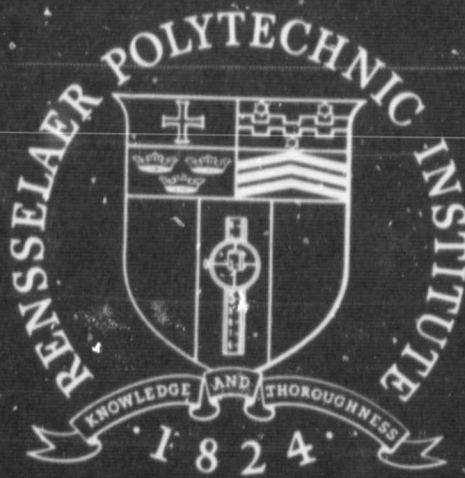


General Disclaimer

One or more of the Following Statements may affect this Document

- This document has been reproduced from the best copy furnished by the organizational source. It is being released in the interest of making available as much information as possible.
- This document may contain data, which exceeds the sheet parameters. It was furnished in this condition by the organizational source and is the best copy available.
- This document may contain tone-on-tone or color graphs, charts and/or pictures, which have been reproduced in black and white.
- This document is paginated as submitted by the original source.
- Portions of this document are not fully legible due to the historical nature of some of the material. However, it is the best reproduction available from the original submission.

T-71-01502



Rensselaer Polytechnic Institute
Troy, New York

FACILITY FORM 602

N71-33962

(ACCESSION NUMBER)

(THRU)

41

(PAGES)

(CODE)

CR-119913

(NASA CR OR TMX OR AD NUMBER)

19

(CATEGORY)

Rensselaer Polytechnic Institute
Troy, New York

Final Report Vol. III
Contract No. NAS8-21131
Covering Period Nov. 4, 1969 - April 4, 1971
NATIONAL AERONAUTICS AND
SPACE ADMINISTRATION

Improved Computational Methods for
Initial State Averaging

by

Derek E. McBrinn

Submitted on behalf of
Rob J. Roy
Professor
Systems Engineering Division

I. Introduction

A major obstacle to the implementation of optimal controllers has been the complexity of such devices. For the general problem the optimal controller is most readily determined in the form of required time histories of the system inputs. This contrasts with the classical output-feedback controller.

Optimal feedback controllers may be determined for the linear-quadratic state regulator problem. Even here, however, they may be difficult to mechanize since they require, in general, time-varying feedback of all system states. Efforts have recently been made^{1,2,3} to determine optimal time-invariant output-feedback controllers for such systems. In particular the finite-control-time problem was considered in [3]; the theory was developed and computational techniques suitable for low order systems were presented.

In this report new computational methods are developed to mechanize the theory presented in [3]. The increased computational efficiency associated with these new methods allows the application of the theory to systems of higher order. It also facilitates the computation of optimal piecewise-constant output-feedback controllers for time varying systems.

The techniques developed in this report are illustrated by a seven state model of a Saturn V booster rocket. An optimal controller is computed for this time-varying system over a portion of its flight.

II. Review of Theory

The following is a review of the theory developed in [3].

We are concerned with determination of the optimal time-invariant output-feedback controller for the linear system.

$$\dot{\underline{x}}(t) = A \underline{x}(t) + B \underline{u}(t) \quad (1)$$

$$\underline{y}(t) = C \underline{x}(t) \quad (2)$$

where $\underline{x}(t)$ is an NS-dimensional state vector

$\underline{u}(t)$ is an NC-dimensional input vector

$\underline{y}(t)$ is an NF-dimensional output vector.

The quadratic cost functional is

$$J = \underline{x}^T(T) F \underline{x}(T) + \int_0^T \underline{x}^T(\tau) Q \underline{x}(\tau) + \underline{u}^T(\tau) R \underline{u}(\tau) d\tau \quad (3)$$

with R , F and Q suitably positive (semi) definite. Thus we seek the gain matrix K which minimizes (3) where

$$\underline{u}(t) = -K^T \underline{y}(t) \quad (4)$$

The above problem cannot be solved for $NF < NS$ unless the state $\underline{x}(t)$ of the system is specified for some time instant. It has been shown however, that the expected value of J can be minimized if the probability distribution of $\underline{x}(0)$, the system initial state, is known. For the particular case where (a) the outputs $\underline{y}(t)$ comprise the first NF system states, and (b) the system initial state $\underline{x}(0)$ is uniformly distributed on the surface of the unit hypersphere in NS -space, minimization of the expected value of J is equivalent to minimization of the function GF defined by

$$GF = \frac{1}{NS} \operatorname{tr} \left\{ F\phi(T)\phi^T(T) + (Q+KRK^T) \int_0^T \phi(\tau)\phi^T(\tau) d\tau \right\} \quad (5)$$

where $\operatorname{tr} \{ \}$ denotes the matrix trace operation

$\phi(t)$ is the state transition matrix of the system

$$\dot{\underline{x}}(t) = (A - BK^T) \underline{x}(t) \quad (6)$$

The restrictions (a) and (b) above are not severe.

The necessary conditions for K to minimize (5) are defined by

$$W = 0 \quad (7)$$

where W is an $NF \times NC$ matrix whose coefficients are given by

$$\begin{aligned} w_{ij} = & \operatorname{tr} \left\{ F\phi \frac{\partial \phi^T}{\partial k_{ij}} \right|_T + (Q+KRK^T) \int_0^T \phi(\tau) \frac{\partial \phi^T(\tau)}{\partial k_{ij}} d\tau \\ & + \frac{\partial K}{\partial k_{ij}} RK^T \int_0^T \phi(\tau) \phi^T(\tau) d\tau \right\} \end{aligned} \quad (8)$$

The gradient of W with respect to the variable feedback gains is given by

$$\begin{aligned} \frac{\partial w_{gh}}{\partial k_{pq}} = & \operatorname{tr} \left\{ F \left[\frac{\partial \phi}{\partial k_{pq}} \frac{\partial \phi^T}{\partial k_{gh}} + \phi \frac{\partial^2 \phi}{\partial k_{pq} \partial k_{gh}} \right] \right|_T \\ & + (Q+KRK^T) \int_0^T \frac{\partial \phi}{\partial k_{pq}} \frac{\partial \phi^T}{\partial k_{gh}} + \phi \frac{\partial^2 \phi^T}{\partial k_{pq} \partial k_{gh}} d\tau \\ & + \frac{\partial K}{\partial k_{pq}} RK^T \int_0^T \frac{\partial \phi}{\partial k_{gh}} \phi^T + \phi \frac{\partial \phi^T}{\partial k_{gh}} d\tau \end{aligned}$$

$$\begin{aligned}
& + \frac{\partial K}{\partial k_{gh}} \text{RK}^T \int_0^T \frac{\partial \phi}{\partial k_{pq}} \phi^T + \frac{\partial \phi^T}{\partial k_{pq}} \phi \, d\tau \\
& + \frac{\partial K}{\partial k_{gh}} \text{R} \frac{\partial K^T}{\partial k_{pq}} \int_0^T \phi \phi^T \, d\tau \} \quad (9)
\end{aligned}$$

The above equations provide the basis of a Newton-Raphson iterative method for finding the optimal K . The appropriate relationship is

$$K_{n+1} = K_n - \mathcal{N}_n \nabla_n^{-1} \underline{w}_n \quad (10)$$

where the suffix denotes the iteration number

$\underline{w}_n$ is a vector arrangement of the matrix w of necessary conditions

∇ is a suitable matrix arrangement of the gradient coefficients defined by (9)

$\mathcal{N}$ is a convergence factor.

III. Computational Algorithm

A method is described below for digital computer mechanization of the Newton-Raphson iterative scheme defined in Section II.

It can be seen from (5)-(10) that the computations required involve products and integrals of $\phi(t)$, $\frac{\partial \phi(t)}{\partial k_{gh}}$ and $\frac{\partial^2 \phi(t)}{\partial k_{gh} \partial k_{pq}}$. It was shown in [3] that

$$\frac{\partial \phi(t)}{\partial k_{gh}} = - \int_0^t \phi(t-\tau) B \frac{\partial K^T}{\partial k_{gh}} \phi(\tau) d\tau \quad (11)$$

$$\frac{\partial^2 \phi(t)}{\partial k_{gh} \partial k_{pq}} = - \int_0^t \phi(t-\tau) B \left[\frac{\partial K^T}{\partial k_{gh}} \frac{\partial \phi(\tau)}{\partial k_{pq}} + \frac{\partial K^T}{\partial k_{pq}} \frac{\partial \phi(\tau)}{\partial k_{gh}} \right] d\tau \quad (12)$$

Assuming the eigenvalues of $[A-BK^T]$ to be distinct

$$\phi(t) = M e^{\Lambda t} M^{-1} \quad (13)$$

where Λ is the diagonal matrix of eigenvalues of $[A-BK^T]$.

M is a corresponding modal matrix of eigenvectors.

Equation (13) is used to compute $\phi(t)$ at time instants $T/32, T/16, T/8, 2T/8, \dots, T$. Using these values, $\frac{\partial \phi(t)}{\partial k_{gh}}$ can be approximated at times $T/16, T/8, 2T/8, \dots, T$ by

$$\frac{\partial \phi}{\partial k_{gh}} (T/16) \approx - \phi(T/32) B \frac{\partial K^T}{\partial k_{gh}} \phi(T/32) T/16 \quad (14)$$

$$\frac{\partial \phi}{\partial k_{gh}} (T/8) \approx - \phi(T/16) B \frac{\partial K^T}{\partial k_{gh}} \phi(T/16) T/8 \quad (15)$$

$$\begin{aligned}
\frac{\partial \phi}{\partial k_{gh}} (2T/8) &\approx \phi(3T/16) B \frac{\partial K^T}{\partial k_{gh}} \phi(T/16) T/8 \\
&+ \phi(T/16) B \frac{\partial K^T}{\partial k_{gh}} \phi(3T/16) T/8 \\
&= \phi(T/8) \frac{\partial \phi}{\partial k_{gh}} (T/8) + \frac{\partial \phi}{\partial k_{gh}} (T/8) \phi(T/8) \quad (16)
\end{aligned}$$

.

$$\frac{\partial \phi}{\partial k_{gh}} \left(\frac{nT+T}{8} \right) \approx \phi(T/8) \frac{\partial \phi}{\partial k_{gh}} \left(\frac{nT}{8} \right) + \frac{\partial \phi}{\partial k_{gh}} \left(\frac{nT}{8} \right) \phi(T/8) \quad (17)$$

In a similar manner $\frac{\partial^2 \phi(t)}{\partial k_{gh} \partial k_{pq}}$ is approximated at times $nT/8$, $n=1, \dots, 8$ by

$$\begin{aligned}
\frac{\partial^2 \phi}{\partial k_{gh} \partial k_{pq}} (T/8) &\approx - \phi(T/16) B \left[\frac{\partial K^T}{\partial k_{gh}} \frac{\partial \phi}{\partial k_{pq}} (T/16) + \frac{\partial K^T}{\partial k_{pq}} \frac{\partial \phi}{\partial k_{gh}} (T/16) \right] T/8 \\
&\triangleq D1MFE + D2MFE \quad (18)
\end{aligned}$$

$$\begin{aligned}
\frac{\partial^2 \phi}{\partial k_{gh} \partial k_{pq}} \left(\frac{nT+T}{8} \right) &\approx \phi(T/8) \frac{\partial^2 \phi}{\partial k_{gh} \partial k_{pq}} \left(\frac{nT}{8} \right) \\
&+ D1MFE \frac{\partial \phi}{\partial k_{pq}} \left(\frac{nT}{8} \right) + D2MFE \frac{\partial \phi}{\partial k_{gh}} \left(\frac{nT}{8} \right) \quad (19)
\end{aligned}$$

The above computations, plus the fact that $\phi(0)$ is the identity matrix and both $\frac{\partial \phi}{\partial k_{gh}}(0)$ and $\frac{\partial^2 \phi}{\partial k_{gh} \partial k_{pq}}(0)$ are null matrices, allows the computation of the various terms in (5)-(9) by Runge-Kutta numerical integration. The remaining computations required for (10)

are routine matrix operations, except for assignment of a value to the convergence factor N .

The optimal value of the convergence factor N is determined iteratively. It is first set to 1. If GF is reduced N is doubled, otherwise N is halved. This process continues until the optimal value of N is straddled. A quadratic interpolation technique then iterates to the optimal N . Note that this "best step" procedure requires only the relatively simple computation of the function GF at each step.

A Fortran IV program listing of the algorithm described above comprises Appendix I of this report.

IV. Results and Discussion

Notable computational improvements have resulted from the use of the algorithm described in Section III as compared to that used in [3]. The improvements increase with the dimensionality of the system. Some representative comparisons of computation times are shown in Table I.

Systems Considered	2 state 1 control 1 feedback	2 state 1 control 2 feedbacks	3 state 1 control 1 feedback
Ref 3 algorithm	69 sec.	193 sec.	976 sec.
Present algorithm	8 sec.	82 sec.	17 sec.

Table 1 Comparison of WATFIV Computation Times
for Old and New Algorithms

The techniques described in this report are illustrated here by a 7 state model of a Saturn V booster rocket. It is supposed that the objective is to produce piecewise constant output feedback gains for the time varying system.

Some preliminary notes are called for. In optimal control theory, if the control interval is long compared to the time constants of the dominant system modes it is most convenient to consider the control interval to be semi-infinite. For this reason the numerical integration techniques described in Section III were designed for control intervals not longer than about five time constants. If it is desired to consider longer control intervals it will be necessary to increase the number of points used in the numerical integration. This is a trivial modification.

The data for the illustrative example represent the Saturn V booster at a time of 80 seconds after lift-off. The control interval is chosen to be 5 seconds; say from lift-off plus 77.5 seconds to lift-off plus 82.5 seconds. The booster characteristics are adequately represented over this time interval by the data for the 80 second point.

The system matrix is

$$A = \begin{bmatrix} 0. & 1. & 0. & 0. & 0. & 0. & 0. \\ 0. & 0. & .203 & -.6535 & -.0020 & 2.558 & 0. \\ -.0137 & 1. & -.0407 & .0002 & -.0146 & -.0334 & 0. \\ 0. & 0. & 0. & 0. & 1. & 0. & 0. \\ 0. & 0. & 0. & -44.67 & -.1337 & 254.6 & 0. \\ 0. & 0. & 0. & 0. & 0. & 0. & 1. \\ 0. & 0. & 0. & 0. & 0. & -50. & -10. \end{bmatrix} \quad (21)$$

and the control matrix is

$$\underline{b} = \begin{bmatrix} 0. \\ 0. \\ 0. \\ 0. \\ 0. \\ 0. \\ 1. \end{bmatrix} \quad (22)$$

The first two states are available for feedback to the single controller. The cost matrices F and Q are chosen to be identity matrices of the appropriate order, and R is equal to .1. The first few pages of program output for this problem comprise Figure 1. The solution achieved is plotted in Figure 2, which shows that the program did indeed attain a minimum expected value of the cost.

STATES 7 CONTROLS 1 FEEDBACKS 2 LGAINS 1

SYSTEM MATRIX A

0.0000000	1.0000000	0.0000000	0.0000000
0.0000000	0.0000000	0.0000000	0.0000000
0.0000000	0.2030470	-0.6534950	-0.0019550
2.5580170	0.0000000	-0.0136615	1.0000000
-0.0406825	0.0000000	-0.0146300	-0.0333820
0.0000000	0.0000000	0.0000000	0.0000000
0.0000000	1.0000000	0.0000000	0.0000000
0.0000000	0.0000000	0.0000000	-44.6681000
-0.1336680	254.6100000	0.0000000	0.0000000
0.0000000	0.0000000	0.0000000	0.0000000
0.0000000	1.0000000	0.0000000	0.0000000
0.0000000	0.0000000	0.0000000	-50.0000000
-10.0000000			

CONTROL MATRIX B

0.0000000	0.0000000	0.0000000	0.0000000
0.0000000	0.0000000	1.0000000	

TERMINAL TIME T = 5.0000000

TERMINAL COST MATRIX F

1.0000000	0.0000000	0.0000000	0.0000000
0.0000000	0.0000000	0.0000000	0.0000000
1.0000000	0.0000000	0.0000000	0.0000000
0.0000000	0.0000000	0.0000000	0.0000000
1.0000000	0.0000000	0.0000000	0.0000000
0.0000000	0.0000000	0.0000000	0.0000000
1.0000000	0.0000000	0.0000000	0.0000000
0.0000000	0.0000000	0.0000000	0.0000000
1.0000000	0.0000000	0.0000000	0.0000000
0.0000000	0.0000000	0.0000000	0.0000000
1.0000000	0.0000000	0.0000000	0.0000000
0.0000000	0.0000000	0.0000000	0.0000000
1.0000000			

STATE WEIGHTING MATRIX Q

1.0000000	0.0000000	0.0000000	0.0000000
0.0000000	0.0000000	0.0000000	0.0000000
1.0000000	0.0000000	0.0000000	0.0000000
0.0000000	0.0000000	0.0000000	0.0000000
1.0000000	0.0000000	0.0000000	0.0000000
0.0000000	0.0000000	0.0000000	0.0000000
1.0000000	0.0000000	0.0000000	0.0000000
0.0000000	0.0000000	0.0000000	0.0000000
1.0000000	0.0000000	0.0000000	0.0000000
0.0000000	0.0000000	0.0000000	0.0000000
1.0000000	0.0000000	0.0000000	0.0000000
0.0000000	0.0000000	0.0000000	0.0000000
1.0000000			

CONTROL WEIGHTING MATRIX R

0.1000000

Figure 1 Program Output for Example

ITERATION NUMBER

1

GAIN MATRIX

-15.6600000

-16.0700000

SYSTEM EIGENVALUES

-0.48447150 01	0.54454500 01
-0.48447150 01	-0.54454590 01
-0.45365630-01	0.61767350 01
-0.45365630-01	-0.61767360 01
-0.13207580 00	0.38655710 00
-0.13207580 00	-0.38655710 00
-0.13003720 00	-0.00000000 00

AVERAGE COST =

0.92222550 03

NECESSARY CONDITIONS VECTOR

0.25544030 02 -0.12713730 03

GRADIENT MATRIX

0.14878930 02	0.11332240 01	0.11332240 01	0.51220860 02
---------------	---------------	---------------	---------------

INVERSE GRADIENT MATRIX

0.67322600-01	-0.14894630-02	-0.14894630-02	0.19556250-01
---------------	----------------	----------------	---------------

Figure 1 (continued)

PRECEDING PAGE BLANK NOT FILMED

ITERATION NUMBER

2

NEW GAINS

-17.5690567

-14.0456240

SYSTEM EIGENVALUES

-0.77709120-01 0.62524660 01

-0.77709120-01 -0.62524660 01

-0.48514500 01 0.53580730 01

-0.48514500 01 -0.53580730 01

-0.10474410-00 0.46073590 00

-0.10474410 00 -0.46073590 00

-0.10654470 00 -0.00000000 00

AVERAGE COST GF2 =

0.65949820 03

STEP SIZE IS DOUBLED, NEW GAINS ARE

-0.19478110-02 -0.11921250 02

SYSTEM EIGENVALUES

-0.10617490 00 0.63291500 01

-0.10617490 00 -0.63291500 01

-0.43617540-01 0.52711080 01

-0.43617540-01 -0.52711080 01

-0.72371860-01 0.51976800 00

-0.72371860-01 -0.51976800 00

-0.93748240-01 -0.00000000 00

AVERAGE COST GF2 =

0.02911980 03

STEP SIZE IS DOUBLED, NEW GAINS ARE

-0.23296230 02 -0.64721960 01

SYSTEM EIGENVALUES

-0.15156790 00 0.64830040 01

-0.15156790 00 -0.64830040 01

-0.48925430 01 0.51000490 01

-0.48925430 01 -0.51000490 01

-0.31482650-02 0.61001830 00

-0.31482650-02 -0.61001830 00

-0.79832210-01 -0.00000000 00

AVERAGE COST GF2 =

0.78329980 03

STEP SIZE IS DOUBLED, NEW GAINS ARE

-0.30932450 02 0.36250080 01

SYSTEM EIGENVALUES

-0.20079170 00 0.67852840 01

-0.20079170 00 -0.67852840 01

Figure 1 (continued)

-0.49879390 01	0.47777350 01
-0.49879390 01	-0.47777360 01
0.13530550 00	0.72554250 00
0.13530550 00	-0.72554250 00
-0.67501210-01	-0.00000000 00

AVERAGE COST GF2 = C.35975060 03

STEP SIZE INTERPOLATION. NEW GAINS ARE
-0.24509520 02 -0.4863440 01

SYSTEM EIGENVALUES

-0.16289030 00	0.65323880 01
-0.16289030 00	-0.65323880 01
-0.49049540 01	0.50463560 01
-0.49049540 01	-0.50463560 01
0.19166540-01	0.63315450 00
0.19166540-01	-0.63315450 00
-0.76994900-01	-0.00000000 00

AVERAGE COST GF2 = C.77171050 03

STEP SIZE INTERPOLATION. NEW GAINS ARE
-0.25470310 02 -0.35976750 01

SYSTEM EIGENVALUES

-0.17085040 00	0.65709280 01
-0.17085040 00	-0.65709280 01
-0.49156020 01	0.50052200 01
-0.49156020 01	-0.50052200 01
0.36811690-01	0.64597800 00
0.36811690-01	-0.64597800 00
-0.75069400-01	-0.00000000 00

AVERAGE COST GF2 = C.76603850 03

STEP SIZE INTERPOLATION. NEW GAINS ARE
-0.25470310 02 -0.35976750 01

ABOVE GAINS ARE BEST STEP FOR THIS ITERATION

GAIN TOLERANCE ACHIEVED = 3.0057528

REQUIRED STOPPING TOLERANCE = 0.0500000

SYSTEM EIGENVALUES

-0.17085040 00	0.65709280 01
-0.17085040 00	-0.65709280 01
-0.49156020 01	0.50052200 01

Figure 1 (continued)

-0.49156020 01 -0.59052200 01
 0.36811690-01 0.64597600 00
 0.36811690-01 -0.64597600 00
 -0.75069400-01 -0.00000000 00

AVERAGE COST -0.76003850 03

NECESSARY CONDITIONS VECTOR

-0.25793130 02 -0.33453640 02

GRADIENT MATRIX

0.16879540 02 0.10940680 01 0.10946680 01 0.19061610 02

INVERSE GRADIENT MATRIX

0.59464790-01 -0.34149380-02 -0.34149380-02 -0.52637570-01

Figure 1 (continued)

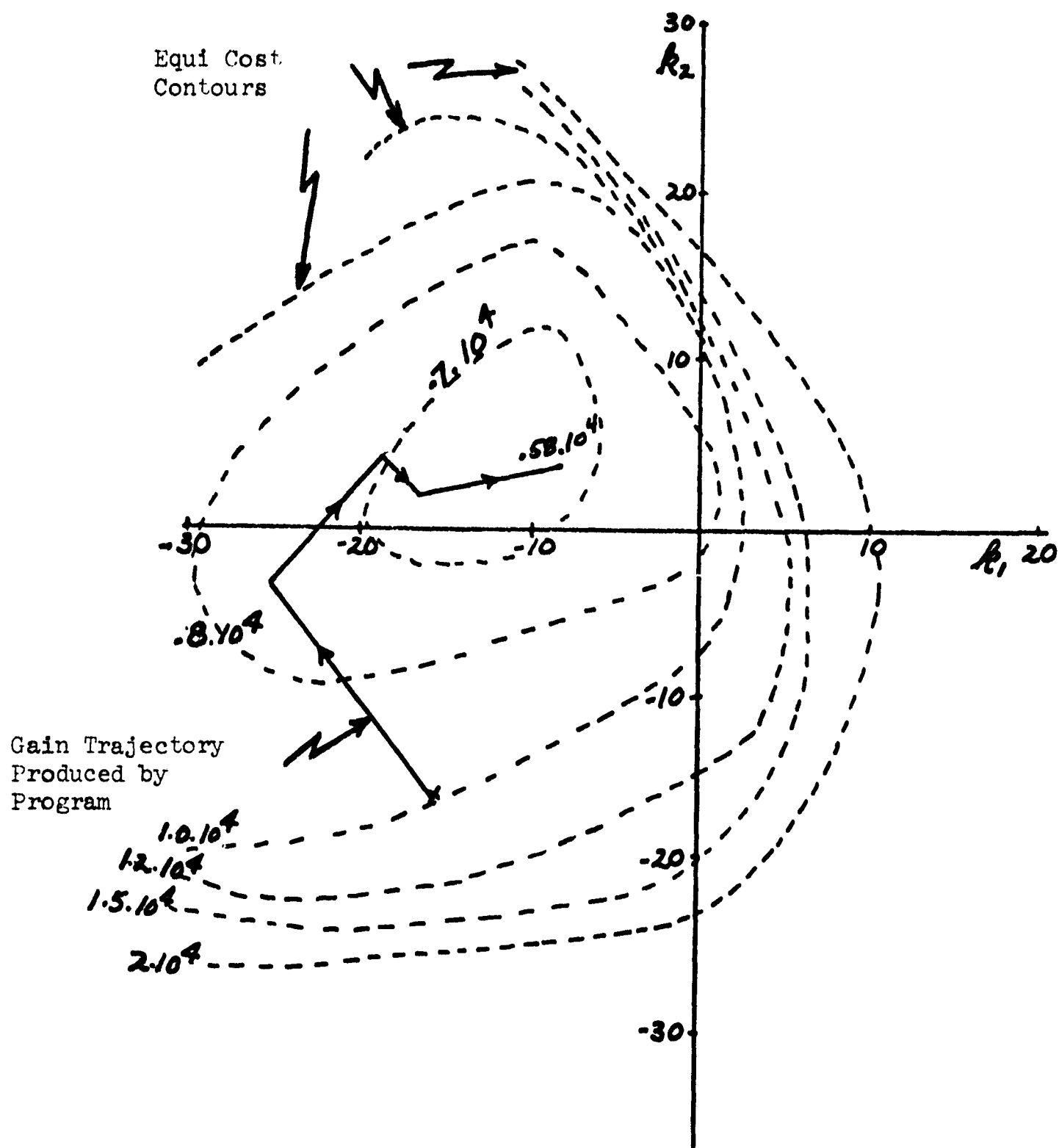


Figure 2 Expected Value of Cost as a Function of Feedback Gains

The example chosen illustrates a perhaps unexpected result. The optimal feedback gains correspond to an unstable system. This arises from the finite control interval used. It simply shows that the cost matrices F , Q and R stressed conservation of control energy at the expense of tightness of control. If stability is necessary then the cost matrices must be chosen accordingly. This differs, of course, from the case of a semi-infinite control interval, where stability of the optimal system is assured.

V. Conclusions

A computational algorithm has been derived to mechanize the theory of optimal time-invariant output-feedback controllers presented in [3]. The new algorithm uses the techniques of numerical integration. It is computationally much faster than the analytic algorithm presented in [3]. This allows its economic use on systems of higher order.

References:

1. Cassidy, J.F., Jr., "Optimal Control with Unavailable States,"
Ph.D. dissertation, Systems Engineering Division, Rensselaer
Polytechnic Institute, Troy, New York, 1969.
2. Levine, W.S., "Optimal Output-Feedback Controllers for Linear
Systems," Ph.D. dissertation, Department of Electrical
Engineering, Massachusetts Institute of Technology, Cambridge,
Massachusetts, 1969.
3. McBrinn, D.E., "Optimal Time-Invariant Output Feedback Controllers,"
Final Report, Vol. 1, Contract No. NAS8-21131, Rensselaer
Polytechnic Institute, Troy, New York, 1970.

```

/JOB          4257      MCRINA,PAGES=100,KP=MIX
C
C      PROGRAM IS.FT
C      DETERMINE OPTIMAL CONSTANT GAIN OUTPUT FEEDBACK CONTROLLERS
C      FOR FINITE TIME STATE REGULATOR PROBLEMS
C      OUTPUTS ARE ASSUMED TO BE FIRST NF STATES
C
1      COMMON M(7,7),MI(7,7),RC(7),AHAT(7,7),T,K(7,2),G(7,2),GHAT(7,7),VG
      1RAD(4,4),GRAD(4,4),VW(4),B(7,2),F(7,7),R(2,2),IS,NC,NF
2      DIMENSION A(7,7),Q(7,7)
3      COMPLEX*16 A,MI,RC
4      DOUBLE PRECISION AHAT,T,K,G,GHAT,GF,GF2,VW,VGRAD,GRAD
5      DOUBLE PRECISION TEST,TEST1,GASTOP
6      DOUBLE PRECISION DMAX1,DABS
7      DOUBLE PRECISION GF3,GF4
8      DOUBLE PRECISION G1(7,2)
9      10      FORMAT (4F18,7)
10     11      FORMAT (4D18,7)
11     13      FORMAT (7I10)
12     20      FORMAT (14I5)
13     25      FORMAT ('1',I4,' STATES',4X,'CONTROLS',4X,'FEEDBACKS',4X,'IGAINS')
14     30      FORMAT ('//T3,' INVERSE GRADIENT MATRIX')
15     35      FORMAT ('//T3,' SYSTEM MATRIX A')
16     40      FORMAT ('//T3,' NEW GAINS')
17     45      FORMAT ('//T3,' CONTROL MATRIX B')
18     50      FORMAT ('//T3,' GAIN MATRIX')
19     55      FORMAT ('//T3,' TERMINAL TIME T =',F18,7)
20     60      FORMAT ('//T3,' GAIN TOLERANCE ACHIEVED =',F18,7)
21     62      FORMAT ('//T3,' AVERAGE COST =',D20,7)
22     65      FORMAT ('//T3,' TERMINAL COST MATRIX F')
23     70      FORMAT ('//T3,' REQUIRED STOPPING TOLERANCE =',F18,7)
24     75      FORMAT ('//T3,' STATE WEIGHTING MATRIX Q')
25     80      FORMAT ('//T3,' CONTROL WEIGHTING MATRIX R')
26     85      FORMAT ('1',I7,' ITERATION NUMBER',I10)
27     86      FORMAT ('//T3,' STEP SIZE IS HALVED. NEW GAINS ARE')
28     87      FORMAT ('//T3,' STEP SIZE IS DOUBLED. NEW GAINS ARE')
29     88      FORMAT ('//T3,' STEP SIZE INTERPOLATION. NEW GAINS ARE')
30     89      FORMAT ('//T3,' ABOVE GAINS ARE BEST STEP FOR THIS ITERATION')
31     90      FORMAT ('//T3,' SOLUTION IS COMPLETE. FOLLOWING GAINS ARE OPTIMAL')
32     100     READ (1,20) NC,NC,NF,IGAINS
33           READ (1,10) ((A(I,J),J=1,NS),I=1,NS)
34           READ (1,10) ((B(I,J),J=1,NC),I=1,NS)
35           READ (1,11) T
36           READ (1,10) ((F(I,J),J=1,NS),I=1,NS)
37           READ (1,10) ((Q(I,J),J=1,NS),I=1,NS)
38           READ (1,10) ((R(I,J),J=1,NC),I=1,NC)
39           READ (1,11) GASTOP
40           WRITE (3,25)
41           WRITE (3,15) NS,NC,NF,IGAINS
42           WRITE (3,35)
43           WRITE (3,10) ((A(I,J),J=1,NS),I=1,NS)
44           WRITE (3,45)
45           WRITE (3,10) ((B(I,J),J=1,NC),I=1,NS)
46           WRITE (3,55) T
47           WRITE (3,65)
48           WRITE (3,10) ((F(I,J),J=1,NS),I=1,NS)
49           WRITE (3,75)
50           WRITE (3,10) ((Q(I,J),J=1,NS),I=1,NS)
51           WRITE (3,80)
52           WRITE (3,10) ((R(I,J),J=1,NC),I=1,NC)
53           NFPI=NF+1

```

```

54      NFC=NFC+NC
55      DO 1000 I=1,NS
56      DO 1000 J=1,NC
57      G(I,J)=0.000
58      1000 K(I,J)=0.000
59      IF(IGAIN5) 1200,1200,1100
60      1100 READ (1,11) ((K(I,J),J=1,NC),I=1,NF)
61      1200 CONTINUE
62      DO 1220 I=1,NS
63      DO 1220 J=NF,15
64      SHAT(I,J)=C(I,J)
65      1220 AHAT(I,J)=A(I,J)
66      IT=1
67      WRITE (3,95) IT
68      WRITE (3,50)
69      WRITE (3,10) ((K(I,J),J=1,NC),I=1,NF)
70      DO 1225 I=1,NF
71      DO 1225 J=1,NC
72      1225 G(I,J)=K(I,J)
73      1230 CONTINUE
74      CALL MATHAT(A,Q)
75      CALL STRAM
76      1260 CALL FEEN(GF)
77      IT=IT+1
78      CALL INVERT(VGRAD,GRADT,NFC)
79      WRITE (3,30)
80      WRITE (3,11) ((GRADT(I,J),J=1,NFC),I=1,NFC)
81      CALL NEWRT
82      WRITE (3,85) IT
83      WRITE (3,40)
84      WRITE (3,10) ((G(I,J),J=1,NC),I=1,NF)
85      1280 VMU=1.
86      DO 1290 I=1,NF
87      DO 1290 J=1,NC
88      1290 G1(I,J)=G(I,J)
89      GF1=GF
90      VMU1=0.
91      CALL MATHAT(A,Q)
92      CALL STRAM
93      CALL GAINZ(GF2)
94      IF(GF-GF2.GT.0.000) GO TO 1330
95      IHALF=1
96      1310 GF3=GF2
97      IHALF=IHAF+1
98      IF(IHALF.GT.5) GO TO 1500
99      WRITE (3,86)
100     VMU=VMU/2.
101     DO 1320 I=1,NF
102     DO 1320 J=1,NC
103     1320 G(I,J)=(K(I,J)+G(I,J))*0.5
104     WRITE (3,11) ((G(I,J),J=1,NC),I=1,NF)
105     CALL MATHAT(A,Q)
106     CALL STRAM
107     CALL GAINZ(GF2)
108     IF(GF2-GF.GT.0.000) GO TO 1310
109     GO TO 1500
110     1380 CONTINUE
111     WRITE (3,87)
112     DO 1390 I=1,NF
113     DO 1390 J=1,NC
114     1390 G(I,J)=2.*G(I,J)-K(I,J)

```

```

115      WRITE (3,11) ((G(I,J),J=1,NC),I=1,NF)
116      CALL MATHAT(A,Q)
117      CALL STRAM
118      CALL GAIN2(CF3)
119      IF(GF3-GF2.GT.0.000) GO TO 1500
120      VMU1=VMU
121      VMU=2.*VMU
122      GF1=GF2
123      GF2=GF3
124      GO TO 1380
125      1500 VMU2=VMU
126      VMU3=2.*VMU
127      KQUAD=1
128      1510 KQUAD=KQUAD+1
129      IF(KQUAD-3.GT.0) GO TO 1800
130      D1MU=VMU2-VMU1
131      D2MU=VMU3-VMU2
132      DGF1=GF2-GF1
133      DGF2=GF3-GF1
134      D3MU=.5*(D2MU+D2MU+DGF1-D1MU+D1MU*(DGF2)/(DGF1+D2MU-DGF2+D1MU))
135      VMU4=VMU1+D3MU
136      IF(ABS(D3MU-D1MU).LT.0.001) GO TO 1800
137      WRITE (3,88)
138      DO 1520 I=1,NF
139      DO 1520 J=1,NC
140      1520 G(I,J)=VMU4*C(I,J)+(1.-VMU4)*K(I,J)
141      WRITE (3,11) ((G(I,J),J=1,NC),I=1,NF)
142      CALL MATHAT(A,Q)
143      CALL STRAM
144      CALL GAIN2(CF4)
145      IF(D3MU.GT.D1MU) GO TO 1600
146      IF(GF4.GT.GF2) GO TO 1550
147      GF3=GF2
148      VMU3=VMU2
149      GF2=GF4
150      VMU2=VMU4
151      GO TO 1510
152      1550 GF1=GF4
153      VMU1=VMU4
154      GO TO 1510
155      1600 IF(GF4.GT.GF2) GO TO 1650
156      GF1=GF2
157      VMU1=VMU2
158      GF2=GF4
159      VMU2=VMU4
160      GO TO 1510
161      1650 GF3=GF4
162      VMU3=VMU4
163      GO TO 1510
164      1800 DO 1810 I=1,NF
165      DO 1810 J=1,NC
166      1810 G(I,J)=VMU2*G(I,J)+(1.-VMU2)*K(I,J)
167      WRITE (3,88)
168      WRITE(3,11) ((G(I,J),J=1,NC),I=1,NF)
169      WRITE (3,87)
170      TEST1=0.000
171      DO 2000 I=1,NF
172      DO 2000 J=1,NC
173      IF (G(I,J).EQ.0.000) GO TO 1900
174      TEST=DABS((G(I,J)-K(I,J))/G(I,J))
175      TEST1=DMAX1(TEST,TEST1)

```

```

176      GO TO 2000
177      1900 IF(G(I,J)-K(I,J).EQ.0.000) GO TO 2000
178      TEST1=:000,*G,STOP
179      2000 K(I,J)=G(I,J)
180      WRITE (3,60) TEST1
181      WRITE (3,70) G,STOP
182      IF(TEST1-G,STOP.GT.0.000) GO TO 1230
183      WRITE (3,90)
184      WRITE (3,10) ((G(I,J),J=1,NC),I=1,NF)
185      WRITE (3,62) GF2
186      GO TO 8000
187      8000 CONTINUE
188      9000 CONTINUE
189      STOP
190      END

```

```

191      SUBROUTINE MATMAT(A,Q)

```

```

C

```

```

C      COMPUTES MATRICES AHAT AND QHAT

```

```

C

```

```

192      COMMON M(7,7),MI(7,7),RC(7),AHAT(7,7),T,K(7,2),G(7,2),QHAT(7,7),VG
193      IRAD(4,4),GRADI(4,4),VW(4),B(7,2),F(7,7),R(2,2),VS,NC,NF
194      DIMENSION A(7,7),Q(7,7)
195      COMPLEX*16 M,MI,RC
196      DOUBLE PRECISION AHAT,T,K,G,CHAT,GF,GF2,VW,VGRAD,GRADI
197      DO 100 I=1,NS
198      DO 100 J=1,NF
199      AHAT(I,J)=A(I,J)
200      QHAT(I,J)=Q(I,J)
201      AHAT(I,J)=AHAT(I,J)+B(I,NI)*G(J,NI)
202      DO 100 NI=1,NC
203      100 QHAT(I,J)=QHAT(I,J)+G(I,NI)*R(NI,N2)*G(J,N2)
204      RETURN
205      END

```

```

206      SUBROUTINE STRAM

```

```

C

```

```

C      COMPUTES THE STATE TRANSITION MATRIX

```

```

C

```

```

207      COMMON M(7,7),MI(7,7),RC(7),AHAT(7,7),T,K(7,2),G(7,2),QHAT(7,7),VG
208      IRAD(4,4),GRADI(4,4),VW(4),B(7,2),F(7,7),R(2,2),VS,NC,NF
209      DIMENSION AAAA(49),RR(7),RI(7),ASQR(7,7),ASQ2(7,7),XR(7),XI(7),
210      VR(7),VI(7),IANA(7),IRGW(7,2),VRN(7),VIN(7),W(7,4)
211      COMPLEX*16 M,MI,RC
212      COMPLEX*16 COMPLEX,DCC,TJG
213      DOUBLE PRECISION AHAT,T,K,G,CHAT,GF,GF2,VW,VGRAD,GRADI
214      DOUBLE PRECISION AAAA,RR,RI,ASQR,ASQ2,XR,XI,VR,VI,VRN,VIN,W,VECMGR
215      1 VECMGT,VECMGS,SW1
216      10 FORMAT (2D19.7)
217      30 FORMAT (7F2,'SYSTEM EIGENVALUES')
218      CALL VECT(AHAT,AAAA,NS)
219      CALL HSUB(NS,AAAA,NS)
220      CALL ATEIG(NS,AAAA,RR,RI,IANA,NS)
221      WRITE (3,30)
222      WRITE (3,10) (RR(I),RI(I),I=1,NS)
223      CALL MSC(AHAT,IS,ASQR)
224      NEIG=0
225      CONTINUE
226      DO 100 I=1,NS
227      RC(I)=DCMPLX(RR(I),RI(I))

```

```

225      DO 100 J=1,NS
226      ASQ2(I,J)=ASC2(I,J)
227      CALL EIGVEC(3,AHAT,ASC2,W,IRCW,XR,XI,VR,VI,RR(I),RI(I),NS,NS,0,
1          SW1,ITER,DIF,2)
228      NEIG=NEIG+1
229      IF(NEIG.GT,5) GO TO 105
230      IF(ITER.LT,15) GO TO 105
231      CALL RAYL(AHAT,RR(I),XI(I),XR,XI,VR,VI,NS,NS)
232      GO TO 50
233      105 CONTINUE
234      VECMGR=0.0DC
235      VECMGI=0.0DC
236      DO 110 I=1,NS
237      VECMGR=VECMGR+VR(I)*XR(I)-VI(I)*XI(I)
238      110 VECMGI=VECMGI+VR(I)*XI(I)+VI(I)*XR(I)
239      VECMGS=VECMGR*VECMGR+VECMGI*VECMGI
240      DO 120 I=1,NS
241      VRN(I)=(VR(I)*VECMGR+VI(I)*VECMGI)/VECMGS
242      VIN(I)=(VI(I)*VECMGR-VR(I)*VECMGI)/VECMGS
243      M(I,1)=DCMPLX(XR(I),XI(I))
244      120 MI(I,1)=DCMPLX(VRN(I),VIN(I))
245      DO 1000 KOUNT=2,NS
246      KOUNT1=KOUNT-1
247      IF(RR(KOUNT)-RR(KOUNT1)) 200,140,200
248      140 CONTINUE
249      DO 150 I=1,NS
250      M(I,KOUNT)=DCONJG(M(I,KOUNT1))
251      150 MI(KOUNT,I)=DCONJG(MI(KOUNT1,I))
252      GO TO 1000
253      200 DO 210 I=1,NS
254      DO 210 J=1,NS
255      210 ASQ2(I,J)=ASC2(I,J)
256      CALL EIGVEC(3,AHAT,ASQ2,W,IRCW,XR,XI,VR,VI,RR(KOUNT),RI(KOUNT),NS,
1          NS,0,SW1,ITER,DIF,2)
257      VECMGR=0.0DC
258      VECMGI=0.0DC
259      DO 220 I=1,NS
260      VECMGR=VECMGR+VR(I)*XR(I)-VI(I)*XI(I)
261      220 VECMGI=VECMGI+VR(I)*XI(I)+VI(I)*XR(I)
262      VECMGS=VECMGR*VECMGR+VECMGI*VECMGI
263      DO 230 I=1,NS
264      VRN(I)=(VR(I)*VECMGR+VI(I)*VECMGI)/VECMGS
265      VIN(I)=(VI(I)*VECMGR-VR(I)*VECMGI)/VECMGS
266      M(I,KOUNT)=DCMPLX(XR(I),XI(I))
267      230 MI(KOUNT,I)=DCMPLX(VRN(I),VIN(I))
268      1000 CONTINUE
269      RETURN
270      END

271      SUBROUTINE FEHNF(GF)
272      COMMON F(7,7),VI(7,7),RC(7),AHAT(7,7),T,K(7,2),S(7,2),QHAT(7,7),VG
1      IRAD(4,4),GRAD(4,4),VW(4),R(7,2),F(7,7),R(2,2),NS,NC,NF
273      COMPLEX*16 A,-1,RC,CDEXP
274      COMPLEX*16 FXS(7,8)
275      DOUBLE PRECISION AHAT,T,K,G,CHAT,GF,GF2,VW,VCRAD,GRAD,
1      FEE(7,7,8),DFEE(7,7,2,2,8),D2FEE(7,7,2,2,2,8),FFT1(7,7),
2      WRK(7,7,8),WRK1(7,7),FCFT1(7,7,2,2),WRK2(7,7,8),WRK3(7,7,8)
3      ,WRK4(7,7)
276      DOUBLE PRECISION FEQTR(7,7),FEHLF(7,7),DFEHLF(7,7,2,2),
1      D1NFE(7,7,2,2,2,2),D2NFE(7,7,2,2,2,2)
277      CALL TRAPS(C,U,100000,0,0)

```

```

278 10  FORMAT (//T3, 'NECESSARY CONDITIONS VECTOR')
279 20  FORMAT(4D18,/)
280 30  FORMAT(//T3, 'GRADIENT MATRIX')
281 62  FORMAT(//T3, 'AVERAGE COST =',D20.7)
282    NFC=NF*NC
283    DO 100 I=1,NS
284    DO 100 KT=1,8
285 100  EX3(I,KT)=CDEXP(RC(I)*T*KT/8.000)
286    DO 200 I=1,NS
287    DO 200 J=1,NS
288    FEQTR(I,J)=C.000
289    FEHLF(I,J)=C.000
290    DO 190 N1=1,NS
291    FEQTR(I,J)=FEQTR(I,J)+M(I,N1)*M1(N1,J)*CDEXP(RC(N1)*T/32.000)
292 190  FEHLF(I,J)=FEHLF(I,J)+M(I,N1)*M1(N1,J)*CDEXP(RC(N1)*T/16.000)
293    DO 200 KT=1,8
294    FEE(I,J,KT)=C.000
295    DO 200 N1=1,NS
296 200  FEE(I,J,KT)=FEE(I,J,KT)+M(I,N1)*M1(N1,J)*EX3(N1,KT)
297    DO 300 K1=1,NF
298    DO 300 K2=1,NC
299    DO 300 I=1,NS
300    DO 300 J=1,NS
301    DFELHF(I,J,K1,K2)=0.000
302    DFEE(I,J,K1,K2,1)=0.000
303    DO 300 N1=1,NS
304    DFELHF(I,J,K1,K2)=DFELHF(I,J,K1,K2)-FEQTR(I,N1)*B(N1,K2)*FEQTR(K1,J)
      TJ)*T/16.000
305 300  DFEE(I,J,K1,K2,1)=DFEE(I,J,K1,K2,1)-FEHLF(I,N1)*B(N1,K2)*FEHLF(K1,J)*T/8
      1J)*T/8.000
306    DO 400 KT=2,8
307    KT1=KT-1
308    DO 400 K1=1,NF
309    DO 400 K2=1,NC
310    DO 400 I=1,NS
311    DO 400 J=1,NS
312    DFEE(I,J,K1,K2,KT1)=0.000
313    DO 400 N1=1,NS
314 400  DFEE(I,J,K1,K2,KT1)=DFEE(I,J,K1,K2,KT1)+FEE(I,N1,1)*DFEE(N1,J,K1,K2,KT1)
      1KT1)+DFEE(I,N1,K1,K2,1)*FEE(N1,J,KT1)
315    DO 500 K1=1,NF
316    DO 500 K2=1,NC
317    DO 500 K3=1,NF
318    DO 500 K4=1,NC
319    DO 500 I=1,NS
320    DO 500 J=1,NS
321    D1MFE(I,J,K1,K2,K3,K4)=0.000
322    D2MFE(I,J,K1,K2,K3,K4)=0.000
323    DO 450 N1=1,NS
324    D1MFE(I,J,K1,K2,K3,K4)=D1MFE(I,J,K1,K2,K3,K4)-
      1FEHLF(I,1)*B(N1,K2)*DFELHF(K1,J,K3,K4)*T/8.000
325 450  D2MFE(I,J,K1,K2,K3,K4)=D2MFE(I,J,K1,K2,K3,K4)-
      1FEHLF(I,N1)*B(N1,K4)*DFELHF(K3,J,K1,K2)*T/8.000
326 500  D2FEE(I,J,K1,K2,K3,K4,1)=D1MFE(I,J,K1,K2,K3,K4)+D2MFE(I,J,K1,K2,K3
      1,K4)
327    DO 600 KT=2,8
328    KT1=KT-1
329    DO 600 K1=1,NF
330    DO 600 K2=1,NC
331    DO 600 K3=1,NF
332    DO 600 K4=1,NC

```

```

333      DO 600 I=1,NS
334      DO 600 J=1,NS
335      D2FEE(I,J,K1,K2,K3,K4,KT)=0.000
336      DO 600 N1=1,NS
337      600  D2FEE(I,J,K1,K2,K3,K4,KT)=D2FEE(I,J,K1,K2,K3,K4,KT)+
1          FEE(I,N1,I)*D2FEE(I,J,K1,K2,K3,K4,KT)+D1MFE(I,I,K1,K2,K3,K
14)*DFEE(N1,J,K3,K4,KT)+D2MFE(I,N1,K1,K2,K3,K4)*DFEE(N1,J,K1,K2,KT)
11)
338      CALL SIMPRD(F,E,FEE,FFT1,T,NS)
339      DO 620 I=1,NS
340      620  FFT1(I,I)=FFT1(I,I)+T/24.000
341      DO 700 K1=1,NF
342      DO 700 K2=1,NC
343      DO 650 I=1,NS
344      DO 650 J=1,NS
345      DO 650 KT=1,8
346      650  WRK(I,J,KT)=DFEE(I,J,K1,K2,KT)
347      CALL SIMPRD(F,E,WRK,WRK1,T,NS)
348      DO 700 I=1,NS
349      DO 700 J=1,NS
350      700  FDFTI(I,J,K1,K2)=WRK1(I,J)
351      GF=0.000
352      DO 1000 N1=1,NS
353      DO 1000 N2=1,NS
354      GF=GF+QHAT(N1,N2)*FFT1(N2,N1)
355      DO 1000 N3=1,NS
356      1000 GF=GF+F(N1,N2)*FEE(N2,N3,8)*FEE(N1,N3,8)
357      GF=GF/NS
358      WRITE(3,62) GF
359      DO 2100 I=1,NF
360      DO 2100 J=1,NC
361      IN=NF*(J-1)+I
362      VW(IN)=0.000
363      DO 2100 N1=1,NS
364      DO 2000 N2=1,NS
365      VW(IN)=VW(IN)+QHAT(N1,N2)*FDFTI(N2,N1,I,J)
366      DO 2000 N3=1,NS
367      2000 VW(IN)=VW(IN)+F(N1,N2)*FEE(N2,N3,8)*DFEE(N1,N3,I,J,8)
368      DO 2100 N4=1,NC
369      2100 VW(IN)=VW(IN)+R(J,N4)*K(N1,N4)*FFT1(N1,I)
370      WRITE(3,10)
371      WRITE(3,20) (VW(I),I=1,NFC)
372      DO 3500 I=1,NF
373      DO 3500 J=1,NC
374      DO 3000 N1=1,NS
375      DO 3000 N2=1,NS
376      DO 3000 KT=1,8
377      3000 WRK(N1,N2,KT)=DFEE(N1,N2,I,J,KT)
378      IN=NF*(J-1)+I
379      DO 3500 K1=1,NF
380      DO 3500 K2=1,NC
381      ID=NF*(K2-1)+K1
382      VGRAD(I-ID)=R(J,K2)*FFT1(K1,I)
383      DO 3100 N1=1,NS
384      DO 3100 N2=1,NS
385      DO 3100 KT=1,8
386      WRK2(N1,N2,KT)=DFEE(N1,N2,K1,K2,KT)
387      3100 WRK3(N1,N2,KT)=D2FEE(N1,N2,I,J,K1,K2,KT)
388      CALL SIMPRD(WRK2,WRK,WRK1,T,NS)
389      CALL SIMPRD(F,E,WRK3,WRK4,T,NS)
390      DO 3500 N1=1,NS

```

```

391      DO 3200 N2=1, NS
392      VGRAD(I1, I2)=VGRAD(I1, I2)+CHAI(N1, N2)*(WRK1(N2, N1)+WRK4(N2, N1))
393      DO 3200 N3=1, NS
394      3200 VGRAD(I1, I2)=VGRAD(I1, I2)+F(N1, N2)*(CFEE(N2, N3, K1, K2, 8)+CFEE(N1, N3
1, I, J, 8)+FEE(N2, N3, 3)+D2FEE(N1, N3, K1, K2, 1, J, 9))
395      DO 3500 N4=1, NC
396      3500 VGRAD(I1, I2)=VGRAD(I1, I2)+R(K1, N4)*K(N1, N4)*(FDFTI(N1, K1, I, J)+FDFI
1(K1, N1, I, J)+R(J, N4)*K(N1, N4)*(FDFTI(N1, I, K1, K2)+FDFTI(I, N1, K1, K2
2))
397      WRITE(3, 301)
398      WRITE(3, 201) ((VGRAD(I, J), J=1, NFC), I=1, NFC)
399      RETURN
400      END

```

```

401      SUBROUTINE GAIJ2(GF2)
402      COMMON M(7,7), V1(7,7), RC(7), AHAT(7,7), T, K(7,2), S(7,2), GHAT(7,7), VG
1RAJ(4,4), GRAD(4,4), VW(4), B(7,2), F(7,7), R(2,2), NS, NC, NF
403      DOUBLE PRECISION AHAT, T, K, G, CHAI, GF, GF2, VW, VGRAD, GRAD
404      DOUBLE PRECISION FEE(7,7,8), FFI(7,7)
405      COMPLEX*16 M, M1, RC, CDEXP
406      COMPLEX*16 EX3(7,8)
407      62 FORMAT(//T3, 'AVERAGE COST GF2 =', D20.7)
408      DO 100 I=1, NS
409      DO 100 KT=1, 8
410      100 EX3(I, KT)=CDEXP(RC(I)*T*KT/8.000)
411      DO 200 J=1, NS
412      DO 200 J=1, NS
413      DO 200 KT=1, 8
414      FEE(I, J, KT)=0.000
415      DO 200 N1=1, NS
416      200 FEE(I, J, KT)=FEE(I, J, KT)+M(I, N1)*M1(N1, J)*EX3(N1, KT)
417      CALL SIMPRD(F, FEE, FFI, I, NS)
418      DO 210 I=1, NS
419      210 FFI(I, I)=FFI(I, I)+T/24.000
420      GF2=0.000
421      DO 1000 N1=1, NS
422      DO 1000 N2=1, NS
423      GF2=GF2+CHAI(N1, N2)*FFI(N2, N1)
424      DO 1000 N3=1, NS
425      1000 GF2=GF2+F(N1, N2)*FEE(N2, N3, 3)+FEE(N1, N3, 3)
426      GF2=GF2/NS
427      WRITE(3, 62) GF2
428      RETURN
429      END

```

```

430      SUBROUTINE SIMPRD(A, B, AB, T, N)
431      DOUBLE PRECISION A(7,7,8), B(7,7,8), AB(7,7), T
432      DO 100 I=1, N
433      DO 100 J=1, N
434      AB(I, J)=0.000
435      DO 100 N1=1, N
436      100 AB(I, J)=AB(I, J)+T/4.000*(A(I, N1, 1)*B(J, N1, 1)+A(I, N1, 3)*B(J, N1, 3)+A(I
1, N1, 5)*B(J, N1, 5)+A(I, N1, 7)*B(J, N1, 7))+2.000*(A(I, N1, 2)*B(J, N1, 2)+
2A(I, N1, 4)*B(J, N1, 4)+A(I, N1, 6)*B(J, N1, 6))+A(I, N1, 8)*B(J, N1, 8))*T/
324.000
437      RETURN
438      END

```

```

439      SUBROUTINE INVERT(A, B, N)

```

C

INVERTS A TO GIVE B

```

C
440      DIMENSION A(4,4),B(4,4)
441      DOUBLE PRECISION A,B,C,D,X
442      DOUBLE PRECISION DABS
443      IF(N-1) 100,100,101
444      100  B(1,1)=1.0007/(1,1)
445      RETURN
446      101  DO 102 I=1,N
447      DO 102 J=1,N
448      102  B(I,J)=0.000
449      DO 103 I=1,N
450      103  B(I,I)=1.000
C      PICK UP PIVOT ELEMENT
451      DO 114 K=1,N
452      L=K
453      IF(N-K) 110,110,104
454      104  I=K+1
455      DO 106 JJ=1,N
456      IF(DABS(A(I,J,K))-DABS(A(L,K)))106,106,105
457      105  L=JJ
458      106  CONTINUE
459      IF(L-K) 107,110,107
C      PERFORM ROW INTERCHANGE
460      107  DO 108 J=K,N
461      C=A(K,J)
462      A(K,J)=A(L,J)
463      108  A(L,J)=C
464      DO 109 J=1,N
465      C=B(K,J)
466      B(K,J)=B(L,J)
467      109  B(L,J)=C
C      COLUMN ELIMINATION
468      110  DO 114 I=1,N
469      IF(K-I) 111,114,111
470      111  D=A(I,K)/A(K,K)
471      DO 112 J=K,N
472      A(I,J)=A(I,J)-D*A(K,J)
473      A(I,K)=0.000
474      DO 113 J=1,N
475      113  B(I,J)=B(I,J)-D*B(K,J)
476      114  CONTINUE
C      SOLVE FOR INVERSE
477      DO 115 J=1,N
478      DO 115 I=1,N
479      X=B(I,J)/A(I,I)
480      115  B(I,J)=X
481      RETURN
482      END
483      SUBROUTINE NEWKIT
C
C      PERFORMS NEWTON-RAPHSON ITERATION
C
484      COMMON M(7,7),MI(7,7),RC(7),AHAT(7,7),T,K(7,2),G(7,2),QHAT(7,7),VG
485      1RAD(4,4),GRADI(4,4),VW(4),B(7,2),F(7,7),R(2,2),NS,NC,NF
486      COMPLEX*16 M,MI,RC
487      DOUBLE PRECISION AHAT,T,K,G,CHAT,GF,GF2,VW,VGRAD,GRADI
488      DO 1000 I=1,NF
489      DO 1000 J=1,NL
490      G(I,J)=K(I,J)
491      IV=NF*(J-1)+I

```

```

491      DO 1000 K1=1,NF
492      DO 1000 K2=1,NC
493      ID=NF*(K2-1)+K1
494      G(I,J)=G(I,J)+GRAC(I,N,TET*V(I,ID))
495      RETURN
496      END

497      SUBROUTINE VECT(AHAT,AAAA,NS)
      C
      C CONVERTS AHAT TO SINGLE SUBSCRIPT FORM AAAA
      C
498      DIMENSION AHAT(7,7),AAAA(49)
499      DOUBLE PRECISION AHAT,AAAA
500      DO 100 J=1,NS
501      DO 100 I=1,NS
502      K=(J-1)*NS+I
503      100 AAAA(K)=AHAT(I,J)
504      RETURN
505      END

506      SUBROUTINE MS(AHAT,NS,ASCR)
      C
      C COMPUTES ASCR=AHAT*AHAT
      C
507      DIMENSION AHAT(7,7),ASCR(7,7)
508      DOUBLE PRECISION AHAT,ASCR
509      DO 100 I=1,NS
510      DO 100 J=1,NS
511      ASCR(I,J)=0.000
512      DO 100 K=1,NS
513      100 ASCR(I,J)=ASCR(I,J)+AHAT(I,K)*AHAT(K,J)
514      RETURN
515      END

516      SUBROUTINE RAYL(A,E,EI,X,XI,V,VI,N,MD)
517      REAL*8 E,EI,X(MD),XI(MD),V(MD),VI(MD),A(MD,MD),
      C DVXR,DVXI,DVAXR,DVAXI,A1,A2,A3
518      REAL*8 DXDR(12),DXDI(12)
      C
      C FOR SYSTEMS OF ORDER HIGHER THAN 12 CHANGE THE FOLLOWING REAL*8
      C STATEMENT
      C
519      REAL*8 DAT(12,12)
      C
520      800 FORMAT(1X,5C12.4)
521      DO 10 T=1,N
522      DO 10 J=1,N
523      10 DA(I,J)=A(I,J)
524      DO 20 T=1,N
525      20 DA(I,I)=DA(I,I)-E
526      DVXR=0.0
527      DO 30 T=1,N
528      DVXR=DVXR+V(I)*X(I)
529      DXDR(I)=0.0
530      DO 30 L=1,N
531      30 DXDR(I)=DXDR(I)+DA(I,L)*X(L)
532      DVAXR=0.0
533      DO 40 T=1,N
534      40 DVAXR=DVAXR+V(I)*DXDR(I)
535      IF(EI) 60,50,60
536      50 E=E+DVAXR/DVXR

```

```

537      RETURN
538      60      DVXI=0.0
539      WRITE(3,800) DVXR
540      WRITE(3,800) (DXDR(LL),LL=1,N)
541      DO 70 I=1,N
542      DVXR=DVXR-V(I)*XI(I)
543      70      DVXI=DVXI+V(I)*XI(I)+V(I)*XI(I)
544      WRITE(3,800) DVXR,DVXI
545      DO 80 I=1,N
546      DXDI(I)=0.0
547      DO 30 J=1,N
548      80      DXDI(I)=DXDI(I)+CA(I,J)*XI(J)
549      WRITE(3,800) (DXDI(LL),LL=1,N)
550      A1=0.0
551      A2=0.0
552      DO 90 I=1,N
553      A1=A1+V(I)*DXDI(I)
554      A2=A2+V(I)*DXDI(I)
555      A3=0.0
556      90      A3=A3+V(I)*DXDI(I)
557      WRITE(3,800) A1,A2,A3
558      DVAXR=DVAXR+EI*DVI-A1
559      DVAXI=A2+A3-EI*DVI
560      A1=DVXR*DVXR+DVXI*DVXI
561      WRITE(3,800) DVAXR,DVAXI
562      WRITE(3,800) A1
563      E=E+(DVXR*DVAXR+DVXI*DVAXI)/A1
564      EI=EI+(DVAXI*DVXR-DVAXR*DVXI)/A1
565      RETURN
566      END

```

```

567      SUBROUTINE FSG(N,A,IA)

```

```

      C
      C
      C

```

```

568      DIMENSION A(4)
569      DOUBLE PRECISION A,PIV,T,S
570      DOUBLE PRECISION DABS
571      L=4
572      NIA=L-IA
573      LIA=NIA-IA
574      20 IF(L-3) 360,40,40
575      40 LIA=LIA-IA
576      L1=L-1
577      L2=L1-1
578      ISUB=LIA+L
579      IPIV=ISUB-IA
580      PIV=DABS(A(IPIV))
581      IF(L-3) 90,50,30
582      50 M=IPIV-IA
583      DO 80 I=L,M,IA
584      T=DABS(A(I))
585      IF(T-PIV) 80,60,60
586      60 IPIV=I
587      PIV=T
588      80 CONTINUE
589      90 IF(PIV) 100,300,100
590      100 IF(PIV-DABS(A(ISUB))) 130,180,120
591      120 M=IPIV-L
592      DO 140 I=1,L
593      J=M+I

```

```

594      T=A(J)
595      K=LIA+I
596      A(J)=A(K)
597      140 A(K)=T
598      M=L2-M/IA
599      DO 160 I=L1,NIA,IA
600      T=A(I)
601      J=I-M
602      A(I)=A(J)
603      160 A(J)=T
604      180 DO 200 I=L,LIA,IA
605      200 A(I)=A(I)/A(I(503))
606      J=-IA
607      DO 240 I=1,L2
608      J=J+IA
609      LJ=L+J
610      DO 220 K=1,L1
611      KJ=K+J
612      KL=K+LIA
613      220 A(KJ)=A(KJ)-A(LJ)*A(KL)
614      240 CONTINUE
615      K=-IA
616      DO 300 I=1,N
617      K=K+IA
618      LK=K+L1
619      S=A(LK)
620      LJ=L-IA
621      DO 280 J=I,L2
622      JK=K+J
623      LJ=LJ+IA
624      280 S=S+A(LJ)*A(JK)*I.000
625      300 A(LK)=S
626      DO 310 I=L,LIA,IA
627      310 ATITE=0.000
628      320 L=L1
629      GO TO 20
630      360 RETURN
631      END

632      SUBROUTINE ATEIG(M,AT,RR,RI,IANA,IA)
      C
      C      COMPUTES ROOTS OF UPPER HESSENBERG MATRIX A
      C
633      DIMENSION A(4, ), PR(7), RI(7), PRR(2), PRI(2), IA, A(7)
634      DOUBLE PRECISION E7,E6,E10,DELTA,PRR,PRI,PAV,PAV1,R,S,T,A,U,V,RR,
      RI,RMCD,EPS,D,G1,G2,G3,CAP,PSI1,PSI2,ALPHA,ETA
635      DOUBLE PRECISION DASS,DSRT,DMAX1
636      INTEGE< P,P1,w
637      E7=1.00-8
638      E6=1.00-6
639      E10=1.00-10
640      DELTA=0.500
641      MAXIT=30
642      N=M
643      20 NI=N-I
644      IN=N1+IA
645      NV=IN+N
646      IF(NI<30,1300,30)
647      30 NP=I+1
648      IT=0
649      DO 40 I=1,2

```

```

650      PRR(1)=0.000
651      40  PRI(1)=0.000
652      PA1=C.000
653      PAN1=0.000
654      R=0.000
655      S=0.000
656      N2=N1-1
657      IN1=IN-1A
658      VN1=IN1+N1
659      N1N=IN+N1
660      N1N1=N1N+N1
661      60  T=A(N1N1)-A(N1)
662      U=T+T
663      V=4.000*A(N1N1)*A(NN1)
664      IF(DABS(V)-0.7) 100,100,65
665      65  T=U+V
666      IF(DABS(T)-DMAX1(U,DABS(V))*E6) 67,67,68
667      67  T=0.000
668      68  U=(A(N1N1)+A(NN1))/2.000
669      V=DSQRT(DABS(T))/2.000
670      IF(T)140,70,70
671      70  IF(U) 80,75,75
672      75  RR(N1)=U+V
673      RR(N1)=U-V
674      GO TO 130
675      80  RR(N1)=U-V
676      RR(N1)=U+V
677      GO TO 130
678      100 IF(T)120,110,110
679      110 RR(N1)=A(N1N1)
680      RR(N1)=A(NN1)
681      GO TO 130
682      120 RR(N1)=A(NN1)
683      RR(N1)=A(N1N1)
684      130 RI(1)=0.000
685      RI(N1)=0.000
686      RI(N1)=0.0
687      GO TO 160
688      140 RR(N1)=U
689      RR(N1)=U
690      RI(N1)=V
691      RI(N1)=-V
692      160 IF(N2)1230,1230,130
693      180 N1N2=N1N1-1A
694      RMCD=RR(N1)*RR(N1)+RI(N1)*RI(N1)
695      EPS=E10*DSQRT(RMCD)
696      IF(DABS(A(N1N2))-EPS) 1230,1230,240
697      240 IF(DABS(A(N1N1))-E10*DABS(A(NN1))) 1300,1300,250
698      250 IF(DABS(PAN1-A(N1N2))-DABS(A(N1N2))*E6) 1240,1240,260
699      260 IF(DABS(PAN1-A(N1N1))-DABS(A(NN1))*E6) 1240,1240,300
700      300 IF(IT-4AXIT) 320,1240,1240
701      320 J=1
702      DO 360 I=1,2
703      K=4P-I
704      IF(DABS(RR(K)-PRR(1))+DABS(RI(K)-PRI(1))-DELTA*(DABS(RR(K))
1      +DABS(RI(K)))) 340,360,360
705      340 J=J+1
706      360 CONTINUE
707      GO TO (440,460,460,430),J
708      440 R=0.000
709      S=0.000

```

```

710      GO TO 500
711      460 J=I+2-J
712      R=RR(J)*RR(J)
713      S=RR(J)+RR(J)
714      GO TO 500
715      480 R=RR(I)*RR(N1)-RI(A)*RI(A1)
716      S=RR(I)+RR(N1)
717      500 PA1=A(I,1)
718      PA11=A(I,1N2)
719      DO 520 I=1,2
720      K=NP-I
721      PR1(I)=RR(K)
722      520 PR1(I)=RI(K)
723      P=N2
724      IF(I-3)600,600,525
725      525 IPI=N1N2
726      DO 580 J=2,N2
727      IPI=IPI-IA-1
728      IF(DABS(A(IPI))-EPS) 630,600,530
729      530 IPIP=IPI+IA
730      IPIP2=IPIP+IA
731      D=A(IPIP)*(A(IPIP)-S)+A(IPIP2)*A(IPIP+1)+R
732      IF(J) 540,560,540
733      540 IF(DABS(A(IPI)*A(IPIP+1))*(DABS(A(IPIP)+A(IPIP2+1)-S)+DABS(A(IPIP2
+2))) - DABS(D)*EPS) 620,620,560
734      560 P=J1-J
735      580 CONTINUE
736      600 Q=P
737      GO TO 680
738      620 P1=P-1
739      Q=P1
740      IF(P1-1)680,600,650
741      650 DO 660 I=2,P1
742      IPI=IPT-IA-1
743      IF(DABS(A(IPI))-EPS) 680,660,660
744      660 Q=Q-1
745      680 II=(P-I)+IA+P
746      DO 1220 I=P,N1
747      III=II-IA
748      IIP=II+IA
749      IF(I-P)720,700,720
750      700 IPI=II+1
751      IPIP=IIP+1
752      G1=A(II)*(A(III)-S)+A(IIP)*A(IPI)+R
753      G2=A(IPI)*(A(IPIP)+A(III)-S)
754      G3=A(IPI)*A(IPIP+1)
755      A(IPI+1)=0.000
756      GO TO 780
757      720 G1=A(III)
758      G2=A(III+1)
759      IF(I-2)740,7+0,760
760      740 G3=A(III+2)
761      GO TO 780
762      760 G3=0.000
763      780 CAP=DSRT(G1+G1+G2+G2+G3+G3)
764      IF(CAP)800,860,800
765      800 IF(G1)820,840,840
766      820 CAP=-CAP
767      840 T=G1+CAP
768      PSI1=G2/T
769      PSI2=G3/T

```

```

770      ALPHA=2.000/(1.0 CD0+PSI1*PSI1+PSI2*PSI2)
771      GO TO 880
772      860  ALPHA=2.000
773      PSI1=0.000
774      PSI2=0.000
775      880  IF(I-1) 700,960,900
776      900  IF(I-2) 920,940,720
777      920  A(111)=-CAP
778      GO TO 960
779      940  A(111)=-A(111)
780      960  IJ=11
781      DO 1240 J=1,N
782      T=PSI1*A(IJ+11)
783      IF(I-1) 980,1000,1000
784      980  IP2J=IJ+2
785      T=T+PSI2*A(IP2J)
786      1000  ETA=ALPHA*(1+A(IJ))
787      A(IJ)=A(IJ)-ETA
788      A(IJ+11)=A(IJ+11)-PSI1*ETA
789      IF(I-1) 1020,1040,1040
790      1020  A(IP2J)=A(IP2J)-PSI2*ETA
791      1040  IJ=IJ+1A
792      IF(I-1) 1080,1060,1060
793      1060  K=1
794      GO TO 1100
795      1080  K=1+2
796      1100  IP=IIP-1
797      DO 1180 J=C,K
798      JIP=IP+J
799      JI=JIP-1A
800      T=PSI1*A(JIP)
801      IF(I-1) 1120,1140,1140
802      1120  JIP2=JIP+1A
803      T=T+PSI2*A(JIP2)
804      1140  ETA=ALPHA*(1+A(JI))
805      A(JI)=A(JI)-ETA
806      A(JIP)=A(JIP)-ETA*PSI1
807      IF(I-1) 1160,1180,1180
808      1160  A(JIP2)=A(JIP2)-ETA*PSI2
809      1180  CONTINUE
810      IF(I-2) 1200,1220,1220
811      1200  JI=JI+3
812      JIP=JI+1A
813      JIP2=JIP+1A
814      ETA=ALPHA*PSI2*A(JIP2)
815      A(JI)=-ETA
816      A(JIP)=-ETA*PSI1
817      A(JIP2)=A(JIP2)-ETA*PSI2
818      1220  II=IIP+1
819      IT=IT+1
820      GO TO 60
821      1240  IF(DABS(A(N11))-DABS(A(N112))) 1300,1280,1280
822      1280  IAA(N)=0
823      IAA(N1)=2
824      I=I2
825      IF(I2) 1400,1400,20
826      1300  RR(I)=A(N1)
827      RI(I)=0.000
828      IAA(N)=1
829      IF(I1) 1400,1400,1320
830      1320  N=11

```

```

831      GO TO 20
832      1400 RETURN
833      END

834      SUBROUTINE EIGVEC(IVC, A, B, W, IROW, XR, XI, VR, VI, RCOTR, CSY1
1      RCOTI, NC, NMAX, T2, SW1, COUNT, ERR, MMM) ESY1 1
C      SUBROUTINE TO FIND THE EIGENVECTORS OF A NON-SYMMETRIC MATRIX ESY1 2
C      BY A MODIFIED WILKINSON'S INVERSE ITERATION METHOD. ESY1 3
C      CONTROL IVC CODE IS ESY1 4
C      1 FIND ONLY THE REGULAR EIGENVECTORS (A X = LAMBDA X) ESY1 5
C      2 FIND ONLY THE TRANSPOSED EIGENVECTORS (AT V = LAMBDA V) ESY1 6
C      3 FIND BOTH TYPES OF EIGENVECTORS. ESY1 7

835      DIMENSION A(7,7), B(7,7), W(7,7), XR(7), XI(7), VR(7), VI(7), IROW(7,2)
836      DOUBLE PRECISION RCOTR, RCOTI, RCOTR2, RCOTI2, TEMP, TEMP2, AMAX, C1, C2,
1      SW1, W, XR, XI, VR, VI, B, ZERR, DCERR, A

837      DOUBLE PRECISION DABS, DSIGN, DSQRT, DMAX1
838      INTEGER COUNT, COUNT2, T2 ESY1 10
839      I01=1
840      I03=3
841      ROOTR = RCOTR ESY1 11
842      RCOTI = RCOTI ESY1 12
843      N = NE ESY1 13
844      MM = MMM - 1 ESY1 14
845      N1 = N - 1 ESY1 15
846      NPI = N + 1 ESY1 16
847      IVC1 = IVC - 1 ESY1 17
848      IVC2 = IVC1 - 1 ESY1 18
849      COUNT = 1 ESY1 19
850      DO 400 I=1, N
851      W(I,1)=0.0DC
852      XR(I)=0.0DC
853      400 CONTINUE
854      CLIM = 1.0E-4 ESY1 20
855      IF(RCOTI) 1, 60, 1 ESY1 21
C      ESY1 22
C      COMPLEX EIGENVALUE. ESY1 23
C      ESY1 24
856      1 TEMP = - ROOTR - RCOTR ESY1 25
857      ISW = 2 ESY1 26
858      TEMP2=ROOTR*ROOTR+RCOTI*RCOTI
859      JJ = 30 ESY1 28
860      DO 606 I = 1, N ESY1 29
861      IF(T2) 600, 603, 600 ESY1 30
862      600 DO 502 J = 1, N ESY1 31
863      JJ = JJ + 1 ESY1 32
864      IF(JJ - 251) 502, 601, 601 ESY1 33
865      601 JJ = 1 ESY1 34
866      READ (T2) (W(LL,1), LL = 1,250) ESY1 35
867      602 B(I,J) = A(I,J)*TEMP + W(JJ,1) ESY1 36
868      GO TO 605 ESY1 37
869      603 DO 604 J = 1, N ESY1 38
870      604 B(I,J) = A(I,J)*TEMP + B(I,J) ESY1 39
871      605 B(I,1) = B(I,1) + TEMP2 ESY1 40
872      606 A(I,1) = A(I,1) - ROOTR ESY1 41
873      IF(T2 .NE. C) REWIND T2 ESY1 42
874      GO TO 700 ESY1 43
875      607 IF(ICC) 622, 608, 622 ESY1 44
C      ESY1 45
C      MATRIX SINGULAR. ESY1 46
C      ESY1 47
876      622 IF(IVC2) 623, 625, 623 ESY1 48

```

877	624	DO 624 LL = 1, N	ESY1 450
878		W(LL,2)=0.000	
879	624	XI(LL)=0.000	
880		IF(IVC1) 625, 614, 625	ESY1 510
881	625	DO 626 LL = 1, N	ESY1 520
882		W(LL,4)=0.000	
883	626	VI(LL)=0.000	
884		GO TO 511	ESY1 540
	C		ESY1 550
	C	MATRIX NOT SINGULAR.	ESY1 560
	C		ESY1 570
885	608	DO 609 LL = 1, N	ESY1 580
886		W(LL,1)=1.000	
887		W(LL,2)=1.000	
888		W(LL,3)=1.000	
889	609	W(LL,4)=1.000	
890	699	IF(IVC2) 610, 612, 610	ESY1 600
891	610	DO 611 I = 1, N	ESY1 610
892		I2 = IROW(I,2)	ESY1 620
893		XI(I2) = W(I,1)*RCOTI	ESY1 630
894		DO 611 J = 1, N	ESY1 640
895	611	XI(I2) = XI(I2) + A(I,J)*W(J,2)	ESY1 650
896		IF(IVC1) 612, 500, 612	ESY1 660
897	612	DO 613 I = 1, N	ESY1 670
898		VI(I) = W(I,3)*RCOTI	ESY1 680
899		DO 613 J = 1, N	ESY1 690
900	613	VI(I) = VI(I) + A(J,I)*W(J,4)	ESY1 700
901		GO TO 499	ESY1 710
902	615	CFRR = 0.0	ESY1 720
903		DCERR=0.000	
904		IF(IVC2) 616, 619, 616	ESY1 730
905	616	DO 618 I = 1, N	ESY1 740
906		XR(I) = -W(I,2)	ESY1 750
907		DO 617 J = 1, N	ESY1 760
908	617	XR(I) = XR(I) + A(I,J)*XI(J)	ESY1 770
909	618	XR(I) = XR(I)/RCOTI	ESY1 780
910		IF(IVC1) 619, 633, 619	ESY1 790
911	619	DO 621 I = 1, N	ESY1 800
912		VR(I) = -W(I,4)	ESY1 810
913		DO 620 J = 1, N	ESY1 820
914	620	VR(I) = VR(I) + A(J,I)*VI(J)	ESY1 830
915	621	VR(I) = VR(I)/RCOTI	ESY1 840
	C		ESY1 850
	C	SEARCH VECTORS FOR LARGEST ELEMENT AND NORMALIZE.	ESY1 860
	C		ESY1 870
916	627	AMAX=0.000	ESY1 890
917		DO 629 L = 1, N	ESY1 900
918		TEMP = VR(L)**2 + VI(L)**2	ESY1 910
919		IF(TEMP - AMAX) 629, 629, 628	ESY1 920
920	628	AMAX = TEMP	ESY1 930
921		I2 = L	ESY1 940
922	629	CONTINUE	ESY1 950
923		C1 = VR(I2)/AMAX	ESY1 960
924		C2 = -VI(I2)/AMAX	ESY1 970
925		DO 630 L = 1, N	ESY1 980
926		TEMP = VI(L)	ESY1 990
927		VI(L) = VR(L)*C2 + TEMP*C1	ESY11000
928	630	VR(L) = VR(L)*C1 - TEMP*C2	ESY11010
929		IF(COUNT .EQ. 1) GO TO 632	ESY11020
930		DO 631 LL = 1, N	
931	631	DCERR=DMAX1(DCERR,DABS(VR(LL)-W(LL,3)),DABS(VI(LL)-W(LL,4)))	

```

932      632 IF(IVC2) 633, 638, 633
933      633 AMAX=0.000
934      DO 635 L = 1, N
935          TEMP = XR(L)*2 + XI(L)*2
936          IF(TEMP - AMAX) 635, 635, 634
937      634 AMAX = TEMP
938      I2 = L
939      635 CONTINUE
940      C1 = XR(I2)/AMAX
941      C2 = -XI(I2)/AMAX
942      DO 636 L = 1, N
943          TEMP = XI(L)
944          XI(L) = XR(L)*C2 + TEMP*C1
945      636 XR(L) = XR(L)*C1 - TEMP*C2
946          IF(COUNT .EQ. 1) GO TO 646
947      DO 637 LL = 1, N
948      637 DCERR=DMAX1(DCERR,DABS(XR(LL)-W(LL,1)),DABS(XI(LL)-W(LL,2)))
          C
          C      TEST FOR CONVERGENCE.
          C
949      638 IF(COUNT .EQ. 1) GO TO 646
950      CERR=DCERR
951      IF(CERR .GE. 1.0E-4) GO TO 639
952      IF(CERR .GE. CLIM) GO TO 648
953      CLIM = CERR
954      IF(CLIM .LE. 1.0E-8) GO TO 648
955      639 IF(COUNT .GE. 15) GO TO 68
956      647 COUNT = COUNT + 1
957      IF(ROOT1) 642, 673, 642
958      642 IF(IVC2) 640, 644, 640
959      640 DO 641 LL = 1, N
960          W(LL,1) = XR(LL)
961      641 W(LL,2) = XI(LL)
962      IF(IVC1) 644, 610, 644
963      644 DO 645 LL = 1, N
964          W(LL,3) = VR(LL)
965      645 W(LL,4) = VI(LL)
966      GO TO 699
967      646 CERR = 0.0
968      DCERR=0.000
969      IF(ICC) 648, 647, 648
970      648 ERR = CERR
971      CCOUNT = COUNT
972      IF(RCOT1) 667, 668, 667
973      667 DO 649 I = 1, N
974      649 A(I,I) = A(I,I) + RCCTR
975      RETURN
976      68 PRINT 101, RCCTR, ROOT1, CERR
977      GO TO 648
          C
          C      REAL EIGENVECTORS.
          C
978      60 ISW = 1
979      DO 651 I = 1, N
980          DO 650 J = 1, N
981      650 B(I,J) = A(I,J)
982      651 B(I,I) = B(I,I) - RCCTR
983      GO TO 700
984      652 IF(ICC) 680, 685, 680
          C
          C      SINGULAR MATRIX.

```

985	C	680 IF(IVC2) 681, 683, 681	ESY11630
986		661 DO 682 L = 1, N	ESY11640
987		682 XI(L)=0.000	ESY11650
988		IF(IVC1) 683, 614, 683	ESY11670
989		683 DO 684 L = 1, N	ESY11680
990		684 VI(L)=0.000	
991		GO TO 511	ESY11700
	C		ESY11710
	C	MATRIX NOT SINGULAR.	ESY11720
	C		ESY11730
992		685 IF(IVC2) 652, 656, 653	ESY11740
993		653 DO 654 L = 1, N	ESY11750
994		654 XI(L)=1.000	
995		IF(IVC1) 656, 500, 656	ESY11770
996		656 DO 657 L = 1, N	ESY11780
997		657 VI(L)=1.000	
998		GO TO 499	ESY11800
	C		ESY11810
	C	NORMALIZE REAL VECTORS.	ESY11820
	C		ESY11830
999		655 CERR = 0.0	ESY11840
1000		DCERR=0.000	
1001		IF(IVC2) 658, 662, 658	ESY11850
1002		658 C1=0.000	
1003		C2=0.000	
1004		DO 660 L = 1, N	ESY11870
1005		TEMP=DABS(XI(L))	
1006		IF(TEMP - C1) 660, 660, 659	ESY11890
1007		659 C1 = TEMP	ESY11900
1008		C2 = XI(L)	ESY11910
1009		660 CONTINUE	ESY11920
1010		DO 661 L = 1, N	ESY11930
1011		XI(L) = XI(L)/C2	ESY11940
1012		DCERR=DMAX1(DCERR,DABS(XI(L)-XR(L)))	
1013		661 XR(L) = XI(L)	ESY11960
1014		IF(IVC1) 662, 638, 662	ESY11970
1015		662 C2=0.000	
1016		C1=0.000	
1017		DO 664 L = 1, N	ESY11990
1018		TEMP=DABS(VI(L))	
1019		IF(TEMP - C1) 664, 664, 663	ESY12010
1020		663 C1 = TEMP	ESY12020
1021		C2 = VI(L)	ESY12030
1022		664 CONTINUE	ESY12040
1023		DO 665 LL = 1, N	ESY12050
1024		VI(LL) = VI(LL)/C2	ESY12060
1025		DCERR=DMAX1(DCERR,DABS(VI(LL)-W(LL,1)))	
1026		W(LL,1)=VI(LL)	
1027		665 VR(LL)=W(LL,1)	
1028		GO TO 638	ESY12090
1029		668 IF(IVC2) 659, 671, 659	ESY12100
1030		659 DO 670 L = 1, N	ESY12110
1031		670 XI(L)=0.000	
1032		IF(IVC1) 671, 70, 671	ESY12130
1033		671 DO 672 L = 1, N	ESY12140
1034		672 VI(L)=0.000	
1035		70 RETURN	ESY12160
1036		673 IF(IVC2) 674, 502, 674	ESY12170
1037		674 DO 675 I = 1, N	ESY12180
1038		I2 = IROW(I,2)	ESY12190

1039	675	XI(I2) = XR(I)	ESY12200
1040		GO TO 500	ESY12210
	C		ESY12220
	C	BACK SUBSTITUTION SECTION.	ESY12230
	C		ESY12240
1041	499	IF(IVC2) 500, 502, 500	ESY12250
1042	500	DO 501 I = 2, N	ESY12260
1043		I1 = I - 1	ESY12270
1044		DO 501 J = 1, I1	ESY12280
1045	501	XI(I) = XI(I) - B(I,J)*XI(J)	ESY12290
1046	511	IF(IVC1) 502, 514, 502	ESY12300
1047	502	DO 510 I = 1, N	ESY12310
1048		I1 = I - 1	ESY12320
1049		IF(I1) 503, 505, 503	ESY12330
1050	503	DO 504 J = 1, I1	ESY12340
1051	504	VI(I) = VI(I) - B(J,I)*VI(J)	ESY12350
1052		IF(ICC) 505, 506, 505	ESY12360
1053	505	IF(B(I,I)) 506, 507, 506	ESY12370
1054	506	VI(I) = VI(I)/B(I,I)	ESY12380
1055		GO TO 510	ESY12390
1056	507	IF(VI(I)) 508, 509, 508	ESY12400
1057	508	VII(I)=VI(I)*1.00+15	
1058		GO TO 510	ESY12420
1059	509	VI(I)=1.000	
1060	510	CONTINUE	ESY12440
1061		IF(IVC2) 514, 525, 514	ESY12450
1062	514	DO 522 I = 1, N	ESY12460
1063		IR = NP1 - I	ESY12470
1064		IF(I - 1) 515, 517, 515	ESY12480
1065	515	I2 = IR + 1	ESY12490
1066		DO 516 J = I2, N	ESY12500
1067	516	XI(IR) = XI(IR) - B(IR,J)*XI(J)	ESY12510
1068		IF(ICC) 517, 518, 517	ESY12520
1069	517	IF(B(IR,IR)) 519, 519, 518	ESY12530
1070	518	XI(IR) = XI(IR)/B(IR,IR)	ESY12540
1071		GO TO 522	ESY12550
1072	519	IF(XI(IR)) 520, 521, 520	ESY12560
1073	520	XI(IR)=XI(IR)*1.00+15	
1074		GO TO 522	ESY12580
1075	521	XI(IR)=1.000	
1076	522	CONTINUE	ESY12600
1077		IF(IVC1) 525, 525, 525	ESY12610
1078	525	DO 526 I = 2, N	ESY12620
1079		IR = NP1 - I	ESY12630
1080		I2 = IR + 1	ESY12640
1081		DO 526 J = I2, N	ESY12650
1082	526	VI(IR) = VI(IR) - B(J,IR)*VI(J)	ESY12660
1083		DO 527 L = 1, N	ESY12670
1084		I2 = IROW(L,1)	ESY12680
1085	527	VR(I2) = VI(L)	ESY12690
1086		DO 528 L = 1, N	ESY12700
1087	528	VI(L) = VR(L)	ESY12710
1088	529	IF(RCOT1) 615, 655, 615	ESY12720
	C		ESY12730
	C	FACTOR MATRIX.	ESY12740
	C		ESY12750
1089	700	ICC = 0	ESY12760
1090		SWI=1.0072	
1091		DO 701 LL = 1, N	ESY12780
1092	701	IROW(LL,1) = LL	ESY12790
1093		DO 708 K = 1, N1	ESY12800

```

1094      AMAX=DABS(B(K,K))
1095      IMAX = K
1096      K1 = K + 1
1097      DO 702 I = K1, N
1098      IF(AMAX.GT.DABS(B(I,K))) GO TO 702
1099      AMAX=DABS(B(I,K))
1100      IMAX = I
1101      702 CONTINUE
1102      IF(AMAX.LT.SW1) SW1 = AMAX
1103      IF(AMAX.GE.1.0D-25) GO TO 723
1104      B(K,K)=0.0DC
1105      ICC = ICC + 1
1106      GO TO 708
1107      723 IF(IMAX.EQ.K) GO TO 704
1108      DO 703 J = 1, N
1109      AMAX = B(K,J)
1110      B(K,J) = B(IMAX,J)
1111      703 B(IMAX,J) = AMAX
1112      I2 = IROW(K,1)
1113      IROW(K,1) = IROW(IMAX,1)
1114      IROW(IMAX,1) = I2
1115      704 DO 707 I = K1, N
1116      IF(B(I,K)) 705, 707, 705
1117      705 B(I,K) = B(I,K)/B(K,K)
1118      DO 706 J = K1, N
1119      706 B(I,J) = B(I,J) - B(K,J)*B(I,K)
1120      707 CONTINUE
1121      708 CONTINUE
1122      AMAX=DABS(B(N,N))
1123      IF(AMAX-1.0D-25) 712,712,713
1124      712 B(N,N)=0.0DC
1125      SW1=0.0DC
1126      ICC = ICC + 1
1127      GO TO 709
1128      713 IF(AMAX.LT.SW1) SW1 = AMAX
1129      709 IF(ICC.LE.ISW) GO TO 710
1130      IF(MM) 1050,1050,1051
1131      1050 WRITE(103,102) ICC
1132      COUNT = 0
1133      RETURN
1134      1051 WRITE(103,1052) ICC
1135      710 DO 711 LL = 1, N
1136      I2 = IROW(LL,1)
1137      711 IROW(I2,2) = LL
1138      IF(ROOTI) 607, 652, 607
1139      1052 FORMAT(//23H ***** WARNING ***** , ' SUBROUTINE EIGVEC HAS
1      1 FOUND AN EIGENVALUE OF APPARENT MULTIPLICITY',
1      14, /23X, ' COMPUTATION OF EIGENVECTORS
2 GENVECTOR(S) CONTINUES AT USER'S OPTION'//)
1140      101 FORMAT(38H MORE THAN 15 LOOPS FOR EIGENVECTOR OF, 2E12.4,
2 14H DIFFERENCE OF, E12.4)
1141      102 FORMAT(16H ***** WARNING ***** , 14, 71H ZEROS ON DIAGONAL OF FACTORED
1 MATRIX, CHECK FOR MULTIPLE EIGENVALUES, /20X,
2 SUBROUTINE EIGVEC WILL NOT PERFORM COMPUTATION FOR THIS EIGENVALUE
3 FOR '//)
1142      END

```

/DATA