NOTICE

THIS DOCUMENT HAS BEEN REPRODUCED FROM MICROFICHE. ALTHOUGH IT IS RECOGNIZED THAT CERTAIN PORTIONS ARE ILLEGIBLE, IT IS BEING RELEASED IN THE INTEREST OF MAKING AVAILABLE AS MUCH INFORMATION AS POSSIBLE.
THE DYNAMICS AND CONTROL OF LARGE FLEXIBLE SPACE STRUCTURES-III

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

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>I-1</td>
</tr>
<tr>
<td>I. INTRODUCTION</td>
<td>I-1</td>
</tr>
<tr>
<td>I.1 References - Introduction</td>
<td>I-6</td>
</tr>
<tr>
<td>II. CONTROL OF A LARGE FLEXIBLE PLATE IN ORBIT</td>
<td>II-1</td>
</tr>
<tr>
<td>II.1 Introduction - Chapter II</td>
<td>II-1</td>
</tr>
<tr>
<td>II.2 Model Development</td>
<td>II-1</td>
</tr>
<tr>
<td>II.2A Rotational Equations of Motion</td>
<td>II-1</td>
</tr>
<tr>
<td>II.2B Generic Mode Equations</td>
<td>II-2</td>
</tr>
<tr>
<td>II.2C Linearization</td>
<td>II-2</td>
</tr>
<tr>
<td>II.2D Modelling of Point Actuators</td>
<td>II-2</td>
</tr>
<tr>
<td>II.2E Modelling of Distributed Actuators</td>
<td>II-2</td>
</tr>
<tr>
<td>II.3 Uncontrolled Motion-Numerical Example</td>
<td>II-3</td>
</tr>
<tr>
<td>II.4 Controlled Motion</td>
<td>II-4</td>
</tr>
<tr>
<td>II.4A Decoupling of Original State Equations Using State Variable Feedback</td>
<td>II-4</td>
</tr>
<tr>
<td>II.4B Stabilizing the System by Pole Clustering</td>
<td>II-5</td>
</tr>
<tr>
<td>II.4C Application of the Linear Regulator Theory</td>
<td>II-6</td>
</tr>
<tr>
<td>II.5 Conclusions</td>
<td>II-7</td>
</tr>
<tr>
<td>II.6 References - Chapter II</td>
<td>II-7</td>
</tr>
<tr>
<td>Appendix A - Chapter II</td>
<td>II-8</td>
</tr>
<tr>
<td>III. FREQUENCIES AND MODE SHAPES FOR RECTANGULAR PLATES</td>
<td>III-1</td>
</tr>
<tr>
<td>III.1 Formulation by Warburton</td>
<td>III-1</td>
</tr>
<tr>
<td>III.2 Formulation by Lemke</td>
<td>III-2</td>
</tr>
</tbody>
</table>
III.3  Finite Element Computer Program            III-5
III.4  Discussion of Numerical Results            III-5
III.5  References - Chapter 3                      III-9

IV.   GENERAL CONCLUSIONS AND RECOMMENDATIONS    IV-1

Appendix
List of Tables by Chapters

<table>
<thead>
<tr>
<th>Table No.</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chapter I</td>
<td>Table 1</td>
<td>I-3</td>
</tr>
<tr>
<td></td>
<td>Study Plan 1977-1980</td>
<td></td>
</tr>
<tr>
<td>Chapter II</td>
<td>Table 1</td>
<td>II-6</td>
</tr>
<tr>
<td></td>
<td>Maximum Force Amplitudes (Newtons) for Different Combinations of Cases with Actuator Locations</td>
<td></td>
</tr>
<tr>
<td>Chapter III</td>
<td>Table 1</td>
<td>III-10</td>
</tr>
<tr>
<td></td>
<td>Evaluation of Parameters in Frequency Expression (Warburton)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Table 2</td>
<td>III-11</td>
</tr>
<tr>
<td></td>
<td>Frequencies and Nodal Patterns Obtained by the Three Methods</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Table 3</td>
<td>III-12</td>
</tr>
<tr>
<td></td>
<td>Normalized Deflections at the Nodal Points Obtained by the Three Methods-Second Mode Cases 1-3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Table 4</td>
<td>III-13</td>
</tr>
<tr>
<td></td>
<td>Normalized Deflections at the Nodal Points Obtained by the Three Methods-Second Mode Case 4</td>
<td></td>
</tr>
<tr>
<td>Fig No.</td>
<td>Caption</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Chapter II</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fig. 1a</td>
<td>Platform following local vertical with major surface normal to the orbit plane - Case (i)</td>
<td>II-9</td>
</tr>
<tr>
<td>Fig. 1b</td>
<td>Platform along local horizontal - Case (ii)</td>
<td>II-9</td>
</tr>
<tr>
<td>Fig. 1c</td>
<td>Platform following local vertical with major surface in the orbit plane - Case (iii)</td>
<td>II-9</td>
</tr>
<tr>
<td>Fig. 2</td>
<td>Roll/yaw motion (uncontrolled) - Case (i)</td>
<td>II-9</td>
</tr>
<tr>
<td>Fig. 3</td>
<td>Location of two sets of actuators (I &amp; II)</td>
<td>II-10</td>
</tr>
<tr>
<td>Fig. 4a</td>
<td>Controlled state response for all combinations of orientations and actuator locations</td>
<td>II-11</td>
</tr>
<tr>
<td>Fig. 4b</td>
<td>Control force time history for Case (iii) - II</td>
<td>II-12</td>
</tr>
<tr>
<td>Fig. 5</td>
<td>Variation of least damped mode negative real part with $\alpha$ and $R$</td>
<td>II-13</td>
</tr>
<tr>
<td>Fig. 6</td>
<td>Maximum force amplitude as a function of $\alpha$ and $R$ for all actuators - (application of linear regulator theory)</td>
<td>II-14</td>
</tr>
<tr>
<td>Chapter III (Figures demonstrate nodal patterns)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fig. 1</td>
<td>First mode (1,1)</td>
<td>III-14</td>
</tr>
<tr>
<td>Fig. 2</td>
<td>Second mode (2,0)-(0,2)</td>
<td>III-14</td>
</tr>
<tr>
<td>Fig. 3a</td>
<td>(2,0) mode</td>
<td>III-14</td>
</tr>
<tr>
<td>Fig. 3b</td>
<td>(0,2) mode</td>
<td>III-14</td>
</tr>
<tr>
<td>Fig. 4</td>
<td>Third mode (2,0) + (0,2)</td>
<td>III-15</td>
</tr>
<tr>
<td>Fig. 5</td>
<td>Fourth mode (2,1)</td>
<td>III-15</td>
</tr>
<tr>
<td>Fig. 6</td>
<td>Sixth mode (3,0)</td>
<td>III-15</td>
</tr>
<tr>
<td>Fig. 7</td>
<td>Eighth (and Seventh) mode (2,2)</td>
<td>III-15</td>
</tr>
<tr>
<td>Fig. 8</td>
<td>Ninth mode (1,3) - (3,1)</td>
<td>III-16</td>
</tr>
<tr>
<td>Fig. 9</td>
<td>Tenth mode (1,3) + (3,1)</td>
<td>III-16</td>
</tr>
</tbody>
</table>
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 after each chapter.
Towards the end of the second grant year the three dimensional model of a free-free plate in orbit was developed and a limited number of computer simulations of the uncontrolled dynamics in response to initial perturbations about a specific equilibrium orientation were performed. Frequency values associated with the basic structural modes of a square plate were obtained from energy considerations based on approximate expressions developed by Warburton. 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, 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:


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. $\phi^{(n)}_x = 0$
   ✓ (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

EP 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 \( N \times P \)

✓ (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


II. CONTROL OF A LARGE FLEXIBLE PLATE IN ORBIT

ORIGINAL PAGE IS OF POOR QUALITY

Abstract

The dynamics and attitude and shape control of a large thin flexible platform in orbit are studied. Attitude and shape control is assumed to result from actuators placed perpendicular to the main surface and one edge and their effect on the rigid body and elastic modes is modeled 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 transmission to earth-based receiving stations. 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 using a slightly modified version of the general formulation of the dynamics of a general flexible orbiting body formulated by Santini. This specific example considered only the in-plane 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 modeling of point actuators located at specific points along the beam with the associated criteria for controllability and also the problem of selecting control law feedback gains based on decoupling techniques and application of the linear regulator problem. 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.

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. Modal 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:

\( \begin{align*}
\dot{\theta}_x &= \frac{I_x}{I_z} \dot{\gamma}_x - \frac{G_{x}}{I_z} \theta^2 + \frac{c}{I_z} \theta + \frac{J_y}{I_z} \dot{\gamma}_z

\dot{\gamma}_x &= \frac{I_y}{I_z} \dot{\theta}_x - \frac{G_{y}}{I_z} \dot{\gamma}^2 + \frac{c}{I_z} \gamma + \frac{J_x}{I_z} \dot{\theta}_z

\dot{\theta}_z &= \frac{I_z}{I_x} \dot{\gamma}_z - \frac{G_{z}}{I_x} \dot{\theta}^2 + \frac{c}{I_x} \theta + \frac{J_x}{I_x} \dot{\gamma}_x
\end{align*} \) (1)

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

\( \begin{align*}
\dot{\theta}_x &= \dot{\psi} + (\dot{\theta}_x \theta) \sin \phi

\dot{\gamma}_x &= (\dot{\theta}_x \theta) \cos \phi \cos \phi + \dot{\phi} \sin \phi

\dot{\phi}_x &= \dot{\phi} \cos \phi - (\dot{\theta}_x \theta) \sin \phi \cos \phi
\end{align*} \) (2)

(Note: Symbols used are defined in Appendix A.)
\[ C_{x_0}, C_{y_0}, C_{z_0} \] represent the gravity-gradient torques about the principal undeformed body axes and can be evaluated as:

\[ C_{x_0} = 3 \omega_z^2 (r_1^2 - r_2^2) \cos \phi \cos \theta \]

\[ C_{y_0} = 3 \omega_x^2 (r_1^2 - r_2^2) \cos \phi \cos \theta \]

\[ C_{z_0} = 3 \omega_y^2 (r_1^2 - r_2^2) \cos \phi \cos \theta \]

(3)

where \( \phi = \sin(\phi) \) and \( \cos(\phi) \).

\section*{5. Generic Mode Equations}

For each of the three nominal orientations considered in terms of the modal amplitude \( A_p \):\n
\[ \bar{A}_x = \left[ (u_{i}^2 + u_{j}^2) - H_{xx} \right] A_x = \frac{E_x}{K_x} \]

where

\[ H_{xx} = u_c^2 \left[ 3(c^2 + e^2 + e^2 - 3c^2 + 3e^2) \right] \]

(4a)

\[ \bar{A}_y = \left[ (u_{j}^2 + u_{i}^2) - H_{yy} \right] A_x = \frac{E_y}{K_y} \]

where

\[ H_{yy} = u_c^2 \left[ 3(c^2 + e^2 + e^2 - 3c^2 + 3e^2) \right] \]

(4b)

\[ \bar{A}_z = \left[ (u_{k}^2 + u_{l}^2) - H_{zz} \right] A_x = \frac{E_z}{K_z} \]

where

\[ H_{zz} = u_c^2 \left[ 3(c^2 + e^2 + e^2 - 3c^2 + 3e^2) \right] \]

(4c)

\section*{6. Linearisation}

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

\[ \ddot{\phi} = \omega_c \left[ \frac{I_y - I_z}{I_x} \right] = \omega_c \left[ \frac{I_y - I_z}{I_x} \right] \phi + \frac{T_x}{I_x} \frac{C_y}{I_y} \]

\[ \ddot{\theta} = \omega_c \left[ \frac{I_z - I_x}{I_y} \right] = \omega_c \left[ \frac{I_z - I_x}{I_y} \right] \theta + \frac{T_y}{I_y} \frac{C_z}{I_z} \]

\[ \ddot{\psi} = 3 \omega_c^2 \left[ \frac{I_y - I_z}{I_x} \right] \theta + \frac{T_x}{I_x} \frac{C_y}{I_y} \]

(5)

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

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

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

Case (iii): \( I_x = I_y = I_z = 2I_x = 2I_y = 2I_z \)

(6)

For small amplitude angles the generic mode equations become:

Case (i): \( \bar{A}_x + (u_{i}^2 + u_{j}^2)A_x = \frac{E_x}{K_x} \)

Case (ii): \( \bar{A}_y + (u_{j}^2 + u_{i}^2)A_x = \frac{E_y}{K_y} \)

Case (iii): \( \bar{A}_z + (u_{l}^2 + u_{k}^2)A_x = \frac{E_z}{K_z} \)

(7)

\section*{D. Modelling of Point Actuators}

For an actuator which can generate a force of the type

\[ F = f_x \dot{x} + f_y \dot{y} + f_z \dot{z} \]

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

\[ T = \frac{E_x}{K_x} f_x \]

\[ T = \frac{E_y}{K_y} f_y \]

\[ T = \frac{E_z}{K_z} f_z \]

(8)

\section*{E. Modelling of Distributed Actuators}

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

\[ F = f_x(x,y,z,t) \]

\[ + f_y(x,y,z,t) \]

\[ + f_z(x,y,z,t) \]

(15)

where the force components are now both spatially and time dependent.
The torque due to such an actuator is given by
\[
\tau = R\tau_f
\]  
(16)
The total torque is given by
\[
\tau = \int (R\tau_f) \, dx \, dy \, dz
\]  
(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)
\[
f_x(x,y,z,t) = \sum_{n=1}^{N} f_{x_n}(x,y,z)g_{x_n}(t)
\]  
(18)
The integral for the torque is then given by
\[
\tau = \int \left[ \sum_{n=1}^{N} f_{x_n}(x,y,z)g_{x_n}(t) \right] \, dx \, dy \, dz
\]  
(19)
The resulting generic force is then obtained in the same manner as in Eq. (13) with the result,
\[
\tau = \int \left[ \sum_{n=1}^{N} f_{x_n}(x,y,z)g_{x_n}(t) \right] \, dx \, dy \, dz
\]  
(20)
3. Uncontrolled Motion—Numerical Example
The platform is assumed to have the following physical properties:
\[ a = 100 \text{ m (side of square plate)} \]
\[ N = 276800 \text{ kg} \]
Minimum Moment of Inertia = 2.354x10^7 kg-m^2
Maximum Moment of Inertia = 4.708x10^7 kg-m^2
For an assumed orbital altitude of 250 n.mi. (circular)
\[ \omega_c = 1.25x10^{-3} \text{ rad/sec} \]
The modal frequencies of the elastic modes have been obtained using a finite element computer algorithm.7 For the first three flexible modes:
\[ \omega_1 = 2.0931947x10^{-2} \text{ rad/sec} \]
\[ \omega_2 = 3.0404741x10^{-2} \text{ rad/sec} \]
\[ \omega_3 = 3.9088122x10^{-2} \text{ rad/sec} \]
The uncontrolled motion of the linear system through small amplitude deviations with respect to each of the three nominal orientations will now be considered.
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 (\( t = \omega_c \cdot t, \dot{z} = \omega_c \cdot \dot{z}, \dot{\theta} = \omega_c \cdot \dot{\theta}, \text{etc} \))
\[ \phi'' = \frac{[I_x-I_y-I_z]}{I_x} \cdot \omega_c^2 \cdot \sin^2 \left( \frac{(1-I_y)/I_x} \right) \]  
(22)
\[ \phi'' = 38 \]  
(23)
The generic mode equations become:
\[ \ddot{z} = -(\omega_c^2 z_y)^2 \frac{z_y}{z} \]  
(24)
The pitch and the generic mode equations are decoupled from roll and yaw. The pitch and generic modes exhibit simple harmonic motion. After substituting inertia values into the roll and yaw equations,
\[ \phi'' = -(2/\omega_c)\phi'' \]  
(25)
\[ \phi'' = \phi'/\omega_c \]  
(26)
The characteristic equation for the system (25) and (26) is, \( s^2(s^2-4/\omega_c^2) = 0 \)
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
\[ \phi'' = -\frac{1}{\omega_c} \phi'' \]  
(27)
\[ \phi'' = \frac{2}{\omega_c} \phi'' \]  
(28)
\[ \phi'' = 38 \]  
(29)
The generic mode equations can be represented by,
\[ \dot{z}^2 = -\frac{(\omega_c^2 \cdot \omega_c^2 z_y)}{z} \frac{z_y}{z} \]  
(30)
From Eq. (29) the pitch amplitude increases exponentially in response to an initial displacement, whereas from Eq. (30), for \( \omega_c^2 \cdot \omega_c \) the generic modal amplitudes exhibit simple harmonic motion.
The characteristic equation for the combined roll/yaw motion is:
\[ z^2(s^2-4/\omega_c^2) = 0 \]  
(31)
The roll/yaw motion is characterized by a double pole at the origin and is thus unstable.
Case (iii): \( I_x = I_y \), and \( I_z = 2I_x = 2I_y \)
The rotational equations of motion are
\[ \phi'' = -\phi; \phi'' = -4\phi; \phi'' = 0 \]  
(32)
while the generic mode equations can be expressed by,
\[ \dot{z}^2 = -\frac{(\omega_c^2 \cdot \omega_c^2 z)}{z} \frac{z}{z} \]  
(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:

\[ X' = AX + BU \]  

(34)

where the state vector, \( X \), is defined as

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

and

\[ x_1 = \dot{\psi}; \quad x_2 = \dot{\theta}; \quad x_3 = \dot{\phi}; \quad x_{4i} = \dot{q}_i; \quad i = 1, 2, \ldots, n \text{ generic modes} \]

\[ x_{4i} = \dot{q}_i; \quad x_{5i} = \dot{\dot{q}}; \quad x_{6i} = \phi_i \]

\[ x_{6i+6} = \frac{1}{2} x_{6i+6}^2; \quad i = 1, 2, \ldots, n \]

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

\[
\begin{bmatrix}
  A_{6x6} & 0_{6x6} \\
  \text{diagonal} & \text{A}_{6x6}
\end{bmatrix}
\]

(35)

The non-zero and non-unity elements appearing in \( A \) are:

\[ A_{7,1} = \frac{4(1-\gamma)I_x}{I_x}; \quad A_{8,2} = -\frac{(1-\gamma)I_x}{I_x}; \]

\[ A_{9,3} = 3\frac{(1-\gamma)I_y}{I_y}; \quad A_{10,4} = -\frac{(1-\gamma)I_y}{I_y}; \]

\[ A_{11,5} = (1-\gamma)^2; \quad A_{12,6} = (1-\gamma)^2; \]

\[ A_{8,7} = \frac{1}{2}(1-\gamma)I_x; \quad A_{9,8} = (1-\gamma)^2I_x \]

The general \( B \) matrix:

\[ B = \begin{bmatrix}
  0_{6x6} \\
  0_{6x6}
\end{bmatrix} \]

(36)

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 \( \mathbf{U} = -\mathbf{K}\mathbf{X} \); (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

\[ \mathbf{X}' = \mathbf{A}\mathbf{X} + \mathbf{B}\mathbf{U} \]  

(37)

where \( \mathbf{X} = (x_1, x_2, \ldots, x_n) \)

After selecting \( \mathbf{U} = \mathbf{K}\mathbf{X} \) we can rewrite the controlled motion equations as

\[ \mathbf{X}' = (\mathbf{A} + \mathbf{B}\mathbf{K})\mathbf{X} + \mathbf{B}\mathbf{U} \]

(38)

\[ \mathbf{K} \]

and \( \mathbf{K} \) are evaluated such that \( \mathbf{A} + \mathbf{B}\mathbf{K} \) 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, (1)–(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: \( a = 100a \)

\[ f_1(\gamma/6, \gamma/6, 0); \quad f_2(\gamma/6, \gamma/6, 0); \quad f_3(\gamma/6, \gamma/6, 0) \]

Second Location: \( a = 100a \)

\[ f_1(\gamma/2, \gamma/2, 0); \quad f_2(\gamma/2, \gamma/2, 0); \quad f_3(\gamma/2, \gamma/2, 0) \]

Case (ii)

First Location: \( a = 100a \)

\[ f_1(0, \gamma/6, \gamma/6); \quad f_2(0, \gamma/6, \gamma/6); \quad f_3(0, \gamma/6, \gamma/6) \]

Second Location: \( a = 100a \)

\[ f_1(0, \gamma/2, \gamma/2); \quad f_2(0, \gamma/2, \gamma/2); \quad f_3(0, \gamma/2, \gamma/2) \]

Case (iii)

First Location: \( a = 100a \)

\[ f_1(0, \gamma/6, \gamma/6); \quad f_2(0, \gamma/6, \gamma/6); \quad f_3(0, \gamma/6, \gamma/6) \]

Second Location: \( a = 100a \)

\[ f_1(0, \gamma/2, \gamma/2); \quad f_2(0, \gamma/2, \gamma/2); \quad f_3(0, \gamma/2, \gamma/2) \]

Actuator positions for the two different sets of locations are illustrated in Fig. 3. The system \( A \) and \( B \) matrices corresponding to different combinations of the three platform orientations and the two sets of actuator location are listed as follows.
The equations of motion of the platform when recast in state space format can be written as

\[ \dot{X} = AX + BU \]
The control, $U = -K$, is selected by using a digital computer algorithm such that (A-K) 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: $f_1(\alpha/6, -\alpha/6, 0)$; $f_2(\alpha/6, -\alpha/6, 0)$; $f_3(-\alpha/6, 0, 0)$; and $f_4(\alpha/2, \alpha/6, 0)$ for $\alpha = 100^\circ$. It is assumed that the slalom damping requirement on the system has a time constant of $(13.33 \text{ min or } 1/20 \text{ dimensional orbital time})$. The control influence matrix is then calculated based on the assumed coordinates of the four actuators. The control $U = -K$ can be calculated by the ORACLS pole clustering algorithm. Based on these gains time histories of the required control forces are then obtained.

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

<table>
<thead>
<tr>
<th>Force</th>
<th>Location I (interior)</th>
<th>Location II (exterior)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Case i</td>
<td>Case ii</td>
</tr>
<tr>
<td>$f_1$</td>
<td>-807.8</td>
<td>270.63</td>
</tr>
<tr>
<td>$f_2$</td>
<td>270.0</td>
<td>136.2</td>
</tr>
<tr>
<td>$f_3$</td>
<td>387.3</td>
<td>-425.0</td>
</tr>
<tr>
<td>$f_4$</td>
<td>527.9</td>
<td>195.6</td>
</tr>
<tr>
<td>$f_5$</td>
<td>489.09</td>
<td>-253.3</td>
</tr>
<tr>
<td>$f_6$</td>
<td>-264.38</td>
<td>-272.0</td>
</tr>
</tbody>
</table>

An interesting comparison can be made between this result and that shown in Table 1 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 $\lambda_1$ with $\lambda_7 = 11.24$). Depending on the nature of the expected disturbance forces this result could be very undesirable.

### Application of the Linear Regulator Theory

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

$$ J = \int \left( X^T X + u^T R u \right) dt $$

where $Q$ and $R$ are positive definite penalty matrices. The steady state solution of the Ricatti 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 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 (I), and control about the first nominal orientation (1) is considered.

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

$$ \dot{\omega}_x^2 = \omega_x^2 - a \omega_y^2 $$

where the system oscillates at the frequency $\omega = \sqrt{a}$. It is desired that the control remove a maximum "transverse" angular rate, $\omega_x = \max\{\omega_x, \omega_y\}$. The maximum rate occurs when $\omega_x = \omega_x(0)$ and $\omega_y = \omega_y(0)$.

$$ \omega^2 \left[ \begin{array}{c} \omega_x \\ -a \omega_y \\ \omega_y \end{array} \right] = \left[ \begin{array}{c} 0 \\ a \\ -b \end{array} \right] \left[ \begin{array}{c} \omega_x \\ \omega_y \\ \omega_y \end{array} \right] $$

where

$$ \omega_x^2 = \omega_x(0)^2 + \omega_y(0)^2 $$
so that a strategy for selecting the elements of Q could be 9.10

\[
Q = \begin{bmatrix} f & 0 \\ 0 & f \end{bmatrix}
\]

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

\[
\begin{bmatrix} \dot{x}_1' \\ \dot{x}_2' \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -(\omega_1/\omega_c)^2 & 0 \end{bmatrix} \begin{bmatrix} x_1' \\ x_2' \end{bmatrix}
\]

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) with actuator locations as given in Section (b)) is obtained using the relations given by Eq. (41) and is a diagonal matrix, \(Q_0\), with the following elements:

\[Q_{1,1} = 4.32 \times 10^9, \quad Q_{2,2} = 8.399 \times 10^9,\]

\[Q_{3,3} = Q_{4,4} = 3.0 \times 10^9, \quad Q_{5,5} = Q_{11,11} = 5.886 \times 10^6,\]

\[Q_{6,6} = Q_{12,12} = 9.748 \times 10^6, \quad Q_{7,7} = Q_{8,8} = 2.222 \times 10^3.\]

The \(R_0\) matrix is chosen as an identity matrix. A parametric study is done using various multiples of the \(Q_0\) (Q=Q0) and \(R_0\) matrices obtained above which are plotted against the negative real part of the least damped mode of the controlled system in Fig. 5. All the loci of the negative real part of the least damped mode approach unity and no significant improvement is observed by increasing the state penalty, \(Q=Q_0\), any further. Thus one wishes to operate on the horizontal line between the points (1) and (2). The maximum amplitudes 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 (non-dimensional):

\[-1.0043, -1.84 \times 10^{-10}, -2.16 \times 10^{-20}, -1.17, -1.70, -1.19, -1.76, -2.36, -1.37, -1.64, -3.66, -1.132, -1.11\]

The maximum force amplitudes as shown in Fig. 6 are less than those corresponding to Case (1) - Location I of Table 1 for comparable transient responses, whereas these are high, as compared to the forces obtained using the pole clustering technique (Section (b)). This is due to the large negative real parts of the other modes in the linear regulator case when compared to the pole clustering technique where all the poles have an equal negative real part (-1.0). Both the linear regulator and pole clustering technique have the 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 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 (ORACLST) 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 performances 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


Appendix A - Nomenclature

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$m_r$</td>
<td>$r$th modal mass</td>
</tr>
<tr>
<td>$T$</td>
<td>Torque due to an actuator</td>
</tr>
<tr>
<td>$T_x, T_y, T_z$</td>
<td>Torque components</td>
</tr>
<tr>
<td>$w_r(x, y)$</td>
<td>$r$th modal shape function</td>
</tr>
<tr>
<td>$z_r$</td>
<td>Nondimensionalized $r$th modal amplitude function</td>
</tr>
<tr>
<td>$\omega_c$</td>
<td>Orbital frequency</td>
</tr>
<tr>
<td>$\omega_1, \omega_2, \omega_3$</td>
<td>Angular body rates</td>
</tr>
<tr>
<td>$\dot{\phi}, \dot{\psi}, \dot{\theta}$</td>
<td>Roll, yaw, pitch, respectively</td>
</tr>
<tr>
<td>$\omega_1, \omega_2, \omega_3$</td>
<td>First three modal frequencies of the plate</td>
</tr>
</tbody>
</table>
Fig. 1a. Platform following local vertical with major surface normal to the orbit plane - Case (i)

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

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

Fig. 2. Roll/yaw motion (uncontrolled) - Case (i)
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 $a$ and $R$. 

$Q = aQ_0$
Fig 6. Maximum force amplitudes as a function of $a$ 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 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 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 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. 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

\[
\frac{\partial^4 W}{\partial x^4} + 2 \frac{\partial^4 W}{\partial x^2 \partial y^2} + \frac{\partial^4 W}{\partial y^4} + \frac{12\rho(1-\nu^2)}{Eh^2} \frac{\partial^2 W}{\partial t^2} = 0 \quad \text{(III-1)}
\]
where \( \rho, \sigma \) and \( E \) are the density, Poisson's ratio and Young's modulus of the plate material, respectively, \( h \) is the plate thickness, and \( g \) is the acceleration due to gravity. The displacement, \( w \), at any point \((x,y)\) at time \( t \) is given by

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

\( \theta(x) \) 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

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

and

\[
\lambda^2 = G_x^4 + G_y^4 \frac{a^4}{b^4} + 2 \frac{a^2}{b^2} \left[ \sigma \gamma_{x'y'} + (1-\sigma) J_{x'y'} \right] \tag{III-4}
\]

where \( \lambda \) is a non-dimensional frequency factor, \( a \) and \( b \) are the length and width of the plate, and \( G_x, G_y, J_x, J_y, \gamma_{x'y'} \), and \( \gamma \) 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

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

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

III-2
Modes \((m,o) + (o,m)\), \(m\) is even
\[
\lambda^2 = (m-\frac{1}{2})^4 + 2 \sigma (m-\frac{1}{2})^2 \frac{8}{\pi^2} \tag{III-6}
\]

Modes \((m,l) \pm (l,m)\), for \(m = 3, 5, 7, \ldots \)
\[
\lambda^2 = (m-\frac{1}{2})^4 + 2(1-\sigma) (m-\frac{1}{2})^2 \left[ 1 + \frac{6}{(m-\frac{1}{2})^2} \right] \frac{12}{\pi^2} \]
\[+ 2 \sigma (m-\frac{1}{2})^2 \frac{24}{\pi^2} \left[ 1 - \frac{2}{(m-\frac{1}{2})^2} \right]^2 + 2(1-\sigma) \frac{192}{\pi^4} \]

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, \(0(x)\cdot\phi(y)\), 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,

\[ W = \sum A_{m,n} \theta_m(x) \phi_n(y) \tag{III-7} \]

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

\[ \theta_m(x) = \frac{\cosh k_m \cos k_m x + \cos k_m \cosh k_m \overline{x}}{\sqrt{\cosh^2 k_m + \cos^2 k_m}} \quad (m \text{ even}) \tag{III-8} \]

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

\(\phi_n(y)\) 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

\[ \tan k_m + \tanh k_m = 0 \quad (m \text{ even}) \]
\[ \tan k_m - \tanh k_m = 0 \quad (m \text{ odd}) \]

which result from the spatial boundary conditions. Further, it was shown by an energy principle that \(^2,^4\)

\[ \omega^2 = \frac{U_{\text{max}}}{\frac{ph}{2g} \int_a^b \int_0^0 W^2 \, dx \, dy} \quad (\text{III-9}) \]

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

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

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

- Density = 2768.0 kg/m$^3$
- Young’s modulus = 0.7441x10$^{10}$ 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. The first three frequencies and mode shapes obtained agree with the frequencies and mode shapes computed from Warburton's formulas (Table 1). However, the next three frequencies obtained by Lemke's method correspond to higher frequencies and mode shapes obtained by Warburton's method. Also the nodal patterns obtained by Lemke's method compare approximately with the nodal patterns obtained by Warburton's method although the frequencies do not correspond in all cases. The results obtained by Lemke do not show the four intermediate frequencies corresponding to the fourth, fifth, sixth and seventh modes obtained by Warburton's method. The frequencies and nodal pattern obtained by Lemke's method are shown in Table 2.

For implementation of the computer program STRUDL, first the plate is divided into four elements. The first six modes (in terms of increasing frequencies) as predicted by STRUDL are also apparent from Warburton's results. The plate is assumed to be divided into 9, 16, 36 and 64 elements, respectively. The results of STRUDL are tabulated in Table 2. It can be seen that STRUDL frequencies approach the frequencies obtained by Warburton's method as the number of plate elements is increased.
However, in the cases of 36 and 64 elements some of the frequencies show a tendency to oscillate about an average value. This probably is due to the computational round off errors which begin to dominate with the increasing computations associated with larger number of elements. Thus, the advantage of taking a large number of elements may not be fully realized due to numerical accuracy limitations. Computation with more elements requires more computation time and a larger computer memory. For the 64 elements case, it was not possible to obtain the mode shapes due to memory limitations. It was also observed that the convergence of the frequencies, with an increase in the number of elements, is faster than the convergence of the mode shapes. It can be seen from Table 2, that the numerical results of STRUDL using 36 elements correlate with the results of Warburton both in frequency and mode shapes.

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

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


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

<table>
<thead>
<tr>
<th>m</th>
<th>( G_x )</th>
<th>( H_x )</th>
<th>( J_x )</th>
<th>( n )</th>
<th>( G_y )</th>
<th>( H_y )</th>
<th>( J_y )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>( \frac{12}{\pi^2} )</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>( \frac{12}{\pi^2} )</td>
</tr>
<tr>
<td>2</td>
<td>1.506</td>
<td>1.248</td>
<td>5.017</td>
<td>2</td>
<td>1.506</td>
<td>1.248</td>
<td>5.017</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>5</td>
<td>( m - \frac{1}{2} )</td>
<td>( \left(m - \frac{1}{2}\right)^2 P )</td>
<td>( \left(m - \frac{1}{2}\right)^2 Q )</td>
<td>( \left(n - \frac{1}{2}\right) )</td>
<td>( \left(n - \frac{1}{2}\right)^2 P )</td>
</tr>
</tbody>
</table>

\[
P = \left[1 - \frac{2}{(m-0.5)\pi}\right] \quad Q = \left[1 + \frac{6}{(m-0.5)\pi}\right]
\]
<table>
<thead>
<tr>
<th>STRUDEL</th>
<th>WARBURTON</th>
<th>LIMKE</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUMBER OF ELEMENTS</td>
<td>Nodal Pattern</td>
<td>Freq. (cps)</td>
</tr>
<tr>
<td>4</td>
<td>9</td>
<td>16</td>
</tr>
<tr>
<td>2</td>
<td>.004291</td>
<td>.004664</td>
</tr>
<tr>
<td>3</td>
<td>.005549</td>
<td>.006133</td>
</tr>
<tr>
<td>4</td>
<td>.007619</td>
<td>.008453</td>
</tr>
<tr>
<td>5</td>
<td>.007619</td>
<td>.008453</td>
</tr>
<tr>
<td>6</td>
<td>.01323</td>
<td>.01512</td>
</tr>
<tr>
<td>7</td>
<td>.01512</td>
<td>.01600</td>
</tr>
<tr>
<td>8</td>
<td>.01512</td>
<td>.01600</td>
</tr>
<tr>
<td>9</td>
<td>.01630</td>
<td>.01737</td>
</tr>
<tr>
<td>10</td>
<td>.01843</td>
<td>.02022</td>
</tr>
<tr>
<td>11</td>
<td>.02465</td>
<td>.02693</td>
</tr>
<tr>
<td>12</td>
<td>.02465</td>
<td>.02693</td>
</tr>
<tr>
<td>13</td>
<td>.03362</td>
<td>.03229</td>
</tr>
</tbody>
</table>

TABLE-III-2. Frequencies and Nodal Patterns Obtained by the Three Methods.
### Case 1: 4 Elements

<table>
<thead>
<tr>
<th>LOCATION</th>
<th>STROUD</th>
<th>LEMKE</th>
<th>WARBURTON</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>B1</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>B2</td>
<td>-1.0</td>
<td>-1.0</td>
<td>-1.0</td>
</tr>
<tr>
<td>A2</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

The grid used for 36 elements.

### Case 2: 9 Elements

<table>
<thead>
<tr>
<th>LOCATION</th>
<th>STROUD</th>
<th>LEMKE</th>
<th>WARBURTON</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>A2</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>A3</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>B1</td>
<td>-1.0</td>
<td>-1.0</td>
<td>-1.0</td>
</tr>
<tr>
<td>B2</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>B3</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>C1</td>
<td>-1.0</td>
<td>-1.0</td>
<td>-1.0</td>
</tr>
<tr>
<td>C2</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>C3</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

### Case 3: 16 Elements

<table>
<thead>
<tr>
<th>LOCATION</th>
<th>STROUD</th>
<th>LEMKE</th>
<th>WARBURTON</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>A2</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>A3</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>B1</td>
<td>-1.0</td>
<td>-1.0</td>
<td>-1.0</td>
</tr>
<tr>
<td>B2</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>B3</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>C1</td>
<td>-1.0</td>
<td>-1.0</td>
<td>-1.0</td>
</tr>
<tr>
<td>C2</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>C3</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

### TABLE-III-3. Normalized Deflections at the Nodal Points Obtained by the Three Methods - Second Mode.
### Case 4: 36 Elements

<table>
<thead>
<tr>
<th>LOCATION</th>
<th>STRUDEL</th>
<th>WARBURTON</th>
<th>LEMKE</th>
<th>LOCATION</th>
<th>STRUDEL</th>
<th>WARBURTON</th>
<th>LEMKE</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>C1</td>
<td>-0.8590</td>
<td>-0.5524</td>
<td>-0.9793</td>
</tr>
<tr>
<td>A2</td>
<td>0.4874</td>
<td>0.4733</td>
<td>0.5346</td>
<td>C2</td>
<td>-0.3595</td>
<td>-0.3772</td>
<td>-0.3447</td>
</tr>
<tr>
<td>A3</td>
<td>0.8590</td>
<td>0.8524</td>
<td>0.9793</td>
<td>C3</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>A4</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>C4</td>
<td>1.3334</td>
<td>1.475</td>
<td>1.207</td>
</tr>
<tr>
<td>B1</td>
<td>-0.4573</td>
<td>-0.4733</td>
<td>-0.5346</td>
<td>D1</td>
<td>-1.0</td>
<td>-1.0</td>
<td>-1.0</td>
</tr>
<tr>
<td>B2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>D2</td>
<td>-0.4938</td>
<td>-0.5267</td>
<td>-0.4654</td>
</tr>
<tr>
<td>B3</td>
<td>0.3594</td>
<td>0.3792</td>
<td>0.3447</td>
<td>D3</td>
<td>-1.3334</td>
<td>-1.475</td>
<td>-1.207</td>
</tr>
<tr>
<td>B4</td>
<td>0.4938</td>
<td>0.5267</td>
<td>0.4654</td>
<td>D4</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

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

Fig. 5: Fourth Mode

Fig. 6: Sixth Mode

Fig. 7: Eighth Mode

(3,0) + (0,2)

(2,0) + (0,2)

(2,1)

(2,2)
Fig. 8: Ninth Mode

Fig. 9: Tenth Mode

(1,3)-(3,1)

(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\textsuperscript{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:

<table>
<thead>
<tr>
<th>Old Name</th>
<th>New Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>TESTSTA</td>
<td>TESTSA</td>
</tr>
<tr>
<td>VARANCE</td>
<td>VARANC</td>
</tr>
<tr>
<td>TRANSIT</td>
<td>TRNSIT</td>
</tr>
<tr>
<td>DISCREG</td>
<td>DISREG</td>
</tr>
<tr>
<td>CNTNREG</td>
<td>CNTREG</td>
</tr>
<tr>
<td>RICTNWT</td>
<td>RICNWT</td>
</tr>
<tr>
<td>ASYMREG</td>
<td>ASMREG</td>
</tr>
<tr>
<td>ASYMFL</td>
<td>ASMFIL</td>
</tr>
<tr>
<td>EXPMDFL</td>
<td>EXPMDF</td>
</tr>
<tr>
<td>IMPMDFL</td>
<td>IMPMDF</td>
</tr>
</tbody>
</table>

(4) Some of the additional supporting subroutines/functions required were added and the names of these subroutines are given here:

(1) PNCH

(2) DIMAG

(3) DREAL

(4) BLOCK DATA
(5) None of the arguments of the subroutines were changed

The listing of the modified ORACLS package is given in the following pages. These routines have to be used in conjunction with Ref. (8). The numbers that appear in front of the FORTRAN statements are line numbers and have to be omitted.
SUBROUTINE ROTITL
IMPLICIT REAL*8 (A-H,O-Z)
COMMON/LINES/TITLE(10),TIL(3),NLP,LIN
COMMON/FORM/FMT1(2),FMT2(2),NEPR
COMMON/TOL/EPSAM,EPSBM,IACM
COMMON/CONV/SUMCV,RICTCV,SECV,MAXSUM
NLP = NO. LINES/PAGE VARIES WITH THE INSTALLATION
READ(5,100,END=90,ERR=91) TITLE
100 FORMAT(10A8)
90 CONTINUE
STOP 1
91 CONTINUE
STOP 2
END
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(1H1,10A8,3AB/)
LIN=2+N
IF (N.GT.NLP) LIN=2
20 RETURN
END
SUBROUTINE READ(I,A,NA,8,N8,C,NC,D,ND,E,NE)
IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION A(1),B(1),C(1),D(1),E(1)
DIMENSION NA(2),NB(2),NC(2),ND(2),NE(2),NZ(2)
READ(5,100) LAB,
      NZ(1), NZ(2)
CALL READ1(A, NA,NZ, LAB)
IF(I .EQ. 1) GO TO 999
READ(5,100) LAB,
      NZ(1), NZ(2)
CALL READ1(B, NB,NZ, LAB)
IF(I .EQ. 2) GO TO 999
READ(5,100) LAB,
      NZ(1), NZ(2)
CALL READ1(C, NC,NZ,LAB)
IF(I .EQ. 3) GO TO 999
READ(5,100) LAB,
      NZ(1), NZ(2)
CALL READ1(D, ND,NZ, LAB)
IF(I .EQ. 4) GO TO 999
READ(5,100) LAB,
      NZ(1), NZ(2)
CALL READ1(E, NE,NZ, LAB)
100 FORMAT(A4,I4)
999 RETURN
10 END
SUBROUTINE PRNT(A, NA, NAM, IOP)

IMPLICIT REAL*8 (A-H, O-Z)
DIMENSION A(1), NA(2)

COMMON /FORM/FMT1(2), FMT2(2), NEPR
COMMON/LINES/TITLE(10), TIL(3), NLP, LIN

C = NOTE NLP NO. LINES/PAGE VARIES WITH THE INSTALLATION.

DATA KZ, KW, KB /1M0, 1M1, 1M2/
NAME = NAM
II = IOP
NR = NA(1)
NC = NA(2)
NLST = NR * NC
IF (NLST .LT. 1 .OR. NR .LT. 1) GO TO 16
IF (NAME .EQ. 0) NAME = KB

C = SKIP HEADLINE IF REQUESTED.
GO TO (11, 10, 13, 12), II

CALL LNCNT(100)
CALL LNCNT(2)
WRITE(6, 177) KZ, NAME, NR, NC
177 FORMAT(A1, SX, A4, 4H MATRIX, SX, I3, 5H ROWS, SX, I3, 8H COLUMNS)
GO TO 13

12 CALL LNCNT(100)
GO TO 13

132 CALL LNCNT(2)
WRITE (6, 891)
891 FORMAT (I8)
GO TO 13

C = BELOW COMPUTE NR OF LINES/ROW = DECIDE IF 1 EXTRA BLANK LINE

JST = 1
C COMPUTE LAST ROW POSITION = 1 BELOW
NLST = NLST - NR
MN = NC
IF (NC .GT. NEPR) MN = NEPR
KLST = NR * (MN - 1)
CONTINUE
GO 912 J = JST, NR
CALL LNCNT(NLPW)
KLST = KLST + 1
WRITE (6, FMT1) (A(N), N = J, KLST, NR)
IF (NC .LE. NEPR) GO TO 912
NLST = NLST + 1
KNR = KLST + NR
WRITE (6, FMT2) (A(N), N = NKR, NLST, NR)
CONTINUE
RETURN

16 CALL LNCNT(1)
WRITE (6, 916) NAM, NA
916 FORMAT (* ERROR IN PRNT MATRIX ',A4,' HAS NA=', 2I6)
RETURN

END
SUBROUTINE EQUATE(A,NA,A,NH)
IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION A(1),B(1),NA(2),NB(2)
NA(1) = NA(1)
NA(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=',I6)
RETURN
END
SUBROUTINE TRANP(A, NA, R, NB)

IMPLICIT REAL*4 (A-H, O-Z)

DIMENSION A(1), B(1), NA(2), NB(2)

NR = NA(1)
NB = NA(2)
NC = NA(2)
LR = NR * NC

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

IR = 0

DO 300 I = 1, NR
   IJ = I - NR
   DO 300 J = 1, NC
      IJ = IJ + NR
      IR = IR + 1
     300 B(IR) = A(IJ)

RETURN

999 CALL LNCNT(1)

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

RETURN

END
SUBROUTINE SCALE (A, NA, M, NB, S)
IMPLICIT REAL*8 (A-H, O-Z)
DIMENSION A(1), R(1), NA(2), NB(2)
NA(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)

IMPLICIT REAL*8 (A-H, O-Z)

DIMENSION A(1), NA(2)

IF(NA(1), NE, NA(2)) GO TO 999

L=NA(1)*NA(2)

DO 100 IT=1, L

100 A(IT)=0.0

J = NA(1)

NAX = NA(1)

DO 300 I=1, NAX

J=NAX + J+1

300 A(J)=1.

GO TO 1000

999 CALL LNCNT (1)

WRITE(*, 50)(NA(I), I=1, 2)

50 FORMAT (' DIMENSION ERROR IN UNITY  NA= ,216)
SUBROUTINE NULL(A, NA)

IMPLICIT REAL*8 (A-H, O-Z)

DIMENSION A(1)
DIMENSION NA(2)
N = NA(1) * NA(2)

IF( NA(1) .LT. 1 .OR. N .LT. 1 ) GO TO 999

DO 10 I = 1, N

10 A(I) = 0, 0

RETURN

C

999 CONTINUE
WRITE (6, 50) NA

50 FORMAT(‘DIMENSION ERROR IN NULL NA = ’, I6)
RETURN
END
SUBROUTINE TRCE (A, NA, TR)
IMPLICIT REAL*8 (A-H, O-Z)
DIMENSION A(1)
DIMENSION NA(2)
IF (NA(1).NE.NA(2)) GO TO 600
N=NA(1)
TR=0.*0.
7 IF (N.LT. 1) GO TO 600
DO 10 I=1,N
4 M=I+N*(I-1)
10 TR=TR+A(M)
RETURN
600 CALL LNCNT(1)
WRITE (6,1600) NA
1600 FORMAT ('TRACE REQUIRES SQUARE MATRIX NA=",216)
RETURN
END
SUBROUTINE ADD (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(1) .NE. NB(1)) OR (NA(2) .NE. NB(2)) GO TO 999

NC(1) = NA(1)
NC(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 C(I) = A(I) + B(I)

GO TO 1000

999 CALL LNCNT (1)

WRITE (6, 50) NA, NB

50 FORMAT (" DIMENSION ERROR IN ADD NA='", 2I6, 5X, "NB='", 2I6)

1000 RETURN

* DIMENSION ERROR IN ADD NA='", 2I6, 5X, "NB='", 2I6)

END
SUBROUTINE SUAT(A, NA, NB, C, NC)
IMPLICIT REAL*8 (A-Z)
DIMENSION A(1:NA), C(1), NA(2), NB(2), NC(2)
IF(NA(1).NE.NB(1)) OR (NA(2).NE.NB(2)) GO TO 999
NC(1) = NA(1)
NC(2) = NA(2)
IF(NA(1).NE.NA(2)) GO TO 999
DO 300 I=1,L
300 C(I) = A(I) - B(I)
GO TO 1000
999 CALL LNCNT(1)
WRITE(6,50) NA, NB
50 FORMAT (' DIMENSION ERROR IN SUAT', NA='2I6,5X,NB='2I6)
1000 RETURN
END
SUBROUTINE MULT(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)
NC(1) = NA(1)
VC(2) = NB(2)
IF (NA(2).NE.NB(1)) GO TO 999
NAR = NA(1)
NAC = NA(2)
NRC = NB(2)
NAA = NAR * NAC
IF (NAR .LT. 1 .OR. NAA .LT. 1 .OR. NRC .LT. 1) GO TO 999
IR = 0
IK = NAC
DO 350 K = 1, NRC
IK = IK + NAC
DO 350 J = 1, NAR
IR = IR + 1
IM = IK
JI = J - NAR
V1 = 0.0
DO 300 I = 1, NAC
JI = JI + NAR
IB = IR + 1
V3 = A(JI)
V4 = B(I)
V2 = V3 + V4
V1 = V1 + V2
300 CONTINUE
350 CONTINUE
GO TO 1000
999 CALL LNCNT(1)
WRITE(6, 500) (NA(I), I = 1, 2), (NB(I), I = 1, 2)
500 FORMAT (* DIMENSION ERROR IN MULT * NA=*, 216, * NB=*, 216)
1000 RETURN
END
SUBROUTINE MAXEL(A, NA, ELMAX)
IMPLICIT REAL*8 (A-H, O-Z)
DIMENSION A(1), NA(2)

C N = NA(1) * NA(2)

ELMAX = DABS(A(1))
DO 100 I = 2, N
ELMAXI = DABS(A(I))
IF(ELMAXI.GT.ELMAX) ELMAX=ELMAXI
100 CONTINUE

RETURN
END
SUBROUTINE NORMS(MAXROW,M,N,A,IOPT,RLNORM)
IMPLICIT REAL*A(A-M,O-Z)
DIMENSION A(1)

C --- INITIALIZATION ---

RLNORM=0.
SUM=0.
I=MAXROW

C --- TRANSFER TO APPROPRIATE LOOP TO COMPUTE THE DESIRED NORM ---

C THIS LOOP COMPUTES THE ONE-NORM

DO 15 K=1,N
I=I+MAXROW
10 DO 15 J=1,M
L=I+J
15 SUM=SUM+ABS(A(L))
RETURN

C THIS LOOP COMPUTES THE EUCLIDEAN NORM

DO 25 K=1,N
I=I+MAXROW
20 DO 25 J=1,M
L=I+J
25 SUM=SUM+ABS(A(L))
RLNORM=SQRT(SUM)
RETURN

C THIS LOOP COMPUTES THE INFINITY-NORM

DO 35 K=1,N
L=L+MAXROW
30 DO 35 J=1,M
L=L+MAXROW
35 SUM=SUM+ABS(A(L))
IF(SUM.GT.RLNORM)RLNORM=SUM
30 SUM=0.
40 RETURN
45 END
SUBROUTINE JUXTC(A, NA, B, NB, C, NC)
IMPLICIT REAL*6(A-H, O-Z)
DIMENSION A(1), B(1), C(1), NA(2), NB(2), NC(2)
IF (NA(1) .NE. NB(1)) GO TO 600
NC(1) = NA(1)
NC(2) = NA(2) + NB(2)
L = NA(1) * NA(2)
NLC = NC(1) * NC(2)
IF (NC(1) .LT. 1 .OR. L .LT. 1) GO TO 600
IF (NC(2) .LT. 1) GO TO 600
MS = NA(1) * NA(2)
DO 10 I = 1, MS
10 C(I) = A(I)
DO 20 J = MS + 1, L
20 C(J) = A(I)
RETURN
600 CALL LNCNT(1)
WRITE (6, 1600) NA, NB
1600 FORMAT ('DIMENSION ERROR IN JUXTC, NA=', 2I6, '5X, 'NB=', 2I6)
RETURN
END
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
MC=NA(2)

MR=NA(1)
MRB=NR(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(*,1600) NA,NB

1600 FORMAT(' DIMENSION ERROR IN JUXTR, NA=',2I6,5X,'NB=',2I6)

RETURN

END
SUBROUTINE FACTOR(Q,NO,D,NO, IOP, IAC, DUMMY)

IMPLICIT REAL*8 (A-H,O-Z)

DIMENSION Q(1), D(1), DUMMY(1)

DIMENSION NO(2), NO(2), NOMUX(2)

IOP = 2
N = NO(1)
M = N**2
N1 = M + 1
N2 = N1 + N

CALL EQUATE(Q, NO, DUMMY, NO)
CALL SNVDEC(IOP, N, N, N, DUMMY, NOS, R, IAC, ZTEST, DUMMY(N1), D, IANK)

IF( IERR .EQ. 0 ) GO TO 200
CALL LNCNT(5)

IF( IERR .GT. 0 ) PRINT 100, IERR

100 FORMAT(//, ' IN FACTOR, SNVDEC HAS FAILED TO CONVERGE TO THE ', I4)
150 FORMAT(//, ' IN FACTOR, THE MATRIX SUBMITTED TO SNVDEC IS CLOSE TO SINGULAR VALUE AFTER 30 ITERATIONS'
160 IF( IERR .EQ. 1 ) PRINT 150, ZTEST, IANK

20 18 REDUCED THE RANK MAY ALSO BE REDUCED',/', CURRENT RANK = ', I4)
22 NOUM(1) = N
23 NOUM(2) = N
24 IF( IERR .EQ. -1 ) CALL PRNT(DUMMY(N1), NOUM, 4, SNVL, 1)
25 IF( IERR .GT. 0 ) RETURN

200 CONTINUE
26 NDUM(1) = N
29 DO 300 J = 1, N
31 M1 = (J-1) * N + 1
32 M2 = J * N
33 DO 300 I = M1, M2
34 K = N2 + 1
35 L = N1 + J - 1
36 IF( DUMMY(L) .EQ. 0 ) GO TO 300
37 DUMMY(K) = DSORT(DUMMY(L)) * DUMMY(1)
38 250 CONTINUE
39 NDUM(2) = N
40 GO TO 350
41 300 NDUM(2) = J - 1
43 350 CONTINUE
44 IF( DUMMY(N2) .LT. 0.0 ) CALL SCALE(DUMMY(N2), NOUM, DUMMY(N2), NOUM)
45 IF( IOP .EQ. 0 ) RETURN

46 CALL TRARP(DUMMY(N2), NDUM, D, ND)
47 CALL LNCNT(4)
49 CALL PRNT(4)
50 PRINT 400
51 400 FORMAT(//, ' FACTOR A AS (O TRANSPOSE)*XQ ', /)
52 CALL PRNT(Q,NO,4M Q, 1)
53 CALL PRNT(D, ND, 4M D, 1)
54 CALL MULT(DUMMY(N2), NOUM, D, ND, DUMMY, NO)
55 CALL PRNT(DUMMY, NO, 4M DTXD, 1)
56 RETURN
58 END
SUBROUTINE EIGEN(MAX, N, A, ER, EI, TSV, ILV, V, WK, IER)

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)

PRELIMINARY REDUCTION

CALL BALANC (MAX,N,A,LOW,IGH,WK)
CALL ELMHES (MAX,N,LOW,IGH,A,INT(1))

IV = TSV + ILV
IF (IV .NE. 0) GO TO 10

COMPUTE ALL EIGENVALUES AND NO EIGENVECTORS

CALL HOR (MAX,N,LOW,IGH,A,ER,EI,IER)
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
CONTINUE

SAVE A MATRIX FOR INVERSE ITERATION AND INITIALIZE WK(I,4)

ARRAY WHICH WILL BE A LOGICAL ARRAY IN CALLED SUBROUTINES

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

COMPUTE ALL EIGENVALUES (UNORDERED)

CALL HOR (N,N,LOW,IGH,A,ER,EI,IER)
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

FIND LARGEST ILV EIGENVALUES AND FLAG THEM

DO 50 I=1,ILV
   P = -1.0
   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)

CONTINUE

IF (EI(K) .EQ. 0.) GO TO 60

IF (EI(K) .GT. 0.) GO TO 55

IF (SELECT(K-1)) GO TO 50

ILV = ILV+1
SELECT(K-1) = .TRUE.

A-19
GO TO 60
CONTINUE
IF (.NOT.SELECT(K)) ILV = ILV+1
CONTINUE
IF (ISV.EQ.0) GO TO 90
C FIND SMALLEST ISV EIGENVALUES AND FLAG THEM
DO 65 I=1,N
   WK(I,3) = DABS(WK(I,3))
65 CONTINUE
DO 90 I=1,ISV
   P = 1.074
   DO 70 J=1,N
      IF (WK(J,3) .GE. P) GO TO 70
      K = J
      P = WK(J,3)
70 CONTINUE
90 CONTINUE
SELECT(K) = .TRUE.
WK(K,3) = 1.074
CONTINUE
IF (EI(K) .EQ. 0.) GO TO 90
IF (EI(K) .GT. 0.) GO TO 95
IF (SELECT(K-1)) GO TO 90
ISV = ISV+1
SELECT(K-1) = .TRUE.
GO TO 90
CONTINUE
IF (.NOT.SELECT(K)) ISV = ISV+1
CONTINUE
C FIND EIGENVECTORS FOR FLAGGED EIGENVALUES
CALL INVIT (MAX,N,A,ER,FI,SELECT,N,M,V,IERR,WK(1,6),WK(1,3)
1   WK(1,5))
C RACK TRANSFORM EIGENVECTORS TO ORIGINAL MATRIX
CALL ELMBAK (MAX,LOW,IGH,LHINT(A),M,V)
CALL BALBAC (MAX,N,LOW,IGH,WK,M,V)
C SEPARATE FLAGGED EIGENVALUES FROM UNFLAGGED EIGENVALUES
IV = ISV + ILV
IF (IV.LE.N) GO TO 100
ILV = N-ISV
IV = 4
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)
A-20
126  &K(IC,2) = EI(I)
127  IC = IC
128  CONTINUE
129  &K(KC,3) = ER(I)**2 + EI(I)**2
130  CONTINUE
131  C
132  C  NORMALIZE VECTORS TO UNIT LENGTH AND STORE FOR REORDERING
133  C
134  J = 0
135  CONTINUE
136  J = J+1
137  IF ( &K(J,2) .NE. 0. ) GO TO 144
138  SUM = 0.
139  DO 152 I=1,N
140      SUM = SUM + V(I,J)**2
141  CONTINUE
142  IF (SUM .EQ. 0.) GO TO 151
143  DO 153 I=1,N
144      V(I,J) = V(I,J)/SUM
145  CONTINUE
146  CONTINUE
147  GO TO 158
148  CONTINUE
149  J = J+1
150  SUM = 0.
151  DO 155 I=1,N
152      SUM = SUM + V(I,J)**2 + V(I,JP1)**2
153  CONTINUE
154  IF (SUM .EQ. 0.) GO TO 157
155  SUM = DSORT(SUM)
156  DO 158 I=1,N
157      T(J,3) = T(J,3) + 1
158      SUM = SUM + V(I,J)**2 + V(I,JP1)**2
159  CONTINUE
160  CONTINUE
161  J = J+1
162  CONTINUE
163  IF ( J .LT. IV ) GO TO 151
164  IC = 0
165  LC = 0
166  IF ( ISV .EQ. 0 ) GO TO 190
167  C  ORDER SMALLEST ISV EIGENVALUES AND EIGENVECTORS FOR OUTPUT
168  C
169  DO 190 I=1,ISV
170      P = 1.074
171      DO 190 J=1,IV
172          IF ( T(J,3) .GE. P ) GO TO 190
173          K = J
174          P = T(K,3)
175      CONTINUE
176      IC = IC+1
177      LC = LC+1
178      ER(IC) = T(K,1)
179      EIC(IC) = T(K,2)
180      CONTINUE
181      DO 190 J=1,N
182          V(J,LC) = T(J,3) + 1.074
183      CONTINUE
184      CONTINUE
185      CONTINUE
186      CONTINUE
187      IF ( IV .EQ. N ) GO TO 220
188  C  A-21
90 C     ORDER UNFLAGGED EIGENVALUES FOR OUTPUT
191 C     IV1 = IV + 1
192       IUF = N - IV
193       DO 210 IV = 1, IUF
194           P = 1.074
195       DO 200 J = IV1, N
196           IF (WK(J,3) .GE. P) GO TO 200
197           K = J
198           P = WK(J,3)
199       200       CONTINUE
200       IC = IC + 1
201       ER(IC) = WK(K,1)
202       EI(IC) = WK(K,2)
203       WK(K,3) = 1.074
204       210       CONTINUE
205       IF (ILV .EQ. 0) GO TO 260
206       IC = IC + 1
207       ER(IC) = WK(K,1)
208       EI(IC) = WK(K,2)
209       DO 240 J = 1, N
210           V(J,LC) = WK(J,4)
211       240       CONTINUE
212       WK(K,3) = 1.074
213       250       CONTINUE
214       260       CONTINUE
215       RETURN
216       END
SUBROUTINE SYMPOS(MAXN,N,ANRHS,B,IOPT,IFAC,DETERM,ISCALE,P,IERR)

IMPLICIT REAL*8 (A-M,0-2)
DIMENSION A(MAXN,N),B(MAXN,ANRHS),P(N)

DATA R1,R2,1.00+75,1.00-75/

TEST FOR A SCALAR MATRIX (IF COEFFICIENT MATRIX IS A SCALAR--SOLVE AND COMPUTE DETERMINANT IF DESIRED)

IERR = 0
NM1 = N-1

IF (NM1.GT.0) GO TO 20

ISCALE = 0
DETERM = A(1,1)
P(1) = 1.0/A(1,1)

DO 10 J=1,ANRHS
B(1,J) = B(1,J)/DETERM
10 CONTINUE
RETURN

TEST TO DETERMINE IF CHOLESKY DECOMPOSITION OF COEFFICIENT MATRIX IS DESIRED

20 IF (IFAC.EQ.1) GO TO 160

25 INITIALIZE DETERMINANT EVALUATION PARAMETERS
DETERM=1.0
ISCALE=0

2A LOOP TO PERFORM CHOLESKY DECOMPOSITION ON THE COEFFICIENT MATRIX A (I.E., MATRIX A WILL BE DECOMPOSED INTO THE PRODUCT OF A UNIT LOWER TRIANGULAR MATRIX (L), A DIAGONAL MATRIX (D), AND THE TRANSPOSE OF L (LTRANSPOSE)).

30 DO 150 I=1,N
IM1 = I-1
35 DO 150 J=1,1
38 X = A(J,I)
39 DO 10 K=1,IM1
40 Y = A(I,K)
42 A(I,K) = Y*P(K)
43 X = X - Y*A(I,K)
46 CONTINUE
48 IF (I.ME.0) GO TO 50
50 IF (X.LE.0.0) IERR = 1

55 COMPUTE INVERSE OF DIAGONAL MATRIX D**-1 = 1/P
57 P(I) = 1.0 / X

58 TEST TO SEE IF DETERMINANT IS TO BE EVALUATED
60 IF (IOPT.EQ.0) GO TO 150

A-23
C SCALE THE DETERMINANT (COMPUTE THE DETERMINANT EVALUATION)
PARAMETERS DETERM AND ISCALE

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(DETERM).LT.R1) GO TO 90
PIVOT = PIVOT*R2
ISCALE = ISCALE+1
GO TO 80
90 IF(DABS(DETERM).GT.R2) GO TO 100
PIVOT = PIVOT*R1
ISCALE = ISCALE-1
GO TO 90
100 DETERM = DETERM*PIVOT
GO TO 150

C USING THE LOWER TRIANGULAR ELEMENTS OF MATRIX A, THIS
SECTION COMPUTES THE UNIT LOWER TRIANGULAR MATRIX
110 JM1 = J-1
120 DO 130 K=1,NM1
       130 X = X + A(I,K)*A(J,K)
140 A(I,J) = X
150 CONTINUE

SECTION TO APPLY BACK SUBSTITUTION TO SOLVE L*Y = B FOR
UNIT LOWER TRIANGULAR MATRIX AND CONSTANT COLUMN VECTOR R
160 IF(IFAC.EQ.2) RETURN
170 DO 180 I=2,N
       180 X = P(I) - B(I,J)
190 CONTINUE

SECTION TO SOLVE (LTRANSPOSE)*X = (D**-1)*Y FOR TRANSPOSE
OF UNIT LOWER TRIANGULAR MATRIX AND INVERSE OF DIAGONAL
MATRIX
200 I = NM1+1
210 Y = P(N)
220 DO 230 J=1,NM1
       230 R(N,M1) = B(N,J)*Y
240 CONTINUE
250 Y = P(NM1)

A-24
26 C
27 DO 220 J=1,NM1
28 X = R(NM1,J)*Y
29 C
30 DO 210 K=1,N
31 X = X - A(K,NM1)*B(K,J)
32 210 CONTINUE
33 C
34 B(NM1,J) = X
35 220 CONTINUE
36 C
37 C
38 C TEST TO DETERMINE IF SOLUTIONS HAVE BEEN DETERMINED FOR ALL COLUMN VECTORS
39 C
40 NM1 = NM1-1
41 IF (NM1 .GT. 0) GO TO 200
42 C
43 RETURN
44 END
SUBROUTINE GELM(NMAX,N,A,NRHS,B,IPIVOT(IFAC,WM,IMM)
IMPLICIT REAL*4 (A-H,O-Z)
DIMENSION A(NMAX,1),B(NMAX,1),IPIVOT(1),WM(1)
IERR=0
C TEST FOR L/U FACTORIZATION
IF(IFAC.EQ.1)GO TO 10
CALL DETFC(NMAX,N,A,IPIVOT,IFAC,DETERM,ISCALE,WM,IERR)
IF(IERR.GT.0)RETURN
C DET=DETERM((10,*(100*ISCALE))
C TEST FOR SCALAR A MATRIX
IF(NM1.GT.0)GO TO 40
DO 20 I=1,NRHS
20 B(I,1)=B(I,1)/A(1,1)
IF(IFAC.E0.2)WK(1)=DETA
RETURN
30 RETURN
40 DO 100 M=1,NRHS
10 P=0.0
100 CONTINUE
C PIVOT THE M-TH COLUMN OF A MATRIX
DO 50 I=1,NM1
50 KI=PIVOT(I)
60 P=P+A(I,K)*WK(K)
70 WK(I)=B(I,M)-P
C FORWARD SUBSTITUTION
WK(1)=B(1,M)
DO 70 I=2,N
IM1=I-1
70 B(I,M)=(B(I,M)-P)/A(I,I)
C BACK_SUBSTITUTION
P(AI,M)=WK(N)/A(N,N)
DO 90 J=1,NM1
I=N-J
90 I=IP1+1
P=WK(I)
DO 100 K=IP1,N
100 P=P+A(I,K)*R(K,M)
R(I,M)=P/A(I,I)
CONTINUE
IF(IFAC.EQ.2)WK(1)=DETA
RETURN
END
SUBROUTINE SNVDEC(IOP, MD, ND, M, N, A, NOS, R, IAC, ZTEST, Q, V, IRANK, APLUS)

IERR)

IMPLICIT REAL*8 (A-H, O-Z)

LOGICAL MITHU, MITHV

DIMENSION A(MD, N), V(ND, N), R(N), E(LI0)

DIMENSION B(MD, NOS), APLUS(ND, M)

C

TEST FOR SCALAR OR VECTOR A

IF( N .GE. 2 ) GO TO 3000

C

IERR = 0

ZTEST = 10.**(1-AIC)

SUM = 0.0

DO 1000 I = 1, M

SUM = SUM + A(I, 1)*A(I, 1)

1000 CONTINUE

SUM = DSQRT(SUM).

IRANK = 0

IF( SUM .GT. ZTEST ) IRANK = 1

C

IF( IOP .EQ. 1 ) RETURN

V(I, 1) = 1.0

IF( IRANK .EQ. 0 ) GO TO 1200

DO 1100 I = 1, M

A(I, 1) = A(I, 1)/SUM

1100 CONTINUE

GO TO 1300

C

IF( IOP .EQ. 2 ) RETURN

IF( IOP .EQ. 4 ) GO TO 1860

IF( IRANK .EQ. 0 ) GO TO 1600

DO 1500 J = 1, NOS

Z = 0

DO 1400 I = 1, M

Z = Z + A(I, 1)*A(I, J)/SUM

1400 CONTINUE

1300 CONTINUE

C

IF( IOP .EQ. 3 ) RETURN

1850 CONTINUE

IF( IRANK .EQ. 0 ) GO TO 2000

DO 1900 I = 1, M

APLUS(I, 1) = A(I, 1)/SUM

1900 CONTINUE

RETURN

C

DO 2100 I = 1, M

APLUS(I, 1) = 0.0

2100 CONTINUE

RETURN
63 C
64 C
65 3000 CONTINUE
66 C
67 C
68 C
69 TOL=1.0D-60
70 SIZE=0.0
71 NPI=N+1
72 C
73 C  COMPUTE THE E-NORM OF MATRIX A AS ZERO TEST FOR SINGULAR VALUES.
74 C
75 SUM=0.0
76 DO 500 I = 1, M
77 DO 500 J = 1, N
78 500 SUM = SUM + A(I,J)**2
79 ZTEST = DSQRT(SUM)
80 ZTEST = ZTEST**10.0**(IAC)
81 C
82 510 IF (IOP .NE. 1) GO TO 515
83 WITHU=.FALSE.
84 WITHV=.FALSE.
85 GO TO 520
86 515 WITHU=.TRUE.
87 WITHV=.TRUE.
88 520 CONTINUE
89 G = 0.0
90 X = 0.0
91 DO 30 I = 1, N
92 C
93 C  HOUSEHOLDER REDUCTION TO BIDIAGONAL FORM.
94 C
95 E(I) = G
96 S = 0.0
97 L = I+1
98 C
99 C  ANNIHILATE THE I-TH COLUMN BELOW DIAGONAL.
100 C
101 DO 3 J = I, M
102 3 S = S + A(J,I)**2
103 G = 0.0
104 IF(S .LT. TOL) GO TO 10
105 G = DSQRT(S)
106 F = A(I,I)
107 IF(F .GE. 0.0) G = -G
108 M = F*G -S
109 A(I,I) = F=G
110 IF(I .EQ. N) GO TO 10
111 DO 9 J = L, N
112 S = 0.0
113 DO 7 K = I, M
114 7 S = S + A(K,I)*A(K,J)
115 F = S/M
116 DO 8 K = I, M
117 8 A(K,J) = A(K,J) + F*A(K,I)
118 9 CONTINUE
119 10 G(I) = G
120 IF(I .EQ. N) GO TO 20
121 C
122 C  ANNIHILATE THE I-TH ROW TO RIGHT OF SUPER-DIAG.
123 C
124 S = 0.0
125 DO 11 J = L, N

A-28
26 11 $S = S + A(I,J) \times 2$
27 $G = 0,0$
28 IF ($S \cdot LT. TOL$) GO TO 20
29 $G = 0.0, SRT(S)$
30 $F = A(I,I+1)$
31 IF ($F \cdot GE. 0,0$) $G = -G$
32 $H = F \times G$ -S
33 $A(I,I+1) = F \times G$
34 DO 15 J = L,N
35 E(J) = A(I,J)/H
36 DO 19 J = L,M
37 $S = 0.0$
38 DO 16 K = L,N
39 IF ($G \cdot LT. TOL$) GO TO 20
40 $S = 9 + A(J,K) \times A(I,K)$
41 $S = G \times A(J,K) + S \times E(K)$
42 CONTINUE
43 $Y = DARS(R(I)) + DARS(E(I))$
44 IF ($Y \cdot GT. SIZE$) SIZE = Y
45 CONTINUE
46 IF ($\cdot NOT. WITHV$) GO TO 41
47 ACCUMULATION OF RIGHT TRANSFORMATIONS.
48 DO 40 II = 1,N
49 $I = NP1 - II$
50 IF ($I \cdot EQ. N$) GO TO 39
51 $I = GP0, 0.0$ GO TO 37
52 $H = A(I,I+1) \times G$
53 DO 32 J = L,N
54 $S = 0,0$
55 DO 31 K = L,N
56 $V(CJ,I) = A(I,J)/H$
57 DO 36 J = L,N
58 $S = 0,0$
59 $V(CJ,I) = 0.0$
60 $V(CJ,1) = 0.0$
61 $G = E(I)$
62 CONTINUE
63 IF ($\cdot NOT. WITHV$) GO TO 53
64 ACCUMULATION OF LEFT TRANSFORMATIONS.
65 DO 52 II = 1,N
66 $I = NP1 - II$
67 $L = I + 1$
68 $G = G(I)$
69 IF ($I \cdot EQ. N$) GO TO 43
70 DO 42 J = L,N
71 $A(I,J) = 0.0$
72 CONTINUE
73 IF ($G \cdot EQ. 0.0$) GO TO 49
74 $H = A(I,I) \times G$
75 $A(I,J) = 0.0$
76 CONTINUE
77 CONTINUE
78 CONTINUE
79 CONTINUE
80 CONTINUE
81 $G = E(I)$
82 $A(I,J) = 0.0$
83 CONTINUE
84 CONTINUE
85 CONTINUE
86 $G = E(I)$
87 $A(I,J) = 0.0$
88 CONTINUE
89 $G = E(I)$
90 $A(I,J) = 0.0$
89 44 S = S + A(K, I)*A(K, J) SNVO10
90 F = 3/K SNVO10
91 DO 45 K = I, M SNVO10
92 45 A(K, J) = A(K, J) + F*A(K, I) SNVO10
93 CONTINUE SNVO10
94 46 CONTINUE SNVO10
95 47 DO 50 J = I, M SNVO10
96 50 A(J, I) = A(J, I)/G SNVO10
97 GO TO 51 SNVO10
98 51 I = 0, 0 SNVO10
99 52 CONTINUE SNVO10
100 53 CONTINUE SNVO10
102 C 103 CONTINUE SNVO10
104 DO 100 K=1, N SNVO10
105 K=KPI=KK SNVO10
106 ITCNT=0 SNVO10
107 KP1=K+1 SNVO10
109 C TEST F SPLITTING. SNVO10
110 CONTINUE SNVO10
112 DO 60 L=1, K SNVO10
114 L=KPI=LL SNVO10
115 IF((SIZE+DABS(E(L)))/EQ.SIZE) GO TO 64 SNVO10
116 L=1=1 SNVO10
117 IF((SIZE+DABS(Q(LM1)))/EQ.SIZE) GO TO 61 SNVO10
118 60 CONTINUE SNVO10
119 C CANCELLATION OF E(L) IF L .GT. 1. SNVO10
121 61 C=0, 0 SNVO10
122 S=1, 0 SNVO10
124 DO 63 I=L, K SNVO10
126 F=S*E(I) SNVO10
127 E(I)=C*E(I) SNVO10
128 IF((SIZE+DABS(F))/EQ.SIZE) GO TO 64 SNVO10
129 G=Q(I) SNVO10
130 Q(I)=SQRT(F+F+G+G) SNVO10
131 H=Q(I) SNVO10
132 C=G/H SNVO10
133 S=F/H SNVO10
134 IF(.NOT.NITHU) GO TO 63 SNVO10
135 DO 62 J=1, M SNVO10
136 Y=A(J, L1) SNVO10
137 Z=A(J, I) SNVO10
138 A(J, L1)=Y*C+Z*S SNVO10
139 A(J, I)=Y*S+Z*C SNVO10
140 62 CONTINUE SNVO10
141 63 CONTINUE SNVO10
142 C TEST F CONVERGENCE. SNVO10
143 CONTINUE SNVO10
146 IF(L.EQ.K) GO TO 75 SNVO10
148 ITCNT .LE. 30) GO TO 65 SNVO10
149 IFRIA = KK SNVO10
150 RETURN SNVO10
151 65 ITCNT = ITCNT + 1 SNVO10
SHIFT FROM LOWER 2X2.

X=G(L)
Y=G(K=1)
G=E(K=1)
F=(Y+Z)*(Y+Z)+(G+H)/(G+H)/(G+H)
G=DSQRT(F+F+1.0)
IF(FLT.0,0) G=G
F = ((X-Z)*(X+Z)+(G+H))/(G+H)

NEXT OR TRANSFORMATION.

C=1.0
S=1.0
LP1=LP1+1
DO 73 I=LP1,K
G=E(I)
Y=G(I)
H=S+G
G=C*G
Z=DSQRT(F+F+H)
E(I-1)=E
C=F/Z
S=H/Z
F=C*G+S
G=x+S+G*C
H=y+S
Y=y+C
IF(NOT.WITHV) GO TO 70
DO 68 J=1,M
X=A(J-I)
Z=A(J,I)
V(J,I-1)=X*C+Z*S
V(J,I)=X*S+Z*C
68 CONTINUE
70 CONTINUE

IF(NOT.WITHV) GO TO 73
DO 72 J=1,M
Y=4(K-I)
Z=4(K,I)
A(J,I-1)=Y*C+Z*S
A(J,I)=Y*S+Z*C
72 CONTINUE

73 E(L) = 0.0
F(K)=F
G(K)=Y
GO TO 59
73 E(L) = 0.0
F(K)=F
G(K)=Y
GO TO 59

CONVERGENCE.

CONTINUE

IF(Z.GE.0.0) GO TO 100

A-31
31  
31  Q(K)=Z  
31  IF(.NOT.NOTHV) GO TO 100  
31  DO 76 J=1,N  
31  76  V(J,K)=V(J,K)  
31  100 CONTINUE  
31  C  
32  IERR=0  
32  DO 240 I=2,N  
32  I=I+1  
32  K=I  
32  P=Q(I)  
32  DO 250 J=I,N  
32  IF (Q(J),LE.P) GO TO 250  
32  K=J  
32  P=Q(J)  
32  250 CONTINUE  
33  C  
33  IF (K.EQ.I) GO TO 240  
33  Q(K)=Q(I)  
33  Q(I)=P  
34  C  
34  IF (IOP.EQ.1) GO TO 240  
34  DO 260 J=1,N  
34  P=V(J,I)  
34  V(J,I)=V(J,K)  
34  V(J,K)=P  
34  260 CONTINUE  
35  C  
35  DO 270 J=1,M  
35  P=A(J,I)  
35  A(J,I)=A(J,K)  
35  A(J,K)=P  
35  270 CONTINUE  
36  C  
36  280 CONTINUE  
37  C  
37  J=N  
37  290 IF (Q(J),GT,ZTEST) GO TO 300  
37  Q(J)=0.0  
37  J=J+1  
37  300 IRANK =J  
38  TEMP = ZTEST/Q(J)  
38  IF (TEMP,GT,.0625) IERR=1  
39  C  
39  IF (IOP.LT.3) RETURN  
39  IF (IOP.GT.3) GO TO 170  
39  DO 160 L=1,NOS  
39  DO 130 J=1,IRANK  
39  SUM=0.0  
39  DO 120 I=1,M  
39  SUM =SUM + A(I,J)*B(I,L)  
39  130 F(J)= SUM/Q(J)  
39  C  
39  DO 150 K=1,N  
39  SUM=0.0  
39  DO 140 I=1,IRANK  
39  SUM =SUM + V(K,I)*E(I)  
39  150 R(K,L)=SUM  
39  160 CONTINUE  
39  C  
39  170 DO 200 J=1,M  
39  
A-32
37 DO : 90 I=1,N
38 SUM=0,N
39 DO 180 K=1,IRANK
40 180 SUM =SUM + V(I,K)*A(J,K)/Q(K)
41 190 APLUS(I,J)= SUM
42 200 CONTINUE
43 C
44 IF ( IOP .EQ. 4) RETURN
45 DO 230 K=1,NOV
46 220 SUM=0.0
47 DO 210 J=1,N
48 210 SUM=SUM+APLUS(I,J)*B(J,K)
49 220 E(I)=SUM
50 DO 225 I=1,N
51 225 B(I,K)=F(I)
52 230 CONTINUE
53 RETURN
54 END
SUBROUTINE SUM(A,NA,B,NB,C,NC,IOP,SYM,DUMMY)

IMPLICIT REAL*4(A-H,O-Z)

DIMENSION A(1),B(1),C(1),DUMMY(1)

DIMENSION NA(2),NB(2),NC(2)

LOGICAL SYM

COMMON/CNV,SUMCV,RICTCV,ERCV,MAXSUM

IF(IOP .EQ. 0) GO TO 100

PRINT 50

50 FORMAT(/, 'LINEAR EQUATION SOLVER X = A*C + B')

CALL PRINT(A,NA,4M A,1)

IF(SYM) GO TO 75

CALL PRINT(C,NC,4M C,1)

GO TO 85

75 CONTINUE

PRINT 80

80 FORMAT(/, 'C = A.TRANSP.',/)

85 CONTINUE

CALL PRINT(B,NB,4M B,1)

100 CONTINUE

N1 = 1 + NA(1) * NC(1)

I=1

200 CONTINUE

CALL MULT(A,NA,B,NB,DUMMY,NB)

CALL MULT(DUMMY,NB,C,NC,DUMMY(N1),NB)

CALL MAXEL(B,NB,WNS)

CALL MAXEL(DUMMY(N1),NB,WN_DX)

IF(WNS .GE. 1.0) GO TO 225

IF(WN_DX .LT. SUMCV) GO TO 300

GO TO 235

225 IF(WN_DX .LT. SUMCV) GO TO 300

GO TO 235

235 CONTINUE

CALL ADD(R,NB,DUMMY(N1),NB,B,NB)

CALL MULT(A,NA,A,NA,DUMMY,NA)

CALL EQUATE(DUMMY,NA,A,NA)

IF(SYM) GO TO 245

CALL MULT(C,NC,C,NC,DUMMY,NC)

CALL EQUATE(DUMMY,NC,C,NC)

GO TO 250

245 CONTINUE

CALL TRAPN(A,NA,C,NC)

I=I+1

IF(I .LE. MAXSUM) GO TO 200

CALL LCNT(3)

PRINT 275,MAXSUM

275 FORMAT(/, 'IN SUM, THE SEQUENCE OF PARTIAL SUMS HAS EXCEEDED STAGS')

276 IF(I .LT. 15) WITHOUT CONVERGENCE')

GO TO 250

300 CONTINUE

IF(IOP .EQ. 0) RETURN

CALL PRINT(9,NB,4M X,1)

RETURN

END
SUBROUTINE BILINE(A, NA, B, NC, C, IOP, BETA, SYM, DUMMY)
IMPLICIT REAL*8 (A-H, O-Z)
DIMENSION A(1), B(1), C(1), DUMMY(1)
DIMENSION NA(2), NB(2), NC(2), DUMMY(2)
DIMENSION IOP(2)
LOGICAL SYM
IF (IOP(1) .EQ. 0) GO TO 300
IF (SYM) GO TO 100
CALL LNCNT(3)
PRINT 50
50 FORMAT(//, 'LINEAR EQUATION SOLVER A X + B = C')
CALL PRINT(A, NA, 4M A, 1)
CALL PRINT(R, NB, 4M B, 1)
GO TO 200
100 CONTINUE
CALL LNCNT(3)
PRINT 150
150 FORMAT(//, 'LINEAR EQUATION SOLVER (B TRANSPOSE) X + B = C')
CALL TRAP(A, NA, DUMMY, NDUM)
CALL PRINT(DUMMY, NDUM, 4M B, 1)
200 CONTINUE
CALL PRINT(C, NC, 4M C, 1)
300 CONTINUE
C
IOPTT = 0
N = NA(1)**2
M = NA(1)**2
C
IF (IOP(2) .EQ. 0) GO TO 500
N1 = N + 1
CALL EQUATE(A, NA, DUMMY, NA)
N2 = N1 + NA(1)
N3 = N2 + NA(1)
ISV = 0
ILV = 0
NEVL = NA(1)
CALL EIGEN(A(1), NA(1), DUMMY, DUMMY(N1), DUMMY(N2), ISV, ILV, V, DUMMY(N1))
IF (IERR .EQ. 0) GO TO 350
CALL LNCNT(3)
PRINT 325, IERR
325 FORMAT(//, 'IN BILINE, THE ', I4, ' EIGENVALUE OF A HAS NOT BEEN DETERMINED AFTER 30 ITERATIONS')
IERR = 1
CALL NORMS(NEVL, NEVL, NEVL, A, IERR, BETA)
RETA = BETA
GO TO 345
350 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 365
BETA = (DUMMY(N1)*CN + DUMMY(J)*CO)/CD
IF (BETA .LE. 0.0) GO TO 365
BETA = DSQRT(BETA)
GO TO 345
365 CONTINUE
C
BETA = 0.0
A-35
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

385 CONTINUE
   IF(SYM) GO TO 900
   CALL EQUATE(8, NB, DUMMY, N8)
   N1 = N1 + 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(NB), 13, IERR)
   IF(IERR .EQ. 0) GO TO 450
   CALL LNCNT(3)
   PRINT 400, IERR
   400 FORMAT('IN BILIN, THE I4 EIGENVALUE OF 0 WAS NOT FOUND AFTER 30 ITERATIONS')
   IERR = 1
   CALL NORMS(NEV, 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

465 CONTINUE
   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

485 CONTINUE
   BETA1 = (BETA + BETA1) / 2.

500 CONTINUE
   N1 = N + 1
CALL EQUATE(A, NA, DUMMY, NA)
CALL EQUATE(A, NA, DUMMY(N1), NA)
CALL SCALE(DUMMY, NA, DUMMY, NA, -1.0)
L = NA(1)
DO 525 I=1, NAX
  L = L + NAX + 1
M1 = L + N
DUMMY(L) = BETA = A(L)
DUMMY(M1) = BETA + A(L)
525 CONTINUE
N2 = N1 + N
N3 = N2 + NC(1)*NC(2)
GAM = -2.0*BETA
IF (.NOT., SYM) GO TO 600
CALL UNITY(DUMMY(N3), NA)
N4 = N3 + N
NDUM(2) = NDUM(2) + NA(1)
N5 = N4 + NA(1)
IFAC = 0
CALL GELIM(NA(1), NA(1), DUMMY, NDUM(2), DUMMY(N1), DUMMY(N4), IFAC, DUMMY(N3), IERR)
IF (IERR .EQ. 1) PRINT 625
CALL EQUATE(DUMMY(N1), NA)
CALL EQUATE(DUMMY(N2), NC, C, NC)
CALL TRANP(DUMMY, NA, DUMMY(N1), NA)
CALL TRANP(DUMMY(N3), NA, DUMMY(N2), NA)
CALL MULT(C, NC, DUMMY(N2), NA, DUMMY(N3), NA)
CALL SCALE(DUMMY(N3), NC, C, NC, GAM)
CALL SCALE(DUMMY(N1), NC, C, NC, GAM)
CALL SUU(DUMMY(N1), DUMMY, DUMMY(N2))
GO TO 700
600 CONTINUE
N4 = N3 + NA(1)
IFAC = 0
CALL GELIM(NA(1), NA(1), DUMMY, NDUM(2), DUMMY(N1), DUMMY(N3), IFAC, DUMMY(N1), IERR)
IF (IERR .EQ. 1) PRINT 625
625 FORMAT('IN BILIN, THE MATRIX (BETA) I = A IS SINGULAR. INCREASE BETA.')
CALL EQUATE(DUMMY(N1), NA, DUMMY, NA)
CALL EQUATE(DUMMY(N2), NC, C, NC)
N2 = M + N
CALL EQUATE(B, NB, DUMMY(N1), NB)
CALL EQUATE(B, NB, DUMMY(N2), NB)
CALL SCALE(DUMMY(N1), NB, DUMMY(N1), NB, -1.0)
L = NB(1)
NAX = NA(1)
DO 650 I = 1, NAX
  L = L + NAX + 1
L = L + N
M1 = L + N2 = 1
DUMMY(L1) = BETA = B(L)
DUMMY(M1) = BETA + B(L)
650 CONTINUE
N3 = N2 + M
CALL TRANP(DUMMY(N1), NB, DUMMY(N3), NB)
CALL EQUATE(DUMMY(N3),NB,DUMMY(N1),NB)
CALL TRAMP(DUMMY(N2),NB,DUMMY(N3),NB)
CALL EQUATE(DUMMY(N3),NB,DUMMY(N2),NB)
CALL TRAMP(C,NC,DUMMY(N3),NDUM)
NSDUM = NDUM(?)
NDUM(2) = NDUM(2) + NB(2)
IFAC = 0
N4 = N3 + NC(1) * NC(2)
N5 = N4 + NB(1)
CALL GELIM(NB(1),NB(1),DUMMY(N1),NDUM(2),DUMMY(N2),DUMMY(N4),IFAC)
DUMMY(N5),IERR)
IF(IFRR .EQ. 1) PRINT 675
675 FORMAT(/**, 'IN BILIN, THE MATRIX (BETA) = B IS SINGULAR, INCREASE**)
1 BETA */)
CALL TRAMP(DUMMY(N2),NB,DUMMY(N1),NB)
NDUM(2) = NSDUM
CALL TRAMP(DUMMY(N3),NDUM,C,NC)
CALL SCALE(C,NC,C,NC,GAM)
N2 = N + M + 1
CALL SUM(DUMMY,NA,C,NC,DUMMY(N1),NB,IOPTT,SYM,DUMMY(N2))
C 9IO210n
700 CONTINUE
IF(IOPT(1) .EQ. 0) RETURN
CALL PRNT(C,NC,4H X ,1)
RETURN.
END
SUBROUTINE BARSTM(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), NOUN1(2), NOUN2(2), NOUN3(2), NOUN4(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 PRINT(A, NA, 4M A +1)
CALL PRINT(B, NB, 4M B +1)
GO TO 200

100 CONTINUE
CALL LNCNT(3)
PRINT 150

150 FORMAT(//,"LINEAR EQUATION SOLVER (B TRANSPOSE)X + XB = C")
CALL TRAPN(A, NA, DUMMY, NOUN1)
CALL PRINT(DUMMY, NOUN1, 4M B +1)

200 CONTINUE
CALL PRINT(C, NC, 4M C +1)

250 CONTINUE
CALL EQUATE(A, N9, OUMMY(LU), N9, M3)
N1 = (NA(1)**2+2)**1
N2 = N1 + NA(1)**2+1
DO 300 I = N1, N2
DUMMY(I) = 0.0
300 CONTINUE
CALL NULL(DUMMY(N1), N9, M3)

310 N = NA(1)**2+1

50 CALL EQUATE(q, N9, OUMMY(LU), N9, M3)
M1 = LU + NB(1)**2 +1
M2 = M1 + NB(1)**2+1
DO 400 I = M1, M2
DUMMY(I) = 0.0
400 CONTINUE
CALL EQUATE(DUMMY(LU), N9, M3, DUMMY, N9, M1)

70 CALL NULL(DUMMY(N1), N9, M3)

90 N2 = N9
300 CONTINUE
CALL NULL(DUMMY(N2), N9, M3)

300 CONTINUE
CALL NULL(DUMMY(N3), N9, M3)

300 CONTINUE
CALL NULL(DUMMY(N4), N9, M3)

400 CONTINUE
CALL AXPY(R(DUMMY, DUMMY(LU), N9, M1), N9, NA(1), DUMMY(LNB), DUMMY(LV), N9(BAR0.42A, 1), M, NB(1), C, NC(1), EPSA, EPSB, NFAIL)
A-39
GO TO 600

500 CONTINUE
CALL TRANP(DUMMY, NUM1, DUMMY(LU), NUM2)
CALL EQUATE(DUMMY(LU), NUM2, DUMMY, NUM1)
CALL ATXPXA(DUMMY, DUMMY(LU), C, N, N(1), N(1), EPSA, NFAIL)

600 CONTINUE
IF(NFAIL .EQ. 0) GO TO 700
CALL LNCNT(3)

650 FORMAT(//,, IN BARST, EITHER THE SUBROUTINE ATXPXA OR ATXPXA IS UNABLE TO REDUCE A OR H TO SCHUR FORM *)
RETURN

700 CONTINUE
IF(IOP .NE. 0) CALL PRNT(C, NC, 4H X , 1)
RETURN
END
SUBROUTINE TESTSA(A,NA,ALPHA,DISC,STABLE,IOP,DUMMY)

IMPLICIT REAL*8 (A-H,O-Z)

DIMENSION A(1),DUMMY(1)
DIMENSION NA(2),NDUM1(2),NDUM2(2)
LOGICAL DISC,STABLE

STABLE = .FALSE.

CALL EQUATE(A,NA,DUMMY,NA)
N1 = NA(1)**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(N1),DUMMY,N1)
NEVL = NA(1)
IF( IERR .EQ. 0 ) GO TO 200
CALL LNCT(N)
PRINT 100,IERR
100 FORMAT(///,' IN TESTSA, THE ','IS',' EIGENVALUE OF A HAS NOT BEEN FOUND AFTER 30 ITERATIONS','/)
RETURN

200 CONTINUE
NDUM1(1) = NEVL
NDUM1(2) = 1
CALL JUXTC(DUMMY(N1),NDUM1,DUMMY(N2),NDUM1,DUMMY,NDUM2)
IF( DISC ) GO TO 400
DO 300 I = 1,NEVL
IF( DUMMY(I) .GE. ALPHA ) GO TO 600
300 CONTINUE
GO TO 550
400 CONTINUE
N = NDUM2(1)*NDUM2(2)+1
DO 500 I = 1,NEVL
K = I + NEVL
L = N + I = 1
DUMMY(L) = DSQRT((DUMMY(I)**2)+(DUMMY(K)**2))
500 CONTINUE
IF( DUMMY(L) .GE. ALPHA ) GO TO 600
550 CONTINUE
STABLE = .TRUE.
600 CONTINUE
IF( IOP .EQ. 0 ) RETURN
CALL LNCT(N)
PRINT 700
700 FORMAT(///,' PROGRAM TO TEST THE RELATIVE STABILITY OF THE MATRIX','/)
CALL PRNT(A,NA,4H A','/)
CALL LNCT(N)
PRINT 750
750 FORMAT(///,' EIGENVALUES OF A ','/)
CALL PRNT(DUMMY,NDUM2,4HEVLA,1)
IF( .NOT. DISC ) GO TO 850
CALL LNCT(N)
PRINT 800
800 FORMAT(///,' MODULI OF EIGENVALUES OF A ','/)
CALL PRNT(DUMMY,N,4HMODA,1)
850 CONTINUE
CALL LNCT(N)
IF( STABLE) PRINT 900, ALPHA
IF( NOT, STABLE) PRINT 950, ALPHA
900 FORMAT(//, 'MATRIX A IS STABLE RELATIVE TO ', E16.8,/) 
950 FORMAT(//, 'MATRIX A IS UNSTABLE RELATIVE TO ', E16.8,/) 
C RETURN
END
SUBROUTINE EXPSER(A, NA, EXPA, NEXPA, T, IOP, DUMMY)
IMPLICIT REAL*8 (A-H, O-Z)
DIMENSION A(1), EXPA(1), DUMMY(1)
DIMENSION NA(2), NEXPA(2)
COMMON/COND/SUMCV, PICTCV, SERCV, MAXSUM
N = NA(1)
L = (N+2) + 1
TT = T
NEXPA(1) = NA(1)
NEXPA(2) = NA(2)
CALL MAXEL(A, NA, ANAA)
ANAA = ANAA*TT
ANAA = DABS(ANAA)
IF( ANAA .GE. 1.E-15 ) GO TO 100
CALL UNITY(EXPA, NEXPA)
GO TO 800
100 CONTINUE
IOP = 2
CALL NORMS(N, N, NA, IOP, ZERO)
ZERO = ZERO/(2.**47)
CALL TRCE(A, NA, TR)
TR = TR/N
DO 200 I = 1, N
M = N*(I-1)
A(M) = A(M) - TR
200 CONTINUE
IOP = 1
CALL NORMS(N, N, NA, IOP, COL)
CALL NORMS(N, N, NA, IOP, ROW)
ANORM = ROW
IF( ANORM .GT. COL ) ANORM = COL
TMAX = 1./ANORM
K = 0
300 CONTINUE
IF( TMAX = TT ) 325, 350, 350
325 CONTINUE
K = K + 1
TT = T/(2**K)
IF( K = 1000 ) 300, 700, 700
350 CONTINUE
SC = TT
CALL SCALE(A, NA, NA, NA, TT)
CALL UNITY(EXPA, NEXPA)
II = 2
CALL ADD(A, NA, EXPA, NEXPA, DUMMY, NA)
CALL EQUATE(A, NA, DUMMY(L), NA)
400 CONTINUE
CALL MULT(A, NA, DUMMY(L), NA, EXPA, NEXPA)
S = 1./II
CALL SCALE(EXPA, NEXPA, DUMMY(L), NA, S)
CALL ADD(DUMMY(L), NA, DUMMY, NA, EXPA, NEXPA)
CALL MAXEL(DUMMY, NA, TOT)
CALL MAXEL(DUMMY(L), NA, DELT)
IF( TOT .GT. 1.0 ) GO TO 500
IF( DELT/TOT .LT. SERCV ) GO TO 600
GO TO 550
500 CONTINUE
IF( DELT .LT. SERCV ) GO TO 600
550 CONTINUE
CALL EQUATE(EXPA,NEXPA,DUMMY,NA)
555 II = II + 1
GO TO 400

600 CONTINUE
IF( K ) 625,675,650

625 CONTINUE
CALL LNCNT(1)
PRINT 635

635 FORMAT( ' ERROR IN EXPSER, K IS NEGATIVE ' )
RETURN

650 CONTINUE
DO 660 I = 1,K
655 TT = 2*TT
CALL EQUATE(EXPA,NEXPA,DUMMY,NA)
CALL EQUATE(DUMMY,NA,DUMMY(L),NA)
CALL MULT(DUMMY(L),NA,DUMMY,EXPA)

660 CONTINUE
T = TT

675 CONTINUE
S = 1./SC
CALL SCALE(A,NA,A,NA,S)

700 CONTINUE
GO TO 800

750 FORMAT( ' ERROR IN EXPSER, K = 1000 ' )
RETURN

800 CONTINUE
IF( IOP .EQ. 0 ) RETURN
CALL LNCNT(4)
PRINT 825

825 FORMAT( ' COMPUTATION OF THE MATRIX EXPONENTIAL EXP(A T) BY THE ' )
1ERIES METHOD ' )
CALL PRNT(A,NA,4H A ,1)
CALL LNCNT(3)
PRINT 850,T

850 FORMAT( ' T = ',D16.9/)
CALL PRNT(EXPA,NEXPA,4HEXPA,1)
RETURN
END

A-44
SUBROUTINE EXPAOE (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
1 S = 0.
DO 5 J=1, N
2 S = S + DABS(A(I,J))
5 CONTINUE
C **** CALCULATE ACCURACY ESTIMATE
17 DIGC = 24.*DFLOAT(N)
18 IF (ANORM .GT. 1.) DIGC = DIGC*ANORM
19 SDIGC = DIGC
20 DIGC = 15 - IFFX(ALOG10(SDIGC) + .5)
21 C DETERMINE POWER OF TWO AND NORMALIZATION FACTOR
22 C **** M = 0
23 IF (ANORM .LE. 1.) GO TO 27
24 FACTOR = 2.
25 DO 26 M=1,46
26 IF (ANORM .LE. FACTOR) GO TO 20
27 FACTOR = FACTOR*2.
28 15 CONTINUE
29 GO TO 125.
30 20 CONTINUE
31 C **** NORMALIZE MATRIX
32 C **** DO 25 I=1,N
33 DO 25 J=1, N
34 A(I,J) = A(I,J) / FACTOR
25 CONTINUE
27 CONTINUE.
C **** SET COEFFICIENTS FOR (9,9) PADE TABLE ENTRY
C **** C(1) = .5
43 C(1) = 1.7647058823520-01
44 C(2) = 1.1756862745980-02
45 C(3) = 1.71568627450980-03
46 C(4) = 2.2549019607840-04
47 C(5) = 6.28456510809450-06
48 C(6) = 2.4408753860510-07
49 C(7) = 5.1011080422450-09
50 C(8) = 5.66789782476050-11
51 C(9) = 2.24408753860510-07
C **** CALCULATE PADE NUMERATOR AND DENOMINATOR BY COLUMNS
53 C **** DO 95 J=1, N
54 NP1 = N+1
55 NP7 = N+7
56 DO 95 J=1, N
57 C **** COMPUTE JTH COLUMN OF FIRST NINE POWERS OF A
58 DO 35 I=1, N
59 S = 0.
60 35 CONTINUE
ON 30 CONTINUE
WK(1,NP1) = S
ON 35 CONTINUE
DO 45 K=NP1,NP7
KP1 = K+1
DO 45 I=1,N
S = 0.
DO 40 L=1,N
S = S + A(I,L)*WK(L,K)
WK(I,KP1) = S
CONTINUE
40 CONTINUE
WK(I,KP1) = S
CONTINUE
45 CONTINUE
C **** COLLECT TERMS FOR JTH COLUMN OF NUMERATOR AND DENOMINATOR
DO 85 I = 1,N
S = 0.
DO 65 L = 1,N
K = N+1-L
KN1 = K-N+1
P = C(KN1)*WK(I,K)
S = S + P
IEO = MOD(KN1,2)
IF (IEO.EQ.0) GO TO 55
U = U - P
GO TO 65
55 CONTINUE
U = U + P
CONTINUE
65 CONTINUE
P = C(1)*A(I,J)
S = S + P
U = U - P
IF (I.NE.J) GO TO 80
S = S + 1.
U = U + 1.
80 CONTINUE
EA(I,J) = S
WK(I,J) = U
85 CONTINUE
C **** CALCULATE NORMALIZED EXP(A) BY WK * EXP(A) = EA
CALL GAUSEL (MAX,N,WK,N,EA,IERR)
IF (IERR.NE.0) GO TO 130
IF (M.EQ.0) GO TO 130
C **** TAKE OUT EFFECT OF NORMALIZATION ON EXP(A)
DO 120 K=1,M
DO 110 I=1,N
S = 0.
DO 105 J=1,N
S = S + EA(I,L)*EA(L,J)
WK(I,J) = S
105 CONTINUE
CONTINUE
110 CONTINUE
DO 115 I=1,N
115 CONTINUE
DO 115 J=1,N
A = 46
115 DO 115 J=1,N

76      EA(I,J) = WK(I,J)
7  115 CONTINUE
8  120 CONTINUE
29. C ****
   0 C    UN-NORMALIZE A
   1 C ****
   29 C   CONTINUE EXPO1290
   30 C   CONTINUE EXPO1300
   32 C   CONTINUE EXPO1310
   33 C   CONTINUE EXPO1320
   34 C   CONTINUE EXPO1330
   35 C   CONTINUE EXPO1340
   36 C   CONTINUE EXPO1350
   37 C   CONTINUE EXPO1360
   38 C   CONTINUE EXPO1370
   39 C   CONTINUE EXPO1380
   40 C   CONTINUE EXPO1390
   41 C   CONTINUE EXPO1400
   42 C   CONTINUE EXPO1410
   43 C   CONTINUE EXPO1420
   44 C   CONTINUE EXPO1430
   45 C   CONTINUE EXPO1440
   46 C   CONTINUE EXPO1450
   47 C   CONTINUE EXPO1460
   48 C   RETURN EXPO1470
   49 C END EXPO1480
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

IOPT = 1

CALL NORMS(N, N, A, IOPT, COL)

IOPT = 3

CALL NORMS(N, N, A, IOPT, ROW)

AANAA = COL

IF(AANAA .GT. ROW) AANAA = ROW

TMAX = 1./AANAA

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

150 CONTINUE

SC = TT

CALL SCALE(A, NA, A, NA, TT)

CALL UNITY(S, 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

46 IF(DELT/TOT .LT. SERCV) GO TO 400

47 GO TO 350

300 CONTINUE

49 IF(DELT .LT. SERCV) GO TO 400

50 350 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

56 GO TO 200

57 CALL EQUATE(DUMMY, NB, B, NB)

59 IF(K .EQ. 245, 500, 450

425 CONTINUE

CALL LNCNT(1)
PRINT 435
FORMAT ("ERROR IN EXPINT, K IS NEGATIVE")
RETURN

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)
CONTINUE

T = TT

500 CONTINUE
CALL EQUATE(DUMMY(L), NC, C, NC)
CALL MULT(DUMY(L), NA, NA, NA, S)
S = 1./SC
CALL SCALE(A, NA, A, NA, S)

IF (IOP .EQ. 0) RETURN
CALL LNCNT(5)
PRINT 550
FORMAT ("COMPUTATION OF THE MATRIX EXPONENTIAL EXP(A'T), 7, IND ITS INTEGRAL OVER (0, T) BY THE SERIES METHOD ",/)
CALL PRNT(A, NA, 4H A, 1)
CALL LNCNT(3)
PRINT 575, T
FORMAT ("T = ", D16.8, ")
CALL PRNT(B, NB, A,HEXPA, 1)
CALL PRNT(C, NC, 4MINT , 1)
RETURN

600 CONTINUE
CALL LNCNT(1)
PRINT 650
FORMAT ("ERROR IN EXPINT, K = 1000 ")
RETURN
END
SUBROUTINE VARANC( A, NA, G, NG, Q, NO, M, NM, IDENT, DISC, IOPT, DUMMY )

IMPLICIT REAL* ( A-H,O-Z )
DIMENSION A(I),G(1),Q(1),M(1),DUMMY(1)
DIMENSION NA(2),NG(2),NO(2),NM(2),NDUM1(2),IOPT(3),IOPT(2)
LOGICAL IDENT,DISC,SYM
COMMON/TOL/EPSAM,EPSRM,IACM

IF( IOPT(1) .eq. 0 ) GO TO 100
CALL LNCNT(5)
IF( DISC ) PRINT 25
IF( NOT. DISC ) PRINT 35

25 FORMAT( /, ' PROGRAM TO SOLVE FOR THE STEADY-STATE VARIANCE MATRIX'), VAR0011
1 FORMAT( /, ' FOR A LINEAR DISCRETE SYSTEM'), VAR0012
35 FORMAT( /, ' FOR A LINEAR CONTINUOUS SYSTEM'), VAR0013
4 CALL PRNT(A,NA,4H A , 1)
5 IF( NOT. IDENT ) GO TO 55
6 CALL LNCNT(3)
7 PRINT 45
8 GO TO 65
9
55 CONTINUE
10 CALL PRNT(G,NG,4H G , 1)
11 65 CONTINUE
12 IF( NOT. IDENT ) GO TO 85
13 CALL LNCNT(3)
14 PRINT 75
15 75 FORMAT( /, ' INTENSITY MATRIX FOR COVARIANCE OF PROCESS NOISE '), VAR0027
16 85 CONTINUE
17 CALL PRNT(G,NO,4H G , 1)
18 CALL LNCNT(3)
19 PRINT 75
20 GO TO 100
21
100 CONTINUE
22 IF( IDENT ) GO TO 200
23 CALL MULT(G,NG,Q,NG,DUMMY,Q)
24 N1 = NG(1)*NG(2) + 1
25 CALL TRNP(G,NG,DUMMY(N1),NDUM1)
26 CALL MULT(DUMMY,NG,DUMMY(N1),NDUM1,G,NG)
27 IF( IOPT(1) .eq. 0 ) GO TO 200
28 CALL LNCNT(3)
29 PRINT 75
30 GO TO 100
31
200 CONTINUE
32 IF( NOT. DISC ) CALL SCALE(N,NW,W,NW,-1.0)
33 IOPT(1) = IOPT(2)
34 IOPT(2) = 1
35 SYM = .TRUE.
36 IF( DISC ) GO TO 300
37 IF( IOPT(3) .eq. 0 ) GO TO 250
38 CALL BILIN(A,NA,A,NA,A,NA,M,1,IOPT,BETA,SYM,DUMMY)
39 GO TO 400
40
300 CONTINUE
41 CALL RARSTW(A,NA,A,NA,A,NA,A,NA,M,1,IOPT,SYM,EPSA,EPSA,DUMMY)
42 GO TO 400
43
250 CONTINUE
44 CALL EQUATE(A,NA,DUMMY,NA)
45 N = NA(1)**2
46 N1 = N + 1
47 CALL TRNP(A,NA,DUMMY(N1),NA) A-50
63       CALL SUM(DUMMY,NA,W,NW,DUMMY(N1),NA,I OPT,SYM,DUMMY(N2))
64       C
65       400 CONTINUE
66       IF( IOP(1),EQ, 0 ) RETURN
67       CALL LNCNT(3)
68       PRINT 450
69       450 FORMAT(  /, ' VARIANCE MATRIX ', /)
70       CALL PRNT(W,N4,4H W , 1)
71       C
72       RETURN
73       END
SUBROUTINE CTROL (A, NA, N9, C, NC, IOP, IAC, IRANK, DUMMY)

IMPLICIT REAL*8 (A-, D-Z)

DIMENSION A(1), B(1), C(1), DUMMY(1)

DIMENSION NA(2), NB(2), NC(2), NV(2), IOP(5)

N = NA(1) * NB(2)
N1 = N + 1
N2 = N + N
K = NA(1) + 1
J = 1

CALL EQUATE (B, NB, DUMMY(N2), NV)
CALL EQUATE (B, NB, DUMMY, NB)
CONTINUE
CALL MULT (A, NA, DUMMY(N2), NV, DUMMY(N1), NB, C, NC)

IF (J .EQ. K) GO TO 200
CALL EQUATE (C, NC, DUMMY(N2), NV)
J = J + 1
GO TO 100

CALL MULT (A, NA, DUMMY(N2), NV, DUMMY(N1), NB, C, NC)

IF (IOP(1) .EQ. 0) GO TO 300
CALL PRNT (A, NA, 4H A, 1)
CALL PRNT (B, NB, 4H B, 1)
CALL LNCNT (4)
PRINT 250
250 FORMAT (//, ' THE MATRIX C IS THE CONTROLLABILITY MATRIX FOR THE ', 15X, 'PAIR')
CALL PRNT (C, NC, 4H C, 1)

IF (IOP(2) .EQ. 0) RETURN
NOS = 0
ILOPT = 2
K = NC(2)
NC(2) = NB(2) * (NA(2) - NB(2) + 1)
N = NC(1) * NC(2)
CALL TRANP (C, NC, DUMMY, NV)
NC(2) = K
N1 = N + 1
N2 = N1 + NV(2)
CALL SNVDEC (ILOPT, NV(1), NV(2), NV(1), NV(2), DUMMY, NOS, B, IAC, ZTEST, DUM)
1MY(N1), DUMMY(N2), IRANK, A, IERR)
IF (IERR .EQ. 0) GO TO 340
CALL LNCNT (5)
IF (IERR .GT. 0) PRINT 310, IERR
IF (IERR .EQ. -1) PRINT 320, ZTEST, IRANK
310 FORMAT (//, ' BASED ON THE ZERO-TEST ', 15X, ' THE RANK OF THE CONTR')
320 FORMAT (//, ' IN CTROL, SNVDEC HAS FAILED TO CONVERGE TO THE ', 15X, ' SINGULAR VALUE AFTER 30 ITERATIONS ')

340 IF (IOP(3) .EQ. 0) RETURN
N1 = N + 1
N2 = N1 + NV(2)
CALL SNVDEC (ILOPT, NV(1), NV(2), NV(1), NV(2), DUMMY, NOS, B, IAC, ZTEST, DUM)
1MY(N1), DUMMY(N2), IRANK, A, IERR)
IF (IERR .EQ. 0) GO TO 340
CALL LNCNT (5)
IF (IERR .GT. 0) PRINT 310, IERR
IF (IERR .EQ. -1) PRINT 320, ZTEST, IRANK
310 FORMAT (//, ' IN CTROL, SNVDEC HAS FAILED TO CONVERGE TO THE ', 15X, ' SINGULAR VALUE AFTER 30 ITERATIONS ')
320 FORMAT (//, ' IN CTROL, THE MATRIX SUBMITTED TO SNVDEC USING ZTEST = ', 15X, ' ITS CLOSE TO A MATRIX WHICH IS OF LOWER RANK ', 15X, ' IF THE CURRENT RANK'
330 FORMAT (//, ' ACCURACY IS REDUCED THE RANK MAY ALSO BE REDUCED', 15X, ' CURRENT RANK')
340 FORMAT (//, ' BASED ON THE ZERO-TEST ', 15X, ' THE RANK OF THE CONTR')
10 A P L I L I T Y M A T R I X I S ' , / 4 , / ' T H E S I N G U L A R V A L U E S A R E * , / )

CALL PRNT(DUMMY(N1),NV,IOPT,3)
CALL EQUATE(DUMMY(N2),NA,DUMMY,NA)
N1 = N +1
N2 = N1 + N
CALL MULT(A,NA,DUMMY,NA,DUMMY(N1),NA)
CALL TRNP(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,NA,DUMMY(N1),NB)

CALL PRNT(DUMMY(N2),NA,IOPT,3)
CALL LNCNT(2)
CALL PRNT(DUMMY(N1),NB,IOPT,3)
CALL LNCNT(2)
PRINT 520
CALL PRNT(DUMMY,NA,IOPT,3)
RETURN
END
SUBROUTINE TRNSIT(A, NA, B, NB, H, NH, G, NG, F, NF, V, NV, T, NX, DISC, STABLE

1     IE, IOP, DUMMY)
2     IMPLICIT REAL*8 (A-H, O-Z)
3     DIMENSION A(1), B(1), H(1), G(1), F(1), V(1), T(1), IOP(4)
4     DIMENSION NA(2), NB(2), NH(2), NG(2), NF(2), NV(2), NX(2), T(2), IOP(4)
5     DIMENSION NDUM1(2), NDUM2(2)

6     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

14 C

CALL LNCNT(4)

16     IF(DISC) PRINT 100
17     IF(.NOT. DISC) PRINT 120
18 100 FORMAT(//, ' COMPUTATION OF TRANSIENT RESPONSE FOR THE DIGITAL SYSTEM')
19 120 FORMAT(//, ' COMPUTATION OF TRANSIENT RESPONSE FOR THE CONTINUOUS SYSTEM')

22     CALL PRNT(A, NA, 4H A , 1)
23     CALL PRNT(B, NB, 4H B , 1)
24     IF((IOP(1) .NE. 1) .AND. (IOP(1) .NE. 0)) GO TO 180
25     CALL LNCNT(3)
26     IF(IOP(1) .EQ. 0) PRINT 140
27     IF(IOP(1) .EQ. 1) PRINT 160
28 140 FORMAT(//' H IS A NULL MATRIX ')
29 160 FORMAT(//' H IS AN IDENTITY MATRIX ')

30 180 CONTINUE

32     CALL PRNT(H, NH, 4H H , 1)
33 200 CONTINUE
34     IF((IOP(2) .NE. 1) .AND. (IOP(2) .NE. 0)) GO TO 260
35     CALL LNCNT(3)
36     IF(IOP(2) .EQ. 0) PRINT 220
37     IF(IOP(2) .EQ. 1) PRINT 240
38 220 FORMAT(//' G IS A NULL MATRIX ')
39 240 FORMAT(//' G IS AN IDENTITY MATRIX ')

40 260 CONTINUE

42 280 CONTINUE

44     CALL PRNT(G, NG, 4H G , 1)
45 295 CONTINUE
46     IF((IOP(3) .NE. 0) .AND. (IOP(3) .NE. 1)) GO TO 285
47     CALL LNCNT(3)
48     IF(IOP(3) .EQ. 0) PRINT 285
49     IF(IOP(3) .EQ. 1) PRINT 290
50 285 FORMAT(//' V IS A NULL MATRIX ')
51 290 FORMAT(//' V IS AN IDENTITY MATRIX ')

52 295 CONTINUE

54 300 CONTINUE

56     CALL EQUATE(A, NA, DUMMY(N6), NA)
57     CALL MULT(R, NB, F, NF, DUMMY, NA)
58     CALL SUBT(A, NA, DUMMY, NA, 4A, NA)
59 C

60     IF(DISC) GO TO 350
61     NMAX = T(1)/T(2)
62     ILOPT = 1
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),NB)
IF(IOP(3).NE.1) GO TO 325
CALL EQUATE(DUMMY(N2),NB,DUMMY(N1),NX)

325 CONTINUE
CALL MULT(DUMMY(N2),NB,V,NV,DUMMY(N1),NX)

350 CONTINUE
NMAX = IOP(4)
IF(IOP(3).NE.0) GO TO 400

500 CONTINUE
IF(IOP(2).EQ.0) GO TO 600
IF(IOP(2).EQ.1) GO TO 525
CALL MULT(G,NG,DUMMY(N2),NV,DUMMY(N4),NDUM1)
GO TO 525

600 CONTINUE
IF(IOP(1).NE.0) CALL EQUATE(DUMMY(N2),NV,DUMMY(N4),NDUM1)

700 CONTINUE
CALL EQUATE(X,NX,DUMMY(N5),NDUM1)

800 CONTINUE
CALL MULT(H,NH,X,NX,DUMMY(N5),NDUM1)
GO TO 575

900 CONTINUE
CALL SCALE(F,NF,F,NF,1.0)

1000 CONTINUE
IF(KSGT,NMAX) GO TO 800
CALL MULT(F,NF,XNX,DUMMY(N2),NV)
IF(IOP(3).NE.0) CALL ADD(DUMMY(N2),NV,V,NV,DUMMY(N2),NV)
CALL MULT(DUMMY,NA,XNX,DUMMY(N3),NX)
IF(IOP(3).EQ.0) GO TO 475
CALL ADD(DUMMY(N1),NX,DUMMY(N3),NX,DUMMY(N3),NX)

1100 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

1200 CONTINUE
CALL EQUATE(X,NX,DUMMY(N5),NDUM1)

1300 CONTINUE
IF(IOP(2).EQ.0) GO TO 600
IF(IOP(2).EQ.1) GO TO 700
CALL ADD(DUMMY(N4),NDUM1,DUMMY(N5),NDUM1,DUMMY(N4),NDUM1)

1500 CONTINUE
CALL SCALE(U,NU,U,NU,1.0)

1600 CONTINUE
IF(IOP(1).NE.0) CALL EQUATE(DUMMY(N5),NDUM1,DUMMY(N4),NDUM1)
CALL LNCNT(5)
IF(.NOT., DISC) GO TO 720
PRINT 710, K
GO TO 740
710 FORMAT(////,15)
720 CONTINUE
730 FORMAT(////, IS)_ TRNO'
GO TO 740 --
TRN01
PRINT 730, TIME
740 _CONTINUE
750 CONTINUE
CALL TRANP(X0N, DUO MMY(N5), N DUM2)
CALL PRNT(DUMMY(NS), NDUM2, 3)
760 CONTINUE
IF(.NOT., DISC) GO TO 860
CALL UNITY(DUMMY(N1), NA)
CALL ST(DUMMY(N1), NA, A, NA)
860 CONTINUE
880 CONTINUE
IF(.NOT., DISC) CALL SCALE(DUMMY, NX, DU MMY, NX, -1.0)
LNCNT(3)
PRINT 880
CALL LNCNT(5)
PRINT 890
FORMAT(////, * STEADY-STATE VALUE OF X TRANSPOSE*)
CALL TRANP(DUMMY, NX, DUMMY(N5), NDUM2)
CALL PRNT(DUMMY(N5), NDUM2, L, 3)
890 CONTINUE
CALL EQUATE(DUMMY(N3), NX, X, NX)
891 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(8, NOpV, NV, DUMMY, NX)
GO TO 840
820 CONTINUE
CALL EQUATE(8, N8, DUMMY, NX)
840 CONTINUE
IF(.NOT., DISC) PRINT 865
IF( DISC ) PRINT 870
718 865 FORMAT(////, IN TRANSIT, THE MATRIX A-BF SUBMITTED TO GELIM IS SINGULAR*)
719 870 FORMAT(////, IN TRANSIT, THE MATRIX I - (A-BF) SUBMITTED TO GELIM IS SINGULAR*)
GO TO 900
880 CONTINUE
890 CONTINUE
CALL EQUATE(DUMMY(N6), NA, A, NA)
RETURN A-56
SUBROUTINE SAMPL(A,NA,R,NB,Q,NG,R,NW,T,IOPT,DUMMY)  
IMPLICIT REAL*8 (A-H, O-Z)  
DIMENSION A(1), R(1), Q(1), R(1), W(1), DUMMY(1)  
DIMENSION NA(2), NB(2), NG(2), R(2), NW(2), IOPT(2), NDUM(2)  
COMMON/CONV/SUMCV,RICTCV,SERCV,MAXSUM  
IF( IOPT(1) .EQ. 0 ) GO TO 100  
IF( IOPT(2) .EQ. 0 ) GO TO 50  
CALL LNCNT(5)  
PRINT 25  
25 FORMAT(//,' COMPUTATION OF WEIGHTING MATRICES FOR THE OPTIMAL SAMPS')  
CALL PRNT(A,NA,' A')  
CALL PRNT(B,NB,' B')  
CALL LNCNT(3)  
PRINT 35  
35 FORMAT(//,' CONTINUOUS PERFORMANCE INDEX WEIGHTING MATRICES')  
CALL PRNT(Q, NA, ' Q')  
CALL LNCNT(3)  
PRINT 45  
45 FORMAT(//,' T = ',10.8)  
GO TO 100  
50 CONTINUE  
CALL LNCNT(8)  
PRINT 75  
75 FORMAT(//,' COMPUTATION OF THE RECONSTRUCTIBILITY GRAMIAN')  
CALL PRNT(A,NA, ' A')  
CALL PRNT(Q, NA, ' Q')  
CALL LNCNT(3)  
PRINT 85  
85 FORMAT(//,' T = ',10.8)  
GO TO 100  
100 CONTINUE  
41 L = ( N**2)  
42 N1 = L + 1  
43 N2 = N1 + L  
44 TT = T  
45 C  
46 IOPT = 1  
47 CALL NORMS(N,N,N,A,IOPT,ANORM)  
48 IOPT = 3  
49 CALL NORMS(N,N,N,A,IOPT,ROWA)  
50 IF( ANORM .GT. ROWA ) ANORM = ROWA  
51 IF( ANORM .LE. 1.E-15 ) GO TO 900  
52 IF( TMAX = TT ) 150, 150, 200  
53 TMAX = 1.0/ANORM  
54 K = 0  
55 C  
57 125 CONTINUE  
58 IF( TMAX = TT ) 150, 150, 200  
59 150 CONTINUE  
60 K = K + 1  
61 TT = T/ ( 2**K)  
62 IF( K = 1000 ) 125, 800, 800
63 C _
64 200 CONTINUE
65 C _
66 I = 0
67 SC = TT
68 CALL SCALE(A, NA, A, NA, TT)
69 CALL SCALE(Q, NO, Q, NO, TT)
70 CALL EQUATE(Q, NO, DUMMY, NO)
71 C _
72 IF (IOP(2) .NE. 0) GO TO 500
73 C _
74 225 CONTINUE
75 II = I + 2
76 I = I + 1
77 F = 1.0/II
78 CALL SCALE(A, NA, DUMMY(N1), NA, F)
79 CALL MULT(DUMMY, NA, DUMMY(N1), NA, DUMMY(N2), NA)
80 CALL TRANP(DUMMY(N2), NA, DUMMY(N1), NA)
81 CALL ADD(DUMMY(N1), NA, DUMMY(N2), NA, DUMMY, NA)
82 C _
83 CALL MAXEL(Q, NO, TOT)
84 CALL MAXEL(DUMMY, NA, DELT).
85 IF (TOT .GT., 1.0) GO TO 250
86 IF (DELT/TOT .LT. SERCV) GO TO 300
87 GO TO 275
88 250 CONTINUE
89 IF (DELT .LT. SERCV) GO TO 300
90 275 CONTINUE
91 CALL ADD(Q, NO, DUMMY, NA, G, NO)
92 GO TO 225
93 C _
94 300 CONTINUE
95 C _
96 IF (K, EQ, 0) GO TO 400
97 N3 = N2 + L
98 G = 1.0
99 IOPT = 0
100 CALL EXPSER(A, NA, DUMMY, NA, G, IOPT, DUMMY(N1))
101 C _
102 350 CONTINUE
103 IF (K, EQ, 0) GO TO 400
104 K = K-1
105 C _
106 CALL TRANP(DUMMY, NA, DUMMY(N1), NA)
107 CALL MULT(G, NO, DUMMY, NA, DUMMY(N2), NA)
108 CALL MULT(DUMMY(N1), NA, DUMMY(N2), NA, DUMMY(N3), NA)
109 CALL ADD(G, NO, DUMMY(N3), NA, G, NO)
110 CALL MULT(DUMMY, NA, DUMMY, NA, DUMMY(N1), NA)
111 CALL EQUATE(DUMMY(N1), NA, DUMMY, NA)
112 C _
113 GO TO 350
114 C _
115 400 CONTINUE
116 S = 1.0/SC
117 CALL SCALE(A, NA, A, NA, S)
118 C _
119 IF (IOP(1) .EQ. 0) RETURN
120 CALL PRNT(G, NO, 4HGRAM, 1)
121 RETURN
122 C _
123 500 CONTINUE
124 CALL SCALE(B, NB, B, NB, TT)
125 N3 = N2 + L
126        N4 = N3 + L
127        N5 = N4 + L
128        N6 = N5 + L
129.C      CONTINUE
130 525 CONTINUE
131 II = I + 2
132 I = I + 1
133 F = 1.0/II
134 CALL SCALE(A,NA,DUMMY(N1),NA,F)
135 CALL TRAPN(DUMMY(N1),NA,DUMMY(N2),NA)
136 CALL MULT(DUMMY,NA,DUMMY(N1),NA,DUMMY(N3),NA)
137 CALL TRAPN(DUMMY(N3),NA,DUMMY(N1),NA)
138 CALL MULT(DUMMY,NA,B,NB,DUMMY(N5),NW)
139 CALL ADD(DUMMY(N1),NA,DUMMY(N3),NA,DUMMY,NA)
140 CALL SCALE(DUMMY(N5),NW,DUMMY(N1),NW,F)
141 IF (I .NE. 1 ) GO TO 550
142 CALL EQUATE(DUMMY(N1),NW,W,NW)
143 CALL EQUATE(DUMMY(N1),NW,DUMMY(N6),NW)
144 CALL ADD(Q,NO,DUMMY,Q,Q)
145 GO TO 525
146 C
147 550 CONTINUE
148 CALL MULT(DUMMY(N2),NA,DUMMY(N6),NW,DUMMY(N5),NW)
149 CALL ADD(DUMMY(N5),NW,DUMMY(N1),NW,DUMMY(N1),NW)
150 CALL TRAPN(B,NB,DUMMY(N2),NDUM)
151 CALL SCALE(DUMMY(N2),NDUM,DUMMY(N2),NDUM,F)
152 CALL MULT(DUMMY(N2),NDUM,DUMMY(N6),NW,DUMMY(N3),NR)
153 CALL TRAPN(DUMMY(N3),NR,DUMMY(N5),NR)
154 CALL ADD(DUMMY(N3),NR,DUMMY(N5),NR,DUMMY(N3),NR)
155 CALL EQUATE(DUMMY(N1),NW,DUMMY(N6),NW)
156 IF (I .NE. 2 ) GO TO 575
157 CALL ADD(Q,NO,DUMMY,Q,Q)
158 CALL ADD(W,NW,DUMMY(N1),NW,W,NW)
159 CALL EQUATE(DUMMY(N3),NR,DUMMY(N4),NR)
160 GO TO 525
161 C
162 575 CONTINUE
163 CALL MAXEL(Q,NQ,TOT)
164 CALL MAXEL(DUMMY,NQ,DELT)
165 IF (TOT .GT. 1.0 ) GO TO 580
166 IF (DELT/TOT .LT. SERCV ) GO TO 585
167 GO TO 595
168 C
169 580 CONTINUE
170 IF (DELT .LT. SERCV ) GO TO 585
171 GO TO 595
172 C
173 585 CONTINUE
174 CALL MAXEL(DUMMY(N4),NR,TOT)
175 CALL MAXEL(DUMMY(N3),NR,DELT)
176 IF (TOT .GT. 1.0 ) GO TO 590
177 IF (DELT/TOT .LT. SERCV ) GO TO 600
178 GO TO 595
179 C
180 590 CONTINUE
181 IF (DELT .LT. SERCV ) GO TO 600
182 C
183 595 CONTINUE
184 CALL ADD(Q,NQ,DUMMY,NQ,Q,NQ)
185 CALL ADD(W,NW,DUMMY(N1),NW,W,NW)
186 CALL ADD(DUMMY(N4),NR,DUMMY(N3),NR,DUMMY(N4),NR)
187 GO TO 525
188 C
252. 925 CONTINUE
253 CALL MULT(U, NO, B, NB, W, NW)
254 CALL SCALE(W, NW, W, NW, T)
255 CALL TRANP(B, NB, DUMMY, NDUM)
256 CALL MULT(DUMMY, NDUM, W, NW, DUMMY(N1), NP)
257 TT = T/3.
258 CALL SCALE(DUMMY(N1), NR, DUMMY, NR, TT)
259 CALL SCALE(R, NR, P, NN, T)
260 CALL ADD(R, NR, DUMMY, NR, R, NP)
261 IF (IOP(1) .EQ. 0) RETURN
262 CALL LNCNT(3)
263 PRINT 750
264 CALL PRNT(Q, NO, O, O, 1);
265 CALL PRNT(W, NW, O, W, 1);
266 CALL PRNT(R, NR, O, R, 1)
267 RETURN
268 C
269 END
SUBROUTINE PREFIL(A,NH,NB,NQ,M,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),NF(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 ELIMINATE CROSS-
             PRODUCT TERM IN QUADRATIC PERFORMANCE INDEX,'
             '/)

  IF(IOP(3).EQ.0)GO TO 50
 CALL PRNT(A,NH,4H A ,1)
 CALL PRNT(B,NB,4H B ,1)
 CALL PRNT(Q,NQ,4H Q ,1)
 CALL PRNT(W,NW,4H W ,1)
 CALL PRNT(R,NR,4H R ,1)

       CONTINUE

 10 CALL TRNAP(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(4,M,DUMMY,NF(2),F,IOPT,IFAC,DETERM,ISCALE,DUMMY(N1),I)

 1PR

 27 IF(IERR .EQ. 0 )GO TO 200
 CALL LNCNT(4)
 PRINT 150

 30 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)

 39 C

 40 IF(IOP(3).EQ.0)GO TO 400
 CALL MULT(B,NB,F,NF,DUMMY,NA)
 CALL SUBT(A,NA,DUMMY,NA,A,NA)

 44 C

 45 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)

 53 C

 54 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)

IMPLICIT REAL*8(A-N, O-Z)
DIMENSION A(1), B(1), F(1), DUMMY(1)
DIMENSION NA(2), NB(2), NF(2), IOP(3), NDUM(2)
DIMENSION IOPT(2)
LOGICAL SYM
COMMON/TOL/EPSA,EPSB,IACM
N = NA(1)*2
N = N + 1

IF(IOP(2) .EQ. 0) GO TO 100
CALL EQUATE(A, NA, DUMMY, NA)
N2 = N1 + NA(1)
N3 = N2 + NA(1)
ISV = 0
ILV = 0
CALL EIGEN(NA(1), NA(1), DUMMY, DUMMY(N1), DUMMY(N2), ISV, ILV, DUMMY(N1), IERR)

M = NA(1)
IF(IERR .EQ. 0) GO TO 50
CALL LNCNT(3)
PRINT 25, IERR
20 CONTINUE
25 FORMAT(/, ' IN CSTAB, THE SUBROUTINE EIGEN FAILED TO DETERMINE THE EIGENVALUE FOR THE MATRIX A AFTER 30 ITERATIONS. ')
21 IERR = 1
22 CALL NORMS(M, M, A, IERR, BETA)
23 CALL SCALEM(A, NA, M, BETA)
24 CALL SCALE(A, A, M, 1.0)
NAX = NA(I)
DO 225 I = 1, N
J = N + I - 1
BETA = DARS(DUMMY(J))
IF(BETA .GT. BETA) BETA = BETA1
75 CONTINUE
BETA = SCLE*(BETA + .001)
GO TO 200
50 CONTINUE
30 BETA = 0.0
DO 75 I = 1, M
J = N1 + I - 1
BETA = DARS(DUMMY(J))
IF(BETA .GT. BETA) BETA = BETA1
75 CONTINUE
BETA = SCLE*(BETA + .001)
GO TO 200
100 CONTINUE
100 BETA = SCLE
200 CONTINUE

CALL TRASP(B, NB, DUMMY, NDUM)
CALL MULT(B, NB, DUMMY, NDUM, DUMMY(N1), NA)
CALL SCALE(C, DUMMY(N1), NA, DUMMY, NA, -2.0)
CALL SCALE(A, NA, DUMMY(N1), NA, -1.0)
J = -NA(1)
NAX = NA(1)
DO 225 I = 1, NAX
J = J + NAX + 1
K = N1 + J - 1
DUMMY(K) = DUMMY(K) - BETA
225 CONTINUE
N2 = N1 + N
SYM = .TRUE.
IOPT(1) = 0

IF(IOP(3) .NE. 0) GO TO 300
EPSA = EPSA
CALL BARSTW(DUMMY(N1), NA, A, NA, DUMMY, NA, IOPT, SYM, EPSA, EPSA, DUMMY(N2), IACM)
300 IACM
A-63
GO TO 350

GO TO 350

CALL BILIN(DUMMY(N1),NA,A,NA,DUMMY,NA,IOP,ASCLE,SYM,DUMMY(N2))

350 CONTINUE

CALL EQUATE(B,NB,DUMMY(N1),NB)

IOPT(1) = 3

IAC = IACM

N3 = N2 + NA(1)

CALL SNVDEC(IOPT,NA(1),NA(1),NA(1),NA(1),DUMMY,NB(2),DUMMY(N1),IACC)

IF(IOERR .EQ. 0 ) GO TO 400

CALL LNCNT(5)

IF(IOERR .GT. 0 ) PRINT 360, IERR

IF(IOERR .EQ. -1) PRINT 370, ZTEST, IRANK

360 FORMAT(//' IN CSTAB, SNVDEC HAS FAILED TO CONVERGE TO THE '*14/',

ISINGULAR VALUE AFTER 30 ITERATIONS'//' )

370 FORMAT(//' IN CSTAB, THE MATRIX SUBMITTED TO SNVDEC USING ZTEST =

1 'D16.B' IS CLOSE TO A MATRIX OF LOWER RANK '//' IF THE ACCURACY IS REDUCED THE RANK MAY ALSO BE REDUCED '//' CURRENT RANK

3 = ',14')

IF( IERR .GT. 0 ) RETURN

NDUM(1) = NA(1)

NDUM(2) = 1

CALL PRNT(DUMMY(N2),NDUM,4HSGVL,1)

400 CONTINUE

CALL TRANP(DUMMY(N1),NB,F,NF)

IF( IOP(1) .EQ. 0 ) RETURN

CALL LNCNT(4)

PRINT 500

500 FORMAT(//' COMPUTATION OF F MATRIX SUCH THAT A-BF IS ASYMPTOTICALLY

STABLE IN THE CONTINUOUS SENSE '//' )

CALL PRNT(A,NA,4H A ,'1)

CALL LNCNT(4)

PRINT 550,BETA

550 FORMAT(//' BETA = ',E16.8,'/)

CALL PRNT(B,NB,4H B ,'1)

CALL PRNT(F,NF,4H F ,'1)

CALL MULT(B,NB,F,NF,DUMMY,NA)

CALL SURT(A,NA,DUMMY,NA,DUMMY,NA)

CALL PRNT(DUMMY,NA,4HA-BF,1)

N2 = N1+NA(1)

N3 = N2+NA(1)

ISV = 0

ILV = 0

CALL EIGEN(NA(1),NA(1),DUMMY,DUMMY(N1),DUMMY(N2),ISV,ILV,V,DUMMY(N3))

13), IERR)

12

13)

IF( IERR .EQ. 0 ) GO TO 600

M = NA(1)+IERR

CALL LNCNT(3)

PRINT 25, IERR

600 CONTINUE

CALL LNCNT(4)

PRINT 650

650 FORMAT(//' EIGENVALUES OF A-BF '//' )

675 FORMAT(10X,2016.8)

CALL LNCNT(M)

DO 700 I=1,M

700 I = N1+I-1

K = N2+I-1
 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 

 IF( .NOT. SING ) GO TO 100 

 IOP(1)=IOP(1) 

 IOP(2) = 1 

 IOP(3) = 0 

 CSCE=1.05 

 CALL CSTAB(A,NA,B,NB,F,NF,IOP,CSCLE,DUMMY) 

 CALL MULT(B,NB,F,DUMMY,NA) 

 CALL EQUATE(DUMMY,NA,DUMMY(N1),NA) 

 GO TO 200 

 100 CONTINUE 

 CALL EQUATE(A,NA,DUMMY,NA) 

 CALL EQUATE(A,NA,DUMMY(N1),NA) 

 CALL LNCNT(3) 

 IF( IERR .EQ. 0 ) GO TO 250 

 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", 
 37 '15, '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 

 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 

 300 CONTINUE 

 ALPHA = SCLE 

 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 

 A-66
DUMMY(J) = DUMMY(J) + ALPHA

CALL EQUATE(B,NA,DUMMY(N2),NB)
N3 = N2 + NA(1)*NB(2)
NRHS = NA(1)+NB(2)
NA = N3 + NA(1)
IFAC = 0
CALL GELIM(NA(1),NA(1),DUMMY,NRHS,DUMMY(N1),DUMMY(N3),IFAC,DUMMY(NOST)
14),IERR)
IF(.IERR .EQ. 0 .) GO TO 500
CALL LNCNT(3)
IF(.NOT. SING ) GO TO 445
PRINT 435
FORMAT(///,*,IN DSTAB, GELIM HAS FOUND THE MATRIX A-8G )
1 SINGULAR *)
CALL PRINT(A,NA,A,1)
CALL PRINT(F,4M G ,1)
GO TO 465
CONTINUE
CALL LNCNT(3)
PRINT 455
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.
CALL MULT(B,NB,F,NF,DUMMY(N1),NA)
CALL MULT(A,NA,DUMMY(N1),NA)
CALL SCALE(A,NA,DUMMY(N1),NA)
IF(.NOT. SING ) GO TO 600
CALL LNCNT(3)
PRINT 650,ZTEST
625 FORMAT(///,*,IN DSTAB, THE MATRIX SUBMITTED TO SNVDEC, USING ZTEST
SINGULAR VALUE AFTER 30 ITERATIONS*)
IAC is reduced the rank may also be reduced. If the accuracy of IAC is reduced, the rank may also be reduced, and the current rank decrease.

```
IERR = IERR + 1
IF ( IERR .GT. 0 ) RETURN

NDUM(1) = NA(1)
NDUM(2) = 1
CALL PRINT(DUMMY(N2),NDUM,4MSVL,1)

700 CONTINUE
CALL TRAPN(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

CALL EQUATE(DUMMY,NF,F,NF)
```

```
1000 FORMAT(//,* COMPUTATION OF F SUCH THAT A-BF IS ASYMPTOTICALLY STABLE IN THE DISCRETE SENSE*,//)
CALL PRINT(A,NA,4M A,1)
CALL PRINT(B,NB,4M B,1)
CALL LNCNT(4)
PRINT 1000
```

```
1100 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(I),NA(I),DUMMY,DUMMY,DUMMY(N0),DUM1),ISV,ISV,V,DUMMY(N1)
IF ( IERR .EQ. 0 ) GO TO 1300
CALL LNCNT(3)
PRINT 1250
```

```
1250 FORMAT(//,* IN OSTA8, THE PROGRAM EIGEN FAILED TO DETERMINE THE EIGENVALUE FOR THE A-BF MATRIX AFTER 30 ITERATIONS*)
NDUM(1) = NA(1) - IERR
```

```
1300 CONTINUE
CALL JUXT(DUMMY(N1),NDUM,DUMMY(N2),NDUM,DUMMY,NDUM1)
CALL PRINT(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) = DSORT(DUMMY(J)**2 + DUMMY(K)**2)
1500 CONTINUE
CALL PRINT(DUMMY,NDUM,4MMOD,1)
```

```
SUBROUTINE DISREG(A,NA,B,NB,H,NH,G,NO,R,NF,P,NP,IOP,IDENT,DUM DISO
IMPLICIT REAL*8(A,H,G=2)
DIMENSION A(1),B(1),H(1),R(1),F(1),P(1),DUMMY(1)
DIMENSION NA(2),NR(2),NO(2),NR(2),NF(2),NP(2)
DIMENSION H(1),NM(2),NOU(2)
LOGICAL IDENT
COMMON/TOL/EPSA,EPSRM,IACM
COMMON/CONV/SUMCV,RICTCV,CRVCV,MAXSUM
N = NA(1)**2
N1 = N + 1
N2 = N1 + N
N3 = N2 + N
KSS = 0.
I = IOP(3)
IF(IOP(1) .EQ. 0) GO TO 85
CALL LNCNT(3)
PRINT 25
25 FORMAT('://" PROGRAM TO SOLVE THE TIME-INVARIANT FINITE-DURATION OPTIMUM'/'
ITIMAL":" DIGITAL REGULATOR PROBLEM WITH NOISE-FREE MEASUREMENTS')
26 CALL PRNT(A,NA,4H A_1)
27 CALL PRNT(B,NB,4H B_1)
28 CALL PRNT(G,NO,4H G_1)
29 PRINT 35
35 FORMAT(//,'H IS AN IDENTITY MATRIX',/)
31 GO TO 65
32 CONTINUE
33 CALL PRNT(H,NH,4H H_1)
34 CALL MULT(Q,NO,H,NH,DUMMY,NH)
35 CALL TRANP(H,NH,DUMMY(N1),NF)
36 CALL MULT(DUMMY(N1),NF,DUMMY,NH,G,NO)
37 CALL LNCNT(3)
38 PRINT 55
55 FORMAT(//,' MATRIX ( H TRANSPOSE )',/)
40 CALL PRNT(Q,NO,4H HTGH,1)
41 CONTINUE
42 CALL PRNT(R,NR,4H R_1)
43 CALL LNCNT(4)
44 PRINT 75
75 FORMAT(//' WEIGHTING ON TERMINAL VALUE OF STATE VECTOR',/)
46 CALL PRNT(P,NP,4H P_1)
47 CONTINUE
48 IF(IOP(1) .NE. 0) OR IDENT) GO TO 100
49 CALL MULT(Q,NO,H,NH,DUMMY,NH)
50 CALL TRANP(H,NH,DUMMY(N1),NF)
51 CALL MULT(DUMMY(N1),NF,DUMMY,NH,G,NO)
52 CALL LNCNT(3)
53 CONTINUE
54 CALL EQUATE(P,NP,DU4MP,NP)
55 CALL MULT(P,NP,A,NA,DUMMY(N1),NA)
56 CALL TRANP(B,NH,DUMMY(N2),NF)
57 CALL MULT(DUMMY(N2),NF,DUMMY(N1),NA,F,NF)
58 CALL MULT(P,NP,B,NA,DUMMY(N1),NR)
59 CALL MULT(DUMMY(N2),NF,DUMMY(N1),NB,DUMMY(N3),NR)
60 CALL ADD(R,NR,DUMMY(N3),NR,DUMMY(N1),NR)
61 CALL ADD(R,NR,DUMMY(N3),NR,DUMMY(N1),NR)
IOP = 3
IAC = IACM
NF = NP(1)
CALL SVODEC(IOP, NF, NF, NF, DUMMY(N1), NF(2), F, IAC, ZTEST, DUMMY(N2))
I = DUMMY(N3), IRANK, APLUS, IERR)
IF (IERR .EQ. 0) GO TO 300
CALL LNCNT(5)
IF (IERR .EQ. -1) PRNT 250, ZTEST, IRANK
200 FORMAT(///, ' IN DISREG, SVODEC HAS FAILED TO CONVERGE TO THE 4, DISR
1 SINGULARVALUE AFTER 30 ITERATIONS', ///)
250 FORMAT(///, ' IN DISREG, THE MATRIX SUBMITTED TO SVODEC USING ZTEST DISR
1 = ', '016.8,' IS CLOSE TO A MATRIX OF LOWER RANK', ///, 'IF THE ACCURACY DISR
2 IAC IS REDUCED THE RANK MAY ALSO BE REDUCED', ///, 'CURRENT RANK = ' DISR
3, 'I4')
78 IF (IERR .GT. 0) RETURN
79 NDUM(1) = NA(1)
80 NDUM(2) = 1
81 CALL PRNT(DUMMY(N2), NDUM, 4H3GVL, 1)
82 C
83 300 CONTINUE
84 CALL MULT(9, NA, F, NF, DUMMY(N1), NF)
85 CALL TRANP(F, NF, DUMMY(N2), NA)
86 CALL MULT(DUMMY(N2), NA, DUMMY(N1), NF, P, NP)
87 CALL ADD(9, NG, P, NP, P, NP)
88 CALL MULT(9, NG, F, NF, DUMMY(N1), NA)
89 CALL SUBT(A, NA, DUMMY(N1), NA, DUMMY(N1), NA)
90 CALL MULT(DUMMY, NA, DUMMY(N1), NA, DUMMY(N2), NA)
91 CALL TRANP(DUMMY(N1), NA, DUMMY(N3), NA)
92 CALL MULT(DUMMY(N3), NA, DUMMY(N2), NA, DUMMY(N1), NA)
93 CALL ADD(P, NP, DUMMY(N1), NA, P, NP)
94 C
95 IF (IOP(2) .EQ. 0) GO TO 400
96 CALL LNCNT(5)
97 PRINT 350, I
98 350 FORMAT(///, ' STAGE ', 'IS, ///)
99 CALL PRNT(F, NF, 4H F, 1)
100 CALL PRNT(P, NP, 4H P, 1)
101 C
102 400 CONTINUE
103 IF (I .EQ. 0) GO TO 600
104 CALL MAXEL(DUMMY, NP, ANORM1)
105 CALL SUBT(DUMMY, NP, P, NP, DUMMY(N2), NP)
106 CALL MAXEL(DUMMY(N2), NP, ANORM2)
107 IF (ANORM1 .NE. 0.0) GO TO 500
108 GO TO 100
109 C
110 500 CONTINUE
111 IF (ANORM1 .GT. 1.0) GO TO 550
112 IF (ANORM2/ANORM1 .LT. RICTCV) KSS = 1
113 GO TO 575
114 550 CONTINUE
115 IF (ANORM2 .LT. RICTCV) KSS = 1
116 575 CONTINUE
117 IF (KSS .EQ. 1) GO TO 600
118 GO TO 100
119 C
120 600 CONTINUE
121 K = IOP(1) + IOP(2)
122 IF (K .EQ. 0) RETURN
123 IF (KSS .EQ. 0) GO TO 700
124 CALL LNCNT(4)
125 PRINT 650
STEADY-STATE SOLUTION HAS BEEN REACHED IN DISREG.

CONTINUE

IF( IOP(2) .NE. 0 ) RETURN

IF( IOP(1) .EQ. 0 ) RETURN

CALL LNCNT(3)

I = IOP(3) - I

PRINT 800, I

FORMAT(//,* F AND P AFTER *,15, * STEPS,/) CALL PRNT(F,NF,4H F,1)

CALL PRNT(P,NP,4H P,1)

RETURN

END
SUBROUTINE CNTREG(A,NA,NB,NH,Q,NQ,R,NR,NZ,W,LAMBDAS,F,NF,P,NP

DIMENSION A(1),B(1),H(1),Q(1),R(1),Z(1),W,LAMBDAS(1),F(1),P(1)

DIMENSION NA(2),NB(2),NH(2),Q(2),R(2),NF(2),P(2),IOP(3),NDUM1(2)

DIMENSION NA(2),NB(2),NH(2),Q(2),R(2),NF(2),P(2),IOP(3),NDUM1(2)

COMMON/CONDV/SUMCV,RCTCV,SENCR,MAXSUM

IF( IOP(1) .EQ. 0 ) GO TO 65

CALL LNCNT(5)

FORMAT(//, 'PROGRAM TO SOLVE THE TIME-INVARIANT INFINITE-DURATION CONTINUOUS OPTIMAL REGULATOR PROBLEM WITH NOISE-FREE MEASUREMENTS')

IF( IOP(3) .NE. 0 ) PRINT 30

FORMAT(//, 'PROGRAM TO SOLVE THE TIME-INVARIANT INFINITE-DURATION CONTINUOUS OPTIMAL REGULATOR PROBLEM WITH NOISE-FREE MEASUREMENTS')

CALL PRNT(A,N,4H A

CALL PRNT(B,N,4H B

CALL PRNT(Q,N,4H A

IF( .NOT. IDENT ) GO TO 45

CALL LNCNT(3)

FORMAT(///, 'H IS AN IDENTITY MATRIX')

GO TO 55

CONTINUE

CALL PRNT(H,N,4H H

CALL MULT(Q,N,4H Q

CALL TRANP(H,N,4H H

CALL MULT(DUMMY(N1),NDUM1,DUMMY,NQ)

CALL LNCNT(3)

PRINT 50

FORMAT(///, 'MATRIX (H TRANSPOSE)QH')

CALL PRNT(Q,N,4H Q

CONTINUE

CALL PRNT(R,N,4H R

IF( IOP(3) .NE. 0 ) GO TO 65

CALL LNCNT(4)

FORMAT(///, 'WEIGHTING ON TERMINAL VALUE OF STATE VECTOR')

CALL PRNT(P,N,4H P

CONTINUE

CALL EQUATE(R,N,4H R

N = NA(1)**2

N1 = NA(1)*NB(2)+1

CALL TRANP(R,N,4H R

L = NR(1)

IFAC = 0

CALL SYMPS(L,L,DUMMY,NB(1),DUMMY(N1),IOP,IFAC,DET,ISCALE,DUMMY(NC)

IF( IER .EQ. n ) GO TO 70
CALL LNCNT(4)  
PRINT 75  
FORMAT(// IN CNTREG, THE SUBROUTINE SYMPOS HAS FOUND THE  
1 R NOT SYMMETRIC POSITIVE DEFINITE/)  
RETURN  
C  
100 CONTINUE  
CALL EQUAT(DUMMY(N1),NDUM1,DUMMY,NDUM1)  
CALL MOLT(BPN8,DUMMY(N1),NDUM1,NDUM1,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 MOLT(Q,NQ,NH,DUMMY(N2),NH)  
CALL TRAPN(H,NH,DUMMY(N3),NDUM1)  
CALL MOLT(DUMMY(N31,NDUM1,DUMMY(N2),NH,N1)  
CALL SCALE(Q,NQ,N1,-1.0)  
200 CONTINUE  
CALL SCALE(Q,NQ,Q,NQ,-1.0)  
CALL JUXTR(A,NQ,Q,Z,N1)  
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)  
NDUM2(1) = 2*N(1)  
NDUM2(2) = NDUM2(1)  
IF( IOP(1) .NE. 0 ) CALL PRNT(Z,NDUM2,4H Z ,1)  
CALL EQUAT(Z,NDUM2,DUMMY(N1),NDUM2)  
M = 4*N  
N2 = N + N1  
L = 2*N(1)  
N3 = N2 + L  
N4 = N3 + L  
ISV = L  
ILV = 0  
CALL EIGEN(L,L,DUMMY(N1),NDUM2,DUMMY(N2),NDUM2,NDUM3,ISV,ILV,L,DUMMY(N4),ICNT)  
100 IF( IERR .EQ. 0 ) GO TO 300  
101 CALL LNCNT(4)  
102 IF( IERR .GT. 0 ) GO TO 250  
103 PRINT 225,IERR  
225 FOR MAT(//' IN CNTREG, EIGEN VALUE OF Z HAS NOT BEEN FO  
L0  
110 UND AFTER 30 ITERATIONS IN EIGEN/)  
111 RETURN  
12 C  
300 CONTINUE  
IF( IOP(1) .EQ. 0 ) GO TO 400  
13 C  
325 PRINT 325  
325 FOR MAT(//' EIGENVALUES OF Z/)  
14 NDUM1(1) = L  
15 NDUM1(2) = 2  
16 CALL PRNT(DUMMY(N2),NDUM1,N1)  
17 CALL LNCNT(3)  
18 PRINT 350  
19 350 FOR MAT(//' CORRESPONDING EIGENVECTORS/)  
20 CALL PRNT(W,NDUM2,3)  
A-74
126 CONTINUE
127 CALL EQUATE(W,NOUM2,DUMMY(N1),NOUM2)
128 J1 = 1
129 J2 = 1
130 M = 2*N
131 NOUM1(1) = L
132 NOUM1(2) = 1
133 K4 = N4
134 C
135 I=1
136 CONTINUE
137 IF( I .GT. L ) GO TO 515
138 K1 = N2+I-1
139 K2 = N1+(I-1)*L
140 K3 = N3+I-1
141 IF(DUMMY(K1) .GT. 0.0) GO TO 425
142 J = (J1-1)*L+M+1
143 J1 = J1+1
144 IF(DUMMY(K3) .NE. 0.0) J1=J1+1
145 GO TO 450
146 CONTINUE
147 DUMMY(K4) = I
148 K4 = K4+1
149 J = (J2-1)*L+1
150 J2 = J2+1
151 IF( DUMMY(K3) .NE. 0.0 ) J2 = J2 + 1
152 CONTINUE
153 CALL EQUATE(DUMMY(K2),NOUM1,W(J),NOUM1)
154 IF(DUMMY(K3) .EQ. 0.0) GO TO 500
155 I = I+1
156 K2 = K2+L
157 J = J+L
158 CALL EQUATE(DUMMY(K2),NOUM1,W(J),NOUM1)
159 CONTINUE
160 I=I+1
161 GO TO 415
162 CONTINUE
163 CALL NULL(LAMBDA,NA)
164 K0 = -1
165 J = -NA(1)
166 NAX = NA(1)
167 I=1
168 CONTINUE
169 CONTINUE
170 IF( I .GT. NAX ) GO TO 530
171 J = NAX + J + 1
172 K0 = K0 + 1
173 K1 = N4 + K0
174 K2 = DUMMY(K1)
175 K = N2+K2=1
176 LAMBDA(J) = DUMMY(K)
177 K3 = N3+K2=1
178 IF( DUMMY(K3) .NE. 0.0 ) GO TO 525
179 K4 = J+1
180 LAMBDA(K4) = DUMMY(K3)
181 K4 = K4+NAX
182 LAMBDA(K4) = DUMMY(K)
183 K4 = K4=1
184 LAMBDA(K4) = DUMMY(K3)
185 K5 = 4 + (I-1)*L + 1
186 K6 = K5 + L
187 CALL EQUATE(W(K5),NOUM1,DUMMY(N1),NOUM1)
188 CALL EQUATE(W(K6),NOUM1,W(K5),NOUM1)
CALL EQUATE(DUMMY(N1),NDUM1,W(N6),NDUM1)  
I = I+1
J = MAX + J +1
CONTINUE
I = I+1
GO TO 520
CONTINUE
IF( IOP(1) .EQ. 0 ) GO TO 700
CALL LNCNT(3)
PRINT 535
FORMAT(/' REORDERED EIGENVECTORS'/)  
CALL PRNT(W,NDUM2,0,3)
CALL LNCNT(4)
PRINT 545
FORMAT(/' LAMBDA MATRIX OF EIGENVALUES OF Z WITH POSITIVE REAL P')
1_0RTS'/)
CALL PRNT(LAMBDA,NA,0,3)
CALL MULT(Z,NDUM2,W,NDUM2,DUMMY(N1),NDUM2)  
L = NDUM2(1)
M = L+2
N2 = N1+N
CALL EQUATE(W,NDUM2,DUMMY(N2),NDUM2)
N3 = N2+N
N4 = N3+L
IFAC = 0
CALL GELIM(L,L,DUMMY,N2,L,DUMMY(N1),DUMMY(N3),IFAC,DUMMY(N4),IERR)
IF( IERR .EQ. 0 ) GO TO 600
CALL LNCNT(4)
PRINT 550
FORMAT(/' IN CNTREG, GELIM HAS FOUND THE REORDERED MATRIX W TO B')
CONTINUE
IF( IOP(1) .EQ. 0 ) GO TO 800
CALL PRNT(DUMMY(N1),NDUM2,4HW1Z,1)  
709 CONTINUE
NDUM1(1) = 2*NA(1)
NDUM1(2) = NA(1)
N2 = 2*N + N1
CALL TRAMP(W,NDUM1,DUMMY(N2),NDUM2)
NW11 = N1
NDUM1(1) = NA(1)
CALL TRAMP(DUMMY(N1),NDUM1,DUMMY(NW11),NDUM1)
L = 2*N+1
NW12 = NW11+N
CALL TRAMP(DUMMY(L),NDUM1,DUMMY(NW12),NDUM1)
CALL TRAMP(DUMMY(L),NDUM1,DUMMY(NW22),NDUM1)
CONTINUE
IF( IOP(1) .EQ. 0 ) GO TO 800
CALL PRNT(DUMMY(NW11),NA,4HW11,1)
CALL PRNT(DUMMY(NW21),NA,4HW21,1)
CALL PRNT(DUMMY(NW12),NA,4HW12,1)
CALL PPNT(DUMMY(NW22),NA,4H22,1)

400 CONTINUE

IF (IOP(3) .NE. 0) GO TO 900

N2 = N1+4*N

CALL MULT(P,NP,DUMMY(NW12),NA,3,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

FORMAT(//' IN CNTREG, GELIM HAS FOUND THE MATRIX W21 = P1W11 TO BE SINGULAR')

RETURN

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

FORMAT(//' IN CNTREG, GELIM HAS FOUND THE MATRIX W21 TO BE SINGULAR')

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 PPNT(P,NP,4H21,1)

CALL PPNT(F,NF,4H21)

RETURN

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

A-77
L5
N5
N6 = N5+N
N7 = NA+NA(1)
KSS = 0
NDUM1(1) = NA(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)
FORMAT(/' EXP(-LAMBDA * ,D16.8,*')
CALL PRNT(DUMMY(N2),NA,0,3)
CALL LNCNT(3)
IF( NMAX .LE. 0 ) RETURN
CALL EQUATE(5,NA,DUMMY(N3),NA)
TIME = 1*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(NW1),NA,DUMMY(N4),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(NW2),NA,DUMMY(N3),NA,DUMMY(N4),NA)
CALL TRANP(DUMMY(NW2),NA,DUMMY(N3),NA,DUMMY(N4),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
FORMAT(/' IN CNTREG AT TIME ,*D16.8,*' P CANNOT BE COMPUTED DUE T
10 MATRIX SINGULARITY IN GELIM*)
RETURN
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
IF( I .NE. NMAX ) CALL PRNT(F,NF,4H F ,1)
GO TO 1400
IF( IOP(2) .EQ. 0 ) GO TO 1400
CALL LNCNT(5)
PRINT 1350,TIME
FORMAT(/' TIME = ,*D16.8/)
CALL PRNT(P,NP,4H P ,1)
IF( I .NE. NMAX ) CALL PRNT(F,NF,4H F ,1)
GO TO 1400
GO TO 1400
378 IF(KSS, EQ, 1) GO TO 1500
379 I = I - 1
380 IF(I .GE. 0) GO TO 1100
381 GO TO 1600
382 1500 CONTINUE
383 CALL LNCNT(4)
384 PRINT 1550
385 1550 FORMAT(/* STEADY-STATE SOLUTION HAS BEEN REACHED IN CNTREG/*)
386 C
387 1600 CONTINUE
388 IF(IOP(2) .NE. 0) RETURN
389 IF(IOP(1) .EQ. 0) RETURN
390 CALL LNCNT(5)
391 PRINT 1350 .TIME
392 CALL PRNT(P,NP,4H P, 1)
393 CALL PRNT(F,NF,4H F, 1)
394 C
395 RETURN
396 END
SUBROUTINE RICHT(\*A, NA, B, N9, H, NH, Q, NQ, R, NR, F, NF, P, NP, IOJ, IDENT, Di RICCO
1 1SC,FNULL,DUMMY) RICCO
2 IMPLICIT REAL*8 (A-H,N9) RICCO
3 DIMENSION A(1),8(1),1(1),P(1),F(1),P(1),DUMMY(1) RICCO
4 DIMENSION NA(2),N8(2),Q(2),NQ(2),R(2),NF(2),NP(2),IOJ(3) RICCO
5 DIMENSION H(1),NH(2),IOJ(2) RICCO
6 LOGICAL IDENT,DISC,FNULL,SYM RICCO
7 COMMON/TOL/EP34M,EP38M,IACM RICCO
8 COMMON/CONV/SUMCV,RICTCV,SERCV,MAXSUM RICCO
9 N = NA(1)**2 RICCO
10 IF(.NOT.IDENT)GO TO 210 RICCO
11 CALL LNCNT(3) RICCO
12 C RICCO
13 N = NA(1)**2 RICCO
14 N1 = N + 1 RICCO
15 C RICCO
16 N2 = N1*N RICCO
17 N3 = N2*N RICCO
18 N4 = N3*N RICCO
19 CALL LNCNT(3) RICCO
20 IF( IOP(1) .EQ. 0 ) GO TO 210 RICCO
21 CALL LNCNT(4) RICCO
22 IF(.NOT. DISC) PRINT 100 RICCO
23 IF(.DISC) PRINT 150 RICCO
24 100 FORMAT(/,' PROGRAM TO SOLVE CONTINUOUS STEADY-STATE RICCATI EQUATRICOr/) RICCO
25 150 FORMAT(/,' PROGRAM TO SOLVE DISCRETE STEADY-STATE RICCATI EQUATRICOr/) RICCO
26 CALL PRNT(A,NA,4H A,1) RICCO
27 CALL PRNT(B,N8,4H B,1) RICCO
28 CALL PRNT(Q,NQ,4H Q,1) RICCO
29 IF(.NOT. IDENT)GO TO 185 RICCO
30 CALL LNCNT(3) RICCO
31 PRINT 190 RICCO
32 190 FORMAT(/,' H IS AN IDENTITY MATRIX',/) RICCO
33 GO TO 200 RICCO
34 185 CONTINUE RICCO
35 CALL PRNT(H,NH,4H H,1) RICCO
36 CALL MULT(Q,NQ,H,NH,DUMMY,NH) RICCO
37 CALL TRANP(H,NH,DUMMY(N2),NP) RICCO
38 CALL MULT(DUMMY(N2),NP,DUMMY,NH,A,NO) RICCO
39 CALL LNCNT(3) RICCO
40 CALL PRNT(F,NF,4H F,1) RICCO
41 CALL TRANP(H,NH,DUtAMY(N2),NP) RICCO
42 CALL MULT(DUMMY(N2),rJP,DUMMY,NH2O) RICCO
43 195 FORMAT(/,' MATRIX (H transpo3E) 0H ',/) RICCO
44 CALL PRNT(Q,NQ,4HMTQH,1) RICCO
45 200 CONTINUE RICCO
46 CALL PRNT(R,NR,4H R,1) RICCO
47 IF( FNULL ) GO TO 210 RICCO
48 CALL LNCNT(3) RICCO
49 PRINT 205 RICCO
50 205 FORMAT(/,' INITIAL F MATRIC',/) RICCO
51 CALL PRNT(F,NF,4H F,1) RICCO
52 CALL TRANP(B,NB,P,NP) A-80 RICCO
53 210 CONTINUE RICCO
54 IF((IOP(1) .NE. 0) .OR. IDENT) GO TO 220 RICCO
55 CALL MULT(Q,NQ,H,NH,DUMMY,NH) RICCO
56 CALL TRANP(H,NH,DUMMY(N2),NP) RICCO
57 CALL MULT(DUMMY(N2),NP,DUMMY,NH,Q,NQ) RICCO
58 220 CONTINUE RICCO
59 CALL TRANP(B,NB,P,NP) RICCO
60 IF(DISC) GO TO 900 RICCO
61 CALL TRANP(B,NB,P,NP) RICCO
62 CALL TRANP(B,NB,P,NP) RICCO
CALL EQUATE(R,NR,DUMMY,NR)
CALL SYMPOS(NR(1),NR(1),DUMMY,NP(2),P,IOPT,IOPT,DET,ISCALE,DUMMY(NRIC)
11),IERR)
IF(IERR.EQ.0) GO TO 250
CALL LNCNT(3)
PRINT 225
FORMAT(*,IN RICNWT, A MATRIX WHICH IS NOT SYMMETRIC POSITIVE DER
IFINFINITE HAS BEEN SUBMITTED TO SYMPOS,*)
RETURN
CONTINUE
CALL EQUATE(P,NP,DUMMY,NF)
CALL MULT(B,NB,DUMMY,NP,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)
IF(FNULL) GO TO 300
CALL MULT(R,NR,F,NF,DUMMY(N3),NF)
CALL TRANP(F,NF,P,NP)
CALL MOLT(P,NP,OUMMY(N3),NFR0UMMY(N4),NA)
CALL TRANP(DUMMY(N4),NA,DUMMY(N3),NA)
CALL ADD(OUMMY(N3),NA,DUMMY(N2),NA)
CALL SCALE(P,NP,P,NP,-1.0)
GO TO 350
CALL TRAPN(A,NA,QUMMY(N2),NA)
CALL SCALE(Q,NQ,P,NP,-1.0)
CALL EQUATE(DUMMY(N3),NA,DUMMY(N2),NA,DUMMY(N2),NA)
CALL TRANP(DUMMY(N2),NA,DUMMY(N3),NA)
CALL ADD(DUMMY(N2),NA,DUMMY(N2),NA)
CALL SCALE(DUMMY(N3),NA,DUMMY(N3),NA,0.5)
CALL ADD(DUMMY(N3),NA,0,NQ,P,NP)
CALL SCALE(P,NP,P,NP,-1.0)
GO TO 350
CALL TRAPN(A,NA,DUMMY(N2),NA)
CALL SCALE(Q,NQ,P,NP,-1.0)
CALL EQUATE(P,NP,DUMMY,NF)
CALL MULT(B,NB,DUMMY,NP,DUMMY(N1),NA)
CALL TRANP(DUMMY(N1),NA,DUMMY(N2),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 MOLT(P,NP,OUMMY(N3),NFR0UMMY(N4),NA)
CALL TRANP(DUMMY(N4),NA,DUMMY(N3),NA)
CALL ADD(OUMMY(N3),NA,DUMMY(N2),NA)
CALL SCALE(P,NP,P,NP,-1.0)
GO TO 350
CALL BARS(B,NB,P,NP,IOPT,SYM,SYMPEPSA)
GO TO 450
IF(IOP(2).NE.0) GO TO 400
CALL LINR(3)
PRINT 500,1
FORMAT(*,IPRINTNATION 15,/) 
CALL PRNT(P,NP,4,P,1)
GO TO 550
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,N)
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)
A-81
CALL SURT(A,NA,DUMMY(N3),NA,DUMMY(N4),NA)
CALL TRAPN(DUMMY(N4),NA,DUMMY(N3),NA)

C     IF(IOP(3) .NE. 0 ) GO TO 650
CALL BARTW(DUMMY(N3),NA,B,NP,NP,IOPT,SYM,EPST,EPST,DUMMY(N4))
GO TO 675

C     CONTINUE
CALL BTLIN(DUMMY(N3),NA,B,NA,P,NP,IOPT,SCLI,SYM,DUMMY(N4))
GO TO 675

C     CONTINUE
C     GO TO 675
CALL HILIN(DUMMY(N3),NA,A1,NP,NP,IOPT,SCLE,SYM,SYM,DUMMY(N4))
CONTINUE

IZI+1
MAXEL(OUTMY(N3),NA,A14)
CALL HILIN(DUMMY(N3),NA,A1,NP,NP,IOPT,SCLE,SYM,DUMMY(N4))
CONTINUE

IF(ANORM2/ANORM1 .LT. RICTCV ) GO TO 800
GO TO 750

C     CONTINUE
IF( ANORM2 .LT. RICTCV ) GO TO 800
CONTINUE

C     CONTINUE
IF( .NOT. FNULL ) GO TO 950
CALL EQUATE(Q,NQ,P,NP)
CALL EQUATE(A,NA,OUTMY(N1),NA)
GO TO 1000

C     CONTINUE
CALL MULT(R,NR,r,NFPDUMMY(N1),NF)
CALL MULT(P,NP,DUMMY(N1),NF,DUMMY(N1),NA)
CALL MULT(R,NR,F,NF,DUMMY(N1),NF)
CALL MULT(P,NP,DUMMY(N1),NF,DUMMY(N1),NA)
CALL MULT(R,NR,F,NF,DUMMY(N1),NA)
CALL MULT(R,NR,F,NF,DUMMY(N1),NA)
CALL TRANP(OUTMY(N1),NA,DUMMY(N2),NA)
CALL TRANP(DUMMY(N2),NA,DUMMY(N1),NA)

C     CONTINUE
CALL MULT(R,NR,F,NF,DUMMY(N1),NF)
CALL TRANP(F,NF,P,NP)
CALL TRANP(F,NF,P,NP)

C     CONTINUE
CALL MULT(P,NP,DUMMY(N1),NF,DUMMY(N2),NA)
CALL TRANP(DUMMY(N2),NA,DUMMY(N1),NA)
CALL MULT(P,NP,DUMMY(N1),NF,DUMMY(N2),NA)
CALL MULT(P,NP,DUMMY(N1),NF,DUMMY(N2),NA)
CALL MULT(P,NP,DUMMY(N1),NF,DUMMY(N2),NA)
CALL MULT(P,NP,DUMMY(N1),NF,DUMMY(N2),NA)
CALL TRANP(DUMMY(N1),NA,DUMMY(N2),NA)

C     CONTINUE
CALL SUM(DUMMY(N2),NA,P,NP,DUMMY(N1),NA,IOPT,SYM,DUMMY(N3))
IF(IOP(2) .EQ. 0 ) GO TO 1100
CALL LNCTN(3)
PRINT 500,1

A-82
CALL PRNT(P,NP,GP,1)  
190 C 
191 1100 CONTINUE 
192 CALL MULT(P,NP,A,NA,DUMMY(N1),NA)  
193 CALL MULT(P,NP,B,NB,DUMMY(N2),NB)  
194 CALL TRANP(B,NB,DUMMY(N3),NF)  
195 CALL MULT(DUMMY(N3),NF,DUMMY(N1),NA,F,NF)  
196 CALL MULT(DUMMY(N3),NF,DUMMY(N2),NB,DUMMY(N1),NR)  
197 CALL TRANP(DUMMY(N1),NR,DUMMY(N2),NR)  
198 CALL ADD(DUMMY(N1),NR,DUMMY(N2),NR,DUMMY(N1),NR)  
199 CALL SCALE(DUMMY(N1,NA,F,IOPT,IOPT,DET,ISCALE,DUMMY)  
200 CALL MAXEL(DUMMY,NA,ANORM1)  
201 CALL SUBT(P,NP,DUMMY,NA,DUMMY(N1),NA)  
202 CALL MAXEL(DUMMY(N1),NA,ANORM2)  
203 IF(ANORM1.GT.1.) GO TO 1269 
204 IF(ANORM2/ANORM1.LT.RICTCV) GO TO 1300 
205 GO TO 1250 
206 CALL LNCNT(3)  
207 IF(I.EQ.1) GO TO 925 
208 CALL LNCNT(3) 
209 PRINT 225 
210 CALL SYMPOS(NR(1),NR(1),DUMMY(N1),NA(1),F,IOPT,IOPT,DET,ISCALE,DUMMY 
211 CALL MAXEL(DUMMY,NA,ANORM1)  
212 CALL MAXEL(DUMMY(N1),NA,ANORM2)  
213 IF(ANORM1.GT.1.) GO TO 1269 
214 IF(ANORM2/ANORM1.LT.RICTCV) GO TO 1300 
215 GO TO 1250 
216 1200 CONTINUE 
217 IF(ANORM2.LT.RICTCV) GO TO 1300 
218 C 
219 1250 CONTINUE 
220 IF(I.EQ.101) GO TO 925 
221 CALL LNCNT(3)  
222 PRINT 775 
223 IOP(1) = 1 
224 C 
225 1300 CONTINUE 
226 IF(IOP(1).EQ.0) RETURN 
227 CALL LNCNT(4) 
228 PRINT 1350,1 
229 1350 FORMAT(//,*FINAL VALUES OF P AND F AFTER*;IS,* ITERATIONS TO CONV 
230 1ERGES/) 
231 CALL PRNT(P,NP,GP,1)  
232 CALL PRNT(F,NF,GF,1)  
233 C 
234 RETURN  
235 END
SUBROUTINE ASMREG(A, NA, B, NB, H, NH, Q, NO, R, NP, F, NF, P, NP, IDENT, DISC, N)

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), H(2), Q(2), R(2), F(2), P(2), IOPT(5), IOPT(3)

LOGICAL IDENT, DISC, NEWT, STABLE, FNULL, SING

N = NA(1) * 2

MI = N + 1

IOPT(0) = 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) = IOPT(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

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 300 CONTINUE

CALL TESTSA(A, NA, ALPHA, DISC, STABLE, IOPTT, DUMMY)

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) = IOPT(1)

IOPT(2) = 1

DSCL = 0.5

ALPHAT = 1. / ALPHA

CALL SCALE(A, NA, A, NA, ALPHAT)

CALL SCALE(B, NB, B, NR, ALPHAT)

CALL DSTAB(A, NA, B, NB, F, NF, SING, IOPT, DSCL, DUMMY)

CALL SCALE(A, NA, A, NA, ALPHA)

CALL SCALE(B, NR, B, NB, ALPHA)

GO TO 225

C 300 CONTINUE

IF ( STABLE ) GO TO 400

A-84
CALL LNCNT(5)

IF( DISC ) GO TO 330

PRINT 310, ALPHA

FORMAT(13,13) IN ASMREG, CSTAB HAS FAILED TO FIND A STABILIZING GAIN

1 MATRIX (F) RELATIVE TO ",", ALPHA = ".016.6/

RETURN

FORMAT(13,13) IN ASMREG, CSTAB HAS FAILED TO FIND A STABILIZING GAIN

1 MATRIX (F) RELATIVE TO ",", ALPHA = ".016.6/

RETURN

CALL RICNTR(A,NA,B,NB,H,NM,Q,NQ,R,NF,P,NP,IOP,IDENT,DISC,FNU)

CALL RICNAT(A,NA,H,B,NB,H,NM,Q,NQ,R,NF,P,NP,IOP,IDENT,DISC,FNU)

CALL RICNAT(A,NA,B,NB,H,NM,Q,NQ,R,NF,P,NP,IOP,IDENT,DISC,FNU)

RETURN

FNUL = .FALSE.

CALL RICNTR(A,NA,B,NB,H,NM,Q,NQ,R,NF,P,NP,IOP,IDENT,DISC,FNU)

CALL RCHTRG(A,NA,B,NB,H,NM,Q,NQ,R,NF,P,NP,IOP,IDENT,DISC,FNU)

CALL RCHTRG(A,NA,B,NB,H,NM,Q,NQ,R,NF,P,NP,IOP,IDENT,DISC,FNU)

GO TO 750

CALL DISREG(A,NA,B,NB,H,NM,Q,NQ,R,NF,P,NP,IOP,IDENT,DUMMY)

GO TO 750

IF( IOP(4) .NE. 0 ) GO TO 1100

N2 = N1 + N

N3 = N2 + N

IF( DISC ) GO TO 800

CALL MULT(P,NP,A,NA,OUMMY(N1),NP)

CALL MULT(P,NP,A,NA,OUMMY(N1),NP)

CALL MULT(P,NP,A,NA,OUMMY(N1),NP)

CALL ADD(DUMMY(N1),NP,DUMMY(N2),NP)

CALL ADD(DUMMY(N1),NP,DUMMY(N2),NP)

CALL ADD(DUMMY(N1),NP,DUMMY(N2),NP)

CALL ADD(DUMMY,NP,DUMMY(NA),NP)

CALL ADD(DUMMY,NP,DUMMY(NA),NP)

CALL ADD(DUMMY,NP,DUMMY(NA),NP)

CALL RCHTRG(A,NA,B,NB,H,NM,Q,NQ,R,NF,P,NP,IOP,IDENT,DUMMY)

GO TO 750

GO TO 900

GO TO 900

GO TO 750

GO TO 750

GO TO 750

GO TO 750

IF( DISC ) GO TO 800

CALL MULT(R,NF,F,NF,DUMMY,NF)

CALL MULT(R,NF,F,NF,DUMMY,NF)

CALL MULT(R,NF,F,NF,DUMMY,NF)

CALL ADD(DUMMY,NF,DUMMY(N1),NA)

CALL ADD(DUMMY,NF,DUMMY(N1),NA)

CALL ADD(DUMMY,NF,DUMMY(N1),NA)

CALL ADD(DUMMY,NF,DUMMY(N1),NA)

CALL ADD(DUMMY,NF,DUMMY(N1),NA)

CALL ADD(DUMMY,NF,DUMMY(N1),NA)

CALL ADD(DUMMY,NF,DUMMY(N1),NA)

CALL ADD(DUMMY,NF,DUMMY(N1),NA)
CALL SUBT(P, NP, DUMMY, NA, DUMMY, NA)

CONTINUE

CALL LNCNT(4)

PRINT 1000

FORMAT(//" RESIDUAL ERROR IN RICCATI EQUATION "/)

CALL PRNT(DUMMY, NP, 4HEROR, 1)

CALL LNCNT(4)

PRINT 1200, IERR

FORMAT(//" IN ASMREG, THE ", IS, ", EIGENVALUE OF P HAS NOT BEEN COMPUTED AFTER 30 ITERATIONS "/)

CALL LNCNT(4)

PRINT 1400, IERR

FORMAT(//" IN ASMREG, THE ", IS, ", EIGENVALUE OF A-BF HAS NOT BEEN COMPUTED AFTER 30 ITERATIONS "/)

CALL LNCNT(4)

PRINT 1600

FORMAT(//" EIGENVALUES OF P "/)

CALL PRNT(DUMMY, NDUM1, 4HEVLP, 1)

CALL LNCNT(4)

PRINT 1700

FORMAT(//" CLOSED-LOOP RESPONSE MATRIX A-BF "/)

CALL PRNT(DUMMY(N1), NA, 4HA-BF, 1)

CALL LNCNT(3)

PRINT 1800

FORMAT(//" EIGENVALUES OF A-BF ")
189 CALL PRINT(' UMMY(N2),NUM3,0,3)
190 RETURN
191 END
SUBROUTINE ASMFIL(A, NA, G, NG, H, NH, Q, NQ, R, NR, F, NF, P, NP, IDENT, DISC, N AS
1  IEWT, STABLE, FNULL, ALPHA, IOP, DUMMY)
2 IMPLICIT REAL*8(A-HO-Z)
3 DIMENSION A(1), G(1), H(1), Q(1), R(1), F(1), P(1), DUMMY(1)
4 DIMENSION NA(2), NG(2), NH(2), NQ(2), NR(2), NF(2), NP(2), IOPT(5), NOUM1(AS
5 12), IOPT(1)
6 LOGICAL IDENT, DISC, NEWT, STABLE, FNULL
7 IF (IOP(1) .EQ. 0) GO TO 100
8 CALL LNCNT(4)
9 IF (DISC) PRINT 15
10 IF (.NOT. DISC) PRINT 25
11 15 FORMAT(//'PROGRAM TO SOLVE THE DISCRETE INFINITE-DURATION OPTIMIZATION',/)
12 1L FILTER PROBLEM',/)
13 25 FORMAT(//'PROGRAM TO SOLVE THE CONTINUOUS INFINITE-DURATION OPTIMIZATION',/)
14 1MAL FILTER PROBLEM',/)
15 CALL PRNT(A, NA, 4M, A, 1)
16 IF (.NOT. IDENT) GO TO 35
17 CALL LNCNT(3)
18 PRINT 30
19 30 FORMAT(//'G IS AN IDENTITY MATRIX',/)
20 GO TO 40
21 35 CONTINUE
22 CALL PRNT(G, NG, 4M G, 1)
23 40 CONTINUE
24 CALL PRNT(H, NH, 4M H, 1)
25 CALL LNCNT(3)
26 PRINT 45
27 45 FORMAT(//'INTENSITY MATRIX FOR COVARIANCE OF MEASUREMENT NOISE',/)
28 CALL PRNT(R, NR, 4R R, 1)
29 C
30 IF (.NOT. IDENT) GO TO 65
31 CALL LNCNT(3)
32 PRINT 55
33 55 FORMAT(//'INTENSITY MATRIX FOR COVARIANCE OF PROCESS NOISE',/)
34 C
35 65 CONTINUE
36 CALL PRNT(Q, NQ, 4H Q, 1)
37 C
38 100 CONTINUE
39 IOPT(1) = IOP(2)
40 IOPT(2) = IOP(3)
41 IOPT(3) = IOP(4)
42 IOPT(4) = IOP(5)
43 IOPT(5) = 0
44 K = 0
45 C
46 200 CONTINUE
47 CALL TRAPN(A, NA, DUMMY, NA)
48 CALL EQUATE(DUMMY, NA, A, NA)
49 CALL TRAPN(H, NH, DUMMY, NOUM1)
50 CALL EQUATE(DUMMY, NOUM1, H, NH)
51 IF (.IDENT) GO TO 250
52 CALL TRAPN(G, NG, DUMMY, NOUM1)
53 CALL EQUATE(DUMMY, NOUM1, G, NG)
54 250 CONTINUE
55 IF (K .EQ. 1) RETURN
56 C
57 K = K + 1
58 CALL ASMREG(A, NA, H, NH, G, NG, Q, NQ, R, NR, F, NF, P, NP, IDENT, DISC, NEWT, ST
59 1ABLE, FNULL, ALPHA, IOP, DUMMY)
60 C
61 N1 = (2 + 2) + 3 * NA(1)
62 CALL TRAPN(F, NF, DUMMY(N1), NOUM1) A-88
CALL EQUIT(E(DUMMY(N1),NDUM1,F,NF))

IF(IOP(1),EQ,0)GO TO 200

IF(IDENT)GO TO 300

CALL LNCNT(3)

PRINT 55

CALL PRNT(Q,NQ,AMGQGT,1)

CONTINUE

CALL LNCNT(3)

PRINT 325

FORMAT(/'FILTER GAIN'/)

CALL PRNT(F,NF,4M F,1)

CALL LNCNT(3)

PRINT 350

FORMAT(/'STEADY-STATE VARIANCE MATRIX OF RECONSTRUCTION ERROR'/)

CALL PPNT(P,NP,4M P,1)

NDUM1(1)=NP(1)

NDUM1(2)=1

CALL LNCNT(3)

PRINT 375

FORMAT(/'EIGENVALUES OF P'/)

CALL PRNT(DUMMY,NDUM1,4MEVLF,1)

N1 = NP(1) + 1

N = NA(1)**2

N2 = N1 + N + 2*NA(1)

CALL TRAMP(DUMMY(N1),NA,DUMMY(N2),NA)

CALL PRNT(DUMMY(N2),NA,4MA=F,F,1)

N2 = N1 + N

CALL LNCNT(3)

PRINT 385

FORMAT(/'EIGENVALUES OF A-MATRICE'/)

NDUM1(1) = NA(1)

NDUM1(2) = 2

CALL PRNT(DUMMY(N2),NDUM1,0,3)

GO TO 200

C

END
SUBROUTINE EXPMDF (A, NA, B, NB, H, NH, AM, NAM, HM, NMH, Q, NO, R, NR, F, NF, P, EXP)
 1 INP, HIDENT, HMIDENT, DISC, NEWT, STABLE, FNULL, ALPHA, IOP, DUMMY)
 2 IMPLICIT REAL*8 (A-H, O-Z)
 3 DIMENSION A(1), B(1), H(1), AM(1), HM(1), Q(1), R(1), P(1), DUMMY(1)
 4 DIMENSION NA(2), NB(2), NH(2), NAM(2), NMH(2), Q(2), R(2), NF(2), P(2)
 5 IOP(1), IOPT(5), NOUM(1), NOUM(2), NOUM(3)
 6 LOGICAL HIDENT, HMIDENT, DISC, NEWT, STABLE, FNULL, SYM
 7 COMMON /TOL/EPSAM, EPSAM, IACM
 8 IF (IOP(1) .EQ. 0) GO TO 300
 9 CALL LNCT(N)
 10 IF (.DISC) PRINT 25
 11 IF (.NOT. .DISC) PRINT 50
 12 25 FORMAT(/, * PROGRAM TO SOLVE ASYMPOTIC DISCRETE EXPLICIT MODEL* EQU
 13 RING PROBLEM*, // *, PLANT DYNAMICS*, /)
 14 50 FORMAT(/, * PROGRAM TO SOLVE ASYMPOTIC CONTINUOUS EXPLICIT MODEL* EQU
 15 RING PROBLEM*, // *, PLANT DYNAMICS*, /)
 16 CALL PRINT(A, NA, AM, 1)
 17 CALL PRINT(B, NB, BM, 1)
 18 IF (.CIDENT) GO TO 75
 19 CALL PRINT(H, NH, HM, 1)
 20 GO TO 100
 21 75 CONTINUE
 22 CALL LNCT(3)
 23 PRINT 85
 24 85 FORMAT(/, * H IS AN IDENTITY MATRIX*, /)
 25 CALL LNCT(4)
 26 PRINT 125
 27 125 FORMAT(/, * MODEL DYNAMICS*, /)
 28 CALL PRINT(AM, NAM, AM, 1)
 29 IF (.HMIDENT) GO TO 175
 30 CALL PRINT(HM, NMH, HM, 1)
 31 GO TO 200
 32 175 CONTINUE
 33 CALL LNCT(3)
 34 PRINT 185
 35 185 FORMAT(/, * HM IS AN IDENTITY MATRIX*, /)
 36 CALL LNCT(4)
 37 PRINT 225
 38 225 FORMAT(/, * WEIGHTING MATRICES*, /)
 39 CALL PRINT(NO, NO, NO, 1)
 40 CALL PRINT(R, NR, NR, 1)
 41 CALL EVALUATE(Q, Q, A, AM, 1)
 42 CALL ASMDIF(A, NA, Q, NB, H, NH, DUMMY, Q, R, NR, F, NF, P, HIDENT, DISC, NE)
 43 300 CONTINUE
 44 IF (.IOP(2) .EQ. 0) GO TO 400
 45 CALL EQATE(Q, NO, DUMMY, NO)
 46 CALL ASMDIF(A, NA, B, NB, H, NH, DUMMY, Q, R, NR, F, NF, P, HIDENT, DISC, NE
 47 400 CONTINUE
IF(IOP(1).EQ.0) GO TO 600
CALL LNCNT(4)
PRINT 425
FORMAT(/'' CONTROL LAW U = - F(COL.(X,X')) , F = (F11,F12)**/) CALL LNCNT(3)
PRINT 459
FORMAT(/'' PART OF F MULTIPLYING X */) CALL PRNT(F,NF,4H CONTROL.L,A4 U a F (COL. (X,X)) .F (F11,F12) /)
PRINT 42S FOR m At(/,• CONTROL.
F (a (F11.F12 ••/) EXPOF
CALL LNCNT(3)
PRINT 475
FORMAT(/'' PLANT CLOSED-LOOP RESPONSE MATRIX A = BF11'') CALL PRNT(DUMMY(N1),NDUM1,0,3)
CALL LNCNT(2)
PRINT 425
FORMAT(/'' EIGENVALUES OF P11'') NDM1(1) = NA(1)
NDUM1(2) = 1
CALL PRNT(DUMMY(N1),NDUM1,0,3)
N1 = N1 + NDUM1(1)
NDUM1(2) = NA(1)
PRINT 500
FORMAT(/'' PLANT CLOSED-LOOP RESPONSE MATRIX'') N1 = N1 + NDUM1(1)*NDUM1(2)
NDUM1(2) = 2
CALL LNCNT(2)
PRINT 525
FORMAT(/'' EIGENVALUES OF CLOSED-LOOP RESPONSE MATRIX'') CALL PRNT(DUMMY(N1),NDUM1,0,3)
600 CONTINUE
NF(1) = NA(2)
NF(2) = NA(1)
CALL MULT(B,NF,F,NF,DUMMY,NA)
CALL SUBST(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)
700 CONTINUE
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 SUBST(Q,N0,DUMMY(N1),NDUM2)
IF( HIDENT ) GO TO 725
CALL MULT(Q,N0,HM,NHM,DUMMY(N1),NDUM2)
725 CONTINUE
IOP(1) = 1177
N2 = N1 + NQ(1)*NHM(2)
CALL TRANP(NH,N0,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
725 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
A-91
CALL EQUATE(DUMMY(N1),NDUM1,P(N3),NDUM2)
IF (.DISC.) GO TO 800
EPSA = EPSAM
CALL HARTW(DUMMY,NA,AM,NAM,P(N3),NDUM2,IOPTT,SYM, EPSA,EPSA,DUMMY)
I(N2))
GO TO 900
800 CONTINUE
CALL SCALE(P(N3),NDUM1,P(N3),NDUM2,-1.0)
N2 = N2+NAM(1)**2
CALL EQUATE(DUMMY,NAM,DUMMY(N2),NAM)
CALL SUM(DUMMY,NA,P(N3),NDUM2,DUMMY(N2),NAM,IOPTT,SYM,DUMMY(N4))
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)
GO TO 1100
1000 CONTINUE
CALL EQUATE(R,NR,DUMMY,NR)
1100 CONTINUE
N1 = NR(1)**2 + 1
CALL SYMPOS(NR(1),NR(1),DUMMY,NR(1),F(N2),IOPTT,IOPTT,NORM,ISCA)
IF (.IERR .EQ. 0) GO TO 1200
CALL LNCNT(3)
PRINT 1150
1150 FORMAT(/,' IN EXPMDF, THE COEFFICIENT MATRIX FOR SYMPOS IS NOT SYMMETRIC POSITIVE DEFINITE ,/) RETURN
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) = NA(1)
CALL PRNT(P(N3),NDUM1,4H P12,1)
RETURN
END
SUBROUTINE IMPMDF(A, NA, B, NB, H, NH, AM, NAM, BM, NBM, Q, NR, F, NF, P, N)

IMPLICIT REAL*8 (A-H, O-Z)

DIMENSION A(1), B(1), H(1), AM(1), BM(1), Q(1), F(1), P(1), DUMMY(1)

DIMENSION NA(2), NH(2), NAM(2), NBM(2), NR(2), NF(2), NBM(2)

IOP(1), IOPT(5), DUMMY(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-FOLLOWING PROBLEM',//,'PLANT DYNAMICS',/) IMPO

50 FORMAT(/,'PROGRAM TO SOLVE ASYMPTOTIC CONTINUOUS IMPLICIT MODEL-FOLLOWING PROBLEM',//,'PLANT DYNAMICS',/) IMPO

CALL PRNT(A, NA, H, AM, A)

CALL PRNT(B, NR, H, BM, B)

IF(IDENT) GO TO 75

CALL PRNT(H, NH, H)

GO TO 100

75 CONTINUE

CALL LNCNT(3)

PRINT 25

85 FORMAT(/,'H IS AN IDENTITY MATRIX',/) IMPO

100 CONTINUE

CALL LNCNT(4)

PRINT 125

125 FORMAT(/,'MODEL DYNAMICS',/) IMPO

CALL PRNT(AM, NAM, 4H AM, 1)

CALL PRNT(RM, NBM, 4H 9M, 1)

CALL LNCNT(4)

PRINT 150

150 FORMAT(/,'WEIGHTING MATRICES',/) IMPO

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, DUMMY, NA)

CALL SUBT(B, NR, BM, DUMMY(N1), NB)

GO TO 400

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)

51.

C

300 CONTINUE

IF(IOP(1) .EQ. 0) GO TO 500

CALL LNCNT(3)

PRINT 450

450 FORMAT(/,'MATRIX HA = AM*-') IMPO

CALL PRNT(DUMMY, NH, 0, 3)

CALL LNCNT(3)

PRINT 475

475 FORMAT(/,'MATRIX HB = BM*-') IMPO

CALL PRNT(DUMMY(N1), NBM, 0, 3)

C
63 500 CONTINUE
64 N2 = N1 + N
65 N3 = N2 + N
66 N4 = N3 + N
67 CALL MULT(Q,N0,DUMMY,NH,DUMMY(N2),NH)
68 CALL MULT(NQ,DUMMY(N1),NBM,DUMMY(N3),NBM)
69 CALL TRANP(DUMMY,NH,DUMMY(N4),NOM1)
70 CALL MULT(DUMMY(NA),NDUM1,DUMMY(N2),NH,DUMMY,NA)
71 CALL MULT(DUMMY(N4),NDUM1,DUMMY(N3),NBM,DUMMY(N2),NB)
72 CALL TRANP(DUMMY(N1),NB4,DUMMY(N4),NDUM1)
73 CALL SCALE(DUMMY(N2),NB,N,DUMMY(N1),NB2,0)
74 CALL MULT(DUMMY(N4),NDUM1,DUMMY(N3),NAM,DUMMY(N2),NR)
75 CALL ADD(DUMMY(N2),NR,SR,NDUM2,NDUM1)
76 IF(IOP(1),EQ.0) GO TO 600
77 CALL LNCNT(3)
78 PRINT 525
79 525 FORMAT(//,'MATRIX (HA-AMH TRANSPOSE)Q( HA=AMH)*)
80 CALL PRNT(DUMMY,NA,0,3)
81 CALL LNCNT(3)
82 PRINT 550
83 550 FORMAT(//,'MATRIX (HA-AMH TRANSPOSE)Q( HB=BM)*)
84 CALL PRNT(DUMMY(N1),NB,0,3)
85 CALL LNCNT(3)
86 PRINT 575
87 575 FORMAT(//,'MATRIX (HB=BM TRANSPOSE)Q( HB=BM) + R*)
88 CALL PRNT(DUMMY(N2),NR,0,3)
89 CALL LNCNT(3)
90 600 CONTINUE
91 I0PT(1)= 0
92 I0PT(2)= 1
93 I0PT(3)= 1
94 N5 = N4 + N
95 CALL EQUATE(A,NA,DUMMY(N3),NA)
96 CALL PREFILT(DUMMY(N3),NA,B,NR,DUMMY,NA,DUMMY(N1),NB,DUMMY(N2),NR,DUMMY(N5))
97 IF(I0PT(1),EQ.0) GO TO 700
98 CALL LNCNT(3)
99 PRINT 625
100 CALL LNCNT(3)
101 625 FORMAT(//,'PREFILTER GAIN')
102 CALL PRT(DUMMY(N4),NF,0,3)
103 CALL LNCNT(3)
104 PRINT 650
105 650 FORMAT(//,'MATRIX A - B(PREFILTER)*)
106 CALL PRNT(DUMMY(N3),NA,0,3)
107 CALL LNCNT(3)
108 PRINT 675
109 675 FORMAT(//,'MODIFIED STATE VECTOR WEIGHTING MATRIX')
110 CALL PRNT(DUMMY,NA,0,3)
111 CALL LNCNT(3)
112 700 CONTINUE
113 CALL EQUATE(DUMMY(N4),NF,DUMMY(N1),NF)
114 CALL LNCNT(3)
115 IF(I0PT(2),EQ.1000) RETURN
116 IF(I0PT(1)=I0PT(2)) RETURN
117 I0PT(1)=I0PT(2)
118 I0PT(2)=I0PT(3)
119 I0PT(3)=I0PT(4)
120 I0PT(4)=0
121 I0PT(5)=0
122 IF(10PT(1),EQ.0) GO TO 800
123 CALL SCALE(DUMMY(N3),NA,B,NH,NDUM,NA,DUMMY(N2),NB,F,NP,N)
124 IF('TRUE,DISC,NEW,STABLE,FINULL,ALPHA,I0PT,DUMMY(N4))
125 IF(I0PT(1),EQ.0) GO TO 800
CALL LNCNT(3)
PRINT 725
725 PRINT(//,* GAIN FROM ASMREG*)
CALL PRNT(F,NF,0,3)
CALL LNCNT(3)
PRINT 775
PRINT(//,* SOLUTION OF ASSOCIATED STEADY-STATE RICCATI EQUATION*)
CALL PRNT(P,NP,0,3)
CALL LNCNT(3)
PRINT 775
PRINT(//,* EIGENVALUES OF P*)
NDUM1(1)= NA(1)
NDUM1(2)= 1
CALL PRNT(NDUMMY(N4),NDUM1,0,3)
800 CONTINUE
CALL ADD(F,NF,DUMMY(N1),NF,F,NF)
IF( IOP(1),EQ, 0 ) RETURN
CALL LNCNT(4)
PRINT 825
PRINT(//,* GAIN FOR MODEL-FOLLOWING CONTROL LAW, U = - F X , F =
LPREFILTER) + (ASMREG)^*
CALL PRNT(F,NF,4H F ,1)
N6 = N4 + NA(1)
CALL PRNT(DUMMY(N6),NA,4HA=BF,1)
NDUM1(2) = 2
N6 = N6 + N
CALL LNCNT(3)
PRINT 850
PRINT(//,* EIGENVALUES OF A-BF*)
CALL PRNT(DUMMY(N6),NDUM1,0,3)
RETURN
END
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
   CALL PRNT (A,NA,NAM)
RETURN
CALL_LNCNT(1)
WRITE (6,916) NAM,NR,NC
916 FORMAT (* ERROR IN READ1 MATRIX *A* HAS NA=*216)
RETURN
END
SUBROUTINE BALANC(NM,N,A,LOW,IGH,SCALE)
IMPLICIT REAL*8(A-H,O-Z)
INTEGER I,J,K,L,M,N,JJ,MM,IGH,LOW,IEXC
DIMENSION A(NM,N),SCALE(N)

REAL C,F,G,R,S,2,RADIX
REAL DABS
LOGICAL NOCONV

************ RADIX IS A MACHINE DEPENDENT PARAMETER SPECIFYING
THE BASE OF THE MACHINE FLOATING POINT REPRESENTATION.

RADI X = 16.
B2 = RADIX * RADIX
K = 1
L = N
GO TO 100

*********** IN-LINE PROCEDURE FOR ROW AND
COLUMN EXCHANGE ***********
SCALE(M) = J
IF (J .EQ. M) GO TO 50

DO 30 I = 1, L
F = A(I,J)
A(I,J) = A(I,M)
A(I,M) = F
30 CONTINUE

DO 40 I = K, N
F = A(J,I)
A(J,I) = A(M,I)
A(M,I) = F
40 CONTINUE

50 GO TO (80,130), IE XC

*********** SEARCH FOR ROWS ISOLATING AN EIGENVALUE
AND PUSH THEM DOWN ***********
80 IF (L .EQ. 1) GO TO 280
L = L - 1

*********** FOR J=L STEP -1 UNTIL 1 DO = ***********
100 DO 120 JJ = 1, L
J = L + 1 - JJ
120 CONTINUE

DO 110 I = 1, L
IF (I .EQ. J) GO TO 110
IF (A(J,I) .NE. 0.000) GO TO 120
110 CONTINUE

M = L
IE XC = 1
GO TO 20

120 CONTINUE

GO TO 140

*********** SEARCH FOR COLUMNS ISOLATING AN EIGENVALUE
AND PUSH THEM LEFT ***********
130 K = K + 1
140 DO 170 J = K, L
170 CONTINUE

DO 150 I = K, L
IF (I.EQ. J) GO TO 150
IF (A(I,J) .NE. 0.000) GO TO 170
150 CONTINUE
M = K
IEYC = 2
GO TO 20
170 CONTINUE

NOW BALANCE THE SUBMATRIX IN ROWS K TO L

DO 180 I = K, L
SCALE(I) = 1.000
180 SCALE(I) = SCALE(I) * F

ITERATIVE LOOP FOR NORM REDUCTION

NOCONV = .FALSE.,

DO 270 I = K, L
C = 0.000
R = 0.000
270 CONTINUE

GUARD AGAINST ZERO C OR R DUE TO UNDERFLOW

IF (C .EQ. 0.000 .OR. RET. = 0.000) GO TO 270
G = R / RADIX
F = 1.000
S = C + R
IF (C .GE. G) GO TO 230
F = F / RADIX
C = C / S
GO TO 230
220 G = R * RADIX
230 F = F * RADIX
C = C * B2
GO TO 230

NOW BALANCE

IF ((C + R) / F .GE. 0.95 * S) GO TO 270
G = 1.000 / F
SCALE(I) = SCALE(I) * F
NOCONV = .TRUE.,

DO 250 J = K, N
A(I,J) = A(I,J) * G
250 A(I,J) = A(I,J) * F

DO 260 J = 1, L
A(J,I) = A(J,I) * F
260 CONTINUE

IF (NOCONV) GO TO 190

LOW = K
IGH = L
RETURN

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)

REAL X,Y

REAL DABS

INTEGER INT(IGH)

LA = IGH - 1
KP1 = LOW + 1

IF (LA .LT. KP1) GO TO 200

DO 180 M = KP1, LA

MM1 = M - 1

X = 0.000

I = M

DO 100 J = M, IGH

IF (DABS(A(J,MM1)) .LE. DABS(X)) GO TO 100

X = A(J,MM1)

I = J

100 CONTINUE

INT(M) = I

IF (I .EQ. M) GO TO 130

DO 110 J = MM1, N

Y = A(I,J)

A(I,J) = A(M,J)

A(M,J) = Y

110 CONTINUE

Y = A(I,I)

A(J,I) = A(J,M)

A(J,M) = Y

120 CONTINUE

130 IF (X .EQ. 0.000) GO TO 180

MP1 = M + 1

DO 160 I = MP1, IGH

Y = A(I,MM1)

IF (Y .EQ. 0.000) GO TO 160

Y = Y / X

A(I,MM1) = Y

DO 140 J = M, N

A(I,J) = A(I,J) - Y * A(M,J)

140 CONTINUE

A(I,MM1) = Y

DO 150 J = I, IGH

A(J,M) = A(J,M) + Y * A(I,J)

150 CONTINUE

160 CONTINUE

180 CONTINUE

RETURN

END

********** LAST CARD OF ELMHES **********

END
SUBROUTINE HNR(N, M, P, NrAvlG, Nr, H, W, WR, WI, IFRR)

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(N, M), WR(N), WI(N)
REAL OSORT, DABS, DSIGN
INTEGER MINO
LOGICAL NOTLAS

C REAL OSORT, DABS, DSIGN
C INTEGER MINO
LOGICAL NOTLAS
C MACHEP = 16.**(<=13)

IERR = 0
NORM = 0.000
K = 1
C ********** STORE ROOTS ISOLATED BY BALANC
C AND COMPUTE MATRIX NORM **********

DO 50 I = 1, N

DO 40 J = K, N

NORM = NORM + DABS(H(I, J))

K = I

IF (I .GE. LOW .AND. I .LE. IGH) GO TO 50

WR(I) = H(I, I)
WI(I) = 0.000.

50 CONTINUE

EN = IGH

T = 0.000

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 -- **********

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. 0.000) 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

DO 120 I = LOW, EN

H(I, I) = H(I, I) - X

120 CONTINUE

A-100
S = DABS(M(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, EN+2

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(S(M,M-1)) .EQ. DABS(Q) .EQ. DABS(R)) .LE. MACHEP .AND. DABS(P)

X .EQ. (DABS(H(M-1,M-1)) + DABS(2Z) + DABS(H(M+1,M+1))) GO TO 150

140 CONTINUE

150 MP2 = M + 2

C

DO 160 I = MP2, EN

M(I-I-2) = 0.000

IF (I .EQ. MP2) GO TO 160

M(I-I-3) = 0.000

160 CONTINUE

C *********** DOUBLE OR STEP INVOLVING ROWS L TO EN AND
C COLUMNS M TO EN ***********

170 S = DSIGN(DSORT(P+P+Q+R*R),P)

180 IF (K .EQ. M) GO TO 190

H(K,K+1) = -S * X

GO TO 190

190 IF (L .NE. M) H(K,K+1) = -H(K,K+1)

200 C

DO 210 J = K, EN

P = H(K,J) + R * H(K+1,J)

IF (.NOT. NOTLASS) GO TO 210

P = P + R * H(K+2,J)

H(K+2,J) = H(K+2,J) - P * ZZ

A-101
126 200 \[ H(K+1,J) = H(K+1,J) - P * Y \]
127 \[ H(K,J) = H(K,J) - P * X \]
128 C CONTINUE
129 C
130 J = MINO(EN,K+3)
131 C *********** COLUMN MODIFICATION ***********
132 DO 230 I = L, J
133 \[ P = X * H(I,K) + Y * H(I,K+1) \]
134 IF (.NOT. NOTLA3) GO TO 220
135 \[ P = P + ZZ * H(I,K+2) \]
136 \[ H(I,K+2) = H(I,K+2) - P * R \]
137 220 \[ H(I,K+1) = H(I,K+1) - P * Q \]
138 \[ H(I,K) = H(I,K) - P \]
139 230 CONTINUE
140 C
141 260 CONTINUE
142 C
143 GO TO 70
144 C *********** ONE ROOT FOUND ***********
145 270 WR(EN) = X + T
146 WI(EN) = 0.000
147 EN = NA
148 GO TO 60
149 C *********** TWO ROOTS FOUND ***********
150 280 P = (Y - X) / 2.000
151 Q = P + P + W
152 ZZ = DSORT(OABS(Q))
153 X = X + T
154 IF (Q .LT. 0.000) GO TO 320
155 C *********** REAL PAIR ***********
156 ZZ = P + DSIGN(ZZ,P)
157 WR(NA) = X + ZZ
158 WR(EN) = WR(NA)
159 IF (ZZ .NE. 0.000) WR(EN) = X - W / ZZ
160 WI(NA) = 0.000
161 WI(EN) = 0.000
162 GO TO 330
163 C *********** COMPLEX PAIR ***********
164 320 WR(NA) = X + P
165 WP(EN) = X + P
166 WI(NA) = ZZ
167 WI(EN) = -ZZ
168 330 EN = ENM2
169 GO TO 60
170 C *********** SET ERROR -- NO CONVERGENCE TO AN
171 C EIGENVALUE AFTER 30 ITERATIONS ***********
172 1000 IERR = EN
173 GO TO 60
174 C *********** LAST CARD OF HQR ***********
175 END
SUBROUTINE INVIT(NM,N,A,WK,W1,SELECT,MM,M,Z,IERA,RA1,RV1,RV2)
IMPLICIT REAL*8 (A-M,O-Z)
REAL*8 NORM,NORMV,ILAMBD,HACHEP
INTEGER I,J,K,L,M,N,S,II,IP,IM,MP,NM,NS,N1,UK,IP1,ITS,KM1,IERR
DIMENSION A(NM,N),WR(N),NI(N),Z(NM,MM),RM1(N,N),RV1(N),RV2(N)
REAL T,W,T,Y,EPS3,NORM,NORMV,GROOT,ILAMBD,HACHEP,RA1,UKROOT
REAL*8 OSORT,CDABS,DAABS,OFLOAT
INTEGER IABS
LOGICAL*1 SELECT(N)
COMPLEX*16 Z3,DCOMPLX
REAL*8 DREAL,OIMAG

MACHEP = 16.*(-13).
IERA = 0
UK = 0.
S = 1.
********** IP = 0, REAL EIGENVALUE
1. FIRST OF CONJUGATE COMPLEX PAIR
-1, SECONO OF CONJUGATE COMPLEX PAIR **********
IP = 0
N1 = N - 1.
DO 980 K = 1, N
IF (W(K),EQ, 0.000 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 (W(K),NE, 0.000) S = S + 1
IF (S .GT. MM) GO TO 1000
IF (UK .GE. K) GO TO 200
********** CHECK FOR POSSIBLE SPLITTING **********
DO 120 UK = K, N
IF (UK ,EQ, N) GO TO 140
IF (A(UK+1,UK), .EQ. 0.000) GO TO 140
120 CONTINUE
********** COMPUTE INFINITY NORM OF LEADING UK BY UK
********** EPS3 REPLACES ZERO PIVOT IN DECOMPOSITION
********** AND CLOSE ROOTS ARE MODIFIED BY EPS3**********
********** GROOT IS THE CRITERION FOR THE GROWTH **********
********** PERTURB EIGENVALUE IF IT IS CLOSE
A-103
63. C  TO ANY PREVIOUS EIGENVALUE **********
64. C  RLANMD = RLANMD + EPS3
65. C  ********** FOR I=1 STEP -1 UNTIL I DO -- **********
66. C  DO 260 I = 1, KMI
67. C  I = - I
68. C  IF (SELECT(I) .AND. DABS(WR(I)-RLAMBD) .LT. EPS3 .AND.
69. C  DABS(WI(I)-ILAMBD) .LT. EPS3) GO TO 220
70. C  260 CONTINUE
71. C  **** **** PERTURB CONJUGATE EIGENVALUE TO MATCH **********
72. C  IP1 = K + IP
73. C  WR(IP1) = RLANMD
74. C  ********** FORM UPPER HESSENBEG A-RLAMBD*I (TRANSPOSED)
75. C  AND INITIAL REAL VECTOR **********
76. C  280 MP = 1
77. C  DO 320 I = 1, UK
78. C  300 J = MP, UK
79. C  RM1(J,I) = A(I,J)
80. C  DO 300 J = MP, UK
81. C  RM1(J,I) = RM1(J,I) - RLANMD
82. C  MP = I
83. C  RV1(I) = EPS3
84. C  320 CONTINUE
85. C  ITS = 0
86. C  IF (ILAMBD .NE. 0.000) GO TO 520
87. C  ********** REAL EIGENVALUE, TRIANGULAR DECOMPOSITION WITH INTERCHANGES,
88. C  REPLACING ZERO PIVOTS BY EPS3 **********
89. C  IF (UK .EQ. 1) GO TO 420
90. C  DO 400 I = 2, UK
91. C  MP = I - 1
92. C  IF (DABS(RM1(MP,I)) .LE. DABS(RM1(MP,MP))) GO TO 360
93. C  DO 340 J = MP, UK
94. C  RM1(J,J) = RM1(J,MP)
95. C  RM1(J,MP) = Y
96. C  Y = RM1(J,J)
97. C  RM1(J,J) = RM1(J,MP)
98. C  340 CONTINUE
99. C  360 IF (RM1(MP,MP) .EQ. 0.000) RM1(MP,MP) = EPS3
100. C  X = RM1(MP,I) / RM1(MP,MP)
101. C  IF (X .EQ. 0.000) GO TO 400
102. C  DO 380 J = I, UK
103. C  RM1(J,I) = RM1(J,MP) - X * RM1(J,MP)
104. C  380 CONTINUE
105. C  400 CONTINUE
106. C  420 IF (RM1(UK,UK) .EQ. 0.000) RM1(UK,UK) = EPS3
107. C  ********** BACK SUBSTITUTION FOR REAL VECTOR
108. C  FOR I=UK STEP -1 UNTIL I DO -- **********
109. C  440 DO 500 II = 1, UK
110. C  I = UK + 1 - II
111. C  500 CONTINUE
112. C  520 CONTINUE
```
126 460     Y = Y - RM1(J,I) * RV1(J)
127 C
128 480     RV1(I) = Y / RM1(I,I)
129 500     CONTINUE
130 C
131     GO TO 740
132 C     ********* COMPLEX EIGENVALUE.
133 C     TRIANGULAR DECOMPOSITION WITH INTERCHANGES.
134 C     REPLACING ZERO PIVOTS BY EPS3. STORE IMAGINARY
135 C     PART IN UPPER TRIANGLE STARTING AT (1,3) *******
136 520     NS = N - 3
137     Z(1,1) = -ILAMBD
138     Z(1,3) = 0.000
139     IF (N .EQ. 2) GO TO 550
140     RM1(1,3) = -ILAMBD
141     Z(1,3) = 0.000
142     IF (N .EQ. 3) GO TO 550
143 C
144     DO 540 I = 4, N
145 540     RM1(1,I) = 0.000
146 C
147 550     DO 640 I = 2, N
148     MP = I - 1
149     W = RM1(MP,I)
150     IF (I .LT. N) T = RM1(MP,I+1)
151     IF (I .EQ. N) T = Z(MP,3-1)
152     X = RM1(MP,MP) * RM1(MP,MP) + T * T
153     IF (W * W .LT. X) GO TO 580
154     X = RM1(MP,MP) / W
155     Y = T / W
156     RM1(MP,MP) = W
157     IF (I .LT. N) RM1(MP,I+1) = 0.000
158     IF (I .EQ. N) Z(MP,3-1) = 0.000
159 C
160     DO 560 J = I, N
161     W = RM1(J,I)
162     RM1(J,I) = RM1(J,MP) * X * W
163     RM1(J,MP) = W
164     IF (J .LT. N1) GO TO 553
165     L = J - NS
166     Z(I,L) = Z(MP,L) * Y * W
167     Z(MP,L) = 0.000
168     GO TO 560
169 555     RM1(I,J+2) = RM1(MP,J+2) * Y * W
170     RM1(MP,J+2) = 0.000
171 560     CONTINUE
172 C
173     RM1(I,I) = RM1(I,I) - Y * ILAMBD
174     IF (I .LT. N1) GO TO 570
175     L = I - NS
176     Z(MP,L) = -ILAMBD
177     Z(I,L) = Z(I,L) * X * ILAMBD
178     GO TO 640
179 570     RM1(MP,I+2) = -ILAMBD
180     RM1(I,I+2) = RM1(I,I+2) * X * ILAMBD
181     GO TO 640
182 580     IF (X .NE. 0.000) GO TO 660
183     RM1(MP,MP) = EPS3
184     IF (I .LT. N) RM1(MP,I+1) = 0.000
185     IF (I .EQ. N) Z(MP,3-1) = 0.000
186     T = 0.000
187     Y = EPS3 * EPS3
188 600     W = N / X
```
PROCEDURE

189  X = W * (M(1,M,P) - M) + W
190  Y = T * W
191  C
192  DO 620 J = 1, UK
193  IF (J .LT. N1) GO TO 610
194  L = J - NS
195  T = Z(MP,L)
196  Z(I,L) = -X * T - Y * RMI(J,MP)
197  GO TO 615
198  T = RMI(MP,J+2)
199  RMI(J,J+2) = -X * T - Y * RMI(J,MP)
200  RMI(J, I) = RMI(J, I) - X * RMI(J, MP) + Y * T
201  CONTINUE
202  C
203  IF (I .LT. N1) GO TO 630
204  L = I - NS
205  Z(I,L) = Z(I,L) - ILAMBD
206  GO TO 640
207  RM(I, I+2) = RMI(I, I+2) - ILAMBD
208  CONTINUE
209  C
210  IF (UK .LT. N1) GO TO 650
211  L = UK - NS
212  T = Z(UK,L)
213  GO TO 655
214  T = RMI(UK,UK+2)
215  IF (RMI(UK,UK) .EQ. 0.00D0 .AND. T .EQ. 0.00D0) RMI(UK,UK) = EPS3
216  C
217  **** BACK SUBSTITUTION FOR COMPLEX VECTOR ****
218  DO 720 II = 1, UK
219  I = UK + 1 - II
220  X = RV1(I)
221  Y = 0.00D0
222  IF (I .EQ. UK) GO TO 700
223  IP1 = I + 1
224  C
225  DO 680 J = IP1, UK
226  IF (J .LT. N1) GO TO 670
227  L = J - NS
228  T = Z(I,L)
229  GO TO 675
230  T = RMI(I,J+2)
231  X = X - RMI(I,J) * RV1(J) + T * RV2(J)
232  Y = Y - RMI(I,J) * RV2(J) - T * RV1(J)
233  CONTINUE
234  C
235  IF (I .LT. N1) GO TO 710
236  L = I - NS
237  T = Z(I,L)
238  GO TO 715
239  T = RMI(I,I+2)
240  Z3 = DCMPLX(X,Y) / DCMPLX(RMI(I,I),T)
241  RV1(I) = DREAL(Z3)
242  RV2(I) = DIMAG(Z3)
243  CONTINUE
244  C
245  **** ACCEPTANCE TEST FOR REAL OR COMPLEX EIGENVECTOR AND NORMALIZATION ****
246  ITS = ITS + 1
247  NORM = 0.00D0
248  NORMV = 0.00D0
249  C
250  DO 780 I = 1, UK
251  IF (ILAMBD .EQ. 0.00D0) X = DABS(RV1(I))
IF (ILAMBDA NE. 0.000) X = CDABS(DCMPLX(RV1(I),RV2(I)))

IF (NORM .GE. x) GO TO 760

NORM = NORM + X

CONTINUE

IF (NORM .LT. GROWTO) GO TO 840

**+*** ACCEPT VECTOR **+

IF (ILAMBDA .EQ. 0.000) X = 1.000 * X

IF (ILAMBDA .NE. 0.000) Y = RV2(J)

GO TO 820

820 CONTINUE

CONTINUE

IF (ITS .GE. UK) GO TO 880

X = UKROOT

Y = EPS3 / (X + 1.000)

RV1(I) = EPS3

GO TO 860

860 RV1(I) = Y

J = UK + ITS + 1

RV1(J) = RV1(J) - EPS3 * X

IF (ILAMBDA .EQ. 0.000) GO TO 440

GO TO 660

SET ERROR = UNACCEPTED EIGENVECTOR

J = 1

IERR = -K

**+*** SET REMAINING VECTOR COMPONENTS TO ZERO

900 DO 920 I = J, N

Z(I,S) = 0.000

920 CONTINUE

GO TO 900

900 DO 920 I = J, N

Z(I,S) = 0.000

920 CONTINUE

GO TO 1001

**+*** SET ERROR = UNDERESTIMATE OF EIGENVECTOR

SPACE REQUIRED

1000 IF (IERR .NE. 0) IERR = IERR + N

IF (IERR .NE. 0) IERR = -(2 * N + 1)

M = S - 1 = IABS(IP)

RETURN

**+*** LAST CARD OF INVIT

END
SUBROUTINE ELMBAK(NM, LOW, IGH, A, INT, M, Z)

IMPLICIT REAL*6 (A-H, O-Z)

INTEGER I, J, M, LA, MM, MP, NM, IGH, KPI, LOW, MP1

DIMENSION A(NM, IGH), Z(NM, M)

REAL X

INTEGER INT(IGH)

IF (M .EQ. 0) GO TO 200

LA = IGH + 1

KPI = LOW + 1

IF (LA .LT. KPI) GO TO 200

************* FOR MP=IGH+1 STEP -1 UNTIL LOW+1 DO = *************

DO 140 MM = KPI, LA

MP = LOW + IGH - MM

MP1 = MP + 1

DO 110 I = MP1, IGH

X = A(I, MP-1)

IF (X .EQ. 0.0) GO TO 110

DO 100 J = 1, M

Z(I, J) = Z(I, J) + X * Z(MP, J)

110 CONTINUE

I = INT(MP)

IF (I .EQ. MP) GO TO 140

DO 130 J = 1, M

X = Z(I, J)

Z(I, J) = Z(MP, J)

Z(MP, J) = X

130 CONTINUE

140 CONTINUE

200 RETURN

************** 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)

REAL S

IF (M.EQ.0) GO TO 200

IF (IGH.EQ.LOW) GO TO 120

DO 110 I = LOW, IGH

S = SCALE(I)

110 CONTINUE

120 CONTINUE

**LAST CARD OF BALBAK**
SUBROUTINE OETFAC(NMAX,N,A,IPIVOT,IDET,DETERM,SCALE,WK,IER)
IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION A(NMAX,1),IPIVOT(1),WK(1)
ISCALE=0
NM1=N-1
IER=0

DETERMINANT_CALCULATION_TEST
IF(IDET.EQ.1)GO TO 230
TEST FOR A SCALAR MATRIX
IF(NM1.GT.0)GO TO 20
DETERM=A(1,1)
RETURN

COMPUTE SCALING FACTORS

CONTINUE
DO 60 I=1,N
P=0.0
DO 30 J=1,N
Q=MAX1(P,DABS(A(I,J)))
IF(Q.GT.P)P=Q
CONTINUE
DO 60 WK(I)=P

DO 210 M=1,NM1
PIVOTAL LOGIC SETUP
P=0.0
DO 110 I=M,N
Q=DABS(A(I,M)/WK(I))
IF(Q.GT.P)110,110,100
IP=I
110 CONTINUE
IP=I

PIVOT(M)=IP
IF(P.EQ.0.)GO TO 40
IF(M.EQ.IP)GO TO 155
PIVOT THE M-TH ROW OF THE A MATRIX
DO 150 I=1,N
P=A(IP,I)
A(IP,I)=A(M,I)
A(M,I)=P
MP1=M+1
150 P=WK(IP)
WK(IP)=WK(M)
WK(M)=P
155 MP1=M+1
LU FACTORIZATION LOGIC

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
   A(I, K) = A(I, K) - Q*A(M, K)

210 CONTINUE

IF (A(N, N) .LE. 0.) GO TO 40

CALCULATION OF THE DETERMINANT OF A

IF (IDET .EQ. 0.) RETURN

SIGN = 1.0

DETERM = 1.0

ADJUST SIGN OF DETERMINANT DUE TO PIVOTAL STRATEGY

DO 250 I = 1, NM1
   IF (I .EQ. IPIVOT(I)) 240, 250, 250
   SIGN = -SIGN

240 CONTINUE

DO 340 I = IPIVOT, N
   P = A(I, I)

260 CONTINUE

IF (R1 .GT. DABS(P)) GO TO 280
   P = P*R2
   ISCALE = ISCALE + 1
   GO TO 260

280 CONTINUE

IF (R2 .LT. DABS(P)) GO TO 290
   P = P*R1
   ISCALE = ISCALE - 1
   GO TO 280

290 CONTINUE

DETERM = DETERM * P

300 CONTINUE

IF (R1 .GT. DABS(DETERM)) GO TO 320
   DETERM = DETERM * R2
   ISCALE = ISCALE + 1
   GO TO 300

320 CONTINUE

IF (R2 .LT. DABS(DETERM)) GO TO 340
   DETERM = DETERM * R1
   ISCALE = ISCALE - 3
   GO TO 320

340 CONTINUE

DETERM = DETERM * SIGN

RETURN

A-111
SUBROUTINE AXPA(A, U, M, NA, NU, B, V, N, NB, NV, C, NC, EPSA, FAIL)

IMPLICIT REAL*8 (A-H, O-Z)

DIMENSION A(NA, 1), U(NU, 1), B(NR, 1), V(NV, 1), C(NC, 1)

INTEGER M1, M2, M, N1, N2, N, N1M1

C IF REQUIRED, REDUCE A TO UPPER REAL SCHUR FORM.

IF(EPSA .LT. 0.) GO TO 35

DO 10, I = 1, M

10 CONTINUE

CALL HSHLDR(A, M, NA)

CALL BCKMLT(A, U, M, NA, NU)

IF(NM .EQ. 0) GO TO 25

DO 20 I = 1, NM

A(I+1, I) = A(I, I)

20 CONTINUE

CALL SCHUR(A, U, M, NA, NU, EPSA, FAIL)

IF(FAIL .NE. 0) RETURN

DO 30 I = 1, M

30 CONTINUE

C IF REQUIRED, REDUCE B TO UPPER REAL SCHUR FORM.

IF(EPSB .LT. 0.) GO TO 45

CALL HSHLDR(B, N, NB)

CALL BCKMLT(B, V, N, NB, NV)

IF(NM .EQ. 0) GO TO 45

DO 40 I = 1, NM

B(I+1, I) = B(I, I)

40 CONTINUE

CALL SCHUR(B, V, N, NB, NV, EPSB, FAIL)

FAIL = -FAIL

IF(FAIL .NE. 0) RETURN

C TRANSFORM C.

DO 60 J = 1, N

DO 60 I = 1, M

60 CONTINUE

45 DO 60 J = 1, N

DO 50 I = 1, M

A(I, M1) = 0.

DO 50 K = 1, M

50 CONTINUE

A(I, M1) = A(I, M1) + U(K, I)*C(K, J)

56 50 CONTINUE

DO 60 I = 1, M

C(I, J) = A(I, M1)

58 60 CONTINUE

DO 80 I = 1, M

80 CONTINUE

DO 70 J = 1, N

70 CONTINUE

B(N1, J) = 0.
63 DO 70 K=1,N
64 B(N1,J) = B(N1,J) + C(I,K)*V(K,J)
65 70 CONTINUE
66 DO 80 J=1,N
67 C(I,J) = B(N1,J)
68 80 CONTINUE
69 CONTINUE

AXPOL

70 C SOLVE THE TRANSFORMED SYSTEM.
71 CALL SHRSLV(A,B,C,M,N,NA,NB,NC)
72 C TRANSFORM C BACK TO THE SOLUTION.
73 DO 100 J=1,N
74 DO 90 I=1,M
75 A(I,M1) = 0,
76 DO 90 K=1,M
77 A(I,M1) = A(I,M1) + U(I,K)*C(K,J)
78 90 CONTINUE
79 C(I,J) = A(I,M1)
80 100 CONTINUE
81 DO 120 I=1,M
82 DO 110 J=1,N
83 B(N1,J) = 0,
84 DO 110 K=1,N
85 B(N1,J) = B(N1,J) + C(I,K)*V(J,K)
86 110 CONTINUE
87 C(I,J) = B(N1,J)
88 120 CONTINUE
89 RETURN
90 END AXPOL
SUBROUTINE SHRSLV(A, C, M, N, NA, NB, NC)
IMPLICIT REAL*8 (A-H, O-Z)
DIMENSION
IA(NA, 1), IB(NB, 1), IC(NC, 1)
INTEGER
10 OK, 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
20 C(I, J) = C(I, J) + C(I, JB)*B(IB, J)
CONTINUE
30 K = 1
DO 40 KM1 = K-1
40 DK = 1
IF(K .EQ. M) GO TO 45
IF(A(K, K+1) .NE. 0.) DK = 2
K = K+DK-1
45 IF(K .EQ. 1) GO TO 60
DO 50 I = K, KK
50 C(I, J) = C(I, J) + A(I, JA)*C(JA, J)
CONTINUE
50 NSYS = 2
CALL SYSSLV
60 C(K, L) = P(1)
C(K, LL) = P(2)
GO TO 100
80 IF(DK .EQ. 2) GO TO 90
90 T(1, 1) = A(K, K) + B(L, L)
T(1, 2) = A(K, KK)
T(2, 1) = C(K, L)
T(2, 2) = A(K, KK) + B(L, L)
P(1) = C(K, L)
P(2) = C(K, LL)
NSYS = 2
CALL SYSSLV
GO TO 100
100 A-115
63 \[ T(1,4) = 0. \]
64 \[ T(2,1) = 4(KK,K) \]
65 \[ T(2,2) = 4(KK,KK) + B(L,L) \]
66 \[ T(2,3) = 0. \]
67 \[ T(2,4) = T(1,3) \]
68 \[ T(3,1) = B(L,L) \]
69 \[ T(3,2) = 0. \]
70 \[ T(3,3) = 4(K, K) + 8(LL, LL) \]
71 \[ T(3,4) = T(1,2) \]
72 \[ T(4,1) = 0. \]
73 \[ T(4,2) = T(3,1) \]
74 \[ T(4,3) = T(2,1) \]
75 \[ T(4,4) = 4(KK,KK) + 8(LL, LL) \]
76 \[ P(1) = C(K,L) \]
77 \[ P(2) = C(KK,L) \]
78 \[ P(3) = C(K,LL) \]
79 \[ P(4) = C(KK,LL) \]
80 \[ NSYS = A \]
81 CALL SYSSLV
82 \[ C(K,L) = P(1) \]
83 \[ C(KK,L) = P(2) \]
84 \[ C(K,LL) = P(3) \]
85 \[ C(KK,LL) = P(4) \]
86 \[ 100 \quad K = K + DK \]
87 IF(K \ LE. M) GO TO 40
88 \[ L = L + DL \]
89 IF(L \ LE. N) GO TO 10
90 RETURN
91 END
SUBROUTINE ATXPKA(A,U,C,N,NA,NU,NC,EPS,FAIL)
IMPLICIT REAL*8(A-H,O-Z)
DIMENSION A(NA,1),U(NU,1),C(NC,1)
INTEGER 1,FAL
N1 = N+1
NM1 = N-1
C IF REQUIRED, REDUCE A TO LOWER REAL SCHUR FORM.
IF(EPS .LT. 0.) GO TO 15
CALL HSMLDR(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,U1,NA,NU,EPS,FAIL)
IF(FAIL .NE. 0) RETURN
C TRANSFORM C.
15 DO 20 I=1,N
C(I,I) = C(I,I)/2.
20 CONTINUE
DO 30 I=1,N
DO 30 J=1,N
A(I,N1) = 0.
DO 30 K=1,N
A(I,J) = A(I,J) + C(I,K)*U(K,J)
30 CONTINUE
C TRANSFORM C BACK TO THE SOLUTION.
DO 60 I=1,N
DO 60 J=1,N
C(I,J) = A(I,N1)
60 CONTINUE
DO 70 I=1,N
DO 70 J=1,N
C(I,J) = C(I,J) + C(J,I)
70 CONTINUE
CALL SYMSLV(A,C,N,NA,NC)
C SOLVE THE TRANSFORMED SYSTEM.
DO 80 I=1,N
C(I,I) = C(I,I)/2.
80 CONTINUE
DO 100 I=1,N
DO 100 J=1,N
A(N1,J) = 0.
DO 100 K=1,N
A(N1,J) = A(N1,J) + C(I,K)*U(J,K)
100 CONTINUE
RETURN
END
DO 100 J=1,N
C(I,J) = A(N1,J)
DO 100 I=1,N
A(I,N1) = 0.
DO 110 K=1,N
A(I,N1) = A(I,N1) + U(I,K)*C(K,J)
CONTINUE
DO 120 I=1,N
C(I,J) = A(I,N1)
CONTINUE
DO 130 I=1,N
DO 130 J=1,N
C(I,J) = C(I,J) + C(J,I)
C(J,I) = C(I,J)
CONTINUE
RETURN
END
SUBROUTINE SYMSLV(A,C,N,NA,NC)
IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION IA(NA,1),C(NC,1)
INTEGER

COMMON/SLVBLXT(5,5),P(5),NSYS
L = 1
0.
10 DL = 1
10 IF(L .EQ. N) GO TO 20
10 IF(A(L+1,L),NE. 0.) DL = 2
11 20 LL = L+DL-1
12 K = L
13 30 KM1 = K-1
14 DD = 1
15 IF(K .EQ. N) GO TO 35
16 IF(A(K+1,K),NE. 0.) DK = 2
17 35 KK = K+DK-1
18 IF(K .EQ. L) GO TO 45
19 DO 40 I=K,KK
20 DO 40 J=LL,DL-1
21 CONTINUE
22 C(I,J) = C(I,J) - A(I,A) * C(I,J)
23 CONTINUE
24 45 IF(DL .EQ. 2) GO TO 60
25 IF(DK .EQ. 2) GO TO 50
26 T(1,1) = A(K,K) + A(L,L)
27 IF(T(1,1),EQ. 0.) STOP
28 C(K,L) = C(K,L)/T(1,1)
29 GO TO 90
30 50 T(1,1) = A(K,K) + A(L,L)
31 T(1,2) = A(K,K)
32 T(2,1) = A(L,K)
33 T(2,2) = A(K,K) + A(L,L)
34 P(1) = C(K,L)
35 P(2) = C(K,K)
36 NSYS = 2
37 CALL SYSSLV
38 C(K,L) = P(1)
39 C(K,L) = P(2)
40 GO TO 90
41 60 IF(DK .EQ. 2) GO TO 70
42 T(1,1) = A(K,K) + A(L,L)
43 T(1,2) = A(L,L)
44 T(2,1) = A(L,L)
45 T(2,2) = A(K,K) + A(L,L)
46 P(1) = C(K,L)
47 P(2) = C(K,K)
48 NSYS = 2
49 CALL SYSSLV
50 C(K,L) = P(1)
51 C(K,L) = P(2)
52 GO TO 90
53 70 IF(K .EQ. L) GO TO 80
54 T(1,1) = A(L,L)
55 T(1,2) = A(L,L)
56 T(1,3) = 0.
57 T(2,1) = A(L,L)
58 T(2,2) = A(L,L) + A(L,L)
59 T(2,3) = T(1,2)
60 T(3,1) = 0.
61 T(3,2) = T(2,1)
62 T(3,3) = A(L,L)
A-119
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(LL,LL) = P(3)
GO TO 90
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)
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 = DMAX(MAXA83(A(I,L)))  
10 CONTINUE  
16 10 CONTINUE  
17 IF(MAX .NE. 0.) GO TO 20  
18 A(L,N1) = 0.  
19 A(N1,L) = 0.  
20 GO TO 80  
21 SUM = 0.  
22 DO 30 I=L1,N  
23 A(I,L) = A(I,L)/MAX  
24 SUM = SUM + A(I,L)**2  
25 30 CONTINUE  
26 S = DSIGN(DSQR(SUM),A(L1,L))  
27 A(L,N1) = -MAX*S  
28 A(L1,L) = S + A(L1,L)  
29 A(N1,L) = S*A(L1,L)  
30 DO 50 J=L1,N  
31 SUM = 0.  
32 DO 40 I=L1,N  
33 SUM = SUM + A(I,L)*A(I,J)  
34 40 CONTINUE  
35 P = SUM/A(N1,L)  
36 DO 50 I=L1,N  
37 A(I,J) = A(I,J) - A(I,L)*P  
38 50 CONTINUE  
39 DO 70 I=1,N  
40 SUM = 0.  
41 DO 60 J=1,N  
42 SUM = SUM + A(I,J)*A(J,L)  
43 60 CONTINUE  
44 P = SUM/A(N1,L)  
45 DO 70 J=L1,N  
46 A(I,J) = A(I,J) - P*A(J,L)  
47 70 CONTINUE  
48 80 CONTINUE  
49 A(N-1,N1) = A(N,N-1)  
50 RETURN  
51 END
SUBROUTINE BCKMLT(A, U, N, NA, NU)
IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION A(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.
SUM = SUM + A(I,L)*U(I,J)
DO 10 I=J,N
SUM = SUM + A(I,L)*U(I,J)
P = SUM/A(N1,L)
DO 20 I=L1,N
U(I,J) = U(I,J) - A(I,L)*P
CONTINUE
DO 25 I=L1,N
U(I,L) = 0.
U(L,I) = 0.
CONTINUE
U(L,L) = 1.
CONTINUE
RETURN
END
SUBROUTINE SCHUR(HrU,NN,NH,NUPEPS,FAIL)
IMPLICIT REAL*8 (A-H, O-Z)
DIMENSION H(NH,1),U(NU,1)
INTEGER IFAIL
LOGICAL ILAST
N = NN
HN = 0.
DO 20 I=1,N
  JL = MAX0(1,I-1)
  RSUM = 0.
20 CONTINUE
RSUM = RSUM + DABS(H(I,J))
DO 10 J=JL,N
  RSUM = RSUM + DABS(H(I,J))
10 CONTINUE
HN = OMAX1(HN,RSUM)
IF(HN .EQ. 0.) GO TO 230
ITS = 0
NA = N-1
NM2 = N-2
DO 50 LL=2,N
  L = N-LL+2
  IF(DABS(H(L,L-1)) .LE. TEST) GO TO 60
50 CONTINUE
GO TO 70
60 H(L,L-1) = 0.
IF(L .LT. NA) GO TO 72
N = L-1
GOTO 50
70 X = H(N,N)/HN
Y = H(NA,NA)/HN
R = (H(N,NA)/HN)*(H(NA,N)/HN)
IF(ITS .LE. 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)
100 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) GO TO 110
110 GO TO 110
63 100 CONTINUE  
64 110 M2 = M+2  
65  M3 = M+3  
66 DO 120 I=M2,N  
67 H(I,I-2) = 0.  
68 120 CONTINUE  
69 IF(M3 .GT. N) GO TO 140  
70 DO 130 I=M3,N  
71 H(I,I-3) = 0.  
72 130 CONTINUE  
73 140 DO 220 K=M,N  
74 LAST = K.EQ.N  
75 IF(K .EQ. M) GO TO 150  
76 P = H(K,K-1)  
77 Q = H(K+1,K-1)  
78 R = 0.  
79 IF(.NOT.LAST) R = H(K+2,K-1)  
80 X = DABS(P) + DABS(Q) + DABS(R)  
81 IF(X .EQ. 0.) GO TO 220  
82 P = P/X  
83 Q = Q/X  
84 R = R/X  
85 150 S = DSQRT(P**2 + Q**2 + R**2)  
86 IF(P .LT. 0.) S = -S  
87 IF(K .NE. M) H(K,K-1) = -8S  
88 IF(K.EQ.M .AND. L.NE.M) H(K,K-1) = -H(K,K-1)  
89 P = P + S  
90 X = P/S  
91 Y = Q/S  
92 Z = R/S  
93 Q = Q/P  
94 R = R/P  
95 DO 170 J=K,NN  
96 P = H(K,J) + Q*H(K+1,J)  
97 IF(LAST) GO TO 160  
98 P = P + R*H(K+2,J)  
99 H(K+2,J) = H(K+2,J) = P*Z  
100 160 H(K+1,J) = H(K+1,J) = P*Y  
101 H(K,J) = H(K,J) = P*X  
102 170 CONTINUE  
103 J = MIN0(K+3,N)  
104 DO 190 I=1,J  
105 P = x*H(I,K) + Y*H(I,K+1)  
106 IF(LAST) GO TO 180  
107 P = P + Z*H(I,K+2)  
108 H(I,K+2) = H(I,K+2) = P*R  
109 180 H(I,K+1) = H(I,K+1) = P*Q  
110 H(I,K) = H(I,K) = P  
111 190 CONTINUE  
112 DO 210 I=1,NN  
113 P = x*U(I,K) + Y*U(I,K+1)  
114 IF(LAST) GO TO 200  
115 P = P + Z*U(I,K+2)  
116 U(I,K+2) = U(I,K+2) = P*R  
117 200 U(I,K+1) = U(I,K+1) = P*G  
118 U(I,K) = U(I,K) = P  
119 210 CONTINUE  
120 220 CONTINUE  
121 GO TO 40  
122 230 FAIL = 0  
123 RETURN  
124 END
SUBROUTINE SYSSLV
IMPLICIT REAL*8 (A-M, O-Z)
COMMON/SLVBLK/A(S, S), B(S), N
REAL*8 MAX
N1 = N + 1

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. 0.0) 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
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
A(K, J) = A(K, J)/A(K, K)
70 CONTINUE
80 CONTINUE
C INTERCHANGE THE COMPONENTS OF B.
100 DO 110 J = 1, N
INTR = A(N1, J)
IF(INTR .EQ. J) GO TO 110
TEMP = B(J)
B(J) = B(INTR)
B(INTR) = TEMP
110 CONTINUE
110 CONTINUE
C SOLVE LX = B.
200 R(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
A-125
63 C SOLVE UX = B.
64 C
65 300 DO 310 II=1,NM1
66   I = NM1-II+1
67    II = I+1
68    DO 310 J=II,N
69     B(I) = B(I) - A(I,J)*B(J)
70 310 CONTINUE
71 RETURN
72 END
SUBROUTINE GAUSEL (MAX, N, A, NR, B, IERR)

C FUNCTION
- COMPUTES SOLUTION TO A SET OF SIMULTANEOUS LINEAR EQUATIONS (DOES NOT GIVE PIVOT OR DETERMINANT DATA)

C USAGE
- CALL GAUSEL (MAX, N, A, NR, B, IERR)

C PARAMETERS
- MAX - ORDER OF A
- A(N,N) - INPUT MATRIX OF COEFFICIENTS (DESTROYED)
- NR - NUMBER OF COLUMNS IN A
- B(MAX, NR) - MATRIX OF CONSTANTS (REPLACED BY SOLUTIONS)
- IERR - INTEGER ERROR CODE
  = 0 NORMAL RETURN

C REQUIRED ROUTINES
- NONE

C SOURCE
NASA, LRC, ANALYSIS AND COMPUTATION DIVISION PROGRAM LIBRARY

C ***
C DIMENSION A(N,N), B(MAX, NR)
C N1 = N-1
C IF (N1 .EQ. 0) GO TO 140
C
C *** FIND LARGEST REMAINING ELEMENT IN I-TH COLUMN FOR PIVOT
C DO 100 I=1, N1
C BIG = 0.
C DO 20 K=1, N
C TERM = DABS(A(K, I))
C IF (TERM .GT. BIG) 20, 20, 10
C 10 BIG = TERM
C L = K
C 20 CONTINUE
C IF (BIG) 40, 50, 40
C IERR = 2
C RETURN
C 30 CONTINUE
C IF (I-L) 50, 60, 50
C RETURN
C 40 CONTINUE
C PIVOT ROWS OF A AND B
C DO 50 J=1, N
C TEMP = A(I, J)
C A(I, J) = A(L, J)
C A(L, J) = TEMP
C 50 CONTINUE
C DO 60 J=1, NR
C TEMP = B(I, J)
C B(I, J) = B(L, J)
C B(L, J) = TEMP
C 60 CONTINUE
C STORE PIVOT AND PERFORM COLUMN OPERATIONS ON A AND B
C IP1 = I+1
C DO 100 II=IP1, N
C A(II, I) = A(II, I) / A(I, I)
C X3 = A(II, I)
C DO 90 K=IP1, N
C A(II, K) = A(II, K) - X3 * A(I, K)
C 90 CONTINUE
C A=127
DO 100 K = 1, NR
  B(I, K) = B(I, K) - 3 * 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 K K = 1, NM1
  I = N - K
  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

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 PNCN (A, NA, NAM, IOP)
C IOP(1)=0, SKIP TITLE; IOP(2)=N, SKIP LINES; IOP(3)=1, TAB 25 SPACES.
DIMENSION A(N), IOP(4), NA(2)
IMPLICIT REAL*8 (A-H, O-Z)
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 FORMAT(2X)
205 CONTINUE
IF (IOP(1).EQ.0) GO TO 210
WRITE(7, 151) NAM, NR, NC
151 FORMAT(A4, //, 215)
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)
REAL*4 A(2), DIMAG
COMPLEX*16 Z, B
EQUIVALENCE (A, B)
B = Z
DIMAG = A(2)
RETURN
END
FUNCTION DREAL(Z)
REAL*8 A(2),DREAL
COMPLEX*16 Z,B
EQUIVALENCE (A,B)
B=Z
DREAL=A(1)
RETURN
END
0  BLOCK DATA
1  IMPLICIT REAL*4 (A-H,O-Z)
2  COMMON/LINES/TITLE(10),TIL(3),NLP,LIN
3  COMMON/FORM/FMT1(2),FMT2(2),NEPR
4  COMMON/TOL/EPSAM,EPS8M,IACM
5  COMMON/CONV/SMCV,RICTCV,SECV,MAXSUM
6  DATA LIN,LP/1,58/
7  DATA NEPR,FMT1/7,8H(1P7D16,.8H7) /  
8  DATA TIL/RM ORA,8HCLS PRO,8HGRAM / 
9  DATA FMT2/8M(3X,1P7D,9H16,.7) /
10  DATA EPSAM/1,E-10/
11  DATA EPS8M/1,E-10/
12  DATA IACM/12/
13  DATA SMCV/1,E-8/
14  DATA RICTCV/1.E-8/
15  DATA SECV/1,E-8/
16  DATA MAXSUM/50/
17  END