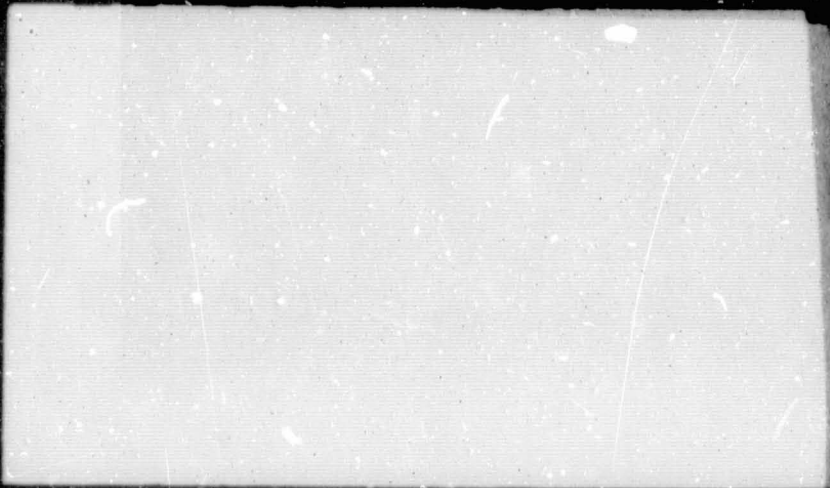




N70-20135	
(ACCESSION NUMBER)	(THRU)
<u>54</u>	<u>1</u>
(PAGES)	(CODE)
<u>CR-102455</u>	<u>08</u>
(NASA CR OR TMX OR AD NUMBER)	(CATEGORY)

Rensselaer Polytechnic Institute
Troy, New York 12181

Final Report

Contract No. NAS8-21131

Covering period Nov. 4, 1968-Nov. 3, 1969

NATIONAL AERONAUTICS AND
SPACE ADMINISTRATION

MSFC

DESCRIPTION AND USER'S GUIDE
GAIN INITIALIZATION PROGRAMS

by Derek E. McBrinn

Submitted on behalf of

Rob Roy
Professor of Systems Engineering

DESCRIPTION AND USER'S GUIDE - GAIN INITIALIZATION PROGRAMS
GRADGN and LERNGN

I. Introduction

GRADGN and LERNGN are Fortran IV digital computer programs designed to automatically determine stabilizing feedback gains for linear time-invariant systems with output feedback. Such gains are required, for example, for initialization of the SOCDES¹ design procedure.

Given the system

$$\dot{\underline{z}}(t) = \underline{A}_1 \underline{z}(t) + \underline{B}_1 \underline{u}(t) \quad (1)$$

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

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

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

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

we wish to determine a linear, time-invariant output-feedback controller

$$\underline{u}(t) = - \underline{K}_1^T \underline{y}(t) \quad (3)$$

to stabilize the system.

We consider an entirely equivalent problem, where the first NU states of the system

$$\dot{\underline{x}}(t) = \underline{A} \underline{x}(t) + \underline{B} \underline{u}(t) \quad (4)$$

are to be fed back by the controller

$$\underline{u}(t) = - \underline{K}^T \underline{x}(t) \quad (5)$$

where $\underline{K}$ has the form

$$K = \begin{bmatrix} k_{11} & k_{12} & \cdots & k_{1,NC} \\ k_{21} & k_{22} & \cdots & k_{2,NC} \\ \vdots & \vdots & \ddots & \vdots \\ k_{NU,1} & k_{NU,2} & \cdots & k_{NU,NC} \end{bmatrix} \begin{matrix} \uparrow \\ NS \\ \downarrow \end{matrix} \quad (6)$$

This yields

$$\begin{aligned} \dot{\underline{x}}(t) &= [A - BK^T] \underline{x}(t) \\ &\triangleq \hat{A} \underline{x}(t) \end{aligned} \quad (7)$$

Both GRADGN and LERNGN are capable of solving the above problem, provided they are started sufficiently close to a solution region in gain space. That is, both GRADGN and LERNGN are designed to reduce the maximum real part of the set of eigenvalues of Eq. (7). An inopportune starting point may result in the programs "stalling out" on a local minimum and failing to determine a satisfactory set of gains, even though such gains may exist.

Both programs normally are started with all gains zero. If, for a specific problem, there exists prior information which indicates a better choice of starting gains, these gains can be used instead.

In general GRADGN is more efficient than LERNGN. LERNGN, however, has some features which may allow it to produce a solution in cases where GRADGN fails. The recommended procedure is to first try GRADGN. If it fails when a solution is felt to exist, try LERNGN or attempt to determine a set of starting gains close to a stable region.

Note that both GRADGN and LERNGN are programmed only to find stabilizing gains. A trivial modification to either program would allow the degree of stability to be specified.

II GRADGN

The basis of the GRADGN program, as developed in [2], is the equation

$$\frac{\partial \lambda_i}{\partial k_{jk}} = \frac{\underline{v}_i^T [0 \dots 0 \mid b_k \mid 0 \dots 0] \underline{w}_i}{\underline{v}_i^T \underline{w}_i}$$

j^{th} column

where λ_i is an eigenvalue of A

$\underline{v}_i^T$ and $\underline{w}_i$ are the corresponding row and column eigenvectors

b_k is the k^{th} column of the matrix B

The gradient of the maximum real part of the set of eigenvalues of $\hat{A}$, with respect to the gain matrix K, is obtained from Eq. (8). As an aid to convergence the conjugate gradient [3] is then computed. For the j^{th} iteration, the conjugate gradient is defined by

$$\text{CGRAD}_j = \text{GRAD}_j + \frac{\|\text{GRAD}_j\|^2}{\|\text{GRAD}_{j-1}\|^2} \text{CGRAD}_{j-1} \quad (9)$$

where GRAD_j is the j^{th} gradient

CGRAD_j is the j^{th} conjugate gradient

$$\text{CGRAD}_1 \triangleq \text{GRAD}_1$$

Thus the conjugate gradient is a linear combination of the actual gradient and the conjugate gradient of the preceding iteration. As is indicated in [3], this combination provides a better direction of gain adjustment than the true gradient. The gains are adjusted along the negative of the conjugate gradient, thereby increasing the stability of the least stable mode at each iteration.

The program, comprising a MAIN and seven subprograms, is described below. A flow chart is given in Fig. 1 and a program listing, together with a sample problem, are given in Appendix A.

MAIN The MAIN program is used for input-output, control of the various subprograms, computation of the gradient and conjugate gradient matrices and adjustment of the gain matrix.

To run a problem the numbers of states, controls, states available for feedback, and the parameter IGAINS are read-in on the first data card using the 4I10 format. IGAINS = 0 sets the initial gain matrix to zero, while IGAINS = 1 indicates that a guess at a stable set of gains is to be read in.

The A matrix of Eq. (4) (AMAT in the program) is then read-in by rows, followed by the B matrix (BMAT in the program) and, if IGAINS = 1, the initial K matrix. The 7F10.4 format is used for all of these.

AMHT Computes $AHAT = AMAT - [BMAT] [K^T]$

VECT Converts AHAT to single-subscript notation AAAA, for use in variable-dimensional subroutines. This conversion is not necessary at some facilities, but is included for maximum utility of the program.

MSQ Computes $ASQR = [AAAA]^2$ for subsequent use in subroutine EIGVEC

HSBG Preconditions AAAA to upper Hessenberg form, for subsequent use by subroutines ATEIG.

ATEIG Determines the eigenvalues of AHAT.

MAXRT Selects from the eigenvalues of AHAT the eigenvalue having the maximum real part.

EIGVEC Computes the row and column eigenvectors of AHAT corresponding to the eigenvalue having the maximum real part. Error messages SW1, ITER and DIF indicate the success of this operation.

The program may fail in the case of a repeated value of the eigenvalue with the maximum real part, since subroutine EIGVEC may fail to determine eigenvectors for such a case.

III LERNGN

Like GRADGN, the LERNGN program is aimed at reduction of the maximum real part of the set of eigenvalues of Eq. (7). Unlike GRADGN, however, LERNGN does not depend on calculation of a gradient. Instead, each gain is adjusted individually to reduce the maximum real part of the set of eigenvalues. The adjustment entails a predetermined variation of the gain followed by an iterative procedure to determine its optimum value. The gains are adjusted in sequence, with the program "learning" better gains at each stage. The process continues until either the system is stabilized or the maximum allowable number of adjustment cycles is reached.

The program comprises a MAIN and six subprograms, several of which are also used in GRADGN. A flow chart is given in Fig.2 and a program listing with a sample problem is given in Appendix B.

Main

Used for input-output, control of the subprograms, and the predetermined variation of each gain. The data cards required to run a problem are exactly the same as in GRADGN.

AMHT, VECT, HSBG, ATEIG

Same as in GRADGN.

Function RMAX

Same as MAXRT in GRADGN.

GITER

Iteratively computes the optimum value of the gain being adjusted.

References

1. Cassidy, John F. Jr., "Optimal Control With Unavailable States",
Final Report, Vol. III, Contract No. NAS8-21131, Rensselaer
Polytechnic Institute, Nov. 1968.
2. _____, "Advanced Control System For A Saturn Booster",
Progress Report, June 4, 1969 to July 4, 1969, Contract
NAS8-21131, Rensselaer Polytechnic Institute.
3. Lasdon, L.S., S.K. Mitter and A.D. Waren, "The Conjugate Gradient
Method For Optimal Control Problems". IEEE Transactions
Vol AC-12, No. 2, April 1967.

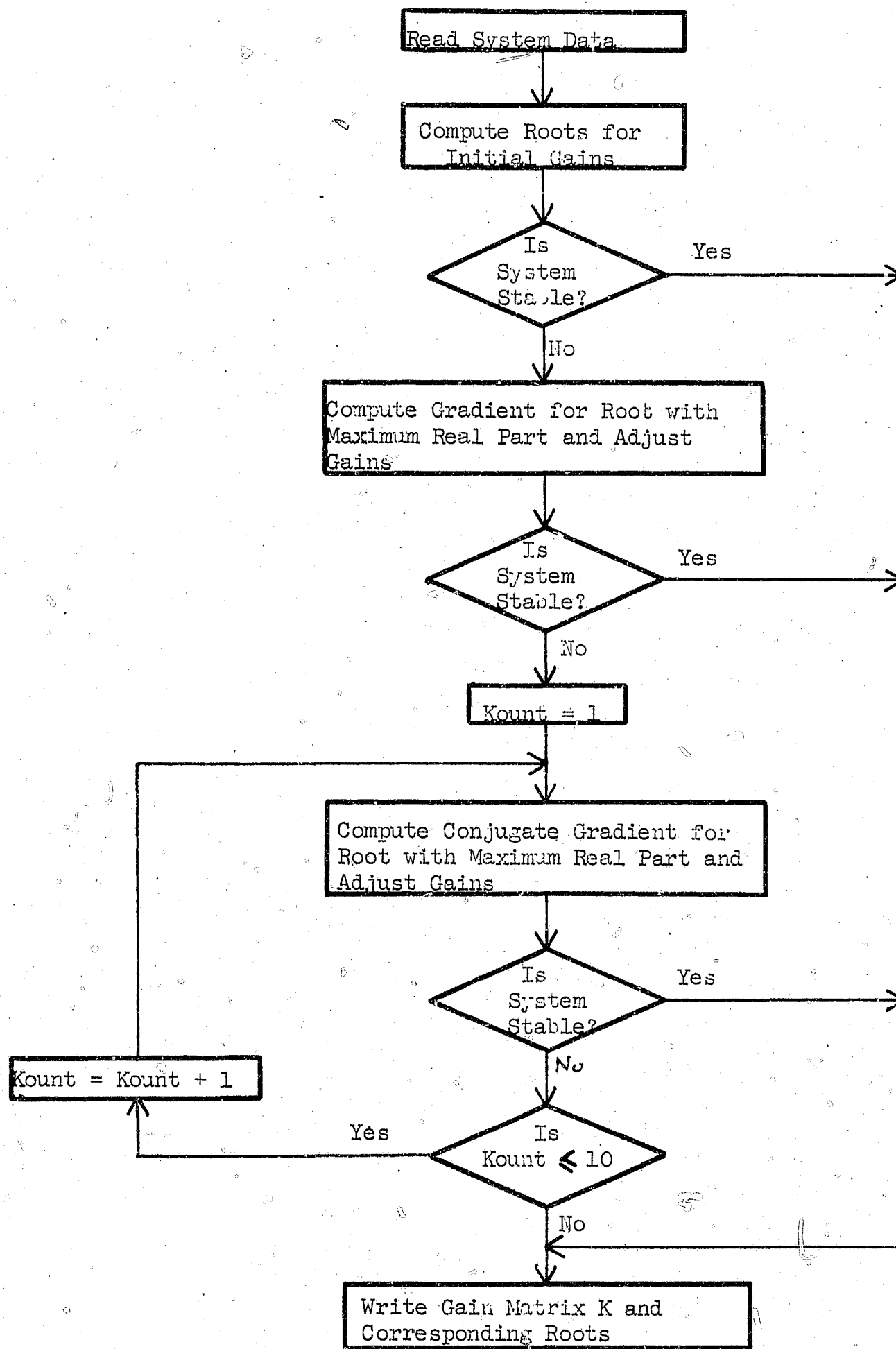


FIGURE 1 FLOW CHART FOR GRADCN

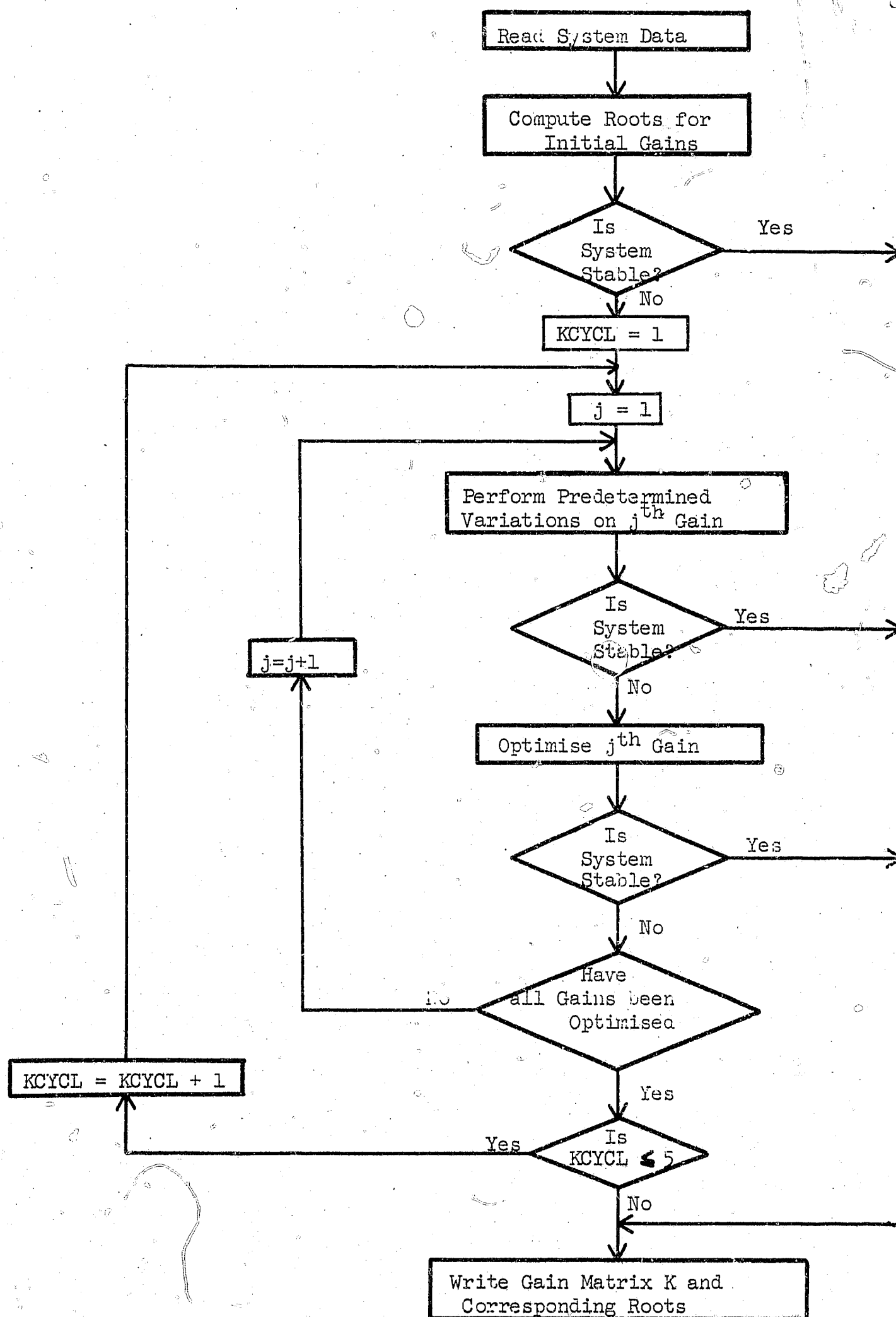


FIGURE 2 FLOW CHART FOR LERNEN

APPENDIX A

A sample problem was run to illustrate the use of GRADGN. The plant was a seven state model of the Saturn V booster, with the first two states fed back to a single controller. Complete ignorance of stabilizing gains was assumed, so the gain matrix was initialized to zero. The input cards required for the problem are shown in Fig. A-1. A program listing, together with the outputs for the sample problem, is given below.

Solution time for this problem on the IBM-360/50 computer was 10.4 sec. compared with 28.2 sec. for solution of the same problem by the LERNGN program.

/JOB

4188

MCBRINN, LINECAT=55

10.

C
C
C
C
C
C
C
C
C
C
C

PROGRAM GRADGN
 COMPUTES STABILISING OUTPUT FEEDBACK GAINS FOR LINEAR SYSTEMS
 WRITTEN BY DEREK E. MC BRINN - SYSTEMS ENGINEERING DIVISION
 RENSSELAER POLYTECHNIC INSTITUTE
 TROY, NEW YORK, 12181
 TELEPHONE 518-270-6324

2ND. SEPTEMBER 1969

COMPLETE USER'S MANUAL AVAILABLE FROM AUTHOR

```

1      DIMENSION AMAT(30,30), BMAT(30,30), AHAT(30,30), AAAA(900), ASQR(30,
      130), RR(30), RI(30), IANA(30), W(30,4), XR(30), XI(30), VR(30), VI(30),
      2 IROW(30,2), GRADR(30,30), GRADI(30,30), GRAC(30,30), CGRAD(30,30),
      3 VRN(30), VIN(30)
2      REAL K(30,30)
3      10  FORMAT (7I10)
      C
      C      IGAINS CONTRCLS INITIALIZATION OF GAIN MATRIX K
      C
4      READ (1,10) NS, NC, NU, IGAINS
5      14  FORMAT (1H1)
6      15  FORMAT (//T4, 'STATES', 4X, 'CONTRCLS', 4X, 'FEEDBACKS')
7      WRITE (3,14)
8      WRITE (3,15)
9      WRITE (3,10) NS, NC, NU
10     IA=NS
11     NS2=NS*NS
12     DO 40 I=1, NS
13     DO 40 J=1, NC
14     40  K(I,J)=0.
15     50  FORMAT (4E18.7)
16     55  FORMAT (2E20.6)
17     60  FORMAT (7F10.4)
      C
      C      SYSTEM DATA MATRICES ARE READ IN BY ROWS
      C
18     READ (1,60) ((AMAT(I,J), J=1, NS), I=1, NS)
19     70  FORMAT (//T3, 'SYSTEM MATRIX AMAT')
20     WRITE (3,70)
21     WRITE (3,50) ((AMAT(I,J), J=1, NS), I=1, NS)
22     READ (1,60) ((BMAT(I,J), J=1, NC), I=1, NS)
23     80  FORMAT (//T3, 'CONTRCL MATRIX BMAT')
24     WRITE (3,80)
25     WRITE (3,50) ((BMAT(I,J), J=1, NC), I=1, NS)
      C
      C      INITIALIZATION OF GAIN MATRIX K
      C
26     IF (IGAINS) 94, 94, 92
27     92  READ (1,60) ((K(I,J), J=1, NC), I=1, NU)
28     94  CONTINUE
29     90  FORMAT (//T3, 'GAIN MATRIX K')
30     WRITE (3,90)
31     WRITE (3,50) ((K(I,J), J=1, NC), I=1, NU)
32     CALL AMHT(AMAT, BMAT, K, NS, NC, AHAT)
33     CALL VECT(AHAT, NS, AAAA)
34     CALL MSC(AAAA, NS, ASQR)

```

C
C
C

COMPUTATION OF EIGENVALUES

```

35      CALL HSBG(NS,AAAA,IA)
36      CALL ATEIG(NS,AAAA,RR,RI,IANA,IA)
37      100 FORMAT (//T2,'RCCTS',5X,'REAL PART',11X,'IMAG PART')
38      WRITE (3,100)
39      WRITE (3,55) (RR(I),RI(I),I=1,NS)
40      CALL MAXRT(NS,RR,RI)
41      RCCTR=RR(1)
42      RCOTI=RI(1)
43      120 FORMAT (//T2,'MAXIMUM REAL PART OF RCCTS')
44      WRITE (3,120)
45      WRITE (3,50) RCCTR
C
C      CHECK FOR SYSTEM STABILITY
C
46      IF (RCCTR) 8000,190,190
47      190 CONTINUE
C
C      COMPUTATION OF EIGENVECTORS
C
48      CALL EIGVEC(3,AHAT,ASQR,h,IRCW,XR,XI,VR,VI,RCCTR,RCOTI,NS,3C,C,
        1SW1,ITER,DIF,2)
C
C      SW1 = 0 FOR AN EXACT EIGENVALUE AND NO ROUND-OFF ERROR
C      ITER = NUMBER OF ITERATIONS USED TO FIND EIGENVECTORS.
C      IF TOLERANCE IS NOT ACHIEVED, PROGRAM ACCEPTS VALUES AT ITER = 15.
C      DIF = LARGEST CHANGE IN ANY EIGENVECTOR COMPONENT AT FINAL ITER.
C
49      200 FORMAT (//T3,'EIGVEC ERROR MESSAGES')
50      210 FORMAT (T3,'SW1=',E10.4,10X,'ITER=',15,10X,'DIF=',E10.4)
51      WRITE (3,200)
52      WRITE (3,210) SW1,ITER,DIF
53      220 FORMAT (//T3,'EIGENVECTORS CORRES TO MRP EIGENVALUE')
54      230 FORMAT (T6,'RCW REAL PT',7X,'RCW IMAG PT',7X,'COL REAL PT',7X,'COL
        1IMAG PT')
55      WRITE (3,220)
56      WRITE (3,230)
57      WRITE (3,50) (VR(I),VI(I),XR(I),XI(I),I=1,NS)
C
C      NORMALISE EIGENVECTORS INNER PRODUCT
C
58      VECMGR=0.
59      VECMGI=0.
60      DO 240 I=1,NS
61      VECMGR=VECMGR+VR(I)*XR(I)-VI(I)*XI(I)
62      240 VECMGI=VECMGI+VR(I)*XI(I)+VI(I)*XR(I)
63      VECMGS=VECMGR*VECMGR+VECMGI*VECMGI
64      DO 250 I=1,NS
65      VRN(I)=(VR(I)*VECMGR+VI(I)*VECMGI)/VECMGS
66      250 VIN(I)=(VI(I)*VECMGR-VR(I)*VECMGI)/VECMGS
C
C      COMPUTE GRADIENT MATRIX
C
67      DO 300 J=1,NU
68      DO 300 L=1,NC
69      GRADR(J,L)=0.
70      GRADI(J,L)=0.
71      DO 290 I=1,NS
72      GRADR(J,L)=GRADR(J,L)-VRN(I)*BMAT(I,L)
73      290 GRADI(J,L)=GRADI(J,L)+VIN(I)*BMAT(I,L)
74      300 GRAD(J,L)=GRADR(J,L)*XR(J)+GRADI(J,L)*XI(J)

```

```

75      310  FORMAT (//T3,'GRADIENT MATRIX')
76      WRITE (3,310)
77      WRITE (3,50) ((GRAD(I,J),J=1,NC),I=1,NU)
78      GRDSC=0.
79      DO 320 I=1,NU
80      DO 320 J=1,NC
81      320  GRDSC=GRDSC+GRAD(I,J)*GRAD(I,J)
      C
      C      ADJUST GAIN MATRIX K
      C
82      DELK=-(RCCTR+.01)/GRDSC
83      DO 340 I=1,NU
84      DO 340 J=1,NC
85      340  K(I,J)=K(I,J)+DELK*GRAD(I,J)
86      WRITE (3,90)
87      WRITE (3,50) ((K(I,J),J=1,NC),I=1,NU)
      C
      C      CONJUGATE GRADIENT IS COMPUTED FOR SECOND AND SUBSEQUENT ITER'S.
      C
88      DO 350 I=1,NU
89      DO 350 J=1,NC
90      350  CGRAD(I,J)=GRAD(I,J)
91      GRDSC1=GRDSC
92      DO 7000 KCUNT=1,10
93      CALL AMHT(AMAT,BMAT,K,NS,NC,AHAT)
94      CALL VECT(AHAT,NS,AAAA)
95      CALL MSG(AAAA,NS,ASQR)
      C
      C      COMPUTATION OF EIGENVALUES
      C
96      CALL HSBG(NS,AAAA,IA)
97      CALL ATEIG(NS,AAAA,RR,RI,IANA,IA)
98      WRITE (3,100)
99      WRITE (3,55) (RR(I),RI(I),I=1,NS)
100     CALL MAXRT(NS,RR,RI)
101     RCOTR=RR(1)
102     RCOTI=RI(1)
103     WRITE (3,120)
104     WRITE (3,50) RCCTR
      C
      C      CHECK FOR SYSTEM STABILITY
      C
105     IF (RCOTR) 8000,380,380
106     380  CONTINUE
107     IF (KCUNT-10) 390,7500,7500
108     390  CONTINUE
      C
      C      COMPUATION OF EIGENVECTORS
      C
109     CALL EIGVEC(3,AHAT,ASQR,h,IRCW,XR,XI,VR,VI,RCCTR,RCOTI,NS,30,0,
110     1SW1,ITER,DIF,2)
111     WRITE (3,200)
112     WRITE (3,210) SW1,ITER,DIF
113     WRITE (3,220)
114     WRITE (3,230)
115     WRITE (3,50) (VR(I),VI(I),XR(I),XI(I),I=1,NS)
      C
      C      NORMALISE EIGENVECTORS INNER PRODUCT
      C
115     VECMGR=C.

```

```

116      VECMGI=C.
117      DO 420 I=1,NS
118      VECMGR=VECMGR+VR(I)*XR(I)-VI(I)*XI(I)
119      420 VECMGI=VECMGI+VR(I)*XI(I)+VI(I)*XR(I)
120      VECMGS=VECMGR*VECMGR+VECMGI*VECMGI
121      DO 440 I=1,NS
122      VRN(I)=(VR(I)*VECMGR+VI(I)*VECMGI)/VECMGS
123      440 VIN(I)=(VI(I)*VECMGR-VR(I)*VECMGI)/VECMGS
      C
      C      COMPUTE GRADIENT MATRIX
      C
124      DO 460 J=1,NU
125      DO 460 L=1,NC
126      GRADR(J,L)=0.
127      GRADI(J,L)=0.
128      DO 450 I=1,NS
129      GRADR(J,L)=GRADR(J,L)-VRN(I)*BMAT(I,L)
130      450 GRADI(J,L)=GRADI(J,L)+VIN(I)*BMAT(I,L)
131      460 GRAD(J,L)=GRADR(J,L)*XR(J)+GRADI(J,L)*XI(J)
132      WRITE (3,310)
133      WRITE (3,50) ((GRAD(I,J),J=1,NC),I=1,NU)
134      GRDSC=0.
135      DO 480 I=1,NU
136      DO 480 J=1,NC
137      480 GRDSC=GRDSC+GRAD(I,J)*GRAD(I,J)
      C
      C      COMPUTE CONJUGATE GRADIENT MATRIX.
      C
138      BETA=GRDSC/GRDSQ1
139      CGRSC=0.
140      DO 500 I=1,NU
141      DO 500 J=1,NC
142      CGRAD(I,J)=BETA*CGRAD(I,J)+GRAD(I,J)
143      500 CGRSC=CGRSC+CGRAD(I,J)*CGRAD(I,J)
144      GRDSQ1=GRDSC
145      510 FORMAT (//T3,'CONJUGATE GRADIENT MATRIX')
146      WRITE (3,510)
147      WRITE (3,50) ((CGRAD(I,J),J=1,NC),I=1,NU)
      C
      C      ADJUST GAIN MATRIX K
      C
148      DELK=-(RCCTR+.01)/CGRSC
149      DO 520 I=1,NU
150      DO 520 J=1,NC
151      520 K(I,J)=K(I,J)+DELK*CGRAD(I,J)
152      WRITE (3,90)
153      WRITE (3,50) ((K(I,J),J=1,NC),I=1,NU)
154      7000 CONTINUE
155      7500 WRITE (3,7600)
156      7600 FORMAT (//T1,'NO STABILISING GAINS FOUND. THIS DOES NOT IMPLY THAT
      1 NO SUCH GAINS EXIST')
157      GO TO 9000
158      8000 WRITE (3,8100)
159      8100 FORMAT (//T3,'LAST GAINS SHOWN GIVE STABLE RCCTS AS LISTED')
160      9000 CONTINUE
161      STOP
162      END

```

163

SUBROUTINE AMHT(AMAT,BMAT,K,NS,NC,AFAT)

14.

C
C
C

COMPUTES AFAT = AMAT - BMAT*K(TRANSPOSE)

164

DIMENSION AMAT(30,30),BMAT(30,30),AFAT(30,30)

165

REAL K(30,30)

166

DO 100 I=1,NS

167

DO 100 J=1,NS

168

AFAT(I,J)=AMAT(I,J)

169

DO 100 L=1,NC

170

100 AFAT(I,J)=AFAT(I,J)-BMAT(I,L)*K(J,L)

171

RETURN

172

END

```
173      SUBROUTINE VECT(AHAT,NS,AAAA)
      C
      C
      C
174      DIMENSION AHAT(30,30),AAAA(900)
175      DO 100 J=1,NS
176      DO 100 I=1,NS
177      K=(J-1)*NS+I
178      100 AAAA(K)=AHAT(I,J)
179      RETURN
180      END
```

181

SUBROUTINE MSC(AAAA,NS,ASQR)

C
C
C

COMPUTES ASQR = AAAA*AAAA

16.

182

DIMENSION AAAA(900),ASQR(30,30)

183

DO 100 I=1,NS

184

DO 100 J=1,NS

185

ASQR(I,J)=0.

186

DO 100 K=1,NS

187

K1=NS*(J-1)+K

188

K2=NS*(K-1)+I

189

100 ASQR(I,J)=ASQR(I,J)+AAAA(K1)*AAAA(K2)

190

RETURN

191

END

```

192      SUBROUTINE HSBG(N,A,IA)
      C
      C
      C
193      DOUBLE PRECISION DABS,DFLCAT,DSIGN,DBLE,DEXP,DLOG,DLOG10,DATAN
194      1,DSIN,DCCS,DSQRT,DTANH,CMOD,CMAX1,CMIN1
195      DOUBLE PRECISION S
196      DIMENSION A(900)
197      L=N
198      NIA=L-IA
199      20 IF(L-3) 360,40,40
200      40 LIA=LIA-IA
201      L1=L-1
202      L2=L1-1
203      ISUB=LIA+L
204      IPIV=ISUB-IA
205      PIV=ABS(A(IPIV))
206      IF(L-3) 90,90,50
207      50 M=IPIV-IA
208      DO 80 I=L,M,IA
209      T=ABS(A(I))
210      IF(T-PIV) 80,80,60
211      60 IPIV=I
212      PIV=T
213      80 CONTINUE
214      90 IF(PIV) 100,320,100
215      100 IF(PIV-ABS(A(ISUB))) 180,180,120
216      120 M=IPIV-L
217      DO 140 I=1,L
218      J=M+I
219      T=A(J)
220      K=LIA+I
221      A(J)=A(K)
222      140 A(K)=T
223      M=L2-M/IA
224      DO 160 I=L1,NIA,IA
225      T=A(I)
226      J=I-M
227      A(I)=A(J)
228      160 A(J)=T
229      180 DO 200 I=L,LIA,IA
230      200 A(I)=A(I)/A(ISUB)
231      J=-IA
232      DO 240 I=1,L2
233      J=J+IA
234      LJ=L+J
235      DO 220 K=1,L1
236      KJ=K+J
237      KL=K+LIA
238      220 A(KJ)=A(KJ)-A(LJ)*A(KL)
239      240 CONTINUE
240      K=-IA
241      DO 300 I=1,N
242      K=K+IA
243      LK=K+L1
244      S=A(LK)
245      LJ=L-IA
246      DO 280 J=1,L2
247      JK=K+J

```

248 LJ=LJ+IA
249 280 S=S+A(LJ)*A(JK)*1.000
250 300 A(LK)=S
251 DO 310 I=L,LIA,IA
252 310 A(I)=0.0
253 320 L=L1
254 GO TO 20
255 360 RETURN
256 END

```

257      SUBROUTINE ATEIG(M,A,RR,RI,IANA,IA)
      C
      C      COMPUTES RCTS OF UPPER HESSENBERG MATRIX A
      C
258      DOUBLE PRECISION DABS,DFLOAT,DSIGN,CBLE,CEXP,DLOG,CLGG10,DATAN
259      1,DSIN,DCCS,DSQRT,DTANH,CMOD,CMAX1,DMIN1
260      DIMENSION A(900),RR(30),RI(30),PRR(2),PRI(2),IANA(30)
261      INTEGER P,P1,C
262      E7=1.0E-8
263      E6=1.0E-6
264      E10=1.0E-10
265      DELTA=0.5
266      MAXIT=30
267      N=M
268      20 N1=N-1
269      IN=N1*IA
270      NN=IN+N
271      IF(N1) 30,1300,30
272      30 NP=N+1
273      IT=0
274      DO 40 I=1,2
275      PRR(I)=C.0
276      40 PRI(I)=C.0
277      PAN=C.0
278      PAN1=0.0
279      R=0.0
280      S=0.0
281      N2=N1-1
282      IN1=IN-IA
283      NN1=IN1+N
284      N1N=IN+N1
285      N1N1=IN1+N1
286      60 T=A(N1N1)-A(NN)
287      U=T*T
288      V=4.0*A(N1N)*A(NN1)
289      IF(ABS(V)-L*E7) 100,100,65
290      65 T=U+V
291      IF(ABS(T)-AMAX1(U,ABS(V))*E6) 67,67,68
292      67 T=0.0
293      68 U=(A(N1N1)+A(NN))/2.0
294      V=SQRT(ABS(T))/2.0
295      IF(T)140,70,70
296      70 IF(U) 80,75,75
297      75 RR(N1)=L+V
298      RR(N)=L-V
299      GO TO 130
300      80 RR(N1)=L-V
301      RR(N)=L+V
302      GO TO 130
303      100 IF(T)120,110,110
304      110 RR(N1)=A(N1N1)
305      RR(N)=A(NN)
306      GO TO 130
307      120 RR(N1)=A(NN)
308      RR(N)=A(N1N1)
309      130 RI(N)=0.0
310      RI(N1)=C.0
311      GO TO 160
312      140 RR(N1)=L
      RR(N)=U

```

```

313      RI(N1)=V
314      RI(N)=-V
315      160 IF(N2)1280,1280,180
316      180 N1N2=N1N1-IA
317      RMOD=RR(N1)*RR(N1)+RI(N1)*RI(N1)
318      EPS=E10*SQRT(RMOD)
319      IF(ABS(A(N1N2))-EPS)1280,1280,240
320      240 IF(ABS(A(NN1))-E10*ABS(A(NN))) 1300,1300,250
321      250 IF(ABS(PAN1-A(N1N2))-ABS(A(N1N2))*E6) 1240,1240,260
322      260 IF(ABS(PAN-A(NN1))-ABS(A(NN1))*E6)1240,1240,300
323      300 IF(IT-MAXIT) 320,1240,1240
324      320 J=1
325      DO 360 I=1,2
326      K=NP-I
327      IF(ABS(RR(K)-PRR(I))+ABS(RI(K)-PRI(I))-DELTA*(ABS(RR(K))
1      +ABS(RI(K)))) 340,360,360
328      340 J=J+I
329      360 CONTINUE
330      GO TO (440,460,460,480),J
331      440 R=0.
332      S=0.
333      GO TO 500
334      460 J=N+2-J
335      R=RR(J)*RR(J)
336      S=RR(J)+RR(J)
337      GO TO 500
338      480 R=RR(N)*RR(N1)-RI(N)*RI(N1)
339      S=RR(N)+RR(N1)
340      500 PAN=A(NN1)
341      PAN1=A(N1N2)
342      DO 520 I=1,2
343      K=NP-I
344      PRR(I)=RR(K)
345      520 PRI(I)=RI(K)
346      P=N2
347      IF(N-3)600,600,525
348      525 IPI=N1N2
349      DO 580 J=2,N2
350      IPI=IPI-IA-1
351      IF(ABS(A(IPI))-EPS) 600,600,530
352      530 IPIP=IPI+IA
353      IPIP2=IPIP+IA
354      D=A(IPIP)*(A(IPIP)-S)+A(IPIP2)*A(IPIP+1)+R
355      IF(D)540,560,540
356      540 IF(ABS(A(IPI)*A(IPIP+1))*(ABS(A(IPIP)+A(IPIP2+1)-S)+ABS(A(IPIP2+1
1      ) -ABS(C)*EPS) 620,620,560
357      560 P=N1-J
358      580 CONTINUE
359      600 Q=P
360      GO TO 680
361      620 P1=P-1
362      Q=P1
363      IF(P1-1)680,680,650
364      650 DO 660 I=2,P1
365      IPI=IPI-IA-1
366      IF(ABS(A(IPI))-EPS)680,680,660
367      660 Q=Q-1
368      680 II=(P-1)*IA+P
369      DO 1220 I=P,N1
370      III=II-IA

```

```

371      IIP=II+IA
372      IF(I-P)720,700,720
373 700   IPI=II+1
374      IPIP=IIP+1
375      G1=A(II)*(A(II)-S)+A(IIP)*A(IPI)+R
376      G2=A(IPI)*(A(IPIP)+A(II)-S)
377      G3=A(IPI)*A(IPIP+1)
378      A(IPI+1)=C.0
379      GO TC 780
380 720   G1=A(III)
381      G2=A(III+1)
382      IF(I-N2)740,740,760
383 740   G3=A(III+2)
384      GO TC 780
385 760   G3=0.0
386 780   CAP=SQRT(G1*G1+G2*G2+G3*G3)
387      IF(CAP)800,860,800
388 800   IF(G1)820,840,840
389 820   CAP=-CAP
390 840   T=G1+CAP
391      PSI1=G2/T
392      PSI2=G3/T
393      ALPHA=2.0/(1.0+PSI1*PSI1+PSI2*PSI2)
394      GO TC 880
395 860   ALPHA=2.0
396      PSI1=0.0
397      PSI2=0.0
398 880   IF(I-C)900,960,900
399 900   IF(I-P)920,940,920
400 920   A(III)=-CAP
401      GO TC 960
402 940   A(III)=-A(III)
403 960   IJ=II
404      DO 1040 J=I,N
405      T=PSI1*A(IJ+1)
406      IF(I-N1)980,1000,1000
407 980   IP2J=IJ+2
408      T=T+PSI2*A(IP2J)
409 1000  ETA=ALPHA*(T+A(IJ))
410      A(IJ)=A(IJ)-ETA
411      A(IJ+1)=A(IJ+1)-PSI1*ETA
412      IF(I-N1)1020,1040,1040
413 1020  A(IP2J)=A(IP2J)-PSI2*ETA
414 1040  IJ=IJ+IA
415      IF(I-N1)1080,1060,1060
416 1060  K=N
417      GO TC 1100
418 1080  K=I+2
419 1100  IP=IIP-I
420      DO 1180 J=C,K
421      JIP=IP+J
422      JI=JIP-IA
423      T=PSI1*A(JIP)
424      IF(I-N1)1120,1140,1140
425 1120  JIP2=JIP+IA
426      T=T+PSI2*A(JIP2)
427 1140  ETA=ALPHA*(T+A(JI))
428      A(JI)=A(JI)-ETA
429      A(JIP)=A(JIP)-ETA*PSI1
430      IF(I-N1)1160,1180,1180

```

```
431 1160 A(JIP2)=A(JIP2)-ETA*PSI2
432 1180 CONTINUE
433 IF(I-N2)1200,1220,1220
434 1200 JI=II+3
435 JIP=JI+IA
436 JIP2=JIP+IA
437 ETA=ALPHA*PSI2*A(JIP2)
438 A(JI)=-ETA
439 A(JIP)=-ETA*PSI1
440 A(JIP2)=A(JIP2)-ETA*PSI2
441 1220 II=IIP+1
442 IT=IT+1
443 GO TC 6C
444 1240 IF(ABS(A(NN1))-ABS(A(NIN2))) 1300,1280,1280
445 1280 IANA(N)=0
446 IANA(N1)=2
447 N=N2
448 IF(N2)1400,1400,20
449 1300 RR(N)=A(NN)
450 RI(N)=0.0
451 IANA(N)=1
452 IF(N1)1400,1400,1320
453 1320 N=N1
454 GO TC 2C
455 1400 RETURN
456 END
```

457

SUBROUTINE MAXRT(NS,RR,RI)

23.

C
C
C

COMPUTES RCCT HAVING MAXIMUM REAL PART

458

DIMENSION RR(30),RI(30)

459

DO 100 I=2,NS

460

IF (RR(1)-RR(I)) 50,100,100

461

50

RI(1)=RI(I)

462

RR(1)=RR(I)

463

100

CONTINUE

464

RETURN

465

END

```

466 SUBROUTINE EIGVEC(IVC, A, B, W, IRCW, XR, XI, VR, VI, RCCTRE,
1 RCOTIE, NE, NMAX, T2, SW1, COUNT, ERR, NMM)
C SUBROUTINE TO FIND THE EIGENVECTORS OF A NON-SYMMETRIC MATRIX
C BY A MODIFIED WILKINSON'S INVERSE ITERATION METHOD.
C CONTROL IVC CODE IS
C 1 FIND ONLY THE REGULAR EIGENVECTORS (A X = LAMBDA X)
C 2 FIND ONLY THE TRANSPOSED EIGENVECTORS (AT V = LAMBDA V)
C 3 FIND BOTH TYPES OF EIGENVECTORS.
467 DIMENSION A(30,30), B(30,30), W(30,4), XR(30), XI(30), VR(30), VI(30),
1 IROW(30,2)
468 INTEGER COLAT, COUNT, T2
469 IOL=1
470 IO3=3
471 RCCTR = RCCTRE
472 RCOTI = RCOTIE
473 N = NE
474 NM = NMM - 1
475 N1 = N - 1
476 NP1 = N + 1
477 IVC1 = IVC - 1
478 IVC2 = IVC1 - 1
479 COUNT = 1
480 DO 400 I=1,N
481 W(I,1)=0.0E0
482 XR(I)=0.0E0
483 400 CONTINUE
484 CLIM = 1.0E-4
485 IF(RCOTI) 1, 60, 1

C
C COMPLEX EIGENVALUE.
C
486 1 TEMP = - RCCTR - RCCTR
487 ISW = 2
488 TEMP2=RCCTR*RCCTR+RCOTI*RCOTI
489 JJ = 300
490 DO 606 I = 1, N
491 IF(T2) 600, 603, 600
492 600 DO 602 J = 1, N
493 JJ = JJ + 1
494 IF(JJ - 251) 602, 601, 601
495 601 JJ = 1
496 READ (T2) (W(LL,1), LL = 1,250)
497 602 B(I,J) = A(I,J)*TEMP + W(JJ,1)
498 GO TO 605
499 603 DO 604 J = 1, N
500 604 B(I,J) = A(I,J)*TEMP + B(I,J)
501 605 B(I,I) = B(I,I) + TEMP2
502 606 A(I,I) = A(I,I) - RCCTR
503 IF(T2 .NE. 0) REWIND T2
504 GO TO 700
505 607 IF(ICC) 622, 608, 622

C
C MATRIX SINGULAR.
C
506 622 IF(IVC2) 623, 625, 623
507 623 DO 624 LL = 1, N
508 W(LL,2)=0.0
509 624 XI(LL)=0.0
510 IF(IVC1) 625, 514, 625
511 625 DO 626 LL = 1, N

```

```

512      W(LL,4)=0.0
513      626 V(LL)=0.0
514      GO TC 511

```

```

C
C
C

```

MATRIX NOT SINGULAR.

```

515      608 DO 609 LL = 1, N
516          W(LL,1)=1.0
517          W(LL,2)=1.0
518          W(LL,3)=1.0
519      609 W(LL,4)=1.0
520      699 IF(IVC2) 610, 612, 610
521      610 DO 611 I = 1, N
522          I2 = IRCH(I,2)
523          XI(I2) = W(I,1)*RCCTI
524          DO 611 J = 1, N
525      611 XI(I2) = XI(I2) + A(I,J)*W(J,2)
526          IF(IVC1) 612, 500, 612
527      612 DO 613 I = 1, N
528          VI(I) = W(I,3)*RCCTI
529          DO 613 J = 1, N
530      613 VI(I) = VI(I) + A(J,I)*W(J,4)
531          GO TC 499
532      615 CERR = 0.0
533          IF(IVC2) 616, 619, 616
534      616 DO 618 I = 1, N
535          XR(I) = -W(I,2)
536          DO 617 J = 1, N
537      617 XR(I) = XR(I) + A(I,J)*XI(J)
538      618 XR(I) = XR(I)/RCCTI
539          IF(IVC1) 619, 633, 619
540      619 DO 621 I = 1, N
541          VR(I) = -W(I,4)
542          DO 620 J = 1, N
543      620 VR(I) = VR(I) + A(J,I)*VI(J)
544      621 VR(I) = VR(I)/RCCTI

```

```

C
C
C

```

SEARCH VECTORS FOR LARGEST ELEMENT AND NORMALIZE.

```

545      627 AMAX = 0.0
546          DO 629 L = 1, N
547          TEMP = VR(L)**2 + VI(L)**2
548          IF(TEMP - AMAX) 629, 629, 628
549      628 AMAX = TEMP
550          I2 = L
551      629 CONTINUE
552          C1 = VR(I2)/AMAX
553          C2 = -VI(I2)/AMAX
554          DO 630 L = 1, N
555          TEMP = VI(L)
556          VI(L) = VR(L)*C2 + TEMP*C1
557      630 VR(L) = VR(L)*C1 - TEMP*C2
558          IF(CCUNT .EQ. 1) GO TC 632
559          DO 631 LL = 1, N
560      631 CERR = AMAX1(CERR, ABS(VR(LL) - W(LL,3)), ABS(VI(LL) - W(
561      632 IF(IVC2) 633, 638, 633
562      633 AMAX = C.0
563          DO 635 L = 1, N
564          TEMP = XR(L)**2 + XI(L)**2
565          IF(TEMP - AMAX) 635, 635, 634

```

```

566      634 AMAX = TEMP
567        I2 = L
568      635 CONTINUE
569        C1 = XR(I2)/AMAX
570        C2 = -XI(I2)/AMAX
571        DO 636 L = 1, N
572          TEMP = XI(L)
573          XI(L) = XR(L)*C2 + TEMP*C1
574      636 XR(L) = XR(L)*C1 - TEMP*C2
575          IF(CCUNT .EQ. 1) GC TC 646
576          DO 637 LL = 1, N
577      637 CERR = AMAX1(CERR, ABS(XR(LL) - W(LL,1)), ABS(XI(LL) - W(LL,2)))

```

C
C
C

TEST FOR CONVERGENCE.

```

578      638 IF(CCUNT .EQ. 1) GC TC 646
579          IF(CERR .GE. 1.0E-4) GC TC 639
580          IF(CERR .GE. CLIM) GC TC 648
581          CLIM = CERR
582          IF(CLIM .LE. 1.0E-8) GC TC 648
583      639 IF(CCUNT .GE. 15) GO TC 68
584      647 COUNT = CCUNT + 1
585          IF(RCOTI) 642, 673, 642
586          642 IF(IVC2) 640, 644, 640
587          640 DO 641 LL = 1, N
588              W(LL,1) = XR(LL)
589          641 W(LL,2) = XI(LL)
590              IF(IVC1) 644, 610, 644
591          644 DO 645 LL = 1, N
592              W(LL,3) = VR(LL)
593          645 W(LL,4) = VI(LL)
594              GO TC 659
595          646 CERR = 0.0
596              IF(ICC) 648, 647, 648
597          648 ERR = CERR
598              COUNT = CCUNT
599              IF(RCOTI) 667, 668, 667
600          667 DO 649 I = 1, N
601          649 A(I,I) = A(I,I) + RCCTR
602              RETURN
603          68 PRINT 101, RCCTR, ROCTI, CERR
604          GO TC 648

```

C
C
C

REAL EIGENVECTORS.

```

605      60 ISW = 1
606          DO 651 I = 1, N
607          DO 650 J = 1, N
608      650 B(I,J) = A(I,J)
609      651 B(I,I) = B(I,I) - RCCTR
610          GO TC 700
611      652 IF(ICC) 680, 685, 680

```

C
C
C

SINGULAR MATRIX.

```

612      680 IF(IVC2) 681, 683, 681
613      681 DO 682 L = 1, N
614      682 XI(L) = 0.0
615          IF(IVC1) 683, 514, 683
616      683 DO 684 L = 1, N

```

```

617      684 VI(L) = 0.0
618      GO TC 511
      C
      C      MATRIX NCT SINGULAR.
      C
619      685 IF(IVC2) 653, 656, 653
620      653 DO 654 L = 1, N
621      654 XI(L) = 1.0
622      IF(IVC1) 656, 500, 656
623      656 DO 657 L = 1, N
624      657 VI(L) = 1.0
625      GO TC 499
      C
      C      NORMALIZE REAL VECTORS.
      C
626      655 CERR = 0.0
627      IF(IVC2) 658, 662, 658
628      658 C1=0.0
629      C2=0.0
630      DO 660 L = 1, N
631      TEMP = ABS(XI(L))
632      IF(TEMP - C1) 660, 660, 659
633      659 C1 = TEMP
634      C2 = XI(L)
635      660 CONTINUE
636      DO 661 L = 1, N
637      XI(L) = XI(L)/C2
638      CERR = AMAX1(CERR, ABS(XI(L) - XR(L)))
639      661 XR(L) = XI(L)
640      IF(IVC1) 662, 638, 662
641      662 C2=0.0
642      C1=0.0
643      DO 664 L = 1, N
644      TEMP = ABS(VI(L))
645      IF(TEMP - C1) 664, 664, 663
646      663 C1 = TEMP
647      C2 = VI(L)
648      664 CONTINUE
649      DO 665 LL = 1, N
650      VI(LL) = VI(LL)/C2
651      CERR = AMAX1(CERR, ABS(VI(LL) - W(LL,1)))
652      W(LL,1)=VI(LL)
653      665 VR(LL)=W(LL,1)
654      GO TC 638
655      668 IF(IVC2) 669, 671, 669
656      669 DO 670 L = 1, N
657      670 XI(L) = 0.0
658      IF(IVC1) 671, 70, 671
659      671 DO 672 L = 1, N
660      672 VI(L) = 0.0
661      70 RETURN
662      673 IF(IVC2) 674, 502, 674
663      674 DO 675 I = 1, N
664      I2 = IRCW(I,2)
665      675 XI(I2) = XR(I)
666      GO TC 500
      C
      C      BACK SUBSTITUTION SECTION.
      C
667      499 IF(IVC2) 500, 502, 500

```

```

668 500 DO 501 I = 2, N
669     I1 = I - 1
670     DO 501 J = 1, I1
671 501 XI(I) = XI(I) - B(I,J)*XI(J)
672 511 IF(IVC1) 502, 514, 502
673 502 DO 510 I = 1, N
674     I1 = I - 1
675     IF(I1) 503, 505, 503
676 503 DO 504 J = 1, I1
677 504 VI(I) = VI(I) - B(J,I)*VI(J)
678     IF(ICC) 505, 506, 505
679 505 IF(B(I,I)) 506, 507, 506
680 506 VI(I) = VI(I)/B(I,I)
681     GO TC 510
682 507 IF(VI(I)) 508, 509, 508
683 508 VI(I) = VI(I)*1.0E+15
684     GO TC 510
685 509 VI(I) = 1.0
686 510 CONTINUE
687     IF(IVC2) 514, 525, 514
688 514 DO 522 I = 1, N
689     IR = NP1 - I
690     IF(I - 1) 515, 517, 515
691 515 I2 = IR + 1
692     DO 516 J = I2, N
693 516 XI(IR) = XI(IR) - B(IR,J)*XI(J)
694     IF(ICC) 517, 518, 517
695 517 IF(B(IR,IR)) 518, 519, 518
696 518 XI(IR) = XI(IR)/B(IR,IR)
697     GO TC 522
698 519 IF(XI(IR)) 520, 521, 520
699 520 XI(IR) = XI(IR)*1.0E+15
700     GO TC 522
701 521 XI(IR) = 1.0
702 522 CONTINUE
703     IF(IVC1) 525, 529, 525
704 525 DO 526 I = 2, N
705     IR = NP1 - I
706     I2 = IR + 1
707     DO 526 J = I2, N
708 526 VI(IR) = VI(IR) - B(J,IR)*VI(J)
709     DO 527 L = 1, N
710     I2 = IRCW(L,1)
711 527 VR(I2) = VI(L)
712     DO 528 L = 1, N
713 528 VI(L) = VR(L)
714 529 IF(RCOT1) 615, 655, 615

```

C
C
C
FACTER MATRIX.

```

715 700 ICC = 0
716     SW1=1.0E72
717     DO 701 LL = 1, N
718 701 IROW(LL,1) = LL
719     DO 708 K = 1, N1
720     AMAX = ABS(B(K,K))
721     IMAX = K
722     K1 = K + 1
723     DO 702 I = K1, N
724     IF(AMAX .GT. ABS(B(I,K))) GC TC 702

```

```

725      AMAX = ABS(B(I,K))
726      IMAX = I
727 702 CONTINUE
728      IF(AMAX .LT. SW1) SW1 = AMAX
729      IF(AMAX .GE. 1.0E-25) GC TO 723
730      B(K,K) = 0.0
731      ICC = ICC + 1
732      GO TC 708
733 723 IF(IMAX .EQ. K) GO TC 704
734      DO 703 J = 1, N
735      AMAX = B(K,J)
736      B(K,J) = B(IMAX,J)
737 703 B(IMAX,J) = AMAX
738      I2 = IRCW(K,1)
739      IROW(K,1) = IRCW(IMAX,1)
740      IROW(IMAX,1) = I2
741 704 DO 707 I = K1, N
742      IF(B(I,K)) 705, 707, 705
743 705 B(I,K) = B(I,K)/B(K,K)
744      DO 706 J = K1, N
745 706 B(I,J) = B(I,J) - B(K,J)*B(I,K)
746 707 CONTINUE
747 708 CONTINUE
748      AMAX = ABS(B(N,N))
749      IF(AMAX - 1.0E-25) 712, 712, 713
750 712 B(N,N)=0.0
751      SW1=0.0
752      ICC = ICC + 1
753      GO TC 709
754 713 IF(AMAX .LT. SW1) SW1 = AMAX
755 709 IF(ICC .LE. ISW) GC TO 710
756      IF(MP) 1050,1050,1051
757 1051 WRITE(IC3,102) ICC
758      COUNT = 0
759      RETURN
760 1050 WRITE(IC3,1052) ICC
761 710 DO 711 LL = 1, N
762      I2 = IRCW(LL,1)
763 711 IROW(I2,2) = LL
764      IF(RCOTI) 607, 652, 607
765 1052 FORMAT(///23H ***** WARNING ***** , ' SUBROUTINE EIGVEC H
1FOUND AN EIGENVALUE OF APPARENT MULTIPLICITY',
1 I4,/23X,' COMPTATION OF
2GENVECTOR(S) CONTINUES AT USER S OPTION'//)
766 101 FORMAT(38HMORE THAN 15 LCOPS FOR EIGENVECTOR CF,2E12.4,
2 14H DIFFERENCE CF,E12.4)
767 102 FORMAT(16H*****WARNING**** , I4, 71H ZERCS ON DIAGONAL CF FACTOR
1 MATRIX. CHECK FOR MULTIPLE EIGENVALUES./20X,
2° SUBROUTINE EIGVEC WILL NCT PERFORM COMPTATION FOR THIS EIGEN
3TOR '//)
768      END

```

/DATA

STATES CONTROLS FEEDBACKS
7 1 2

SYSTEM MATRIX AMAT

0.000000E 00	0.100000E 01	0.000000E 00	0.000000E 00
0.000000E 00	0.000000E 00	0.000000E 00	0.000000E 00
0.000000E 00	0.2030470E 00	-0.6534950E 00	-0.1955000E-02
0.2558020E 01	0.000000E 00	-0.1366150E-01	0.100000E 01
-0.4068250E-01	0.1998600E-03	-0.1463000E-01	-0.3338200E-01
0.000000E 00	0.000000E 00	0.000000E 00	0.000000E 00
0.000000E 00	0.100000E 01	0.000000E 00	0.000000E 00
0.000000E 00	0.000000E 00	0.000000E 00	-0.4466811E 02
-0.1336680E 00	0.2546100E 03	0.000000E 00	0.000000E 00
0.000000E 00	0.000000E 00	0.000000E 00	0.000000E 00
0.000000E 00	0.100000E 01	0.000000E 00	0.000000E 00
0.000000E 00	0.000000E 00	0.000000E 00	-0.500000E 02
-0.100000E 02			

CONTROL MATRIX BMAT

0.000000E 00	0.000000E 00	0.000000E 00	0.000000E 00
0.000000E 00	0.000000E 00	0.500000E 02	

GAIN MATRIX K

0.000000E 00	0.000000E 00
--------------	--------------

ROOTS	REAL PART	IMAG PART
	-0.477798E 00	0.000000E 00
	0.423402E 00	0.000000E 00
	0.137118E-01	0.000000E 00
	-0.668340E-01	0.668309E 01
	-0.668340E-01	-0.668309E 01
	-0.500000E 01	0.500000E 01
	-0.500000E 01	-0.500000E 01

MAXIMUM REAL PART OF ROOTS
0.4234025E 00

EIGVEC ERROR MESSAGES

SW1=0.8345E-06

ITER= 4

CIF=0.3576E-06

EIGENVECTORS CORRESPONDING TO MRP EIGENVALUE

ROW	REAL PT	ROW	IMAG PT	COL	REAL PT	COL	IMAG PT
-0.1411699E-01		0.000000E 00		0.100000E 01		0.000000E 00	
0.100000E 01		0.000000E 00		0.4234034E 00		0.000000E 00	
0.4375202E 00		0.000000E 00		0.8829014E 00		0.000000E 00	
0.2059637E-03		0.000000E 00		-0.2525228E-26		0.000000E 00	
-0.1463000E-01		0.000000E 00		0.8821868E-26		0.000000E 00	
-0.2263338E 00		0.000000E 00		-0.1349601E-27		0.000000E 00	
-0.2171400E-01		0.000000E 00		-0.4707256E-27		0.000000E 00	

GRADIENT MATRIX
0.1364676E 01 0.5778083E 00

GAIN MATRIX K
-0.2693076E 00 -0.1140257E 00

ROOTS	REAL PART	IMAG PART
	-0.108505E 00	0.651177E 01
	-0.108505E 00	-0.651177E 01
	-0.492104E 01	0.511454E 01
	-0.492104E 01	-0.511454E 01
	0.130264E-01	0.357715E 00
	0.130264E-01	-0.357715E 00
	-0.141324E 00	0.000000E 00

MAXIMUM REAL PART OF ROOTS
0.1302638E-01

EIGVEC ERROR MESSAGES
SW1=0.2023E-04 ITER= 6 DIF=0.6199E-05

EIGENVECTORS CORRES TO MRP EIGENVALUE

ROW REAL PT	ROW IMAG PT	CCL REAL PT	CCL IMAG PT
0.6533355E-01	0.49011871E 00	0.6386257E 00	-0.4902583E-01
0.9999998E 00	0.1192093E-06	0.2584410E-01	0.2278044E 00
0.8332038E-01	-0.5551251E 00	0.6149467E 00	0.6074160E-01
0.1014182E-02	-0.1339428E-01	0.9999993E 00	0.2384186E-01
-0.1473719E-01	-0.6705523E-05	0.1301199E-01	0.3577101E 00
-0.2389640E 00	0.1174525E-01	0.1749420E 00	0.2243631E-03
-0.2379218E-01	0.2042998E-02	0.2205972E-02	0.6258488E-01

GRADIENT MATRIX
0.2796646E 00 0.5744416E 00

CONJUGATE GRADIENT MATRIX
0.5333089E 00 0.6818354E 00

GAIN MATRIX K
-0.2856960E 00 -0.1349784E 00

ROOTS	REAL PART	IMAG PART
	-0.107712E 00	0.648022E 01
	-0.107712E 00	-0.648022E 01
	-0.491035E 01	0.514299E 01
	-0.491035E 01	-0.514299E 01
	-0.328955E-02	0.381773E 00
	-0.328955E-02	-0.381773E 00
	-0.131366E 00	0.000000E 00

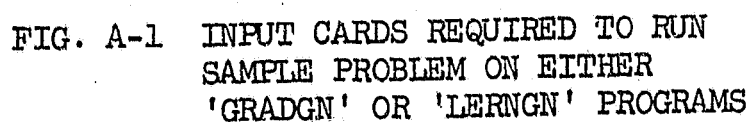
MAXIMUM REAL PART OF RCCTS
-0.3289552E-02

32.

LAST GAINS SHOWN GIVE STABLE RCCTS AS LISTED

COMPILE TIME= 34.22 SEC, EXECUTION TIME= 10.43 SEC, OBJECT CODE= 37944 BY

33.



APPENDIX B

The use of LERNGN is illustrated by the same sample problem as was used for GRADGN (see Appendix A). The input cards are unchanged, and are shown in Fig. A-1. We note again that LERNGN took 28.2 sec. to solve this problem, compared with 10.4 sec for GRADGN. A program listing and the outputs for the sample problem follow.

```

/JCB          4188          MCBRIAN,LINECNT=55
C
C  PROGRAM LERNGA
C  COMPUTES STABILISING OUTPUT FEEDBACK GAINS FOR LINEAR SYSTEMS
C  WRITTEN BY DEREK E. MC BRIAN - SYSTEMS ENGINEERING DIVISION
C                                     RENSSELAER POLYTECHNIC INSTITUT
C                                     TRCY, NEW YORK, 12181
C                                     TELEPHONE 518-270-6324
C
C  2ND. SEPTEMBER 1969
C  COMPLETE USER'S MANUAL AVAILABLE FROM AUTHOR
C
1  DIMENSION AMAT(30,30),BMAT(30,30),GV(4),RM(4),RR(30),RI(30),
2  1AHAT(30,30),AAAA(900),IANA(30)
3  REAL K(30,30)
3  10  FORMAT (7I10)
C
C  IGAINS CONTROLS INITIALIZATION OF GAIN MATRIX K
C
4  READ (1,10) NS,NC,NU,IGAINS
5  14  FORMAT (1H1)
6  15  FORMAT (//T4,'STATES',4X,'CONTROLS',4X,'FEEDBACKS')
7  WRITE (3,14)
8  WRITE (3,15)
9  WRITE (3,10) NS,NC,NU
10  IA=NS
11  NS2=NS*NS
12  50  FORMAT (4E18.7)
13  55  FORMAT (2E20.6)
14  60  FORMAT (7F10.4)
C
C  SYSTEM DATA MATRICES ARE READ IN BY rows
C
15  READ (1,60) ((AMAT(I,J), J=1,NS), I=1,NS)
16  70  FORMAT (//T3,'SYSTEM MATRIX AMAT')
17  WRITE (3,70)
18  WRITE (3,50) ((AMAT(I,J), J=1,NS), I=1,NS)
19  READ (1,60) ((BMAT(I,J), J=1,NC), I=1,NS)
20  80  FORMAT (//T3,'CONTROL MATRIX BMAT')
21  WRITE (3,80)
22  WRITE (3,50) ((BMAT(I,J), J=1,NC), I=1,NS)
C
C  INITIALIZATION OF GAIN MATRIX K
C
23  DO 40 I=1,NS
24  DO 40 J=1,NC
25  40  K(I,J)=0.
26  IF (IGAINS) 94,94,92
27  92  READ (1,60) ((K(I,J), J=1,NC), I=1,NU)
28  94  CONTINUE
29  90  FORMAT (//T3,'GAIN MATRIX K')
30  WRITE (3,90)
31  WRITE (3,50) ((K(I,J), J=1,NC), I=1,NU)
32  CALL AMHT(AMAT,BMAT,K,NS,NC,AHAT)
33  CALL VECT(AHAT,NS,AAAA)
C
C  COMPUTATION OF EIGENVALUES
C
34  CALL HSBG(NS,AAAA,IA)
35  CALL ATEIG(NS,AAAA,RR,RI,IANA,IA)
36  100 FORMAT (//T2,'RCCTS',5X,'REAL PART',11X,'IMAG PART')

```

```

37      WRITE (3,100)
38      WRITE (3,55) (RR(I),RI(I),I=1,NS)
      C
      C
      C      COMPUTATION OF RCCT HAVING MAXIMUM REAL PART
39      RTM=RMAX(NS,RR)
40      120 FORMAT (//T2,'MAXIMUM REAL PART OF RCOTS')
41      WRITE (3,120)
42      WRITE (3,50) RTM
43      DO 3000 KCYCL=1,5
44      DO 3000 K1=1,NU
45      DO 3000 K2=1,NC
      C
      C
      C      CHECK FOR SYSTEM STABILITY
46      IF (RTM) 8000,200,200
      C
      C
      C      PERFORM PREDETERMINED VARIATIONS OF GAIN
47      200 RM(3)=RTM
48      GV(3)=K(K1,K2)
49      K(K1,K2)=K(K1,K2)+1.
50      WRITE (3,90)
51      WRITE (3,50) ((K(I,J),J=1,NC),I=1,NU)
52      CALL AMHT(AMAT,BMAT,K,NS,NC,AMAT)
53      CALL VECT(AHAT,NS,AAAA)
54      CALL HSBG(NS,AAAA,IA)
55      CALL ATEIG(NS,AAAA,RR,RI,IANA,IA)
56      WRITE (3,100)
57      WRITE (3,55) (RR(I),RI(I),I=1,NS)
58      RTM=RMAX(NS,RR)
59      WRITE (3,120)
60      WRITE (3,50) RTM
61      IF (RTM) 8000,400,400
62      400 RM(4)=RTM
63      GV(4)=K(K1,K2)
64      K(K1,K2)=K(K1,K2)-1.5
65      WRITE (3,90)
66      WRITE (3,50) ((K(I,J),J=1,NC),I=1,NU)
67      CALL AMHT(AMAT,BMAT,K,NS,NC,AMAT)
68      CALL VECT(AHAT,NS,AAAA)
69      CALL HSBG(NS,AAAA,IA)
70      CALL ATEIG(NS,AAAA,RR,RI,IANA,IA)
71      WRITE (3,100)
72      WRITE (3,55) (RR(I),RI(I),I=1,NS)
73      RTM=RMAX(NS,RR)
74      WRITE (3,120)
75      WRITE (3,50) RTM
76      IF (RTM) 8000,600,600
77      600 RM(2)=RTM
78      GV(2)=K(K1,K2)
79      K(K1,K2)=K(K1,K2)-0.5
80      WRITE (3,90)
81      WRITE (3,50) ((K(I,J),J=1,NC),I=1,NU)
82      CALL AMHT(AMAT,BMAT,K,NS,NC,AMAT)
83      CALL VECT(AHAT,NS,AAAA)
84      CALL HSBG(NS,AAAA,IA)
85      CALL ATEIG(NS,AAAA,RR,RI,IANA,IA)
86      WRITE (3,100)
87      WRITE (3,55) (RR(I),RI(I),I=1,NS)

```

```

88      RTM=RMAX(NS,RR)
89      WRITE (3,120)
90      WRITE (3,50) RTM
91      IF (RTM) 8000,800,800
92      800 RM(1)=RTM
93      GV(1)=K(K1,K2)

C
C      OPTIMISE GAIN BEING VARIED
C

94      DO 2000 KITER=1,6
95      CALL GITER(GV,RP,NNG)
96      IF (NNG-5) 1000,3000,3000
97      1000 K(K1,K2)=GV(NNG)
98      WRITE (3,90)
99      WRITE (3,50) ((K(I,J),J=1,NC),I=1,NU)
100     CALL AMPT(AMAT,BMAT,K,NS,NC,AHAT)
101     CALL VECT(AHAT,NS,AAAA)
102     CALL HSBG(NS,AAAA,IA)
103     CALL ATEIG(NS,AAAA,RR,RI,IANA,IA)
104     WRITE (3,100)
105     WRITE (3,55) (RR(I),RI(I),I=1,NS)
106     RTM=RMAX(NS,RR)
107     WRITE (3,120)
108     WRITE (3,50) RTM

C
C      CHECK FOR SYSTEM STABILITY
C

109     IF (RTM) 8000,1100,1100
110     1100 RM(NNG)=RTM
111     2000 CONTINUE
112     3000 CONTINUE
113     7500 WRITE (3,7600)
114     7600 FORMAT (//T1,'NO STABILISING GAINS FOUND. THIS DOES NOT IMPLY TH
          1 NO SUCH GAINS EXIST')
115     GO TO 9000
116     8000 WRITE (3,8100)
117     8100 FORMAT (//T3,'ABOVE GAINS GIVE A STABLE SYSTEM WITH LISTED RCCTS
118     9000 CONTINUE
119     STOP
120     END

```

```
121      SUBROUTINE AMHT(AMAT,BMAT,K,NS,NC,AHAT)
      C
      C
      C
122      DIMENSION AMAT(30,30),BMAT(30,30),AHAT(30,30)
123      REAL K(30,30)
124      DO 100 I=1,NS
125      DO 100 J=1,NS
126      AHAT(I,J)=AMAT(I,J)
127      DO 100 L=1,NC
128      100 AHAT(I,J)=AHAT(I,J)-BMAT(I,L)*K(J,L)
129      RETURN
130      END
```

```

131      SUBROUTINE VECT(AHAT,NS,AAAA)
      C
      C
      C
132      DIMENSION AHAT(30,30),AAAA(900)
133      DO 100 J=1,NS
134      DO 100 I=1,NS
135      K=(J-1)*NS+I
136      100 AAAA(K)=AHAT(I,J)
137      RETURN
138      END

```

```

139      SUBROUTINE FSBG(N,A,IA)
      C
      C
      C
140      DOUBLE PRECISION DABS,DFLOAT,DSIGN,CBLE,CEXP,CLOG,CLCGIC,DATAN
1      ,DSIN,DCCS,DSQRT,DTANH,DMCC,CMAX1,CMINI
141      DOUBLE PRECISION S
142      DIMENSION A(900)
143      L=N
144      NIA=L*IA
145      LIA=NIA-IA
146      20 IF(L-3) 360,40,40
147      40 LIA=LIA-IA
148      L1=L-1
149      L2=L1-1
150      ISUB=LIA+L
151      IPIV=ISUB-IA
152      PIV=ABS(A(IPIV))
153      IF(L-3) 90,90,50
154      50 M=IPIV-IA
155      DO 80 I=L,M,IA
156      T=ABS(A(I))
157      IF(T-PIV) 80,80,60
158      60 IPIV=I
159      PIV=T
160      80 CONTINUE
161      90 IF(PIV) 100,320,100
162      100 IF(PIV-ABS(A(ISUB))) 180,180,120
163      120 M=IPIV-L
164      DO 140 I=1,L
165      J=M+I
166      T=A(J)
167      K=LIA+I
168      A(J)=A(K)
169      140 A(K)=T
170      M=L2-M/IA
171      DO 160 I=L1,NIA,IA
172      T=A(I)
173      J=I-M
174      A(I)=A(J)
175      160 A(J)=T
176      180 DO 200 I=L,LIA,IA
177      200 A(I)=A(I)/A(ISUB)
178      J=-IA
179      DO 240 I=1,L2
180      J=J+IA
181      LJ=L+J
182      DO 220 K=1,L1
183      KJ=K+J
184      KL=K+LIA
185      220 A(KJ)=A(KJ)-A(LJ)*A(KL)
186      240 CONTINUE
187      K=-IA
188      DO 300 I=1,N
189      K=K+IA
190      LK=K+L1
191      S=A(LK)
192      LJ=L-IA
193      DO 280 J=1,L2
194      JK=K+J

```

195 LJ=LJ+IA
196 280 S=S+A(LJ)*A(JK)*1.000
197 300 A(LK)=S
198 DO 310 I=L,LIA,IA
199 310 A(I)=0.0
200 320 L=L1
201 GO TC 20
202 360 RETURN
203 END

42.

```

204      SUBROUTINE ATEIG(M,A,RR,RI,IANA,IA)
      C
      C      COMPUTES RCTS OF UPPER HESSENBERG MATRIX A
      C
205      DOUBLE PRECISION DABS,DFLOAT,DSIGN,DBLE,DEXP,DLOG,DLOG10,DATAN
206      1,DSIN,DCCS,DSQRT,CTANH,DMOD,CMAX1,DMIN1
207      DIMENSION A(900),RR(30),RI(30),PRR(2),PRI(2),IANA(30)
208      INTEGER P,P1,C
209      E7=1.0E-8
210      E6=1.0E-6
211      E10=1.0E-10
212      DELTA=0.5
213      MAXIT=30
214      N=M
215      20 N1=N-1
216      IN=N1*IA
217      NN=IN+N
218      IF(N1) 30,1300,30
219      30 NP=N+1
220      IT=0
221      DO 40 I=1,2
222      PRR(I)=0.0
223      40 PRI(I)=0.0
224      PAN=0.0
225      PAN1=0.0
226      R=0.0
227      S=0.0
228      N2=N1-1
229      IN1=IN-IA
230      NN1=IN1+N
231      N1N=IN+N1
232      N1N1=IN1+N1
233      60 T=A(N1N1)-A(NN)
234      U=T*T
235      V=4.0*A(N1N)*A(NN1)
236      IF(ABS(V)-U*E7) 100,100,65
237      65 T=U+V
238      IF(ABS(T)-AMAX1(U,ABS(V))*E6) 67,67,68
239      67 T=0.0
240      68 U=(A(N1N1)+A(NN))/2.0
241      V=SQRT(ABS(T))/2.0
242      IF(T)140,70,7C
243      70 IF(U) 80,75,75
244      75 RR(N1)=L+V
245      RR(N)=U-V
246      GO TC 130
247      80 RR(N1)=L-V
248      RR(N)=U+V
249      GO TC 130
250      100 IF(T)120,110,110
251      110 RR(N1)=A(N1N1)
252      RR(N)=A(NN)
253      GO TC 130
254      120 RR(N1)=A(NN)
255      RR(N)=A(N1N1)
256      130 RI(N)=0.0
257      RI(N1)=0.0
258      GO TC 160
259      140 RR(N1)=L
      RR(N)=U

```

```

260      RI(N1)=V
261      RI(N)=-V
262 160 IF(N2)1280,1280,180
263 180 N1N2=N1N1-IA
264      RMOD=RR(N1)*RR(N1)+RI(N1)*RI(N1)
265      EPS=E10*SQRT(RMOD)
266      IF(ABS(A(N1N2))-EPS)1280,1280,240
267 240 IF(ABS(A(NN1))-E10*ABS(A(NN))) 1300,1300,250
268 250 IF(ABS(PAN1-A(N1N2))-ABS(A(N1N2))*E6) 1240,1240,260
269 260 IF(ABS(PAN-A(NN1))-ABS(A(NN1))*E6)1240,1240,300
270 300 IF(IT-MAXIT) 320,1240,1240
271 320 J=1
272      DO 360 I=1,2
273      K=NP-I
274      IF(ABS(RR(K)-PRR(I))+ABS(RI(K)-PRI(I))-DELTA*(ABS(RR(K))
1      +ABS(RI(K)))) 340,360,360
275 340 J=J+I
276 360 CONTINUE
277      GO TC (440,460,460,480),J
278 440 R=0.0
279      S=0.0
280      GO TC 500
281 460 J=N+2-J
282      R=RR(J)*RR(J)
283      S=RR(J)+RR(J)
284      GO TC 500
285 480 R=RR(N)*RR(N1)-RI(N)*RI(N1)
286      S=RR(N)+RR(N1)
287 500 PAN=A(NN1)
288      PAN1=A(N1N2)
289      DO 520 I=1,2
290      K=NP-I
291      PRR(I)=RR(K)
292 520 PRI(I)=RI(K)
293      P=N2
294      IF(N-3)600,600,525
295 525 IPI=N1N2
296      DO 580 J=2,N2
297      IPI=IPI-IA-1
298      IF(ABS(A(IPI))-EPS) 600,600,530
299 530 IPIP=IPI+IA
300      IPIP2=IPIP+IA
301      D=A(IPIP)*(A(IPIP)-S)+A(IPIP2)*A(IPIP+1)+R
302      IF(D)540,560,540
303 540 IF(ABS(A(IPI)*A(IPIP+1))*(ABS(A(IPIP)+A(IPIP2+1)-S)+ABS(A(IPIP,
1  )) -ABS(D)*EPS) 620,620,560
304 560 P=N1-J
305 580 CONTINUE
306 600 Q=P
307      GO TC 680
308 620 P1=P-1
309      Q=P1
310      IF(P1-1)680,680,650
311 650 DO 660 I=2,P1
312      IPI=IPI-IA-1
313      IF(ABS(A(IPI))-EPS)680,680,660
314 660 Q=Q-1
315 680 II=(P-1)*IA+P
316      DO 1220 I=P,N1
317      III=II-IA

```

44.

```

318      IIP=II+IA
319      IF(I-P)720,700,720
320      700 IPI=II+1
321          IPIP=IIP+1
322      G1=A(II)*(A(II)-S)+A(IIP)*A(IPI)+R
323      G2=A(IPI)*(A(IPIP)+A(II)-S)
324      G3=A(IPI)*A(IPIP+1)
325      A(IPI+1)=0.0
326      GO TC 780
327      720 G1=A(III)
328          G2=A(III+1)
329          IF(I-N2)740,740,760
330      740 G3=A(III+2)
331          GO TC 780
332      760 G3=0.0
333      780 CAP=SQRT(G1*G1+G2*G2+G3*G3)
334          IF(CAP)800,860,800
335      800 IF(G1)820,840,840
336      820 CAP=-CAP
337      840 T=G1+CAP
338          PSI1=G2/T
339          PSI2=G3/T
340          ALPHA=2.0/(1.0+PSI1*PSI1+PSI2*PSI2)
341          GO TC 880
342      860 ALPHA=2.0
343          PSI1=0.0
344          PSI2=0.0
345      880 IF(I-Q)900,960,900
346      900 IF(I-P)920,940,920
347      920 A(III)=-CAP
348          GO TC 960
349      940 A(III)=-A(III)
350      960 IJ=II
351          DO 1040 J=I,N
352          T=PSI1*A(IJ+1)
353          IF(I-N1)980,1000,1000
354      980 IP2J=IJ+2
355          T=T+PSI2*A(IP2J)
356      1000 ETA=ALPHA*(T+A(IJ))
357          A(IJ)=A(IJ)-ETA
358          A(IJ+1)=A(IJ+1)-PSI1*ETA
359          IF(I-N1)1020,1040,1040
360      1020 A(IP2J)=A(IP2J)-PSI2*ETA
361      1040 IJ=IJ+1A
362          IF(I-N1)1080,1060,1060
363      1060 K=N
364          GO TC 1100
365      1080 K=I+2
366      1100 IP=IIP-I
367          DO 1180 J=C,K
368          JIP=IP+J
369          JI=JIP-IA
370          T=PSI1*A(JIP)
371          IF(I-N1)1120,1140,1140
372      1120 JIP2=JIP+IA
373          T=T+PSI2*A(JIP2)
374      1140 ETA=ALPHA*(T+A(JI))
375          A(JI)=A(JI)-ETA
376          A(JIP)=A(JIP)-ETA*PSI1
377          IF(I-N1)1160,1180,1180

```

```
378 1160 A(JIP2)=A(JIP2)-ETA*PSI2
379 1180 CONTINUE
380 IF(I-N2)1200,1220,1220
381 1200 JI=JI+3
382 JIP=JI+IA
383 JIP2=JIP+IA
384 ETA=ALPHA*PSI2*A(JIP2)
385 A(JI)=-ETA
386 A(JIP)=-ETA*PSI1
387 A(JIP2)=A(JIP2)-ETA*PSI2
388 1220 II=IIP+1
389 IT=IT+1
390 GO TO 60
391 1240 IF(ABS(A(NN1))-ABS(A(NIN2))) 1300,1280,1280
392 1280 IANA(N)=0
393 IANA(N1)=2
394 N=N2
395 IF(N2)1400,1400,20
396 1300 RR(N)=A(NN)
397 RI(N)=0.0
398 IANA(N)=1
399 IF(N1)1400,1400,1320
400 1320 N=N1
401 GO TO 20
402 1400 RETURN
403 END
```

404

FUNCTION RMAX(NS,RR)

46.

C
C
C

COMPUTES RCCT HAVING MAXIMUM REAL PART

405

DIMENSION RR(30)

406

DO 100 I=2,NS

407

IF (RR(1)-RR(I)) 50,100,100

408

50 RR(1)=RR(I)

409

100 CONTINUE

410

RMAX=RR(1)

411

RETURN

412

END

```

413      SUBROUTINE GITER(GV,RM,NNG)
      C
      C
      C
414      DIMENSION GV(4),RM(4)
415      IF (RM(1)-RM(2)) 100,200,300
416      100 IF (RM(1)-RM(3)) 120,120,160
417      120 IF (RM(1)-RM(4)) 500,500,600
418      160 IF (RM(3)-RM(4)) 500,500,600
419      200 IF (RM(1)-RM(3)) 220,240,260
420      220 IF (RM(1)-RM(4)) 500,600,600
421      240 IF (RM(1)-RM(4)) 500,700,600
422      260 IF (RM(3)-RM(4)) 500,500,600
423      300 IF (RM(2)-RM(3)) 320,320,360
424      320 IF (RM(2)-RM(4)) 800,600,600
425      360 IF (RM(3)-RM(4)) 900,900,600
426      500 NNG=1
427      DO 510 I=1,3
428      GV(5-I)=GV(4-I)
429      510 RM(5-I)=RM(4-I)
430      GV(1)=2.*GV(2)
431      RETURN
432      600 NNG=4
433      DO 610 I=1,3
434      GV(I)=GV(I+1)
435      610 RM(I)=RM(I+1)
436      GV(4)=2.*GV(3)
437      RETURN
438      700 NNG=5
439      RETURN
440      800 GV(4)=GV(3)
441      RM(4)=RM(3)
442      IF (RM(1)-RM(3)) 810,810,860
443      810 GVT=.8*GV(1)+.2*GV(3)
444      IF (GV(2)-GVT) 820,820,830
445      820 NNG=3
446      GV(3)=GVT
447      RETURN
448      830 NNG=2
449      GV(3)=GV(2)
450      RM(3)=RM(2)
451      GV(2)=GVT
452      RETURN
453      860 GVT=.2*GV(1)+.8*GV(3)
454      IF (GV(2)-GVT) 870,870,880
455      870 NNG=3
456      GV(3)=GVT
457      RETURN
458      880 NNG=2
459      GV(3)=GV(2)
460      RM(3)=RM(2)
461      GV(2)=GVT
462      RETURN
463      900 GV(1)=GV(2)
464      RM(1)=RM(2)
465      IF (GV(2)-GV(4)) 910,910,960
466      910 GVT=.8*GV(2)+.2*GV(4)
467      IF (GV(3)-GVT) 920,920,930
468      920 NNG=3
469      GV(2)=GV(3)

```

470 RM(2)=RM(3)
471 GV(3)=GVT
472 RETURN
473 930 NNG=2
474 GV(2)=GVT
475 RETURN
476 960 GVT=.2*GV(2)+.8*GV(4)
477 IF (GV(3)-GVT) 970,970,980
478 970 NNG=3
479 GV(2)=GV(3)
480 RM(2)=RM(3)
481 GV(3)=GVT
482 RETURN
483 980 NNG=2
484 GV(2)=GVT
485 RETURN
486 END

/DATA

STATES CONTROLS FEEDBACKS
7 1 2

SYSTEM MATRIX AMAT

0.0000000E 00	0.1000000E 01	0.0000000E 00	0.0000000E 00
0.0000000E 00	0.0000000E 00	0.0000000E 00	0.0000000E 00
0.0000000E 00	0.2030470E 00	-0.6534950E 00	-0.1955000E-02
0.2558020E 01	0.0000000E 00	-0.1366150E-01	0.1000000E 01
-0.4068250E-01	0.1998600E-03	-0.1463000E-01	-0.3338200E-01
0.0000000E 00	0.0000000E 00	0.0000000E 00	0.0000000E 00
0.0000000E 00	0.1000000E 01	0.0000000E 00	0.0000000E 00
0.0000000E 00	0.0000000E 00	0.0000000E 00	-0.4466811E 02
-0.1336680E 00	0.2546100E 03	0.0000000E 00	0.0000000E 00
0.0000000E 00	0.0000000E 00	0.0000000E 00	0.0000000E 00
0.0000000E 00	0.1000000E 01	0.0000000E 00	0.0000000E 00
0.0000000E 00	0.0000000E 00	0.0000000E 00	-0.5000000E 02
-0.1000000E 02			

CONTROL MATRIX BMAT

0.0000000E 00	0.0000000E 00	0.0000000E 00	0.0000000E 00
0.0000000E 00	0.0000000E 00	0.5000000E 02	

GAIN MATRIX K

0.0000000E 00	0.0000000E 00
---------------	---------------

ROOTS

REAL PART	IMAG PART
-0.477798E 00	0.000000E 00
0.423402E 00	0.000000E 00
0.137118E-01	0.000000E 00
-0.668340E-01	0.668309E 01
-0.668340E-01	-0.668309E 01
-0.500000E 01	0.500000E 01
-0.500000E 01	-0.500000E 01

MAXIMUM REAL PART OF ROOTS

0.4234025E 00

GAIN MATRIX K

0.1000000E 01	0.0000000E 00
---------------	---------------

ROOTS

REAL PART	IMAG PART
0.132216E 00	0.670811E 01
0.132216E 00	-0.670811E 01
-0.509800E 01	0.517533E 01
-0.509800E 01	-0.517533E 01
-0.124851E 01	0.000000E 00
0.104275E 01	0.000000E 00
-0.370012E-01	0.000000E 00

MAXIMUM REAL PART OF RCCTS
0.1042751E 01

GAIN MATRIX K
-0.500000E 00 0.000000E 00

ROOTS	REAL PART	IMAG PART
	-0.174675E 00	0.667893E 01
	-0.174675E 00	-0.667893E 01
	-0.495482E 01	0.490275E 01
	-0.495482E 01	-0.490275E 01
	0.800452E-01	0.632677E 00
	0.800452E-01	-0.632677E 00
	-0.753986E-01	0.000000E 00

MAXIMUM REAL PART OF RCCTS
0.8004516E-01

GAIN MATRIX K
-0.100000E 01 0.000000E 00

ROOTS	REAL PART	IMAG PART
	-0.287612E 00	0.668237E 01
	-0.287612E 00	-0.668237E 01
	-0.491385E 01	0.479901E 01
	-0.491385E 01	-0.479901E 01
	0.143153E 00	0.101065E 01
	0.143153E 00	-0.101065E 01
	-0.576860E-01	0.000000E 00

MAXIMUM REAL PART OF RCCTS
0.1431528E 00

GAIN MATRIX K
-0.800000E 00 0.000000E 00

ROOTS	REAL PART	IMAG PART
	-0.241874E 00	0.667997E 01
	-0.241874E 00	-0.667997E 01
	-0.492960E 01	0.484130E 01
	-0.492960E 01	-0.484130E 01
	0.115094E 00	0.877612E 00
	0.115094E 00	-0.877612E 00
	-0.614523E-01	0.000000E 00

MAXIMUM REAL PART OF RCCTS
0.1150939E 00

GAIN MATRIX K
-0.640000E 00 0.000000E 00

ROOTS	REAL PART	IMAG PART
	-0.205817E 00	0.667905E 01
	-0.205817E 00	-0.667905E 01
	-0.494289E 01	0.487430E 01
	-0.494289E 01	-0.487430E 01
	0.949065E-01	0.756081E 00
	0.949065E-01	-0.756081E 00
	-0.667472E-01	0.000000E 00

MAXIMUM REAL PART OF RCCTS
0.9490651E-01

GAIN MATRIX K
-0.5120000E 00 0.0000000E 00

ROOTS	REAL PART	IMAG PART
	-0.177361E 00	0.667887E 01
	-0.177361E 00	-0.667887E 01
	-0.495378E 01	0.490034E 01
	-0.495378E 01	-0.490034E 01
	0.811626E-01	0.644128E 00
	0.811626E-01	-0.644128E 00
	-0.743992E-01	0.000000E 00

MAXIMUM REAL PART OF RCCTS
0.8116257E-01

GAIN MATRIX K
-0.4096000E 00 0.0000000E 00

ROOTS	REAL PART	IMAG PART
	-0.154808E 00	0.667915E 01
	-0.154808E 00	-0.667915E 01
	-0.496271E 01	0.492085E 01
	-0.496271E 01	-0.492085E 01
	0.733345E-01	0.539701E 00
	0.733345E-01	-0.539701E 00
	-0.859719E-01	0.000000E 00

MAXIMUM REAL PART OF RCCTS
0.7333446E-01

GAIN MATRIX K
-0.4000000E 00 0.0000000E 00

ROOTS	REAL PART	IMAG PART
	-0.152725E 00	0.667923E 01
	-0.152725E 00	-0.667923E 01
	-0.496355E 01	0.492278E 01

-0.496355E 01	-0.492278E 01
0.728598E-01	0.528975E 00
0.728598E-01	-0.528975E 00
-0.875382E-01	0.000000E 00

MAXIMUM REAL PART OF RCCTS
0.7285982E-01

GAIN MATRIX K
-0.3276799E 00 0.0000000E 00

ROOTS	REAL PART	IMAG PART
	-0.136940E 00	0.667955E 01
	-0.136940E 00	-0.667955E 01
	-0.496999E 01	0.493684E 01
	-0.496999E 01	-0.493684E 01
	0.719894E-01	0.441165E 00
	0.719894E-01	-0.441165E 00
	-0.104508E 00	0.000000E 00

MAXIMUM REAL PART OF RCCTS
0.7198936E-01

GAIN MATRIX K
-0.3276799E 00 0.1000000E 01

ROOTS	REAL PART	IMAG PART
	0.332208E-01	0.783429E 01
	0.332208E-01	-0.783429E 01
	-0.559947E 01	0.384622E 01
	-0.559947E 01	-0.384622E 01
	0.810661E 00	0.000000E 00
	0.233089E 00	0.000000E 00
	-0.854441E-01	0.000000E 00

MAXIMUM REAL PART OF RCCTS
0.8106608E 00

GAIN MATRIX K
-0.3276799E 00 -0.5000000E 00

ROOTS	REAL PART	IMAG PART
	-0.482469E 01	0.571536E 01
	-0.482469E 01	-0.571536E 01
	0.541153E-01	0.591901E 01
	0.541153E-01	-0.591901E 01
	-0.248026E 00	0.329629E 00
	-0.248026E 00	-0.329629E 00
	-0.137090E 00	0.000000E 00

MAXIMUM REAL PART OF RCCTS
0.5411530E-01

GAIN MATRIX K
-0.3276799E 00 -0.1000000E 01

ROOTS	REAL PART	IMAG PART
-0.494302E 01	0.645551E 01	
-0.494302E 01	-0.645551E 01	
0.542241E 00	0.530754E 01	
0.542241E 00	-0.530754E 01	
-0.114563E 01	0.000000E 00	
-0.113634E 00	0.911074E-01	
-0.113634E 00	-0.911074E-01	

MAXIMUM REAL PART OF RCCTS
0.5422406E 00

GAIN MATRIX K
-0.3276799E 00 -0.2000000E 00

ROOTS	REAL PART	IMAG PART
-0.100922E 00	0.637914E 01	
-0.100922E 00	-0.637914E 01	
-0.488015E 01	0.523478E 01	
-0.488015E 01	-0.523478E 01	
-0.498866E-01	0.438476E 00	
-0.498866E-01	-0.438476E 00	
-0.112543E 00	0.000000E 00	

MAXIMUM REAL PART OF RCCTS
-0.4988664E-01

ABOVE GAINS GIVE A STABLE SYSTEM WITH LISTED RCCTS

COMPILE TIME= 19.36 SEC, EXECUTION TIME= 28.20 SEC, OBJECT CCDE= 2307: