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
FESTAL REPORT
NASA CRAM: NSG-1414, Suppl. 2
THE DYNAMICS AND CONTROL OF LARGE FLEXIBLE SPACE STRUCTURES-III
PART A: SHAPE AND ORIENTATION CONTROL OF A PLATFORM IN ORBIT USING POINT ACTUATORS

(NASA-CR-163253) THE DYNAMICS AND CONTROL OF LARGE FLEXIBLE SPACE STRUCTURES, 3. PART II: SHAPE AND ORIENTATION CONTROL OF A PLATFORM IN ORBIT USING POINT ACTUATORS.
Final Report (Howard Univ.) 179 p
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
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>Chapter</th>
<th>Table No.</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chapter I</td>
<td>Table 1</td>
<td>Study Plan 1977-1980</td>
<td>I-3</td>
</tr>
<tr>
<td>Chapter II</td>
<td>Table 1</td>
<td>Maximum Force Amplitudes (Newtons) for Different Combinations of Cases with Actuator Locations</td>
<td>II-6</td>
</tr>
<tr>
<td>Chapter III</td>
<td>Table 1</td>
<td>Evaluation of Parameters in Frequency Expression (Warburton)</td>
<td>III-10</td>
</tr>
<tr>
<td></td>
<td>Table 2</td>
<td>Frequencies and Nodal Patterns Obtained by the Three Methods</td>
<td>III-11</td>
</tr>
<tr>
<td></td>
<td>Table 3</td>
<td>Normalized Deflections at the Nodal Points Obtained by the Three Methods—Second Mode Cases 1-3</td>
<td>III-12</td>
</tr>
<tr>
<td></td>
<td>Table 4</td>
<td>Normalized Deflections at the Nodal Points Obtained by the Three Methods—Second Mode Case 4</td>
<td>III-13</td>
</tr>
</tbody>
</table>
List of Figures by Chapters

<table>
<thead>
<tr>
<th>Fig No.</th>
<th>Caption</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Chapter II</strong></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><strong>Chapter III</strong> (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. \( \phi_x^{(n)} = 0 \)

✓ (2). The Case of No Yaw: i.e. \( \psi = 0 \)

C. Determination of Modal Shape Functions and Frequencies for Different Structural Models

✓ (1). Circular Homogenous Membrane

✓ (2). Rectangular Homogenous Membrane

✓ (3). Rectangular Homogenous Plate (and Square Plate)

✓ (4). Circular Homogenous Plate

✓ (5). Shallow Spherical Shell Structure

D. Implementation of One or More of the Structural Models for Digital Simulation

✓ (1). Rectangular Homogenous Plate

✓ (2). Thin-Homogenous Beam with Stabilizing Dumbbell (Local Horizontal Orientation)

✓ (3). Square Plate with Stabilizing Dumbbell (Local Horizontal Orientation)

✓ (4). Shallow Spherical Shell Structure with Stabilizing Dumbbell

✓ (5). Circular Homogenous Plate with Stabilizing Dumbbell

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(P=N)

✓ (2). Establish relationship between P and N according to controllability requirements (applications of theorems developed by Balas) P can be less than N. (Applied to continuum beam model 1978-79).

✓ (3). Selection of control system gains - considers both position and rate feedback. (Applied to continuum beam model 1978-79)

✓ a. Develop criteria for complete decoupling of linearized controlled equations using the fundamental theorem of a system of N linear equations and P unknowns

For unique solution of gains, P=N consistent with modal control; for non unique solution P<N

✓ b. Application of linear regulator problem to the original linearized and/or transformed equations

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

IP C. Modelling of Sensors—the Problem of Observability

D. Treatment of Observation and Control Spillover

✓ Items completed

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

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

Part B of this report, under separate cover, concentrates on the mathematical modelling and analysis of more complex structures such as beams and plates with connected gimballed dumbbells to provide gravitational stability about the local horizontal orientation, and also the analysis of the dynamics of a shallow shell-type structure in orbit.
I.1 References - Introduction


II. CONTROL OF A LARGE FLEXIBLE PLATE IN ORBIT

ABSTRACT

The dynamics and attitude and shape control of a large thin flexible platform in orbit are studied. Attitude and shape control is assumed to result from actuators placed perpendicular to the main surface and one edge and their effect on the rigid body and elastic modes is modelled to first order. The equations of motion are linearised 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 sail systems, and as possible collectors of solar energy for transmittal 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 attitude 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 modelling 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. 1, 3, 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:

A. Rotational Equations of Motion:

\[
\begin{align*}
\dot{\alpha}_x &= \frac{1}{I_x} \left( \tau_x - \frac{G_x}{2} \alpha_x + \frac{C_x}{2} \alpha_y \right) \\
\dot{\alpha}_y &= \frac{1}{I_y} \left( \tau_y - \frac{G_y}{2} \alpha_y + \frac{C_y}{2} \alpha_z \right) \\
\dot{\alpha}_z &= \frac{1}{I_z} \left( \tau_z - \frac{G_z}{2} \alpha_z + \frac{C_z}{2} \alpha_x \right)
\end{align*}
\]

(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:

\[
\begin{align*}
\dot{\psi} &= \dot{\psi} + (\dot{\psi} + \dot{\omega}_z) \sin \theta \\
\dot{\theta} &= (\dot{\psi} + \dot{\omega}_z) \cos \psi \cot \phi + \dot{\theta} \cos \theta \\
\dot{\phi} &= \dot{\phi} - (\dot{\psi} + \dot{\omega}_z) \sin \theta \cos \phi
\end{align*}
\]

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

\[ \begin{align*}
G_{m_x} &= 3u_c^2(I_x-I_y) + (-csc+ecsc)\cos(csc+csc)
G_{m_y} &= 3u_c^2(I_y-I_z) + (-csc+ecsc)\cos(csc+csc)
G_{m_z} &= 3u_c^2(I_z-I_x) + (-csc+ecsc)\cos(csc+csc)
\end{align*} \tag{3} \]

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

### B. Generic Mode Equations

The generic modal equations may be obtained for each of the three nominal orientations considered in terms of the modal amplitude \((A_\phi)^T\).

#### For Case (i)

\[ \ddot{A}_x + [u_c^2-(u_c^2+u_c^2)\omega^2_\phi]A_x = E_x/N_x \tag{4a} \]

where

\[ M_{x\phi} = u_c^2 [3(2u_c^2+ecsc^2+2u_c^2+2csc^2+4csc^2+4csc^2)] \]

#### For Case (ii)

\[ \ddot{A}_x + [u_c^2-(u_c^2+u_c^2)\omega^2_\phi]A_x = E_x/N_x \tag{4b} \]

\[ M_{xy} = u_c^2 [3(2u_c^2+4csc^2+4csc^2+4csc^2)] \]

#### For Case (iii)

\[ \ddot{A}_x + [u_c^2-(u_c^2+u_c^2)\omega^2_\phi]A_x = E_x/N_x \tag{4c} \]

\[ M_{yy} = u_c^2 [3(2u_c^2+4csc^2+4csc^2+4csc^2)] \]

### C. Linearisation

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

\[ \begin{align*}
\ddot{\phi} &= \omega^2 \begin{bmatrix} I_y & -I_x \\ I_x & I_y \end{bmatrix} \omega^2 - 2u_c^2 \begin{bmatrix} \omega^2_\phi & -\omega^2_\phi \\ -\omega^2_\phi & \omega^2_\phi \end{bmatrix} \omega^2 + 2C \begin{bmatrix} I_x \\ I_y \end{bmatrix} \\
\dot{\theta} &= \omega^2 \begin{bmatrix} I_y & I_x \\ I_y & I_x \end{bmatrix} \omega^2 + 2u_c^2 \begin{bmatrix} \omega^2_\phi & \omega^2_\phi \\ -\omega^2_\phi & -\omega^2_\phi \end{bmatrix} \omega^2 + 2C \begin{bmatrix} I_x \\ I_y \end{bmatrix} \\
\ddot{\theta} &= 3u_c^2 \begin{bmatrix} I_y & -I_x \\ I_x & I_y \end{bmatrix} \omega^2 + 2C \begin{bmatrix} I_x \\ I_y \end{bmatrix}
\end{align*} \tag{5} \]

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): \( I_x = I_y \) and \( I_z = 2I_x = 2I_y \)
- Case (iii): \( I_x = I_y = I_z = 2I_x = 2I_y \) \tag{6} \]

For small amplitude angles the generic mode equations become:

- Case (i): \( A_x + (u_c^2-2u_c^2)A_x = E_x/N_x \)
- Case (ii): \( A_x + (u_c^2-3u_c^2)A_x = E_x/N_x \) \tag{7} \]

### D. Modelling of Point Actuators

For an actuator which can generate a force of the type

\[ \mathbf{T} = f_x \hat{i} + f_y \hat{j} + f_z \hat{k} \tag{8} \]

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

\[ \mathbf{T} = H \mathbf{T} \tag{9} \]

where \( H = \frac{M_0 + \hat{y} \hat{z} - \hat{z} \hat{y}}{M_0} \) describes the position of the actuator on the surface (or edge) of the plate. Actuators can be placed perpendicular to the \(XY\), \(YZ\) or \(XZ\) planes of the plate, so for an actuator whose force axis is perpendicular to the \(XY\) plane the torque is given by (since \( f_x = f_y = 0 \))

\[ \mathbf{T} = -f_x \hat{i} - f_z \hat{k} \tag{10} \]

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

\[ \mathbf{T} = f_y \hat{j} \tag{11} \]

For an actuator perpendicular to the \(XZ\) plane, the torque is given by (since \( f_y = f_z = 0 \))

\[ \mathbf{T} = f_x \hat{i} \tag{12} \]

The generic force due to an \( r^{th} \) actuator on the \( n^{th} \) mode is given by

\[ F_x = \int W(x,y) f_x(x,y) \mathrm{d}x \mathrm{d}y \]

\[ F_y = \int W(x,y) f_y(x,y) \mathrm{d}x \mathrm{d}y \]

\[ F_z = \int W(x,y) f_z(x,y) \mathrm{d}x \mathrm{d}y \]

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

For \( n \) actuators placed on the \( XY \) plane of the plate with force axes normal to that deformed surface, the generic force on \( n^{th} \) mode is given by

\[ F_x = \sum_{i=1}^{n} F_{x_i} + f_x(x,y) f_i(x,y) \]

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

### E. Modelling of Distributed Actuators

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

\[ \mathbf{T} = f_x(x,y,z,t) \hat{i} + f_y(x,y,z,t) \hat{j} + f_z(x,y,z,t) \hat{k} \tag{15} \]

where the force components are now both spatially and time dependent.
The torque due to such an actuator is given by

$$T = R\tilde{F}$$

(16)

The total torque is given by

$$T = \int (R\tilde{F}) \, dx \, dy \, dz$$

(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),

$$f_x(x,y,z) = \sum_{n=1}^{N} \sum_{m=1}^{M} f_{x,m}^{n}(x,y,z)g_{m}^{n}(t)$$

(18)

The integral for the torque is then given by

$$T = \int \left[ \sum_{n=1}^{N} \sum_{m=1}^{M} f_{x,m}^{n}(x,y,z)g_{m}^{n}(t) \right] \, dx \, dy \, dz$$

(19)

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

$$F_t = \int [U_t(x,y)] \sum_{n=1}^{N} \sum_{m=1}^{M} f_{x,m}^{n}(x,y,z)g_{m}^{n}(t) \, dx \, dy \, dz$$

(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 \, \text{kg} \cdot \text{m}^2
- Maximum Moment of Inertia = 4.7088x10^7 \, \text{kg} \cdot \text{m}^2

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

$$\omega_c = 1.23 \times 10^{-3} \, \text{rad/sec}$$

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

$$\omega_1 = 2.0931947 \times 10^{-2} \, \text{rad/sec}$$
$$\omega_2 = 3.0404741 \times 10^{-2} \, \text{rad/sec}$$
$$\omega_3 = 3.9088122 \times 10^{-2} \, \text{rad/sec}$$

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

Case (I): \(I_x = I_y, I_x = 2I_y \Rightarrow 2I_y \)

The rotational equations of motion and the generic modal equations are non-dimensionalized by the orbital period and the length variable \((r = \omega_c, \tau = A/\omega_c, \theta = \theta/\omega_c, \text{etc})\)

$$\theta'' = \left[ (I_y-I_z-I_x)/\omega_c I_x \right] \theta' + \left[ (I_x-I_z)/I_x \right] \theta$$

(22)

$$\theta'' = \left[ (I_x-I_z)/\omega_c I_x \right] \theta' + \left[ (I_y-I_z)/I_x \right] \theta$$

(23)

The generic mode equations become:

$$z'' = -\left( \omega_c^2 / \omega_c^2 \right)^2 z$$

(24)

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

$$\phi'' = -(2/\omega_c) \phi'$$

(25)

$$\psi'' = -(2/\omega_c) \psi'$$

(26)

The characteristic equation for the system (25) and (26) is,

$$s^2(s^2-1+2/\omega_c^2) = 0$$

(27)

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

$$\phi'' = -(1/\omega_c) \phi'$$

(27)

$$\psi'' = -(2/\omega_c) \psi'$$

(28)

$$\theta'' = 3\theta$$

(29)

The generic mode equations can be represented by,

$$z'' = -\left[ (\omega_c^2 / \omega_c^2)^2 \right]^2 z$$

(30)

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

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

$$s^2(s^2-1+2/\omega_c^2) = 0$$

(31)

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

Case (III): \(I_x = I_y, I_x = 2I_z \Rightarrow 2I_z \)

The rotational equations of motion are

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

(32)

while the generic mode equations can be expressed by,

$$z'' = -\left[ (\omega_c^2 / \omega_c^2)^2 \right]^2 z$$

(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:

\[ \dot{X} = AX + BU \]  

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

\[ X = (x_1, x_2, x_3, \ldots, x_{n+6}, x_{n+6+1}, \ldots, x_{2n+6})^T \]

and

\[ x_1 = \psi; x_2 = \psi; x_3 = \theta; \quad x_{1+i} = \Delta_i \omega_i; \]

\[ i = 1, 2, \ldots, n \text{ generic modes} \]

\[ x_{n+6+i} = \xi_i; \quad x_{n+6+3i} = \xi_i; \quad x_{n+6+4i} = \xi_i; \quad i = 1, 2, \ldots, n \]

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:

\[ A = \begin{bmatrix} \text{diagonal} & \text{0} \\ \text{0} & \text{0} \end{bmatrix} \]

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

\[ A_{7,1} = 4(\xi_i - \xi_j)/I_x; \quad A_{8,2} = -(\xi_i - \xi_j)/I_x; \]

\[ A_{9,3} = 3(\xi_i - \xi_j)/I_y; \quad A_{10,4} = -(\omega_1/\omega_2)^2; \]

\[ A_{11,5} = (\omega_2/\omega_1)^2; \quad A_{12,6} = -(\omega_3/\omega_4)^2; \]

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

The general \( B \) matrix:

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

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

Control Law Selection

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

\[ \dot{X} = AX + BU \]

where

\[ X = (x_1, x_2, \ldots, x_{n+6}) \]

After selecting \( U = K_1X + K_2x \) we can rewrite the controlled motion equations as

\[ \dot{X} = (A+K_1X + K_2x)X \]

\[ K_1 \text{ and } K_2 \text{ are evaluated such that } (A+K_1X + K_2x) \text{ are diagonalized and thus yield required damping and frequency of the controlled modes.} \]

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 (1)**

- First Location
  - \( a = 100\text{m} \)
  - \( f_1(-a/6,0); f_2(a/6,0); f_3(-a/6,0) \)
  - \( f_4(a/6,0); f_5(-a/6,0); f_6(a/6,0) \)

- Second Location
  - \( a = 100\text{m} \)
  - \( f_1(-a/2,0); f_2(a/2,0); f_3(-a/2,0) \)
  - \( f_4(a/2,0); f_5(-a/2,0); f_6(a/2,0) \)

**Case (ii)**

- First Location
  - \( a = 100\text{m} \)
  - \( f_1(-a/6,0); f_2(0,-a/6); f_3(0,a/6) \)
  - \( f_4(0,a/6); f_5(a/6,0); f_6(a/6,0) \)

- Second Location
  - \( a = 100\text{m} \)
  - \( f_1(-a/2,0); f_2(a/2,0); f_3(-a/2,0) \)
  - \( f_4(a/2,0); f_5(-a/2,0); f_6(a/2,0) \)

**Case (iii)**

- First Location
  - \( a = 100\text{m} \)
  - \( f_1(-a/6,0); f_2(0,-a/6); f_3(0,a/6) \)
  - \( f_4(0,a/6); f_5(a/6,0); f_6(a/6,0) \)

- Second Location
  - \( a = 100\text{m} \)
  - \( f_1(-a/2,0); f_2(a/2,0); f_3(-a/2,0) \)
  - \( f_4(a/2,0); f_5(-a/2,0); f_6(a/2,0) \)

Actuator positions for the two different sets of orientations are illustrated in Fig. 3. The system A and B matrices corresponding to different combinations of the three platform orientations and the two sets of actuator location are listed as follows.
Case (i) Platform Following Local Vertical With Major Surface Normal to the Orbit Plane.

The non zero elements of the A matrix are:

\[ A_{1,4} = -1 \text{ for } i = 1, \ldots, 6; \ A_{7,8} = 800; \ A_{8,7} = -1600; \ A_{8,2} = 1; \ A_{9,3} = -3; \ A_{10,4} = 277.414; \ A_{11,5} = -588.647; \ A_{12,6} = -976.844. \]

Case (ii) Platform Along Local Horizontal

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

\[ A_{7,1} = 4; \ A_{7,6} = 1600; \ A_{8,2} = 0; \ A_{8,7} = -800; \ A_{9,3} = 3. \]

Case (iii) Major Surface in Orbit Plane

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

\[ A_{7,1} = -4; \ A_{7,6} = 0; \ A_{8,2} = -1; \ A_{8,7} = 0; \ A_{9,3} = 0. \]

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

As an example of the time history of the required control forces, Fig. 4b shows such a time response for the exterior (12th) location of the actuators with the platform nominally following the local vertical and the major surface area of the platform in the orbital plane. A complete summary of the maximum force amplitudes required for all combinations of actuator locations and platform orientations is given in Table 1. In interpreting the results of Table 1, it should be pointed out that, in the process of achieving both orientation and shape control, the maximum force(s) required of any actuator will vary with both the moment arm about the principal body axes and the value of the modal shape function at the particular actuator location for all modes contained within the mathematical model.

(b) Stabilizing the System by Pole Clustering

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

\[ \dot{x} = Ax + Bu \]

\[ x = 2(n+3)x1 \]
The control, \( U = -K X \) is selected by using a digital computer algorithm such that \((A - KX)\) has the required identical negative real part in each of its eigenvectors. 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 when 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: \( f_1 (-\alpha/6, 0, 0) \); \( f_2 (\alpha/6, -\alpha/6, 0) \); \( f_3 (-\alpha/6, 0, 0) \); and \( f_4 (\alpha/2, \alpha/6, 0) \) where \( \alpha = 100\% \). It is assumed that the minimum damping requirement on the system has a time constant of \((13.33\, \text{min} \times 1/20\, \text{dimensional orbital time})\). The control influence matrix is then calculated based on the assumed coordinates of the four actuators. The control \( U = -KX \) is calculated by the ORACLS pole clustering algorithm. Based on these gains time histories of the required control forces are then obtained.

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

| B matrix (Lower Part) | \[0 \quad 0 \quad 0 \quad -0.22645 \]
| \[-0.4529 \quad -0.4529 \quad 0.0 \quad 0.0 \]
| \[0.4529 \quad -0.4529 \quad 0.0 \quad 0.0 \]
| \[0.003126 \quad -0.003126 \quad 0.0 \quad 0.0 \]
| \[0.0 \quad 0.0 \quad -0.0030844 \quad 0.0 \]
| \[0.008786 \quad -0.008786 \quad -0.0115 \quad 0.0 \]

The real part is \(-1.0\) and the imaginary parts are \(+0.9933, +16.82, +24.26, +31.33\) and \(+132.37\). The maximum force amplitudes (Newtons) are calculated as:

\[
|f_1| = 78.5, \quad |f_2| = 36.4, \quad |f_3| = 169.5, \quad \text{and} \quad |f_4| = 35.3.
\]

An interesting comparison can now 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 \(13.01\) with \(976 = 11.24\)). Depending on the nature of the expected disturbance forces this result could be very undesirable.

(c) Application of the Linear Regulator Theory

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

\[
J = \int_{0}^{\infty} (X^TQX + U^TRU)dt
\]

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

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

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

\[
\begin{pmatrix}
\omega_1^2 \\ \\ \\ -\omega_2^2
\end{pmatrix} = \begin{pmatrix} 0 & a \\ -b & 0 \end{pmatrix} \begin{pmatrix} \omega_1 \\ \omega_2 \end{pmatrix}
\]

where the system oscillates at the frequency \( \omega = \sqrt{ab} \). It is desired that the control remove a maximum "transverse" angular rate, \( \omega_{\text{max}} = \max \sqrt{\omega_2^2 + \omega_3^2} = \sqrt{\omega_2(0)^2 + \omega_3(0)^2} \)
so that a strategy for selecting the elements of Q could be

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

(41)

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

\[ \begin{bmatrix} x_1' \\ x_2' \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -(w_1/w_c)^2 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \]  

(42)

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

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

\[ Q_{1,1} = 4.32 \times 10^3 \], \( Q_{2,2} = 8.539 \times 10^3 \),
\[ Q_{3,3} = Q_{9,9} = 3.0 \times 10^6 \], \( Q_{4,4} = Q_{10,10} = 2.77414 \times 10^6 \),
\[ Q_{5,5} = Q_{11,11} = 5.88647 \times 10^6 \],
\[ Q_{6,6} = Q_{12,12} = 9.74844 \times 10^6 \], \( Q_{7,7} = Q_{8,8} = 2.222 \times 10^3 \).

The \( R \) matrix is chosen as an identity matrix. A parametric study is done using various multiples of the \( Q_a \) and \( R \) 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 amplitude of the forces for \( R = I \) and \( R = 1000 I \) are calculated and plotted in Fig. 6. The closed loop poles of the controlled system at points (1) and (2) are virtually the same and are given as follows (nondimensionalized):

\[ -1.0043, -1.8216.64, -2.16424.20, -17.19419.79, -26.23, -36.22, -137.64 \]  

and \[ -38.66 \text{ to } -38.66 \times 1132.11 \]

The maximum force amplitudes as shown in Fig. 6 are less than those corresponding to Case (1) - Location 1 of Table 1 for comparable transient response, 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 advantage that they can be applied to situations where the number of actuators is less than the number of modes in the mathematical model, in contrast to the decoupling technique of Section (a).

5. Conclusions

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

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

a) The decoupling of the original state equations using state variable feedback eliminates the need of a transformation from the original coordinates to the modal coordinates and provides a method of specifying directly the amount of damping and frequency of the individual components of the state vector. However, with this technique the number of actuators must be equal to the number of coordinates (modes) in the model.

b) The pole placement algorithm (ORACLSS) guarantees the over-all required damping of the system and does not restrict the number of actuators to be equal to the number of modes in the model. However, it is seen that the closed-loop frequencies may be greatly increased when compared to the open-loop values which may cause problems with externally induced periodic excitations.

c) The linear regulator theory can provide acceptable performance once the state and penalty matrices are properly selected, and the number of actuators can be less than the number of modes in the model. Computer capacity and accuracy limit the number of modes that can be considered. Here, too, an undesirable increase in the closed-loop frequencies may result in order to provide satisfactory responses with maximum allowable force amplitudes.

References


Appendix A - Nomenclature

\( A_T \) \( r \)th modal amplitude function
\( B \) Control influence matrix
\( C_x, C_y, C_z \) Disturbance torques about the principal undeformed body axes
\( E_T \) Generic force on \( r \)th mode
\( f \) Force due to an actuator
\( f_x, f_y, f_z \) Force components due to an actuator
\( G_{x}, G_{y}, G_{z} \) Gravity gradient torques about the principal undeformed body axes
\( I_x, I_y, I_z \) Moments of inertia about the principal axes
\( K \) Gain matrix
\( K_T, K_P \) Rate and position feedback gain matrices

\( \lambda_t \) \( r \)th modal mass
\( T \) Torque due to an actuator
\( T_x, T_y, T_z \) Torque components
\( W_r(x, y) \) \( r \)th modal shape function
\( \lambda_T \) Nondimensionalized \( r \)th modal amplitude function
\( \omega_c \) Orbital frequency
\( \omega_x, \omega_y, \omega_z \) Angular body rates
\( \phi, \psi, \theta \) Roll, yaw, pitch, respectively
\( \omega_1, \omega_2, \omega_3 \) First three modal frequencies of the plate
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. 3. Location of two sets of actuators (I & II)
Fig 4a. Controlled state response for all combinations of orientations and actuator locations.

\[ \theta(0) = \psi(0) = \phi(0) = 0.01 \]
\[ z_i(0) = 0.01, \ i = 1,2,3 \]
\[ \dot{\theta}(0) = \dot{\psi}(0) = \dot{\phi}(0) = 0 \]
\[ \ddot{z}_i(0) = 0, \ i = 1,2,3 \]

(20% critical damping)

(\(z_2, z_3\): 10% critical damping)
Fig 4b. Control force time history for Case (iii) - II
Fig. 5. Variation of least damped mode negative real part with α and R.
Fig 6. Maximum force amplitudes as a function of $a$ 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

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

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

\( \theta(x) \) 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

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

and

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

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

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

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

2. Formulation by Lemke\(^2\)

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

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

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

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

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

\( \phi_n(y) \) 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

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

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

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

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.

\[
W(x,y) = x_1 y_1 + 0.0325 (x_1 y_3 + x_3 y_1) - 0.005 x_3 y_3
\]
\[
- 0.00257 (x_1 y_5 + x_5 y_1) + 0.00121 (x_3 y_5 + x_5 y_3)
\]
\[
- 0.000365 x_5 y_5 + \ldots
\]

and

\[
\omega = \frac{13.086}{a^2} \sqrt{\frac{E \cdot h^3}{12 \cdot \rho (1-\sigma^2)}} \quad \text{for } \sigma = 0.343
\]
3. **Finite Element Computer Program**

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

4. **Discussion of Numerical Results**

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

\[
\text{density} = 2768.0 \text{ kg/m}^3 \\
\text{Young's modulus} = 0.7441 \times 10^{10} \text{ kg/m}^2 \\
\text{Poisson's ratio} = 0.33
\]
Using Warburton's results, Eq. (III-4), Eq. (III-5), Table (III-1), and expressions for $\theta(x)$ and $\phi(y)$, frequencies and mode shapes are calculated for different combinations of the number of nodal lines, $m$ and $n$, starting with combinations of $m=0$ and $n=1$, through $m=3$ and $n=3$. The first three combinations of nodal line numbers, $(0,0)$, $(1,0)$ and $(0,1)$, represent rigid body motion. The first fundamental flexural frequency is seen to be due to a combination of $(1,1)$. 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>$\frac{3}{2}$</td>
<td>$m - \frac{1}{2}$</td>
<td>$(m - \frac{1}{2})^2P$</td>
<td>$(m - \frac{1}{2})^2Q$</td>
<td>$\frac{3}{4}$</td>
<td>$n - \frac{1}{2}$</td>
<td>$(n - \frac{1}{2})^2P$</td>
<td>$(n - \frac{1}{2})^2Q$</td>
</tr>
</tbody>
</table>

$P = \frac{1-2}{(m-0.5)\pi}$  
$Q = \frac{1+6}{(m-0.5)\pi}$
<table>
<thead>
<tr>
<th>STRUDL NUMBER OF ELEMENTS</th>
<th>4</th>
<th>9</th>
<th>16</th>
<th>36</th>
<th>64</th>
<th>WARBURTON Freq. (cps)</th>
<th>(m,n)</th>
<th>LIME Freq. (cps)</th>
<th>Nodal Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.003165</td>
<td>0.003284</td>
<td>0.003315</td>
<td>0.00334</td>
<td>0.003358</td>
<td>0.003471</td>
<td>(1,1)</td>
<td>0.003421</td>
<td>0.003421</td>
</tr>
<tr>
<td>2</td>
<td>0.004291</td>
<td>0.004664</td>
<td>0.004776</td>
<td>0.004839</td>
<td>0.004856</td>
<td>0.004846</td>
<td>(2,0)-(0,0)</td>
<td>0.006175</td>
<td>0.006175</td>
</tr>
<tr>
<td>3</td>
<td>0.005549</td>
<td>0.006133</td>
<td>0.006221</td>
<td>0.006221</td>
<td>0.006207</td>
<td>0.006175</td>
<td>(2,0)+(0,2)</td>
<td>0.015744</td>
<td>0.015744</td>
</tr>
<tr>
<td>4</td>
<td>0.007619</td>
<td>0.008453</td>
<td>0.008632</td>
<td>0.008690</td>
<td>0.008672</td>
<td>0.00400</td>
<td>(2,1)</td>
<td>0.01767</td>
<td>0.01767</td>
</tr>
<tr>
<td>5</td>
<td>0.007619</td>
<td>0.008453</td>
<td>0.008632</td>
<td>0.008690</td>
<td>0.008692</td>
<td>0.00400</td>
<td>(1,2)</td>
<td>0.01944</td>
<td>0.01944</td>
</tr>
<tr>
<td>6</td>
<td>0.01323</td>
<td>0.01512</td>
<td>0.01600</td>
<td>0.01596</td>
<td>0.01579</td>
<td>0.01542</td>
<td>(3,0)</td>
<td>0.01715</td>
<td>0.01715</td>
</tr>
<tr>
<td>7</td>
<td>0.01512</td>
<td>0.01600</td>
<td>0.01596</td>
<td>0.01579</td>
<td>0.01542</td>
<td>(0,3)</td>
<td>0.01656</td>
<td>(2,2)</td>
<td>0.01656</td>
</tr>
<tr>
<td>8</td>
<td>0.01584</td>
<td>0.01618</td>
<td>0.01615</td>
<td>0.01607</td>
<td>0.01656</td>
<td>(2,2)</td>
<td>0.01656</td>
<td>(2,2)</td>
<td>0.01656</td>
</tr>
<tr>
<td>9</td>
<td>0.01630</td>
<td>0.01727</td>
<td>0.01750</td>
<td>0.01746</td>
<td>0.01715</td>
<td>(3,1)-(1,3)</td>
<td>0.01715</td>
<td>(3,1)-(1,3)</td>
<td>0.01715</td>
</tr>
<tr>
<td>10</td>
<td>0.01843</td>
<td>0.02022</td>
<td>0.02031</td>
<td>0.02005</td>
<td>0.0208</td>
<td>(1,3)+(3,3)</td>
<td>0.0208</td>
<td>(1,3)+(3,3)</td>
<td>0.0208</td>
</tr>
<tr>
<td>11</td>
<td>0.02465</td>
<td>0.02693</td>
<td>0.02693</td>
<td>0.02700</td>
<td>0.02726</td>
<td>(3,2)</td>
<td>0.02726</td>
<td>(3,2)</td>
<td>0.02726</td>
</tr>
<tr>
<td>12</td>
<td>0.02465</td>
<td>0.02673</td>
<td>0.02693</td>
<td>0.02700</td>
<td>0.02726</td>
<td>(2,3)</td>
<td>0.02726</td>
<td>(2,3)</td>
<td>0.02726</td>
</tr>
<tr>
<td>13</td>
<td>0.03362</td>
<td>0.03029</td>
<td>0.03162</td>
<td>0.03104</td>
<td>0.0344</td>
<td>(8,3)</td>
<td>0.0344</td>
<td>(8,3)</td>
<td>0.0344</td>
</tr>
</tbody>
</table>

**TABLE-III-2.** Frequencies and Nodal Patterns obtained by the Three Methods.
<table>
<thead>
<tr>
<th>Location</th>
<th>STRUDL</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>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>
</tbody>
</table>

grid used for 36 elements

Case 2: 9 Elements

<table>
<thead>
<tr>
<th>Location</th>
<th>STRUDL</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>STRUDL</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>STRUDL</th>
<th>WARBU-RTON</th>
<th>LEMKE</th>
<th>LOCATION</th>
<th>STRUDL</th>
<th>WARBU-RTON</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>0.1334</td>
<td>0.1475</td>
<td>0.1267</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>-0.1334</td>
<td>-0.1775</td>
<td>-0.1207</td>
</tr>
<tr>
<td>B4</td>
<td>0.4933</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
(2,0) + (0,2)

Fig. 5: Fourth Mode
(2,1)

Fig. 6: Sixth Mode
(3,0)

Fig. 7: Eighth Mode
(2,2)
Fig. 8: Ninth Mode

Fig. 9: Tenth Mode
IV. GENERAL CONCLUSIONS AND RECOMMENDATIONS

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

(1) the decoupling of the original state equations using state variable feedback;

(2) a pole placement algorithm; and

(3) an application of the linear regulator theory

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

As a logical extension to the present study which assumes perfect instantaneous knowledge of the state, the modelling of the sensor dynamics and related problem of observability should be considered, once specific information on the types of sensors required for monitoring the performance of large flexible systems is available.
The problems caused by both observation and control spillover could also be treated, perhaps by beginning with the simpler model of the control of a long, flexible beam in orbit and then extending this analysis to the three dimensional model of the platform.

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

Modifications to ORACLS Software Package

The ORACLS\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>(1) TESTSTA</td>
<td>TESTSA</td>
</tr>
<tr>
<td>(2) VARANCE</td>
<td>VARANC</td>
</tr>
<tr>
<td>(3) TRANSIT</td>
<td>TRNSIT</td>
</tr>
<tr>
<td>(4) DISCREG</td>
<td>DISREG</td>
</tr>
<tr>
<td>(5) CNTNREG</td>
<td>CNTREG</td>
</tr>
<tr>
<td>(6) RICNWT</td>
<td>RICNWT</td>
</tr>
<tr>
<td>(7) ASYMREG</td>
<td>ASMREG</td>
</tr>
<tr>
<td>(8) ASYMFIL</td>
<td>ASMFIL</td>
</tr>
<tr>
<td>(9) EXPMDFL</td>
<td>EXPMDF</td>
</tr>
<tr>
<td>(10) 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:

<table>
<thead>
<tr>
<th>Old Name</th>
<th>New Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) PNCH</td>
<td></td>
</tr>
<tr>
<td>(2) DIMAG</td>
<td></td>
</tr>
<tr>
<td>(3) DREAL</td>
<td></td>
</tr>
<tr>
<td>(4) BLOCK DATA</td>
<td></td>
</tr>
</tbody>
</table>
(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 ROTIL
IMPLICIT REAL*8 (A-M, O-Z)
COMMON/LINES/TITLE(10), TIL(3), NLP, LIN
COMMON/FORM/FMT1(2), FMT2(2), NEPR
COMMON/TOL/EPSAM, EPSAM, IACM
COMMON/CONV/SUMCV, RICTCV, SERCV, MAXSUM
NLP & NO. LINES/PAGE VARES WITH THE INSTALLATION
READ(5, 100, END=90, ERR=91) TITLE

100 FORMAT(10A8)
CALL LNCNT(100)
RETURN
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(1H1,10A8,3A8/)
LIN=2+N
IF (N.GT.NLP) LIN=2
20 RETURN
END
SURROUNTE READ(I,A,NA,B,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,

CALL READ1(A, NA, NZ, LAB)

IF(I .EQ. 1) GO TO 999

READ(5,100) LAB,

CALL READ1(B, NB, NZ, LAB)

IF(I .EQ. 2) GO TO 999

READ(5,100) LAB,

CALL READ1(C, NC, NZ, LAB)

IF(I .EQ. 3) GO TO 999

READ(5,100) LAB,

CALL READ1(D, ND, NZ, LAB)

IF(I .EQ. 4) GO TO 999

READ(5,100) LAB,

CALL READ1(E, NE, NZ, LAB)

100 FORMAT (A8,4X,2I4)

999 RETURN

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 /1,0,1,1,1,1/
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 (11,10,132,12,10)
10 CALL LNCNT(100)
11 CALL LNCNT(2)
3. WRITE(6,177) KZ,NR,NAM
177 FORMATT(A,5X,A4,8H MATRIX,5X,I3,5M ROWS,5X,I3,8M COLUMNS)
GO TO 13
12 CALL LNCNT(100)
GO TO 13
13 CALL LNCNT(2)
14 WRITE(6,891)
891 FORMAT(A80)
GO TO 13
15 J = (NC-1)/NEPR+1
C = BELOW COMPUTE NR OF LINES/ ROW --DECIDE IF 1 EXTRA BLANK LINE
16 JST = 1
1. C COMPUTE LAST ROW POSITION -1 BELOW
2 NLST = NLST +NR
3 MN=NC
4 IF (NC,GT,NEPR) MN=NEPR
5 KLAST=NR*(MN-1)
6 CONTINUE
7 DO 912 J = JST, NR
8 CALL LNCNT(NLPW)
9 KLAST = KLAST +1
10 WRITE (6,FMT1) A(N), NR,KLAST,NR
11 IF (NC.LE.NEPR) GO TO 912
12 NLST = NLST +1
13 KNR=KLST+NR
14 WRITE (6,FMT2) A(NR), NR,KNR,NLST,NR
15 CONTINUE
16 RETURN
17 CALL LNCNT(1)
18 WRITE (6,916) NAME
916 FORMAT (* ERROR IN PRNT MATRIX "A", HAS NAME",216)
20 RETURN
21 END

ORIGINAL PAGE IS OF POOR QUALITY
SUBROUTINE EQUATE(A, NA, A, NA)

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

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

L = NA(1) * NA(2)

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

DO 300 I = 1, L

300 B(I) = A(I)

1000 RETURN

999 CALL LNCNT(1)

WRITE (6,50) NA

50 FORMAT ("DIMENSION ERROR IN EQUATE NA=", 2I6)

RETURN

END
SUBROUTINE TRANP(A, NA, R, NB)
IMPLICIT REAL*4(A-N, O-Z)
DIMENSION A(1), B(1), NA(2), NB(2)
NR(1) = NA(2)
NR(2) = NA(1)
NR = NA(1)
NC = NA(2)
LNR = NC
IF (NR .LT. 1 .OR. L .LT. 1) GO TO 999
IRE = 0
DO 300 I = 1, NR
II = I - NR
DO 300 J = 1, NC
II = I + NR
IRE = IRE + 1
300 B(I) = A(IJ)
RETURN
999 CALL LNCNT(1)
WRITE (6, 50) NA
50 FORMAT ('DIMENSION ERROR IN TRANP: NA=', 216)
RETURN
END
SUBROUTINE SCALE (A, NA, R, NB, S)
IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION A(1),R(1),NA(2),NB(2)
NA(1) = NA(1)
NA(2) = NA(2)
L = NA(1)*NA(2)
IF( NA(1), 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
300 I = 1,L
1000 RETURN
50 FORMAT (' DIMENSION ERROR IN SCALE NA=', 2I6)
RETURN
END
SUBROUTINE UNITY(A,NA)
IMPLICIT REAL*8 (A-M, 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(6,50)(NA(I), I=1,2)
50 FORMAT ("DIMENSION ERROR IN UNITY NA=", 2I6)

1000 RETURN

END
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(2 DIMENSION ERROR IN NULL NA = 2I6)
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
IF (N.LT. 1) GO TO 600
DO 10 I=1, N
M=I+NA(I-1)
10 TR=TR+A(M)
RETURN
600 CALL LNCNT(1)
WRITE (60, 1600) NA
1600 FORMAT (* TRACE REQUIRES SQUARE MATRIX NA=*216)
RETURN
END
SUBROUTINE ADD (A, NA, A, 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
IF (NA(1) .NE. NA(2)) GO TO 999
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
END
SUBROUTINE SUAT(A, NA, R, NA, 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

300 C(I) = A(I) + B(I)

GO TO 1000

999 CALL LCNT (1)

WRITE (6, 50) NA, NB

50 FORMAT (* DIMENSION ERROR IN SUBT 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
NA = NA(1)
NAC = NA(2)
NRC = NB(2)
NAB = NAR = NAC
IF (NAR .LT. 1 .OR. NAA .LT. 1 .OR. NBB .LT. 1) GO TO 999
IR = 0
IK = NAC
DO 350 K = 1, NAC
IK = IK + NAC
DO 350 J = 1, NAR
IR = IR + 1
IB = IK
JI = JAR
V1 = 0.0
DO 300 I = 1, NAC
JI = JI + NAR
IB = IB + 1
V3 = A(JI)
V4 = B(I)
V2 = V3 + V4
V1 = V1 + V2
300 CONTINUE
310 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=",216,9X,"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*8(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

IF(IOPT=2)5,20,30

C THIS LOOP COMPUTES THE ONE-NORM

DO 15 K=1,N
I=I+MAXROW
10 DO 15 J=1,M
L=I+J
5 SUM=ABS(A(L))+SUM
15 IF(SUM_GT_RLNORM)RLNORM=SUM
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=A(L)
30 RLNORM=SUM+SUM+RLNORM
35 RETURN

C THIS LOOP COMPUTES THE INFINITY-NORM

DO 40 J=1,M
L=L+MAXROW
40 SUM=ABS(A(L))+SUM
45 IF(SUM_GT_RLNORM)RLNORM=SUM
50 RETURN

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)
NC=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)

16 20 C(J)=A(I)
RETURN

600 CALL LNCNT(1)
WRITE (6,1600) NA, NB
1600 FORMAT (' DIMENSION ERROR IN JUXTC, NA=',2I6,'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

MCA = NA(2)
MRB = NA(1)
MRC = NC(1)

DO 10 I = 1, MCA
10 L = J + MRC * (I = 1)

DO 20 I = 1, MCA
20 C(L) = A(K)

DO 20 J = 1, MRB
20 K = J + MRB * (I = 1)

L = MRB * J + MRC * (I = 1)

C(L) = B(K)

RETURN

CALL LNCNT(1)

WRITE(6, 1600) NA, NB

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

RETURN

END
SUBROUTINE FACTOR(Q,NQ,D,NO,IOP,IA,C,DUMMY)

IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION Q(1),D(1),DUMMY(1)
DIMENSION NQ(2),NO(2),NDUM(2)

IOPT = 2
N = NQ(1)
M = NO(2)
N1 = M + 1
N2 = N1 + N

CALL EQUATE(Q,NQ,DUMMY,NQ)
CALL SNVDEC(IOPT,N,N,N,N,DUMMY,NOS,IA,C,ZTEST,DUMMY(N1),D,IRANK)

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

IF( IERR .GT. 0 ) PRINT 100,IERR
100 FORMAT(//, 'IN FACTOR, SNVDEC MAY FAILED TO CONVERGE TO THE ',I4)

IF(IERR .EQ. -1) PRINT 150,ZTEST,IRANK
150 FORMAT(//, 'SINGULAR VALUE AFTER 30 ITERATIONS')

10 2. IS REDUCED . THE RANK MAY ALSO BE REDUCED',//, 'CURRENT RANK =',I4)

NDUM(1) = N
NDUM(2) = 1

IF(IERR .EQ. -1) CALL PRNT(DUMMY(N1),NDUM,4HNSVL,1)
IF(IERR .GT. 0 ) RETURN

200 CONTINUE
NDUM(1) = N

DO 250 J = 1,N

M1 = (J-1)*N + 1
N2 = J*N

DO 250 I = M1,M2

K = N2+I-1
L = N1+J-1

IF( DUMMY(L) .EQ. 0.0 ) GO TO 300
DUMMY(K) = DSQRT(DUMMY(L)*DUMMY(1))

250 CONTINUE

NDUM(2) = N
GO TO 350

300 NOUM(2) = J - 1

350 CONTINUE

IF( DUMMY(N2) .LT. 0.0 ) CALL SCALE(DUMMY(N2),NDUM,DUMMY(N2),NDUM)

1 = 1.0

CALL TRANP(DUMMY(N2),NDUM,D,ND)

IF( IOP .EQ. 0 ) RETURN
CALL LNCNT(4)

CALL SCALE(NQ,NQ,4H1,0)
CALL PRT(ND,ND,4H0,1)
CALL SCALE(DUMMY(N2),ND,N2,DUMMY,NQ)
CALL PRT(DUMMY,NQ,4H0,0)

RETURN
END

A-18
SUBROUTINE EIGEN(MAX, N, A, ER, EI, TSV, ILV, V, WK, IERP)  
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 MOR (MAX,N,LOW,IGH,A,ER,EI,IERR)  
IF (IERR .NE. 0) GO TO 260  
DO 5 I=1,N  
  WK(I,1) = ER(I)  
  WK(I,2) = EI(I)  
  WK(I,3) = ER(I)**2 + EI(I)**2  
5 CONTINUE  
IC = 0  
GO TO 190  

CONTINUE  
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 MOR (N,N,LOW,IGH,WK(1,6),ER,EI,IERR)  
IF (IERR .NE. 0) GO TO 260  
DO 30 I=1,N  
  WK(I,3) = ER(I)**2 + EI(I)**2  
30 CONTINUE  

FIND LARGEST ILV EIGENVALUES AND FLAG THEM  
DO 50 I=1,ILV  
  P = -1.0D0  
  DO 40 J=1,N  
    IF (WK(J,3) .LE. P) GO TO 40  
    K = J  
    P = WK(K,3)  
40 CONTINUE  
  SELECT(K) = .TRUE.  
  WK(K,3) = -WK(K,3)  
50 CONTINUE  
  IF (EI(K) .EQ. 0.) GO TO 60  
  IF (EI(K) .GT. 0.) GO TO 55  
  IF (SELECT(K-1)) GO TO 60  
  ILV = ILV+1  
SELECT(K-1) = .TRUE.  

EXTERNAL RALANC, ELMHES, MOR, NOR, RITNIT  
RITNIT, CALL RHFAC}
GO TO 60
CONTINUE
IF (.NOT.SELECT(K+1)) ILV = ILV+1
CONTINUE
IF (ISV .EQ. 0) GO TO 90
CONTINUE
FIND SMALLEST ISV EIGENVALUES AND FLAG THEM
DO 65 IM=1,N
WK(I,3) = DABS(WK(I,3))
CONTINUE
DO 70 IM=1,ISV
P = 1.074
GO TO 70
IF (WK(J,3) .GE. P) GO TO 70
K = J
P = WK(J,3)
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 99
ISV = ISV+1
SELECT(K+1) = .TRUE.
GO TO 90
CONTINUE
IF (.NOT.SELECT(K+1)) ISV = ISV+1
CONTINUE
FIND EIGENVECTORS FOR FLAGGED EIGENVALUES
CALL INV(\(M\),N,A,ER,EI,SELECT,N,M,V,IERR,WK(1,6),WK(1,3),
1
WK(1,5))
CALL ELMBAK(\(M\),\(L\),\(H\),A,INT(1),M,V)
CALL BALBAX(\(M\),\(L\),\(H\),WK,M,V)
SEPARATE FLAGGED EIGENVALUES FROM UNFLAGGED EIGENVALUES
IV = ISV + ILV
IF (IV .LE. N) GO TO 100
ILV = N-IV
IV = 1
IC = 0
JC = IV
DO 130 IM=1,N
IF (SELECT(I)) GO TO 120
IF (EI(I) .GE. 0.) GO TO 110
IF (SFLECT(I+1)) GO TO 120
CONTINUE
JC = JC+1
WK(JC,1) = ER(I)
WK(JC,2) = EI(I)
KC = JC
GO TO 130
CONTINUE
IC = IC+1
WK(IC,1) = ER(I)
• 126 $\text{NK}(IC,2) = EI(I)$
• 127 $\text{KC} = IC$
• 128 130 CONTINUE
• 129 $\text{NK}(KC,3) = ER(I)**2 + EI(I)**2$
• 130 150 CONTINUE
• 131 C NORMALIZE VECTORS TO UNIT LENGTH AND STORE FOR REORDERING
• 132 C
• 133 C
• 134 J = 0
• 135 151 CONTINUE
• 136 J = J+1
• 137 IF ($\text{NK}(J,2) \cdot \text{NE.} \cdot 0.$) GO TO 154
• 138 SUM = 0.
• 139 DO 152 I=1,N
  140 SUM = SUM + $V(I,J)**2$
• 141 152 CONTINUE
• 142 IF (SUM \cdot \text{EQ.} \cdot 0.) GO TO 158
• 143 SUM = DSORT(SUM)
• 144 DO 153 I=1,N
  145 $W(K,I,J+4) = V(I,J) / \text{SUM}$
• 146 153 CONTINUE
• 147 GO TO 158
• 148 154 CONTINUE
• 149 J1 = J+1
• 150 SUM = 0.
• 151 DO 155 I=1,N
  152 SUM = SUM + $V(I,J)**2 + V(I,J1)**2$
• 153 155 CONTINUE
• 154 IF (SUM \cdot \text{EQ.} \cdot 0.) GO TO 157
• 155 SUM = DSORT(SUM)
• 156 DO 159 I=1,N
  157 $W(K,I,J+4) = V(I,J) / \text{SUM}$
  158 $W(K,I,J+5) = V(I,J1) / \text{SUM}$
• 159 156 CONTINUE
• 160 157 CONTINUE
• 161 J = J1
• 162 158 CONTINUE
• 163 IF (J \cdot \text{LT.} \cdot IV) GO TO 151
• 164 IC = 0
• 165 LC = 0
• 166 IF (ISV \cdot \text{EQ.} \cdot 0.) GO TO 190
• 167 C ORDER SMALLEST ISV EIGENVALUES AND EIGENVECTORS FOR OUTPUT
• 168 C
• 169 C
• 170 DO 190 I=1,ISV
  171 P = 1.074
  172 DO 160 J=1,IV
    173 IF ($W(K,J,3) \cdot \text{GE.} \cdot P$) GO TO 160
    174 K = J
    175 P = $W(K,3)$
  176 160 CONTINUE
  177 IC = IC+1
  178 LC = LC+1
  179 ER(IC) = $W(K,1)$
  180 EE(IC) = $W(K,2)$
  181 DO 170 J=1,N
    182 $V(J,LC) = W(J,K+4)$
  183 170 CONTINUE
  184 $W(K,3) = 1.074$
• 185 180 CONTINUE
• 186 190 CONTINUE
• 187 IF (IV \cdot \text{EQ.} \cdot N) GO TO 220
• 188 C
C OPROF UNFLAGGED EIGENVALUES FOR OUTPUT

C

IV1 = IV + 1
IUF = N - IV
DO 210 I = 1, IUF
     P = 1.074
     DO 200 J = IV1, N
          IF (WK(J,3) .GE. P) GO TO 200
          K = J
          P = WK(J,3)
     200 CONTINUE
     IC = IC + 1
     ER(IC) = WK(K,1)
     EII(IC) = WK(K,2)
     WK(K,3) = 1.074
     IC = IC + 1
210 CONTINUE
IF (ILV .EQ. 0) GO TO 260
10 DO 250 I = 1, ILV
212 DO 230 J = 1, IV
         IF (WK(J,3) .GE. P) GO TO 230
         K = J
         P = WK(J,3)
    230 CONTINUE
     IC = IC + 1
     LC = LC + 1
     ER(IC) = WK(K,1)
     EII(IC) = WK(K,2)
     DO 240 J = 1, N
         V(J,LC) = WK(J,K+4)
     240 CONTINUE
     WK(K,3) = 1.074
250 CONTINUE
260 CONTINUE
RETURN
END
SUBROUTINE SYMPOS (MAXN,N,A,NRHS,B,IOPT,IFAC,DETERM,SCALE,P,IERR)

IMPLICIT REAL*8 (A-M,0-Z)

DIMENSION A(MAXN,N),B(MAXN,NRHS),P(N)

DATA P1,P2/1.00+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
TF( A(1,1) .LE. 0.0 ) IERR = 1

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

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

TEST TO DETERMINE IF CHOLESKY DECOMPOSITION OF COEFFICIENT MATRIX IS DESIRED

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

DETERM=1.0
ISCALE=0

'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
A(I,1) = I-1

35 IM1 = I-1

36 DO 150 J=1,I
X = A(J,I)

39 C DETERMINE IF ELEMENTS ARE ABOVE OR BELOW THE DIAGONAL

40 IF (I.GT.J) GO TO 110

42 C USING THE DIAGONAL ELEMENTS OF MATRIX A, THIS SECTION COMPUTES DIAGONAL MATRIX AND DETERMINES IF MATRIX A IS SYMMETRIC POSITIVE DEFINITE

46 IF (IM1.GT.0) GO TO 50

47 DO 40 K=1,IM1
Y = A(I,K)

49 A(I,K) = Y*P(K)

50 X = X - Y*A(I,K)

51 40 CONTINUE

52 C TEST IF COEFFICIENT MATRIX IS POSITIVE DEFINITE

54 50 IF( X.LE.0.0 ) IERR = 1

55 C COMPUTE INVERSE OF DIAGONAL MATRIX D**-1 = 1/P

56 P(I) = 1.0/X

58 C TEST TO SEE IF DETERMINANT IS TO BE EVALUATED

60 IF (IOPT .EQ. 0) GO TO 150

61 C

62 C
6 C SCALE THE DETERMINANT (COMPUTE THE DETERMINANT EVALUATION SYN0064

6 C PARAMETERS DETERM AND ISCALE SYN0065

65 C PIVOTI = x SYN0066
66 60 IF(DABS(DETERM).LT.R1) GO TO 70 SYN0067
67 DETERM = DETERM*R2 SYN0068
68 ISCALE = ISCALE+1 SYN0069
69 GO TO 60 SYN0070
70 70 IF(DABS(DETERM).GT.R2) GO TO 80 SYN0071
71 DETERM = DETERM/R1 SYN0072
72 ISCALE = ISCALE-1 SYN0073
73 GO TO 70 SYN0074
74 80 IF(DABS(PIVOTI).LT.R1) GO TO 90 SYN0075
75 PIVOTI = PIVOTI*R2 SYN0076
76 ISCALE = ISCALE+1 SYN0077
77 GO TO 80 SYN0078
78 90 IF(DABS(PIVOTI).GT.R2) GO TO 100 SYN0079
79 PIVOTI = PIVOTI*1 SYN0080
80 ISCALE = ISCALE-1 SYN0081
81 GO TO 90 SYN0082
82 100 DETERM = DETERM*PIVOTI SYN0083
83 GO TO 150 SYN0084
84 C SYN0085
85 C USING THE LOWER TRIANGULAR ELEMENTS OF MATRIX A, THIS SYN0086
86 SECTION COMPUTES THE UNIT LOWER TRIANGULAR MATRIX SYN0087
87 SYN0088
88 110 JM1 = J-1 SYN0089
89 IF (JM1 .EQ. 0) GO TO 140 SYN0090
90 DO 120 K=1,JM1 SYN0091
91 X = X = A(I,K)*A(J,K) SYN0092
92 120 CONTINUE SYN0093
93 C SYN0094
94 140 A(I,J) = X SYN0095
95 C SYN0096
96 150 CONTINUE SYN0097
97 C SYN0098
98 SECTION TO APPLY BACK SUBSTITUTION TO SOLVE L*Y = B FOR SYN0099
99 UNIT LOWER TRIANGULAR MATRIX AND CONSTANT COLUMN VECTOR R SYN100
100 C SYN101
101 160 IF( IFAC .EQ. 2 ) RETURN SYN102
102 DO 190 J=2,N SYN103
103 JM1 = I-1 SYN104
104 C SYN105
105 DO 190 J=1, NRHS SYN106
106 X = B(I,J) SYN107
107 C SYN108
108 DO 170 K=1,IM1 SYN109
109 X = X = A(I,K)*B(K,J) SYN110
110 170 CONTINUE SYN111
111 C SYN112
112 R(I,J) = X SYN113
113 180 CONTINUE SYN114
114 C SYN115
115 SECTION TO SOLVE (LTRANSPOSE)*X = (D**-1)*Y FOR TRANSPOSE SYN116
116 OF UNIT LOWER TRIANGULAR MATRIX AND INVERSE OF DIAGONAL SYN117
117 C SYN118
118 Y = P(N) SYN119
119 DO 190 J=1, NRHS SYN120
120 R(N,J) = B(N,J)*Y SYN121
121 190 CONTINUE SYN122
122 C SYN123
123 200 I = NM1+1 SYN124
124 Y = P(NM1) SYN125

A-24
26 C
27 DO 220 J=1,NRMS
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
39 C ALL COLUMN VECTORS
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,WK,IERR)
IMPLICIT REAL*4 (A-H,O-Z)
DIMENSION A(NMAX,1),B(NMAX,1),IPIVOT(1),WK(1)
IERR=0

TEST FOR L/U FACTORIZATION
IF(IFAC.EQ.1)GO TO 10
CALL DETFAC(NMAX,N,A,IPIVOT,IFAC,DETERM,ISCALE,WK,IERR)
IF(IERR.GT.0)RETURN
10 IF (IFAC.EQ.2) DETERM*=(10.**((100*ISCALE))
IMPLICIT REAL*4 (A•4,0•7)
DIMENSION A(NMAX,i),9(NMAX,1),IPIVOT(1),WK(1)

TEST FOR SCALAR A MATRIX
IF(NM1.GT.0)GO TO 40
IF(A(1,1).EQ.0.)GO TO 30
DO 20 I=1,NRHS
20 B(I,1)=B(I,1)/A(1,1)
IF (IFAC.EQ.2) WK(1)=DETERM
RETURN
30 IERR=1
RETURN
40 DO 100 M=1,NRHS
100 CONTINUE

PIVOT THE M-TH COLUMN OF A MATRIX
DO 50 I=1,NM1
KI=IPIVOT(I)
P=A(KI,M)
R(KI,M)=R(I,M)
50 A(I,M)=P

FORWARD SUBSTITUTION
WK(1)=R(1,M)
DO 70 I=2,N
IM1=I-1
P=0.0
DO 60 K=1,IM1
60 P=P+A(I,K)*WK(K)
70 WK(I)=B(I,M)-P

BACK SUBSTITUTION
RN=WK(N)/A(N,N)
DO 90 J=1,NM1
I=N-J
IP1=I+1
P=WK(I)
DO 80 K=IP1,N
80 P=P+A(I,K)*R(K,M)
90 R(I,M)=P/R(I,I)

CONTINUE
IF (IFAC.EQ.2) WK(1)=DETERM
RETURN
END
SUBROUTINE SNVD(ND, N, A, NOS, R, IAC, ZTEST, Q, V, IRANK, APLUS, SNVO00
1 IERR)
2 IMPLICIT REAL*8 (A-H, O-Z)
3 LOGICAL MITHU, MITHV
4 DIMENSION A(ND, N), V(ND, N), R(N), E(IAC)
5 DIMENSION B(ND, NOS), APLUS(ND, N)
6 C
7 C TEST FOR SCALAR OR VECTOR A
8 C
9 C IF ( N .GE. 2 ) GO TO 3000
10 C
11 IF ( IERR .EQ. 0 ) GO TO 1200
12 C
13 IF ( ZTEST .EQ. 10. ** (-IAC) ) GO TO 1100
14 C
15 DO 1000 I = 1, M
16 SUM = SUM + A(I, 1) * A(I, 1)
17 1000 CONTINUE
18 SUM = DSQRT(SUM)
19 C
20 IF ( SUM .GT. ZTEST ) IRANK .EQ. 1
21 C
22 IF ( IERR .EQ. 1 ) RETURN
23 C
24 IF ( IOP .EQ. 1 ) RETURN
25 C
26 IF ( IRANK .EQ. 0 ) GO TO 1200
27 DO 1100 I = 1, M
28 A(I, 1) = A(I, 1) / SUM
29 1100 CONTINUE
30 GO TO 1300
31 C
32 IF ( IOP .EQ. 2 ) RETURN
33 C
34 IF ( IOP .EQ. 4 ) GO TO 1500
35 C
36 IF ( IRANK .EQ. 0 ) GO TO 1600
37 DO 1500 J = 1, NOS
38 Z = 0
39 DO 1400 I = 1, M
40 Z = Z + A(I, 1) * A(I, J) / SUM
41 1400 CONTINUE
42 B(1, J) = Z
43 1500 CONTINUE
44 GO TO 1800
45 C
46 IF ( IOP .EQ. 3 ) RETURN
47 C
48 IF ( IRANK .EQ. 0 ) GO TO 2000
49 C
50 APLUS(I, 1) = A(I, 1) / SUM
51 1900 CONTINUE
52 RETURN
53 C
54 IF ( IOP .EQ. 0 ) GO TO 2000
55 DO 2100 I = 1, M
56 APLUS(I, 1) = 0.0
57 2000 CONTINUE
58 RETURN
59 C
60 2100 CONTINUE
61 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 S-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.+**(-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 HOUSEHOLDER REDUCTION TO BIDIAGONAL FORM.
93 C
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 + 4(AJ,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 Q(I) = G
120 IF(I .EQ. N) GO TO 20
121 C ANNIHILATE THE I-TH ROW TO RIGHT OF SUPER-DIAG.
122 C
123 C
124 S = 0,0
125 DO 11 J = L,N
A-28
26 11 $ S = S + A(I,J) \times A(I,J)$
27 $G = 0.0$
28 IF ($S \leq TOL$) GO TO 20
29 $G = 0.0$
30 $F = A(I,I+1)$
31 IF ($F \geq 0.0$) $G = -G$
32 $H = F \times G$
33 $A(I,I+1) = F + G$
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 $S = S + A(J,K) \times A(I,K)$
40 DO 17 $K = L,N$
41 $A(J,K) = A(J,K) + S \times E(K)$
42 CONTINUE
43 $Y = DARS(G(I)) + DARS(E(I))$
44 IF ($Y \geq 0.0$) SIZE = $Y$
45 CONTINUE
46 IF (NOT .WITHV) GO TO 41
47 ACCUMULATION OF RIGHT TRANSFORMATIONS.
48 DO 40 $II = 1,N$
49 $I = NP1 - II$
50 IF ($I \leq 0$) GO TO 39
51 IF ($G \geq 0.0$) GO TO 37
52 $H = A(I,I+1) \times G$
53 DO 32 $J = L,N$
54 $V(J,I) = A(I,J)/H$
55 DO 36 $J = L,N$
56 $S = 0.0$
57 DO 33 $K = L,N$
58 $S = S + A(I,K) \times V(K,J)$
59 DO 34 $K = L,N$
60 $V(I,J) = V(K,J) + S \times V(K,I)$
61 CONTINUE
62 $V(I,J) = 0.0$
63 $V(I,I) = 0.0$
64 $G = E(I)$
65 $L = I$
66 CONTINUE
67 IF (.NOT. .WITHV) GO TO 53
68 ACCUMULATION OF LEFT TRANSFORMATIONS.
69 DO 52 $II = 1,N$
70 $I = NP1 - II$
71 $L = I + 1$
72 $G = G(I)$
73 IF ($I \leq 0$) GO TO 43
74 $V(I,J) = 0.0$
75 DO 42 $J = L,N$
76 $A(I,J) = 0.0$
77 CONTINUE
78 IF ($G \geq 0.0$) GO TO 49
79 IF ($I \leq 0$) GO TO 47
80 $H = A(I,I) \times G$
81 DO 46 $J = L,N$
82 $S = 0.0$
83 DO 44 $K = L,M$
189 44 \( S = S + A(K,I)A(K,J) \)
190 45 \( F = S \)
191 46 \( \text{DO} \ 45 \ K = 1,M \)
192 47 \( A(K,J) = A(K,J) + F* A(K,I) \)
193 48 \( \text{CONTINUE} \)
194 49 \( \text{GO TO} \ 51 \)
195 50 \( \text{DO} \ 48 \ J = 1,M \)
196 51 \( A(J,I) = A(J,I)/G \)
197 52 \( \text{GO TO} \ 51 \)
198 53 \( \text{CONTINUE} \)
199 54 \( \text{CONTINUE} \)

202 C **DIAGONALIZATION OF BIDIAGONAL FORM.**
203 C **TEST F SPLITTING.**
204 C **CANCELLATION OF E(L) IF L .GT. 1.**
205 C **C**
206 C **C**
207 C **C**
208 C **C**
209 C **C**
210 C **C**
211 C **C**
212 C **C**
213 C **C**
214 C **C**
215 C **C**
216 C **C**
217 C **C**
218 C **C**
219 C **C**
220 C **C**
221 C **C**
222 C **C**
223 C **C**
224 C **C**
225 C **C**
226 C **C**
227 C **C**
228 C **C**
229 C **C**
230 C **C**
231 C **C**
232 C **C**
233 C **C**
234 C **C**
235 C **C**
236 C **C**
237 C **C**
238 C **C**
239 C **C**
240 C **C**
241 C **C**
242 C **C**
243 C **C**
244 C **C**
245 C **C**
246 C **C**
247 C **C**
248 C **C**
249 C **C**
250 C **C**
251 C **C**

```plaintext
CONTINUE
```

A-30

```plaintext
GO TO 51
```
SHIFT FROM LOWER 2X2.

X=G(L)
Y=G(K-1)
G=E(K-1)
H=E(K)
F=(Y-Z)*(Y+Z)+(G-H)*(G+H)/(2.0*H*Y)
G=D*SQRT(F+F+H+1.0)
IF(FLT.0.0) G=G
F=(((X-Z)*(X+Z)*H*(Y/(F+G)-H))/X

NEXT OR TRANSFORMATION.

C=1.0
S=1.0
LP1=L+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)=Z
C=F/Z
S=M/Z
F=S*C+G+S
G=X*S+G*C
H=Y+S
Y=Y+C
IF(.NOT.WITHV) GO TO 70
DO 68 J=1,N
X=V(J,I-1)
Z=V(J,I)
V(J,I-1)=X*C+Z*S
V(J,I)=X*S+Z*C
CONTINUE

DO 70 I=1,M
Y=A(J,I-1)
Z=A(J,I)
A(J,I-1)=Y*C+Z*S
A(J,I)=Y*S+Z*C
CONTINUE

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

CONVERGENCE.
31 Q(K)=Z
31 IF(.NOT,.ofile) GO TO 100
31 DO 76 J=1,N
31 76 V(J,K)=V(J,K)
31 100 CONTINUE
32 C
32 IERR=0
32 DO 240 I=2,N
32 240 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
32 C
32 IF (K.EQ.I) GO TO 240
32 Q(K)=Q(I)
32 Q(I)=P
32 C
32 IF(IOA.EQ.1) GO TO 240
32 DO 260 J=1,N
32 P=V(J,I)
32 V(J,I)=V(J,K)
32 V(J,K)=P
32 260 CONTINUE
32 C
32 DO 270 J=1,M
32 P=A(J,I)
32 A(J,I)=A(J,K)
32 A(J,K)=P
32 270 CONTINUE
32 C
32 280 CONTINUE
32 C
32 290 IF (Q(J),GT,TEST) GO TO 300
32 Q(J)=0.0
32 J=J+1
32 GO TO 290
32 300 IRANK =J
32 TEMP = TEST/Q(J)
32 IF (TEMP.GT.0.0625) IERR=1
32 C
32 IF (IOA.LT.3) RETURN
32 IF(IOA,.LT.3) GO TO 170
32 DO 160 L=1,NOS
32 160 J=1,IRANK
32 SUM=0.0
32 DO 120 I=1,M
32 SUM=SUM+A(I,J)*B(I,L)
32 120 CONTINUE
32 130 F(J)=SUM/Q(J)
32 C
32 DO 150 K=1,N
32 SUM=0.0
32 DO 140 I=1,IRANK
32 SUM=SUM+V(K,I)*E(I)
32 140 SUM=SUM
32 150 CONTINUE
32 160 CONTINUE
32 170 DO 200 J=1,M
32 200 CONTINUE

A-32
DO 180 K=1,IRANK
180 SUM =SUM + V(I,K)*A(J,K)/Q(K)
190 APLUS(I,J) = SUM
200 CONTINUE
34 C
35 IF( IOP .EQ. 4 ) RETURN
36 DO 230 K=1,NOS
37 DO 220 I=1,N
38 SUM = SUM + V(I,K)*A(J,K)
39 DO 210 J=1,N
40 SUM = SUM + APLUS(I,J)*B(J,K)
41 DO 220 I=1,N
42 B(I,K) = E(I)
43 RETURN
44 END
SUBROUTINE SUM(A, NA, B, NB, C, NC, IOP, SYM, DUMMY)
IMPLICIT REAL A (A-H, O-Z)
DIMENSION A(1), B(1), C(1), DUMMY(1)
DIMENSION NA(2), NB(2), NC(2)
LOGICAL SYM
COMMON/CONV/SUMCV, RICTCV, SERCV, MAXSUM
IF (IOP .EQ. 0) GO TO 100
PRINT 50
50 FORMAT(/, 'LINEAR EQUATION SOLVER X = AXC + B')
CALL PRNT(A, NA, 4M A, 1)
IF (SYM) GO TO 75
CALL PRNT(C, NC, 4M C, 1)
GO TO 85
75 CONTINUE
PRINT 80
80 FORMAT(/, 'C = A TRANSPOSE')
85 CONTINUE
CALL PRNT(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, WNSX)
IF (WNS .GE. 1.) GO TO 225
IF (WNSX .LT. SUMCV) GO TO 300
GO TO 235
225 IF (WNSX .LT. SUMCV) GO TO 300
300 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 TRANP(A, NA, 4M C, NC)
250 CONTINUE
I = I + 1
IF (I .LE. MAXSUM) GO TO 200
CALL LNCNT(3)
PRINT 275, MAXSUM
275 FORMAT(/, 'IN SUM, THE SEQUENCE OF PARTIAL SUMS HAS EXCEEDED STAGS')
1 E, 'IS, ' WITHOUT CONVERGENCE')
300 CONTINUE
IF (IOP .EQ. 0) RETURN
CALL PRNT(B, NB, 4M X, 1)
RETURN
END
SUBROUTINE BILIN(A, NA, B, NB, C, NC, 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), NDUM(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 AX + XB = C ')
CALL PRNT(A, NA, 4M A, 1)
CALL PRNT(R, NB, 4M R, 1)
GO TO 200
100 CONTINUE
CALL LNCNT(3)
PRINT 150
150 FORMAT(' LINEAR EQUATION SOLVER ( B TRANSPOSE )X + XB = C ')
CALL TRANP(A, NA, DUMMY, NDUM)
CALL PRNT(DUMMY, NDUM, 4M B, 1)
200 CONTINUE
CALL PRNT(C, NC, 4M C, 1)
300 CONTINUE
C
C
IOPTT = 0
N = NA(1)**2
M = NA(1)**2
C
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(N3), IERR)
IF (IERR .EQ. 0) GO TO 350
CALL LNCNT(3)
PRINT 325, IERR
325 FORMAT(' IN BILIN, THE ', I4, ' EIGENVALUE OF A HAS NOT BEEN DETERMINED AFTER 30 ITERATIONS')
IERR=1
CALL NORMS(NEVL, NEVL, NEVL, A, IERR, BETA)
BETA = 2, * BETA
GO TO 355
350 CONTINUE
J = N1 + NEVL - 1
K = N2 + NEVL - 1
CD = 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) * CD) / CD
IF (BETA .LE. 0.0) GO TO 365
BETA = DSQRT(BETA)
GO TO 345
365 CONTINUE
C
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(B,NB,DUMMY,NB)
**N1** = M + 1
**N2** = N1 + NB(1)
**N3** = N2 + NB(1)
**NEVL** = NB(1)
**CALL** E IG(EIG(NB(1),NB(1),DUMMY,DUMMY(N1),DUMMY(N2),ISV,ILV,V,DUMMY(NB),(I),IERR)
**IF** (IERR .EQ. 0) **GO TO** 450
**CALL** LNCNT(3)
**PRINT** 400,IERR

**400 FORMAT(/"IN BILIN, THE */"I4,"/"EIGENVALUE OF B WAS NOT FOUND AFTER 30 ITERATIONS")
IERR = 1
**CALL** NORMS(NEVL,NEVL,NEVL,B,IERR,BETA1)
BETA1 = 2*BETA1
**GO TO** 485

**450 CONTINUE**
J = N1 + NEVL - 1
K = N2 + NEVL - 1
**C0** = DSQRT(DUMMY(N1)**2 + DUMMY(N2)**2)
**C1** = 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)*CD)/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**
BETA = (BETA + BETA1)/2.

**500 CONTINUE**

**IF** (IOP(1) .EQ. 0 ) **GO TO** 520
**CALL** LNCNT(4)
**PRINT** 515,HETA

**515 FORMAT(/"BETA = *E16.8/")
**520 CONTINUE**

**525 CONTINUE**
N1 = N+1
CALL EQUATE(A,NA,DUMMY,NA)
CALL EQUATE(A,NA,DUMMY(N1),NA)
CALL SCALE(DUMMY,NA,DUMMY,NA,-1.)
L = -NA(1)
DO 525 I=1,NAX
   M1 = L + N
   DUMMY(L) = BETA = A(L)
   DUMMY(M1) = BETA - A(L)
   525 CONTINUE
N2 = N1 + N
NDUM(2) = NDUM(2) + NA(1)
N3 = N2 + NC(1)*NC(2)
GAM = -2.*BETA
C 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(N5),IERR)
IF(IERR.EQ.1) PRINT 625
CALL EQUATE(DUMMY(N1),NA,DUMMY,NA)
CALL EQUATE(DUMMY(N2),NC,C,NC)
CALL TRAMP(DUMMY,NA,DUMMY(N1),NA)
CALL TRAMP(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 SUM(DUMMY,NA,C,NC,DUMMY(N1),NA,IOPTT,SYM,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(N4),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=NB(1)
DO 650 I=1,NAX
   L=L + NAX +1
   L1 = L + N
   M1 = L + N2-1
   DUMMY(L1) = BETA = B(L)
   DUMMY(M1) = BETA + B(L)
   650 CONTINUE
N3 = N2 + M
CALL TRAMP(DUMMY(N1),NB,DUMMY(N3),NB)
CONTINUE
CALL EQUATE(DUMMY(N3),NA,DUMMY(N1),NR)
CALL TRAN(DUMMY(N2),NB,DUMMY(N3),NB)
CALL EQUATE(DUMMY(N3),NB,DUMMY(N2),NB)
CALL TRAN(C,NC,DUMMY(N3),NDUM)
NSDUM = NDUM(?)
NDUM(2) = NDUM.2 + NB(2)
IFAC = 0
N4 = N3 + NC(1) * NC(2)
NS = 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)I = R IS SINGULAR, INCREASE
BETA ')
CALL TRAN(DUMMY(N2),NB,DUMMY(N1),NB)
NDUM(2) = NSDUM
CALL TRAN(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,IOP(1),SYM,DUMMY(N2))
C
CONTINUE
IF( IOP(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), NDUM1(2), NDUM2(2), NDUM3(2), NDUM4(2)
LOGICAL SYM
IF (IOP .EQ. 0) GO TO 250
IF (SYM) GO TO 100
CALL LNCNT(3)
PRINT 50
50 FORMAT(/, 'LINEAR EQUATION SOLVER  AX + XB = C ')
CALL PRINT(A, NA, 4H A , 1)
CALL PRINT(B, NB, 4H B , 1)
GO TO 200
100 CONTINUE
CALL LNCNT(3)
PRINT 150
150 FORMAT(/, 'LINEAR EQUATION SOLVER  (B TRANSPOSE)X + XB = C ')
CALL TRAMP(A, NA, DUMMY, NDUM1)
CALL PRINT(DUMMY, NDUM1, 4H B , 1)
200 CONTINUE
CALL PRINT(C, NC, 4H C , 1)
C
250 CONTINUE
CALL EQUATE(A, NA, DUMMY, NDUM1)
30 N1 = (NA(1) + 2) + 1
31 N2 = N1 + NA(1) - 1
32 D1 = 300I = N1, N2
33 DUMMY(1) = 0.0
34 300 CONTINUE
35 NDUM1(2) = NDUM1(2) + 1
36 NDUM2(1) = 1
37 NDUM2(2) = NDUM1(2)
38 NDUM1(1) = NDUM1(2) + 1
39 CALL NULL(DUMMY(N1), NDUM2)
40 LU = (NA(1) + 1)**2 + 1
41 CALL JUXTR(DUMMY, NDUM1, DUMMY(N1), NDUM2, DUMMY(LU), NDUM3)
42 CALL EQUATE(DUMMY(LU), NDUM3, DUMMY, NDUM1)
43 N = NA(1) + 1
44 310 CONTINUE
45 IF (SYM) GO TO 500
C
46 CALL EQUATE(B, NB, DUMMY(LU), NDUM2)
47 M1 = LU + NB(1) + 2
48 M2 = M1 + NB(1) - 1
49 D040 I = M1, M2
50 DUMMY(I) = 0.0
51 400 CONTINUE
52 NDUM2(2) = NDUM2(2) + 1
53 NDUM3(1) = 1
54 NDUM3(2) = NDUM2(2)
55 M1 = NDUM2(1)*NDUM2(2) + LU
56 CALL NULL(DUMMY(M1), NDUM3)
57 M2 = LU + (NA(1) + 1)**2
58 CALL JUXTR(DUMMY(LU), NDUM2, DUMMY(M1), NDUM3, DUMMY(M2), NDUM4)
59 CALL EQUATE(DUMMY(M2), NDUM4, DUMMY(LU), NDUM2)
60 M = NB(1) + 1
61 LNB = LU
62 LU = LU + (NB(1) + 1)**2
63 LV = LU + NA(1)**2
64 CALL AXPY(DUMMY, DUMMY(LU), N(1), N, NA(1), DUMMY(LNB), DUMMY(LV), NB(1)
   1), M, NB(1), C, NC(1), EPSA, EPSB, DFAIL)
GO TO 600

500 CONTINUE

CALL TRANP(DUMMY, NOU1, DUMMY(LU), NOUM2)
CALL EQUATE(DUMMY(LU), NOUM2, DUMMY, NOUM1)
CALL ATXPXA(DUMMY, DUMMY(LU), C, NA(1), N, NA(I), NC(1), EPSA, NFAIL)

600 CONTINUE

IF(NFAIL .EQ. 0) GO TO 700

CALL LNCNT(3)

RETURN

650 FORMAT(/,53) IN BARSTH, EITHER THE SUBROUTINE ATXPXA OR ATXPXA IS UNABLE TO REDUCE A OR B 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(NTE0013)) 
IF( IERR .EQ. 0 ) GO TO 200 
CALL LCNCT(4) 
PRINT 100,IERR 
100 FORMAT(///, ' IN TESTSA, THE ',I5, ' EIGENVALUE OF A HAS NOT BEEN FOUND AFTER ',I5, ' ITERATIONS',/) 
RETURN 
200 CONTINUE 
NDUM1(1) = NEVL 
NDUM1(2) = 1 
CALL JUXTA(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 LCNCT(4) 
PRINT 700 
700 FORMAT(///, ' PROGRAM TO TEST THE RELATIVE STABILITY OF THE MATRIX A',/) 
CALL PRNT(A,NA,4H A ,1) 
CALL LCNCT(4) 
PRINT 750 
750 FORMAT(///, ' EIGENVALUES OF A ',/) 
CALL PRNT(DUMMY,NDUM2,4HEVL,1) 
IF( .NOT. DISC ) GO TO 850 
CALL LCNCT(4) 
PRINT 800 
800 FORMAT(///, ' MODULI OF EIGENVALUES OF A ',/) 
CALL PRNT(DUMMY,N,NDUM1,4HMODA,1) 
C 
850 CONTINUE 
CALL LCNCT(4)
IF(STATE) PRINT 900
ALPHA
950 FORMAT(3H, ALPMA)
950 ALPHA IS STABLE
950 ALPH
950 IS UNSTABLE
950 RELATIVE
950 TO .E16.8/)
950 FORMAT(3H, ALPMA)
950 ALPHA IS STABLE
950 ALPH
950 IS UNSTABLE
950 RELATIVE
950 TO .E16.8/)
RETURN
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/CONV/SUMCV,PICTCV,SECV,MAXSUM
C
N = NA(1)
L = (N+2) + 1
TT = T
NEXPA(1) = NA(1)
NEXPA(2) = NA(2)
C
CALL MAXEL(A,NA,ANAA)
ANAA = ANAA*TT
ANAA = OARS(ANAA)
IF( ANAA .GT. 1.E-15 ) GO TO 100
CALL UNITY(EXPA,NEXPA)
GO TO 900
C
100 CONTINUE
IOP = 2
CALL NORMS(N,N,N,A,IOP,ZERO)
ZERO = ZERO / (2.**47)
CALL TRCE(A,NA,TR)
TR = TR/N
DO 200 I = 1,N
M = N*N(I) + 1
A(M) = A(M) - TR
200 CONTINUE
C
IOP = 1
CALL NORMS(N,N,N,A,IOP,COL)
IOP = 3
CALL NORMS(N,N,N,A,IOP,ROW)
ANORM = ROW
I = 0
ANO = 1
IF( ANORM .GT. COL ) ANORM = COL
TMAX = 1./ANORM
K = 0
C
300 CONTINUE
IF( TMAX = TT ) 325,350,350
325 CONTINUE
K = K + 1
TT = T/(2**K)
IF( K .GE. 1000 ) 300,700,700
350 CONTINUE
SC = TT
CALL SCALE(A,NA,ANAA,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./I
CALL SCALE(EXPA,EXPA,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
G O T O 600
550 CONTINUE
CALL EQUATE(EXPA,NEXPA,DUMMY,NA)
      IT = IT + 1
GO TO 400
57 c
600 CONTINUE
IF(K) 625,675,650
625 CONTINUE
CALL LNCNT(1)
PRINT 635
635 FORMAT( ' ERROR IN EXPSER, K IS NEGATIVE ' )
RETURN
7 C
650 CONTINUE
DO 680 I = 1,K
       TT = 2*TT
       CALL EQUATE(EXPA,NEXPA,DUMMY,NA)
       CALL EQUATE(DUMMY(NA),DUMMY(L),NA)
       CALL MULT(DUMMY(L),NA,DUMMY,NA,EXPA)
700 CONTINUE
T = TT
4 675 CONTINUE
S = 1./SC
CALL SCALE(A,NA,A,NA,S)
DO 685 I = 1,N
     M = I + N*(I-1)
     A(M) = A(M) + TR
     IF( NABS(A(M)) .LE. ZERO.) A(M) = 0.0
1 689 CONTINUE
2 C
3 TR = TR*T
S = DEXP(1.)
CALL SCALE(EXPA,NEXPA,EXPA,NEXPA,S)
GO TO 800
7 C
700 CONTINUE
CALL LNCNT(1)
PRINT 750
750 FORMAT( ' ERROR IN EXPSER, K = 1000 ' )
RETURN
1 C
400 CONTINUE
IF( IOP .NE. 0 ) RETURN
CALL LNCNT(4)
PRINT 825
825 FORMAT( // ' COMPUTATION OF THE MATRIX EXPONENTIAL EXP(A T) BY THE ' )
       EXP0109 1ERIES METHOD ' )
       CALL PRNT(A,NA,4H A ,1)
       CALL LNCNT(3)
       PRINT 850,T
850 FORMAT( ' T = ',7F9.9)
       CALL PRNT(EXPA,NEXPA,4HEXP,A,1)
RETURN
SUBROUTINE EXPAD (MAX, N, A, EA, IDIG, WK, IERR)

C 

REAL*8, DIMENSION (MAX,N) A(MAX,N), EA(MAX,N), WK(N,1), C(9)

IERR = 0

C 

CALCULATE NORM OF A

ANORM = 0

DO 10 I=1,N

S = 0

DO 5 J=1,N

S = S + DABS(A(I,J))

CONTINUE

10 IF (S .GT. ANORM) ANORM = S

CONTINUE

C ****

CALCULATE ACCURACY ESTIMATE

DIGC = 24.*DFLOAT(N)

IF (ANORM .GT. 1.) DIGC = DIGC*ANORM

DIGC = DIGC/15

IDIG = 15 - IFIX(ALOG10(DIGC) .+ 5)

C DETERMINE POWER OF TWO AND NORMALIZATION FACTOR

M = 0

DO 15 M=1,46

IF (ANORM .LE. 1.) GO TO 27

FACTOR = 2.

FACTOR = FACTOR*2.

15 CONTINUE

GO TO 125.

C ****

NORMALIZE MATRIX

DO 25 I=1,N

DO 27 J=1,N

A(I,J) = A(I,J)/FACTOR

CONTINUE

25 CONTINUE

27 CONTINUE.

C ****

SET COEFFICIENTS FOR (9,9) PADE TABLE ENTRY

C(1) = 5

C(2) = 1.1764705882352D-01

C(3) = 1.71568627450980D-02

C(4) = 1.71568627459080D-03

C(5) = 1.22549019607840D-04

C(6) = 6.28456510809450D-06

C(7) = 2.4448753860510D-07

C(8) = 5.1011080422450D-09

C(9) = 6.6789782476050D-11

C ****

CALCULATE PADE NUMERATOR AND DENOMINATOR BY COLUMNS

NP1 = N+1

NP7 = N+7

DO 95 J=1,N

C ****

COMPUTE JTH COLUMN OF FIRST NINE POWERS OF A

DO 35 I=1,N

S = 0.

35 CONTINUE
ON 30 PRINT N

45 CONTINUE

66 WK(I,NP1) = S

75 CONTINUE

80 DO 45 K=NP1,N7

90 KP1 = K+1

100 DO 45 I=1,N

110 S = S + A(I,L)*WK(L,K)

120 CONTINUE

130 WK(I,KP1) = S

145 CONTINUE

160 CONTINUE

170 Collect terms for Jth column of numerator and denominator

180 CONTINUE

190 DO 85 I=1,N

200 S = 0

210 DO 65 L=1,B

220 K = N+L

230 KN1 = K+N+1

240 P = C(KN1)*WK(I,K)

250 S = S + P

260 CONTINUE

270 S = WK(I,K)

280 IF (I.EQ.J) GO TO 80

290 S = S + EA(I,J)

300 WK(I,J) = S

310 CONTINUE

320 CALL GAUSEL (MAX,N,WK,N,EA,IERR)

330 IF (IERR .NE. 0) GO TO 130

340 IF ('4, EQ. 0) GO TO 130

350 TAKE OUT EFFECT OF NORMALIZATION ON EXP(A)

360 CONTINUE

370 DO 120 K=1,M

380 DO 110 I=1,N

390 S = 0

400 DO 105 L=1,N

410 S = S + EA(I,L)*EA(L,J)

420 CONTINUE

430 WK(I,J) = S

440 CONTINUE

450 DO 115 I=1,N

460 DO 115 J=1,N

470 A=45

480 DO 120 K=1,M

490 DO 110 I=1,N

500 S = 0

510 DO 105 L=1,N

520 S = S + EA(I,L)*EA(L,J)

530 CONTINUE

540 WK(I,J) = S

550 CONTINUE

560 DO 115 I=1,N

570 DO 115 J=1,N

580 A=45
I

\[
E(i,j) = w(k(i,j))
\]

CONTINUE

CONTINUE

C ****

UN-NORMALIZE A

DO 122 I=1,N

DO 122 J=1,N

A(i,j) = A(i,j) * FACTOR

CONTINUE

GO TO 130

C ****

NORM OF A IS EXCESSIVE

IERR = 1

EXIT ROUTINE

CONTINUE

RETURN

END
SUBROUTINE EXPINT(A, NA, B, NB, C, NC, T, IOPT, DUMMY)

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

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

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, NA, IOPT, COL)

IOPT = 3

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

ANAA = COL

IF( ANAA .GT. ROW ) ANAA = ROW

TMAX = 1./ANAA

K = 0

100 CONTINUE

IF( TMAX = TT ) 125, 150, 150

125 CONTINUE

K = K + 1

TT = T/(2**K)

IF( K = 1000 ) 100, 600, 600

150 CONTINUE

SC = TT

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

CALL UNITY(B, NB)

CALL SCALE(B, NB, DUMMY, NB, TT)

S = TT/2.

CALL SCALE(A, NA, DUMMY(L), NA, S)

II = 2

CALL ADD(DUMMY, NA, DUMMY(L), NA, DUMMY(L), NA)

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

CALL EQUATE(A, NA, C, NC)

200 CONTINUE

CALL MULT(A, NA, C, NC, B, NB)

S = 1./II

CALL SCALE(B, NB, C, NC, S)

CALL MAXEL(DUMMY, NA, TOT)

CALL MAXEL(C, NC, DELT)

IF( TOT .GT. 1.0 ) GO TO 300

IF( DELT/TOT .LT. SERCV ) GO TO 300

GO TO 350

300 CONTINUE

IF( DELT .LT. SERCV ) GO TO 300

350 CONTINUE

S = TT/(II + 1)

CALL SCALE(C, NC, D, NB, S)

CALL ADD(D, NB, DUMMY(L), NB, DUMMY(L), NB)

CALL ADD(C, NC, DUMMY, NC, DUMMY, NC)

II = II + 1

GO TO 200

C

400 CONTINUE

CALL EQUATE(DUMMY, NB, B, NB)

IF( K ) 425, 500, 450

425 CONTINUE

CALL LNCNT(1)

A-48
3  PRINT 435
4  435 FORMAT(• ERROR IN EXPINT, K IS NEGATIVE•)
5  RETURN
6 C
7  450 CONTINUE
8 DO 475 J = 1,k
9  T = 2*TT
10  CALL EQUATE(B,NB,DUMMY,NA)
11  CALL ADD(DUMMY,NA,DUMMY(L),NA,C,NC)
12  CALL MULT(DUMMY,NA,DUMMY(L),NA,C,NC,DUMMY(L),NC)
13 CALL MULT(DUMMY,NA,DUMMY,NA,B,NB)
14  475 CONTINUE
15  T = TT
16 C
17  500 CONTINUE
18 CALL EQUATE(DUMMY(L),NC,C,NC)
19  S = 1./SC
20  CALL SCALE(A,NA,A,NA,S)
21 C
22 IF (IOP .EQ. 0) RETURN
23 CALL LNCNT(5)
24 PRINT 550
25  550 FORMAT(• COMPUTATION OF THE MATRIX EXPONENTIAL "EXP(A'T)",/,
26  1ND ITS INTEGRAL OVER (0,T) BY THE SERIES METHOD "/,)
27  CALL PRINT(A,NA,4M A ,1)
28  CALL LNCNT(3)
29  PRINT 575, T
30  575 FORMAT(• T = •,D16.8,/) 
31  CALL PRINT(B,NB,4HEXA,1)
32  CALL PRINT(C,NC,4MINT ,1)
33  RETURN
34 C
35  600 CONTINUE
36  CALL LNCNT(1)
37  PRINT 650
38  650 FORMAT(• ERROR IN EXPINT, K = 1000 •)
39  RETURN
40 C
41 END
SUBROUTINE VARANC(A, NA, G, NG, Q, NO, W, NW, IDENT, DISC, IOPT, DUMMY)

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

DIMENSION A(1), G(1), NG(1), W(1), DUMMY(1)

DIMENSION NA(2), NG(2), NO(2), NW(2), NDU(2), IOPT(3), IOPT(2)

LOGICAL IDENT, DISC, SYM

COMMON/TOL/EPSAM, EPSRM, IACM

IF (IOP(1) .EQ. 0) GO TO 100

CALL LNCT(5)

IF (DISC) PRINT 25

IF (.NOT. DISC) PRINT 35

25 FORMAT(/,' PROGRAM TO SOLVE FOR THE STEADY-STATE VARIANCE MATRIX'/)

1 1, /* FOR A LINEAR DISCRETE SYSTEM*/

35 FORMAT(/,' PROGRAM TO SOLVE FOR THE STEADY-STATE VARIANCE MATRIX'/)

1 /* FOR A LINEAR CONTINUOUS SYSTEM*/

4 CALL PRNT(A, NA, 4H 0 , 1)

5 IF (.NOT. IDENT) GO TO 55

6 CALL LNCT(3)

7 PRINT 45

8 G IS AN IDENTITY MATRIX */

9 GO TO 65

55 CONTINUE

10 CALL PRNT(G, NG, 4H G , 1)

65 CONTINUE

11 IF (.NOT. IDENT) GO TO 85

12 CALL LNCT(3)

13 PRINT 75

14 INTENSITY MATRIX FOR COVARIANCE OF PROCESS NOISE */

15 GO TO 85

85 CONTINUE

16 CALL PRNT(G, NO, 4H G , 1)

20 CALL MULT(G, NG, G, NO, W, NW, NW)

25 IF (IOP(1) .EQ. 0) GO TO 200

30 CALL PRNT(G, NO, 4H G , 1)

100 CONTINUE

31 IF (IDENT) GO TO 200

32 CALL MULT(G, NG, Q, NO, DUMMY, NG)

33 N1 = NG(1) * NG(2) + 1

34 CALL TRANP(G, NG, DUMMY(N1), NDU1)

35 CALL MULT(DUMMY, NG, DUMMY(N1), NDU1, G, NG)

37 CALL IOP(1) .EQ. 0 ) GO TO 200

38 CALL LNCT(3)

40 PRINT 75

41 CALL PRNT(G, NO, 4H G , 1)

42 CALL MULT(G, NG, DUMMY, NA)

43 CALL TRANP(A, NA, A, NA, W, NW, NW, NDU, IOPT, BETA, SYM, DUMMY)

44 GO TO 400

200 CONTINUE

45 IF (.NOT. DISC) CALL SCALE(N, NW, W, NW, 1.0)

46 IOPT(1) = IOPT(2)

47 IOPT(2) = 1

48 SYM = .TRUE.

49 IF (DISC) GO TO 300

50 CALL BILIN(A, NA, A, NA, W, NW, IOPT, BETA, SYM, DUMMY)

51 GO TO 400

250 CONTINUE

52 CALL RARSNW(A, NA, A, NA, W, NW, IOPT, SYM, EPSA, EPSA, DUMMY)

53 GO TO 400

250 CONTINUE

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

55 GO TO 400

250 CONTINUE

56 CALL TRANP(A, NA, DUMMY(N1), NA)

57 300 CONTINUE

N = NA(1) + 2

59 N1 = N + 1

60 CALL TRANP(A, NA, DUMMY(N1), NA)

61 N2 = N1 + N

62
CALL SUM(DUMMY, NA, NW, DUMMY(N1), NA, IOPT, SYM, DUMMY(N2))
C 400 CONTINUE
IF (IOP(1) .EQ. 0) RETURN
CALL LNCT(3)
PRINT 450
FORMAT(/, * VARIANCE MATRIX */)
CALL PRNT(W, NW, 4H W , 1)
C
RETURN
END
SUBROUTINE CTROL(A,NA,R,N4,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 = N1+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)
CALL JUXTC(DUMMY(N2),NV,DUMMY(N1),NB,C,NC)

IF( J .EQ. K ) GO TO 200

CALL EQUATE(C,NC,DUMMY(N2),NV)

GO_TO 100

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

FORMAT(//,' THE MATRIX C IS THE CONTROLLABILITY MATRIX FOR THE A/CTR003
18 PAIR'/)

CALL PRNT(C,NC,4H C ,1)

IF( IOP(2) .EQ. 0 ) RETURN
NOS = 0
IOPT = 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(IOPT,NV(1),NV(2),NV(1),NV(2),DUMMY,NOS,B,IAC,ZTEST,DUM
MY(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

FORMAT(//,' IN CTROL, SNVDEC HAS FAILED TO CONVERGE TO THE '+,I4,' SINGULAR VALUE AFTER 30 ITERATIONS ')

FORMAT(//,' IN CTROL, THE MATRIX SUBMITTED TO SNVDEC USING ZTEST = '+,I4,' IS CLOSE TO A MATRIX WHICH IS OF LOWER RANK'+,I4,' CURRENT RANK '+,I4,' MAY ALSO BE REDUCED '+,I4,' CURRENT RACK '+,I4,'6)

310 FORMAT(//,' BASED ON THE ZERO-TEST '+,D16.9,' THE RANK OF THE CONTRC

THE SINGULAR VALUES ARE * */
SUBROUTINE TRNSIT(A, NA, B, NB, H, NH, G, NG, F, NF, V, NV, T, NX, DISC, STABL

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

DIMENSION A(1), B(1), H(1), G(1), F(1), V(1), T(1), IOPT(4)

DIMENSION NA(2), NB(2), NH(2), NG(2), NF(2), NV(2), NX(2), T(2), IOPT(4)

DIMENSION NDUM1(2), NDUM2(2)

LOGICAL DISC, STABLE

N = NA(1)*NA(2)
N1 = N + 1
N2 = N + N1
N3 = N + N2
N4 = N + N3
N5 = N + N4
N6 = N + N5

CALL LNCNT(4)

IF(DISC) PRINT 100
IF(.NOT. DISC) PRINT 120

100 FORMAT(//, 'COMPUTATION OF TRANSIENT RESPONSE FOR THE DIGITAL SYSTEM',//)
120 FORMAT(//, 'COMPUTATION OF TRANSIENT RESPONSE FOR THE CONTINUOUS SYSTEM',//)

CALL PRNT(A, NA, 4H A , 1)
CALL PRNT(B, NB, 4H B , 1)

IF( (IOPT(1) .NE. 1) .AND. (IOPT(1) .NE. 0 ) ) GO TO 180

CALL LNCNT(3)

IF( IOPT(1) .EQ. 0 ) PRINT 140
IF( IOPT(1) .EQ. 1 ) PRINT 160

140 FORMAT(//, 'H IS A NULL MATRIX ')
160 FORMAT(//, 'H IS AN IDENTITY MATRIX ')

GO TO 200

180 CONTINUE

CALL PRNT(H, NH, 4H H , 1)

200 CONTINUE

IF( (IOPT(2) .NE. 1) .AND. (IOPT(2) .NE. 0 ) ) GO TO 260

CALL LNCNT(3)

IF( IOPT(2) .EQ. 0 ) PRINT 220
IF( IOPT(2) .EQ. 1 ) PRINT 240

220 FORMAT(//, 'G IS A NULL MATRIX ')
240 FORMAT(//, 'G IS AN IDENTITY MATRIX ')

GO TO 260

260 CONTINUE

CALL PRNT(G, NG, 4H G , 1)

280 CONTINUE

IF( (IOPT(3) .NE. 0) .AND. (IOPT(3) .NE. 1) ) GO TO 295

CALL LNCNT(3)

IF(IOPT(3) .EQ. 0 ) PRINT 285
IF(IOPT(3) .EQ. 1 ) PRINT 290

285 FORMAT(//, 'V IS A NULL MATRIX ')
290 FORMAT(//, 'V IS AN IDENTITY MATRIX ')

GO TO 300

300 CONTINUE

CALL PRNT(V, NV, 4H V , 1)

320 CONTINUE

CALL EQUATE(A, NA, DUMMY(N6), NA)
CALL MULT(R, NB, F, NF, DUMMY, NA)
CALL SUBT(A, NA, DUMMY, NA, A, NA)

IF(DISC) GO TO 350

NMAX = T(1)/T(2)
ILOPT = 1
63 TT = T(2)
64 IF(IOPT(3) .NE. 0 ) GO TO 315
65 CALL EXPSER(A,NA,DUMMY,NA,TT,IOPT,DUMMY(N1))
66 GO TO 400
67 315 CONTINUE
68 CALL EXPINT(A,NA,DUMMY,NA,DUMMY(N1),NA,TT,IOPT,DUMMY(N2))
69 CALL MULT(DUMMY(N1),NA,B,NB,DUMMY(N2),NB)
70 IF(IOPT(3) .NE. 1 ) GO TO 325
71 CALL EQUATE(DUMMY(N2),NB,DUMMY(N1),NX)
72 GO TO 400
73 325 CONTINUE
74 CALL MULT(DUMMY(N2),NB,NV,DUMMY(N1),NX)
75 GO TO 400
76 350 CONTINUE
77 NMAX = IOPT(4)
78 IF(IOPT(3) .NE. 0 ) GO TO 400
79 CALL MULT(B,NB,NV,DUMMY(N1),NX)
80 CALL MULT(DUMMY(N2),NB,NV,DUMMY(N1),NX)
81 C
82 400 CONTINUE
83 CALL LNCNT(4)
84 PRINT 420
85 420 FORMAT(/, ' STRUCTURE OF PRINTING TO FOLLOW',/)
86 CALL LNCNT(6)
87 PRINT 440
88 440 FORMAT( ' TIME OR STAGE',/,' STATE = X TRANSPOSE = FROM DX = AX',/)
89 1 ' + BU=',/,' OUTPUT = Y TRANSPOSE = FROM Y = HX + GU IF DIFFERENT',/)
90 2 ' FROM X',/,' CONTROL = U TRANSPOSE = FROM U = -FX + V',//)
91 C
92 K = 0
93 L = 0
94 CALL SCALE(F,NF,F,NF,-1.0)
95 C
96 450 CONTINUE
97 IF(K .GT. NMAX ) GO TO 800
98 CALL MULT(F,NF,X,NX,DUMMY(N2),NV)
99 IF(IOPT(3) .NE. 0 ) CALL ADD(DUMMY(N2),NV,V,NV,DUMMY(N2),NV)
100 CALL MULT(DUMMY,NA,X,NX,DUMMY(N3),NX)
101 IF(IOPT(3) .EQ. 0 ) GO TO 475
102 CALL ADD(DUMMY(N1),NX,DUMMY(N3),NX,DUMMY(N3),NX)
103 475 CONTINUE
104 IF(IOPT(2) .EQ. 0 ) GO TO 525
105 IF(IOPT(2) .EQ. 1 ) GO TO 500
106 CALL MULT(G,NG,DUMMY(N2),NV,DUMMY(N4),NDUM1)
107 GO TO 525
108 500 CONTINUE
109 CALL EQUATE(DUMMY(N2),NV,DUMMY(N4),NDUM1)
110 525 CONTINUE
111 IF(IOPT(1) .EQ. 0 ) GO TO 575
112 IF(IOPT(1) .EQ. 1 ) GO TO 550
113 CALL MULT(H,NH,X,NX,DUMMY(N5),NDUM1)
114 GO TO 575
115 550 CONTINUE
116 CALL EQUATE(X,NX,DUMMY(N5),NDUM1)
117 575 CONTINUE
118 IF(IOPT(2) .EQ. 0 ) GO TO 600
119 IF(IOPT(1) .EQ. 0 ) GO TO 700
120 CALL ADD(DUMMY(N4),NDUM1,DUMMY(N5),NDUM1,DUMMY(N4),NDUM1)
121 GO TO 700
122 600 CONTINUE
123 IF(IOPT(1) .NE. 0 ) CALL EQUATE(DUMMY(N5),NDUM1,DUMMY(N4),NDUM1)
124 C
125 700 CONTINUE
126 CALL LNCNT(5)
127 IF(.NOT., DISC ) GO TO 720
128 PRINT 710,K
129 710 FORMAT(/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/,/
SUBROUTINE SAMPLE(A,NA,B,NB,Q,NG,R,NR,W,NW,T,IOP,DUMMY)
IMPLICIT REAL*8 (A-H,Z)
DIMENSION A(1),R(1),Q(1),W(1),DUMMY(1)
DIMENSION NA(2),NB(2),NG(2),NR(2),NW(2),IOP(2),DU(2)
COMMON/CONV/SUMCV,RICTCV,SECV,MAXSUM
IF( IOP(1) .EQ. 0 ) GO TO 100
IF( IOP(2) .EQ. 0 ) GO TO 50
CALL LNCNT(5)
PRINT 25
25 FORMAT(/, ' COMPUTATION OF WEIGHTING MATRICES FOR THE OPTIMAL SAMPLE
1LED-DATA REGULATOR PROBLEM',//)
CALL PRNT(A,NA,4H A,1)
CALL PRNT(B,NB,4H B,1)
CALL LNCNT(3)
PRINT 35
35 FORMAT(/, ' CONTINUOUS PERFORMANCE INDEX WEIGHTING MATRICES',//)
CALL PRNT(A,NA,4H A,1)
CALL PRNT(Q,NA,4H Q,1)
CALL LNCNT(3)
PRINT 85,T
85 FORMAT(/, ' SAMPLE TIME = ',16.8,/) GO TO 100
50 CONTINUE CALL LNCNT(8)
PRINT 75
75 FORMAT(/, ' COMPUTATION OF THE RECONSTRUCTIBILITY GRAMIAN',//, ' FOR
1. THE (A,H) SYSTEM OVER THE INTERVAL (0,T) ',//, ' THE MATRIX Q IS
2 H TRANPOSE ) X H',//)
CALL PRNT(A,NA,4H A,1)
CALL PRNT(Q,NA,4H Q,1)
CALL LNCNT(3)
PRINT 85,T
85 FORMAT(/, ' T = ',16.8,/) 100 CONTINUE
N = NA(1)
L = ( N**2)
N1 = L + 1
N2 = N1 + L
TT = T
IOPT = 1
CALL NORMS(N,N,N,A,IOPT,ANORM)
IOPT = 3
CALL NORMS(N,N,N,A,IOPT,ROWA)
IF( ANORM .GT. ROWA ) ANORM = ROWA
IF( ANORM .LE. 1.E-15 ) GO TO 900
TMAX = 1.0/ANORM
K = 0
125 CONTINUE
IF( TMAX = TT ) 150,150,200
150 CONTINUE
K = K + 1
TT = T/( 2**K)
IF( K = 1000 ) 125,800,800
63 C 200 CONTINUE
64 C
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 IF(IOP(2) .NE. 0) GO TO 500
72 C
73 C 225 CONTINUE
74 II = I + 2
75 I = I + 1
76 F = 1.0/II
77 CALL SCALE(A, NA, DUMMY(N1), NA, F)
78 CALL MULT(DUMMY(N1), NA, DUMMY(N2), NA)
79 CALL TRANP(DUMMY(N2), NA, DUMMY(N1), NA)
80 CALL ADD(DUMMY(N1), NA, DUMMY(N2), NA, DUMMY, NA)
81 CALL MAXEL(Q, NO, TOT)
82 C
83 CALL MAXEL(DUMMY, NA, DELT)
84 IF(TOT .GT. 1.0) GO TO 250
85 IF(DELT/TOT .LT. SERCV) GO TO 300
86 GO TO 275
87 250 CONTINUE
88 IF(DELT .LT. SERCV) GO TO 300
89 275 CONTINUE
90 CALL ADD(Q, NO, DUMMY, NA, Q, NO)
91 GO TO 225
92 C 300 CONTINUE
93 C
94 IF(K .EQ. 0) GO TO 400
95 N3 = N2 + L
96 G = 1.0
97 IOPT = 0
98 CALL EXPSER(A, NA, DUMMY, NA, G, IOPT, DUMMY(N1))
99 CALL TRANP(DUMMY, NA, DUMMY(N1), NA)
100 CALL MULT(Q, NO, DUMMY, NA, DUMMY(N2), NA)
101 CALL MULT(DUMMY(N1), NA, DUMMY(N2), NA, DUMMY(N3), NA)
102 CALL ADD(Q, NO, DUMMY, NA, NA, NO)
103 CALL MULT(DUMMY, NA, DUMMY, NA, DUMMY(N1), NA)
104 CALL EQUATE(DUMMY(N1), NA, DUMMY, NA)
105 C GO TO 350
106 350 CONTINUE
107 IF(K .EQ. 0) GO TO 400
108 K = K - 1
109 CALL TRANP(DUMMY, NA, DUMMY(N1), NA)
110 CALL MULT(Q, NO, DUMMY, NA, DUMMY(N2), NA)
111 CALL MULT(DUMMY(N1), NA, DUMMY(N2), NA, DUMMY(N3), NA)
112 CALL ADD(Q, NO, DUMMY, NA, NA, NO)
113 CALL MULT(DUMMY, NA, DUMMY, NA, DUMMY(N1), NA)
114 C GO TO 350
115 400 CONTINUE
116 S = 1.0/SC
117 CALL SCALE(A, NA, A, NA, S)
118 C IF(IOP(1) .EQ. 0) RETURN
119 CALL PRNT(Q, NO, 4HGRAM, 1)
120 RETURN
121 C 500 CONTINUE
122 500 CONTINUE
123 CALL SCALE(B, NB, B, NB, TT)
124 N3 = N2 + L
126 \( N_4 = N_3 + L \)  
127 \( N_5 = N_4 + L \)  
128 \( N_b = N_5 + L \)  
129 C  
130 525 CONTINUE  
131 \( I = I + 2 \)  
132 \( I = I + 1 \)  
133 \( F = I_0 / I \)  
134 CALL SCALE(A, NA, DUMMY(N1), NA, F)  
135 CALL TRAP(DUMMY(N1), NA, DUMMY(N2), NA)  
136 CALL MULT(DUMMY, NA, DUMMY(N1), NA, DUMMY(N3), NA)  
137 CALL TRAP(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, NO, Q, NO)  
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 TRAP(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 TRAP(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, NO, Q, NO)  
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, NO, TOT)  
164 CALL MAXEL(DUMMY, NO, 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, NO, DUMMY, NO, Q, NO)  
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
IF( k .EQ. 0 ) GO TO 700

G = 1.0

IF( k .EQ. 0 ) GO TO 700

CALL EXPINT(A,NA,DUMMY,NA,DUMMY(N1),NA,G,IOPT,DUMMY(N2))

CALL MULT(DUMMY(N1),NA,B,NB,DUMMY(N2),NB)

CALL EQUATE(DUMMY(N2),NB,DUMMY(N1),NB)

650 CONTINUE

IF( k .EQ. 0 ) GO TO 700

K = K - 1

CALL MULT(G,NG,DUMMY,NA,DUMMY(N2),NA)

CALL TRAPN(DUMMY,NA,DUMMY(N3),NA)

CALL MULT(DUMMY(N3),NA,DUMMY(N2),NA,DUMMY(N5),NA)

CALL MULT(G,NG,DUMMY(N1),NB,DUMMY(N2),NB)

CALL ADD(G,NG,DUMMY(N5),NA,G,NG)

CALL MULT(DUMMY(N3),NA,DUMMY(N2),NB,DUMMY(N5),NB)

CALL MULT(DUMMY(N3),NA,W,NW,DUMMY(N6),NW)

CALL ADD(DUMMY(N5),NW,DUMMY(N6),NW,DUMMY(N5),NW)

CALL TRAPN(DUMMY(N1),NB,DUMMY(N6),NDUM)

CALL MULT(DUMMY(N6),NDUM,W,NW,DUMMY(N3),NR)

CALL ADD(W,NW,DUMMY(N5),NW,W,NW)

CALL MULT(DUMMY(N6),NDUM,W,NW,DUMMY(N3),NR)

CALL ADD(DUMMY(N5),NR,DUMMY(N3),NR,DUMMY(N5),NR)

CALL TRAPN(DUMMY(N3),NR,DUMMY(N6),NR)

CALL ADD(DUMMY(N5),NR,DUMMY(N6),NR,DUMMY(N5),NR)

CALL ADD(DUMMY(N6),NR,DUMMY(N4),NR,DUMMY(N4),NR)

CALL SCALE(DUMMY(N4),NR,DUMMY(N4),NR,2.0)

CALL MULT(DUMMY(N3),NA,DUMMY(N1),NB,DUMMY(N1),NB)

CALL MULT(DUMMY,NA,DUMMY,NA,DUMMY(N3),NA)

GO TO 650

700 CONTINUE

CALL SCALE(R,NR,R,NR,T)

IF( IOP(2) .NE. 0 ) GO TO 925

IF( IOP(1) .NE. 0 ) CALL PRNT(G,NO,4HGRAM,1)

RETURN

800 CONTINUE

CALL LNCNT(1)

PRINT 850

FORMAT(/',' DISCRETE PERFORMANCE INDEX WEIGHTING MATRICES/)

CALL PRNT(G,NG,4H Q ,1)

CALL PRNT(W,NW,4H W ,1)

CALL PRNT(R,NR,4H R ,1)

RETURN

900 CONTINUE

CALL SCALE(Q,NG,Q,NG,T)

IF( IOP(2) .NE. 0 ) GO TO 925

IF( IOP(1) .NE. 0 ) CALL PRNT(G,NG,4HGRAM,1)

RETURN

A-60
252 925 CONTINUE
253 CALL MULT(U, NO, B, NB, W, NW)
254 CALL SCALE(W, *, 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, 4H, Q, 1);  
265 CALL PRNT(W, NW, 4H, W, 1); 
266 CALL PRNT(R, NR, 4H, R, 1); 
267 RETURN
268 C
269 END
SUBROUTINE PREFIL(A,NA,B,NB,Q,NO,W,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),Q(2),W(2),R(2),F(2),NF(2),IOF(3)
IF( IOP(1) .EQ. 0 ) GO TO 100
CALL LNCT(5)
PRINT 25
25 FORMAT(/,,* PROGRAM TO COMPUTE PREFILTER GAIN & TO ELIMINATE CROSS* 
19-PRODUCT TERM ''/,,'' IN QUADRATIC PERFORMANCE INDEX '',/)
IF( IOP(3) .EQ. 0 ) GO TO 50
CALL PRNT(A,NA,AH,1)
CALL PRNT(B,NB,BH,1)
50 CONTINUE
CALL PRNT(Q,NO,QH,1)
CALL PRNT(W,NO,WH,1)
CALL PRNT(R,NR,RH,1)
100 CONTINUE
IF( IOP(2) .EQ. 0 ) GO TO 300
CALL TRAN(W,NW,F,NF)
CALL SCALE(F,NF,F,NF,0.5)
CALL EQUATE(R,NR,DUMMY,NR)
10 IFAC=0
20 N1=NR(1)**2+1
CALL SYMPDS(M,M,DUMMY,NF(2),F,IOPT,IFAC,DETERM,ISCALE,DUMMY(N1),I)
25 IF( IERR .EQ. 0 ) GO TO 200
CALL LNCT(4)
PRINT 150
30 150 FORMAT(/,,* IN PREFIL, THE MATRIX R IS NOT SYMMETRIC POSITIVE DEF* 
10-ITE*,/)
10 RETURN
C
100 CONTINUE
IF( IOP(2) .EQ. 0 ) GO TO 300
CALL MULT(W,NW,F,NF,DUMMY,NO)
CALL SCALE(DUMMY,NO,DUMMY,NO,0.5)
CALL SUBT(Q,NO,DUMMY,NO,Q,NO)
30 CONTINUE
IF( IOP(3) .EQ. 0 ) GO TO 400
CALL MULT(B,NB,F,NF,DUMMY,NA)
CALL SUBT(A,NA,DUMMY,NA,A,NA)
40 CONTINUE
IF( IOP(1) .EQ. 0 ) RETURN
CALL PRNT(F,NF,FH,1)
IF( IOP(2) .EQ. 0 ) GO TO 500
CALL LNCT(3)
PRINT 450
50 450 FORMAT(/,,* MATRIX Q = (W/2)*',/)
CALL PRNT(Q,NO,QHNEWQ,1)
C
500 CONTINUE
IF( IOP(3) .EQ. 0 ) RETURN
CALL PRNT(A,NA,AHNEWA,1)
RETURN
END
SUBROUTINE CSTAB(A, NA, B, NB, F, NF, 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), IOP(3), NDUM(2)

DIMENSION IOPT(2)

LOGICAL SYM

COMMON/TOL/EPASM, EPSCM, IACM

N = NA(1)**2

N1 = 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(1), DUMMY(2), ISV, ILV, V, DUMMY(2), N2)

CALL LNCNT(1)

PRINT 25, IERR

IF (IERR .EQ. 0) GO TO 50

CALL LNCNT(3)

FORMAT(' IN CSTAB, THE SUBROUTINE EIGEN FAILED TO DETERMINE THE EIGENVALUE FOR THE MATRIX A AFTER 30 ITERATIONS.')

IERR = 1

CALL NORMS(M, M, A, IERR, BETA)

BETA = 2.0 * BETA

GO TO 200

50 CONTINUE

BETA = 0.0

DO 75 I = 1, M

J = N1 + I - 1

BETA1 = DARS(DUMMY(J))

IF (BETA1 .GT. BETA) BETA = BETA1

75 CONTINUE

BETA = SCLE * (BETA + .001)

GO TO 200

100 CONTINUE

BETA = SCLE

200 CONTINUE

CALL TRANP(B, NB, DUMMY, NDUM)

CALL MULT(B, NB, DUMMY, NDUM, DUMMY(1), NA)

CALL SCALE(C, DUMMY(1), NA, DUMMY(1), -2.0)

CALL SCALE(A, NA, DUMMY(1), NA, -1.0)

J = -NA(1)

NAX = NA(1)

DO 225 I = 1, NAX

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 = EPSAM

CALL BARSTW(DUMMY(1), NA, A, NA, DUMMY, NA, IOPT, SYM, EPSA, EPSA, DUMMY(2)

11)
GO TO 350
300 CONTINUE
IOPT(2) = 1
CALL SBLIN(DUMMY(N1),NA,A,NA,DUMMY,NA,IOPT,ASCLE,SYM,DUMMY(N2))
350 CONTINUE
CALL EQUATE(B,NB,DUMMY(N1),NB)
IOPT(1) = 3
IAC = IACM
N3 = N2 + NA(1)
CALL SVDDEC(IOPT,NA(1),NA(1),NA(1),NA(1),DUMMY,NB(2),DUMMY(N1),IAC)
IF(IERR .EQ. 0 ) GO TO 400
CALL LNCNT(5)
IF(IERR .GT. 0 ) PRINT 360, IERR
IF(IERR .EQ. -1) PRINT 370, ZTEST, IERR
360 FORMAT(/, ' IN CSTAB, SVDDEC HAS FAILED TO CONVERGE TO THE ',14,'/'
      SINGULAR VALUE AFTER 30 ITERATIONS',/)
370 FORMAT(/, ' IN CSTAB, THE MATRIX SUBMITTED TO SVDDEC USING ZTEST = ',14,'/'
      ' IS CLOSE TO A MATRIX OF LOWER RANK ',14,'/'
      ' IF THE ACCURACY IAC IS REDUCED THE RANK MAY ALSO BE REDUCED',/,' CURRENT RANK ',14,'/)
5 = 1,14)
IF( IERR .GT. 0 ) RETURN
NDUM(1) = NA(1)
NDUM(2) = 1
CALL PRINT(DUMMY(N2),NDUM,4HSGVL,1)
400 CONTINUE
CALL TRANP(DUMMY(N1),NB,F,NF)
IF( IOPT(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 PRINT(A,NA,4H A ,1)
CALL LNCNT(4)
PRINT 550,BETA
550 FORMAT(/, ' BETA = ',14,'/'
      ' COMPUTATION OF F MATRIX SUCH THAT A-BF IS ASYMPTOTICALLY STABLE IN THE CONTINUOUS SENSE ',/)
CALL PRINT(B,NB,4H B ,1)
CALL PRINT(F,NF,4H F ,1)
CALL MULT(B,NB,F,NF,DUMMY,NA)
CALL SURT(A,NA,DUMMY,NA,DUMMY,NA)
CALL PRINT(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(NST)
13),IERP)
M = NA(1)+IERP
13)
IF( IERR .EQ. 0 ) GO TO 600
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(4)
\(d\) 700 I=1, M
' = N1+I-1
k = N2+I-1
\(A-64\)
PRINT 675, DUMMY(J), DUMMY(K)  
700 CONTINUE  
C  
RETURN  
END
SUBROUTINE DSTAB(A,NA,B,NB,F,NF,SING,IOP,SCALE,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/EPSTOLB,EPSTOLB,IACM

N = NA(1)**2

N1 = N + 1

N2 = N1 + N

IF( .NOT. SING ) GO TO 100

IOPT(1) = IOP(1)

IOPT(2) = 1

IOPT(3) = 0

CSCL = 1.05

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

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

CALL SIBST(A,NA,DUMMY,NA,DUMMY,NA)

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

GO TO 200

C

100 CONTINUE

CALL EQUATE(A,NA,DUMMY,NA)

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

C

200 CONTINUE

IF( IOP(2) .EQ. 0 ) GO TO 300

N3 = N2 + NA(1)

N4 = N3 + NA(1)

ISV = 0

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

M = NA(1)

MP = DUMMY(1)

IF( IERR .EQ. 0 ) GO TO 250

CALL LNCNT(3)

PRINT 225, IERR

FORMAT(//"IN DSTAB, THE PROGRAM EIGEN FAILED TO DETERMINE","
115, 'EIGENVALUE FOR THE MATRIX A=BG AFTER 30 ITERATIONS'")

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

IF( SING ) CALL PRNT(F,NF,4H,G,1)

RETURN

C

250 CONTINUE

ALPHA = 1.0

DO 275 I = 1,N

I1 = N2 + I - 1

I2 = N3 + I - 1

ALPHA1 = DSQRT(DUMMY(I1)**2 + DUMMY(I2)**2)

IF( ALPHA1 .LT. ALPHA .AND. ALPHA1 .NE. 0 ) ALPHA = ALPHA1

275 CONTINUE

ALPHA = SCLE*ALPHA

GO TO 400

C

300 CONTINUE

ALPHA = SCLE

C

400 CONTINUE

J = -NA(1)

NAX = NA(1)

DO 425 I = 1,NAX

J = J + NAX + 1

425 K = N1 + J - 1

DUMMY(K) = DUMMY(J) = ALPHA
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(N5)

IF (. IERR .EQ. 0) .GO TO 500
CALL LNCNT(3)
PRINT 435
3 FORMAT(/,, 4H IN DSTAB, GELIM HAS FOUND THE MATRIX ( A-BG) + (ALPHA))

I SINGULAR:
CALL PRINT(A, NA, 4M A ,1)
CALL PRINT(4F, 4M G ,1)
GO TO 465
CONTINUE
PRINT 455
4 FORMAT(/,, 4H IN DSTAB, GELIM HAS FOUND THE MATRIX A + (ALPHA)I SING)

I LAR *
CALL PRINT(A, NA, 4M A ,1)
CONTINUE
PRINT 465
5 FORMAT(/,, 4H RETURN

CALL EQUATE(DUMMY(N1), NA, DUMMY, NA)
CALL TRANP(DUMMY(N2), NB, DUMMY(N1), NDUM)
N3 = N2 + N
CALL MUL(DUMMY(N2), NB, DUMMY(N1), NDUM, DUMMY(N3), NA)
CALL SCALE(DUMMY(N3), NA, DUMMY(N1), NA, 4, 0)
SYM = .TRUE.
I OPT(1) = 0
EPSA = EPSAM
CALL BARSTW(DUMMY, NA, B, NB, DUMMY(N1), NA, I OPT, SYM, EPSA, EPSAM, DUMMY(N2))

CONTINUE
CALL EQUATE(DUMMY(N1), NA, DUMMY, NA)
CALL TRANP(B, NB, DUMMY(N1), NDUM)
CALL MUL(B, NB, DUMMY(N1), NDUM, DUMMY(N2), NA)
CALL ADD(DUMMY, NA, DUMMY(N2), NA, DUMMY(N1))
CALL EQUATE(A, NA, DUMMY(N1), NA)
IF (. NOT. SING ) .GO TO 600
CALL MUL(B, NB, F, NF, DUMMY(N1), NA)
CALL SUBT(A, NA, DUMMY(N1), NA, DUMMY(N1), NA)

CONTINUE
I OPT(1) = 3
M = NA(1)
I AC = IACM
CALL SNVDEC(IOPT, M, M, M, M, DUMMY, M, DUMMY(N1), IAC, ZTEST, DUMMY(N2), DUM
MY(N3), IRANK, APLUS, IERR)
IF ( IERR .EQ. 0 ) .GO TO 700
CALL LNCNT(5)

IF ( IERR .GT. 0 ) PRINT 625, IERR
IF ( IERR .EQ. -1) PRINT 650, ZTEST, IRANK
625 FORMAT(/,, 4H IN DSTAB, SNVDEC HAS FAILED TO CONVERGE TO THE "S", IS,
SINGULAR VALUE AFTER 30 ITERATIONS")
650 FORMAT(/,, 4H IN DSTAB, THE MATRIX SUBMITTED TO SNVDEC, USING ZTEST
I am close to a matrix of lower rank if the accuracy is reduced the rank may also be reduced, current rank DST.

If the accuracy is reduced the rank may also be reduced, current rank DST.

Call PRINT(DUMMY(N2),NDUM,4MSGVL,1).

700 CONTINUE
Call TRANP(B,NB,DUMMY(N2),NDUM).
Call MULT(DUMMY(N2),NDUM,DUMMY(N1),NA,DUMMY,NF).
If (.NOT. SING) Go To 800
Call ADD(F,NF,DUMMY,N,F,NF).
Go To 900
800 CONTINUE
Call EQUATE(DUMMY,NF,F,NF).
900 CONTINUE
If (IOP(1) .EQ. 0) Return.
Call LNCNT(4).
Print 1000.
1000 FORMAT(///,* COMPUTATION OF F SUCH THAT A-BF IS ASYMPTOTICALLY STABLE*).
Call PRNT(A,NA,4M A,1).
Call PRNT(B,NB,4M B,1).
Call LNCNT(4).
Print 1100, ALPHA.
1100 FORMAT(///,* ALPHA = *,D16.8,/) Call PRNT(F,NF,4M F,1).
Call MULT(B,NF,F,NF,DUMMY,NA).
Call SUBT(A,NA,DUMMY,NA,DUMMY,NA).
Call PRNT(DUMMY,NA,4M A-BF,1).
LNCNT(3).
Print 1200.
1200 FORMAT(///,* EIGENVALUES OF A-BF*).
NDUM(1) = NA(1).
NDUM(2) = 1.
N2 = N1 + NA(1).
N3 = N2 + NA(1).
ISV = 0.
Call EIGEN(NA(1),NA(1),DUMMY,DUMMY(N1),DUMMY(N2),ISV,ISV,V,DUMMY,NDUM).
IERR.
If (IERR .EQ. 0) Go To 1300.
Call LNCNT(3).
Print 1250.
1250 FORMAT(///,* IN Dstab, the program EIGEN failed to determine the*).
Call LNCNT(3).
1300 CONTINUE
Call JUXTC(DUMMY(N1),NDUM,DUMMY(N2),NDUM,DUMMY,NDUM).
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 PRNT(DUMMY,NDUM,4M MOD,1).
A-68
SUBROUTINE DISREG(A, NA, B, NB, H, NH, Q, NO, R, NR, F, NF, P, NP, IOP, IDENT, DU)

IMPLICIT REAL*8(A, H, Q, 0)

DIMENSION A(1, 0, B(1, 0, R(1, 0, F(1, 0, DUMMY(1, 0, 1)

DIMENSION NA(2, 0, NR(2, 0, Q(2, 0, NO(2, 0, 

DIMENSION IOP(3, 0, 0)

DIMENSION H(1, 0, NH(2, 0, NO(2, 0, 

LOGICAL IDENT

COMMON/TOL/EPSAM, EPSBM, IACM

COMMON/CONV/UMCV, RICTCV, SERCV, MAXSUM

N = NA(1) * 2

I = IOP(3, 0, 0)

IF(IOP(1, 0, 0) .EQ., 0) GO TO 85

CALL LNCNT(5)

PRINT 25

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

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

CALL PRNT(Q, NO, 4H Q, 1)

IF(.NOT., IDENT) GO TO 45

CALL LNCNT(3)

PRINT 35

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

GO TO 65

CALL PRNT(H, NH, 4H H, 1)

CALL MULT(Q, NO, H, NH, DUMMY, NH)

CALL TRANP(H, NH, DUMMY(N1), NF)

CALL MULT(DUMMY(N1), NF, DUMMY, NH, Q, NO)

CALL LNCNT(3)

PRINT 55

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

CALL PRNT(Q, NO, 4H QMT, 1)

CALL PRNT(R, NR, 4H R, 1)

CALL LNCNT(4)

PRINT 75

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

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

CALL LNCNT(3)

CALL EQUATE(P, NP, DUMMY, P)

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

CALL TRANP(B, NH, DUMMY(N2), NF)

CALL MULT(DUMMY(N2), NF, DUMMY(N1), NA, F, NF)

CALL MULT(P, NP, B, NR, DUMMY(N1), NR)

CALL MULT(DUMMY(N2), NF, DUMMY(N1), NB, DUMMY(N3), NR)

CALL ADD(R, NR, DUMMY(N3), NR, DUMMY(N1), NR)
IOP(2) = 3
IAC = IAM
MF = MF(N)
CALL SNVDEC(IOP, MF, MF, MF, DUMMY(N), NF, IAC, ZTEST, DUMMY(N))
IF( IERR .EQ. 0 ) GO TO 300
CALL LNCNT()
IF(IERR .EQ. 0) PRINT 200, IERR
IF(IERR .EQ. -1) PRINT 200, ZTEST
IF(IERR .EQ. 0) GO TO 400
IF(IOP(2) .EQ. 0 ) GO TO 400
CALL LNCNT()
PRINT 350, I
CALL PRNT(F, NF, NH F, 1)
CALL PRNT(P, NP, PH P, 1)
CALL MAXEL(DUMMY, NP, ANORM1)
CALL SUBT(DUMMY, NP, P, NP, DUMMY(N), NP)
CALL MAXEL(DUMMY(N), NP, ANORM2)
IF( ANORM1 .NE. 0.0 ) GO TO 500
GO TO 100
IF( K .EQ. 0 ) RETURN
K = IOP(1) + IOP(2)
IF( K .EQ. 0 ) RETURN
IF( KSS .EQ. 0 ) GO TO 700
CALL LNCNT()
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)-1
PRINT 800, I
800 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,NG,R,NNZ,W,LAMDA,F,PF,PNP)

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

DIMENSION A(1),B(1),H(1),Q(1),R(1),Z(1),W(1),LAMDA(1),S(1),F(1),PF(1),PNP(1)

DIMENSION NA(2),NB(2),NH(2),Q(2),R(2),NNZ(2),NP(2),IOP(3),NDUM1(2),NDUM2(2)

LOGICAL IDENT

COMMON/CONV/SUMCV,RICTCV,SCV,MAXSUM

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

CALL LNCNT(5)

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

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

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

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

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

CALL PRNT(Q,NG,4H Q , 1)

IF( .NOT. IDENT ) GO TO 45

CALL LNCNT(3)

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

GO TO 55

CALL PRNT(H,NH,4H H , 1)

CALL PRNT(R,NNZ,4H R , 1)

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

CALL LNCNT(4)

55 CONTINUE

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

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

IF( IERP , EQ, 0 ) GO TO 100

A-73
CALL LNCNT(4)
PRINT 75
CALL LNCNT(3)
PRINT 325
IF( IDENT .OR. (IOP(1) .NE. 0) ) GO TO 200
CALL SCALE(Q,NQ,Q,NQ,-1.0)
L = 2*N + 1
NDUM2(1) = 2*N+1
NDUM2(2) = NDUM2(1)
IF( IOP(1) .NE. 0 ) CALL PRNT(Z,NDUM2,4,M,Z,1)
CALL EIGEN(L,L,DUMMY(N1),NDUM1,DUMMY(N2),NDUM3,DUMMY(N3),ISV,ILV,W,DUMMY(N4),ICNTO)
IF( IERR .EQ. 0 ) GO TO 300
CALL LNCNT(4)
IF( IERR .GT. 0 ) GO TO 250
PRINT 225,IERR
225 FOR MAT(//' IN CNTREG, THE EIGENVALUE OF Z HAS NOT BEEN FOUND AFTER 30 ITERATIONS IN EIGEN'/)
RETURN
250 CONTINUE
275 FOR MAT(//' IN CNTREG, THE EIGENVALUES OF Z'*
RETURN
300 CONTINUE
IF( IOP(1) .EQ. 0 ) GO TO 400
CALL LNCNT(3)
PRINT 325
325 FOR MAT(//' EIGENVALUES OF Z'*
NDUM1(1) = L
NDUM1(2) = 2
CALL PRNT(DUMMY(N2),NDUM1,0,3)
CALL LNCNT(3)
PRINT 350
350 FOR MAT(//' CORRESPONDING EIGENVECTORS'*
CALL PRNT(W,NDUM2,0,3)
126 CONTINUE
127 CALL EQUATE(W,NDUM2,DUMMY(N1),NDUM2)
128 J1 = 1
129 J2 = 1
130 M = N
131 NDUM1(1) = L
132 NDUM1(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),NDUM1,W(J),NDUM1)
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),NDUM1,W(J),NDUM1)
159 CONTINUE
160 I = I+1
161 GO TO 415
162 CONTINUE
163 C
164 CALL NULL(LAMBDA,NA)
165 K0 = -1
166 J = -NA(1)
167 NAX = NA(1)
168 I = 1
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) .EQ. 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 = M + (I-1)*L + 1
186 K6 = K5 + L
187 CALL EQUATE(W(K5),NDUM1,DUMMY(N1),NDUM1)
188 CALL EQUATE(W(K6),NDUM1,W(K5),NDUM1)
CALL PPNT(DUMMY(NM22),NA,4NM22,1)

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 SUBST(NO,DUMMY(NW22),NA,S,NA)
CALL SUBST(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 925
825 FORMAT(/" IN CNTREG, GELIM HAS FOUND THE MATRIX W21 = P1xW11 TO BE SINGULAR")
RETURN
625 FOR FORMAT(/" MATRICES (A INVERSE)X(B TRANSPOSE)")
CALL PPNT(P,NP,FMT,1)
RETURN
1000 CONTINUE
N2 = N1+4+N
CALL TRAMP(DUMMY(NW12),NA,DUMMY(N2),NA)
CALL TRAMP(DUMMY(NW22),NA,P,NP)
N3 = N2+N
IFAC = 0
L = NA(1)
N4 = N3+NA(1)
CALL GELIM(L,L,DUMMY(N2),L,P,DUMMY(N3),IFAC,DUMMY(N4),IERR)
IF (IERR .EQ. 0 ) GO TO 950
CALL LNCNT(4)
PRINT 925
925 FORMAT(/" IN CNTREG, GELIM HAS FOUND THE MATRIX W12 TO BE SINGULAR")
RETURN
950 CONTINUE
N2 = N1+4+N
CALL PPNT(P,NP,FMT,1)
RETURN
900 CONTINUE
N2 = N1+4+N
CALL TRAMP(DUMMY(NW12),NA,DUMMY(N2),NA)
CALL TRAMP(DUMMY(NW22),NA,P,NP)
N3 = N2+N
IFAC = 0
L = NA(1)
N4 = N3+NA(1)
CALL GELIM(L,L,DUMMY(N2),L,P,DUMMY(N3),IFAC,DUMMY(N4),IERR)
IF (IERR .EQ. 0 ) GO TO 950
CALL LNCNT(4)
PRINT 925
925 FORMAT(/" IN CNTREG, GELIM HAS FOUND THE MATRIX W12 TO BE SINGULAR")
RETURN
300 NDUM1(1) = NR(1)
301 NDUM1(2) = NA(1)
302 CALL MULT(DUMMY,NDUM1,P,NP,F,NF)
303 IF (IOP(1) .EQ. 0 ) RETURN
304 CALL PPNT(P,NP,4FMT,1)
305 CALL PPNT(F,NP,4FMT,1)
306 RETURN
317 C
308 1000 CONTINUE
NMAX = T(1)/T(2)
I = NMAX
CALL EQUATE(LAMBDA,NA,DUMMY(N2),NA)
TT = -T(2)
 Nu = N3+N
NS = N4-N
L5, N6, N7 = NA+NA(1)
KSS = 0
NDUM1(1) = NA(1)
NDUM1(2) = NA(1)
CALL EXPSER(DUMMY(N2), NA, DUMMY(N3), NA, T, 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)
CONTINUE
IF ( NMAX .LE. 0 ) RETURN
CALL EQUATE(S, NA, DUMMY(N3), NA)
TIME = I*T(2)
IF ( I .NE. NMAX ) CALL EQUATE(DUMMY(N5), NA, P, NP)
CALL MULT(DUMMY(N3), NA, DUMMY(N2), NA, DUMMY(N4), NA)
CALL MULT(DUMMY(N2), NA, DUMMY(N4), NA, DUMMY(N3), NA)
CALL ADD(DUMMY(N12), NA, DUMMY(N4), NA, DUMMY(N3), NA)
CALL TRANP(DUMMY(N4), NA, DUMMY(N5), NA)
CALL EQUATE(DUMMY(N5), NA, DUMMY(N4), NA)
CALL MULT(DUMMY(NW21), NA, DUMMY(N5), NA, DUMMY(N3), NA)
CALL ADD(DUMMY(NW22), NA, DUMMY(N5), NA, DUMMY(N5), NA)
CALL TRANP(DUMMY(N5), NA, DUMMY(N6), NA)
CALL EQUATE(DUMMY(N6), NA, DUMMY(N5), NA)
L = NA(1)
IFAC = 0
CALL GELIM(L, L, DUMMY(N4), L, DUMMY(N5), DUMMY(N6), IFAC, DUMMY(N7), IERRCN)
1)
IF ( IERR .EQ. 0 ) GO TO 1200
CALL LNCNT(3)
PRINT 1150, TIME
IF ( I .NE. NMAX ) CALL MAXEL(P, NP, ANORM1)
CALL SUBT(DUMMY(N5), NA, P, NP, DUMMY(N4), NA)
CALL MAXEL(DUMMY(N4), NA, ANORM2)
GO TO 1300
GO TO 1225
ANORM1 .GT. 0.0 ) GO TO 1225
GO TO 1300
IF ( I .NE. NMAX ) CALL PRNT(F, NF, FH, F, 1)
GO TO 1400
IF ( IOP(2), EQ. 0 ) GO TO 1400
CALL LNCNT(5)
PRINT 1350, TIME
FORMAT(///" IN CNTREG AT TIME ", D16.8, " P CANNOT BE COMPUTED DUE TO
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)
GO TO 1300
IF ( ANORM1 .LE. 0.0 ) GO TO 1225
GO TO 1300
IF ( ANORM2/ANORM1 .LT. RICTCV ) KSS = 1
GO TO 1300
IF ( ANORM2 .LT. RICTCV ) KSS = 1
GO TO 1300
CALL PRNT(P, NP, 4H P, 1)
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

A-79
SUBROUTINE RICCATI(A, NA, B, NH, Q, NO, R, NR, F, NF, P, NP, IOP, IDENT, DI

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

DIMENSION A(1), B(1), Q(1), R(1), F(1), P(1), DUMMY(1)

DIMENSION NA(2), NB(2), NO(2), NR(2), NF(2), NP(2), IOP(3)

DIMENSION H(1), NH(2), IOPT(2)

LOGICAL IDENT, DISC, FNULL, SYM

COMMON/CONV/SUMCV, RICTCV, SERCV, MAXSUM

IF (IOP(1) .EQ. 0) GO TO 210

CALL LNCNT(4)

IF (.NOT. DISC) PRINT 100
IF (.NOT. DISC) PRINT 150

100 FORMAT(/** PROGRAM TO SOLVE CONTINUOUS STEADY-STATE RICCATI EQUATION, BY THE NEWTON ALGORITHM/**)

150 FORMAT(/** PROGRAM TO SOLVE DISCRETE STEADY-STATE RICCATI EQUATION, BY THE NEWTON ALGORITHM/**)

CALL PRNT(A, NA, 4H A, 1)
CALL PRNT(B, NB, 4H B, 1)
CALL PRNT(Q, NO, 4H Q, 1)
IF (.NOT. IDENT) GO TO 185
CALL LNCNT(3)
PRINT 190

180 FORMAT(/** H IS AN IDENTITY MATRIX**,)

GO TO 200

185 CONTINUE

CALL PRNT(H, NH, 4H H, 1)
CALL MULT(Q, NO, H, NH, DUMMY, NH)
CALL TRANP(H, NH, DUMMY(N2), NP)
CALL MULT(DUMMY(N2), NP, DUMMY, NH, Q, NO)
CALL LNCNT(3)
PRINT 195

195 FORMAT(/** MATRIX (H TRANPOSE)QH **,)

CALL PRNT(Q, NO, 4H MTQH, 1)

200 CONTINUE

CALL PRNT(R, NR, 4H R, 1)
IF (FNULL) GO TO 210
CALL LNCNT(3)
PRINT 205

205 FORMAT(/** INITIAL F MATRIX**,)

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

210 CONTINUE

IF ((IOP(1) .NE. 0) OR IDENT) GO TO 220

CALL MULT(Q, NO, H, NH, DUMMY, NH)
CALL TRANP(H, NH, DUMMY(N2), NP)
CALL MULT(DUMMY(N2), NP, DUMMY, NH, Q, NO)

220 CONTINUE

IF (DISC) GO TO 900

CALL TRANP(B, NB, P, NP)
CALL EQUATE(R,NR,DUMMY,NR)
CALL SYMPOS(NR(1),NR(1),DUMMY,NP(2),P,IOPT,IOPT,DET,ISCALE,DUMMY(NR)
11),IERR)
IF(IERR .EQ. 0) GO TO 250
CALL LNCNT(3)
PRINT 225
FORMAT(1/,"IN RNWT, A MATRIX WHICH IS NOT SYMMETRIC POSITIVE DERIC
FINITE HAS BEEN SUBMITTED TO SYMPOS")
RETURN

IF(NERR) GO TO 300
CALL EQUATE(P,NP,DUMMY,NF)
CALL MULT(B,NB,DUMMY,NF,DUMMY(N1),NA)
CALL TRANP(DUMMY(N1),NA,DUMMY(N2),NA)
CALL ADD(DUMMY(N1),NA,DUMMY(N2),NA,DUMMY(N1),NA)
CALL SCALE(DUMMY(N1),NA,DUMMY(N1),NA,0.5)

IF(IEP(3) .NE. 0) GO TO 400
EPSA = EPSAM
CALL BARSB(NB,NA,P,NP,IOPT,SCLE,SYM,DUMMY(N2))
GO TO 450

IF(TOP(2).EQ. 0) GO TO 550
CALL R(CNT(3)

PRINT 500, I
FORMAT(1/,"ITERATION ",I5)
CALL PRINT(P,NP,4=NP,1)

CALL SCALE(P,NP,0.5)
CALL ADD(Q,NP,P,NP,1.0)
CALL SCALE(P,NP,-1.0)
GO TO 350

350 CONTINUE
CALL MULT(DUMMY(N1),NA,DUMMY(N2),NA)
CALL TRANP(A,NA,DUMMY(N2),NA)
CALL SCALE(0.0,P,NP)
CALL SCALE(0.0,P,NP)
CALL ADD(P,NP,P,NP,1.0)
GO TO 350

300 CONTINUE
CALL TRANP(A,NA,DUMMY(N2),NA)
CALL SCALE(0.0,P,NP)
CALL SCALE(0.0,P,NP)
CALL ADD(P,NP,P,NP,1.0)
GO TO 350

400 CONTINUE

IOPT(2)=1
CALL BILIN(DUMMY(N2),NA,B,NB,P,NP,IOPT,SCLE,SYM,DUMMY(N3))

450 CONTINUE
CALL EQUATE(P,NP,DUMMY(N2),NP)
IF(IOPT(2).EQ. 0) GO TO 550
CALL LNCNT(3)
PRINT 500, I
FORMAT(1/,"ITERATION ",I5)
CALL PRINT(P,NP,4=NP,1)

550 CONTINUE
CALL MULT(DUMMY(N1),NA,P,NP,DUMMY(N3),NA)
CALL MULT(P,NP,DUMMY(N1),NA,DUMMY(N4),NA)
CALL TRANP(DUMMY(N4),NA,DUMMY(N5),NA)
CALL ADD(DUMMY(N4),NA,DUMMY(N3),NA,DUMMY(N3),NA)
CALL SCALE(DUMMY(N3),NA,DUMMY(N3),NA,0.5)
CALL ADD(DUMMY(N3),NA,0.0,P,NP)
CALL SCALE(P,NP,0.0)
CALL SCALE(P,NP,-1.0)
GO TO 350

500 CONTINUE
CALL MULT(DUMMY(N1),NA,P,NP,DUMMY(N3),NA)
CALL MULT(P,NP,DUMMY(N1),NA,DUMMY(N4),NA)
CALL TRANP(DUMMY(N4),NA,P,NA)
CALL ADD(P,NP,DUMMY(N4),NA,P,NP)
CALL SCALE(P,NP,0.0)
CALL ADD(Q,NP,P,NP,1.0)
CALL SCALE(P,NP,0.0)
A-81
CALL SURT(A, NA, DUMMY(N3), NA, DUMMY(N4), NA)

C

IF(IOP(3) .NE. 0 ) GO TO 650
CALL BARSTW(DUMMY(N3), NA, B, NP, P, IOPT, sym, EPSA, EPSA, DUMMY(N4))
GO TO 675

C

CONTINUE
CALL BLIN(DUMMY(N3), NA, B, NP, IOPT, SCLE, sym, DUMMY(N4))

I = I + 1
CALL MAXEL(DUMMY(N2), NA, ANORM1)
CALL SUBT(P, NP, DUMMY(N2), NA, DUMMY(N3), NA)
CALL MAXEL(DUMMY(N3), NA, ANORM2)

IF(ANORM1 .GT. 1.0) GO TO 700
IF(ANORM2/ANORM1 .LT. RICTCV) GO TO 800

GO TO 750

C

GO TO 700
IF(ANORM2 .LT. RICTCV) GO TO 800

C

GO TO 450
CALL LNCNT(3)
PRINT 775

C

CALL SCALE(DUMMY(N1), NA, DUMMY(N1), NA, DUMMY(N1), NA)
CALL A0D(A, NO, DUMMY(N1), NA, P, NP)

C

CALL MULT(R, NR, RF, NF, DUMMY(N1), NF)
CALL TRANP(F, NF, P, NP)
CALL MULT(P, NP, DUMMY(N1), NF, DUMMY(N2), NA)
CALL TRANP(DUMMY(N2), NA, DUMMY(N1), NA)
CALL ADD(DUMMY(N1), NA, DUMMY(N2), NA, DUMMY(N1), NA)
CALL SCALE(DUMMY(N1), NA, DUMMY(N2), NA, DUMMY(N1), NA)
CALL ADD(Q, NQ, DUMMY(N1), NA, P, NP)
CALL MULT(R, NR, RF, NF, DUMMY(N1), NA)
CALL SUBT(A, NA, DUMMY(N1), NA, DUMMY(N1), NA)
CALL TRANP(DUMMY(N1), NA, DUMMY(N2), NA)

C

GO TO 1000
CALL SUM(DUMMY(N2), NA, P, NP, DUMMY(N1), NA, IOPT, sym, DUMMY(N3))
IF(IOP(2) .EQ. 0) GO TO 1100
CALL LNCNT(3)

PRINT 500, I
189 CALL PRNT(P, NP, QH P , 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), NR, DUMMY(N1), NR, 0, 5)  
200 CALL ADD(P, NR, DUMMY(N1), NR, DUMMY(N1), NR)  
201 CALL SYMPOS(NR(1), NR(1), DUMMY(N1), NA(1), F, IOPT, IOPT, DET, ISCALE, DUMMY)  
202 CALL LNCNT(3)  
203 PRINT 225  
204 RETURN  
205 PRINT 225  
206 RETURN  
207 C  
208 1150 CONTINUE  
209 IF(I .EQ. 1) GO TO 925  
210 CALL MAXEL(DUMMY, NA, ANORM1)  
211 CALL SUBT(P, NP, DUMMY, NA, DUMMY(N1), NA)  
212 CALL MAXEL(DUMMY(N1), NA, ANORM2)  
213 IF(ANORM1 .GT. 1) GO TO 1200  
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 .LE. 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)  
230 250 1ERGE*, /)  
231 CALL PRNT(P, NP, QH P , 1)  
232 CALL PRNT(F, NF, QH F , 1)  
233 C  
234 RETURN  
235 END  

A-83
SUBROUTINE ASMREG(A, NA, B, NB, F, NF, P, NP, ID, DISC, N)

DIMENSION A(1), NA(1), R(1), F(1), P(1), DUMMY(1)

DIMENSION NA(2), NB(2), R(2), F(2), P(2), IOPT(5), IOPT(3)

IMPLICIT REAL (A-H, O-Z)

DIMENSION A(1), NA(1), R(1), F(1), P(1), DUMMY(1)

DIMENSION A(2), NA(2), R(2), F(2), P(2), IOPT(3)

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

N = NA(1) + 2

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, IOPT, DUMMY(N1))

GO TO 200

100 CONTINUE

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

200 CONTINUE

21 IF ( STABLE ) GO TO 500

22 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, IOPT, DUMMY(N1))

GO TO 300

300 CONTINUE

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

400 CONTINUE

41 IF ( STABLE ) GO TO 400

42 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, IOPT, DUMMY(N1))

GO TO 225

225 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 DSTAR(A, NA, B, NB, F, NF, SING, IOPT, DSCL, DUMMY)

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

GO TO 225

300 CONTINUE

IF ( STABLE ) GO TO 400

A-84
CALL LNCNT(5)
IF( DISC ) GO TO 330
PRINT 310, ALPHA
FORMAT(//*, C, 1,0A4, 3,0A4) IN ASMREG, CSTAB HAS FAILED TO FIND A STABILIZING GAIN
1 'MATRIX (F) RELATIVE TO ', ' ', ALPHA = ',016,8/
RETURN
CONTINUE
PRINT 340, ALPHA
RETURN
CONTINUE
RETURN
CONTINUE
RETURN
CONTINUE
RETURN
CALL RICNT(A, NA, F, NB, H, NM, Q, NG, R, NR, F, NF, P, NP, IOP, IDENT, DISC, FNU)
CONTINUE
CALL RICNT(A, NA, B, NB, H, NM, Q, NG, R, NR, F, NF, P, NP, IOP, IDENT, DISMY)
CONTINUE
IF( IOP(4) .EQ. 0 ) GO TO 1100
IF( DISC ) GO TO 800
CALL MULT(P, NP, R, NB, DUMMY, NB)
CALL MULT(DUMMY, NA, F, NP, DUMMY(N1), NP)
CALL ADD(DUMMY, NA, F, NP, DUMMY(N1), NP)
CALL SCALE(DUMMY, NA, F, NP, DUMMY(N1), NP, 0, 5)
CALL MULT(P, NA, F, DUMMY(N1), NA)
CALL ADD(DUMMY, NA, F, DUMMY(N1), NA)
CALL MULT(DUMMY(N1), NA, DUMMY(N1), NA)
CALL MULT(DUMMY(N1), NA, DUMMY(N3), NA)
CALL MULT(DUMMY(N1), NA, DUMMY(N1), NA)
CALL ADD(DUMMY, NA, DUMMY(N1), NA, DUMMY, NA)
CALL SINT(P, NP, DUMMY, NA, DUMMY, NA)
CONTINUE
CALL LNCNT(4)
PRINT 1000
1000 FORMAT(/ "RESIDUAL ERROR IN RICCATI EQUATION "/)
CALL PRNT(DUMMY, NP, XERROR, 1)
CONTINUE
N2 = N1 + NA(1)
N3 = N2 + NA(1)
ISV = 0
CALL EQUATE(P, NP, DUMMY, NP)
CALL EIGEN(NA(1), NA(1), DUMMY, DUMMY(N1), DUMMY(N2), ISV, ISV, V, DUMMY(NA(NASMO1)))
NEVL = NA(1)
IF( IERR .EQ. 0 ) GO TO 1300
NEVL = NA(1) - IERR
CALL LNCNT(4)
PRINT 1200, IERR
1200 FORMAT(/ IN ASMREG, THE I5, I5, I5, I5 EIGENVALUE OF P HAS NOT BEEN COMPUTED AFTER 30 ITERATIONS */)
CONTINUE
NDUM1(1) = NEVL
NDUM1(2) = 1
CALL EQUATE(DUMMY(N1), NDUM1, DUMMY, NDUM1)
N1 = NDUM1(1) + 1
CALL MULT(B, NB, F, NF, DUMMY(N1), NA)
CALL SURAT(A, NA, DUMMY(N1), NA, DUMMY(N1), NA)
N2 = N1 + N
CALL EQUATE(DUMMY(N1), NA, DUMMY(N2), NA)
N3 = N2 + N
N4 = N3 + NA(1)
N5 = N4 + NA(1)
CALL EIGEN(NA(1), NA(1), DUMMY(N2), DUMMY(N3), DUMMY(N4), ISV, ISV, V, DUMMY(NA(NASMO1)))
1MY(I5), IERR
NEVL = NA(1)
IF( IERR .EQ. 0 ) GO TO 1500
NEVL = NA(1) - IERR
CALL LNCNT(4)
PRINT 1400, IERR
1400 FORMAT(/ IN ASMREG, THE I5, I5, I5, I5 EIGENVALUE OF A-BF HAS NOT BEEN COMPUTED AFTER 30 ITERATIONS */)
CONTINUE
NDUM2(1) = NEVL
NDUM2(2) = 1
CALL JUXTC(DUMMY(N3), NDUM2, DUMMY(N4), NDUM2, DUMMY(N2), NDUM3)
IF( IOP(5) .EQ. 0 ) RETURN
CALL LNCNT(4)
PRINT 1600
1600 FORMAT(/ "EIGENVALUES OF P */)
CALL PRNT(DUMMY, NDUM1, XHEVLP, 1)
CALL LNCNT(4)
PRINT 1700
1700 FORMAT(/ "CLOSED-LOOP RESPONSE MATRIX A-BF */)
CALL PRNT(DUMMY(N1), NA, XHEA-BF, 1)
CALL LNCNT(3)
PRINT 1800
1800 FORMAT(/ "EIGENVALUES OF A-BF */)
CALL PRINT ('UMMY(NZ),NUM3,0,3)
RETURN
END
SUBROUTINE ASM08FIL(A,NA,G,NG,M,NH,Q,NQ,R,NR,F,NF,P,NP,IDNT,DISC,N AS
1 1EW,STAB,L,FNULL,ALPHA,IOPT,DUMMY)
2 IMPLICIT REAL*(A-H,O-Z) 43`1
3 DIMENSION A(1),G(1),M(1),N(1),R(1),F(1),P(1),DUMMY(1)
4 DIMENSION NA(2),NG(2),MH(2),NH(2),NR(2),NF(2),NP(2),IOPT(5),NDUM1(AS
5 12),IOPT(1)
6 LOGICAL IDENT,DISC,NEW,T,STAB,L,FNULL
7 IF( IOPT(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 OPTIMAAS
12 1L FILTER PROBLEM",/)
13 25 FORMAT(/," PROGRAM TO SOLVE THE CONTINUOUS INFINITE-DURATION OPTIAAS
14 MAL FILTER PROBLEM",/)
15 CALL PRNT(A,NA,4HM 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 40 CONTINUE
22 CALL PRNT(G,NG,4HG ,1)
23 45 CONTINUE
24 CALL PRNT(H,NH,4HM ,1)
25 CALL LNCNT(3)
26 PRINT 45
27 45 FORMAT(/,"INTENSITY MATRIX FOR COVARIANCE OF MEASUREMENT NOISE",/)
28 CALL PRNT(R,NR,4HR ,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,4HQ ,1)
37 C
38 100 CONTINUE
39 IOPT(1)=IOPT(2)
40 IOPT(2)=IOPT(3)
41 IOPT(3)=IOPT(4)
42 IOPT(4)=IOPT(5)
43 IOPT(5)=0
44 K = 0
45 C
46 200 CONTINUE
47 CALL TRNSP(A,NA,DUMMY,NA)
48 CALL EQUATE(DUMMY,NA,A,NA)
49 CALL TRNSP(H,NH,DUMMY,NDUM1)
50 CALL EQUATE(DUMMY,NDUM1,H,NH)
51 IF( IDENT ) GO TO 250
52 CALL TRNSP(G,NG,DUMMY,NDUM1)
53 CALL EQUATE(DUMMY,NDUM1,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,IDNT,DISC,NEW,T,ST AS
59 4ABLE,FNULL,ALPHA,IOPT,DUMMY)
60 C
61 N1=(NA(1)**2)+3*NA(1)+1
62 CALL TRNSP(F,NF,DUMMY(N1),NDUM1) A-88
CALL EQUATE(DUMMY(N1),NOMUX,F,NF)
IF(IOP(1).EQ.0) GO TO 200
IF(IDENT) GO TO 300
CALL LNCNT(3)
PRINT 55
CALL PRNT(0,NQ,4MGOQT,1)

300 CONTINUE
CALL LNCNT(3)
PRINT 325
325 FORMAT(//,' FILTER GAIN',//)
CALL PRNT(F,NF,4M F ,1)
CALL LNCNT(3)
PRINT 350
350 FORMAT(//,'STEADY-STATE VARIANCE MATRIX OF RECONSTRUCTION ERROR',//)
CALL PRNT(P,NP,4M P ,1)
NOMUX(1)=NP(1)
NOMUX(2)=1
CALL LNCNT(3)
PRINT 375
375 FORMAT(//,' EIGENVALUES OF P',//)
CALL PRNT(DUMMY,NOMUX,4MELP,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=FH,1)
N2 = N1 + N
CALL LNCNT(3)
PRINT 385
385 FORMAT(//,' EIGENVALUES OF A=FH MATRIX',//)
NOMUX(1) = NA(1)
NOMUX(2) = 2
CALL PRNT(DUMMY(N2),NOMUX,NOMUX,3)
GO TO 200
END
63 IF ( IOP(1) .EQ. 0 ) GO TO 600
64 CALL LNCNT(4)
65 PRINT 425
66 FORMAT(/, "CONTROL LAW U = -F( COL.(X,XM) ), F = (F11,F12)'/)
67 CALL LNCNT(3)
68 PRINT 459
69 FORMAT(/, "PART OF F MULTIPLYING X ",/)
70 CALL PRINT(F,NF,NM,F11,1)
71 IF( NOT. DISC .AND. IOP(2) .EQ. 0 ) GO TO 600
72 CALL PRINT(F,NP,NM,F11,1)
73 IF( IOP(2) .EQ. 0 ) GO TO 600
74 CALL LNCNT(2)
75 PRINT 475
76 475 FORMAT(/, "EIGENVALUES OF P11")
77 NDUM1(1) = NA(1)
78 NDUM1(2) = 1
79 CALL PRINT(DUMMY(N1),NDUM1,0,3)
80 N1 = N1 + NDUM1(1)
81 NDUM1(2) = NA(1)
82 CALL LNCNT(2)
83 PRINT 500
84 500 FORMAT(/, "PLANT CLOSED-LOOP RESPONSE MATRIX A = BF11")
85 CALL PRINT(DUMMY(N1),NDUM1,0,3)
86 CALL LNCNT(2)
87 PRINT 525
88 525 FORMAT(/, "EIGENVALUES OF CLOSED-LOOP RESPONSE MATRIX")
89 N1 = N1 + NDUM1(1)*NDUM1(2)
90 NDUM1(2) = 2
91 CALL PRINT(DUMMY(N1),NDUM1,0,3)
92 CALL LNCNT(2)
93 600 CONTINUE
94 NF(1) = NA(2)
95 NF(2) = NA(1)
96 CALL MULT(B,NB,F,NF,DUMMY,NA)
97 CALL SUST(A,NA,DUMMY,NA,0UM2)
98 IF( IOP(1) .EQ. 0 ) OR. IOP(2) .NE. 0 ) GO TO 700
99 CALL LNCNT(2)
100 PRINT 500
101 CALL PRINT(DUMMY,NA,0,3)
102 CALL LNCNT(2)
103 700 CONTINUE
104 N1 = NA(1)**2 + 1
105 CALL TRAPN(DUMMY,NA,DUMMY(N1),NA)
106 CALL EQUATE(DUMMY(N1),NA,DUMMY,NA)
107 NF(2) = NA(1) + NAM(1)
108 NP(2) = NF(2)
109 IF( NOT. DISC .AND. IOP(2) .EQ. 0 ) NP(2) = NAM(2)
110 IOPTT = 0
111 SYM = .FALSE.
112 CALL EQUATE( Q,NQ,DUMMY(N1),NDUM2)
113 IF( HIDENT ) GO TO 725
114 CALL MULT(Q,NQ,NM,NM,DUMMY(N1),NDUM2)
115 725 CONTINUE
116 IF( HIDENT ) GO TO 750
117 N2 = N1 + NQ(1)*NM(2)
118 CALL TRAPN(H,NH,DUMMY(N2),NDUM1)
119 N3 = N2 + NH(1)*NH(2)
120 CALL MULT(DUMMY(N2),NDUM1,DUMMY(N1),NH,NM,DUMMY(N3),NDUM2)
121 CALL EQUATE(DUMMY(N3),NDUM2,DUMMY(N1),NDUM2)
122 750 CONTINUE
123 N2 = NA(1)**2 + NA(1)*NM(2) + 1
124 N3 = NA(1)**2 + 1
125 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 BARSTW(DUMMY,NA,AM,NAM,P(N3),NDUM2,IOPTT,SYM,EPSA,EPSA,DUMMY)
I(N2))
GO TO 900
CALL SCALE(P(N3),NOUM2,P(N3),NOUM2,-1.0)
N8 = N2 +NAM(1)**2
CALL EQUATE(AM,NAM,DUMMY(N2),NAM)
CALL SUM(DUMMY,NA,P(N3),NDUM2,DUMMY(N2),NAM,IOPTT,SYM,DUMMY(N4))
GO TO 900
CALL SCALE(P(N3),NOUM2,P(N3),NOUM2,0.0)
CALL EQUATE(AM,NAM,66MMY(N2),NAM)
CALL SUMI(DUMMY,NA,P(N3),NDUM2,DUMMY(N2),NAM,IOPTT,SYM,DUMMY(N4))
GO TO 1100
CALL TRANP(B,N8,DUMMY,NDUM1)
CALL MULT(DUMMY,NDUM1,P(N3),NDUM2,F(N2),NDUM3)
IF (.NOT. DISC ) GO TO 1000
N1 = NB(1)**2 + 1
CALL MULT(DUMMY,NDUM1,F,NA,DUMMY(N1),NDUM2)
CALL MULT(DUMMY(N1),NDUM2,B,NB,DUMMY,NR)
CALL ADD(R,NR,DUMMY,NR,DUMMY,NDUM2)
GO TO 1100
CALL EQUATE(R,NR,DUMMY,NR)
CALL SCALE(R,NR,DUMY,NR,NOUM3,IOPTT)
CALL EQUATE(DUMMY,NR,NDUM1,NDUM3)
IF (.NOT. DISC ) GO TO 1300
CALL MULT(F(N2),NDUM3,AM,NAM,DUMMY,NDUM1)
CALL EQUATE(DUMMY,NDUM1,F(N2),NDUM1)
CALL SCALE(P(N3),NOUM1,P(N3),NOUM1,0.0)
CALL EQUATE(DUMMY,NR,NDUM1,NDUM3)
CALL SCALE(DUMMY(1),NOUM1,P(N3),NOUM1,0.0)
IF (.NOT. DISC ) GO TO 1300
CALL MULT(F(N2),NDUM3,4H12,1)
NDUM1(1) = NA(1)
NDUM1(2) = NAM(1)
CALL PRNT(P(N3),NDUM1,4H12,1)
RETURN
END
SUBROUTINE IMPMOF(A, NA, B, NB, NH, AM, NAM, BM, NBM, Q, NQ, R, NR, F, NF, P, N)

IMPLICIT REAL*(A-H,O-Z)
DIMENSION A(1), B(1), H(1), AM(1), BM(1), Q(1), R(1), F(1), P(1), DUMMY(1)
DIMENSION NA(2), NB(2), NH(2), NAM(2), NBM(2), NQ(2), NR(2), NF(2), 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
GO TO 100
75 CONTINUE
CALL LNCNT(3)
PRINT 125
FORMAT(/,'H IS AN IDENTITY MATRIX',/) IMPO
100 CONTINUE
CALL LNCNT(4)
PRINT 125
125 FORMAT(/,'MODEL DYNAMICS',/) IMPO
CALL PRNT(AM, NAM, NH, AM, 1)
CALL PRNT(BM, NBM, NH, BM, 1)
CALL LNCNT(4)
PRINT 150
32 FORMAT(/,'WEIGHTING MATRICES',/) IMPO
CALL PRNT(Q, NQ, NH, Q, 1)
CALL PRNT(R, NR, NH, R, 1)
300 CONTINUE
CALL SUBT(A, NA, AM, DUMMY, NA)
CALL SUBT(B, NR, BM, DUMMY(N1), NB)
GO TO 400
C
30 CONTINUE
CALL LNCNT(3)
PRINT 450
450 FORMAT(/,'MATRICES',/) IMPO
CALL PRNT(DUMMY(N1), NH, 0, 3)
CALL LNCNT(3)
PRINT 475
475 FORMAT(/,'MATRICES',/) IMPO
CALL PRNT(DUMMY(N1), NAM, 0, 3)
C
A-93
63 500 CONTINUE
64 N2 = N1 + N
65 N3 = N2 + N
66 N4 = N3 + N
67 CALL MULT(Q,NQ,DUMMY,NH,DUMMY(N2),NH)
68 CALL MULT(NQ,NO,DUMMY(N1),NBM,DUMMY(N3),NBM)
69 CALL TRANP(DUMMY,NH,DUMMY(N4),NDUM1)
70 CALL MULT(DUMMY(N4),NDUM1,DUMMY(N2),NH,DUMMY,NA)
71 CALL MULT(DUMMY(N4),NDUM1,DUMMY(N3),NBM,DUMMY(N2),NB)
72 CALL TRN(DUMMY(N1),NBM,DUMMY(N4),NDUM1)
73 CALL SCALE(DUMMY(N2),NB,DUMMY(N1),NB,2.0)
74 CALL MULT(DUMMY(N4),NDUM1,DUMMY(N3),NBM,DUMMY(N2),NB)
75 CALL ADD(DUMMY(N2),NR,NR,DUMMY(N2),NR)
76 IF(IOP(1),EQ,0 ) GO TO 600
77 CALL LNCNT(3)
78 PRINT 525
79 525 FORMAT(/,' MATRIX NA - AMM TRANSPOSEQ( NA - AMH')
80 CALL PRNT(DUMMY,NA,0,3)
81 CALL LNCNT(3)
82 PRINT 550
83 550 FORMAT(/,' MATRIX 2C HA - AMH TRANSPOSEQ( HB - BM')
84 CALL PRNT(DUMMY(N1),NB,0,3)
85 CALL LNCNT(3)
86 PRINT 575
87 575 FORMAT(/,' MATRIX ( HB - BM TRANSPOSEQ( HB - BM ) + R')
88 CALL PRNT(DUMMY(N2),NR,0,3)
89 C
90 600 CONTINUE
91 IOP(1)= 0
92 IOP(2)= 1
93 IOP(3)= 1
94 N5 = N4 + N
95 CALL EQUATE(A,NA,DUMMY(N3),NA)
96 CALL PREPIL(DUMMY(N1),NA,NA,NA,DUMMY,N1,NA,DUMMY(N2),NR,DISC)
97 IOMY(N4),NF,IOP,DUMMY(N5))
98 IF(IOP(1),EQ,0 ) GO TO 700
99 CALL LNCNT(3)
100 PRINT 625
101 625 FORMAT(/,' PREFILTER GAIN')
102 CALL PRNT(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 C
112 700 CONTINUE
113 CALL EQUATE(DUMMY(N4),NF,DUMMY(N1),NF)
114 C
115 IF(IOP(2),EQ,1000 ) RETURN
116 C
117 IOP(1) = IOP(2)
118 IOP(2) = IOP(3)
119 IOP(3) = IOP(4)
120 IOP(4) = 0
121 IOP(5) = 0
122 HIDENT = .TRUE.
123 CALL ASMREG(DUMMY(N3),NA,NR,NR,NR,NH,DUMMY,NA,DUMMY(N2),NR,F,NF,P,N
124 IP,HIDENT,DISC,NEAT,STABLE,FULL,ALPHA,IOP,DUMMY(N4))
125 IF(IOP(1),EQ,0 ) GO TO 800

A-94
CALL LNCNT(3)
PRINT 725
FORMAT(/ , "GAIN FROM ASMREG")
CALL PRNT(F,NF,0,3)
CALL LNCNT(3)
PRINT 750
FORMAT(/ , "SOLUTION OF ASSOCIATED STEADY-STATE RICCATI EQUATION")
CALL PRNT(P,NP,0,3)
CALL LNCNT(3)
PRINT 775
CALL LNCNT(3)
FORMAT(/ , "EIGENVALUES OF P")
CALL PRNT(DUMMY(N4),NOUM1,0,3)
CALL LNCNT(3)
FORMAT(/ , "EIGENVALUES OF P")
CALL PRNT(DUMMY(N4),NOUM1,0,3)
CALL LNCNT(3)
FORMAT(/ , "GAIN FOR MODEL-FOLLOWING CONTROL LAW, U = - F X , F =")
CALL ADD(F,NF,DUMMY(N1),NF,F,NF)
IF( IOP(1) .EQ. 0 ) RETURN
CALL LNCNT(4)
PRINT 825
FORMAT(/ , "GAIN FOR MODEL-FOLLOWING CONTROL LAW, U = - F X , F =")
CALL PRNT(F,NF,4H F ,1)
N6 = N4 + NA(1)
CALL PRNT(DUMMY(N6),NA,4H A-8F,1)
NOUM1(2) = 2
N6 = N6 + N
CALL LNCNT(3)
PRINT 850
FORMAT(/ , "EIGENVALUES OF A-8F")
CALL PRNT(DUMMY(N6),NOUM1,0,3)
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 (S, 101) (A(J), J = I, NLST, NR)
!

NA(1) = NR
NA(2) = NC
!
CALL LNCNNT(1)
!
WRITE (6, 916) NAM, NR, NC
!
FORMAT ("ERROR IN READ1

M1XX 'A', NA, 'HAS NA=", 216)
!
RETURN
!
END

SUBROUTINE BALANC(NM,N,A,LOW,HIGH,SCALE)
IMPLICIT REAL*8 (A-H,O-Z)
INTEGER I,J,K,L,M,N,JJ,NN,LOW,HIGH,IEXC
DIMENSION A(NM,N),SCALE(N)
REAL DABS
LOGICAL NOCONV

********** RADIX IS A MACHINE DEPENDENT PARAMETER SPECIFYING
THE BASE OF THE MACHINE FLOATING POINT REPRESENTATION.
RADIX = 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
DO 50 GO TO (80,130), IEXC
********** SEARCH FOR ROWS ISOLATING AN EIGENVALUE
AND PUSH THEM DOWN **********
80 IF (L .EQ. 1) GO TO 280
L = L - 1
100 DO 120 JJ = 1, L
J = L + 1 - JJ
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
IEXC = 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
DO 150 I = K, L
A -97
IF (I, EQ, J) GO TO 150
65 150 CONTINUE
66 C
67 M = K
68 IEY = 2
69 GO TO 20
70 170 CONTINUE
71 C ************ NOW BALANCE THE SUBMATRIX IN ROWS K TO L ************
72 DO 180 I = K, L
73 SCALE(I) = 1.000
74 C ************ ITERATIVE LOOP FOR NORM REDUCTION ************
75 190 NOCONV = .FALSE.
76 C
77 DO 270 I = K, L
78 C = 0.000
79 R = 0.000
80 C
81 DO 200 J = K, L
82 IF (J, EQ, I) GO TO 200
83 C = C + DABS(A(J, I))
84 R = R + DABS(A(I, J))
85 200 CONTINUE
86 C ************ GUARD AGAINST ZERO C OR R DUE TO UNDERFLOW ************
87 IF (C, EQ, 0.000, OR, R, EQ, 0.000) GO TO 270
88 G = R / RADIX
89 F = 1.000
90 S = C + R
91 210 IF (C, GE, G) GO TO 220
92 F = F * RADIX
93 C = C * 82
94 GO TO 210
95 220 G = R * RADIX
96 230 IF (C, LT, G) GO TO 240
97 F = F / RADIX
98 C = C / 82
99 GO TO 230
100 C ************ NOW BALANCE ************
101 240 IF ((C + R) / F, GE, 0.95, * S) GO TO 270
102 G = 1.000 / F
103 SCALE(I) = SCALE(I) * F
104 NOCONV = .TRUE.
105 C
106 DO 250 J = K, N
107 250 A(I, J) = A(I, J) * G
108 C
109 DO 260 J = 1, L
110 260 A(J, I) = A(J, I) * F
111 C
112 270 CONTINUE
113 C
114 IF (NOCONV) GO TO 190
115 C
116 280 LOW = K
117 IGH = L
118 RETURN
119 C ************ LAST CARD OF BALANC ************
120 END
SUBROUTINE ELMHES(NM,N,LOW,IGH,A,INT)

IMPLICIT REAL*8 (A-H,O-Z)
INTEGER I,J,M,N,LA,NM,IGH,KP1,LOW,MM1,MP1
DIMENSION A(NM,N)

C REAL X,Y
C REAL DABS

INTEGER INT(IGH)

LA = IGH - 1
KP1 = LOW + 1

IF (LA .LT. KP1) GO TO 200

DO 180 M = KP1, LA

MM1 = M - 1 - 100

X = 0.000

I = M

DO 170 J = M, IGH

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

X = A(J,MM1)

I = J

100 CONTINUE

180 CONTINUE

INT(M) = I

IF (I .EQ. M) GO TO 130

*********** INTERCHANGE ROWS AND COLUMNS OF A ***********

DO 110 J = MM1, N

Y = A(I,J)

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

A(M,J) = Y

110 CONTINUE

DO 120 J = 1, IGH

Y = A(J,I)

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

A(J,M) = Y

120 CONTINUE

*********** END INTERCHANGE ***********

130 IF (X .EQ. 0.000) GO TO 190

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

DO 150 J = 1, IGH

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

150 CONTINUE

160 CONTINUE

190 CONTINUE

200 RETURN

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

END

A-99
SUBROUTINE HQR(NM,N,LIN,IGH,H,MH,WJ,IEVR)

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

INTEGER I,J,K,L,M,N,EN,LL,MM,NA,NM,IGH,ITS,LIN,M2,ENM2,IEVR

DIMENSION H(NM,N),W(R(N),W(N))


REAL*8 OSORT,DABS,DSIGN

INTEGER MIND

LOGICAL NOTLA3

C REAL P,Q,R,S,T,W,X,Y,ZZ,NORM, MACHEP

C INTEGER MIND —

C -- 

C MORO

C -- 

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO

C --

C MORO
DO 140 MM = L, EN+2
   M = EN+2 - L - MM
   ZZ = H(M,M)
   R = X - ZZ
   S = Y - ZZ
   P = (R + S - W) / H(M+1,M) + H(M,M+1)
   Q = H(M+1,M+1) - ZZ - R - S
   R = H(M+2,M+1)
   S = DABS(P) + DABS(Q) + DABS(R)
   P = P / S
   Q = Q / S
   R = R / S
   IF (M .EQ. L) GO TO 150
   IF (DABS(H(M,M-1)) .LE. MACHEP + DABS(P)) GO TO 150
   X = (DABS(H(M-1,M-1)) + DABS(ZZ) + DABS(H(M+1,M+1)))
   GO TO 150
140 CONTINUE

150 MP2 = M + 2

DO 160 I = MP2, EN
   H(I, I-2) = 0.000
   IF (I .EQ. MP2) GO TO 160
   H(I, I-3) = 0.000
160 CONTINUE

********** DOUBLED OR STEP INVOLVING ROWS L TO EN AND COLUMNS M TO EN **********

DO 260 K = M, NA
   NOTLAS = K .NE. NA
   IF (K .EQ. M) GO TO 170
   IF (I .EQ. MP2) GO TO 260
   R = 0.000
   IF (NOTLAS) R = H(K+2,K-1)
260 CONTINUE

170 S = DSIGN(DSORT(P+P+Q+R+R),P)
180 IF (K .EQ. M) GO TO 180
   H(K,K-1) = -S * X
190 GO TO 190

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

DO 210 J = K, EN
   P = H(K,J) + R * H(K+1,J)
   IF (.NOT. NOTLAS) GO TO 210
   P = P + R * H(K+2,J)
210 H(K+2,J) = H(K+2,J) + P * ZZ
126 200  H(K+1,J) = H(K+1,J) - P * Y
127  H(K,J) = H(K,J) - P * X
128 210 CONTINUE
129.
130.J = MINO(EN,K+3)
131.
********** 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.
140.
141.260 CONTINUE
142.
143.GO TO 70
144.
********** ONE ROOT FOUND **********
145.270 WR(EN) = X + T
146.WI(EN) = 0.000
147.EN = NA
148.GO TO 60
149.
********** 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.
********** 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.
********** COMPLEX PAIR **********
164.320 WR(NA) = X + P
165.WR(EN) = X + P
166.WI(NA) = ZZ
167.WI(EN) = -ZZ
168.330 EN = ENM2
169.GO TO 60
170.
********** SET ERROR -- NO CONVERGENCE TO AN
171.
********** EIGENVALUE AFTER 30 ITERATIONS **********
172.1000 IERR = EN
173.1001 RETURN
174.
********** LAST CARD OF HQR **********
175.END

A-102
SUBROUTINE INVIT(NM,N,A,W,K,SELECT,MM,Z,KERR,RM1,RV1,RV2)
IMPLICIT REAL*8 (A-H,O-Z)
REAL*8 NORM,NORMV,ILAMBD,MACHEP
INTEGER I,J,K,L,M,N,S,II,IP,NM,MP,NM1,N1,UK,IP,ITS,KM1,KERR
DIMENSION A(NM,N),W(N),M(N),Z(NM,NM),RM1(N,N),RV1(N),RV2(N)
REAL T,W,X,Y,EPS3,NORM,NORMV,GROWTO,ILAMBD,MACHEP,RLAMBD,UKROOT
REAL*8 DSORT,CDABS,DAABS,DFLOAT
INTEGER IA83
LOGICAL*1 SELECT(N)
COMPLEX*16 Z3,DCOMPLX
REAL*8 OREAL,OIMAG
C INVO
_ 4 DIMENSION A(NM,N),WR(N),M(N,N);RVI(N),RV2(N) -INVO
_ 5 REAL*8 NORM,NORMV,ILAMBD,MACHEP,RLAMBD,UKROOT
_ 6 REAL*8 OREAL,OIMAG
C INVO
_ 7 INTEGER IA83 RNVO
_ 8 LOGICAL*1 SELECT( 14) INVOO
_ 9 C REAL T,W
_ 10 X,Y,EP33,NORM,NORMV
_ 11 C REAL*8 OREAL,OIMAG
_ 12 C INVO
_ 13 C INVO
_ 14 IERR = 0
_ 15 UK = 0.
_ 16 S = 1.
_ 17 ********** IP = 0, REAL EIGENVALUE
_ 18 C 1, FIRST OF CONJUGATE COMPLEX PAIR
_ 19 C -1, SECONO OF CONJUGATE COMPLEX PAIR **********
_ 20 C IP = 0
_ 21 N1 = N = 1.
_ 22 C DO 980 K = 1, N
_ 23 IF (W(K) .EQ. 0.000 .OR. IP .LT. 0) GO TO 100
_ 24 IF (SELECT(K) .AND. SELECT(K+1)) SELECT(K+1) = .FALSE.
_ 25 IF (.NOT. SELECT(K)) GO TO 980
_ 26 IF (W(K) .NE. 0.000) S = S + 1
_ 27 IF (S .GT. MM) GO TO 1000
_ 28 IF (UK .GE. K) GO TO 200
_ 29 C ********** CHECK FOR POSSIBLE SPLITTING **********
_ 30 IF (UK .EQ. N) GO TO 1140
_ 31 IF (UK .EQ. N) GO TO 100
_ 32 DO 120 UK = K, N
_ 33 IF (UK .EQ. N) GO TO 140
_ 34 IF (A(UK+1,UK) .EQ. 0.000) GO TO 140
_ 35 120 CONTINUE
_ 36 C ********** COMPUTE INFINITY NORM OF LEADING UK BY UK
_ 37 C (HESSENBERG) MATRIX **********
_ 38 140 NORM = 0.000
_ 39 MP = 1
_ 40 DO 180 I = 1, UK
_ 41 X = 0.000
_ 42 C CONTINUE
_ 43 DO 160 J = MP, UK
_ 44 X = X + DABS(A(I,J))
_ 45 160 CONTINUE
_ 46 IF (X .GT. NORM) NORM = X
_ 47 MP = I
_ 48 CONTINUE
_ 49 180 CONTINUE
_ 50 C ********** EPS3 REPLACES ZER0 PIVOT IN DECOMPOSITION
_ 51 C AND CLOSE ROOTS ARE MODIFIED BY EPS3**********
_ 52 IF (NORM .EQ. 0.000) NORM = 1.000
_ 53 EPS3 = MACHEP * NORM
_ 54 C ********** GROWTO IS THE CRITERION FOR THE GROWTH **********
_ 55 UKROOT = DSORT(DFLOAT(UK))
_ 56 GROWTO = 1.00-1 / UKROOT
_ 57 200 RLAMBD = W(K)
_ 58 ILAMBD = W(K)
_ 59 IF (K .EQ. 1) GO TO 280
_ 60 KM1 = K - 1
_ 61 GO TO 240
_ 62 C ********** PEPTURB EIGENVALUE IF IT IS CLOSE
TO ANY PREVIOUS EIGENVALUE **********

** FOR IK=1 STEP =1 UNTIL 1 DO -- **********

IF (SELECT(I),AND,ABS(WR(I)-RLAMBD), LT, EPS3, AND,

DABS(WI(I)-ILAMBD), LT, EPS3) GO TO 220

CONTINUE

WR(K) = RLAMBD

PERTURB CONJUGATE EIGENVALUE TO MATCH **********

IP1 = K + IP

WR(IP1) = RLAMBD

FORM UPPER HESSENBERG A-RLAMBD*I (TRANSPOSED)

AND INITIAL REAL VECTOR **********

DO 320 I = 1, UK

DO 300 J = MP, UK

RM1(J,I) = A(I,J)

RM1(I,I) = RM1(I,I) = RLAMBD

MP = I

RV1(I) = EPS3

CONTINUE

IF (ILA990 .NE. 0.000) GO TO 520

REAL EIGENVALUE. INVO

TRIANGULAR DECOMPOSITION WITH INTERCHANGES, INVO

REPLACING ZERO PIVOTS BY EPS3 **********

IF (UK .EQ. 1) GO TO 420

DO 340 J = MP, UK

RM1(J,I) = RM1(J,MP)

RM1(J,MP) = Y

CONTINUE

IF (RM1(MP,MP) .EQ. 0.000) RM1(MP,MP) = EPS3

X = RM1(MP,I) / RM1(MP,MP)

IF (X .EQ. 0.000) GO TO 400

DO 380 J = I, UK

RM1(J,I) = RM1(J,I) = X * RM1(J,MP)

CONTINUE

DO 400 I = UK, +1

DO 460 J = IP1, UK

ORGINAL PAGE IS OF POOR QUALITY
126 460 \text{Y} = Y - RM1(J,I) \times RV1(J)
127 \text{C}
128 480 RV1(I) = Y / RM1(I,I)
129 500 CONTINUE
130 \text{C}
131 \text{GO TO 740}
132 \text{C} \quad \text{********** COMPLEX EIGENVALUE.}
133 \text{C} \quad \text{TRIANGULAR DECOMPOSITION WITH INTERCHANGES.}
134 \text{C} \quad \text{REPLACING ZERO PIVOTS BY EPS3. STORE IMAGINARY}
135 \text{C} \quad \text{PART IN UPPER TRIANGLE STARTING AT (1,1) **********}
136 520 NS = N = S
137 \quad Z(1,S-1) = -ILAMBD
138 \quad Z(1,S) = 0.000
139 \quad \text{IF (N .EQ. 2) GO TO 550}
140 \quad RM1(1,1) = -ILAMBD
141 \quad Z(1,S-1) = 0.000
142 \quad \text{IF (N .EQ. 3) GO TO 550}
143 \text{C}
144 \text{DO 540 I = 4, N}
145 540 RM1(I,I) = 0.000
146 \text{C}
147 550 \text{DO 640 I = 2, UK}
148 \quad MP = I - 1
149 \quad W = RM1(MP,I)
150 \quad \text{IF (I .LT. N) T = RM1(MP,I+1)}
151 \quad \text{IF (I .EQ. N) T = Z(MP,S-1)}
152 \quad X = RM1(MP,MP) \times RM1(MP,MP) + T \times T
153 \quad \text{IF (W \times W .LE. X) GO TO 580}
154 \quad X = RM1(MP,MP) / W
155 \quad Y = T / W
156 \quad \text{RM1(MP,MP) = W}
157 \quad \text{IF (I .LT. N) RM1(MP,I+1) = 0.000}
158 \quad \text{IF (I .EQ. N) Z(MP,S-1) = 0.000}
159 \text{C}
160 \text{DO 560 J = I, UK}
161 \quad W = RM1(J,I)
162 \quad RM1(J,J) = RM1(J,MP) - X \times W
163 \quad RM1(J,MP) = W
164 \quad \text{IF (J .LT. N1) GO TO 555}
165 \quad L = J - NS
166 \quad Z(I,L) = Z(MP,L) - Y \times W
167 \quad Z(MP,L) = 0.000
168 \quad \text{GO TO 560}
169 555 \quad \text{RM1(I,J+2) = RM1(MP,J+2) - Y \times W}
170 \quad \text{RM1(MP,J+2) = 0.000}
171 560 \text{CONTINUE}
172 \text{C}
173 \quad \text{RM1(I,I) = RM1(I,I) - Y \times ILAMBD}
174 \quad \text{IF (I .LT. N1) GO TO 570}
175 \quad L = I - NS
176 \quad Z(MP,L) = -ILAMBD
177 \quad Z(I,L) = Z(I,L) + X \times ILAMBD
178 \quad \text{GO TO 640}
179 570 \quad RM1(MP,1+2) = -ILAMBD
180 \quad \text{RM1(I,I+2) = RM1(I,I+2) + X \times ILAMBD}
181 \quad \text{GO TO 640}
182 580 \quad \text{IF (X .NE. 0.000) GO TO 660}
183 \quad \text{RM1(MP,MP) = EPS3}
184 \quad \text{IF (I .LT. N) RM1(MP,I+1) = 0.000}
185 \quad \text{IF (I .EQ. N) Z(MP,S-1) = 0.000}
186 \quad T = 0.000
187 \quad \text{GO TO 640}
188 600 \quad Y = EPS3 \times EPS3
189 \quad \text{W = N / X}
190 \text{A-105}
X = RM1(MP, MP) * W
Y = -T * W

DO 620 J = I, UK
  IF (J .LT. N1) GO TO 610
  L = J - NS
  T = Z(MP, L)
  Z(I, L) = -X * T - Y * RM1(J, MP)
  GO TO 615
  RM1(I, J+2) = -X * T - Y * RM1(J, MP)
  RM1(J, I) = X * RM1(J, MP) + Y * T

CONTINUE

IF (I .LT. N1) GO TO 630

L = UK - NS
T = Z(UK, L)
GO TO 655

T = RM1(UK, UK+2)
IF (RM1(UK, UK) .EQ. 0.000 .AND. T .EQ. 0.000) RM1(UK, UK) = EPS3

******** BACK SUBSTITUTION FOR COMPLEX VECTOR
FOR I=UK STEP 1 UNTIL 1 00 ********

DO 720 I = 1, UK
  T = UK + 1 - I
  X = RV1(I)
  Y = 0.000
  IF (I .EQ. UK) GO TO 700
  IP1 = I + 1

DO 680 J = IP1, UK
  IF (J .LT. N1) GO TO 670
  L = J - NS
  T = Z(I, L)
  GO TO 675
  T = RM1(I, J+2)
  X = X - RM1(J, I) * RV1(J) + T * RV2(J)
  Y = Y - RM1(J, I) * RV2(J) - T * RV1(J)

CONTINUE

IF (I .EQ. UK) GO TO 710

L = I - NS
T = Z(I, L)
GO TO 715

T = RM1(I, I+2)
Z3 = DCMPLX(X, Y) / DCMPLX(RM1(I, I), T)
RV1(I) = DREAL(Z3)
RV2(I) = DIMAG(Z3)

CONTINUE

******** ACCEPTANCE TEST FOR REAL OR COMPLEX EIGENVECTOR AND NORMALIZATION ********

ITS = ITS + 1
NORM = 0.000
NORMV = 0.000

DO 780 I = 1, UK
  IF (ILAMBO .EQ. 0.000) X = DABS(RV1(I))
IF (ILAMBD .NE. 0.000) X = CDABS(DCMPLX(RV1(I),RV2(I)))
IF (NORMV .GE. X) GO TO 760
NORMV = X
J = I
NORM = NORM + X
CONTINUE

IF (NORM .LT. GROWTO) GO TO 840
ACCEPT VECTOR
Z(I,S) = RV1(I) * X
DO 860 I = 2, UK
RV1(I) = Y
J = UK - ITS + 1
RV1(J) = RV1(J) - EPS3 * X
IF (ILAMBD .EQ. 0.000) GO TO 440
GO TO 660
CONTINUE

A NEW STARTING VECTOR

IN-LINE PROCEDURE FOR CHOOSING

SET ERROR -- UNACCEPTED EIGENVECTORS

SET REMAINING VECTOR COMPONENTS TO ZERO

ORIGINAL PAGE IS OF POOR QUALITY
SUBROUTINE ELMBAK(NM,LOW,IGH,A,INT,M,Z)

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

INTEGER I,J,M,LA,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)

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

CONTINUE

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,NN,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

*** LEFT HAND EIGENVECTORS ARE BACK TRANSFORMED BY

DO 100 J = 1, M

Z(I,J) = Z(I,J) * S

100 CONTINUE

FOR I=LOW+1 STEP -1 UNTIL 1,

IGH+1 STEP 1 UNTIL N DO -- ********

120 DO 140 II = 1, N

I = II

IF (I.GE. LOW, AND, I.LE. IGH) GO TO 140

IF (I.LT. LOW) I = LOW - II

K = SCALE(I)

IF (K.EQ. I) GO TO 140

DO 130 J = 1, M

S = Z(I,J)

Z(I,J) = Z(K,J)

Z(K,J) = S

130 CONTINUE

140 CONTINUE

200 RETURN

*** LAST CARD OF BALBAK ********

END
SUBROUTINE DETFAC(NMAX,N,A,IPIVOT,IDET,DETERM,SCALE,WK,IERR)

IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION A(NMAX,1),IPIVOT(1),WK(1)

ISCALE=0
NM1=N-1
IERR=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
DO 60 I=1,N
P=0.0
DO 30 J=1,N
Q=MAXP(D,MABS(A(I,J)))
IF(Q.GT.P)P=Q
30 CONTINUE
IF(P.EQ.0)GO TO 40
40 DETERM=0.0
IERR=1
RETURN
60 WK(I)=P

PIVOTAL LOGIC SETUP
P=0.0
DO 110 I=M,N
Q=MABS(A(I,M))/WK(I)
IF(Q.P)110,110,100
100 P=Q
IP=I
110 CONTINUE
IPIVOT(M)=IP

IF(P.EQ.0)GO TO 40
IF(M.EQ.IP)GO TO 155

If the M-th row of the A matrix is chosen as the pivot row:
DO 150 I=1,N
P=A(IP,I)
A(IP,I)=A(M,I)
A(M,I)=P
150 CONTINUE

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)
180 CONTINUE
IPIVOT(N) = N
IF (A(N,N) .EQ. 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 == IPIVOT(I)) 240, 250, 240
240 SIGN = SIGN
250 CONTINUE
DO 340 I = 1, N
P = A(I, I)
260 CONTINUE
IF (R1 .GT. DABS(P)) GO TO 280
P = P*R2
ISCALE = ISCALE + 1
280 CONTINUE
IF (R2 .LT. DABS(P)) GO TO 290
P = P*R1
ISCALE = ISCALE - 1
290 CONTINUE
DETERM = DETERM * P
DETERM = DETERM * P
IF (R1 .GT. DABS(DETERM)) GO TO 320
DETERM = DETERM * R2
ISCALE = ISCALE + 1
320 CONTINUE
IF (R2 .LT. DABS(DETERM)) GO TO 340
DETERM = DETERM * R1
ISCALE = ISCALE - 1
340 CONTINUE
DETERM = DETERM * SIGN
RETURN
SUBROUTINE AXPXB(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 = M+1
MM1 = M-1
N1 = N+1
NM1 = N-1

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,NU,NA)

IF(NM1 .EQ. 0) GO TO 25

DO 20 I=1,NM1

20 CONTINUE

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

IF(FAIL .NE. 0) RETURN

DO 30 I=1,M

30 CONTINUE

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,NU)

IF(NM1 .EQ. 0) GO TO 45

DO 40 I=1,NM1

40 CONTINUE

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

FAIL = -FAIL

IF(FAIL .NE. 0) RETURN

C TRANSFORM C.

DO 50 J=1,N

50 CONTINUE

DO 60 I=1,M

60 CONTINUE

B(N1,J) = 0.

A(I,M1) = 0.

DO 70 K=1,M

70 CONTINUE

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

DO 80 I=1,M

80 CONTINUE

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

DO 90 J=1,N

90 CONTINUE

B(N1,J) = 0.
DO 70 K=1,N
   8(N1,J) = 8(N1,J) + C(I,K)*V(K,J)
70 CONTINUE

DO 80 J=1,N
   C(I,J) = 8(N1,J)
80 CONTINUE

C SOLVE THE TRANSFORMED SYSTEM.

CALL SHRSLV(A,B,C,M,N,NA,NB,NC)

C TRANSFORM C BACK TO THE SOLUTION.

DO 100 J=1,N
   DO 90 I=1,M
      A(I,M1) = 0.
   END DO
90 DO 90 K=1,M
   A(I,M1) = A(I,M1) + U(I,K)*C(K,J)
END DO

DO 100 I=1,M
   C(I,J) = A(I,M1)
END DO

DO 120 J=1,N
   B(N1,J) = 0.
120 DO 110 K=1,N
   B(N1,J) = B(N1,J) + C(I,K)*V(J,K)
110 CONTINUE

DO 120 J=1,N
   C(I,J) = 8(N1,J)
120 CONTINUE

RETURN
END
SUBROUTINE SHRSLV(A, R, C, M, N, NA, NB, NC)

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

DIMENSION IA(NA,1), IB(NB,1), C(NC,1)

INTEGER

LO, DL

COMMON/SLVBLV/T(S, 5), P(S), NSYS

L = 1

10 LM1 = L-1

DL = 1

IF(L .EQ. N) GO TO 15

IF(9(L+1, L) .NE. O.) DL = 2

15 LL = L+DL-1

IF(L .EQ. 1) GO TO 30

DO 20 J=1, LL

20 IF (LO .EQ. N) GO TO 10

CONTINUE

10 K = 1

IF (K .EQ. M) GO TO 45

IF(A(K, K+1) .NE. O.) DK = 2

45 KK = K+DK-1

IF(K .EQ. 1) GO TO 60

DO 50 I=K, LL

50 C(I, J) = C(I, J) + A(I, J)*C(J, J)

CONTINUE

60 IF(DL .EQ. 2) GO TO 80

IF(K .EQ. 2) GO TO 70

70 T(1, 1) = A(K, K) + B(L, L)

80 CONTINUE

GO TO 100

30 70 T(1, 1) = A(K, K) + B(L, L)

31 70 T(1, 2) = A(K, KK)

32 70 T(2, 1) = A(KK, K)

33 70 T(2, 2) = A(KK, KK) + B(L, L)

34 70 P(1) = C(K, L)

35 70 P(2) = C(KK, L)

36 70 NSYS = 2

37 70 CALL SYSSLV

38 70 C(K, L) = P(1)

39 70 C(KK, L) = P(2)

40 70 GO TO 100

41 80 IF(DK .EQ. 2) GO TO 90

42 80 T(1, 1) = A(K, K) + B(L, L)

43 80 T(1, 2) = A(K, KK)

44 80 T(2, 1) = A(KK, K)

45 80 T(2, 2) = A(KK, KK) + B(L, L)

46 80 P(1) = C(K, L)

47 80 P(2) = C(KK, L)

48 80 NSYS = 2

49 80 CALL SYSSLV

50 80 C(K, L) = P(1)

51 80 C(KK, L) = P(2)

52 80 GO TO 100

53 90 T(1, 1) = A(K, K) + B(L, L)

54 90 T(1, 2) = A(K, KK)

55 90 T(1, 3) = B(L, L)
IF (K .LE. M) GO TO 40

L = L + DL
IF (L .LE. N) GO TO 10
RETURN
END
SUBROUTINE ATXPXA(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 FAIL
2 N1 = N+1
3 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,N1,NA,NU,EPS,FAIL)
IF(Fail .NE. 0) RETURN

C TRANSFORM C.

DO 50 I = 1,N
   C(I,I) = C(I,I)/2.
50 CONTINUE
DO 60 J = 1,N
   A(I,N1) = 0.
60 CONTINUE
DO 70 K = 1,N
   A(I,N1) = A(I,N1) + U(K,I)*C(K,J)
70 CONTINUE
DO 80 I = 1,N
   C(I,J) = A(I,N1)
80 CONTINUE
DO 90 J = 1,N
   C(I,J) = C(I,J) + C(J,I)
90 CONTINUE
DO 100 K = 1,N
   C(K,J) = C(K,J) + C(J,K)
C SOLVE THE TRANSFORMED SYSTEM.
CALL SYMSLV(A,C,N,NA,NC)

C TRANSFORM C BACK TO THE SOLUTION.

DO 110 I = 1,N
   C(I,I) = C(I,I)/2.
110 CONTINUE
DO 120 J = 1,N
   A(I,N1) = 0.
120 CONTINUE
DO 130 K = 1,N
   A(I,N1) = A(I,N1) + U(J,K)*C(J,K)
63 90 CONTINUE
64 DO 100 J=1,N
65 C(I,J) = A(N1,J)
66 100 CONTINUE
67 DO 120 J=1,N
68 DO 110 I=1,N
69 A(I,N1) = 0.
70 DO 110 K=1,N
71 A(I,N1) = A(I,N1) + U(I,K)*C(K,J)
72 110 CONTINUE
73 DO 120 I=1,N
74 C(I,J) = A(I,N1)
75 120 CONTINUE
76 DO 130 I=1,N
77 DO 130 J=1,N
78 C(I,J) = C(I,J) + C(J,I)
79 C(J,I) = C(I,J)
80 130 CONTINUE
81 RETURN
82 END
SUBROUTINE SYMSLV(A,C,N,NA,NC)
IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION A(NA,1),C(NC,1)
INTEGER L
COMMON/SLVBLK/T(5,5),P(5),NSYS
L = 1

10 DL = 1
IF(L.EQ. N) GO TO 20
IF(A(L+1,L).NE. 0.) DL = 2
LL = L+DL+1
K = L

30 KM1 = K-1
DK = 1

15 IF(K.EQ. N) GO TO 35
16 IF(A(K+1,K).NE. 0.) DK = 2
17 KK = K+DK-1
18 IF(K.EQ. L) GO TO 45

DO 40 I=K, KK
    DO 40 J=LL,KM1

40 CONTINUE

C(I,J) = C(I,J) - A(IA,I)*C(IA,J)

23 CONTINUE
24 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 T(1,1) = A(K,K) + A(L,L)
31 T(1,2) = A(K,K)
32 T(2,1) = A(K,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 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,L)
48 NSYS = 2
49 CALL SYSSLV
50 C(K,L) = P(1)
51 C(K,L) = P(2)
52 GO TO 90
53 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
C(L,L) = P(1)
C(LL,L) = P(2)
C(LL,LL) = P(3)
CALL SYSSLV
P(1) = C(K,L)
P(2) = C(KK,L)
P(3) = C(K,LL)
P(4) = C(KK,LL)
NSYS = 4
CALL SYSSLV
C(K,L) = P(1)
C(KK,L) = P(2)
C(K,LL) = P(3)
C(KK,LL) = P(4)
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)
    CONTINUE
    DO 110 K=L,LL
        C(I,J) = C(I,J) - C(I,K)*A(K,J) = A(K,I)*C(K,J)
    CONTINUE
    C(J,I) = C(I,J)
CONTINUE
L = LDL
GO TO 10
END
SUBROUTINE HSHLDOR(ApN, 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

IF(NA .LT. L1) RETURN

DO 10 I=L1, N

10 MAX = DMAX1(MAX, DA83(A(I, L))

IF(MAX .NE. 0.) GO TO 20

A(L, N1) = 0.
A(N1, L) = 0.

GO TO 80

20 SUM = 0.

DO 30 I=L1, N

A(I, L) = A(I, L)/MAX

SUM = SUM + A(I, L)**2

30 CONTINUE

40 S = DSIGN(DSQR(3, SUM), A(L1, L))

A(L1, N1) = -MAX*S
A(L1, L) = S + A(L1, L)
A(N1, L) = S*A(L1, L)

DO 50 J=L1, N

SUM = 0.

DO 40 I=L1, N

SUM = SUM + A(I, L)*A(I, J)

40 CONTINUE

P = SUM/A(N1, L)

DO 50 I=L1, N

A(I, J) = A(I, J) - A(I, L)*P

50 CONTINUE

DO 70 I=1, N

SUM = 0.

DO 60 J=L1, N

SUM = SUM + A(I, J)*A(J, L)

60 CONTINUE

P = SUM/A(N1, L)

DO 70 J=L1, N

A(I, J) = A(I, J) - P*A(J, L)

70 CONTINUE

80 CONTINUE

A(N-1, N1) = A(N, N-1)

RETURN

END
SUBROUTINE BCKMLT(A,U,N,NA,NU)
IMPLICIT REAL*8 (A-H.O-Z)
DIMENSION 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.
20 SUM = SUM + A(I,L)*U(I,J)
CONTINUE
P = SUM/A(N1,L)
DO 25 I=L1,N
U(I,J) = U(I,J) - A(I,L)*P
CONTINUE
DO 30 I=L1,N
U(L,I) = 0.
30 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 1,FAIL,LOGICAL 1LAST

N = NN

HN = 0.

DO 20 I=1,N

JL = MAX0(1,I-1)

RSUM = 0.

DO 10 J=JL,N

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

10 CONTINUE

HN = OMAX1(HN,RSUM)

IF(HN .EQ. 0.) GO TO 230

IF(N .LE. 1) 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

S = X + Y

Y = X*Y + R

GO TO 90

50 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-Z) + Y + R*(H(M,M+1)/HN)

Q = R*(X+Z-S)

R = R*(H(M+2,M+1)/HN)

W = DABS(P) + DABS(Q) + DABS(R)

P = P/W

Q = Q/W

R = R/W

IF(M .EQ. L) GO TO 110

100 IF(DABS(H(M,M-1)) .LE. DABS(P)*TEST) 90

100 IGO TO 110
63  100 CONTINUE
64  110  M2 = M+2
65  120  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,NA
74  LAST = K.EQ.NA
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.) Y = -S
87  IF(K .NE. M) H(K,K-1) = -S*S
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  P = 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*Q
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(5,5), B(5), N
REAL*8 MAX
1 N = N - 1
N1 = N + 1

C COMPUTE THE LU FACTORIZATION OF A.

DO 80 K = 1, N
KM1 = K - 1
DO 10 I = K, N
DO 10 J = 1, KM1
A(I, K) = A(I, K) - A(I, J)*A(J, K)
10 CONTINUE

DO 20 K = 1, N
KM1 = K - 1
DO 20 J = 1, KM1
IF(K .EQ. 1) GO TO 50
DO 40 I = 1, N
A(K, J) = A(K, J) - A(K, I)*A(I, J)
40 CONTINUE
50 CONTINUE
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
30 CONTINUE

IF(INTR .EQ. K) GO TO 50
DO 40 J = 1, N
A(K, J) = A(K, J) - A(K, I)*A(I, J)
40 CONTINUE
50 CONTINUE

IF(K .EQ. 1) GO TO 70
DO 60 I = 1, N
A(K, J) = A(K, J) - A(K, I)*A(I, J)
60 CONTINUE
70 CONTINUE

IF(INTR .EQ. J) GO TO 110
DO 110 J = 1, N
INTR = A(N1, J)
IF(INTR .EQ. J) GO TO 110
DO 210 I = 1, N
B(I) = B(I) - A(I, J)*B(J)
210 CONTINUE
110 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

C SOLVE LX = B.

50 DO 220 I = 1, N
IM1 = I - 1
DO 220 J = 1, IM1
B(I) = B(I) - A(I, J)*B(J)
220 CONTINUE

220 CONTINUE

C 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)

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

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 - MAXIMUM ROW DIMENSION OF B

C N - ORDER OF A

C A(N,N) - INPUT MATRIX OF COEFFICIENTS (DESTROYED)

C NR - NUMBER OF COLUMNS IN B

C B(MAX,NR) - MATRIX OF CONSTANTS (REPLACED BY SOLUTIONS)

C IERR - INTEGER ERROR CODE

= 0 NORMAL RETURN

C = 2 INPUT MATRIX IS SINGULAR

C REQUIRED ROUTINES - NONE

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

C ****

DIMENSION A(N,N),B(MAX,NR)

NM1 = N+1

IF (NM1 .EQ. 0) GO TO 140

C **** FIND LARGEST REMAINING ELEMENT IN I-TH COLUMN FOR PIVOT

C ****

DO 100 I=1,NM1

BIG = 0.

DO 20 K=1,N

TERM = DBLE(A(K,I))

IF (TERM .GT. BIG) 20,20,10

10 BIG = TERM

L = K

CONTINUE

20 IF (BIG) 40,30,10

30 IERR = 2

RETURN

40 IF (I-L) 50,60,50

CONTINUE

C **** PIVOT ROWS OF A AND B

C ****

DO 50 J=1,N

TEMP = A(I,J)

A(I,J) = A(L,J)

A(L,J) = TEMP

CONTINUE

50 CONTINUE

C **** STORE PIVOT AND PERFORM COLUMN OPERATIONS ON A AND B

C ****

IP1 = I+1

DO 100 II=IP1,N

A(II,I) = A(II,I)/A(I,I)

X3 = A(II,I)

DO 90 K=IP1,N

A(II,K) = A(II,K) - X3*A(I,K)

CONTINUE

A=127
DO 100 K=1,NR
B(I+1,K) = B(I+1,K) - X3*B(I,K)
100 CONTINUE

C ****
PERFORM BACK SUBSTITUTION
C ****
DO 110 IC=1, NR
B(N,IC) = B(N,IC)/A(N,N)
110 CONTINUE

DO 130 KK=1, NM1
I = N-KK
IP1 = I+1
DO 130 J=1, NR
SUM = B(I,J)
130 CONTINUE

DO 120 K=IP1,N
SUM = SUM - A(I,K)*B(K,J)
120 CONTINUE
B(I,J) = SUM/A(I,I)

CONTINUE
RETURN

IF (A(1,1) .EQ. 0.0) GO TO 300
DO 150 J=1, NR
B(I,J) = B(I,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(NA(1)), IOP(4), NA(2)

IMPLICIT REAL*8 (A-M0-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 WRITE(7,150)

150 FORMAT(2X)

205 CONTINUE
IF (IOP(1).EQ.0) GO TO 210
WRITE(7,151) NAM, NR, NC

151 FORMAT(A4,/,2I5)

210 CONTINUE
DO 250 I=1, NR

IF (IOP(3).EQ.0) WRITE(7,152) (A(J), J=I, NMAX, NR)

152 FORMAT(6(I13,5))

IF (IOP(3).NE.0) WRITE(7,153) (A(J), J=I, NMAX, NR)

153 FORMAT(25X,6(I13,5))

250 CONTINUE

RETURN

END
FUNCTION DIMAG(Z)
REAL* 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
BLOCK DATA

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,EPSBN,IACM

COMMON/CONV/SUMCV,RICTCV,SERCV,MAXSUM

DATA LIN,NLP/1,58/

DATA NEPP,FMT1/7,8H(1P7D16.,8H7) /

DATA TIL/RM ORA,8HCLS PRO,8HGRAM /

DATA FMT2/8M(3X,1P7D,9H16.7) /

DATA EPSAM/1,E-10/

DATA EPSBN/1,E-10/

DATA IACM/12/

DATA SUMCV/1,E-8/

DATA RICTCV/1,E-8/

DATA SERCV/1,E-8/

DATA MAXSUM/50/

END