),NP,DUMMY,NP)
      CALL TRANP(DUMMY(N1),NP,DUMMY(N2),NP)
      CALL ADD(DUMMY,NP,DUMMY(N2),NP,DUMMY,NP)
      GO TO 900
C
  800 CONTINUE
      CALL MULT(R,NR,F,NF,DUMMY,NF)
      CALL TRANP(F,NF,DUMMY(N1),NB)
      CALL MULT(DUMMY(N1),NB,DUMMY,NF,DUMMY(N2),NA)
      CALL ADD(DUMMY(N2),NA,Q,NQ,DUMMY,NA)
      CALL MULT(B,NB,F,NF,DUMMY(N1),NA)
      CALL SUBT(A,NA,DUMMY(N1),NA,DUMMY(N1),NA)
      CALL MULT(P,NP,DUMMY(N1),NA,DUMMY(N2),NA)
      CALL TRANP(DUMMY(N1),NA,DUMMY(N3),NA)
      CALL MULT(DUMMY(N3),NA,DUMMY(N2),NA,DUMMY(N1),NA)
      CALL ADD(DUMMY,NA,DUMMY(N1),NA,DUMMY,NA)
```

```
      CALL SUBT(P,NP,DUMMY,NA,DUMMY,NA)                                 ASM01
C                                                                       ASM01
  900 CONTINUE                                                          ASM01
      CALL LNCNT(4)                                                     ASM01
      PRINT 1000                                                        ASM01
 1000 FORMAT(//' RESIDUAL ERROR IN RICCATI EQUATION '/)                 ASM01
      CALL PRNT(DUMMY,NP,4HEROR,1)                                      ASM01
C                                                                       ASM01
 1100 CONTINUE                                                          ASM01
      N2= N1+NA(1)                                                      ASM01
      N3= N2+NA(1)                                                      ASM01
      ISV = 0                                                           ASM01
      CALL EQUATE(P,NP,DUMMY,NP)                                        ASM01
      CALL EIGEN(NA(1),NA(1),DUMMY,DUMMY(N1),DUMMY(N2),ISV,ISV,V,DUMMY(NASM01
     13),IERR)                                                          ASM01
      NEVL = NA(1)                                                      ASM01
      IF( IERR .EQ. 0) GO TO 1300                                       ASM01
      NEVL=NA(1)-IERR                                                   ASM01
      CALL LNCNT(4)                                                     ASM01
      PRINT 1200,IERR                                                   ASM01
 1200 FORMAT(//' IN ASMREG, THE ',I5, ' EIGENVALUE OF P  HAS NOT BEEN COASM01
     1PUTED AFTER 30 ITERATIONS '/)                                     ASM01
C                                                                       ASM01
 1300 CONTINUE                                                          ASM01
      NDUM1(1) = NEVL                                                   ASM01
      NDUM1(2) = 1                                                      ASM01
      CALL EQUATE(DUMMY(N1),NDUM1,DUMMY,NDUM1)                          ASM01
      N1 = NDUM1(1) +1                                                  ASM01
      CALL MULT(B,NB,F,NF,DUMMY(N1),NA)                                 ASM01
      CALL SUBT(A,NA,DUMMY(N1),NA,DUMMY(N1),NA)                         ASM01
      N2 = N1+N                                                         ASM01
      CALL EQUATE(DUMMY(N1),NA,DUMMY(N2),NA)                            ASM01
      N3=N2+N                                                           ASM01
      N4=N3+NA(1)                                                       ASM01
      N5=N4+NA(1)                                                       ASM01
      CALL EIGEN(NA(1),NA(1),DUMMY(N2),DUMMY(N3),DUMMY(N4),ISV,ISV,V,DUMASM01
     1MY(N5),IERR)                                                      ASM01
      NEVL = NA(1)                                                      ASM01
      IF( IERR .EQ. 0 ) GO TO 1500                                      ASM01
      NEVL=NA(1)-IERR                                                   ASM01
      CALL LNCNT(4)                                                     ASM01
      PRINT 1400,IERR                                                   ASM01
 1400 FORMAT(//' IN ASMREG, THE ',I5,' EIGENVALUE OF A-BF HAS NOT BEEN CASM01
     1OMPUTED AFTER 30 ITERATIONS'/)                                    ASM01
C                                                                       ASM01
 1500 CONTINUE                                                          ASM01
      NDUM2(1) = NEVL                                                   ASM01
      NDUM2(2) = 1                                                      ASM01
      CALL JUXTC(DUMMY(N3),NDUM2,DUMMY(N4),NDUM2,DUMMY(N2),NDUM3)       ASM01
C                                                                       ASM01
      IF ( IOP(5) .EQ. 0 ) RETURN                                       ASM01
C                                                                       ASM01
      CALL LNCNT(4)                                                     ASM01
      PRINT 1600                                                        ASM01
 1600 FORMAT(//' EIGENVALUES OF P '/)                                   ASM01
      CALL PRNT(DUMMY,NDUM1,4HEVLP,1)                                   ASM01
      CALL LNCNT(4)                                                     ASM01
      PRINT 1700                                                        ASM01
 1700 FORMAT(//' CLOSED-LOOP RESPONSE MATRIX A-BF '/)                   ASM01
      CALL PRNT(DUMMY(N1),NA,4HA-BF,1)                                  ASM01
      CALL LNCNT(3)                                                     ASM01
      PRINT 1800                                                        ASM01
 1800 FORMAT(//' EIGENVALUES OF A-BF')                                  ASM01
```

```
      CALL PRNT(DUMMY(N2),NDUM3,0,3)                                   ASMO
C                                                                      ASMO
      RETURN                                                           ASMO
      END                                                              ASMO
```

```
      SUBROUTINE ASMFIL(A,NA,G,NG,H,NH,Q,NQ,R,NR,F,NF,P,NP,IDENT,DISC,N
     1EWT,STABLE,FNULL,ALPHA,IOP,DUMMY)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(1),G(1),H(1),Q(1),R(1),F(1),P(1),DUMMY(1)
      DIMENSION NA(2),NG(2),NH(2),NQ(2),NR(2),NF(2),NP(2),IOPT(5),NDUM1(
     12),IOP(1)
      LOGICAL  IDENT,DISC,NEWT,STABLE,FNULL
      IF( IOP(1) .EQ. 0 ) GO TO 100
      CALL LNCNT(4)
      IF(DISC)  PRINT 15
      IF( .NOT. DISC )  PRINT 25
   15 FORMAT(//,' PROGRAM TO SOLVE THE DISCRETE INFINITE-DURATION OPTIMA
     1L FILTER PROBLEM',/)
   25 FORMAT(//,' PROGRAM TO SOLVE THE CONTINUOUS INFINITE-DURATION OPTI
     1MAL FILTER PROBLEM',/)
      CALL PRNT(A,NA,4H A  ,1)
      IF( .NOT.  IDENT )  GO TO 35
      CALL LNCNT(3)
      PRINT 30
   30 FORMAT(/,' G IS AN IDENTITY MATRIX',/)
      GO TO 40
   35 CONTINUE
      CALL PRNT(G,NG,4H G  ,1)
   40 CONTINUE
      CALL PRNT(H,NH,4H H  ,1)
      CALL LNCNT(3)
      PRINT 45
   45 FORMAT(/,'INTENSITY MATRIX FOR COVARIANCE OF MEASUREMENT NOISE',/)
      CALL PRNT(R,NR,4H R  ,1)
C
      IF( .NOT. IDENT ) GO TO 65
      CALL LNCNT(3)
      PRINT 55
   55 FORMAT(/,' INTENSITY MATRIX FOR COVARIANCE OF PROCESS NOISE',/)
C
   65 CONTINUE
      CALL PRNT(Q,NQ,4H Q  ,1)
C
  100 CONTINUE
      IOPT(1)=IOP(2)
      IOPT(2)=IOP(3)
      IOPT(3)=IOP(4)
      IOPT(4)=IOP(5)
      IOPT(5)=0
      K = 0
C
  200 CONTINUE
      CALL TRANP(A,NA,DUMMY,NA)
      CALL EQUATE(DUMMY,NA,A,NA)
      CALL TRANP(H,NH,DUMMY,NDUM1)
      CALL EQUATE(DUMMY,NDUM1,H,NH)
      IF( IDENT )  GO TO 250
      CALL TRANP(G,NG,DUMMY,NDUM1)
      CALL EQUATE(DUMMY,NDUM1,G,NG)
  250 CONTINUE
      IF ( K .EQ. 1 ) RETURN
C
      K = K+1
      CALL ASMREG(A,NA,H,NH,G,NG,Q,NQ,R,NR,F,NF,P,NP,IDENT,DISC,NEWT,ST
     1ABLE,FNULL,ALPHA,IOPT,DUMMY)
C
      N1=(NA(1)**2)+3*NA(1)+1
      CALL TRANP(F,NF,DUMMY(N1),NDUM1)
```

```
      CALL EQUATE(DUMMY(N1),NDUM1,F,NF)
C
      IF( IOP(1) .EQ. 0 ) GO TO 200
C
      IF(IDENT) GO TO 300
      CALL LNCNT(3)
      PRINT 55
      CALL PRNT(Q,NQ,4HGQGT,1)
C
  300 CONTINUE
      CALL LNCNT(3)
      PRINT 325
  325 FORMAT(/,' FILTER GAIN',/)
      CALL PRNT(F,NF,4H F  ,1)
      CALL LNCNT(3)
      PRINT 350
  350 FORMAT(/,'STEADY-STATE VARIANCE MATRIX OF RECONSTRUCTION ERROR',/)
      CALL PRNT(P,NP,4H P  ,1)
      NDUM1(1)=NP(1)
      NDUM1(2)=1
      CALL LNCNT(3)
      PRINT 375
  375 FORMAT(/,' EIGENVALUES OF P ',/)
      CALL PRNT(DUMMY,NDUM1,4HEVLP,1)
      N1 = NP(1) + 1
      N = NA(1)**2
      N2 = N1 + N + 2*NA(1)
      CALL TRANP(DUMMY(N1),NA,DUMMY(N2),NA)
      CALL PRNT(DUMMY(N2),NA,4HA-FH,1)
      N2 = N1 + N
      CALL LNCNT(3)
      PRINT 385
  385 FORMAT(/,' EIGENVALUES OF A-FH MATRIX',/)
      NDUM1(1) = NA(1)
      NDUM1(2) =  2
      CALL PRNT(DUMMY(N2),NDUM1,0,3)
C
      GO TO 200
C
      END
```

```
      SUBROUTINE EXPMDF (A,NA,B,NB,H,NH,AM,NAM,HM,NHM,Q,NQ,R,NR,F,NF,P, EXPO
     1NP,HIDENT,HMDENT,DISC,NEWT,STABLE,FNULL,ALPHA,IOP,DUMMY)          EXPO
      IMPLICIT REAL*8 (A-H,O-Z)                                         EXPO
      DIMENSION A(1),B(1),H(1),AM(1),HM(1),Q(1),R(1),F(1),P(1),DUMMY(1) EXPO
      DIMENSION NA(2),NB(2),NH(2),NAM(2),NHM(2),NQ(2),NR(2),NF(2),NP(2),EXPO
     1IOP(1),IOPT(5),NDUM1(2),NDUM2(2),NDUM3(2)                         EXPO
      LOGICAL  HIDENT,HMDENT,DISC,NEWT,STABLE,FNULL,SYM                 EXPO
      COMMON/TOL/EPSAM,EPSBM,IACM                                       EXPO
      IF( IOP(1) .EQ. 0 ) GO TO 300                                     EXPO
      CALL LNCNT(6)                                                     EXPO
      IF( DISC ) PRINT 25                                               EXPO
      IF( .NOT. DISC ) PRINT 50                                         EXPO
   25 FORMAT(/,' PROGRAM TO SOLVE ASYMPTOTIC DISCRETE EXPLICIT MODEL-FOLEXPO
     1LOWING PROBLEM',//,' PLANT DYNAMICS',/)                           EXPO
   50 FORMAT(/,' PROGRAM TO SOLVE ASYMPTOTIC CONTINUOUS EXPLICIT MODEL-FEXPO
     1OLLOWING PROBLEM',//,' PLANT DYNAMICS',/)                         EXPO
      CALL PRNT(A,NA,4H A   ,1)                                         EXPO
      CALL PRNT(B,NB,4H B   ,1)                                         EXPO
      IF( HIDENT ) GO TO 75                                             EXPO
      CALL PRNT(H,NH,4H H   ,1)                                         EXPO
      GO TO 100                                                         EXPO
   75 CONTINUE                                                          EXPO
      CALL LNCNT(3)                                                     EXPO
      PRINT 85                                                          EXPO
   85 FORMAT(/,' H IS AN IDENTITY MATRIX',/)                            EXPO
C                                                                       EXPO
  100 CONTINUE                                                          EXPO
      CALL LNCNT(4)                                                     EXPO
      PRINT 125                                                         EXPO
  125 FORMAT(//,' MODEL DYNAMICS',/)                                    EXPO
      CALL PRNT(AM,NAM,4H AM ,1)                                        EXPO
      IF( HMDENT ) GO TO 175                                            EXPO
      CALL PRNT(HM,NHM,4H HM ,1)                                        EXPO
      GO TO 200                                                         EXPO
  175 CONTINUE                                                          EXPO
      CALL LNCNT(3)                                                     EXPO
      PRINT 185                                                         EXPO
  185 FORMAT(/,' HM IS AN IDENTITY MATRIX ',/)                          EXPO
C                                                                       EXPO
  200 CONTINUE                                                          EXPO
      CALL LNCNT(4)                                                     EXPO
      PRINT 225                                                         EXPO
  225 FORMAT(//,' WEIGHTING MATRICES ',/)                               EXPO
      CALL PRNT(Q,NQ,4H Q   ,1)                                         EXPO
      CALL PRNT(R,NR,4H R   ,1)                                         EXPO
C                                                                       EXPO
  300 CONTINUE                                                          EXPO
      IF( IOP(2) .EQ. 0 ) GO TO 400                                     EXPO
      NF(1) = NB(2)                                                     EXPO
      NF(2) = NA(1)                                                     EXPO
      NP(1) = NA(1)                                                     EXPO
      NP(2) = NA(1)                                                     EXPO
      IOPT(1) = IOP(3)                                                  EXPO
      IOPT(2) = IOP(4)                                                  EXPO
      IOPT(3) = IOP(5)                                                  EXPO
      IOPT(4) =  0                                                      EXPO
      IOPT(5) =  0                                                      EXPO
      N1 = NA(1)*NA(2) + 1                                              EXPO
      CALL EQUATE(Q,NQ,DUMMY,NQ)                                        EXPO
      CALL ASMREG(A,NA,B,NB,H,NH,DUMMY,NQ,R,NR,F,NF,P,NP,HIDENT,DISC,NE EXPO
     1WT,STABLE,FNULL,ALPHA,IOPT,DUMMY(N1))                             EXPO
C                                                                       EXPO
  400 CONTINUE                                                          EXPO
```

```
      IF( IOP(1) .EQ. 0 ) GO TO 600
      CALL LNCNT(4)
      PRINT 425
  425 FORMAT(//,' CONTROL LAW U = -F( COL.(X,XM) ),  F = (F11,F12)',/)
      CALL LNCNT(3)
      PRINT 450
  450 FORMAT(/,' PART OF F MULTIPLYING  X ',/)
      CALL PRNT(F,NF,4H F11,1)
      IF( .NOT. DISC  .AND. IOP(2) .EQ. 0 ) GO  TO 600
      CALL PRNT(P,NP,4H P11,1)
      IF(  IOP(2) .EQ. 0 ) GO TO 600
      CALL LNCNT(2)
      PRINT 475
  475 FORMAT(/,' EIGENVALUES OF P11')
      NDUM1(1) = NA(1)
      NDUM1(2) = 1
      CALL PRNT(DUMMY(N1),NDUM1,0,3)
      N1 = N1 + NDUM1(1)
      NDUM1(2) = NA(1)
      CALL LNCNT(2)
      PRINT 500
  500 FORMAT(/,' PLANT CLOSED-LOOP RESPONSE MATRIX A - BF11')
      CALL PRNT(DUMMY(N1),NDUM1,0,3)
      CALL LNCNT(2)
      PRINT 525
  525 FORMAT(/,' EIGENVALUES OF CLOSED-LOOP RESPONSE MATRIX')
      N1 = N1 + NDUM1(1)*NDUM1(2)
      NDUM1(2) = 2
      CALL PRNT(DUMMY(N1),NDUM1,0,3)
C
  600 CONTINUE
      NF(1)= NB(2)
      NF(2)= NA(1)
      CALL MULT(B,NB,F,NF,DUMMY,NA)
      CALL SUBT(A,NA,DUMMY,NA,DUMMY,NA)
      IF(  IOP(1).EQ. 0  .OR.  IOP(2) .NE. 0 ) GO TO 700
      CALL LNCNT(2)
      PRINT 500
      CALL PRNT(DUMMY,NA,0,3)
C
  700 CONTINUE
      N1 =  NA(1)**2 +1
      CALL TRANP(DUMMY,NA,DUMMY(N1),NA)
      CALL EQUATE(DUMMY(N1),NA,DUMMY,NA)
      NF(2) = NA(1) + NAM(1)
      NP(2) = NF(2)
      IF( .NOT. DISC .AND. IOP(2).EQ. 0 ) NP(2) = NAM(2)
      IOPTT=0
      SYM = .FALSE.
      CALL EQUATE( Q,NQ,DUMMY(N1),NDUM2)
      IF( HMDENT ) GO TO 725
      CALL MULT(Q,NQ,HM,NHM,DUMMY(N1),NDUM2)
  725 CONTINUE
      IF( HIDENT ) GO TO 750
      N2 = N1 + NQ(1)*NHM(2)
      CALL TRANP(H,NH,DUMMY(N2),NDUM1)
      N3 = N2 + NH(1)*NH(2)
      CALL MULT(DUMMY(N2),NDUM1,DUMMY(N1),NHM,DUMMY(N3),NDUM2)
      CALL EQUATE(DUMMY(N3),NDUM2,DUMMY(N1),NDUM2)
  750 CONTINUE
      N2 = NA(1)**2 + NA(1)*NHM(2) + 1
      N3 = NA(1)**2 + 1
      IF( .NOT. DISC .AND. IOP(2) .EQ. 0 ) N3 = 1
```

```
      CALL EQUATE(DUMMY(N1),NDUM2,P(N3),NDUM2)
      IF( DISC ) GO TO 800
      EPSA = EPSAM
      CALL BARSTW(DUMMY,NA,AM,NAM,P(N3),NDUM2,IOPTT,SYM ,EPSA,EPSA,DUMMY
     1(N2))
      GO TO 900
C
  800 CONTINUE
      CALL SCALE(P(N3),NDUM2,P(N3),NDUM2,-1.0)
      N4 = N2 +NAM(1)**2
      CALL EQUATE(AM,NAM,DUMMY(N2),NAM)
      CALL SUM(DUMMY,NA,P(N3),NDUM2,DUMMY(N2),NAM,IOPTT,SYM,DUMMY(N4))
C
  900 CONTINUE
      N2 = NB(2)*NA(1) + 1
      CALL TRANP(B,NB,DUMMY,NDUM1)
      CALL MULT(DUMMY,NDUM1,P(N3),NDUM2,F(N2),NDUM3)
      IF( .NOT. DISC ) GO TO 1000
      N1 = NB(1)*NB(2) + 1
      CALL MULT(DUMMY,NDUM1,P,NA,DUMMY(N1),NDUM2)
      CALL MULT(DUMMY(N1),NDUM2,B,NB,DUMMY,NR)
      CALL ADD(R,NR,DUMMY,NR,DUMMY,NR)
      GO TO 1100
C
 1000 CONTINUE
      CALL EQUATE(R,NR,DUMMY,NR)
C
 1100 CONTINUE
      N1 = NR(1)**2 + 1
      CALL SYMPDS(NR(1),NR(1),DUMMY,NHM(2),F(N2),IOPTT,IOPTT,DETERM,ISCA
     1LE,DUMMY(N1),IERR)
      IF( IERR .EQ. 0 ) GO TO 1200
      CALL LNCNT(3)
      PRINT 1150
 1150 FORMAT(/,' IN EXPMDF, THE COEFFICIENT MATRIX FOR SYMPDS IS NOT SY
     1MMETRIC POSITIVE DEFINITE ',/)
      RETURN
C
 1200 CONTINUE
      IF( .NOT. DISC ) GO TO 1300
      CALL MULT(F(N2),NDUM3,AM,NAM,DUMMY,NDUM1)
      CALL EQUATE(DUMMY,NDUM1,F(N2),NDUM1)
 1300 CONTINUE
      IF( IOP(1) .EQ. 0 ) RETURN
      CALL LNCNT(3)
      PRINT 1325
 1325 FORMAT(/,' PART OF F MULTIPLYING XM ',/)
      CALL PRNT(F(N2),NDUM3,4H F12,1)
      NDUM1(1) = NA(1)
      NDUM1(2) = NAM(1)
      CALL PRNT(P(N3),NDUM1,4H P12,1)
      RETURN
      END
```

```
      SUBROUTINE IMPMDF(A,NA,B,NB,H,NH,AM,NAM,BM,NBM,Q,NQ,R,NR,F,NF,P,N
     1P,IDENT,DISC,NEWT,STABLE,FNULL,ALPHA,IOP,DUMMY)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(1),B(1),H(1),AM(1),BM(1),Q(1),R(1),F(1),P(1),DUMMY(1)
      DIMENSION NA(2),NB(2),NH(2),NAM(2),NBM(2),NQ(2),NR(2),NF(2),NP(2),
     1IOP(1),IOPT(5),NDUM1(2)
      LOGICAL IDENT,DISC,NEWT,STABLE,FNULL,HIDENT
      IF( IOP(1) .EQ. 0 ) GO TO 200
      CALL LNCNT(6)
      IF( DISC ) PRINT 25
      IF( .NOT. DISC ) PRINT 50
   25 FORMAT(/,'PROGRAM TO SOLVE ASYMPTOTIC DISCRETE IMPLICIT MODEL-FOLL
     1OWING PROBLEM',//,' PLANT DYNAMICS ',/)
   50 FORMAT(/,' PROGRAM TO SOLVE ASYMPTOTIC CONTINUOUS IMPLICIT MODEL-F
     1OLLOWING PROBLEM',//,' PLANT DYNAMICS',/)
      CALL PRNT(A,NA,4H A   ,1)
      CALL PRNT(B,NB,4H B   ,1)
      IF( IDENT ) GO TO 75
      CALL PRNT(H,NH,4H H   ,1)
      GO TO 100
   75 CONTINUE
      CALL LNCNT(3)
      PRINT 85
   85 FORMAT(/,' H IS AN IDENTITY MATRIX',/)
C
  100 CONTINUE
      CALL LNCNT(4)
      PRINT 125
  125 FORMAT(//,' MODEL DYNAMICS',/)
      CALL PRNT(AM,NAM,4H AM ,1)
      CALL PRNT(BM,NBM,4H BM ,1)
      CALL LNCNT(4)
      PRINT 150
  150 FORMAT(//,' WEIGHTING MATRICES',/)
      CALL PRNT(Q,NQ,4H Q   ,1)
      CALL PRNT(R,NR,4H R   ,1)
C
  200 CONTINUE
      N = NA(1)**2
      N1 = N + 1
      IF( .NOT. IDENT ) GO TO 300
      CALL SUBT(A,NA,AM,NAM,DUMMY,NA)
      CALL SUBT(B,NB,BM,NBM,DUMMY(N1),NB)
      GO TO 400
C
  300 CONTINUE
      CALL MULT(H,NH,A,NA,DUMMY,NH)
      CALL MULT(AM,NAM,H,NH,DUMMY(N1),NH)
      CALL SUBT(DUMMY,NH,DUMMY(N1),NH,DUMMY,NH)
      CALL MULT(H,NH,B,NB,DUMMY(N1),NBM)
      CALL SUBT(DUMMY(N1),NBM,BM,NBM,DUMMY(N1),NBM)
C
  400 CONTINUE
      IF( IOP(1) .EQ. 0 ) GO TO 500
      CALL LNCNT(3)
      PRINT 450
  450 FORMAT(//,' MATRIX HA - AMH')
      CALL PRNT(DUMMY,NH,0,3)
      CALL LNCNT(3)
      PRINT 475
  475 FORMAT(//,' MATRIX HB - BM')
      CALL PRNT(DUMMY(N1),NBM,0,3)
C
```

```
  500 CONTINUE
      N2 = N1 + N
      N3 = N2 + N
      N4 = N3 + N
      CALL MULT(Q,NQ,DUMMY,NH,DUMMY(N2),NH)
      CALL MULT(Q,NQ,DUMMY(N1),NBM,DUMMY(N3),NBM)
      CALL TRANP(DUMMY,NH,DUMMY(N4),NDUM1)
      CALL MULT(DUMMY(N4),NDUM1,DUMMY(N2),NH,DUMMY,NA)
      CALL MULT(DUMMY(N4),NDUM1,DUMMY(N3),NBM,DUMMY(N2),NB)
      CALL TRANP(DUMMY(N1),NBM,DUMMY(N4),NDUM1)
      CALL SCALE(DUMMY(N2),NB,DUMMY(N1),NB,2.0)
      CALL MULT(DUMMY(N4),NDUM1,DUMMY(N3),NBM,DUMMY(N2),NR)
      CALL ADD(DUMMY(N2),NR,R,NR,DUMMY(N2),NR)
      IF( IOP(1) .EQ. 0 ) GO TO 600
      CALL LNCNT(3)
      PRINT 525
  525 FORMAT(//,' MATRIX (HA - AMH TRANSPOSE)Q( HA - AMH)')
      CALL PRNT(DUMMY,NA,0,3)
      CALL LNCNT(3)
      PRINT 550
  550 FORMAT(//,' MATRIX 2( HA - AMH  TRANSPOSE)Q( HB - BM)')
      CALL PRNT(DUMMY(N1),NB,0,3)
      CALL LNCNT(3)
      PRINT 575
  575 FORMAT(//,' MATRIX ( HB - BM TRANSPOSE)Q( HB - BM ) + R')
      CALL PRNT(DUMMY(N2),NR,0,3)
C
  600 CONTINUE
      IOPT(1)= 0
      IOPT(2)= 1
      IOPT(3)= 1
      N5 = N4 + N
      CALL EQUATE(A,NA,DUMMY(N3),NA)
      CALL PREFIL(DUMMY(N3),NA,B,NB,DUMMY,NA,DUMMY(N1),NB,DUMMY(N2),NR,D
     1UMMY(N4),NF,IOPT,DUMMY(N5))
      IF(IOP(1) .EQ. 0 ) GO TO 700
      CALL LNCNT(3)
      PRINT 625
  625 FORMAT(//,' PREFILTER GAIN')
      CALL PRNT(DUMMY(N4),NF,0,3)
      CALL LNCNT(3)
      PRINT 650
  650 FORMAT(//,' MATRIX A - B(PREFILTER)')
      CALL PRNT(DUMMY(N3),NA,0,3)
      CALL LNCNT(3)
      PRINT 675
  675 FORMAT(//,' MODIFIED STATE VECTOR WEIGHTING MATRIX')
      CALL PRNT(DUMMY,NA,0,3)
C
  700 CONTINUE
      CALL EQUATE(DUMMY(N4),NF,DUMMY(N1),NF)
C
      IF( IOP(2) .EQ. -1000 ) RETURN
C
      IOPT(1) = IOP(2)
      IOPT(2) = IOP(3)
      IOPT(3) = IOP(4)
      IOPT(4) = 0
      IOPT(5) = 0
      HIDENT = .TRUE.
      CALL ASMREG(DUMMY(N3),NA,B,NB,H,NH,DUMMY,NA,DUMMY(N2),NR,F,NF,P,N
     1P,HIDENT,DISC,NEWT,STABLE,FNULL,ALPHA,IOPT,DUMMY(N4))
      IF( IOP(1) .EQ. 0 ) GO TO 800
```

```
      CALL LNCNT(3)                                                    IMPO
      PRINT 725                                                        IMPO
  725 FORMAT(//,' GAIN FROM ASMREG')                                   IMPO
      CALL PRNT(F,NF,0,3)                                              IMPO
      CALL LNCNT(3)                                                    IMPO
      PRINT 750                                                        IMPO
  750 FORMAT(//,' SOLUTION OF ASSOCIATED STEADY-STATE RICCATI EQUATION')IMPO
      CALL PRNT(P,NP,0,3)                                              IMPO
      CALL LNCNT(3)                                                    IMPO
      PRINT 775                                                        IMPO
  775 FORMAT(//,' EIGENVALUES OF P')                                   IMPO
      NDUM1(1)= NA(1)                                                  IMPO
      NDUM1(2)= 1                                                      IMPO
      CALL PRNT(DUMMY(N4),NDUM1,0,3)                                   IMPO
C                                                                      IMPO
  800 CONTINUE                                                         IMPO
      CALL ADD(F,NF,DUMMY(N1),NF,F,NF)                                 IMPO
      IF( IOP(1) .EQ. 0 ) RETURN                                       IMPO
      CALL LNCNT(4)                                                    IMPO
      PRINT 825                                                        IMPO
  825 FORMAT(//,' GAIN FOR MODEL-FOLLOWING CONTROL LAW, U = - F X   , F =IMPO
     1(PREFILTER) + (ASMREG)',/)                                       IMPO
      CALL PRNT(F,NF,4H F   ,1)                                        IMPO
      N6 = N4 + NA(1)                                                  IMPO
      CALL PRNT(DUMMY(N6),NA,4HA-BF,1)                                 IMPO
      NDUM1(2) = 2                                                     IMPO
      N6 = N6 + N                                                      IMPO
      CALL LNCNT(3)                                                    IMPO
      PRINT 850                                                        IMPO
  850 FORMAT(//,' EIGENVALUES OF A-BF')                                IMPO
      CALL PRNT(DUMMY(N6),NDUM1,0,3)                                   IMPO
C                                                                      IMPO
      RETURN                                                           IMPO
      END                                                              IMPO
```

```
      SUBROUTINE READ1 (A,NA,NZ,NAM)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(1),NA(2),NZ(2)
      IF  (NZ(1).EQ.0)  GO TO 410
      NR=NZ(1)
      NC=NZ(2)
      NLST=NR*NC
      IF( NLST .LT. 1 .OR. NR .LT. 1 ) GO TO 16
      DO 400 I = 1, NR
  400 READ (5,101) (A(  J), J = I,NLST,NR)
      NA(1)=NR
      NA(2)=NC
  410 CALL  PRNT (A,NA,NAM,1)
  101 FORMAT(8D10.2)
      RETURN
   16 CALL LNCNT(1)
      WRITE  (6,916)  NAM,NR,NC
  916 FORMAT  (' ERROR IN READ1   MATRIX ',A4,' HAS NA=',2I6)
      RETURN
      END
```

```
      SUBROUTINE BALANC(NM,N,A,LOW,IGH,SCALE)                           BAL00
      IMPLICIT REAL*8 (A-H,O-Z)                                         BAL00
      INTEGER I,J,K,L,M,N,JJ,NM,IGH,LOW,IEXC                            BAL00
      DIMENSION A(NM,N),SCALE(N)                                        BAL00
C     REAL C,F,G,R,S,B2,RADIX                                           BAL00
C     REAL DABS                                                         BAL00
      LOGICAL NOCONV                                                    BAL00
C                                                                       BAL00
C                                                                       BAL00
C     ********** RADIX IS A MACHINE DEPENDENT PARAMETER SPECIFYING      BAL00
C                THE BASE OF THE MACHINE FLOATING POINT REPRESENTATION. BAL00
C                                                                       BAL00
C                                                                       BAL00
      RADIX = 16.                                                       BAL00
C                                                                       BAL00
      B2 = RADIX * RADIX                                                BAL00
      K = 1                                                             BAL00
      L = N                                                             BAL00
      GO TO 100                                                         BAL00
C     ********** IN-LINE PROCEDURE FOR ROW AND                          BAL00
C                COLUMN EXCHANGE **********                             BAL00
   20 SCALE(M) = J                                                      BAL00
      IF (J .EQ. M) GO TO 50                                            BAL00
C                                                                       BAL00
      DO 30 I = 1, L                                                    BAL00
         F = A(I,J)                                                     BAL00
         A(I,J) = A(I,M)                                                BAL00
         A(I,M) = F                                                     BAL00
   30 CONTINUE                                                          BAL00
C                                                                       BAL00
      DO 40 I = K, N                                                    BAL00
         F = A(J,I)                                                     BAL00
         A(J,I) = A(M,I)                                                BAL00
         A(M,I) = F                                                     BAL00
   40 CONTINUE                                                          BAL00
C                                                                       BAL00
   50 GO TO (80,130), IEXC                                              BAL00
C     ********** SEARCH FOR ROWS ISOLATING AN EIGENVALUE                BAL00
C                AND PUSH THEM DOWN **********                          BAL00
   80 IF (L .EQ. 1) GO TO 280                                           BAL00
      L = L - 1                                                         BAL00
C     ********** FOR J=L STEP -1 UNTIL 1 DO -- **********               BAL00
  100 DO 120 JJ = 1, L                                                  BAL00
         J = L + 1 - JJ                                                 BAL00
C                                                                       BAL00
         DO 110 I = 1, L                                                BAL00
            IF (I .EQ. J) GO TO 110                                     BAL00
            IF (A(J,I) .NE. 0.0D0) GO TO 120                            BAL00
  110    CONTINUE                                                       BAL00
C                                                                       BAL00
         M = L                                                          BAL00
         IEXC = 1                                                       BAL00
         GO TO 20                                                       BAL00
  120 CONTINUE                                                          BAL00
C                                                                       BAL00
      GO TO 140                                                         BAL00
C     ********** SEARCH FOR COLUMNS ISOLATING AN EIGENVALUE             BAL00
C                AND PUSH THEM LEFT **********                          BAL00
  130 K = K + 1                                                         BAL00
C                                                                       BAL00
  140 DO 170 J = K, L                                                   BAL00
C                                                                       BAL00
         DO 150 I = K, L                                                BAL00
```

```
            IF (I .EQ. J) GO TO 150
            IF (A(I,J) .NE. 0.0D0) GO TO 170
  150    CONTINUE
C
         M = K
         IEXC = 2
         GO TO 20
  170 CONTINUE
C     ********** NOW BALANCE THE SUBMATRIX IN ROWS K TO L **********
      DO 180 I = K, L
  180 SCALE(I) = 1.0D0
C     ********** ITERATIVE LOOP FOR NORM REDUCTION **********
  190 NOCONV = .FALSE.
C
      DO  270  I=K, L
         C = 0.0D0
         R = 0.0D0
C
         DO 200 J = K, L
            IF (J .EQ. I) GO TO 200
            C = C + DABS(A(J,I))
            R = R + DABS(A(I,J))
  200    CONTINUE
C     ********** GUARD AGAINST ZERO C OR R DUE TO UNDERFLOW **********
         IF (C .EQ. 0.0D0 .OR. R .EQ. 0.0D0) GO TO 270
         G = R / RADIX
         F = 1.0D0
         S = C + R
  210    IF (C .GE. G) GO TO 220
         F = F * RADIX
         C = C * B2
         GO TO 210
  220    G = R * RADIX
  230    IF (C .LT. G) GO TO 240
         F = F / RADIX
         C = C / B2
         GO TO 230
C     ********** NOW BALANCE **********
  240    IF ((C + R) / F .GE. 0.95 * S) GO TO 270
         G = 1.0D0 / F
         SCALE(I) = SCALE(I) * F
         NOCONV = .TRUE.
C
         DO 250 J = K, N
  250    A(I,J) = A(I,J) * G
C
         DO 260 J = 1, L
  260    A(J,I) = A(J,I) * F
C
  270 CONTINUE
C
      IF (NOCONV) GO TO 190
C
  280 LOW = K
      IGH = L
      RETURN
C     ********** LAST CARD OF BALANC **********
      END
```

```
      SUBROUTINE ELMHES(NM,N,LOW,IGH,A,INT)
      IMPLICIT REAL*8 (A-H,O-Z)
      INTEGER I,J,M,N,LA,NM,IGH,KP1,LOW,MM1,MP1
      DIMENSION A(NM,N)
C     REAL X,Y
C     REAL DABS
      INTEGER INT(IGH)
C
      LA = IGH - 1
      KP1 = LOW + 1
      IF (LA .LT. KP1) GO TO 200
C
      DO 180 M = KP1, LA
         MM1 = M - 1
         X = 0.0D0
         I = M
C
         DO 100 J = M, IGH
            IF (DABS(A(J,MM1)) .LE. DABS(X)) GO TO 100
            X = A(J,MM1)
            I = J
  100    CONTINUE
C
         INT(M) = I
         IF (I .EQ. M) GO TO 130
C    ********** INTERCHANGE ROWS AND COLUMNS OF A **********
         DO 110 J = MM1, N
            Y = A(I,J)
            A(I,J) = A(M,J)
            A(M,J) = Y
  110    CONTINUE
C
         DO 120 J = 1, IGH
            Y = A(J,I)
            A(J,I) = A(J,M)
            A(J,M) = Y
  120    CONTINUE
C    ********** END INTERCHANGE **********
  130    IF (X .EQ. 0.0D0) GO TO 180
         MP1 = M + 1
C
         DO 160 I = MP1, IGH
            Y = A(I,MM1)
            IF (Y .EQ. 0.0D0) GO TO 160
            Y = Y / X
            A(I,MM1) = Y
C
            DO 140 J = M, N
  140       A(I,J) = A(I,J) - Y * A(M,J)
C
            DO 150 J = 1, IGH
  150       A(J,M) = A(J,M) + Y * A(J,I)
C
  160    CONTINUE
C
  180 CONTINUE
C
  200 RETURN
C    ********** LAST CARD OF ELMHES **********
      END
```

```
      SUBROUTINE HQR(NM,N,LOW,IGH,H,WR,WI,IERR)
      IMPLICIT REAL*8 (A-H,O-Z)
       REAL*8 NORM,MACHEP
      INTEGER I,J,K,L,M,N,EN,LL,MM,NA,NM,IGH,ITS,LOW,MP2,ENM2,IERR
      DIMENSION H(NM,N),WR(N),WI(N)
C     REAL P,Q,R,S,T,W,X,Y,ZZ,NORM,MACHEP
      REAL*8 DSQRT,DABS,DSIGN
C     INTEGER MIN0
      LOGICAL NOTLAS
C
C
C     ********** MACHEP IS A MACHINE DEPENDENT PARAMETER SPECIFYING
C                THE RELATIVE PRECISION OF FLOATING POINT ARITHMETIC.
C
C
      MACHEP = 16.**(-13)
C
      IERR = 0
      NORM = 0.0D0
      K = 1
C     ********** STORE ROOTS ISOLATED BY BALANC
C                AND COMPUTE MATRIX NORM **********
      DO 50 I = 1, N
C
         DO 40 J = K, N
   40    NORM = NORM + DABS(H(I,J))
C
         K = I
         IF (I .GE. LOW .AND. I .LE. IGH) GO TO 50
         WR(I) = H(I,I)
         WI(I) = 0.0D0
   50 CONTINUE
C
      EN = IGH
      T = 0.0D0
C     ********** SEARCH FOR NEXT EIGENVALUES **********
   60 IF (EN .LT. LOW) GO TO 1001
      ITS = 0
      NA = EN - 1
      ENM2 = NA - 1
C     ********** LOOK FOR SINGLE SMALL SUB-DIAGONAL ELEMENT
C                FOR L=EN STEP -1 UNTIL LOW DO -- **********
   70 DO 80 LL = LOW, EN
         L = EN + LOW - LL
         IF (L .EQ. LOW) GO TO 100
         S = DABS(H(L-1,L-1)) + DABS(H(L,L))
         IF (S .EQ. 0.0D0) S = NORM
         IF (DABS(H(L,L-1)) .LE. MACHEP * S) GO TO 100
   80 CONTINUE
C     ********** FORM SHIFT **********
  100 X = H(EN,EN)
      IF (L .EQ. EN) GO TO 270
      Y = H(NA,NA)
      W = H(EN,NA) * H(NA,EN)
      IF (L .EQ. NA) GO TO 280
      IF (ITS .EQ. 30) GO TO 1000
      IF (ITS .NE. 10 .AND. ITS .NE. 20) GO TO 130
C     ********** FORM EXCEPTIONAL SHIFT **********
      T = T + X
C
      DO 120 I = LOW, EN
  120 H(I,I) = H(I,I) - X
C
```

```
      S = DABS(H(EN,NA)) + DABS(H(NA,ENM2))
      X = 0.75 * S
      Y = X
      W = -0.4375 * S * S
  130 ITS = ITS + 1
C     ********** LOOK FOR TWO CONSECUTIVE SMALL
C                SUB-DIAGONAL ELEMENTS.
C                FOR M=EN-2 STEP -1 UNTIL L DO -- **********
      DO 140 MM = L, ENM2
         M = ENM2 + L - MM
         ZZ = H(M,M)
         R = X - ZZ
         S = Y - ZZ
         P = (R * S - W) / H(M+1,M) + H(M,M+1)
         Q = H(M+1,M+1) - ZZ - R - S
         R = H(M+2,M+1)
         S = DABS(P) + DABS(Q) + DABS(R)
         P = P / S
         Q = Q / S
         R = R / S
         IF (M .EQ. L) GO TO 150
         IF (DABS(H(M,M-1)) * (DABS(Q) + DABS(R)) .LE. MACHEP * DABS(P)
     X    * (DABS(H(M-1,M-1)) + DABS(ZZ) + DABS(H(M+1,M+1)))) GO TO 150
  140 CONTINUE
C
  150 MP2 = M + 2
C
      DO 160 I = MP2, EN
         H(I,I-2) = 0.0D0
         IF (I .EQ. MP2) GO TO 160
         H(I,I-3) = 0.0D0
  160 CONTINUE
C     ********** DOUBLE QR STEP INVOLVING ROWS L TO EN AND
C                COLUMNS M TO EN **********
      DO 260 K = M, NA
         NOTLAS = K .NE. NA
         IF (K .EQ. M) GO TO 170
         P = H(K,K-1)
         Q = H(K+1,K-1)
         R = 0.0D0
         IF (NOTLAS) R = H(K+2,K-1)
         X = DABS(P) + DABS(Q) + DABS(R)
         IF (X .EQ. 0.0D0) GO TO 260
         P = P / X
         Q = Q / X
         R = R / X
  170    S = DSIGN(DSQRT(P*P+Q*Q+R*R),P)
         IF (K .EQ. M) GO TO 180
         H(K,K-1) = -S * X
      GO TO 190
  180    IF (L .NE. M) H(K,K-1) = -H(K,K-1)
  190    P = P + S
         X = P / S
         Y = Q / S
         ZZ = R / S
         Q = Q / P
         R = R / P
C     ********** ROW MODIFICATION **********
         DO 210 J = K, EN
            P = H(K,J) + Q * H(K+1,J)
            IF (.NOT. NOTLAS) GO TO 200
            P = P + R * H(K+2,J)
            H(K+2,J) = H(K+2,J) - P * ZZ
```

```
  200       H(K+1,J) = H(K+1,J) - P * Y
            H(K,J) = H(K,J) - P * X
  210    CONTINUE
C
         J = MIN0(EN,K+3)
C     *********** COLUMN MODIFICATION ***********
         DO 230 I = L, J
            P = X * H(I,K) + Y * H(I,K+1)
            IF (.NOT. NOTLAS) GO TO 220
            P = P + ZZ * H(I,K+2)
            H(I,K+2) = H(I,K+2) - P * R
  220       H(I,K+1) = H(I,K+1) - P * Q
            H(I,K) = H(I,K) - P
  230    CONTINUE
C
  260 CONTINUE
C
      GO TO 70
C     *********** ONE ROOT FOUND ***********
  270 WR(EN) = X + T
      WI(EN) = 0.0D0
      EN = NA
      GO TO 60
C     *********** TWO ROOTS FOUND ***********
  280 P = (Y - X) / 2.0D0
      Q = P * P + W
      ZZ = DSQRT(DABS(Q))
      X = X + T
      IF (Q .LT. 0.0D0) GO TO 320
C     *********** REAL PAIR ***********
      ZZ = P + DSIGN(ZZ,P)
      WR(NA) = X + ZZ
      WR(EN) = WR(NA)
      IF (ZZ .NE. 0.0D0) WR(EN) = X - W / ZZ
      WI(NA) = 0.0D0
      WI(EN) = 0.0D0
      GO TO 330
C     *********** COMPLEX PAIR ***********
  320 WR(NA) = X + P
      WR(EN) = X + P
      WI(NA) = ZZ
      WI(EN) = -ZZ
  330 EN = ENM2
      GO TO 60
C     *********** SET ERROR -- NO CONVERGENCE TO AN
C                EIGENVALUE AFTER 30 ITERATIONS ***********
 1000 IERR = EN
 1001 RETURN
C     *********** LAST CARD OF HQR ***********
      END
```

```
      SUBROUTINE INVIT(NM,N,A,WR,WI,SELECT,MM,M,Z,IERR,RM1,RV1,RV2)
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*8 NORM,NORMV,ILAMBD,MACHEP
      INTEGER I,J,K,L,M,N,S,II,IP,MM,MP,NM,NS,N1,UK,IP1,ITS,KM1,IERR
      DIMENSION A(NM,N),WR(N),WI(N),Z(NM,MM),RM1(N,N),RV1(N),RV2(N)
C     REAL T,W,X,Y,EPS3,NORM,NORMV,GROWTO,ILAMBD,MACHEP,RLAMBD,UKROOT
      REAL*8 DSQRT,CDABS,DABS,DFLOAT
      INTEGER IABS
      LOGICAL*1 SELECT(N)
      COMPLEX*16 Z3,DCMPLX
      REAL*8 DREAL,DIMAG
C
      MACHEP = 16.**(-13)
C
      IERR = 0
      UK = 0
      S = 1
C     ********** IP = 0, REAL EIGENVALUE
C                     1, FIRST OF CONJUGATE COMPLEX PAIR
C                    -1, SECOND OF CONJUGATE COMPLEX PAIR **********
      IP = 0
      N1 = N - 1
C
      DO 980 K = 1, N
         IF (WI(K) .EQ. 0.0D0 .OR. IP .LT. 0) GO TO 100
         IP = 1
         IF (SELECT(K) .AND. SELECT(K+1)) SELECT(K+1) = .FALSE.
  100    IF (.NOT. SELECT(K)) GO TO 960
         IF (WI(K) .NE. 0.0D0) S = S + 1
         IF (S .GT. MM) GO TO 1000
         IF (UK .GE. K) GO TO 200
C     ********** CHECK FOR POSSIBLE SPLITTING **********
         DO 120 UK = K, N
            IF (UK .EQ. N) GO TO 140
            IF (A(UK+1,UK) .EQ. 0.0D0) GO TO 140
  120    CONTINUE
C     ********** COMPUTE INFINITY NORM OF LEADING UK BY UK
C                (HESSENBERG) MATRIX **********
  140    NORM = 0.0D0
         MP = 1
C
         DO 180 I = 1, UK
            X = 0.0D0
C
            DO 160 J = MP, UK
  160       X = X + DABS(A(I,J))
C
            IF (X .GT. NORM) NORM = X
            MP = I
  180    CONTINUE
C     ********** EPS3 REPLACES ZERO PIVOT IN DECOMPOSITION
C                AND CLOSE ROOTS ARE MODIFIED BY EPS3**********
         IF (NORM .EQ. 0.0D0) NORM = 1.0D0
         EPS3 = MACHEP * NORM
C     ********** GROWTO IS THE CRITERION FOR THE GROWTH **********
         UKROOT = DSQRT(DFLOAT(UK))
         GROWTO = 1.0D-1 / UKROOT
  200    RLAMBD = WR(K)
         ILAMBD = WI(K)
         IF (K .EQ. 1) GO TO 280
         KM1 = K - 1
         GO TO 240
C     ********** PERTURB EIGENVALUE IF IT IS CLOSE
```

```
C                         TO ANY PREVIOUS EIGENVALUE **********
  220   RLAMBD = RLAMBD + EPS3
C       ********** FOR I=K-1 STEP -1 UNTIL 1 DO -- **********
  240   DO 260 II = 1, KM1
           I = K - II
           IF (SELECT(I) .AND. DABS(WR(I)-RLAMBD) .LT. EPS3 .AND.
     X         DABS(WI(I)-ILAMBD) .LT. EPS3) GO TO 220
  260   CONTINUE
C
        WR(K) = RLAMBD
C       ********** PERTURB CONJUGATE EIGENVALUE TO MATCH **********
        IP1 = K + IP
        WR(IP1) = RLAMBD
C       ********** FORM UPPER HESSENBERG A-RLAMBD*I (TRANSPOSED)
C                  AND INITIAL REAL VECTOR **********
  280   MP = 1
C
        DO 320 I = 1, UK
C
           DO 300 J = MP, UK
  300      RM1(J,I) = A(I,J)
C
           RM1(I,I) = RM1(I,I) - RLAMBD
           MP = I
           RV1(I) = EPS3
  320   CONTINUE
C
        ITS = 0
        IF (ILAMBD .NE. 0.0D0) GO TO 520
C       ********** REAL EIGENVALUE.
C                  TRIANGULAR DECOMPOSITION WITH INTERCHANGES,
C                  REPLACING ZERO PIVOTS BY EPS3 **********
        IF (UK .EQ. 1) GO TO 420
C
        DO 400 I = 2, UK
           MP = I - 1
           IF (DABS(RM1(MP,I)) .LE. DABS(RM1(MP,MP))) GO TO 360
C
           DO 340 J = MP, UK
              Y = RM1(J,I)
              RM1(J,I) = RM1(J,MP)
              RM1(J,MP) = Y
  340      CONTINUE
C
  360      IF (RM1(MP,MP) .EQ. 0.0D0) RM1(MP,MP) = EPS3
           X = RM1(MP,I) / RM1(MP,MP)
           IF (X .EQ. 0.0D0) GO TO 400
C
           DO 380 J = I, UK
  380      RM1(J,I) = RM1(J,I) - X * RM1(J,MP)
C
  400   CONTINUE
C
  420   IF (RM1(UK,UK) .EQ. 0.0D0) RM1(UK,UK) = EPS3
C       ********** BACK SUBSTITUTION FOR REAL VECTOR
C                  FOR I=UK STEP -1 UNTIL 1 DO -- **********
  440   DO 500 II = 1, UK
           I = UK + 1 - II
           Y = RV1(I)
           IF (I .EQ. UK) GO TO 480
           IP1 = I + 1
C
           DO 460 J = IP1, UK
```

ORIGINAL PAGE IS OF POOR QUALITY

```
  460     Y = Y - RM1(J,I) * RV1(J)
C
  480     RV1(I) = Y / RM1(I,I)
  500  CONTINUE
C
       GO TO 740
C     ********** COMPLEX EIGENVALUE.
C                TRIANGULAR DECOMPOSITION WITH INTERCHANGES,
C                REPLACING ZERO PIVOTS BY EPS3.  STORE IMAGINARY
C                PARTS IN UPPER TRIANGLE STARTING AT (1,3) **********
  520  NS = N - S
       Z(1,S-1) = -ILAMBD
       Z(1,S) = 0.0D0
       IF (N .EQ. 2) GO TO 550
       RM1(1,3) = -ILAMBD
       Z(1,S-1) = 0.0D0
       IF (N .EQ. 3) GO TO 550
C
       DO 540 I = 4, N
  540  RM1(1,I) = 0.0D0
C
  550  DO 640 I = 2, UK
          MP = I - 1
          W = RM1(MP,I)
          IF (I .LT. N) T = RM1(MP,I+1)
          IF (I .EQ. N) T = Z(MP,S-1)
          X = RM1(MP,MP) * RM1(MP,MP) + T * T
          IF (W * W .LE. X) GO TO 580
          X = RM1(MP,MP) / W
          Y = T / W
          RM1(MP,MP) = W
          IF (I .LT. N) RM1(MP,I+1) = 0.0D0
          IF (I .EQ. N) Z(MP,S-1) = 0.0D0
C
          DO 560 J = I, UK
             W = RM1(J,I)
             RM1(J,I) = RM1(J,MP) - X * W
             RM1(J,MP) = W
             IF (J .LT. N1) GO TO 555
             L = J - NS
             Z(I,L) = Z(MP,L) - Y * W
             Z(MP,L) = 0.0D0
             GO TO 560
  555        RM1(I,J+2) = RM1(MP,J+2) - Y * W
             RM1(MP,J+2) = 0.0D0
  560     CONTINUE
C
          RM1(I,I) = RM1(I,I) - Y * ILAMBD
          IF (I .LT. N1) GO TO 570
          L = I - NS
          Z(MP,L) = -ILAMBD
          Z(I,L) = Z(I,L) + X * ILAMBD
          GO TO 640
  570     RM1(MP,I+2) = -ILAMBD
          RM1(I,I+2) = RM1(I,I+2) + X * ILAMBD
          GO TO 640
  580     IF (X .NE. 0.0D0) GO TO 600
          RM1(MP,MP) = EPS3
          IF (I .LT. N) RM1(MP,I+1) = 0.0D0
          IF (I .EQ. N) Z(MP,S-1) = 0.0D0
          T = 0.0D0
          X = EPS3 * EPS3
  600     W = W / X
```

```
            X = RM1(MP,MP) * W
            Y = -T * W
C
            DO 620 J = I, UK
               IF (J .LT. N1) GO TO 610
               L = J - NS
               T = Z(MP,L)
               Z(I,L) = -X * T - Y * RM1(J,MP)
               GO TO 615
  610          T = RM1(MP,J+2)
               RM1(I,J+2) = -X * T - Y * RM1(J,MP)
  615          RM1(J,I) = RM1(J,I) - X * RM1(J,MP) + Y * T
  620       CONTINUE
C
            IF (I .LT. N1) GO TO 630
            L = I - NS
            Z(I,L) = Z(I,L) - ILAMBD
            GO TO 640
  630       RM1(I,I+2) = RM1(I,I+2) - ILAMBD
  640    CONTINUE
C
         IF (UK .LT. N1) GO TO 650
         L = UK - NS
         T = Z(UK,L)
         GO TO 655
  650    T = RM1(UK,UK+2)
  655    IF (RM1(UK,UK) .EQ. 0.0D0 .AND. T .EQ. 0.0D0) RM1(UK,UK) = EPS3
C     ********** BACK SUBSTITUTION FOR COMPLEX VECTOR
C                FOR I=UK STEP -1 UNTIL 1 DO -- **********
  660    DO 720 II = 1, UK
            I = UK + 1 - II
            X = RV1(I)
            Y = 0.0D0
            IF (I .EQ. UK) GO TO 700
            IP1 = I + 1
C
            DO 680 J = IP1, UK
               IF (J .LT. N1) GO TO 670
               L = J - NS
               T = Z(I,L)
               GO TO 675
  670          T = RM1(I,J+2)
  675          X = X - RM1(J,I) * RV1(J) + T * RV2(J)
               Y = Y - RM1(J,I) * RV2(J) - T * RV1(J)
  680       CONTINUE
C
  700       IF (I .LT. N1) GO TO 710
            L = I - NS
            T = Z(I,L)
            GO TO 715
  710       T = RM1(I,I+2)
  715       Z3 = DCMPLX(X,Y) / DCMPLX(RM1(I,I),T)
            RV1(I) = DREAL(Z3)
            RV2(I) = DIMAG(Z3)
  720    CONTINUE
C     ********** ACCEPTANCE TEST FOR REAL OR COMPLEX
C                EIGENVECTOR AND NORMALIZATION **********
  740    ITS = ITS + 1
         NORM = 0.0D0
         NORMV = 0.0D0
C
         DO 780 I = 1, UK
            IF (ILAMBD .EQ. 0.0D0) X = DABS(RV1(I))
```

```
            IF (ILAMBD .NE. 0.0D0) X = CDABS(DCMPLX(RV1(I),RV2(I)))
            IF (NORMV .GE. X) GO TO 760
            NORMV = X
            J = I
  760       NORM = NORM + X
  780    CONTINUE
C
         IF (NORM .LT. GROWTO) GO TO 840
C     ********** ACCEPT VECTOR **********
         X = RV1(J)
         IF (ILAMBD .EQ. 0.0D0) X = 1.0D0 / X
         IF (ILAMBD .NE. 0.0D0) Y = RV2(J)
C
         DO 820 I = 1, UK
            IF (ILAMBD .NE. 0.0D0) GO TO 800
            Z(I,S) = RV1(I) * X
            GO TO 820
  800       Z3 = DCMPLX(RV1(I),RV2(I)) / DCMPLX(X,Y)
            Z(I,S-1) = DREAL(Z3)
            Z(I,S) = DIMAG(Z3)
  820    CONTINUE
C
         IF (UK .EQ. N) GO TO 940
         J = UK + 1
         GO TO 900
C     ********** IN-LINE PROCEDURE FOR CHOOSING
C                A NEW STARTING VECTOR **********
  840    IF (ITS .GE. UK) GO TO 880
         X = UKROOT
         Y = EPS3 / (X + 1.0D0)
         RV1(1) = EPS3
C
         DO 860 I = 2, UK
  860    RV1(I) = Y
C
         J = UK - ITS + 1
         RV1 J) = RV1(J) - EPS3 * X
         IF (ILAMBD .EQ. 0.0D0) GO TO 440
         GO TO 660
C     ********** SET ERROR -- UNACCEPTED EIGENVECTOR **********
  880    J = 1
         IERR = -K
C     ********** SET REMAINING VECTOR COMPONENTS TO ZERO **********
  900    DO 920 I = J, N
            Z(I,S) = 0.0D0
            IF (ILAMBD .NE. 0.0D0) Z(I,S-1) = 0.0D0
  920    CONTINUE
C
  940    S = S + 1
  960    IF (IP .EQ. (-1)) IP = 0
         IF (IP .EQ. 1) IP = -1
  980 CONTINUE
C
      GO TO 1001
C     ********** SET ERROR -- UNDERESTIMATE OF EIGENVECTOR
C                SPACE REQUIRED **********
 1000 IF (IERR .NE. 0) IERR = IERR - N
      IF (IERR .EQ. 0) IERR = -(2 * N + 1)
 1001 M = S - 1 - IABS(IP)
      RETURN
C     ********** LAST CARD OF INVIT **********
      END
```

```
      SUBROUTINE ELMBAK(NM,LOW,IGH,A,INT,M,Z)
      IMPLICIT REAL*8 (A-H,O-Z)
      INTEGER I,J,M,LA,MM,MP,NM,IGH,KP1,LOW,MP1
      DIMENSION A(NM,IGH),Z(NM,M)
C     REAL X
      INTEGER INT(IGH)
C
C
      IF (M .EQ. 0) GO TO 200
      LA = IGH - 1
      KP1 = LOW + 1
      IF (LA .LT. KP1) GO TO 200
C     ********** FOR MP=IGH-1 STEP -1 UNTIL LOW+1 DO -- **********
      DO 140 MM = KP1, LA
         MP = LOW + IGH - MM
         MP1 = MP + 1
C
         DO 110 I = MP1, IGH
            X = A(I,MP-1)
            IF (X .EQ. 0.0) GO TO 110
C
            DO 100 J = 1, M
  100       Z(I,J) = Z(I,J) + X * Z(MP,J)
C
  110    CONTINUE
C
         I = INT(MP)
         IF (I .EQ. MP) GO TO 140
C
         DO 130 J = 1, M
            X = Z(I,J)
            Z(I,J) = Z(MP,J)
            Z(MP,J) = X
  130    CONTINUE
C
  140 CONTINUE
C
  200 RETURN
C     ********** LAST CARD OF ELMBAK **********
      END
```

```
      SUBROUTINE BALBAK(NM,N,LOW,IGH,SCALE,M,Z)
      IMPLICIT REAL*8 (A-H,O-Z)
      INTEGER I,J,K,M,N,II,NM,IGH,LOW
      DIMENSION SCALE(N),Z(NM,M)
C     REAL S
      IF (M .EQ. 0) GO TO 200
      IF (IGH .EQ. LOW) GO TO 120
C
      DO 110 I = LOW, IGH
         S = SCALE(I)
C     ********** LEFT HAND EIGENVECTORS ARE BACK TRANSFORMED
C                IF THE FOREGOING STATEMENT IS REPLACED BY
C                S=1.0/SCALE(I). **********
         DO 100 J = 1, M
  100    Z(I,J) = Z(I,J) * S
C
  110 CONTINUE
C     ********* FOR I=LOW-1 STEP -1 UNTIL 1,
C               IGH+1 STEP 1 UNTIL N DO -- **********
  120 DO 140 II = 1, N
         I = II
         IF (I .GE. LOW .AND. I .LE. IGH) GO TO 140
         IF (I .LT. LOW) I = LOW - II
         K = SCALE(I)
         IF (K .EQ. I) GO TO 140
C
         DO 130 J = 1, M
            S = Z(I,J)
            Z(I,J) = Z(K,J)
            Z(K,J) = S
  130    CONTINUE
C
  140 CONTINUE
C
  200 RETURN
C     ********** LAST CARD OF BALBAK **********
      END
```

```
      SUBROUTINE DETFAC(NMAX,N,A,IPIVOT,IDET,DETERM,ISCALE,WK,IERR)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(NMAX,1),IPIVOT(1),WK(1)
C
C
      ISCALE=0
      NM1=N-1
      IERR=0
C
C     DETERMINANT CALCULATION TEST
C
      IF(IDET.EQ.1)GO TO 230
C
C     TEST FOR A SCALAR MATRIX
C
      IF(NM1.GT.0)GO TO 20
      DETERM=A(1,1)
      RETURN
C
C     COMPUTE SCALING FACTORS
C
   20 CONTINUE
      DO 60 I=1,N
      P=0.0
      DO 30 J=1,N
      Q=DMAX1(P,DABS(A(I,J)))
      IF(Q.GT.P)P=Q
   30 CONTINUE
      IF(P)60,40,60
   40 DETERM=0.0
      IERR=1
      RETURN
   60 WK(I)=P
C
      DO 210 M=1,NM1
C
C     PIVOTAL LOGIC SETUP
C
      P=0.0
      DO 110 I=M,N
      Q=DABS(A(I,M)/WK(I))
      IF(Q-P)110,110,100
  100 P=Q
      IP=I
  110 CONTINUE
C
      IPIVOT(M)=IP
C
      IF(P.EQ.0.)GO TO 40
      IF(M.EQ.IP)GO TO 155
C
C     PIVOT THE M-TH ROW OF THE A MATRIX
C
      DO 150 I=1,N
      P=A(IP,I)
      A(IP,I)=A(M,I)
  150 A(M,I)=P
C
      P=WK(IP)
      WK(IP)=WK(M)
      WK(M)=P
C
  155 MP1=M+1
```

```
C
C     L/U FACTORIZATION LOGIC
C
      P=A(M,M)
      DO 180 I=MP1,N
      A(I,M)=A(I,M)/P
      Q=A(I,M)
      DO 180 K=MP1,N
  180 A(I,K)=A(I,K)-Q*A(M,K)
C
  210 CONTINUE
C
      IPIVOT(N)=N
      IF (A(N,N) .EQ. 0.0) GO TO 40
C
C     CALCULATION OF THE DETERMINANT OF A
C
      IF(IDET.EQ.0)RETURN
C
  230 SIGN=1.0
      DETERM=1.0
C
C     ADJUST SIGN OF DETERMINANT DUE TO PIVOTAL STRATEGY
C
      DO 250 I=1,NM1
      IF(I-IPIVOT(I))240,250,240
  240 SIGN=-SIGN
  250 CONTINUE
C
      DO 340 I=1,N
      P=A(I,I)
C
  260 CONTINUE
      IF(R1.GT.DABS(P))GO TO 280
      P=P*R2
      ISCALE=ISCALE+1
      GO TO 260
C
  280 CONTINUE
      IF(R2.LT.DABS(P))GO TO 290
      P=P*R1
      ISCALE=ISCALE-1
      GO TO 280
C
  290 DETERM=DETERM*P
C
  300 CONTINUE
      IF(R1.GT.DABS(DETERM))GO TO 320
      DETERM=DETERM*R2
      ISCALE=ISCALE+1
      GO TO 300
C
  320 CONTINUE
      IF(R2.LT.DABS(DETERM))GO TO 340
      DETERM=DETERM*R1
      ISCALE=ISCALE-1
      GO TO 320
C
  340 CONTINUE
C
      DETERM=DETERM*SIGN
C
      RETURN
```

126 END

DET

```
      SUBROUTINE AXPXB(A,U,M,NA,NU,B,V,N,NB,NV,C,NC,EPSA,
     1EPSB,FAIL)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION
     1A(NA,1),U(NU,1),B(NB,1),V(NV,1),C(NC,1)
      INTEGER
C
     1 FAIL
      M1 = M+1
      MM1 = M-1
      N1 = N+1
      NM1 = N-1
C IF REQUIRED, REDUCE A TO UPPER REAL SCHUR FORM.
C
      IF(EPSA .LT. 0.) GO TO 35
      DO 10 I=1,M
        DO 10 J=I,M
          TEMP = A(I,J)
          A(I,J) = A(J,I)
          A(J,I) = TEMP
   10 CONTINUE
      CALL HSHLDR(A,M,NA)
      CALL BCKMLT(A,U,M,NA,NU)
      IF(MM1 .EQ. 0) GO TO 25
      DO 20 I=1,MM1
        A(I+1,I) = A(I,M1)
   20 CONTINUE
      CALL SCHUR(A,U,M,NA,NU,EPSA,FAIL)
      IF(FAIL .NE. 0) RETURN
   25 DO 30 I=1,M
        DO 30 J=I,M
          TEMP = A(I,J)
          A(I,J) = A(J,I)
          A(J,I) = TEMP
   30 CONTINUE
C
C IF REQUIRED, REDUCE B TO UPPER REAL SCHUR FORM.
C
   35 IF(EPSB .LT. 0.) GO TO 45
      CALL HSHLDR(B,N,NB)
      CALL BCKMLT(B,V,N,NB,NV)
      IF(NM1 .EQ. 0) GO TO 45
      DO 40 I=1,NM1
        B(I+1,I) = B(I,N1)
   40 CONTINUE
      CALL SCHUR(B,V,N,NB,NV,EPSB,FAIL)
      FAIL = -FAIL
      IF(FAIL .NE. 0) RETURN
C
C TRANSFORM C.
C
   45 DO 60 J=1,N
        DO 50 I=1,M
          A(I,M1) = 0.
          DO 50 K=1,M
            A(I,M1) = A(I,M1) + U(K,I)*C(K,J)
   50 CONTINUE
      DO 60 I=1,M
        C(I,J) = A(I,M1)
   60 CONTINUE
      DO 80 I=1,M
        DO 70 J=1,N
          B(N1,J) = 0.
```

```
         DO 70 K=1,N
           B(N1,J) = B(N1,J) + C(I,K)*V(K,J)
   70 CONTINUE
      DO 80 J=1,N
        C(I,J) = B(N1,J)
   80 CONTINUE
C
C SOLVE THE TRANSFORMED SYSTEM.
C
      CALL SHRSLV(A,B,C,M,N,NA,NB,NC)
C
C TRANSFORM C BACK TO THE SOLUTION.
C
      DO 100 J=1,N
        DO 90 I=1,M
          A(I,M1) = 0.
          DO 90 K=1,M
            A(I,M1) = A(I,M1) + U(I,K)*C(K,J)
   90 CONTINUE
      DO 100 I=1,M
        C(I,J) = A(I,M1)
  100 CONTINUE
      DO 120 I=1,M
        DO 110 J=1,N
          B(N1,J) = 0.
          DO 110 K=1,N
            B(N1,J) = B(N1,J) + C(I,K)*V(J,K)
  110     CONTINUE
         DO 120 J=1,N
           C(I,J) = B(N1,J)
  120  CONTINUE
       RETURN
       END
```

```
      SUBROUTINE SHRSLV(A,B,C,M,N,NA,NB,NC)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION
     1A(NA,1),B(NB,1),C(NC,1)
      INTEGER
     1 DK,DL
      COMMON/SLVBLK/T(5,5),P(5),NSYS
      L = 1
   10 LM1 = L-1
      DL = 1
      IF(L .EQ. N) GO TO 15
      IF(B(L+1,L) .NE. 0.) DL = 2
   15 LL = L+DL-1
      IF(L .EQ. 1) GO TO 30
      DO 20 J=L,LL
        DO 20 I=1,M
          DO 20 IB=1,LM1
            C(I,J) = C(I,J) - C(I,IB)*B(IB,J)
   20 CONTINUE
   30 K = 1
   40   KM1 = K-1
        DK = 1
        IF(K .EQ. M) GO TO 45
        IF(A(K,K+1) .NE. 0.) DK = 2
   45   KK = K+DK-1
        IF(K .EQ. 1) GO TO 60
        DO 50 I=K,KK
          DO 50 J=L,LL
            DO 50 JA=1,KM1
              C(I,J) = C(I,J) - A(I,JA)*C(JA,J)
   50   CONTINUE
   60   IF(DL .EQ. 2) GO TO 80
        IF(DK .EQ. 2) GO TO 70
        T(1,1) = A(K,K) + B(L,L)
        IF(T(1,1) .EQ. 0.) STOP
        C(K,L) = C(K,L)/T(1,1)
        GO TO 100
   70   T(1,1) = A(K,K) + B(L,L)
        T(1,2) = A(K,KK)
        T(2,1) = A(KK,K)
        T(2,2) = A(KK,KK) + B(L,L)
        P(1) = C(K,L)
        P(2) = C(KK,L)
        NSYS = 2
        CALL SYSSLV
        C(K,L) = P(1)
        C(KK,L) = P(2)
        GO TO 100
   80   IF(DK .EQ. 2) GO TO 90
        T(1,1) = A(K,K) + B(L,L)
        T(1,2) = B(LL,L)
        T(2,1) = B(L,LL)
        T(2,2) = A(K,K) + B(LL,LL)
        P(1) = C(K,L)
        P(2) = C(K,LL)
        NSYS = 2
        CALL SYSSLV
        C(K,L) = P(1)
        C(K,LL) = P(2)
        GO TO 100
   90   T(1,1) = A(K,K) + B(L,L)
        T(1,2) = A(K,KK)
        T(1,3) = B(LL,L)
```

```
      T(1,4) = 0.                                                       SHR00
      T(2,1) = A(KK,K)                                                  SHR00
      T(2,2) = A(KK,KK) + B(L,L)                                        SHR00
      T(2,3) = 0.                                                       SHR00
      T(2,4) = T(1,3)                                                   SHR00
      T(3,1) = B(L,LL)                                                  SHR00
      T(3,2) = 0.                                                       SHR00
      T(3,3) = A(K,K) + B(LL,LL)                                        SHR00
      T(3,4) = T(1,2)                                                   SHR00
      T(4,1) = 0.                                                       SHR00
      T(4,2) = T(3,1)                                                   SHR00
      T(4,3) = T(2,1)                                                   SHR00
      T(4,4) = A(KK,KK) + B(LL,LL)                                      SHR00
      P(1) = C(K,L)                                                     SHR00
      P(2) = C(KK,L)                                                    SHR00
      P(3) = C(K,LL)                                                    SHR00
      P(4) = C(KK,LL)                                                   SHR00
      NSYS = 4                                                          SHR00
      CALL SYSSLV                                                       SHR00
      C(K,L) = P(1)                                                     SHR00
      C(KK,L) = P(2)                                                    SHR00
      C(K,LL) = P(3)                                                    SHR00
      C(KK,LL) = P(4)                                                   SHR00
  100   K = K + DK                                                      SHR00
        IF(K .LE. M) GO TO 40                                           SHR00
      L = L + DL                                                        SHR00
      IF(L .LE. N) GO TO 10                                             SHR00
      RETURN                                                            SHR00
      END                                                               SHR00
```

```
      SUBROUTINE ATXPXA(A,U,C,N,NA,NU,NC,EPS,FAIL)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION
     1A(NA,1),U(NU,1),C(NC,1)
      INTEGER
     1 FAIL
      N1 = N+1
      NM1 = N-1
C
C IF REQUIRED, REDUCE A TO LOWER REAL SCHUR FORM.
C
      IF(EPS .LT. 0.) GO TO 15
      CALL HSHLDR(A,N,NA)
      CALL BCKMLT(A,U,N,NA,NU)
      DO 10 I=1,NM1
        A(I+1,I) = A(I,N1)
   10 CONTINUE
      CALL SCHUR(A,U,N,NA,NU,EPS,FAIL)
      IF(FAIL .NE. 0) RETURN
C
C TRANSFORM C.
C
   15 DO 20 I=1,N
          C(I,I)=C(I,I)/2.
   20 CONTINUE
      DO 40 I=1,N
        DO 30 J=1,N
          A(N1,J) = 0.
          DO 30 K=I,N
            A(N1,J) = A(N1,J) + C(I,K)*U(K,J)
   30   CONTINUE
          DO 40 J=1,N
          C(I,J) = A(N1,J)
   40 CONTINUE
      DO 60 J=1,N
        DO 50 I=1,N
          A(I,N1) = 0.
          DO 50 K=1,N
            A(I,N1) = A(I,N1) + U(K,I)*C(K,J)
   50   CONTINUE
        DO 60 I=1,N
          C(I,J) = A(I,N1)
   60 CONTINUE
      DO 70 I=1,N
        DO 70 J=I,N
          C(I,J) = C(I,J) + C(J,I)
          C(J,I) = C(I,J)
   70 CONTINUE
C
C SOLVE THE TRANSFORMED SYSTEM.
C
      CALL SYMSLV(A,C,N,NA,NC)
C
C TRANSFORM C BACK TO THE SOLUTION.
C
      DO 80 I=1,N
        C(I,I) = C(I,I)/2.
   80 CONTINUE
      DO 100 I=1,N
        DO 90 J=1,N
          A(N1,J) = 0.
          DO 90 K=I,N
            A(N1,J) = A(N1,J) + C(I,K)*U(J,K)
```

```
   90   CONTINUE                                      ATX00
        DO 100 J=1,N                                  ATX00
          C(I,J) = A(N1,J)                            ATX00
  100 CONTINUE                                        ATX00
      DO 120 J=1,N                                    ATX00
        DO 110 I=1,N                                  ATX00
          A(I,N1) = 0.                                ATX00
          DO 110 K=1,N                                ATX00
            A(I,N1) = A(I,N1) + U(I,K)*C(K,J)         ATX00
  110   CONTINUE                                      ATX00
        DO 120 I=1,N                                  ATX00
          C(I,J) = A(I,N1)                            ATX00
  120 CONTINUE                                        ATX00
      DO 130 I=1,N                                    ATX00
        DO 130 J=I,N                                  ATX00
          C(I,J) = C(I,J) + C(J,I)                    ATX00
          C(J,I) = C(I,J)                             ATX00
  130 CONTINUE                                        ATX00
      RETURN                                          ATX00
      END                                             ATX00
```

```
      SUBROUTINE SYMSLV(A,C,N,NA,NC)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION
     1A(NA,1),C(NC,1)
      INTEGER
     1 DK,DL
      COMMON/SLVBLK/T(5,5),P(5),NSYS
      L = 1
   10    DL = 1
         IF(L .EQ. N) GO TO 20
         IF(A(L+1,L) .NE. 0.) DL = 2
   20    LL = L+DL-1
         K = L
   30       KM1 = K-1
            DK = 1
            IF(K .EQ. N) GO TO 35
            IF(A(K+1,K) .NE. 0.) DK = 2
   35       KK = K+DK-1
            IF(K .EQ. L) GO TO 45
            DO 40 I=K,KK
               DO 40 J=L,LL
                  DO 40 IA=L,KM1
                     C(I,J) = C(I,J) - A(IA,I)*C(IA,J)
   40       CONTINUE
   45       IF(DL .EQ. 2) GO TO 60
            IF(DK .EQ. 2 ) GO TO 50
            T(1,1) = A(K,K) + A(L,L)
            IF(T(1,1) .EQ. 0.) STOP
            C(K,L) = C(K,L)/T(1,1)
            GO TO 90
   50       T(1,1) = A(K,K) + A(L,L)
            T(1,2) = A(KK,K)
            T(2,1) = A(K,KK)
            T(2,2) = A(KK,KK) + A(L,L)
            P(1) = C(K,L)
            P(2) = C(KK,L)
            NSYS = 2
            CALL SYSSLV
            C(K,L) = P(1)
         C(KK,L) = P(2)
            GO TO 90
   60       IF(DK .EQ. 2) GO TO 70
            T(1,1) = A(K,K) + A(L,L)
            T(1,2) = A(LL,L)
            T(2,1) = A(L,LL)
            T(2,2) = A(K,K) + A(LL,LL)
            P(1) = C(K,L)
            P(2) = C(K,LL)
            NSYS = 2
            CALL SYSSLV
            C(K,L) = P(1)
            C(K,LL) = P(2)
            GO TO 90
   70       IF(K .NE. L) GO TO 80
            T(1,1) = A(L,L)
            T(1,2) = A(LL,L)
            T(1,3) = 0.
            T(2,1) = A(L,LL)
            T(2,2) = A(L,L) + A(LL,LL)
            T(2,3) = T(1,2)
            T(3,1) = 0.
            T(3,2) = T(2,1)
            T(3,3) = A(LL,LL)
```

```
      P(1) = C(L,L)/2.
      P(2) = C(LL,L)
      P(3) = C(LL,LL)/2.
      NSYS = 3
      CALL SYSSLV
      C(L,L) = P(1)
      C(LL,L) = P(2)
      C(L,LL) = P(2)
      C(LL,LL) = P(3)
      GO TO 90
 80   T(1,1) = A(K,K) + A(L,L)
      T(1,2) = A(KK,K)
      T(1,3) = A(LL,L)
      T(1,4) = 0.
      T(2,1) = A(K,KK)
      T(2,2) = A(KK,KK) + A(L,L)
      T(2,3) = 0.
      T(2,4) = T(1,3)
      T(3,1) = A(L,LL)
      T(3,2) = 0.
      T(3,3) = A(K,K) + A(LL,LL)
      T(3,4) = T(1,2)
      T(4,1) = 0.
      T(4,2) = T(3,1)
      T(4,3) = T(2,1)
      T(4,4) = A(KK,KK) + A(LL,LL)
      P(1) = C(K,L)
      P(2) = C(KK,L)
      P(3) = C(K,LL)
      P(4) = C(KK,LL)
      NSYS = 4
      CALL SYSSLV
      C(K,L) = P(1)
      C(KK,L) = P(2)
      C(K,LL) = P(3)
      C(KK,LL) = P(4)
 90   K = K + DK
      IF(K .LE. N) GO TO 30
      LDL = L + DL
      IF(LDL .GT. N) RETURN
      DO 120 J=LDL,N
        DO 100 I=L,LL
          C(I,J) = C(J,I)
 100    CONTINUE
        DO 120 I=J,N
          DO 110 K=L,LL
            C(I,J) = C(I,J) - C(I,K)*A(K,J) - A(K,I)*C(K,J)
 110    CONTINUE
        C(J,I) = C(I,J)
 120  CONTINUE
      L = LDL
      GO TO 10
      END
```

```
      SUBROUTINE HSHLDR(A,N,NA)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(NA,1)
      REAL*8 MAX
C
      NM2 = N-2
      N1 = N+1
      IF(N .EQ. 1) RETURN
      IF(N .GT. 2) GO TO 5
      A(1,N1) = A(2,1)
      RETURN
    5 DO 80 L=1,NM2
        L1 = L+1
        MAX = 0.
        DO 10 I=L1,N
          MAX = DMAX1(MAX,DABS(A(I,L)))
   10   CONTINUE
        IF(MAX .NE. 0.) GO TO 20
        A(L,N1) = 0.
        A(N1,L) = 0.
        GO TO 80
   20   SUM = 0.
        DO 30 I=L1,N
          A(I,L) = A(I,L)/MAX
          SUM = SUM + A(I,L)**2
   30   CONTINUE
        S = DSIGN(DSQRT(SUM),A(L1,L))
        A(L,N1) = -MAX*S
        A(L1,L) = S + A(L1,L)
        A(N1,L) = S*A(L1,L)
        DO 50 J=L1,N
          SUM = 0.
          DO 40 I=L1,N
            SUM = SUM + A(I,L)*A(I,J)
   40     CONTINUE
          P = SUM/A(N1,L)
          DO 50 I=L1,N
            A(I,J) = A(I,J) - A(I,L)*P
   50   CONTINUE
        DO 70 I=1,N
          SUM = 0.
          DO 60 J=L1,N
            SUM = SUM + A(I,J)*A(J,L)
   60     CONTINUE
          P = SUM/A(N1,L)
          DO 70 J=L1,N
            A(I,J) = A(I,J) - P*A(J,L)
   70   CONTINUE
   80 CONTINUE
      A(N-1,N1) = A(N,N-1)
      RETURN
      END
```

```
      SUBROUTINE BCKMLT(A,U,N,NA,NU)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION
     1A(NA,1),U(NU,1)
C
      N1 = N+1
      NM1 = N-1
      NM2 = N-2
      U(N,N) = 1.
      IF(NM1 .EQ. 0) RETURN
      U(NM1,N) = 0.
      U(N,NM1) = 0.
      U(NM1,NM1) = 1.
      IF(NM2 .EQ. 0) RETURN
      DO 40 LL=1,NM2
        L = NM2-LL+1
        L1 = L+1
        IF(A(N1,L) .EQ. 0.) GO TO 25
        DO 20 J=L1,N
          SUM = 0.
          DO 10 I=L1,N
            SUM = SUM + A(I,L)*U(I,J)
   10     CONTINUE
          P = SUM/A(N1,L)
          DO 20 I=L1,N
            U(I,J) = U(I,J) - A(I,L)*P
   20   CONTINUE
   25   DO 30 I=L1,N
          U(I,L) = 0.
          U(L,I) = 0.
   30   CONTINUE
        U(L,L) = 1.
   40 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE SCHUR(H,U,NN,NH,NU,EPS,FAIL)
      IMPLICIT REAL*8 (A-H,O-Z)
       DIMENSION
     1H(NH,1),U(NU,1)
      INTEGER
     1 FAIL
      LOGICAL
     1LAST
      N = NN
      HN = 0.
      DO 20 I=1,N
        JL = MAX0(1,I-1)
        RSUM = 0.
      DO 10 J=JL,N
          RSUM = RSUM + DABS(H(I,J))
   10   CONTINUE
        HN = DMAX1(HN,RSUM)
   20 CONTINUE
      TEST = EPS*HN
      IF(HN .EQ. 0.) GO TO 230
   30 IF(N .LE. 1) GO TO 230
      ITS = 0
      NA = N-1
      NM2 = N-2
   40 DO 50 LL=2,N
      L = N-LL+2
        IF(DABS(H(L,L-1)) .LE. TEST) GO TO 60
   50 CONTINUE
      L = 1
      GO TO 70
   60 H(L,L-1) = 0.
   70 IF(L .LT. NA) GO TO 72
      N = L-1
      GO TO 30
   72 X = H(N,N)/HN
      Y = H(NA,NA)/HN
      R = (H(N,NA)/HN)*(H(NA,N)/HN)
      IF(ITS .LT. 30) GO TO 75
      FAIL = N
      RETURN
   75 IF(ITS.EQ.10 .OR. ITS.EQ.20) GO TO 80
      S = X + Y
      Y = X*Y - R
      GO TO 90
   80 Y = (DABS(H(N,NA)) + DABS(H(NA,NM2)))/HN
      S = 1.5*Y
      Y = Y**2
   90 ITS = ITS + 1
      DO 100 MM=L,NM2
        M = NM2-MM+L
        X = H(M,M)/HN
        R = H(M+1,M)/HN
        Z = H(M+1,M+1)/HN
        P = X*(X-S) + Y + R*(H(M,M+1)/HN)
        Q = R*(X+Z-S)
        R = R*(H(M+2,M+1)/HN)
        W = DABS(P) + DABS(Q) + DABS(R)
        P = P/W
        Q = Q/W
        R = R/W
        IF(M .EQ. L) GO TO 110
       IF(DABS(H(M,M-1))*(DABS(Q)+DABS(R)) .LE. DABS(P)*TEST)
      1GO TO 110
```

```
  100 CONTINUE
  110 M2 = M+2
      M3 = M+3
      DO 120 I=M2,N
         H(I,I-2) = 0.
  120 CONTINUE
      IF(M3 .GT. N) GO TO 140
      DO 130 I=M3,N
         H(I,I-3) = 0.
  130 CONTINUE
  140 DO 220 K=M,NA
         LAST = K.EQ.NA
         IF(K .EQ. M) GO TO 150
         P = H(K,K-1)
         Q = H(K+1,K-1)
         R = 0.
         IF(.NOT.LAST) R = H(K+2,K-1)
         X = DABS(P) + DABS(Q) + DABS(R)
         IF(X .EQ. 0.) GO TO 220
         P = P/X
         Q = Q/X
         R = R/X
  150    S = DSQRT(P**2 + Q**2 + R**2)
         IF(P .LT. 0.) S = -S
         IF(K .NE. M) H(K,K-1) = -S*X
         IF(K.EQ.M .AND. L.NE.M) H(K,K-1) = -H(K,K-1)
         P = P + S
         X = P/S
         Y = Q/S
         Z = R/S
         Q = Q/P
         R = R/P
         DO 170 J=K,NN
            P = H(K,J) + Q*H(K+1,J)
            IF(LAST) GO TO 160
            P = P + R*H(K+2,J)
            H(K+2,J) = H(K+2,J) - P*Z
  160       H(K+1,J) = H(K+1,J) - P*Y
            H(K,J) = H(K,J) - P*X
  170    CONTINUE
         J = MIN0(K+3,N)
         DO 190 I=1,J
            P = X*H(I,K) + Y*H(I,K+1)
            IF(LAST) GO TO 180
            P = P + Z*H(I,K+2)
            H(I,K+2) = H(I,K+2) - P*R
  180       H(I,K+1) = H(I,K+1) - P*Q
            H(I,K) = H(I,K) - P
  190    CONTINUE
         DO 210 I=1,NN
            P = X*U(I,K) + Y*U(I,K+1)
            IF(LAST) GO TO 200
            P = P + Z*U(I,K+2)
            U(I,K+2) = U(I,K+2) - P*R
  200       U(I,K+1) = U(I,K+1) - P*Q
            U(I,K) = U(I,K) - P
  210    CONTINUE
  220 CONTINUE
      GO TO 40
  230 FAIL = 0
      RETURN
      END
```

```
      SUBROUTINE SYSSLV
      IMPLICIT REAL*8 (A-H,O-Z)
C
      COMMON/SLVBLK/A(5,5),B(5),N
      REAL*8 MAX
    1 NM1 = N - 1
      N1 = N+1
C
C COMPUTE THE LU FACTORIZATION OF A.
      DO 80 K=1,N
        KM1 = K-1
        IF(K.EQ.1) GO TO 20
        DO 10 I=K,N
          DO 10 J=1,KM1
            A(I,K) = A(I,K) - A(I,J)*A(J,K)
   10   CONTINUE
   20   IF(K.EQ.N) GO TO 100
        KP1 = K+1
      MAX = DABS(A(K,K))
        INTR = K
        DO 30 I=KP1,N
          AA = DABS(A(I,K))
          IF(AA .LE. MAX) GO TO 30
          MAX = AA
          INTR = I
   30   CONTINUE
        IF(MAX .EQ. 0.) STOP
        A(N1,K) = INTR
        IF(INTR .EQ. K) GO TO 50
        DO 40 J=1,N
          TEMP = A(K,J)
          A(K,J) = A(INTR,J)
          A(INTR,J) = TEMP
   40   CONTINUE
   50   DO 80 J=KP1,N
          IF(K.EQ.1) GO TO 70
          DO 60 I=1,KM1
            A(K,J) = A(K,J) - A(K,I)*A(I,J)
   60     CONTINUE
   70     A(K,J) = A(K,J)/A(K,K)
   80 CONTINUE
C
C INTERCHANGE THE COMPONENTS OF B.
C
  100 DO 110 J=1,NM1
        INTR = A(N1,J)
        IF(INTR .EQ. J) GO TO 110
        TEMP = B(J)
        B(J) = B(INTR)
        B(INTR) = TEMP
  110 CONTINUE
C
C SOLVE LX = B.
C
  200 B(1) = B(1)/A(1,1)
      DO 220 I=2,N
        IM1 = I-1
        DO 210 J=1,IM1
          B(I) = B(I) - A(I,J)*B(J)
  210   CONTINUE
        B(I) = B(I)/A(I,I)
  220 CONTINUE
C
```

```
C SOLVE UX = B.
C
  300 DO 310 II=1,NM1
      I = NM1-II+1
      I1 = I+1
      DO 310 J=I1,N
         B(I) = B(I) - A(I,J)*B(J)
  310 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE GAUSEL (MAX, N, A, NR, B, IERR)
      IMPLICIT REAL*8 (A-H,O-Z)
C   FUNCTION               - COMPUTES SOLUTION TO A SET OF SIMULTANEOUS
C                               LINEAR EQUATIONS (DOES NOT GIVE PIVOT OR
C                               DETERMINANT DATA)
C   USAGE                  - CALL GAUSEL (MAX,N,A,NR,B,IERR)
C   PARAMETERS   MAX       - MAXIMUM ROW DIMENSION OF B
C                N         - ORDER OF A
C                A(N,N)    - INPUT MATRIX OF COEFFICIENTS (DESTROYED)
C                NR        - NUMBER OF COLUMNS IN B
C                B(MAX,NR) - MATRIX OF CONSTANTS (REPLACED BY SOLUTIONS)
C                IERR      - INTEGER ERROR CODE
C                               = 0   NORMAL RETURN
C                               = 2   INPUT MATRIX IS SINGULAR
C   REQUIRED ROUTINES      - NONE
C
C     SOURCE
C              NASA, LRC, ANALYSIS AND COMPUTATION DIVISION SUBPROGRAM
C              LIBRARY
C ****
      DIMENSION A(N,N),B(MAX,NR)
      NM1 = N-1
      IF (NM1 .EQ. 0) GO TO 140
C ****
C     FIND LARGEST REMAINING ELEMENT IN I-TH COLUMN FOR PIVOT
C ****
      DO 100 I=1,NM1
         BIG = 0.
         DO 20 K=I,N
            TERM = DABS(A(K,I))
            IF (TERM - BIG) 20,20,10
   10       BIG = TERM
            L = K
   20    CONTINUE
         IF (BIG) 40,30,40
   30    IERR = 2
         RETURN
   40    IF (I-L) 50,80,50
C ****
C     PIVOT ROWS OF A AND B
C ****
   50    CONTINUE
         DO 60 J=1,N
            TEMP = A(I,J)
            A(I,J) = A(L,J)
            A(L,J) = TEMP
   60    CONTINUE
         DO 70 J=1,NR
            TEMP = B(I,J)
            B(I,J) = B(L,J)
            B(L,J) = TEMP
   70    CONTINUE
   80    CONTINUE
C ****
C     STORE PIVOT AND PERFORM COLUMN OPERATIONS ON A AND B
C ****
         IP1 = I+1
         DO 100 II=IP1,N
            A(II,I) = A(II,I)/A(I,I)
            X3 = A(II,I)
            DO 90 K=IP1,N
               A(II,K) = A(II,K) - X3*A(I,K)
   90       CONTINUE
```

```
              DO 100 K=1,NR
                 B(II,K) = B(II,K) - X3*B(I,K)
  100   CONTINUE
C ****
C       PERFORM BACK SUBSTITUTION
C ****
        DO 110 IC=1,NR
           B(N,IC) = B(N,IC)/A(N,N)
  110   CONTINUE
        DO 130 KK=1,NM1
           I = N-KK
           IP1 = I+1
           DO 130 J=1,NR
              SUM = B(I,J)
              DO 120 K=IP1,N
                 SUM = SUM - A(I,K)*B(K,J)
  120         CONTINUE
              B(I,J) = SUM/A(I,I)
  130   CONTINUE
        RETURN
  140   CONTINUE
        IF (A(1,1) .EQ. 0.) GO TO 300
        DO 150 J=1,NR
           B(1,J) = B(1,J)/A(1,1)
  150   CONTINUE
        RETURN
  300   IERR = 2
        RETURN
        END
```

```
      SUBROUTINE PNCH (A,NA,NAM,IOP)
C IOP(1)=0, SKIP TITLE; IOP(2)=N, SKIP LINES; IOP(3)=1, TAB 25 SPACES.
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(1),IOP(4),NA(2)
      NR=NA(1)
      NC=NA(2)
      NMAX=NR*NC
      NSKIP=IOP(2)
      IF (IOP(2).EQ.0) GO TO 205
      DO 200 I=1,NSKIP
  200 WRITE(7,150)
  150 FORMAT(2X)
  205 CONTINUE
      IF (IOP(1).EQ.0) GO TO 210
      WRITE(7,151) NAM,NR,NC
  151 FORMAT(A4,/,2I5)
  210 CONTINUE
      DO 250 I=1,NR
      IF (IOP(3).EQ.0) WRITE(7,152) (A(J),J=I,NMAX,NR)
      IF (IOP(3).NE.0) WRITE(7,153) (A(J),J=I,NMAX,NR)
  250 CONTINUE
  152 FORMAT(6(1PD13.5))
  153 FORMAT(25X,6(1PD13.5))
      RETURN
      END
```

```
      FUNCTION DIMAG(Z)                                                 DIMC
      REAL*8 A(2),DIMAG                                                 DIMC
      COMPLEX*16 Z,B                                                    DIMC
      EQUIVALENCE (A,B)                                                 DIMC
      B=Z                                                               DIMC
      DIMAG=A(2)                                                        DIMC
      RETURN                                                            DIMC
      END                                                               DIMC
```

```
      FUNCTION DREAL(Z)                    DREOO
      REAL*8 A(2),DREAL                    DREOO
      COMPLEX*16 Z,B                       DREOO
      EQUIVALENCE (A,B)                    DREOO
      B=Z                                  DREOO
      DREAL=A(1)                           DREOO
      RETURN                               DREOO
      END                                  DREOO
```

```
      BLOCK DATA                                              MODO0
      IMPLICIT REAL*8 (A-H,O-Z)                               MODO0
      COMMON/LINES/TITLE(10),TIL(3),NLP,LIN                   MODO0
      COMMON/FORM/FMT1(2),FMT2(2),NEPR                        MODO0
      COMMON/TOL/EPSAM,EPSBM,IACM                             MODO0
      COMMON/CONV/SUMCV,RICTCV,SERCV,MAXSUM                   MODO0
      DATA LIN,NLP/1,58/                                      MODO0
      DATA NEPR,FMT1/7,8H(1P7D16.,8H7)          /             MODO0
      DATA TIL/8H     ORA,8HCLS  PRO,8HGRAM     /             MODO0
      DATA FMT2/8H(3X,1P7D,8H16.7)    /                       MODO0
      DATA EPSAM/1.E-10/                                      MODO0
      DATA EPSBM/1,E-10/                                      MODO0
      DATA IACM/12/
      DATA SUMCV/1,E-8/                                       MODO0
      DATA RICTCV/1,E-8/                                      MODO0
      DATA SERCV/1,E-8/                                       MODO0
      DATA MAXSUM/50/                                         MODO0
      END                                                     MODO0
```