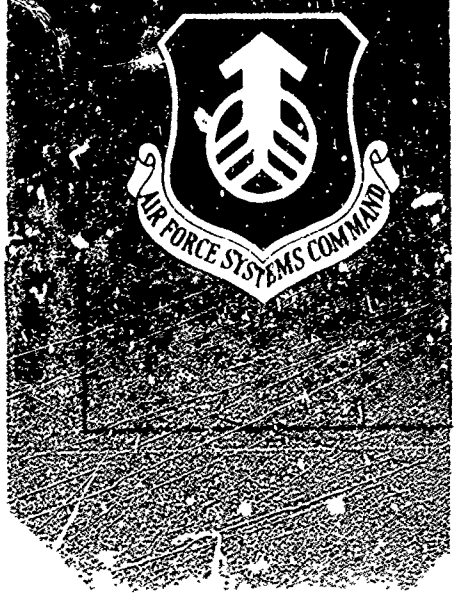


FRANK J. SEILER RESEARCH LABORATORY

SRL-TR-72-0004

MAY 1972

D D C
RECEIVED
AUG 8 1972
C

AD 746922

AN ENGINEER'S GUIDE TO BUILDING NONLINEAR FILTERS

VOLUME II

Richard S Bucy
Calvin Hecht
Capt Kenneth D Senne

SEE AD 746921

APPROVED FOR PUBLIC RELEASE;
DISTRIBUTION UNLIMITED.

PROJECT 7904

AIR FORCE SYSTEMS COMMAND
UNITED STATES AIR FORCE

230

Additional Appendix A.

A Two-Dimensional Point-Mass Program for the Passive Receiver Problem

This program was written in a format which is easily adaptable to an arbitrary two-dimensional problem. The program is set up to produce data for the movie of Chapter VI . Contour maps are printed to provide information regarding the density shapes and data is written on magnetic tape.

The program was compiled on one of the Kirtland AFB CDC 6600's using the RUN compiler. The random number generator used is similar to but inferior to the one described in Chapter IV.

Programmer: K.D. Senne

```
      PROGRAM  KNLF(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,TAPE7)          7MAR71     1
C        KIRTLAND AIR FORCE BASE VERSION                                  KFNLF      3
C        27 FEBRUARY 1971    CDC 6600                                     KFNLF      4
C*****************************************************************      KFNLF      5
C*                                                               **      KFNLF      6
C*                     GENERALIZED  TWO-DIMENSIONAL              **      KFNLF      7
C*                     DISCRETE NONLINEAR PREDICTOR              **      KFNLF      8
C*                                                               **      KFNLF      9
C*                       CAPTAIN KENNETH D. SENNE                **      KFNLF     10
C*                  FRANK J. SEILER RESEARCH LABORATORY          **      KFNLF     11
C*                       AIR FORCE SYSTEMS COMMAND               **      KFNLF     12
C*                    U.S. AIR FORCE ACADEMY, CO 80840           **      KFNLF     13
C*                          FEBRUARY 1971                        **      KFNLF     14
C*                                                               **      KFNLF     15
C*****************************************************************      KFNLF     16
000002        REAL  JC(31), EJ(991), N(991), H1(991), H2(991), H3(991)     7MAR71     2
000002        INTEGER  IX1(31), IX2(31)                                    7MAR71     3
000002        DIMENSION  XL(2), PL(3), XL1(2), PL1(3)                      KFNLF     19
000002        DIMENSION  EG(991), XP(2), XT(2), PLI(3)                     19MAR71    1
000002        REAL  NORMAL                                                 19MAR71    2
000002        COMMON /CM1/M, NSJ, MMAX, A11, A12, A22, NPTS, NPQ, NP, NQ, SCANX  KFNLF  20
000002        COMMON  / CM2/ YM2(2), RLAMN2(2), RUN2(4), YM1(2), RLAMN1(2),      KFNLF  21
     1    RUN1(4), PN1(3), YN(2), PN(3), XKN(3)                            KFNLF     22
000002        COMMON  / CM3/ R(3), A(3), D(3), SQT2(3), SQTR(3)            KFNLF     23
000002        COMMON  / CM4/ X(2), Z(2), U(2), V(2), XC(2), XK(4)          KFNLF     24
000002        COMMON  / CM5/ C, Y1, Y2, S11, S12, S22                      KFNLF     25
000002        COMMON  / CM6/ PI, TWOPI, NSJC, NMC, NMAX, TS, PS, RST, MLF, CP  26MAR71  1
000002        COMMON  / CM7/ NM, XM(20), PC(20)                            KFNLF     27
000002        COMMON  / CM8/ FL, SL, HL, XLF(4), XLS(4), XLH(4)            KFNLF     28
000002        COMMON  / CM9/ E1C(200), VMC(300)                            KFNLF     29
000002        COMMON  /C11/ IST, IZ(2), XZ(2)                              KFNLF     30
000002        COMMON  /C111/ XKNI(3)                                       KFNLF     31
000002        LOGICAL  SCANX, TS, PS, RST, FL, SL, HL, MLF, CP             26MAR71    2
000002        INTEGER  RE, PR, TP                                          7MAR71     4
000002        DATA  (RE=5), (PR=6), (TP=7)                                 KFNLF     34
000002        CALL  INPUT(MC,N,IX1,IX2,EJ,N,JC,H1,H2,H3)                   KFNLF     35
000014        MPR=NMC/10                                                   26MAR71    3
000017        IF (MPR.LT.1) MPR=1                                          26MAR71    4
000023        MCS=MC                                                       KFNLF     36
000025        DO 600 MC=MCS,NMC                                            KFNLF     37
000026        IF (N.GE.NMAX) GO TO 450                                     KFNLF     38
000030        IF (N.EQ.0) CALL  INITLS(MC,TS,JC,N,EJ,IX1,IX2)              KFNLF     39
000037        IF (MLF) GO TO 25                                            19MAR71    3
000041        PS=.TRUE.                                                    19MAR71    4
000041        IF (MOD(MC,10).EQ.1.OR.MC.EQ.MCS.OR.MC.EQ.NMC) PS=.FALSE.    19MAR71    5
000060        IF (CP) CALL  SMKPLT(JC,JC,EJ,MMAX,10)                       19MAR71    6
000065    2   DO 5 I=1,2                                                   26MAR71    5
000074        XL(I)=YN(I)                                                  KFNLF     41
000075        XL1(I)=YN(I)                                                 KFNLF     42
000076        PL(I)=PN(I)                                                  KFNLF     43
000076    5   PL1(I)=PN(I)                                                 KFNLF     44
000100        PL(3)=PN(3)                                                  KFNLF     45
000101        PL1(3)=PN(3)                                                 KFNLF     46
000102        NS=N+1                                                       KFNLF     47
000104        IF (.NOT.PS) WRITE (PR,6000) MC                              KFNLF     48
000113  6000  FORMAT(1H1,34X,47HSAMPLE PATH INFORMATION FOR MONTE CARLO NUMBER  KFNLF  49
     1    ,I5//11X,4HNPTS/2X,1HN,4X,5HV(N-1),4X,5HZ(N-1),4X,5HU(N-1),      KFNLF     50
     2    5X,4HX(N),2X,11HXHAT(N/N-1),4X,12HSIGMA(N/N-1),3X,               KFNLF     51
```

Reproduced from best available copy.

```
                3   11HXHAT(N/N-2),4X,12HSIGMA(N/N-2),5X,7HSQTLAMU,3X,3HPHI //)      KFNLF      52
000113          IF (.NOT.PS) CALL  OUTPUT(N)                                        KFNLF      53
000116          IF (.NOT.PS) WRITE (PR,60050) IN, XL(1), PL(1), PL(2), XL1(1),      KFNLF      54
                1   PL1(1), PL1(2), XL(2), PL(2), PL(3), XL1(2), PL1(2), PL1(3)     KFNLF      55
000155    6005  FORMAT(18H LINEARIZED CHECK-,I4,16H ITERATION(S)... ,7X,6E10.3/     KFNLF      56
                1   45X,6E11.3/)                                                    KFNLF      57
          C     CALL TIMER(1)                                                       KFNLF      58
000155          DO 400 N=NS,NMAX                                                    KFNLF      59
000157          IF (.NOT.MLF) CALL  SHIFT(N,EJ,JC)                                  26MAR71     6
000163          NRPTS=5                                                             21MAR71     1
000164    10    CALL  MEAS(N)                                                       KFNLF      61
000166          IF (MLF) GO TO 310                                                  26MAR71     7
000170          CALL  FILLH(N,Z,R,JC,H1,H2,H3)                                      KFNLF      62
000176          CALL  ELLPSE(N,IX1,IX2,Z,YN,PN,NPTS,XKN)                            KFNLF      63
000206          IF (SCANX) GO TO 290                                                KFNLF      64
000210          CALL  YCNV(MC,N,W,EJ,JC,H1,H2,H3,IX1,IX2)                           KFNLF      65
000221          GO TO 250                                                           KFNLF      66
000222    20    CALL  XCNV(MC,N,W,EJ,JC,H1,H2,H3,IX1,IX2)                           KFNLF      67
000234    25    IF (C.GT.1.E-20) GO TO 300                                          KFNLF      68
000240          WRITE (PR,60100) N, Z(1), Z(2)                                      KFNLF      69
000251    60100 FORMAT(37X,6HAT N =,I4,7H, Z = (,E10.3,1H,,E10.3,15H) WAS DISCARDE  KFNLF      70
                1D. )                                                             KFNLF      71
000251          NRPTS=NRPTS-1                                                       21MAR71     2
000253          IF (NRPTS.GT.0) GO TO 100                                           21MAR71     3
000255          STOP                                                                21MAR71     4
000256    30    CALL  SCALE(W,EJ)                                                   KFNLF      73
000260    31    CALL  STATS(N)                                                      26MAR71     8
000262          IF (.NOT.TS) CALL  WRITE(MC,N,W,EJ,IX1,IX2)                         KFNLF      75
000270          IF (.NOT.PS.OR.MLF) CALL FLIN(XL,PL,XL1,PL1,N,IN)                   26MAR71     9
000277          IF (.NOT.MLF) GO TO 32                                              26MAR71    10
000301          YN(1)=XL(1)                                                         26MAR71    11
000302          YN(2)=XL(2)                                                         26MAR71    12
000304    32    IF (.NOT.(MLF.AN).OR.)) GO TO 330                                   26MAR71    13
000307          DETC=PL(1)*PL(3)-PL(2)*PL(2)                                        19MAR71     7
000312          CX=1./(TWOPI*SQRT(DETC))                                            19MAR71     8
000317          CALL  MINV(PL,PLI,2)                                                19MAR71     9
000321          K=1                                                                 19MAR71    10
000322          DO 325 J=1,MMAX                                                     19MAR71    11
000324          XP(2)=JC(J)                                                         19MAR71    12
000326          DO 325 I=1,MMAX                                                     19MAR71    13
000327          XP(1)=JC(I)                                                         19MAR71    14
000331          CALL  XUNPRM(XP,RUN1,RLAMN1,YN1,XT)                                 19MAR71    15
000334          LG(K)=CX*NORMAL(XT,XL,PLI)                                          19MAR71    16
000342    32    K=K+1                                                               19MAR71    17
000347          CALL  QWKPLT(J,,JC,,E,,MMAX,13)                                     19MAR71    18
000353    33    IF (OP.AND.(.NOT.MLF)) CALL   QWKPLT(JC,JC,EJ,MMAX,13)              19MAR71    19
000361          IF (PS) GO TO 350                                                   19MAR71    20
000363          CALL  OUTPUT(N)                                                     KFNLF      78
000364          WRITE (PR,60050) IN, XL(1), PL(1), PL(2), XL1(1), PL1(1), PL1(2),   KFNLF      79
                1   XL(2), PL(2), PL(3), XL1(2), PL1(2), PL1(3)                     KFNLF      80
000422    35    CALL  MCAV(MC,N,X,YN,EMC,VMC)                                       KFNLF      81
000426    40    CONTINUE                                                            KFNLF      82
000431    45    N=0                                                                 KFNLF      83
000432          IF (.NOT.TS) CALL  MCRITE(MC)                                       KFNLF      84
000435          IF (MLF.AN).(MOD(MC,MP2).EQ..OR.MC.EQ.NMC)) WRITE (PR,60150) MC     26MAR71    14
000455    60150 FORMAT(1H1,25X,22HMONTE CARLO AVERAGE OF ,I5,                       26MAR71    15
                1   44H SAMPLE PATHS(S) OF THE LINEARIZED PREDICTOR                 26MAR71    16
                2   //25X,14HN,23X,13HCUM AVG ERROR ,17X,19HCUM AVG CORRELATION //) 26MAR71    17
```

Reproduced from best available copy.

```
000455          IF (.NOT.MLF) WRITE (PR,60200) MC
000464    6020  FORMAT(1H1,39X,22HMONTE CARLO AVERAGE OF ,I5,16H SAMPLE PATH(S),
               1   //26X,14H,23X,13HCUM AVG ERROR ,17X,19HCUM AVG CORRELATION //)
000464          IF (MLF.AND.(MOD(MC,MPR).EQ.0.OR.MC.EQ.NMC)) GO TO 475
000476          IF (MLF) GO TO 550
000477      475 CONTINUE
000477          I=1
000500          NM3=3*NMAX
000502          DO 500 J=1,NM3,3
000503          K=(I+1)/2
000505          WRITE (PR,60300) K, EMC(I), VMC(J), VMC(J+1), EMC(I+1), VMC(J+1),
               1   VMC(J+2)
000527    6030  FORMAT(21X,I10,15X,E15.8,10X,2E15.8/+6X,E15.8,10X,2E15.8/)
000527      50  I=I+2
000533     550  CONTINUE
           C    CALL  TIMER(D)
000533     600  CONTINUE
000536          IF (.NOT.TS) REWIND TP
000541          CALL EXIT
000542          END
```

```
26MAR71     10
KFNLF       86
KFNLF       87
26MAR71     19
26MAR71     20
26MAR71     21
KFNLF       88
KFNLF       89
KFNLF       90
KFNLF       91
KFNLF       92
KFNLF       93
KFNLF       94
KFNLF       95
26MAR71     22
KFNLF       96
KFNLF       97
KFNLF       98
KFNLF       99
KFNLF      100
```


PROGRAM LENGTH INCLUDING I/O BUFFERS
022701

Reproduced from best available copy.

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

2	-	000165	100	-	000227	250	-	000235			
[illegible]	-	000267	320	-	000301	330	-	000354			
FL	-	000423	450	-	000432	475	-	000511	550	-	000534
6	-	000534	600	-	000557	60000	-	000612	60100	-	000624
60000	-	000637	60200	-	000664	60300	-	000702			

BLOCK NAMES AND LENGTHS

CM1	-	000016	CM2	-	000033	CM3	-	000017	CM4	-	000616
CM5	-	000000	CM6	-	000012	CM7	-	000007	CM8	-	000017
CM9	-	000764	CM10	-	000005	CM11	-	000003			

VARIABLE ASSIGNMENTS

A	-	000363033	C	-	000000015	CP	-	000011006	CK	-	014607
DETC	-	014605	E5	-	012624	EJ	-	000761	EMC	-	000000011
FL	-	000000016	H1	-	000000016	H1	-	014657	H2	-	005616
H3	-	010559	I	-	014602	IM	-	014604	IX1	-	012514
IX2	-	012557	IZ	-	000001012	J	-	014611	JC	-	000722
K	-	014601	MC	-	014575	MCS	-	014601	MLF	-	000013005
MMAX	-	000000001	MPR	-	014600	N	-	014577	NMAX	-	000004005
NMC	-	000000005	NM3	-	014612	NORMAL	-	014572	NPTS	-	000005001
NPPTS	-	014535	NS	-	014603	PC	-	000025007	PL	-	012614
PLI	-	014567	PL1	-	012621	PN	-	000025002	PN1	-	000020003
PR	-	014577	PS	-	000006 6	Q	-	000006003	R	-	000007003
RE	-	014571	RLAMN1	-	000112 2	RLAMN2	-	000002002	RST	-	000007005
RUN1	-	000 14003	RUN2	-	000004002	SQANX	-	000012001	SL	-	000001010
SQTO	-	000 110 3	SUM	-	000140 3	TP	-	014575	TS	-	000105005
TWOPI	-	000 01005	U	-	000040 4	V	-	000005004	VMC	-	000031011
W	-	00272	X	-	000004 4	XC	-	000112004	XK	-	000012004
XKN	-	000 3 002	XMNT	-	000000013	XL	-	012512	XLF	-	000003011

```
XLH    -  000013C10 XLS    -  000007C10 XL1    -  012617    XM     -  000001C07
XP     -  014563    XT     -  014565    XZ     -  000003C12 YN     -  000023C02
YN1    -  000010C02 YN2    -  000003C 2 Z      -  000002C04

START OF CONSTANTS
000545

START OF TEMPORARIES
000711

START OF INDIRECTS
000721

UNUSED COMPILER SPACE
034000
```

```
                FUNCTION ALAG3D (XO,YO,X,Y,Z,NX,NY)                                  19MAR71
          C                                                                          19MAR71
          C     ALAG3D, A THREE-DIMENSIONAL LAGRANGIAN INTERPOLATION ROUTINE.        19MAR71
          C                                                                          19MAR71
          C     ALAG3D INTERPOLATES A TABLE OF Z AS A FUNCTION OF THE INDEPENDENT    19MAR71
          C     VARIABLE ARRAYS, X AND Y, AT ENTRY POINTS XO,YO.                     19MAR71
          C     ALAG3D USES TWO-WAY FOUR-POINT LAGRANGIAN INTERPOLATION.             19MAR71
          C     NOTE -  BOTH THE X AND Y ARRAYS MUST BE IN ASCENDING ORDER.          19MAR71
          C                                                                          19MAR71
          C     ROUTINE BY CAPTAIN CHESTER F. DEAN AND HARRY M. MURPHY, JR.          19MAR71
          C     AIR FORCE WEAPONS LABORATORY, KIRTLAND A F.B., NEW MEXICO.           19MAR71
          C     SEPTEMBER 1969.                                                      19MAR71
          C                                                                          19MAR71
          C              *****   AMERICAN STANDARD FORTRAN   *****                   19MAR71
          C                                                                          19MAR71
000011          DIMENSION X(NX), Y(NY), Z(NX,NY)                                     19MAR71
          C                                                                          19MAR71
000011          INTEGER I,J,L,M                                                      19MAR71
          C                                                                          19MAR71
000011          REAL U1,U2,U3,U4,V1,V2,V3,V4                                         19MAR71
          C                                                                          19MAR71
          C                                                                          19MAR71
          C     START. BINARY SEARCH X-ARRAY FOR ENTRY POINT, XO.                    19MAR71
          C                                                                          19MAR71
000011    1     L=2                                                                  19MAR71
000012          M=NX-1                                                               19MAR71
000013          IF (2-M) 2,2,14                                                      19MAR71
          C                                                                          19MAR71
000016    2     I=(L+M)/2                                                            19MAR71
000020          IF (L-I) 3,7,2                                                       19MAR71
000022    3     IF (X(I)-XO) 5,7,4                                                   19MAR71
000025    4     M=I                                                                  19MAR71
000027          GO TO 2                                                              19MAR71
000027    5     IF (X(I+1)-XO) 6,7,7                                                 19MAR71
000032    6     L=I                                                                  19MAR71
000034          GO TO 2                                                              19MAR71
          C                                                                          19MAR71
          C     BINARY SEARCH Y-ARRAY FOR ENTRY POINT, YO.                           19MAR71
          C                                                                          19MAR71
000034    7     L=2                                                                  19MAR71
000035          M=NY-1                                                               19MAR71
000037          IF (2-M) 8,8,14                                                      19MAR71
          C                                                                          19MAR71
000041    8     J=(L+M)/2                                                            19MAR71
000043          IF (L-J) 9,13,8                                                      19MAR71
000045    9     IF (Y(J)-YO) 11,13,10                                                19MAR71
000050    10    M=J                                                                  19MAR71
000052          GO TO 8                                                              19MAR71
000052    11    IF (Y(J+1)-YO) 12,13,13                                              19MAR71
000055    12    L=J                                                                  19MAR71
000057          GO TO 8                                                              19MAR71
          C                                                                          19MAR71
          C     COMPUTE X-ARRAY LAGRANGIAN COEFFICIENTS.                             19MAR71
          C                                                                          19MAR71
000223    13    J1=(XO-X(I))*(XO-X(I+1))*(XO-X(I+2))/((X(I-1)-X(I))*(X(I-1)-X(I+1)   19MAR71
               1)*(X(I-1)-X(I+2)))                                                   19MAR71
000235          J2=(XO-X(I-1))*(XO-X(I+1))*(XO-X(I+2))/((X(I)-X(I-1))*(X(I)-X(I+1)   19MAR71
               1)*(X(I)-X(I+2)))                                                     19MAR71
```

```
000250          U3=(XO-X(I-1))*(XO-X(I))*(XO-X(I+2))/((X(I+1)-X(I-1))*(X(I+1)-X(I)  19MAR71    79
               1)*(X(I+1)-X(I+2)))                                                  19MAR71    80
000263          U4=(XO-X(I-1))*(XO-X(I))*(XO-X(I+1))/((X(I+2)-X(I-1))*(X(I+2)-X(I)  19MAR71    81
               1)*(X(I+2)-X(I+1)))                                                  19MAR71    82
          C                                                                         19MAR71    83
          C     COMPUTE Y-ARRAY LAGRANGIAN COEFFICIENTS.                            19MAR71    84
          C                                                                         19MAR71    85
000276          V1=(YO-Y(J))*(YO-Y(J+1))*(YO-Y(J+2))/((Y(J-1)-Y(J))*(Y(J-1)-Y(J+1)  19MAR71    86
               1)*(Y(J-1)-Y(J+2)))                                                  19MAR71    87
000312          V2=(YO-Y(J-1))*(YO-Y(J+1))*(YO-Y(J+2))/((Y(J)-Y(J-1))*(Y(J)-Y(J+1)  19MAR71    88
               1)*(Y(J)-Y(J+2)))                                                    19MAR71    89
000325          V3=(YO-Y(J-1))*(YO-Y(J))*(YO-Y(J+2))/((Y(J+1)-Y(J-1))*(Y(J+1)-Y(J)  19MAR71    90
               1)*(Y(J+1)-Y(J+2)))                                                  19MAR71    91
000340          V4=(YO-Y(J-1))*(YO-Y(J))*(YO-Y(J+1))/((Y(J+2)-Y(J-1))*(Y(J+2)-Y(J)  19MAR71    92
               1)*(Y(J+2)-Y(J+1)))                                                  19MAR71    93
          C                                                                         19MAR71    94
          C     INTERPOLATE Z VALUE AS MEAN OF X AND Y INTERPOLATIONS.              19MAR71    95
          C                                                                         19MAR71    96
000353          ALAG3D=0.5*((Z(I-1,J-1)*U1+Z(I,J-1)*U2+Z(I+1,J-1)*U3+Z(I+2,J-1)*U4  19MAR71    97
               1)*V1+(Z(I-1,J)*U1+Z(I,J)*U2+Z(I+1,J)*U3+Z(I+2,J)*U4)*V2+(Z(I-1,J+1  19MAR71    98
               2)*U1+Z(I,J+1)*U2+Z(I+1,J+1)*U3+Z(I+2,J+1)*U4)*V3+(Z(I-1,J+2)*U1+Z(  19MAR71    99
               3I,J+2)*U2+Z(I+1,J+2)*U3+Z(I+2,J+2)*U4)*V4+(Z(I-1,J-1)*V1+Z(I-1,J)*  19MAR71   100
               4V2+Z(I-1,J+1)*V3+Z(I-1,J+2)*V4)*U1+(Z(I,J-1)*V1+Z(I,J)*V2+Z(I,J+1)  19MAR71   101
               5*V3+Z(I,J+2)*V4)*U2+(Z(I+1,J-1)*V1+Z(I+1,J)*V2+Z(I+1,J+1)*V3+Z(I+1  19MAR71   102
               6,J+2)*V4)*U3+(Z(I+2,J-1)*V1+Z(I+2,J)*V2+Z(I+2,J+1)*V3+Z(I+2,J+2)*V  19MAR71   103
               7+1)*U4)                                                             19MAR71   104
000474          RETURN                                                              19MAR71   105
          C                                                                         19MAR71   106
000474    14    WRITE (6,15) NX,NY                                                  19MAR71   107
000504          ALAG3D=0.0                                                          19MAR71   108
000505          PAUSE                                                               19MAR71   109
000513          RETURN                                                              19MAR71   110
          C                                                                         19MAR71   111
000515    15    FORMAT (97H *****  FUNCTION ALAG3D.  INCORRECT VALUE OF NX AND/OR   19MAR71   112
               1NY.  PAUSE WITH ALAG3D SET TO 0.0.   *****/6H NX = I4,3H,   NY = ,  19MAR71   113
               2I4,1H./1X)                                                          19MAR71   114
000515          END                                                                 19MAR71   115
```

SUBPROGRAM LENGTH
001053

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

1	-	000012	2	-	000017	3	-	000023	4	-	000026
5	-	000032	6	-	000033	7	-	000035	8	-	000042
9	-	000046	10	-	000051	11	-	000053	12	-	000056
13	-	000060	14	-	000475	15	-	000524			

BLOCK NAMES AND LENGTHS

VARIABLE ASSIGNMENTS

ALAG3D	-	001036	I	-	001037	J	-	001040	L	-	001041
M	-	001042	NY	-	000001	U1	-	001043	U2	-	001044
U3	-	001045	U4	-	001046	V1	-	001047	V2	-	001050
V3	-	001051	V4	-	001052						

```
                SUBROUTINE  AX(S,A,N)
000005          DIMENSION  S(1), A(1), T(4)
000005          COMMON  / CM1/ NT(8), NP
000005          COMMON  / CM3/ RT(6), Q(3)
000005          T(1)=S(1)
000005          T(2)=S(2)
000007          IF (NP.EQ.1) GO TO 10
000011          T(2)=S(3)
000012          T(3)=S(2)
000014          T(4)=S(4)
000016       10 CALL  FTAF(T,Q,A,NP,2)
000022          RETURN
000023          END
```

```
KFNLF     101
KFNLF     102
KFNLF     103
KFNLF     104
KFNLF     105
KFNLF     106
KFNLF     107
KFNLF     108
KFNLF     109
KFNLF     110
KFNLF     111
KFNLF     112
KFNLF     113
```


SUBPROGRAM LENGTH
000043

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
10 - 000017

BLOCK NAMES AND LENGTHS
CM1 - 000011 CM3 - 000011

VARIABLE ASSIGNMENTS
NP - 000100C01 NT - 000000C01 Q - 000006C02 RT - 000000C02
T - 000037

START OF CONSTANTS
000026

START OF TEMPORARIES
000030

START OF INDIRECTS
000032

UNUSED COMPILER SPACE
037100

```
                FUNCTION  CUMAVG(AV,X,I)                          KFNLF     114
000005          CUMAVG=((I-1.)*AV+X)/I                            KFNLF     115
000011          RETURN                                            KFNLF     116
000012          END                                               KFNLF     117


SUBPROGRAM LENGTH
000026

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

BLOCK NAMES AND LENGTHS

VARIABLE ASSIGNMENTS
CUMAVG -  000025

START OF CONSTANTS
000015

START OF TEMPORARIES
000017

START OF INDIRECTS
000025

UNUSED COMPILER SPACE
037200
```

```
                SUBROUTINE  DENSTY(JC,EJ)                                       KFNLF     118
000004          REAL  JC(1), EJ(1), CI(3), XP(2), X(2), XM1(2), C(3)            KFNLF     119
***TT ********************** DENSTY
000004          REAL NORMAL                                                     19MAR71   116
000004          COMMON  / CM6/ PI, TWOPI                                        19MAR71   117
000004          COMMON  / CM1/ M, NSD, MMAX                                     KFNLF     120
000004          COMMON  / CM2/ YT(8), YN1(2), RLAMN1(2), RUN1(4)                KFNLF     121
000004          COMMON  / CM7/ N1, XM(20), PC(30)                               KFNLF     122
000004          MM=MMAX*MMAX                                                    KFNLF     123
000005          N3=NM*3                                                         KFNLF     124
000007          DO 10 I=1,MM                                                    KFNLF     125
000014       10 EJ(I)=0.                                                        KFNLF     126
000016          T=0.                                                            KFNLF     127
000016          K1=1                                                            KFNLF     128
000020          DO 30 K=1,N3,3                                                  KFNLF     129
000022          C(1)=PC(K)                                                      KFNLF     130
000023          C(2)=PC(K+1)                                                    KFNLF     131
000024          C(3)=PC(K+2)                                                    KFNLF     132
000026          CALL  MINV(C,CI,2)                                              KFNLF     133
000031          XM1(1)=XM(K1)                                                   KFNLF     134
000033          XM1(2)=XM(K1+1)                                                 KFNLF     135
000036          DETC=C(1)*C(3)-C(2)*C(2)                                        19MAR71   118
000041          CX=1./(TWOPI*SQRT(DETC))                                        19MAR71   119
000045          L=1                                                             KFNLF     136
000047          DO 20 J=1,MMAX                                                  KFNLF     137
000052          XP(2)=JC(J)                                                     KFNLF     138
000054          DO 20 I=1,MMAX                                                  KFNLF     139
000055          XP(1)=JC(I)                                                     KFNLF     140
000057          CALL  XUMPRM(XP,RUN1,RLAMN1,YN1,X)                              KFNLF     141
000062          Z=CX*NORMAL(X,XM1,CI)                                           19MAR71   120
000071          EJ(L)=EJ(L)+Z                                                   KFNLF     143
000072          T=T+Z                                                           KFNLF     144
000074       20 L=L+1                                                           KFNLF     145
000101       30 K1=K1+2                                                         KFNLF     146
000105          D=(T*NSD*NSD*RLAMN1(1)*RLAMN1(2))/FLOAT(M*M)                    KFNLF     147
000114          DO 40 I=1,MM                                                    KFNLF     148
000122       40 EJ(I)=EJ(I)/D                                                   KFNLF     149
000124          RETURN                                                          KFNLF     150
000124          END                                                             KFNLF     151


SUBPROGRAM LENGTH
000175

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

BLOCK NAMES AND LENGTHS
CM6     -  000002     CM1     -  000003     CM2     -  000020     CM7     -  000063

VARIABLE ASSIGNMENTS
C       -  000155     CI      -  000144     CX      -  000170     D       -  000174
DETC    -  000167     I       -  000163     J       -  000172     K       -  000166
K1      -  000165     L       -  000171     M       -  000000C02  MM      -  000161
MMAX    -  000002C02  N1      -  000000C04  NORMAL  -  000160     NSD     -  000001C02
N3      -  000162     PC      -  000025C04  RLAMN1  -  000012C03  RUN1    -  000014C03
```

```
T       -  000164     TWOPI  -  000001C01 X       -  000151     XM      -  000001C04
XM1     -  000153     XP     -  000147     YN1     -  000019C03 YT      -  000003C03
Z       -  000173

START OF CONSTANTS
000127

START OF TEMPORARIES
000132

START OF INDIRECTS
000142

UNUSED COMPILER SPACE
076500

II********VARIABLE GIVEN CONFLICTING TYPES
    000005
```

```
                SUBROUTINE ECHECK(N,MC,IX,IY,EJ,JC,H1,H2,H3,IX1,IX2,XT,Y,I1,I2,IT) KFNLF     152
000022          REAL  EJ(1), JC(1), H1(1), H2(1), H3(1), XT(1), Y(1)               KFNLF     153
***TT ********************** ECHECK
000022          INTEGER  IX1(1), IX2(1)                                            KFNLF     154
***TT ********************** ECHECK
000022          DIMENSION  EG(441), EH(441), EI(441), XP(2), X(2)                  KFNLF     155
000022          COMMON  / CM1/ M, NSD, MMAX, A11, A12, A22, NPTS, NPQ              KFNLF     156
000022          COMMON  / CM2/ RN2(2), RLAMN2(2), RUN2(4)                          KFNLF     157
000022          COMMON  / CM6/ PT(9), CP                                           26MAR71    23
000022          COMMON  /CM11/ XKNI(3)                                             KFNLF     158
000022          LOGICAL  CP                                                        25MAR71    24
000022          INTEGER  PR                                                        KFNLF     159
000022          DATA  (PR=6)                                                       KFNLF     160
000022          MMAX=0.                                                            KFNLF     161
000022          XH=XT(1)                                                           KFNLF     162
000024          YH=XT(2)                                                           KFNLF     163
000026          K=1                                                                KFNLF     164
000027          DO 200 J=1,MMAX                                                    KFNLF     165
000031          XP(2)=JC(J)                                                        KFNLF     166
000033          DO 200 I=1,MMAX                                                    KFNLF     167
000034          XP(1)=JC(I)                                                        KFNLF     168
000036          CALL  XUNPRM(XP,RUN2,RLAMN2,YN2,X)                                 KFNLF     169
000045          Z1=Y(1)+H1(K)                                                      KFNLF     170
000050          Z2=Y(2)+H2(K)                                                      KFNLF     171
000054          Q=H3(K)+(A11*Z1+A12*Z2)*Z1+A22*Z2*Z2                               KFNLF     172
000063          Q=EXP(Q)*EJ(K)                                                     KFNLF     173
000073          IF (Q.LT.HMAX) GO TO 100                                           KFNLF     174
000075          MMAX=0                                                             KFNLF     175
000076          IM=I                                                               KFNLF     176
000077          JM=J                                                               KFNLF     177
000101      100 EG(K)=Q                                                            KFNLF     178
000102          Z1=X(1)-XH                                                         KFNLF     179
000105          Z2=X(2)-YH                                                         KFNLF     180
000107          Q=XKNI(1)*Z1*Z1+2.*XKNI(2)*Z1*Z2+XKNI(3)*Z2*Z2                     KFNLF     181
000116          EH(K)=SQRT(Q)                                                      KFNLF     182
000124      200 K=K+1                                                              KFNLF     183
000133          IF (CP) CALL  QWKPLT(JC,JC,EG,MMAX,1J)                             13MAR71   121
000137          MM=MMAX*MMAX                                                       KFNLF     184
000141          DO 300 I=1,MM                                                      KFNLF     185
000146      300 EI(I)=HMAX*EXP(-.5*EH(I)*EH(I))                                    KFNLF     186
000163          WRITE (PR,6000) MC, N, IX, IY, (EG(I),I=1,MM)                      KFNLF     187
000201  6000 FORMAT(9HCFOR MC =,I4,4H N =,I4,4H I =,I4,4H J =,I4,                  KFNLF     188
               1   41H, THE KERNEL W(.,.)JN(.) IS AS FOLLOWS...//(1X,E7.1,14E8.2)) KFNLF     189
000205          WRITE (PR,6010) (EH(I),I=1,MM)                                     KFNLF     190
000213  6010 FORMAT(69H1THE THEORETICAL NO. OF STANDARD DEVIATIONS IS AS FOLLOW    21MAR71    5
               1...//(1X,E7.1,14E8.2))                                            KFNLF     192
000217          WRITE (PR,6020) (EI(I),I=1,MM)                                     KFNLF     193
000225  6020 FORMAT(53H1THE ADJUSTED THEORETICAL KERNEL IS COMPUTED TO BE...//     21MAR71    6
               1   (1X,E7.1,14E8.2))                                              KFNLF     195
000231          WRITE (PR,6030) HMAX, JM, IM                                       KFNLF     196
000242  6030 FORMAT(30H1THE MAXIMUM OF W(.,.)JN(.) IS ,E12.4,                      KFNLF     197
               1   24H AND HAS COORDINATES J =,I4,4H I =,I4)                      KFNLF     198
000246          WRITE (PR,6040) I2,I1,(I,IX1(I),IX2(I),I=1,NPQ)                    KFNLF     199
000274  6040 FORMAT(40H1THE TRACKING ELLIPSE IS CENTERED AT J =,I4,4H I =,I4//     KFNLF     200
               1   51H THE RELATIVE ELLIPSE DESCRIPTION IS AS FOLLOWS... /        KFNLF     201
               2   (20X,I5,10X,2I10))                                             KFNLF     202
000300          IF (IT.EQ.3) WRITE (PR,6050)                                       KFNLF     203
000311  6050 FORMAT(33H1THE SCAN IS ALONG X WITH Y FIXED )                         KFNLF     204
```

```
000311          IF (IT.NE.0) WRITE (PR,60600)                                   KFNLF     205
000322    60600 FORMAT(33H0THE SCAN IS ALONG Y WITH X FIXED )                   KFNLF     206
000322          RETURN                                                          KFNLF     207
000323          END                                                             KFNLF     208


SUBPROGRAM  LENGTH
003150

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
100     -  000102    60000  -  000332    60100  -  000350    60200  -  000362
60300   -  000374    60400  -  000410    60500  -  000430    60600  -  000436

BLOCK NAMES AND LENGTHS
CM1     -  000010    CM2    -  000010    CM6    -  000012    CM11   -  000003

VARIABLE ASSIGNMENTS
A11     -  000003C01 A12    -  000004C01 A22    -  000005C01 CP     -  000011C03
EG      -  000454    EH     -  001345    EI     -  002236    HMAX   -  003134
H1      -  000000    H2     -  000001    H3     -  000002    I      -  003141
IM      -  003145    IT     -  000011    IX1    -  000003    IX2    -  000004
I1      -  000007    I2     -  000010    J      -  003140    JM     -  003146
K       -  003137    MM     -  003147    MMAX   -  000002C01 NPQ    -  000007C01
PR      -  003133    PT     -  000000C03 Q      -  003144    RLAMN2 -  000002C02
RUN2    -  000004C02 X      -  003131    XH     -  003135    XKNI   -  000000C04
XP      -  003127    XT     -  000005    Y      -  000006    YH     -  003136
YN2     -  000000C02 Z1     -  003142    Z2     -  003143

START OF CONSTANTS
000326

START OF TEMPORARIES
000444

START OF INDIRECTS
000452

UNUSED COMPILER SPACE
035400

TT*******VARIABLE GIVEN CONFLICTING TYPES
    000023,000023
```

```
                SUBROUTINE  EIGEN(P,U,XL)                                KFNLF     209
000005          DIMENSION  P(1), U(1), XL(1)                             KFNLF     210
          C        DETERMINES EIGENVALUES AND EIGENVECTORS OF P           KFNLF     211
000006          A=(P(1)+P(3))/2.                                         KFNLF     212
000007          B=P(1)-P(3)                                              KFNLF     213
000011          B=.5*SQRT(B*B+4.*P(2)*P(2))                              KFNLF     214
000023          XL(1)=A+B                                                KFNLF     215
000024          XL(2)=A-B                                                KFNLF     216
000025          A=P(3)-XL(1)                                             KFNLF     217
000027          B=-P(2)                                                  KFNLF     218
000030          C=SQRT(A*A+B*B)                                          KFNLF     219
000034          IF (C.LT.1.E-40) GO TO 10                                KFNLF     220
000043          U(1)=A/C                                                 KFNLF     221
000044          U(2)=B/C                                                 KFNLF     222
000046          GO TO 20                                                 KFNLF     223
000046       10 U(1)=1.                                                  KFNLF     224
000050          U(2)=0.                                                  KFNLF     225
000051          U(3)=0.                                                  7MAR71      5
000051          U(4)=1.                                                  7MAR71      6
000052          IF (P(1).GE.P(3)) RETURN                                 7MAR71      7
000056          U(1)=0.                                                  7MAR71      8
000057          U(2)=-1.                                                 7MAR71      9
000061          U(3)=1.                                                  7MAR71     10
000062          U(4)=0.                                                  7MAR71     11
000063          RETURN                                                   7MAR71     12
000064       20 A=-P(2)                                                  KFNLF     226
000065          B=P(1)-XL(2)                                             KFNLF     227
000070          C=SQRT(A*A+B*B)                                          KFNLF     228
000073          IF (C.LT.1.E-40) GO TO 30                                KFNLF     229
000102          U(3)=A/C                                                 KFNLF     230
000104          U(4)=B/C                                                 KFNLF     231
000105          RETURN                                                   KFNLF     232
000106       30 U(3)=0.                                                  KFNLF     233
000107          U(4)=1.                                                  KFNLF     234
000111          RETURN                                                   KFNLF     235
000111          END                                                      KFNLF     236


SUBPROGRAM LENGTH
000154

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
10     - 000047    20     - 000065    30     - 000107

BLOCK NAMES AND LENGTHS

VARIABLE ASSIGNMENTS
A      - 000151    B      - 000152    C      - 000153

START OF CONSTANTS
000114

START OF TEMPORARIES
000122
```

```
                SUBROUTINE  ELLPSE(N,IX1,IX2,Z,Y,P,NPTS,XKN)                    KENLF    237
000012          DIMENSION  IX1(1), IX2(1), P(1), Y(1), Z(1), XKN(1)             KENLF    238
          C       DETERMINES TRACKING ELLIPSE PARAMETERS                        KENLF    239
000012          DIMENSION  E1(2), XL(2), V1(2), V2(2), ZERO(2)                  KENLF    240
000012          DIMENSION  A(3), AI(3), PI(3), RI(3), XKNIR(3)                  KENLF    241
000012          DIMENSION  DF(4), DH(4), U(4), C1(4), C2(4)                     KENLF    242
000012          COMMON  / CM1/ M, NSQ, MMAX, A11, A12, A22, NZ, NPQ, NP, NQ, SCANX KENLF 243
000012          COMMON  / CM2/  YN2(2), RLAMN2(2), RUN2(4)                      KENLF    244
000012          COMMON  / CM3/ R(3)                                             KENLF    245
000012          COMMON  / CM6/ TP(2), NSOC                                      KENLF    246
000012          COMMON  /C411/ XKNI(3)                                          KENLF    247
000012          COMMON  / CM4/ XZ(8), XC(2), XK(4)                              KENLF    248
000012          LOGICAL  SCANX                                                  KENLF    249
000012          ZERO(1)=0.                                                      KENLF    250
000012          ZERO(2)=0.                                                      KENLF    251
000013          CALL  SX(Y,C1,N)                                                KENLF    252
000016          CALL  AX(C1,A,N)                                                KENLF    253
000025          CALL  MINV(A,AI,2)                                              KENLF    254
000030          CALL MINV(R,RI,N)                                               KENLF    255
000033          CALL MINV(P,PI,2)                                               KENLF    256
000041          CALL  DFX(Y,DF,N)                                               KENLF    257
000047          CALL DHX(Z,DH,N)                                                KENLF    258
000055          CALL FTAF(DH,RI,C1,NN,2)                                        KENLF    259
000067          CALL FTSF(DF,AI,C2,2,2)                                         KENLF    260
00096           DO 10 I=1,3                                                     KENLF    261
000?7           XKN(I)=C1(I)+C2(I)+PI(I)                                        KENLF    262
00010           CALL MINV(XKNI,XKN,2)                                           KENLF    263
00010?          CALL FICEN(XKN,U,XL)                                            KENLF    264
00010?          CALL APPM(U,RUN2,RLAMN2,ZERO,E1)                                KENLF    265
000113          SCANX=.FALSE.                                                   KENLF    266
000114          IF (ABS(E1(2)).GE.ABS(E1(1))) SCANX=.TRUE.                      KENLF    267
000125          CALL  ROTATE(XKNI,XKNIR,RUN2,RLAMN2)                            KENLF    268
000130          NSQ=NSOC*NSOC                                                   KENLF    269
000132          XNS=FLOAT(NSQ*NSQ)/FLOAT(M*M)                                   KENLF    270
000137          DO 15 I=1,3                                                     KENLF    271
000147       15 XKNIR(I)=XKNIR(I)*XNS                                           KENLF    272
000151          NPTS=0                                                          KENLF    273
000152          DO 80 K=1,MMAX                                                  KENLF    274
000157          NPQ=K                                                           KENLF    275
000157          L=E1(2)*(K-1)/E1(1)+.5                                          KENLF    276
000166          IF (SCANX) L=E1(1)*(K-1)/E1(2)+.5                               KENLF    277
000176          DO 30 K1=1,MMAX                                                 KENLF    278
000200          I=L+1-K1                                                        KENLF    279
000202          IF (SCANX) GO TO 20                                             KENLF    280
000204          IF (QUAD(K-1,I,XKNIR).GT.NSQ) GO TO 40                          KENLF    281
000220          GO TO 30                                                        KENLF    282
000220       20 IF (QUAD(I,K-1,XKNIR).GT.NSQ) GO TO 40                          KENLF    283
000235       30 CONTINUE                                                        KENLF    284
000240       40 IX1(K)=I                                                        KENLF    285
000242          DO 60 K1=1,MMAX                                                 KENLF    286
000244          I=K1+L                                                          KENLF    287
000245          IF (SCANX) GO TO 50                                             KENLF    288
000247          IF (QUAD(K-1,I,XKNIR).GT.NSQ) GO TO 70                          KENLF    289
000264          GO TO 60                                                        KENLF    290
000264       50 IF (QUAD(I,K-1,XKNIR).GT.NSQ) GO TO 70                          KENLF    291
000301       60 CONTINUE                                                        KENLF    292
000304       70 IX2(K)=I-IX1(K)                                                 KENLF    293
000306          IF (I.GT.0.AND.IX1(K).LE.0) IX2(K)=IX2(K)+1                     KENLF    294
```

```
000320          NPTS=NPTS+IX2(K)                                        KFNLF   295
000323          IF (K.EQ.0) IR=NPTS                                     KFNLF   296
000325          IF (SCANX) GO TO 75                                     KFNLF   297
000327          IF (QUAD(K-1,L,XKNIR).GT.NSQ) GO TO 90                  KFNLF   298
000343          GO TO 80                                                KFNLF   299
000343       75 IF (QUAD(L,K-1,XKNIR).GT.NSQ) GO TO 90                  KFNLF   300
000360       80 CONTINUE                                                KFNLF   301
000363       90 NPTS=2*NPTS-IR                                          KFNLF   302
000365          CALL  FX(Y,V1,N)                                        KFNLF   303
000370          CALL VMULT(2,2,2,I,DF,Y,V2)                             KFNLF   304
000403          V2(1)=V2(1)-V1(1)                                       KFNLF   305
000405          V2(2)=V2(2)-V1(2)                                       KFNLF   306
000407          CALL FTA(DF,AI,C1,2,2)                                  KFNLF   307
000413          CALL  TMULT(XKN,C1,XK,2)                                KFNLF   308
000417          CALL VMULT(2,2,2,1,XK,V2,XC)                            KFNLF   309
000426          CALL FTA(DH,RI,C1,NQ,2)                                 KFNLF   310
000432          CALL HX(Y,V1,N)                                         KFNLF   311
000440          CALL VMULT(NQ,NQ,2,1,DH,Y,V2)                           KFNLF   312
000457          V2(1)=V2(1)-V1(1)+Z(1)                                  KFNLF   313
000462          V2(2)=V2(2)-V1(2)+Z(2)                                  KFNLF   314
000465          CALL VMULT(2,2,NJ,NQ,C1,V2,V1)                          KFNLF   315
000475          CALL  TMULT(P,Y,V2,1)                                   KFNLF   316
000501          V1(1)=V1(1)+V2(1)                                       KFNLF   317
000503          V1(2)=V1(2)+V2(2)                                       KFNLF   318
000505          CALL  TMULT(XKN,V1,V2,1)                                KFNLF   319
000511          XC(1)=XC(1)+V2(1)                                       KFNLF   320
000513          XC(2)=XC(2)+V2(2)                                       KFNLF   321
000515          RETURN                                                  KFNLF   322
000516          END                                                     KFNLF   323
```

SUBPROGRAM LENGTH
000622

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

20	-	000221	30	-	000236	40	-	000241	50	-	000265
60	-	000302	7	-	000305	75	-	000344	80	-	000361
90	-	000364									

BLOCK NAMES AND LENGTHS

CM1	-	000013	CM2	-	000010	CM3	-	000003	CM6	-	000003
CM11	-	000003	CM4	-	000016						

VARIABLE ASSIGNMENTS

A	-	000550	AI	-	000553	C1	-	000503	C2	-	000607
DF	-	000567	DH	-	000573	E1	-	000535	I	-	000613
IR	-	000521	K	-	000616	K1	-	000620	L	-	000617
M	-	000300C01	MMAX	-	000002C01	NPQ	-	000007C01	NPTS	-	000000
NQ	-	000011C01	NSQ	-	000001C01	NSQC	-	000002C04	NSQ	-	000614
PI	-	000556	R	-	000003C03	PI	-	000551	RLAMN2	-	000102C02
RUN2	-	000134C02	SCANX	-	000012C01	TP	-	000000C04	U	-	000577
V1	-	000542	V2	-	000544	XC	-	000010C06	XK	-	000012C06
XKN	-	000401	XKNI	-	000000C05	XKNIR	-	000564	XL	-	000540
XNS	-	000615	XZ	-	000000C06	YN2	-	000000C02	ZERO	-	000546

```
                SUBROUTINE  FILLH(N,Z,R,JC,H1,H2,H3)                              KFNLF     324
000011          REAL  Z(1), R(1), JC(1), H1(1), H2(1), H3(1)                      KFNLF     325
          C       GENERATES H TABLES TO REDUCE CONVOLUTION CALCULATIONS           KFNLF     326
***** ********************** FILLH
000011          COMMON  / CM1/ IT(2), MMAX, AT(3), NT(3), NQ                      KFNLF     327
000011          COMMON  / CM2/ RN2(2), RLAMN2(2), RUN2(4)                         KFNLF     328
000011          DIMENSION  X(2), XP(2), PI(3), XT(2)                              KFNLF     329
000011          K=1                                                               KFNLF     330
000012          CALL  MINV(R,RI,NQ)                                               KFNLF     331
000014          DO 10 J=1,MMAX                                                    KFNLF     332
000021          XP(2)=JC(J)                                                       KFNLF     333
000023          DO 10 I=1,MMAX                                                    KFNLF     334
000024          XP(1)=JC(I)                                                       KFNLF     335
000026          CALL XUNPRM(XP,RUN2,RLAMN2,RN2,X)                                 KFNLF     336
000031          CALL FX(X,XT,N)                                                   KFNLF     337
000044          H1(K)=-XT(1)                                                      KFNLF     338
000045          H2(K)=-XT(2)                                                      KFNLF     339
000047          CALL  HX(X,XT,N)                                                  KFNLF     340
000052          X(1)=Z(1)-XT(1)                                                   KFNLF     341
000057          IF (NQ.GT.1) X(2)=Z(2)-XT(2)                                      KFNLF     342
000065          H3(K)=X(1)*RI(1)*X(1)                                             KFNLF     343
000067          IF (NQ.GT.1) H3(K)=H3(K)+(2.*X(1)*R(2)+X(2)*R(3))*X(2)            KFNLF     344
000101          H3(K)=-.5*H3(K)                                                   KFNLF     345
000104       10 K=K+1                                                             KFNLF     346
000112          RETURN                                                            KFNLF     347
000113          END                                                               KFNLF     348


SUBPROGRAM LENGTH
000145

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

BLOCK NAMES AND LENGTHS
CM1    -  000012    CM2    -  000010

VARIABLE ASSIGNMENTS
AT     -  000002C01 H3     -  000000    I      -  000144    IT     -  000003C01
J      -  000143    K      -  000142    MMAX   -  000002C01 NQ     -  000011C01
NT     -  000006C01 RI     -  000135    RLAMN2 -  000002C02 RUN2   -  000004C02
X      -  000131    XP     -  000133    XT     -  000140    YN2    -  000000C02

START OF CONSTANTS
000116

START OF TEMPORARIES
000121

START OF INDIRECTS
000125

UNUSED COMPILER SPACE
036600

**********VARIABLE GIVEN CONFLICTING TYPES
```

```
                SUBROUTINE  FINDPF(NM,XM,PC,F,P)                        KFNLF     349
000007          DIMENSION   XM(1), PC(1), Y(1), P(1), PINV(3)           7MAR71     13
000007          NM2=2*NM                                                7MAR71     14
000010          DO 10 I=1,2                                             7MAR71     15
000013          F(I)=0.                                                 7MAR71     16
000014       10 P(I)=0.                                                 7MAR71     17
000020          P(3)=0.                                                 7MAR71     18
000021          J=1                                                     7MAR71     19
000022          DO 20 I=1,NM2,2                                         7MAR71     20
000036          Y(1)=Y(1)+PC(J)*XM(I)+PC(J+1)*XM(I+1)                   7MAR71     21
000044          Y(2)=Y(2)+PC(J+1)*XM(I)+PC(J+2)*XM(I+1)                 7MAR71     22
000051          P(1)=P(1)+PC(J)                                         7MAR71     23
000054          P(2)=P(2)+PC(J+1)                                       7MAR71     24
000057          P(3)=P(3)+PC(J+2)                                       7MAR71     25
000061       20 J=J+3                                                   7MAR71     26
000070          CALL  MINV(P,PINV,2)                                    7MAR71     27
000103          YT=Y(1)*PINV(1)+Y(2)*PINV(2)                            7MAR71     28
000105          Y(2)=Y(1)*PINV(2)+Y(2)*PINV(3)                          7MAR71     29
000110          Y(1)=YT                                                 7MAR71     30
000111          RETURN                                                  7MAR71     31
000112          END                                                     7MAR71     32
```

SUBPROGRAM LENGTH
000142

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

BLOCK NAMES AND LENGTHS

VARIABLE ASSIGNMENTS

I	- 000137	J	- 000140	NM2	- 000136	PINV	- 000133
YT	- 000141						

START OF CONSTANTS
000115

START OF TEMPORARIES
000117

START OF INDIRECTS
000121

UNUSED COMPILER SPACE
036700

```
      SUBROUTINE  FLIN(XH1,PN1,XL1,PL1,N,IN)                            KFNLF     367
000010        DIMENSION  XH1(1), PN1(1), XL1(1), PL1(1)                 KFNLF     368
000010        DIMENSION  AK(4), DF(4), DH(4), DH1(4), F(2), H(2), PN(3),KFNLF     369
     1    TM(4), TM1(4), TM2(4), XH(2)                                  KFNLF     370
000010        COMMON  / CM1/ NT(8), NP, NQ                              KFNLF     371
000010        COMMON  / CM3/ R(3), A(3)                                 KFNLF     372
000010        COMMON  / CM4/ XT(2), Z(2)                                KFNLF     373
000010        COMMON  / CM6/ PI, TWOPI                                  26MAR71    25
000010        CALL  DHX(XH1,DH,N)                                       KFNLF     375
000012        CALL  HX(XH1,H,N)                                         KFNLF     376
000020        DO 1000 K=1,NI                                            KFNLF     377
000025        IN=K                                                      KFNLF     378
000026        TM(1)=DH(1)                                               KFNLF     379
000027        TM(2)=DH(2)                                               KFNLF     380
000030        IF (NQ.EQ.1) GO TO 100                                    KFNLF     381
000033        TM(2)=DH(3)                                               KFNLF     382
000034        TM(3)=DH(2)                                               KFNLF     383
000035        TM(4)=DH(4)                                               KFNLF     384
000037   10 CALL  FTAF(TM,PN1,TM1,2,NQ)                                 KFNLF     385
000042        TM1(1)=TM1(1)+R(1)                                        KFNLF     386
000044        IF (NQ.EQ.1) GO TO 200                                    KFNLF     387
000052        TM1(2)=TM1(2)+R(2)                                        KFNLF     388
000054        TM1(3)=TM1(3)+R(3)                                        KFNLF     389
000056   200 CALL  MINV(TM1,TM2,NQ)                                     KFNLF     390
000061        CALL  VMULT(2,2,NQ,NQ,TM,TM2,TM1)                         KFNLF     391
000071        CALL  TMULT(PN1,TM1,AK,NQ)                                KFNLF     392
000077        DO 300 I=1,NQ                                             KFNLF     393
000110        TM(I)=Z(I)-H(I)                                           26MAR71    26
000113        IF (ABS(TM(I)).LT.PI) GO TO 350                           26MAR71    27
000116        IF (Z(I).GT.H(I)) TM(I)=TM(I)-TWOPI                       26MAR71    28
000123   300 IF (Z(I).LT.H(I)) TM(I)=TM(I)+TWOPI                        26MAR71    29
000136   350 CONTINUE                                                   26MAR71    30
000136        CALL  VMULT(2,2,NQ,NQ,AK,TM,TM1)                          KFNLF     395
000151        XH(1)=XH1(1)+TM1(1)                                       KFNLF     396
000152        XH(2)=XH1(2)+TM1(2)                                       KFNLF     397
000154        IF (NI.LE.1) GO TO 1100                                   26MAR71    31
000157        CALL  DHX(XH,DH1,N)                                       KFNLF     398
000162        NQ2=2*NQ                                                  KFNLF     399
000164        DO 400 I=1,NQ2                                            KFNLF     400
000176        TM1(I)=DH(I)-DH1(I)                                       KFNLF     401
000200   400 TM2(I)=TM1(I)                                              KFNLF     402
000201        IF (NQ.EQ.1) GO TO 500                                    KFNLF     403
000206        TM2(2)=TM1(3)                                             KFNLF     404
000210        TM2(3)=TM1(2)                                             KFNLF     405
000211    00 CALL  VMULT(2,2,NQ,2,TM1,TM2,TM)                           KFNLF     406
000220        A1=0.                                                     KFNLF     407
000221        DO 600 I=1,NQ2,2                                          KFNLF     408
000232   600 A1=A1+TM(I)                                                KFNLF     409
000234        TM1(1)=DH(1)                                              KFNLF     410
000235        TM1(2)=DH(2)                                              KFNLF     411
000237        IF (NQ.EQ.1) GO TO 700                                    KFNLF     412
000245        TM1(2)=DH(3)                                              KFNLF     413
000246        TM1(3)=DH(2)                                              KFNLF     414
000247        TM1(4)=DH(4)                                              KFNLF     415
000251   700 CALL  VMULT(2,2,NQ,2,TM1,DH,TM)                           KFNLF     416
000260        A2=0.                                                     KFNLF     417
000261        DO 800 I=1,NQ2,2                                          KFNLF     418
000272   800 A2=A2+TM(I)                                                KFNLF     419
```

```
000274          IF (K.GE.NI) GO TO 1100                                 26MAR71    32
000302          IF (A1.LE.EPS*A2) GO TO 1100                            KFNLF     420
000306          DO 900 I=1,NQ2                                          KFNLF     421
000313      900 DH(I)=DH1(I)                                            KFNLF     422
000320     1000 CONTINUE                                                KFNLF     423
000323     1100 CALL  DFX(XH,DF,N)                                      KFNLF     424
000326          CALL  FX(XH,F,N)                                        KFNLF     425
000334          CALL  VMULT(2,2,NQ,2,AK,DH,TM)                          KFNLF     426
000343          GO TO 1350                                              26MAR71    33
000347     1301 CONTINUE                                                25MAR71    34
000347          TM1(1)=1.-TM(1)                                         KFNLF     427
000351          TM1(2)=-TM(3)                                           KFNLF     428
000352          TM1(3)=-TM(2)                                           KFNLF     429
000354          TM1(4)=1.-TM(4)                                         KFNLF     430
000356          CALL  FTAF(TM1,PN1,PN,2,2)                              KFNLF     431
000361          TM1(1)=AK(1)                                            KFNLF     432
000362          TM1(2)=AK(2)                                            KFNLF     433
000364          IF (NQ.EQ.1) GO TO 1200                                 KFNLF     434
000372          TM1(2)=AK(3)                                            KFNLF     435
000373          TM1(3)=AK(2)                                            KFNLF     436
000374          TM1(4)=AK(4)                                            KFNLF     437
000376     1200 CALL  FTAF(TM1,R,TM,NQ,2)                               KFNLF     438
000402          DO 1300 I=1,3                                           KFNLF     439
000412     1301 PN(I)=PN(I)+TM(I)                                       KFNLF     440
000424     1350 PN(1)=(1.-TM(1))*PN1(1)-TM(3)*PN1(2)                    26MAR71    35
000427          PN(2)=(1.-TM(1))*PN1(2)-TM(3)*PN1(3)                    26MAR71    36
000432          PN(3)=-TM(2)*PN1(2)+(1.-TM(4))*PN1(3)                   26MAR71    37
000440          CALL  FX(XH1,XL1,N)                                     KFNLF     441
000445          XH1(1)=F(1)                                             KFNLF     442
000446          XH1(2)=F(2)                                             KFNLF     443
000450          TM(1)=DF(1)                                             KFNLF     444
000451          TM(2)=DF(3)                                             KFNLF     445
000452          TM(3)=DF(2)                                             KFNLF     446
000454          TM(4)=DF(4)                                             KFNLF     447
000456          CALL  FTAF(TM,PN1,PL1,2,2)                              KFNLF     448
000461          CALL  FTAF(TM,PN,PN1,2,2)                               KFNLF     449
000471          DO 1400 I=1,3                                           KFNLF     450
000501          PL1(I)=PL1(I)+A(I)                                      KFNLF     451
000502     1400 PN1(I)=PN1(I)+A(I)                                      KFNLF     452
000504          RETURN                                                  KFNLF     453
000505          END                                                     KFNLF     454


SUBPROGRAM LENGTH
000600

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
100    -  000340    200    -  000057    300    -  000124    350    -  000137
900    -  000212    700    -  000252    1100   -  000324    1200   -  000377
1301   -  000350    1350   -  000421

BLOCK NAMES AND LENGTHS
CM1    -  000312    CM3    -  000306    CM4    -  000304    CM6    -  000002

VARIABLE ASSIGNMENTS
```

A	-	000003C02	AK	-	000524	A1	-	000575	A2	-	000576
DF	-	000530	DH	-	000534	DH1	-	000542	EPS	-	000577
F	-	000544	H	-	000546	I	-	000573	K	-	000571
NI	-	000572	NM	-	000011C01	NQ2	-	000574	NT	-	000CC0C01
PI	-	000003C04	PN	-	000550	R	-	000008C02	TM	-	000553
TM1	-	000557	TM2	-	000563	TWOPI	-	000001C04	XM	-	000567
XT	-	000003C03	Z	-	000002C03						

START OF CONSTANTS
000510

START OF TEMPORARIES
000513

START OF INDIRECTS
000521

UNUSED COMPILER SPACE
035000

```
                SUBROUTINE  FTA(F,A,C,NR,NC)                                    KFNLF    453
000007          DIMENSION  F(1), A(1),C(1)                                      KFNLF    456
          C        COMPUTES  C=F(TRANSPOSE)*A  WHERE  A  IS SYMMETRIC AND STORED KFNLF    457
          C        IN TRIANGULAR FORM                                           KFNLF    458
000007          C(1)=F(1)*A(1)                                                  KFNLF    459
000010          IF (NC.GT.1) GO TO 10                                           KFNLF    460
000013          IF (NR.LE.1) RETURN                                             KFNLF    461
000017          C(1)=C(1)+F(2)*A(2)                                             KFNLF    462
000022          C(2)=F(1)*A(2)+F(2)*A(3)                                        KFNLF    463
000026          RETURN                                                          KFNLF    464
000026       10 IF (NR.GT.1) GO TO 20                                           KFNLF    465
000031          C(2)=F(2)*A(1)                                                  KFNLF    466
000033          RETURN                                                          KFNLF    467
000040       20 C(1)=C(1)+F(2)*A(2)                                             KFNLF    468
000043          C(2)=F(3)*A(1)+F(4)*A(2)                                        KFNLF    469
000050          C(3)=F(1)*A(2)+F(2)*A(3)                                        KFNLF    470
000054          C(4)=F(3)*A(2)+F(4)*A(3)                                        KFNLF    471
000057          RETURN                                                          KFNLF    472
000060          END                                                             KFNLF    473
```

```
SUBPROGRAM LENGTH
000101

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
10      -  000027    20      -  000034

BLOCK NAMES AND LENGTHS

VARIABLE ASSIGNMENTS

START OF CONSTANTS
000063

START OF TEMPORARIES
000064

START OF INDIRECTS
000066

UNUSED COMPILER SPACE
037000
```

```
                SUBROUTINE  FTAF(F,A,C,NR,NC)                          KFNLF     474
000007          DIMENSION  F(1), A(1), C(1)                            KFNLF     475
          C
          C       EVALUATES  F(TRANSPOSE) A F                          KFNLF     476
          C       F HAS NR ROWS AND NC COLUMNS                         KFNLF     477
000007          C(1)=F(1)*F(1)*A(1)                                    KFNLF     478
000011          IF (NR.GT.1) GO TO 10                                  KFNLF     479
000014          IF (NC.LE.1) RETURN                                    KFNLF     480
000020          C(2)=F(1)*F(2)*A(1)                                    KFNLF     481
000023          C(3)=F(2)*F(2)*A(1)                                    KFNLF     482
000024          RETURN                                                 KFNLF     483
000024       1. C(1)=C(1)+(2.*A(2)*F(1)+A(3)*F(2))*F(2)                KFNLF     484
000034          IF (NC.LE.1) RETURN                                    KFNLF     485
000042          C1=A(1)*F(3)+A(2)*F(4)                                 KFNLF     486
000045          C2=A(2)*F(3)+A(3)*F(4)                                 KFNLF     487
000047          C(2)=C1*F(1)+C2*F(2)                                   KFNLF     488
000054          C(3)=C1*F(3)+C2*F(4)                                   KFNLF     489
000057          RETURN                                                 KFNLF     490
000057          END                                                    KFNLF     491


SUBPROGRAM LENGTH
000103

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
10      -  000025

BLOCK NAMES AND LENGTHS

VARIABLE ASSIGNMENTS
C1      -  000101    C2      -  000102

START OF CONSTANTS
000062

START OF TEMPORARIES
000064

START OF INDIRECTS
000070

UNUSED COMPILER SPACE
037000
```

```
                SUBROUTINE  FX(X,C,N)
000005          DIMENSION   X(1), C(1), T(2)
000005          COMMON  / CH8 / FL, SL, HL, XLF(4)
000005          LOGICAL  FL, SL  HL
000005          IF (FL) GO TO 10
                C          NONLINEAR DYNAMICS COMPUTED HERE
                C          IDENTITY LINEAR IS SPECIAL CASE
000006          C(1)=X(1)
000007          C(2)=X(2)
000011          RETURN
000012       10 T(1)=XLF(1)*X(1)+XLF(3)*X(2)
000015          T(2)=XLF(2)*X(1)+XLF(4)*X(2)
000020          C(1)=T(1)
000021          C(2)=T(2)
000022          RETURN
000023          ENTRY  DFX
000033          IF (FL) GO TO 20
                C          NONLINEAR GRADIENT COMPUTED HERE
                C          IDENTITY IS SPECIAL CASE
000035          C(1)=1.
000036          C(2)=0.
000037          C(3)=0.
000040          C(4)=1.
000041          RETURN
000042       20 C(1)=XLF(1)
000044          C(2)=XLF(2)
000045          C(3)=XLF(3)
000047          C(4)=XLF(4)
000050          RETURN
000051          END
```

```
KFNLF     492
KFNLF     493
KFNLF     494
KFNLF     495
KFNLF     496
KFNLF     497
KFNLF     498
KFNLF     499
KFNLF     500
KFNLF     501
KFNLF     502
KFNLF     503
KFNLF     504
KFNLF     505
KFNLF     506
KFNLF     507
KFNLF     508
KFNLF     509
KFNLF     510
KFNLF     511
KFNLF     512
KFNLF     513
KFNLF     514
KFNLF     515
KFNLF     516
KFNLF     517
KFNLF     518
KFNLF     519
KFNLF     520
KFNLF     521
```

SUBPROGRAM LENGTH
000067

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
10 - 000012 20 - 000043

BLOCK NAMES AND LENGTHS
CH8 - 000107

VARIABLE ASSIGNMENTS
DFX - 000064 FL - 000000C01 HL - 000002C01 SL - 000001C01
T - 000065 XLF - 000003C01

START OF CONSTANTS
000054

START OF TEMPORARIES
000056

START OF INDIRECTS
000060

UNUSED COMPILER SPACE

```
                FUNCTION  GRN01(NN)                                          KFNLF     522
000002          COMMON  / CM6/ PI, TWOPI                                      KFNLF     523
000002          COMMON  /CM10/ IST, IR(2), X(2)                               KFNLF     524
000002          DATA  (IM=34359738687), (JM=25173271)                         KFNLF     525
000002          J=IR(1)                                                       KFNLF     526
000003          IF (NN.NE.1) GO TO 10                                         KFNLF     527
000005          GO TO (20,40), J                                              KFNLF     528
000012       10 IR(2)=IST                                                     KFNLF     529
000014       20 IR(1)=2                                                       KFNLF     530
000015          DO 30 I=1,2                                                   KFNLF     531
000025          IR(2)=MOD(IR(2)*JM,IM)                                        KFNLF     532
000032       30 X(I)=FLOAT(IR(2))/FLOAT(IM)                                   KFNLF     533
000035          X1=SQRT(ABS(-2.*ALOG(X(1))))                                  KFNLF     534
000043          X(2)=TWOPI*X(2)                                               KFNLF     535
000045          X(1)=X1*SIN(X(2))                                             KFNLF     536
000050          X(2)=X1*COS(X(2))                                             KFNLF     537
000053          GRN01=X(1)                                                    KFNLF     538
000055          RETURN                                                        KFNLF     539
000056       40 IR(1)=1                                                       KFNLF     540
000057          GRN01=X(2)                                                    KFNLF     541
000061          RETURN                                                        KFNLF     542
000061          END                                                           KFNLF     543


SUBPROGRAM LENGTH
000126

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
10      -  000013    20      -  000015    40      -  000057

BLOCK NAMES AND LENGTHS
CM6     -  000002    CM10    -  000005

VARIABLE ASSIGNMENTS
GRN01   -  000120    I       -  000124    IM      -  000121    IR      -  000001002
IST     -  000001002 J       -  000123    JM      -  000122    TWOPI   -  000001001
X       -  000003002 X1      -  000125

START OF CONSTANTS
000064

START OF TEMPORARIES
000066

START OF INDIRECTS
000111

UNUSED COMPILER SPACE
037000
```

```
                SUBROUTINE  HX(X,C,N)
000005          DIMENSION  X(1), C(1), T(2)
000005          COMMON  / CM1/ NT(9), NQ
000005          COMMON  / CM6/ PI, TWOPI
000005          COMMON / CM8/ FL, SL, HL, XL(8), XLH(4)
000005          LOGICAL  FL, SL, HL
000005          IF (HL) GO TO 10
                C       NONLINEAR SENSOR COMPUTED HERE
                C       H(X1,X2)=ARCTAN(X2-S/X1-C) IS A SPECIAL CASE HERE
000006          B=.01
000007          BEN=AMOD(N*B,TWOPI)
000014          CEN=COS(BEN)
000015          SEN=SIN(BEN)
000017          C(1)=ATAN2(X(2)-SEN,X(1)-CEN)
000033          C(2)=0.
000034          RETURN
000035       10 IF (NQ.GT.1) GO TO 20
000041          C(1)=XLH(1)*X(1)+XLH(2)*X(2)
000044          C(2)=0.
000045          RETURN
000047       20 T(1)=XLH(1)*X(1)+XLH(3)*X(2)
000052          T(2)=XLH(2)*X(1)+XLH(4)*X(2)
000055          C(1)=T(1)
000056          C(2)=T(2)
000057          RETURN
000060          ENTRY  DHX
000070          IF (HL) GO TO 30
                C       NONLINEAR GRADIENT COMPUTED HERE
                C       HX(1)=-A/(A*A+B*B)   HX(2)=B/(A*A+B*B)
000072          B=.01
000073          BEN=AMOD(N*B,TWOPI)
000077          CEN=COS(BEN)
000101          SEN=SIN(BEN)
000106          B=X(1)-CEN
000107          A=X(2)-SEN
000111          D=A*A+B*B
000113          C(1)=-A/D
000115          C(2)=B/D
000116          RETURN
000117       30 C(1)=XLH(1)
000121          C(2)=XLH(2)
000122          C(3)=XLH(3)
000124          C(4)=XLH(4)
000125          RETURN
000126          END

SUBPROGRAM LENGTH
000166

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
10      -  000136    20      -  000042    30      -  000120

BLOCK NAMES AND LENGTHS
CM1     -  000012    CM6     -  000002    CM8     -  000017
```

```
KFNLF     544
KFNLF     545
KFNLF     546
KFNLF     547
KFNLF     548
KFNLF     549
KFNLF     550
KFNLF     551
KFNLF     552
26MAR71    38
KFNLF     554
KFNLF     555
KFNLF     556
KFNLF     557
KFNLF     558
KFNLF     559
KFNLF     560
KFNLF     561
KFNLF     562
KFNLF     563
KFNLF     564
KFNLF     565
KFNLF     566
KFNLF     567
KFNLF     568
KFNLF     569
KFNLF     570
KFNLF     571
KFNLF     572
26MAR71    39
KFNLF     574
KFNLF     575
KFNLF     576
KFNLF     577
KFNLF     578
KFNLF     579
KFNLF     580
KFNLF     581
KFNLF     582
KFNLF     583
KFNLF     584
KFNLF     585
KFNLF     586
KFNLF     587
KFNLF     588
```

VARIABLE ASSIGNMENTS

A	-	000164	B	-	000163	BEN	-	000161	CEN	-	000162
D	-	000165	DHX	-	000156	FL	-	000000C03	HL	-	000002C03
NQ	-	000111C01	NT	-	000000C01	SEN	-	000163	SL	-	000001C03
T	-	000156	TWOPI	-	000001C02	XL	-	000003C03	XLH	-	000013C03

START OF CONSTANTS
000131

START OF TEMPORARIES
000133

START OF INDIRECTS
000150

UNUSED COMPILER SPACE
036500

```
                SUBROUTINE  INITLS(MC,TS,JC,H,EJ,IX1,IX2)                          KFNLF     589
000011          REAL  JC(1), EJ(1), H(1), XT(2), S(4), AI(3)                       KFNLF     590
***TT ********************** INITLS
000011          INTEGER  IX1(1), IX2(1)                                            KFNLF     591
***TT ********************** INITLS
000011          LOGICAL  TS, MLF                                                   26MAR71    40
***TT ********************** INITLS
000011          COMMON  / CM1/ NT(3), A11, A12, A22                                KFNLF     593
000011          COMMON  / CM2/ YN2(2), RLAMN2(2), RUN2(4), YN1(2), RLAMN1(2),      KFNLF     594
               1   RUN1(4), PN1(3), YN(2), PN(3)                                  KFNLF     595
000011          COMMON  / CM3/ RT(3), A(3)                                         KFNLF     596
000011          COMMON  / CM4/ X(2)                                                KFNLF     597
000011          COMMON  / CM6/ LT(8), MLF                                          26MAR71    41
000011          COMMON  / CM7/ NM, XM(20), PC(30)                                  KFNLF     598
000011          CALL FINDPT(NM,XM,PC,YN,PN)                                        KFNLF     599
000014          CALL  XFM(PN,RUN1,RLAMN1)                                          KFNLF     600
000017          DO 10 I=1,2                                                        KFNLF     601
000031          RLAMN2(I)=RLAMN1(I)                                                KFNLF     602
000032          RUN2(I)=RUN1(I)                                                    KFNLF     603
000034          YN1(I)=YN(I)                                                       KFNLF     604
000035          YN2(I)=YN(I)                                                       KFNLF     605
000035       10 PN1(I)=PN(I)                                                       KFNLF     606
000037          PN1(3)=PN(3)                                                       KFNLF     607
000040          RUN2(3)=RUN1(3)                                                    KFNLF     608
000041          RUN2(4)=RUN1(4)                                                    KFNLF     609
000043          IF (.NOT.MLF) CALL DENSTY(JC,EJ)                                   26MAR71    42
000052          NN=0                                                               KFNLF     611
000053          IF (MC.LE.1) NN=99                                                 KFNLF     612
000062          XT(1)=GRND1(NN)                                                    KFNLF     613
000064          XT(2)=GRND1(0)                                                     KFNLF     614
000066          IF (NM.LE.1) GO TO 20                                              21MAR71     7
000075          MS=NM*RANF(0.)+1                                                   21MAR71     8
000102          IF(MS.LT.1) MS=1                                                   21MAR71     9
000111          IF (MS.GT.NM) MS=NM                                                21MAR71    10
000114          XC=XM(2*MS-1)                                                      21MAR71    11
000116          YC=XM(2*MS)                                                        21MAR71    12
000120          P11=PC(3*MS-2)                                                     21MAR71    13
000122          P12=PC(3*MS-1)                                                     21MAR71    14
000125          P22=PC(3*MS)                                                       21MAR71    15
000127          SP11=SQRT(P11)                                                     21MAR71    16
000131          SP12=P12/SP11                                                      21MAR71    17
000133          SP22=SQRT(P22-SP12*SP12)                                           21MAR71    18
000137          X(1)=XC+SP11*XT(1)                                                 21MAR71    19
000141          X(2)=YC+SP12*XT(1)+SP22*XT(2)                                      21MAR71    20
000146          GO TO 30                                                           21MAR71    21
000152       20 CALL  XUNPRM(XM,RUN1,RLAMN1,YN1,X)                                 21MAR71    22
000162       30 CALL  SX(X,S,0)                                                    21MAR71    23
000165          CALL  AX(S,A,0)                                                    KFNLF     617
000170          CALL  MINV(A,AI,2)                                                 KFNLF     618
000173          A11=-.5*AI(1)                                                      KFNLF     619
000175          A12=-AI(2)                                                         KFNLF     620
000176          A22=-.5*AI(3)                                                      KFNLF     621
000200          IF (.NOT.TS) CALL  WRITE(MC,J,H,EJ,IX1,IX2)                        KFNLF     622
000210          RETURN                                                             KFNLF     623
000215          END                                                                KFNLF     624
```

SUBPROGRAM LENGTH
000277

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
20 - 000153 30 - 000163

BLOCK NAMES AND LENGTHS
CM1 - 000305 CM2 - 000330 CM3 - 000056 CM4 - 000602
CM6 - 000311 CM7 - 000063

VARIABLE ASSIGNMENTS
A - 000303C03 AI - 000261 A11 - 000003C01 A12 - 000004C01
A22 - 000005C01 I - 000264 IX2 - 000000 LT - 000000C05
MLF - 000010C05 MS - 000266 NM - 000000C06 NN - 000265
NT - 000 00C01 PC - 000025C06 PN - 000025C02 PN1 - 000020C02
P11 - 000271 P12 - 000272 P22 - 000273 RLAMN1 - 000012C02
RLAMN2 - 000002C02 RT - 000000C03 RUN1 - 000014C02 RUN2 - 000004C02
S - 000255 SP11 - 000274 SP12 - 000275 SP22 - 000276
X - 000001C04 XC - 000267 XM - 000001C06 XT - 000253
YC - 000270 YM - 000023C02 YN1 - 000010C02 YN2 - 000000C02

START OF CONSTANTS
000220

START OF TEMPORARIES
000231

START OF INDIRECTS
000246

UNUSED COMPILER SPACE
036000

TT********VARIABLE GIVEN CONFLICTING TYPES
000712,000012,000012

```
            SUBROUTINE  INPUT(MC,N,IX1,IX2,EJ,H,JC,H1,H2,H3)                    KFNLF     625
000014       EAL  EJ(1), H(1), JC(1), H1(1), H2(1), H3(1)                       KFNLF     626
***TT ********************** INPUT
000014      -     INTEGER  IX1(1), IX2(1)                                       KFNLF     627
      C        INPUTS ALL VARIABLES FROM CARDS AND/OR FORMATTED TAPE           KFNLF     628
***TT ********************** INPUT
000014       COMMON  / CM1/ M, NSD, MMAX, AT(3), NPTS, NPQ, NP, NQ, SCANX       KFNLF     629
000014       COMMON  / CM2/ YN2(2), RLAMN2(2), RUN2(4), PN1(2), RLAMN1(2),      KFNLF     630
            1   RUN1(4), PN1(3), YN(2), PN(3)                                   KFNLF     631
000014       COMMON  / CM3/ R(3), A(3), Q(3), SQTQ(3), SQTR(3)                  KFNLF     632
000014       COMMON  / CM4/ X(2), Z(2), U(2), V(2), XC(2), XK(4)                KFNLF     633
000014       COMMON  / CM6/ PI, TWOPI, NSDC, NMC, NMAX, TS, PS, RST, MLF, CP    26MAR71    43
000014       COMMON  / CM7/ NM, XM(20), PC(30)                                  KFNLF     635
000014       COMMON  / CM8/ FL, SL, HL, XLF(4), XLS(4), XLH(4)                  KFNLF     636
000014       COMMON  / CM9/ EMC(200), VMC(300)                                  KFNLF     637
000014       COMMON  /CM10/ IST, IZ(2), XZ(2)                                   KFNLF     638
000014       LOGICAL  SCANX, TS, PS, RST, FL, SL, HL, MLF, CP                   26MAR71    44
000014       INTEGER  RE, PR, TP                                                KFNLF     640
000014       DATA  (RE=5), (PR=6), (TP=7)                                       KFNLF     641
000014       READ (RE,50000)                                                    KFNLF     642
000017  50000 FORMAT(80H                                                        KFNLF     643
            1                    )                                             KFNLF     644
000023       READ (RE,50100) NMC, TS, PS, RST, MLF, CP                          26MAR71    45
000042  50100 FORMAT(I5,5L5)                                                    26MAR71    46
000046       IF (.NOT.MLF) GO TO 10                                             26MAR71    47
000047       TS=.TRUE.                                                          26MAR71    48
00005C     1 WRITE (PR,50000)                                                   26MAR71    49
000054  60000 FORMAT(1H1////////////47X,28HGENERALIZED  TWO-DIMENSIONAL /47X,   KFNLF     648
            1   28HDISCRETE NONLINEAR PREDICTOR //////////49X,                 KFNLF     649
            2   24HCAPTAIN KENNETH O. SENNE /43X,                              KFNLF     650
            3   35HFRANK J. SEILER RESEARCH LABORATORY /48X,                   KFNLF     651
            4   25HAIR FORCE SYSTEMS COMMAND /45X,                             KFNLF     652
            5   32HU.S. AIR FORCE ACADEMY, CO 80840 /53X,                      KFNLF     653
            6   13HFEBRUARY 1971 /1H1)                                         KFNLF     654
000060       WRITE (PR,50000)                                                   KFNLF     655
000063       IF (PST) GO TO 1000                                                KFNLF     656
      C        BEGIN CARD INPUT                                                KFNLF     657
000070     5 READ (RE,50200) M, NMAX, NM, NSD, NSDC, NP, NQ, IST, FL, SL, HL    KFNLF     658
000122  50200 FORMAT(7I5,I15,3L5)                                               KFNLF     659
000126       IF (NMAX.GT.100) NMAX=100                                          KFNLF     660
000131       IQ=1                                                               KFNLF     661
000132       IF (NP.GT.1) IQ=3                                                  KFNLF     662
000136       READ (RE,50300) (Q(I),I=1,IQ)                                      KFNLF     663
000145  50300 FORMAT(8F10.3)                                                    KFNLF     664
000151       IR=1                                                               KFNLF     665
000151       IF (NQ.GT.1) IR=3                                                  KFNLF     666
000155       READ (RE,50300) (R(I),I=1,IR)                                      KFNLF     667
000164       IM=2*NM                                                            KFNLF     668
000165       IP=IM+M                                                            KFNLF     669
000167       READ (RE,50300) (XM(I),I=1,IM), (PC(I),I=1,IP)                     KFNLF     670
000204       IR=IP+1                                                            KFNLF     671
000206       IQ=IQ+1                                                            KFNLF     672
000207       IF (FL) READ (RE,50300) (XLF(I),I=1,4)                             KFNLF     673
000225       IF (SL) READ (RE,50300) (XLS(I),I=1,IQ)                            KFNLF     674
000241       IF (HL) READ (RE,50300) (XLH(I),I=1,IR)                            KFNLF     675
000255       IF (TS) GO TO 103                                                  KFNLF     676
000257       WRITE (TP)  M, NMAX, NM, NSD, NSDC, NP, NQ, IST, FL, SL, HL,       21MAR71    24
            1   Q, R, XLF, XLS, XLH, XM, PC                                     KFNLF     678
```

```
000326   7000  FORMAT(8I20,3L20,68E20.14)                                    KFNLF     679
000332         END FILE TP                                                   KFNLF     680
000334         BACKSPACE  TP                                                 KFNLF     681
               C     BEGIN COMMON FILL                                       KFNLF     682
000336     10  PI=4.*ATAN(1.)                                               KFNLF     683
000342         TWOPI=2.*PI                                                   KFNLF     684
000343         MMAX=2*M+1                                                    KFNLF     685
000345         DO 200 I=1,MMAX                                               KFNLF     686
000360     20  JC(I)=FLOAT((I-M-1)*NSD)/FLOAT(M)                             KFNLF     687
000366         SQTQ(1)=SQRT(Q(1))                                            KFNLF     688
000371         IF (NP.EQ.1) GO TO 300                                        KFNLF     689
000377         SQTQ(2)=Q(2)/SQTQ(1)                                          KFNLF     690
000400         SQTQ(3)=SQRT(Q(3)-SQTQ(2)*SQTQ(2))                            KFNLF     691
000410     30  SQTR(1)=SQRT(R(1))                                            KFNLF     692
000413         IF (NQ.EQ.1) GO TO 400                                        KFNLF     693
000421         SQTR(2)=R(2)/SQTR(1)                                          KFNLF     694
000422         SQTR(3)=SQRT(R(3)-SQTR(2)*SQTR(2))                            KFNLF     695
000432     40  IF (TS) GO TO 500                                             KFNLF     696
000434         WRITE (TP)       PI, TWOPI, MMAX, SQTQ, SQTR, (JC(I),I=1,MMAX)  7MAR71    33
000454   7010  FORMAT(2E2).14,I20,57E20.14)                                  KFNLF     698
000460         END FILE  TP                                                  KFNLF     699
000462         BACKSPACE  TP                                                 KFNLF     700
               C        ZERO MONTE CARLO TABLE                               KFNLF     701
000464     50  N=1                                                           KFNLF     702
000465         MC=0                                                          KFNLF     703
000465         DO 600 I=1,200                                                KFNLF     704
000472         SMC(I)=0.                                                     KFNLF     705
000473     60  VMC(I)=0.                                                     KFNLF     706
000474         DO 700 I=201,300                                              KFNLF     707
000504     70  VMC(I)=0.                                                     KFNLF     708
000506         IF (.NOT.TS) CALL  MCRITE(MC)                                 KFNLF     709
000514         MC=1                                                          KFNLF     710
               C        PRINT PROBLEM PARAMETERS                             KFNLF     711
000521     80  IF (TS) WRITE  (PR,6010)                                      KFNLF     712
000532   6010  FORMAT(//52X,22HTAPE OUTPUT SUPPRESSED /)                     KFNLF     713
000532         IF (PS) WRITE  (PR,6020)                                      KFNLF     714
000543   6020  FORMAT(//45X,31HSAMPLE PATH PRINTOUT SUPPRESSED /)            KFNLF     715
000543         IF (MLF) WRITE (PR,6025)                                      26MAR71    50
000554   6025  FORMAT(//44X,32HLINEARIZED PREDICTOR MONTE CARLO  )           26MAR71    51
000554         WRITE (PR,6030) NMC, MMAX, M, NSD, NSDC, NP, NQ, IST          KFNLF     716
000600   6030  FORMAT(//52X,18HPROBLEM PARAMETERS ///21X,                    KFNLF     717
              1   19HMAXIMUM MONTE CARLO ,33X,16HSAMPLES PER PATH //28X,I5,44X,I5  KFNLF  718
              2   ///16X,9HGRID SIZE ,29X,15HNO. GRID SIGMAS, 18X,          KFNLF     719
              3   27HNO. TRACKING ELLIPSE SIGMAS//17X,3HM = ,I5,31X,5HNSD = ,I5,  KFNLF  720
              4   30X,6HNSDC = ,I5///13X,16HNUMBER OF INPUTS ,24X           KFNLF     721
              5   17HNUMBER OF OUTPUTS ,21X,21HINITIAL RANDOM NUMBER //16X,4HNP =  KFNLF  722
              6   ,I5,31X,4HNQ = ,I5,29X,I15///)                           KFNLF     723
000604         IF (NP.EQ.1) WRITE (PR,6040) Q(1)                             KFNLF     724
000617   6040  FORMAT(54X,13HPROCESS NOISE /54X,13HCOVARIANCE  Q //(51X,2F10.3))  KFNLF  725
000617         IF (NP.NE.1) WRITE (PR,6040) Q(1), Q(2), Q(2), Q(3)           KFNLF     726
000641         IF (NQ.EQ.1) WRITE (PR,6050) R(1)                             KFNLF     727
000655   6050  FORMAT(//52X,17HMEASUREMENT NOISE /54X,13HCOVARIANCE  R //    KFNLF     728
              1   (51X,2F10.3))                                             KFNLF     729
000655         IF (NQ.NE.1) WRITE (PR,6050) R(1), R(2), R(2), R(3)           KFNLF     730
000677         WRITE (PR,6060)                                               KFNLF     731
000703   6060  FORMAT(///51X,17HLINEAR PARAMETERS)                           KFNLF     732
000707         IP=2*NP                                                       KFNLF     733
000710         IQ=2*NQ                                                       KFNLF     734
```

```
000711          IF (FL) WRITE (PR,6070 ) (XLF(I),I=1,4)                  KFNLF    735
000725    6070) FORMAT(//45X,1HF,9X,2F10.3/55X,2F10.3)                   KFNLF    736
000725          IF (SL) WRITE (PR,6080) (XLS(I),I=1,IP)                  KFNLF    737
000741    6080) FORMAT(//43X,5HSIGMA,7X,2F10.3/55X,2F10.3)               KFNLF    738
000741          IF (HL) WRITE (PR,6090) (XLH(I),I=1,IQ)                  KFNLF    739
000755    6090) FORMAT(//45X,1HH,9X,2F10.3/55X,2F10.3)                   KFNLF    740
000755          WRITE (PR,61000)                                         KFNLF    741
000761    6100) FORMAT(1H1)                                              KFNLF    742
000765          RETURN                                                   KFNLF    743
          C         TAPE INPUT BEGINS HERE                             KFNLF    744
000765     100) READ (TP)   M, NMAX, N1, NSD, NSDC, NP, NQ, IST, FL,     21MAR71   25
              1   SL, HL, Q, R, XLF, XLS, XLH, XM, PC                  KFNLF    746
001034          IF (EOF,TP) 50,1010                                      KFNLF    747
001043     101) READ (TP)            PI, TWOPI, MMAX, SQTQ, SQTR,       7MAR71    34
              1   (JC(I),I=1,MMAX)                                     KFNLF    749
001063          IF (EOF,TP) 100,1020                                     KFNLF    750
001072    1020) IS=1                                                     KFNLF    751
001073     105) IP=NMC+1                                                 KFNLF    752
001075          IN=NMAX+1                                                KFNLF    753
001077          DO 140) I=IS,IP                                          KFNLF    754
001101          MC=I-1                                                   KFNLF    755
001102          IF (MCREAD(MC)) 1100,1200,1200                           KFNLF    756
001110     110) IF (MC) 50),500,1700                                     KFNLF    757
001112     120) MC=I                                                     KFNLF    758
001113          DO 140) J=1,IN                                           KFNLF    759
001115          N=J-1                                                    KFNLF    760
001116          IF (MCREAD(MC,N,H,EJ,IX1,IX2)) 1500,1300,1300            KFNLF    761
001126     130) IF (N.NE.0) CALL  MCAV(MC,N,X,YN,EMC,VMC)                KFNLF    762
001136     140) CONTINUE                                                 KFNLF    763
001143          IS=IP+1                                                  KFNLF    764
001144          NMC=NMC+10                                               KFNLF    765
001146          GO TO 1050                                               KFNLF    766
001147     150) IF (N.EQ.0) GO TO 1710                                   KFNLF    767
001150          CALL MCAV(MC,N,X,YN,EMC,VMC)                             KFNLF    768
001153          N1=N+1                                                   KFNLF    769
001160          GO TO 1800                                               KFNLF    770
001161     170) N1=N                                                     KFNLF    771
001162     180) WRITE (PR,61100) MC, N1                                  KFNLF    772
001172   61100) FORMAT(//32X,31HTAPE RESTART ON MONTE CARLO NO. ,I5,     KFNLF    773
              1   16H WITH SAMPLE NO. ,I5/)                            KFNLF    774
001176          GO TO 300                                                KFNLF    775
001176          END                                                      KFNLF    776
```

```
SUBPROGRAM LENGTH
001532

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
10     - 000351    50     - 000371    100    - 000737    300    - 000411
400    - 000433    500    - 000465    800    - 000522    1000   - 000766
1010   - 001044    1020   - 001073    1050   - 001074    1100   - 001111
1200   - 001113    1300   - 001127    1400   - 001137    1500   - 001150
1700   - 001162    1800   - 001163    50000  - 001202    50100  - 001214
50200  - 001271    50300  - 001274    60000  - 001216    60100  - 001315
60200  - 001322    60250  - 001330    60300  - 001336    60400  - 001412
```

```
60500  -  001422    60600  -  001423    60700  -  001440    60800  -  001445
60900  -  001452    61000  -  001457    61100  -  001461    70000  -  001302
70100  -  001311

BLOCK NAMES AND LENGTHS
CM1    -  000313    CM2    -  000031    CM3    -  000017    CM4    -  000016
CM6    -  000312    CM7    -  000063    CM8    -  000017    CM9    -  000764
CM10   -  000005

VARIABLE ASSIGNMENTS
A      -  000003C03 AT     -  000003C01 CP     -  000011C05 EMC    -  000000C10
FL     -  000000C07 HL     -  000002C07 H1     -  000001    H2     -  000002
H3     -  000003    I      -  001522    IH     -  001524    IN     -  001527
IP     -  001525    IQ     -  001521    IR     -  001523    IS     -  001526
IST    -  000000C11 IZ     -  000001C11 J      -  001530    JC     -  000000
K      -  000000C01 MLF    -  000010C05 MMAX   -  000002C01 NM     -  000000C05
NMAX   -  000004C05 NMC    -  000003C05 NP     -  000010C01 NQ     -  000011C01
NSO    -  000001C01 NSOC   -  000002C05 N1     -  001531    PC     -  000025C06
PI     -  000000C05 PN     -  000025C02 PN1    -  000020C02 PR     -  001517
PS     -  000006C05 Q      -  000006C03 P      -  000000C03 RE     -  001516
RLAMN1 -  000012C02 RLAMN2 -  000002C 2 RST    -  000007C05 RUN1   -  000014C02
RUN2   -  000,04C02 SCANX  -  000012C01 SL     -  000001C07 SQTQ   -  000011C03
SQTR   -  000014C03 TP     -  001520    TS     -  000005C05 TWOPI  -  000001C05
U      -  000004C04 V      -  000006C04 VMC    -  000310C10 X      -  000000C04
XC     -  000010C04 XK     -  000012C04 XLF    -  000003C07 XLH    -  000013C07
XLS    -  000 07C07 XM     -  000001C06 XZ     -  000007C11 YN     -  000023C02
YN1    -  000.1'C02 YN2    -  000000C02 Z      -  000002C04

START OF CONSTANTS
001201

START OF TEMPORARIES
001473

START OF INDIRECTS
001510

UNUSED COMPILER SPACE
032300

T1*******VARIABLE GIVEN CONFLICTING TYPES
    000115,000015
```

```
000010          SUBROUTINE  MCAV(MC,N,X,YN,EMC,VMC)
000011          DIMENSION  X(1), YN(1), EMC(1), VMC(1), Y(2)
000012          N1= 2*(N-1)
000013          N2= 3*(N-1)
000015          Y(1)=X(1)-YN(1)
000020          Y(2)=X(2)-YN(2)
000032          EMC(N1+1)=CUMAVG(EMC(N1+1),Y(1),MC)
000046          EMC(N1+2)=CUMAVG(EMC(N1+2),Y(2),MC)
000061          VMC(N2+1)=CUMAVG(VMC(N2+1),Y(1)*Y(1),MC)
000075          VMC(N2+2)=CUMAVG(VMC(N2+2),Y(1)*Y(2),MC)
000111          VMC(N2+3)=CUMAVG(VMC(N2+3),Y(2)*Y(2),MC)
000111          RETURN
000111          END
```

```
KFNLF   777
KFNLF   778
KFNLF   779
KFNLF   780
KFNLF   781
KFNLF   782
KFNLF   783
KFNLF   784
KFNLF   785
KFNLF   786
KFNLF   787
KFNLF   788
KFNLF   789
```

SUBPROGRAM LENGTH
000155

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

BLOCK NAMES AND LENGTHS

VARIABLE ASSIGNMENTS
N1 - 000153 N2 - 000154 Y - 000151

START OF CONSTANTS
000114

START OF TEMPORARIES
000115

START OF INDIRECTS
000136

UNUSED COMPILER SPACE
036700

```
                FUNCTION  MCREAD(MC)                                                KFNLF     790
000002          COMMON  / CM9/ EMC(200), VMC(300)                                   KFNLF     791
000002          INTEGER  TP                                                         KFNLF     792
000002          DATA (TP=7)                                                         KFNLF     793
000002          NRH=0                                                               KFNLF     794
000003          READ (TP)                MCH, N, NR, EMC                            7MAR71     35
000016    70000 FORMAT(3I20,200E20.14)                                              KFNLF     796
000017          IF (EOF,TP) 40,10                                                   KFNLF     797
000022       10 NRH=NRH+1                                                           KFNLF     798
000024          IF (N.NE.-1.OR.MC.NE.MCH.OR.NR.NE.NRH) GO TO 50                     KFNLF     799
000036          READ (TP)                MCH, N, NR, (VMC(I),I=1,150)               7MAR71     36
000051          IF (EOF,TP) 40,20                                                   KFNLF     801
000055       20 NRH=NRH+1                                                           KFNLF     802
000057          IF (N.NE.-1.OR.MC.NE.MCH.OR.NR.NE.NRH) GO TO 50                     KFNLF     803
000071          READ (TP)                MCH, N, NR, (VMC(I),I=151,300)             7MAR71     37
000104          IF (EOF,TP) 40,30                                                   KFNLF     805
000110       30 NRH=NRH+1                                                           KFNLF     806
000112          IF (N.NE.-1.OR.MC.NE.MCH.OR.NR.NE.NRH) GO TO 50                     KFNLF     807
000124          MCREAD=1                                                            KFNLF     808
000125          RETURN                                                              KFNLF     809
000125       40 BACKSPACE  TP                                                       KFNLF     810
000127       50 MCREAD=-1                                                           KFNLF     811
000130       60 IF (NRH.EQ.0) RETURN                                                KFNLF     812
000133          BACKSPACE  TP                                                       KFNLF     813
000135          NRH=NRH-1                                                           KFNLF     814
000137          GO TO 60                                                            KFNLF     815
000137          END                                                                 KFNLF     816


SUBPROGRAM LENGTH
000164

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
10      -  000023    20      -  000056    30      -  000111    40      -  000126
50      -  000130    6       -  000131    70000   -  000144

BLOCK NAMES AND LENGTHS
CM9     -  000764

VARIABLE ASSIGNMENTS
EMC     -  000300C01 I       -  000163    MCH     -  000150    MCREAD -  000155
N       -  000161    NR      -  000162    NRH     -  000157    TP      -  000156
VMC     -  000310C01

START OF CONSTANTS
000142

START OF TEMPORARIES
000147

START OF INDIRECTS
000155

UNUSED COMPILER SPACE
036600
```

```
                SUBROUTINE  MCRITE(MC)
000002          COMMON  / CM9/ EMC(2,0), VMC(300)
000002          INTEGER  TP
000002          DATA (TP=7)
000002          N=-1
000003          NR=1
000004          WRITE (TP)       MC, N, NR, EMC
000017    7000) FORMAT(3I20,247E20.14)
000020          END FILE  TP
000022          BACKSPACE  TP
000024          NR=2
000025          WRITE (TP) MC, N, NR, (VMC(I),I=1,150)
000040          END FILE  TP
000043          BACKSPACE  TP
000045          NR=3
000046          WRITE (TP) MC, N, NR, (VMC(I),I=151,300)
000061          END FILE  TP
000064          BACKSPACE  TP
000066          RETURN
000067          END
```

```
KFNLF     817
KFNLF     818
KFNLF     819
KFNLF     820
KFNLF     821
KFNLF     822
7MAR71     38
KFNLF     824
KFNLF     825
KFNLF     826
KFNLF     827
7MAR71     39
KFNLF     829
KFNLF     830
KFNLF     831
7MAR71     40
KFNLF     833
KFNLF     834
KFNLF     835
KFNLF     836
```


SUBPROGRAM LENGTH
000107

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
70000 - 000074

BLOCK NAMES AND LENGTHS
CM9   - 000764

VARIABLE ASSIGNMENTS
EMC   - 000000C01 I   - 000106   N   - 000104   NR   - 000105
TP   - 000103   VMC   - 00001JC.1

START OF CONSTANTS
000072

START OF TEMPORARIES
000077

START OF INDIRECTS
000103

UNUSED COMPILER SPACE
037000

```
                SUBROUTINE  MEAS(N)                                          KFNLF    837
          C        GENERATES MEASURED OUTPUT VARIABLES                       KFNLF    838
000002          COMMON  / CM1/ IT(3), AT(3), IT1(3), NQ                      KFNLF    839
000002          COMMON  / CM3/ TM(12), SQTR(3)                               KFNLF    840
000002          COMMON  / CM4/ X(2), Z(2), U(2), V(2)                        KFNLF    841
000002          DIMENSION  H(2)                                              KFNLF    842
000002          DO 10 I=1,NQ                                                 KFNLF    843
000004       10 V(I)=GRN01(0)                                                KFNLF    844
000012          VT=V(1)                                                      KFNLF    845
000013          V(1)=SQTR(1)*V(1)                                            KFNLF    846
000015          IF (NQ.GT.1) V(2)=SQTR(2)*VT+SQTR(3)*V(2)                    KFNLF    847
000023          CALL  HX(X,H,N)                                              KFNLF    848
000026          DO 20 I=1,NQ                                                 KFNLF    849
000035       20  Z(I)=H(I)+V(I)                                              KFNLF    850
000037          RETURN                                                       KFNLF    851
000040          END                                                          KFNLF    852


SUBPROGRAM LENGTH
000056

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

BLOCK NAMES AND LENGTHS
CM1     -  000012    CM3     -  000017    CM4     -  000010

VARIABLE ASSIGNMENTS
AT      -  000003C01 H       -  000052    I       -  000054    IT      -  000000C01
IT1     -  000006C01 NQ      -  000011C01 SQTR    -  000014C02 TM      -  000000C02
U       -  000004C03 V       -  000006C03 VT      -  000055    X       -  000000C03
Z       -  000002C03

START OF CONSTANTS
000043

START OF TEMPORARIES
000045

START OF INDIRECTS
000051

UNUSED COMPILER SPACE
037000
```

```
                SUBROUTINE  MINV(A,AI,N)                                  KFNLF    853
000005          DIMENSION  A(1), AI(1)                                    KFNLF    854
          C        COMPUTES ACTUAL OR MOORE-PENROSE PSEUDO INVERSE OF A   KFNLF    855
000005          IF (N.GT.1) GO TO 10                                      KFNLF    856
000010          AI(1)=0.                                                  KFNLF    857
000010          IF  (ABS(A(1)).GT.1.E-40) AI(1)=1./A(1)                   KFNLF    858
000015          RETURN                                                    KFNLF    859
000016       10 D=A(1)*A(3)-A(2)*A(2)                                     KFNLF    860
000022          IF (ABS(D).LT.1.E-40) GO TO 20                            KFNLF    861
000025          AT=A(3)/D                                                 KFNLF    862
000026          AI(2)=-A(2)/D                                             KFNLF    863
000030          AI(3)=A(1)/D                                              KFNLF    864
000032          AI(1)=AT                                                  KFNLF    865
000033          RETURN                                                    KFNLF    866
000033       20 D=ABS(A(1)+A(3))                                          KFNLF    867
000035          IF (D.GT.1.E-40) GO TO 30                                 KFNLF    868
000041          AI(1)=0.                                                  KFNLF    869
000041          AI(2)=0.                                                  KFNLF    870
000042          AI(3)=0.                                                  KFNLF    871
000043          RETURN                                                    KFNLF    872
000045       30 D=D*D                                                     KFNLF    873
000045          AI(1)=A(1)/D                                              KFNLF    874
000047          AI(2)=A(2)/D                                              KFNLF    875
000050          AI(3)=A(3)/D                                              KFNLF    876
000052          RETURN                                                    KFNLF    877
000052          END                                                       KFNLF    878

SUBPROGRAM LENGTH
000077

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
10     -  000017    20     -  000034    30     -  000045

BLOCK NAMES AND LENGTHS

VARIABLE ASSIGNMENTS
AT     -  000076    D      -  000075

START OF CONSTANTS
000055

START OF TEMPORARIES
000060

START OF INDIRECTS
000064

UNUSED COMPILER SPACE
037000
```

```
000005          REAL FUNCTION  NORMAL(X,XM,C)                          KFNLF     879
000005          DIMENSION  X(1), XM(1), C(1)                           KFNLF     880
000006          Y1=X(1)-XM(1)                                          KFNLF     881
000010          Y2=X(2)-XM(2)                                          KFNLF     882
000025          NORMAL=EXP(-((Y1*Y1*C(1)+2.*Y1*Y2*C(2)+Y2*Y2*C(3))/2.) KFNLF     883
000025          RETURN                                                 KFNLF     884
000025          END                                                    KFNLF     885


SUBPROGRAM LENGTH
000054

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

BLOCK NAMES AND LENGTHS

VARIABLE ASSIGNMENTS
NORMAL - 000051    Y1     - 000052    Y2     - 000053

START OF CONSTANTS
000030

START OF TEMPORARIES
000033

START OF INDIRECTS
000042

UNUSED COMPILER SPACE
037100
```

```
                FUNCTION  NREAD(MC,N,W,EJ,IX1,IX2)                                  KFNLF     886
000010          DIMENSION  W(1), EJ(1), IX1(1), IX2(1), AI(3), S(4)                 KFNLF     887
000010          COMMON  / CM1/ NT(2), MMAX, A11, A12, A22, NPTS, NPQ, NT1(2),       KFNLF     888
               1   SCANX                                                           KFNLF     889
000010          COMMON  / CM2/ YN2(2), RLAMN2(2), RUN2(4), YN1(2), RLAMN1(2),       KFNLF     890
               1   RUN1(4), PN1(3), YN(2), PN(3), XKN(3)                           KFNLF     891
000010          COMMON  / CM3/ RT(3), A(3)                                          KFNLF     892
000010          COMMON  / CM4/ X(2), Z(2), U(2), V(2), XC(2), XK(4)                 KFNLF     893
000010          COMMON  / CM5/ C                                                    KFNLF     894
000010          COMMON  /CM10/ IST, IZ(2), XZ(2)                                    KFNLF     895
000010          INTEGER  TP                                                         KFNLF     896
000010          LOGICAL  SCANX                                                      KFNLF     897
000010          DATA  (TP=7)                                                        KFNLF     898
000010          MM=MMAX*MMAX                                                        KFNLF     899
000011          NWR=MM/247                                                          KFNLF     900
000015          NLR=MOD(MM,247)                                                     KFNLF     901
000020          NBS=1                                                               KFNLF     902
000021       10 NRH=0                                                               KFNLF     903
000022          READ (TP)       MCH, NH, NR, NPTS, NPQ, IZ,      XZ, SCANX, C,      21MAR71    26
               1   U, X, V, Z, XC, XK, XKN, PN, PN1, PN2, RLAMN1, RLAMN2,          KFNLF     905
               2   RUN1, RUN2, PN, PN1, (IX1(I),I=1,MMAX), (IX2(I),I=1,MMAX)       KFNLF     906
000123    7000 FORMAT(7I20,2E20.14,L20,42E20.14,198I20)                            KFNLF     907
000127          IF (EOF,TP) 130,20                                                  KFNLF     908
000132       20 NRH=NRH+1                                                           KFNLF     909
000134          IF (N.NE.NH.OR.MC.NE.MCH.OR.NR.NE.NRH) GO TO 140                    KFNLF     910
000146          NU=0                                                                KFNLF     911
000146          IF (NWR.LE.0) GO TO 50                                              KFNLF     912
000150          DO 40 I=1,NWR                                                       KFNLF     913
000152          NL=(I-1)*247+1                                                      KFNLF     914
000155          NU=I*247                                                            KFNLF     915
000157          READ (TP)                MCH, NH, NR, (W(J),J=NL,NU)                7MAR71     41
000177    7010 FORMAT(3I21,247E20.14)                                               KFNLF     917
000203          IF (EOF,TP) 130,30                                                  KFNLF     918
000206       30 NRH=NRH+1                                                           KFNLF     919
000210          IF (N.NE.NH.OR.MC.NE.MCH.OR.NR.NE.NRH) GO TO 140                    KFNLF     920
000222       40 CONTINUE                                                            KFNLF     921
000224       50 IF (NLR.EQ.0) GO TO 70                                              KFNLF     922
000225          NL=NU+1                                                             KFNLF     923
000227          NU=NU+NLR                                                           KFNLF     924
000230          READ (TP)                MCH, NH, NR, (W(J),J=NL,NU)                7MAR71     42
000250          IF (EOF,TP) 130,60                                                  KFNLF     926
000257       60 NRH=NRH+1                                                           KFNLF     927
000261          IF (N.NE.NH.OR.MC.NE.MCH.OR.NR.NE.NRH) GO TO 140                    KFNLF     928
000273       70 NU=0                                                                KFNLF     929
000274          IF (NWR.EQ.0) GO TO 10                                              KFNLF     930
000275          DO 90 I=1,NWR                                                       KFNLF     931
000277          NL=(I-1)*247+1                                                      KFNLF     932
000302          NU=247*I                                                            KFNLF     933
000304          READ (TP)                MCH, NH, NR, (EJ(J),J=NL,NU)               7MAR71     43
000324          IF (EOF,TP) 130,80                                                  KFNLF     935
000333       80 NRH=NRH+1                                                           KFNLF     936
000335          IF (N.NE.NH.OR.MC.NE.MCH.OR.NR.NE.NRH) GO TO 140                    KFNLF     937
000347       90 CONTINUE                                                            KFNLF     938
000351      100 IF (NLR.EQ.0) GO TO 120                                             KFNLF     939
000352          NL=NU+1                                                             KFNLF     940
000354          NU=NU+NLR                                                           KFNLF     941
000355          READ (TP)                MCH, NH, NR, (EJ(J),J=NL,NU)               7MAR71     44
000375          IF (EOF,TP) 130,110                                                 KFNLF     943
```

```
000404     110 NRH=NRH+1                                                    KFNLF     944
000406         IF (N.NE.NH.OR.MC.NE.MCH.OR.NR.NE.NRH) GO TO 140             KFNLF     945
000420     120 IF (NBS.NE.0) NREAD=1                                        KFNLF     946
000422         CALL  SX(X,S,N)                                              KFNLF     947
000425         CALL  AX(S,A,N)                                              KFNLF     948
000434         CALL  MINV(A,AI,2)                                           KFNLF     949
000437         A11=-.5*AI(1)                                                KFNLF     950
000441         A12=-AI(2)                                                   KFNLF     951
000442         A22=-.5*AI(3)                                                KFNLF     952
000444         RETURN                                                       KFNLF     953
000446     130 BACKSPACE  TP                                                KFNLF     954
000450     140 NREAD=-1                                                     KFNLF     955
000451     150 IF (NRH.LE.0) GO TO 160                                      KFNLF     956
000453         BACKSPACE  TP                                                KFNLF     957
000455         NRH=NRH-1                                                    KFNLF     958
000457         GO TO 150                                                    KFNLF     959
000457     160 IF (N.EQ.0) RETURN                                           KFNLF     960
000462         IF (NBS.EQ.0) GO TO 10                                       KFNLF     961
000463         BACKSPACE  TP                                                KFNLF     962
000465         READ (TP) MC, N, NRH                                         7MAR71     45
000476         NBS=0                                                        KFNLF     964
000477         GO TO 150                                                    KFNLF     965
000503         END                                                          KFNLF     966
```

Reproduced from best available copy.

```
SUBPROGRAM LENGTH
000555

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
10      -  000122    20      -  000133    30      -  000207    50      -  000225
60      -  000262    70      -  000274    80      -  000334    100     -  000352
110     -  000405    120     -  000421    130     -  000447    140     -  000451
150     -  000457    160     -  000463    70000   -  000512    70100   -  000517

BLOCK NAMES AND LENGTHS
CM1     -  000013    CM2     -  000033    CM3     -  000006    CM4     -  000016
CM5     -  000001    CM10    -  000005

VARIABLE ASSIGNMENTS
A       -  000103C03 AI      -  000531    A11     -  000003C01 A12     -  000004C01
A22     -  000005C01 C       -  000001C05 I       -  000551    IZ      -  000001C06
J       -  000554    MCH     -  000546    MM      -  000541    MMAX    -  000002C01
NBS     -  000544    NH      -  000547    NL      -  000553    NLR     -  000543
NPQ     -  000107C01 NPTS    -  000106C01 NR      -  000553    NREAD   -  000530
NRH     -  000545    NT      -  000000C01 NT1     -  000010C01 NU      -  000552
NWR     -  000542    PN      -  000025C02 PN1     -  000020C02 RLAMN1  -  000112C02
RLAMN2  -  000102C02 RT      -  000100C03 RUN1    -  000014C02 RUN2    -  000004C02
S       -  000534    SCANX   -  000012C01 TP      -  000540    U       -  000004C04
V       -  000106C04 X       -  000000C04 XC      -  000010C04 XK      -  000112C04
XKN     -  000103C02 XZ      -  000003C 6 ZN      -  000023C02 ZN1     -  000110C02
ZN2     -  000021C02 Z       -  000002C 4

START OF CONSTANTS
000506
```

```
                SUBROUTINE  NRITE(MC,N,W,EJ,IX1,IX2)                                KFNLF     967
000010          DIMENSION  W(1), EJ(1), IX1(1), IX2(1)                              KFNLF     968
000010          COMMON  / CM1/ NT(2), MMAX, AT(3), NPTS, NPQ, NT1(2), SCANX         KFNLF     969
000010          COMMON  / CM2/ YN2(2), RLAMN2(2), RUN2(4), YN1(2), RLAMN1(2),       KFNLF     970
                1   RUN1(4), PN1(3), YN(2), PN(3), XKN(3)                           KFNLF     971
000010          COMMON  / CM4/ X(2), Z(2), U(2), V(2), XC(2), XK(4)                 KFNLF     972
000010          COMMON  / CM5/ C                                                    KFNLF     973
000010          COMMON  /CM10/ IST, IZ(2), XZ(2)                                    KFNLF     974
000010          INTEGER  TP                                                         KFNLF     975
000010          LOGICAL  SCANX                                                      KFNLF     976
000010          DATA  (TP=7)                                                        KFNLF     977
000010          MM=MMAX*MMAX                                                        KFNLF     978
000011          NWR=MM/247                                                          KFNLF     979
000015          NLR=MOD(MM,247)                                                     KFNLF     980
000020          NR=1                                                                KFNLF     981
000021          WRITE (TP)  MC, N, NR, NPTS, NPQ, IZ, XZ,  SCANX, C,               21MAR71    27
                1   U, X, V, Z, XC, XK, XKN, YN, YN1, YN2, RLAMN1, RLAMN2,          KFNLF     983
                2   RUN1, RUN2, PN, PN1, (IX1(I),I=1,MMAX), (IX2(I),I=1,MMAX)       KFNLF     984
000122   70000  FORMAT(7I20,2E20.14,L20,42E20.14,198I20)                            KFNLF     985
000126          END FILE  TP                                                        KFNLF     986
000130          BACKSPACE  TP                                                       KFNLF     987
000132          NR=NR+1                                                             KFNLF     988
000134          NU=0                                                                KFNLF     989
000134          IF (NWR.EQ.0) GO TO 20                                              KFNLF     990
000136          DO 10 I=1,NWR                                                       KFNLF     991
000137          NL=(I-1)*247+1                                                      KFNLF     992
000142          NU=I*247                                                            KFNLF     993
000144          WRITE (TP) MC, N, NR, (W(J),J=NL,NU)                                7MAR71     46
000164   70100  FORMAT(3I20,247E20.14)                                              KFNLF     995
000170          END FILE  TP                                                        KFNLF     996
000172          BACKSPACE  TP                                                       KFNLF     997
000174       10 NR=NR+1                                                             KFNLF     998
000200       20 IF (NLR.EQ.0) GO TO 30                                              KFNLF     999
000201          NL=NU+1                                                             KFNLF    1000
000203          NU=NU+NLR                                                           KFNLF    1001
000204          WRITE (TP) MC, N, NR, (W(J),J=NL,NU)                                7MAR71     47
000224          END FILE  TP                                                        KFNLF    1003
000232          BACKSPACE  TP                                                       KFNLF    1004
000234          NR=NR+1                                                             KFNLF    1005
000236       30 NU=0                                                                KFNLF    1006
000237          IF (NWR.EQ.0) GO TO 50                                              KFNLF    1007
000240          DO 40 I=1,NWR                                                       KFNLF    1008
000242          NL=(I-1)*247+1                                                      KFNLF    1009
000245          NU=247*I                                                            KFNLF    1010
000247          WRITE (TP) MC, N, NR, (EJ(J),J=NL,NU)                               7MAR71     48
000267          END FILE  TP                                                        KFNLF    1012
000275          BACKSPACE  TP                                                       KFNLF    1013
000277       40 NR=NR+1                                                             KFNLF    1014
000303       50 IF (NLR.EQ.0) RETURN                                                KFNLF    1015
000305          NL=NU+1                                                             KFNLF    1016
000307          NU=NU+NLR                                                           KFNLF    1017
000310          WRITE (TP) MC, N, NR, (EJ(J),J=NL,NU)                               7MAR71     49
000330          END FILE  TP                                                        KFNLF    1019
000336          BACKSPACE  TP                                                       KFNLF    1020
000340          RETURN                                                              KFNLF    1021
000341          END                                                                 KFNLF    1022
```

SUBPROGRAM LENGTH
000377

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

20	-	000201	30	-	000237	50	-	000304
70100	-	000355						

BLOCK NAMES AND LENGTHS

CM1	-	000313	CM2	-	000033	CM4	-	000016
CM10	-	000005						

VARIABLE ASSIGNMENTS

AT	-	000303C01	C	-	000000C04	I	-	000373
J	-	000375	MM	-	000367	MMAX	-	000102C01
NLR	-	000371	NPQ	-	000007C01	NPTS	-	000006C01
NT	-	000300C01	NT1	-	000010C01	NU	-	000374
PN	-	000125C02	PN1	-	000020C02	RLAMN1	-	000312C02
RUN1	-	000014C02	RUN2	-	000004C02	SCANX	-	000012C01
U	-	000004C03	V	-	000006C03	X	-	000000C03
XK	-	000312C03	XKN	-	000030C02	XZ	-	000003C05
ZN1	-	000017C02	ZN2	-	000000C02	Z	-	000002C03

IZ	-	000001C05						
NL	-	000375						
NR	-	000372						
NWR	-	000370						
RLAMN2	-	000C02C02						
TP	-	000366						
XC	-	000010C03						
YN	-	000123C02						

START OF CONSTANTS
000344

START OF TEMPORARIES
000360

START OF INDIRECTS
000366

UNUSED COMPILER SPACE
035600

```
                SUBROUTINE  OUTPUT(N)                                          KFNLF    1023
000002           OMMON  / CM1/ NT(6), NPTS                                     KFNLF    1024
000002          COMMON  / CM2/ YT(8), YN1(2), RLAMN1(2), RUN1(4), PN1(3), YN(2),KFNLF    1025
                1   PN(3), XKN(3)                                              KFNLF    1026
000002          COMMON  / CM4/ X(2), Z(2), U(2), V(2)                          KFNLF    1027
000002          COMMON  / CM6/ PI                                              KFNLF    1028
000002          INTEGER  PR                                                    KFNLF    1029
000002          DATA  (PR=5)                                                   KFNLF    1030
000002          PHI=ATAN2(RUN1(2),RUN1(1))*180./PI                             KFNLF    1031
000007          IF (N.GT.0) GO TO 10                                           KFNLF    1032
000013          WRITE (PR,60000) N, X(1), YN(1), PN(1), PN(2), YN1(1), PN1(1), KFNLF    1033
                1   PN1(2), RLAMN1(1), X(2), YN(2), PN(2), PN(3), YN1(2), PN1(2),KFNLF    1034
                2   PN1(3), RLAMN1(2), PHI                                     KFNLF    1035
000062   6000   FORMAT(I4,31X,5E10.3/35X,8E10.3,F5.0/)                         KFNLF    1036
000063          RETURN                                                         KFNLF    1037
000064       10 WRITE (PR,60100) N, V(1), Z(1), U(1), X(1), YN(1), PN(1), PN(2),KFNLF    1038
                1   YN1(1), PN1(1), PN1(2), RLAMN1(1), NPTS, V(2), Z(2), U(2), KFNLF    1039
                2   X(2), YN(2), PN(2), PN(3), YN1(2), PN1(2), PN1(3), RLAMN1(2),KFNLF    1040
                3   PHI                                                        KFNLF    1041
000152   60100  FORMAT(I4,1X,11E10.3,I5/5X,11E10.3,F5.0/)                      KFNLF    1042
000153          RETURN                                                         KFNLF    1043
000154          END                                                            KFNLF    1044
```

SUBPROGRAM LENGTH
000203

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
10 - 000065 60000 - 000161 60100 - 000166

BLOCK NAMES AND LENGTHS
CM1 - 000007 CM2 - 000033 CM4 - 000010 CM6 - 000001

VARIABLE ASSIGNMENTS
NPTS - 000006C01 NT - 000001C01 PHI - 000202 PI - 000000C04
PN - 000025C02 PN1 - 000021C02 PR - 000201 RLAMN1 - 000012C02
RUN1 - 000014C02 U - 000004C03 V - 000006C03 X - 000000C03
XKN - 000033C02 YN - 000023C02 YN1 - 000010C02 YT - 000000C02
Z - 000002C03

START OF CONSTANTS
000157

START OF TEMPORARIES
000173

START OF INDIRECTS
000177

UNUSED COMPILER SPACE
036500

```
                FUNCTION  QUAD(I,J,P)                                  KFNLF    1045
000005          DIMENSION  P(1)                                        KFNLF    1046
000005          QUAD=(I*P(1)+J*P(2))*I+(I*P(2)+J*P(3))*J                KFNLF    1047
000017          RETURN                                                  KFNLF    1048
000017          END                                                     KFNLF    1049


SUBPROGRAM LENGTH
000035

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

BLOCK NAMES AND LENGTHS

VARIABLE ASSIGNMENTS
QUAD    -  000034

START OF CONSTANTS
000022

START OF TEMPORARIES
000023

START OF INDIRECTS
000031

UNUSED COMPILER SPACE
037100
```

```
                SUBROUTINE  QWKPLT(X,Y,Z,NP,NL)                                  19MAR71   122
000007          DIMENSION  X(1), Y(1), Z(1), IA(36), IO(119), ZH(991)            19MAR71   123
000007          INTEGER  PR                                                      19MAR71   124
000007          DATA (PR=6),(I3=1H )                                             19MAR71   125
000007          DATA ((IA(I),I=1,36)=1HC,1H1,1H2,1H3,1H4,1H5,1H6,1H7,1H8,1H9,1HA,19MAR71   126
                1   1HB,1HC,1HD,1HE,1HF,1HG,1HH,1HI,1HJ,1HK,1HL,1HM,1HN,1HO,1HP,1HQ19MAR71   127
                2   ,1HR,1HS,1HT,1HU,1HV,1HW,1HX,1HY,1HZ)                        19MAR71   128
000007          IF (NL.GT.36) NL=36                                              19MAR71   129
000013          ZMIN=Z(1)                                                        19MAR71   130
000013          ZMAX=Z(1)                                                        19MAR71   131
000014          K=1                                                              19MAR71   132
000016          DO 10 J=1,NP                                                     19MAR71   133
000017          DO 10 I=1,NP                                                     19MAR71   134
000024          IF (Z(K).GT.ZMAX) ZMAX=Z(K)                                      19MAR71   135
000030          IF (Z(K).LT.ZMIN) ZMIN=Z(K)                                      19MAR71   136
000034       10 K=K+1                                                            19MAR71   137
000044          WRITE (PR,60000) NL, X(1), X(NP), Y(1), Y(NP), ZMIN, ZMAX        19MAR71   138
000101   60000  FORMAT(1H0,21HQWIKPLOT OUTPUT WITH ,I5,7H LEVELS/1H ,            19MAR71   139
                1   20HX LIMITS-HORIZONTAL-, 2E15.7/1H ,18HY LIMITS-VERTICAL-,2X,19MAR71   140
                2   2E15.7/1H ,9HZ LIMITS-,11X,2E15.7/1H1)                       19MAR71   141
000104          IF (ZMAX.LT.1.E-100) RETURN                                      19MAR71   142
000107          XNL=NL+1                                                         19MAR71   143
000111          XNL=ZMAX*EXP(-XNL*XNL/2.)                                        19MAR71   144
000116          MM=NP*NP                                                         19MAR71   145
000124          DO 15 I=1,MM                                                     19MAR71   146
000125          ZH(I)=0.                                                         19MAR71   147
000126       15 IF (Z(I).GT.XNL) ZH(I)=SQRT(ABS(2.*ALOG(Z(I)/ZMAX)))-NL-1.       19MAR71   148
000156          DO 30 I=1,60                                                     19MAR71   149
000157          Y(I)=Y(NP)-(I-1.)*(Y(NP)-Y(1))/59.                               19MAR71   150
000165          DO 20 J=1,119                                                    19MAR71   151
000167          XI=(J-1.)*(X(NP)-X(1))/118.+X(1)                                 19MAR71   152
000175          IL=3.*(ALAG30(XI,YI,X,Y,ZH,NP,NP)+NL+2.)+1.                      19MAR71   153
000214          IO(J)=IB                                                         19MAR71   154
000217          IF (MOD(IL,3).NE.0) GO TO 20                                     19MAR71   155
000223          IL=IL/3                                                          19MAR71   156
000225          IO(J)=IA(IL)                                                     19MAR71   157
000230       20 CONTINUE                                                         19MAR71   158
000232       30 WRITE (PR,60100) IO                                              19MAR71   159
000245   60100  FORMAT(1H ,119A1)                                                19MAR71   160
000245          WRITE (PR,60200)                                                 19MAR71   161
000251   60200  FORMAT(1H1)                                                      19MAR71   162
000254          RETURN                                                           19MAR71   163
000255          END                                                              19MAR71   164


SUBPROGRAM LENGTH
002550

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
10      -  000035    20      -  000231    60000  -  000261    60100  -  000313
60200   -  000316

BLOCK NAMES AND LENGTHS

VARIABLE ASSIGNMENTS
```

```
I       -  002542    IA      -  000342    IB      -  002535    IL      -  002547
IO      -  000405    J       -  002541    K       -  002540    MM      -  002544
PR      -  002534    XI      -  002546    XNI     -  012543    YI      -  002545
ZH      -  000575    ZMAX    -  002537    ZMIN    -  002536

START OF CONSTANTS
000260

START OF TEMPORARIES
000320

START OF INDIRECTS
000334

UNUSED COMPILER SPACE
035700
```

```
                SUBROUTINE  ROTATE(P,PR,U,RL)
000006          DIMENSION  P(1), PR(1), U(1), RL(1)
000015          PR(1)=(U(1)*P(1)+U(2)*P(2))*U(1)+(U(1)*P(2)+U(2)*P(3))*U(2)
000027          PR(2)=(U(1)*P(1)+U(2)*P(2))*U(3)+(U(1)*P(2)+U(2)*P(3))*U(4)
000041          PR(3)=(U(3)*P(1)+U(4)*P(2))*U(3)+(U(3)*P(2)+U(4)*P(3))*U(4)
000053          PR(1)=RL(1)*PR(1)
000054          PR(2)=SQRT(RL(1)*RL(2))*PR(2)
000064          PR(3)=RL(2)*PR(3)
000067          RETURN
000067          END

SUBPROGRAM LENGTH
000134

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

BLOCK NAMES AND LENGTHS

VARIABLE ASSIGNMENTS

START OF CONSTANTS
000072

START OF TEMPORARIES
000073

START OF INDIRECTS
000114

UNUSED COMPILER SPACE
037000
```

```
KFNLF   1050
KFNLF   1051
KFNLF   1052
KFNLF   1053
KFNLF   1054
KFNLF   1055
KFNLF   1056
KFNLF   1057
KFNLF   1058
KFNLF   1059
```

```
                SUBROUTINE SCALE(W,EJ)                                  KFNLF   1060
000004          DIMENSION  W(1), EJ(1)                                  KFNLF   1061
      C            SCALES THE NEW DENSITY W TO INTEGRATE TO ONE         KFNLF   1062
000004          COMMON  / CM1/ M, NSD, MMAX                             KFNLF   1063
000004          COMMON  / CM2/ AT(10), RLAMN1(2), AT1(7), YN(2), PN(3)  KFNLF   1064
000004          COMMON  / CM5/ C, Y1, Y2, S11, S12, S22                 KFNLF   1065
000006          YN(1)=Y1/C                                              KFNLF   1066
000007          YN(2)=Y2/C                                              KFNLF   1067
000011          PN(1)=S11/C-YN(1)*YN(1)                                 KFNLF   1068
000013          PN(2)=S12/C-YN(1)*YN(2)                                 KFNLF   1069
000015          PN(3)=S22/C-YN(2)*YN(2)                                 KFNLF   1070
000020          MM=MMAX*MMAX                                            KFNLF   1071
000022          XS=(C**NSD*NSD*RLAMN1(1)*RLAMN1(2))/(M*M)               KFNLF   1072
000032          DO 10 I=1,MM                                            KFNLF   1073
000040       10 EJ(I)=W(I)/XS                                           KFNLF   1074
000042          RETURN                                                  KFNLF   1075
000042          END                                                     KFNLF   1076


SUBPROGRAM LENGTH
000064

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

BLOCK NAMES AND LENGTHS
CM1    -  000003    CM2    -  000030    CM5    -  000006

VARIABLE ASSIGNMENTS
AT     -  000010C02 AT1    -  000014C02 C      -  000000C03 I      -  000063
M      -  000001C01 MM     -  000061    MMAX   -  000002C01 NSD    -  000001C01
PN     -  000025C02 RLAMN1 -  000012C02 S11    -  000003C03 S12    -  000004C03
S22    -  000005C03 XS     -  000062    YN     -  000023C02 Y1     -  000001C03
Y2     -  000002C03

START OF CONSTANTS
000045

START OF TEMPORARIES
000047

START OF INDIRECTS
000057

UNUSED COMPILER SPACE
037000
```

```
                SUBROUTINE  SHIFT(N,EJ,JC)                                      KFNLF   1077
000005           EAL  JC(1), EJ(1)                                              KFNLF   1078
          C        SHIFTS AND UPDATES GRID PARAMETERS                           KFNLF   1079
***TT ************************ SHIFT
000005           DIMENSION  X(2), F(2), A(3), XP(2), S(4)                       KFNLF   1080
000005           COMMON  / CM1/ M, NSD, MMAX                                    KFNLF   1081
000005           COMMON  / CM2/ YN2(2), RLAMN2(2 , RUN2(4), YN1(2), RLAMN1(2),  KFNLF   1082
                1   RUN1(4), PN1(3)                                             KFNLF   1083
000005           DO 10 I=1,2                                                    KFNLF   1084
000013           YN2(I)=YN1(I)                                                  KFNLF   1085
000014           RLAMN2(I)=RLAMN1(I)                                            KFNLF   1086
000015        10 RUN2(I)=RUN1(I)                                                KFNLF   1087
000017           RUN2(3)=RUN1(3)                                                KFNLF   1088
000020           RUN2(4)=RUN1(4)                                                KFNLF   1089
000022           DO 20 I=1,2                                                    KFNLF   1090
000030           YN1(I)=0.                                                      KFNLF   1091
000031        20 PN1(I)=0.                                                      KFNLF   1092
000032           PN1(3)=0.                                                      KFNLF   1093
000032           K=1                                                            KFNLF   1094
000034           DO 40 J=1,MMAX                                                 KFNLF   1095
000037           XP(2)=JC(J)                                                    KFNLF   1096
000041           DO 40 I=1,MMAX                                                 KFNLF   1097
000042           XP(1)=JC(I)                                                    KFNLF   1098
000044           CALL  XUNPRM(XP,RUN2,RLAMN2,YN2,X)                             KFNLF   1099
000047           CALL  FX(X-F,N)                                                KFNLF   1100
000054           CALL  SX(X,S,N)                                                KFNLF   1101
000061           CALL  AX(S,A,N)                                                KFNLF   1102
000066           DO 30 L=1,2                                                    KFNLF   1103
000100           YN1(L)=F(L)*EJ(K)+YN1(L)                                       KFNLF   1104
000102        30 PN1(L)=(F(1)*F(L)+A(L))*EJ(K)+PN1(L)                           KFNLF   1105
000106           PN1(3)=(F(2)*F(2)+A(3))*EJ(K)+PN1(3)                           KFNLF   1106
000114        40 K=K+1                                                          KFNLF   1107
000123           D=NSD*NSD*RLAMN2(1)*RLAMN2(2)/FLOAT(M*M)                       KFNLF   1108
000130           YN1(1)=YN1(1)*D                                                KFNLF   1109
000132           YN1(2)=YN1(2)*D                                                KFNLF   1110
000133           PN1(1)=PN1(1)*D-YN1(1)*YN1(1)                                  KFNLF   1111
000135           PN1(2)=PN1(2)*D-YN1(1)*YN1(2)                                  KFNLF   1112
000137           PN1(3)=PN1(3)*D-YN1(2)*YN1(2)                                  KFNLF   1113
000141           CALL  XFM(PN1,RUN1,RLAMN1)                                     KFNLF   1114
000144           RETURN                                                         KFNLF   1115
000145           END                                                            KFNLF   1116


SUBPROGRAM LENGTH
000214

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

BLOCK NAMES AND LENGTHS
CM1     -  000003    CM2     -  000023

VARIABLE ASSIGNMENTS
A       -  000176    D       -  000213    F       -  000174    I       -  000207
J       -  000211    K       -  000210    L       -  000212    M       -  000000C01
MMAX    -  000002C01 NSD     -  000001C01 PN1     -  000020C02 RLAMN1  -  000012C02
```

```
RLAMN2 -  000302C02 RUN1   -  000314C02 RUN2   -  000004C02 S      -  000203
X      -  000172    XP     -  000201    YN1    -  000010C02 YN2    -  000000C02

START OF CONSTANTS
000150

START OF TEMPORARIES
000151

START OF INDIRECTS
000162

UNUSED COMPILER SPACE
036400

TT********VARIABLE GIVEN CONFLICTING TYPES
    000006
```

```
                SUBROUTINE  STATE(N)                                          KFNLF    1117
          C        GENERATES THE STATE VECTOR UPDATE                          KFNLF    1118
000002          DIMENSION  SIG(4), SIGU(2), AI(3)                             KFNLF    1119
000002          COMMON  / CM1/ IT(3), A11, A12, A22, IT1(2), NP               KFNLF    1120
000002          COMMON  / CM3/ RT(3), A(3), TT(3), SQTQ(3)                    KFNLF    1121
000002          COMMON  / CM4/ X(2), Z(2), U(2)                               KFNLF    1122
000002          DO 10 I=1,NP                                                  KFNLF    1123
000004       10 U(I)=GRN01(0)                                                 KFNLF    1124
000012          UT=U(1)                                                       KFNLF    1125
000013          U(1)=SQTQ(1)*U(1)                                             KFNLF    1126
000015          IF (NP.GT.1) U(2)=SQTQ(2)*UT+SQTQ(3)*U(2)                     KFNLF    1127
000023          CALL  SX(X,SIG,N)                                             KFNLF    1128
000026          CALL  AX(SIG,A,N)                                             KFNLF    1129
000032          CALL  MINV(A,AI,2)                                            KFNLF    1130
000035          A11=-.5*AI(1)                                                 KFNLF    1131
000037          A12=-AI(2)                                                    KFNLF    1132
000040          A22=-.5*AI(3)                                                 KFNLF    1133
000042          SIGU(1)=SIG(1)*U(1)                                           KFNLF    1134
000044          IF (NP.GT.1) SIGU(1)=SIGU(1)+SIG(3)*U(2)                      KFNLF    1135
000052          SIGU(2)=SIG(2)*U(1)                                           KFNLF    1136
000054          IF (NP.GT.1) SIGU(2)=SIGU(2)+SIG(4)*U(2)                      KFNLF    1137
000060          CALL   X(X,X,N)                                               KFNLF    1138
000063          DO 20 I=1,2                                                   KFNLF    1139
000071       20 X(I)=X(I)+SIGU(I)                                             KFNLF    1140
000073          RETURN                                                        KFNLF    1141
000074          END                                                           KFNLF    1142


SUBPROGRAM LENGTH
000123

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

BLOCK NAMES AND LENGTHS
CM1     -  000011     CM3     -  000014     CM4     -  000006

VARIABLE ASSIGNMENTS
A       -  000003C02 AI      -  000116     A11     -  000003C01 A12     -  000004C01
A22     -  000005C01 I       -  000121     IT      -  000000C01 IT1     -  000006C01
NP      -  000010C01 RT      -  000000C02 SIG     -  000110     SIGU    -  000114
SQTQ    -  000011C02 TT      -  000006C02 U       -  000004C03 UT      -  000122
X       -  000000C03 Z       -  000002C03

START OF CONSTANTS
000077

START OF TEMPORARIES
000103

START OF INDIRECTS
000107

UNUSED COMPILER SPACE
036600
```

```
                SUBROUTINE  SX(X,S,N)                                   KFNLF   1143
000005          DIMENSION  X(1), S(1), T(4)                             KFNLF   1144
000005          COMMON  / CM1/ NT(8), NP                                KFNLF   1145
000005          COMMON  / CM8/ FL, SL, HL, XLF(4), XLS(4)               KFNLF   1146
000005          LOGICAL  FL, SL, HL                                     KFNLF   1147
000005          IF (SL) GO TO 20                                        KFNLF   1148
          C        NONLINEAR SIGMA COMPUTED HERE                        KFNLF   1149
          C        LINEAR IS SPECIAL CASE                               KFNLF   1150
000006          IF (NP.GT.1) GO TO 10                                   KFNLF   1151
000012          S(1)=1.                                                 KFNLF   1152
000013          S(2)=1.                                                 KFNLF   1153
000014          RETURN                                                  KFNLF   1154
000014       10 S(1)=1.                                                 KFNLF   1155
000016          S(2)=0.                                                 KFNLF   1156
000017          S(3)=0.                                                 KFNLF   1157
000017          S(4)=1.                                                 KFNLF   1158
000020          RETURN                                                  KFNLF   1159
000021       20 S(1)=XLS(1)                                             KFNLF   1160
000023          S(2)=XLS(2)                                             KFNLF   1161
000024          S(3)=XLS(3)                                             KFNLF   1162
000026          S(4)=XLS(4)                                             KFNLF   1163
000027          RETURN                                                  KFNLF   1164
000030          END                                                     KFNLF   1165


SUBPROGRAM LENGTH
000047

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
10      -  000015     20      -  000022

BLOCK NAMES AND LENGTHS
CM1     -  000011     CM8     -  000013

VARIABLE ASSIGNMENTS
FL      -  000000C02 HL      -  000002C02 NP      -  000010C01 NT      -  000000C01
SL      -  000001C02 T       -  000043    XLF     -  000003C02 XLS     -  000007C02

START OF CONSTANTS
000033

START OF TEMPORARIES
000035

START OF INDIRECTS
000037

UNUSED COMPILER SPACE
037000
```

```
                SUBROUTINE  TMULT(A,B,C,N)
000006          DIMENSION  A(1), B(1), C(1)
000010          C(1)=A(1)*B(1)+A(2)*B(2)
000013          C(2)=A(2)*B(1)+A(3)*B(2)
000016          IF (N.EQ.1) RETURN
000022          C(3)=A(1)*B(3)+A(2)*B(4)
000026          C(4)=A(2)*B(3)+A(3)*B(4)
000030          RETURN
000031          END
```

```
KFNLF   1166
KFNLF   1167
KFNLF   1168
KFNLF   1169
KFNLF   1170
KFNLF   1171
KFNLF   1172
KFNLF   1173
KFNLF   1174
```

SUBPROGRAM LENGTH
000046

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

BLOCK NAMES AND LENGTHS

VARIABLE ASSIGNMENTS

START OF CONSTANTS
000034

START OF TEMPORARIES
000035

START OF INDIRECTS
000037

UNUSED COMPILER SPACE
037100

```
                SUBROUTINE  VMULT(K,L,M,N,A,B,C)                        KFNLF    1175
000011          DIMENSION  A(1), B(1), C(1)                             KFNLF    1176
000011          KL=K                                                    KFNLF    1177
000011          IF (L.EQ.1) KL=1                                        KFNLF    1178
000014          KM=K                                                    KFNLF    1179
000015          IF (M.EQ.1) KM=1                                        KFNLF    1180
000017       60 DO 200 II=1,L                                           KFNLF    1181
000021          JK=0                                                    KFNLF    1182
000021          LL=II                                                   KFNLF    1183
000023          DO 200 JJ=1,N                                           KFNLF    1184
000024          NN=JK                                                   KFNLF    1185
000025          MM=II                                                   KFNLF    1186
000026          C(LL)=0.                                                KFNLF    1187
000031          DO 100 KK=1,M                                           KFNLF    1188
000041          NN=NN+1                                                 KFNLF    1189
000042          C(LL)=C(LL)+A(MM)*B(NN)                                 KFNLF    1190
000044       10 MM=MM+KL                                                KFNLF    1191
000052          LL=LL+KL                                                KFNLF    1192
000057       20 JK=JK+KM                                                KFNLF    1193
000065          RETURN                                                  KFNLF    1194
000065          END                                                     KFNLF    1195
```

SUBPROGRAM LENGTH
000113

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
60 - 000120

BLOCK NAMES AND LENGTHS

VARIABLE ASSIGNMENTS

C	- 000000	II	- 000104	JJ	- 000107	JK	- 000105	
KK	- 000112	KL	- 000102	KM	- 000103	LL	- 000106	
MM	- 000111	NN	- 000110					

START OF CONSTANTS
000070

START OF TEMPORARIES
000071

START OF INDIRECTS
000073

UNUSED COMPILER SPACE
036700

```
            SUBROUTINE  XCNV(MC,N,H,EJ,JC,H1,H2,H3,IX1,IX2)                    KFNLF    1196
000014      INTEGER  IX1(1), IX2(1)                                            KFNLF    1197
***TT ********************** XCNV
000014      REAL  H(1), EJ(1), JC(1), H1(1), H2(1), H3(1)                      KFNLF    1198
      C        COMPUTES CONVOLUTION INTEGRAL WITH X-SCANNED ELLIPSES           KFNLF    1199
***TT ********************** XCNV
000014      DIMENSION  XC(2), XP(2), XT(2), Y(2)                               KFNLF    1200
000014      COMMON  / CM1/ M, NSD, MMAX, A11, A12, A22, NPTS, NPQ, NP          KFNLF    1201
000014      COMMON  / CM2/ YN2(2), RLAMN2(2), RUN2(4), YN1(2), RLAMN1(2),      KFNLF    1202
     1   RUN1(4)                                                               KFNLF    1203
000014      COMMON  / CM4/ XZ(8), XCN(2), XK(4)                                KFNLF    1204
000014      COMMON  / CM5/ C, Y1, Y2, S11, S12, S22                            KFNLF    1205
000014      COMMON  / CM6/ TT(6), PS                                           74AR71     50
000014      LOGICAL  PS                                                        74AR71     51
000014      C=0.                                                               KFNLF    1206
000014      Y1=0.                                                              KFNLF    1207
000015      Y2=0.                                                              KFNLF    1208
000016      S11=0.                                                             KFNLF    1209
000016      S12=0.                                                             KFNLF    1210
000017      S22=0.                                                             KFNLF    1211
000017      XMN=FLOAT(M)/FLOAT(NSD)                                            KFNLF    1212
000022      XM1=M+1.                                                           KFNLF    1213
000024      IJ=1                                                               KFNLF    1214
000026      DO 50 J=1,MMAX                                                     KFNLF    1215
000027      XC(2)=JC(J)                                                        KFNLF    1216
000031      DO 50 I=1,MMAX                                                     KFNLF    1217
000032      XC(1)=JC(I)                                                        KFNLF    1218
000034      CALL  XUMP2M(XC,RUN1,RLAMN1,YN1,Y)                                 KFNLF    1219
000037      XT(1)=XCN(1)+XK(1)*Y(1)+XK(3)*Y(2)                                 KFNLF    1220
000043      XT(2)=XCN(2)+XK(2)*Y(1)+XK(4)*Y(2)                                 KFNLF    1221
000050      CALL  XPRM(XT,RUN2,RLAMN2,YN2,XP)                                  KFNLF    1222
000054      IXC1=XMN*XP(1)+XM1                                                 KFNLF    1223
000057      IXC2=XMN*XP(2)+XM1                                                 KFNLF    1224
000062      IF (.NOT.PS.AND.M.LE.1.AND.I.EQ.M+1.AND.J.EQ.M+1)                  74AR71     52
     1   CALL  ECHECK(N,MC,I,J,EJ,JC,H1,H2,H3,IX1,IX2,XT,Y,IXC1,IXC2,C)      KFNLF    1226
000127      H1=0.                                                              KFNLF    1227
000130      DO 40 K=1,NPQ                                                      KFNLF    1228
000135      NM=1                                                               KFNLF    1229
000135      IC=K+IXC2-1                                                        KFNLF    1230
000140      IF (IC.LT.1) GO TO 40                                              KFNLF    1231
000142      IF (IC.GT.MMAX) GO TO 30                                           KFNLF    1232
000145      IC=(IC-1)*MMAX                                                     KFNLF    1233
000146      IS=IXC1+IX1(K)                                                     KFNLF    1234
000151      IF (IS.GT.MMAX) IS=MMAX                                            KFNLF    1235
000154      IF (IS.LT.1) IS=1                                                  KFNLF    1236
000160      IS=IS+IC                                                           KFNLF    1237
000161      IL=IS+IX2(K)                                                       KFNLF    1238
000164      IF (IL.GT.IC+MMAX) IL=IC+MMAX                                      KFNLF    1239
000170   10 DO 20 IS=IS,IL                                                     KFNLF    1240
000172      Z1=Y(1)+H1(IS)                                                     KFNLF    1241
000175      Z2=Y(2)+H2(IS)                                                     KFNLF    1242
000200      Q=H3(IS)+(A11*Z1+A12*Z2)*Z1+A22*Z2*Z2                              KFNLF    1243
000206      IF (Q.LT.-230) GO TO 20                                            KFNLF    1244
000211      W1=W1+EXP(Q)*EJ(IS)                                                KFNLF    1245
000222   20 CONTINUE                                                           KFNLF    1246
000225      GO TO (30,40), NM                                                  KFNLF    1247
000233   30 IF (K.LE.1) GO TO 40                                               KFNLF    1248
000236      NM=2                                                               KFNLF    1249
```

```
000236          IC=-K+IXC2+1                                                    KFNLF   1250
000240          IF (IC.LT.1.OR.IC.GT.MMAX) GO TO 40                             KFNLF   1251
000247          IC=(IC-1)*MMAX                                                  KFNLF   1252
000251          IS=-IX1(K)+IX2(K)+IXC1                                          KFNLF   1253
000255          IF (IS.LT.1) IS=1                                               KFNLF   1254
000260          IF (IS.GT.MMAX) IS=MMAX                                         KFNLF   1255
000263          IS=IS+IC                                                        KFNLF   1256
000264          IL=IS+IX2(K)                                                    KFNLF   1257
000267          IF (IL.GT.IC+MMAX) IL=IC+MMAX                                   KFNLF   1258
000272          GO TO 10                                                        KFNLF   1259
000273       40 CONTINUE                                                        KFNLF   1260
000301          C=C+W1                                                          KFNLF   1261
000302          S11=S11+W1*Y(1)*Y(1)                                            KFNLF   1262
000304          S12=S12+W1*Y(1)*Y(2)                                            KFNLF   1263
000306          S22=S22+W1*Y(2)*Y(2)                                            KFNLF   1264
000311          Y1=Y1+W1*Y(1)                                                   KFNLF   1265
000313          Y2=Y2+W1*Y(2)                                                   KFNLF   1266
000314          W(IJ)=W1                                                        KFNLF   1267
000316       50 IJ=IJ+1                                                         KFNLF   1268
000324          RETURN                                                          KFNLF   1269
000324          END                                                             KFNLF   1270

SUBPROGRAM LENGTH
000370

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
10      -  000171     20      -  000223     30      -  000234     40      -  000274

BLOCK NAMES AND LENGTHS
CM1     -  000011     CM2     -  000020     CM4     -  000315     CM5     -  000006
CM6     -  000007

VARIABLE ASSIGNMENTS
A11     -  000363C01  A12     -  000004C01  A22     -  000005C01  C       -  000000C04
H2      -  000305     H3      -  000001     I       -  000354     IC      -  000362
IJ      -  000352     IL      -  000364     IS      -  000363     IXC1    -  000355
IXC2    -  000356     IX1     -  000002     IX2     -  000003     J       -  000353
K       -  000360     M       -  000000C01  MMAX    -  000002C01  NM      -  000361
NPQ     -  000007C01  NSD     -  000001C01  PS      -  000006C05  Q       -  000367
RLAMN1  -  000012C02  RLAMN2  -  000002C02  RUN1    -  000014C02  RUN2    -  000004C02
S11     -  000003C04  S12     -  000004C04  S22     -  000005C04  TT      -  000000C05
W1      -  000357     XC      -  000340     XCN     -  000013C03  XK      -  000012C03
XMN     -  000350     XM1     -  000351     XP      -  000342     XT      -  000344
XZ      -  000010C03  Y       -  000346     YN1     -  000010C02  YN2     -  000000C02
Y1      -  000001C04  Y2      -  000002C04  Z1      -  000365     Z2      -  000366

START OF CONSTANTS
000327

START OF TEMPORARIES
000332

START OF INDIRECTS
000340
```

UNUSED COMPILER SPACE
035500

TT******VARIABLE GIVEN CONFLICTING TYPES
000015,000015

```
                SUBROUTINE  XFM(P,RU,RLAM)                                      KFNLF   1271
000005          DIMENSION   P(1), RU(1), RLAM(1)                                KFNLF   1272
          C        DETERMINES ROTATION MATRIX RU CORRESPONDING TO THE CORRELATION KFNLF   1273
          C        MATRIX P, STORED IN LOWER TRIANGULAR FORM. RLAM IS THE VECTOR  KFNLF   1274
          C        OF SQUARE ROOTS OF EIGENVALUES.                                KFNLF   1275
000005          CALL  EIGEN(P,RU,RLAM)                                          KFNLF   1276
000006          RLAM(1)=SQRT(RLAM(1))                                           KFNLF   1277
000014          RLAM(2)=SQRT(RLAM(2))                                           KFNLF   1278
000023          RETURN                                                          KFNLF   1279
000023          END                                                             KFNLF   1280


SUBPROGRAM LENGTH
000037

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

BLOCK NAMES AND LENGTHS

VARIABLE ASSIGNMENTS

START OF CONSTANTS
000026

START OF TEMPORARIES
000027

START OF INDIRECTS
000035

UNUSED COMPILER SPACE
037100
```

```
                SUBROUTINE  XPR1(X,XU,XRL,XM,XP)                                KFNLF   1281
000007          DIMENSION  X(1), XU(1), XRL(1), XM(1), XP(1)                    KFNLF   1282
          C       ROTATES THE VECTOR X INTO THE PRIMED VECTOR XP ACCORDING TO THE KFNLF   1283
          C       ROTATION MATRIX XU; THE SQUARE-ROOT EIGENVALUE VECTOR XRL, AND KFNLF   1284
          C       THE TRANSLATION VECTOR XM.                                     KFNLF   1285
000007          K=1                                                            KFNLF   1286
000010          DO 10 I=1,2                                                    KFNLF   1287
000011          XP(I)=0.                                                       KFNLF   1288
000012          DO 10 J=1,2                                                    KFNLF   1289
000022          XP(I)=XU(K)*(X(J)-XM(J))/XRL(I)+XP(I)                          KFNLF   1290
000027       10 K=K+1                                                          KFNLF   1291
000034          RETURN                                                         KFNLF   1292
000034          END                                                            KFNLF   1293

SUBPROGRAM LENGTH
000055

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

BLOCK NAMES AND LENGTHS

VARIABLE ASSIGNMENTS
I       -  000053    J       -  000054    K       -  000052

START OF CONSTANTS
000037

START OF TEMPORARIES
000040

START OF INDIRECTS
000045

UNUSED COMPILER SPACE
037100
```

```
                SUBROUTINE  XUNPRM(XP,XU,XRL,XM,X)                                  KFNLF   1294
000007          DIMENSION  XP(1), XU(1), XRL(1), XM(1), X(1)                        KFNLF   1295
C                  ROTATES THE PRIMED VECTOR XP BACK TO THE UNPRIMED VERSION X,     KFNLF   1296
C                  ACCORDING TO THE ROTATION MATRIX XU, THE SQUARE-ROOT EIGENVALUE  KFNLF   1297
C                  VECTOR XRL, AND THE TRANSLATION VECTOR XM.                       KFNLF   1298
000007          DO 10 I=1,2                                                         KFNLF   1299
000011          K=I                                                                 KFNLF   1300
000011          X(I)=XM(I)                                                          KFNLF   1301
000013          DO 10 J=1,2                                                         KFNLF   1302
000021          X(I)=X(I)+XU(K)*XP(J)*XRL(J)                                        KFNLF   1303
000025       10 K=K+2                                                               KFNLF   1304
000034          RETURN                                                              KFNLF   1305
000035          END                                                                 KFNLF   1306


SUBPROGRAM LENGTH
000051

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

BLOCK NAMES AND LENGTHS

VARIABLE ASSIGNMENTS
I       -  000046    J       -  000050    K       -  000047

START OF CONSTANTS
000040

START OF TEMPORARIES
000041

START OF INDIRECTS
000043

UNUSED COMPILER SPACE
037000
```

```
                SUBROUTINE  YCNV(MC,N,M,EJ,JC,H1,H2,H3,IX1,IX2)                  KFNLF    1307
000014          INTEGER  IX1(1),IX2(1)                                           KFNLF    1308
***TT ********** ************ YCNV
000014          REAL  H1(1), EJ(1), JC(1), H1(1), H2(1), H3(1)                   KFNLF    1309
      C         COMPUTES CONVOLUTION INTEGRAL WITH Y-SCANNED ELLIPSES            KFNLF    1310
***TT ********************** /CNV
000014          DIMENSION  XC(2), XP(2), XT(2), Y(2)                             KFNLF    1311
000014          COMMON  / CM1/ M, NSO, MMAX, A11, A12, A22, NPTS, NPQ, NP        KFNLF    1312
000014          COMMON  / CM2/ YN2(2), RLAMN2(2), RUN2(4), YN1(2), RLAMN1(2),    KFNLF    1313
               1  RUN1(4)                                                       KFNLF    1314
000014          COMMON  / CM4/ XZ(8), XCN(2), XK(4)                              KFNLF    1315
000014          COMMON  / CM5/ C, Y1, Y2, S11, S12, S22                          KFNLF    1316
000014          COMMON  / CM6/ TT(6), PS                                         7MAR71     53
000014          LOGICAL  PS                                                      7MAR71     54
000014          C=0.                                                             KFNLF    1317
000014          Y1=0.                                                            KFNLF    1318
000015          Y2=0.                                                            KFNLF    1319
000016          S11=0.                                                           KFNLF    1320
000016          S12=0.                                                           KFNLF    1321
000017          S22=0.                                                           KFNLF    1322
000017          XMN=FLOAT(M)/FLOAT(NSO)                                          KFNLF    1323
000022          XM1=M+1.                                                         KFNLF    1324
000024          MM1=(MMAX-1)*MMAX                                                KFNLF    1325
000027          IJ=1                                                             KFNLF    1326
000031          DO 50 J=1,MMAX                                                   KFNLF    1327
000032          XC(2)=JC(J)                                                      KFNLF    1328
000034          DO 50 I=1,MMAX                                                   KFNLF    1329
000035          XC(1)=JC(I)                                                      KFNLF    1330
000037          CALL XUNPRM(XC,RUN1,RLAMN1,YN1,Y)                                KFNLF    1331
000042          XT(1)=XCN(1)+XK(1)*Y(1)+XK(3)*Y(2)                               KFNLF    1332
000046          XT(2)=XCN(2)+XK(2)*Y(1)+XK(4)*Y(2)                               KFNLF    1333
000053          CALL XPRM(XT,RUN2,RLAMN2,YN2,XP)                                 KFNLF    1334
000057          IXC1=XMN*XP(1)+XM1                                               KFNLF    1335
000062          IXC2=XMN*XP(2)+XM1                                               KFNLF    1336
000065          IF (.NOT.PS.AND.M.LE.1.AND.I.EQ.M+1.AND.J.EQ.M+1)                7MAR71     55
               1   CALL  ECHECK(N,MC,I,J,EJ,JC,H1,H2,H3,IX1,IX2,XT,Y,IXC1,IXC2,1)  KFNLF    1338
000132          A1=0.                                                            KFNLF    1339
000133          DO 40 K=1,NPQ                                                    KFNLF    1340
000140          NM=1                                                             KFNLF    1341
000140          IR=K+IXC1-1                                                      KFNLF    1342
000143          IF (IR.LT.1) GO TO 40                                            KFNLF    1343
000145          IF (IR.GT.MMAX) GO TO 40                                         KFNLF    1344
000150          IS=IXC2+IX1(K)                                                   KFNLF    1345
000152          IF (IS.GT.MMAX) IS=MMAX                                          KFNLF    1346
000155          IF (IS.LT.1) IS=1                                                KFNLF    1347
000161          IS=(IS-1)*MMAX+IR                                                KFNLF    1348
000164          IL=IS+IX2(K)*MMAX                                                KFNLF    1349
000170          IF (IL.GT.IR+MM1) IL=IR+MM1                                      KFNLF    1350
000174       10 DO 20 IS=IS,IL,MMAX                                              KFNLF    1351
000176          Z1=Y(1)+H1(IS)                                                   KFNLF    1352
000201          Z2=Y(2)+H2(IS)                                                   KFNLF    1353
000204          Q=H3(IS)+(A11*Z1+A12*Z2)*Z1+A22*Z2*Z2                            KFNLF    1354
000212          IF (Q.LT.-230) GO TO 20                                          KFNLF    1355
000215          A1=A1+EXP(Q)*EJ(IS)                                              KFNLF    1356
000226       20 CONTINUE                                                         KFNLF    1357
000231          GO TO (30,40), NM                                                KFNLF    1358
000237       30 IF (K.LE.1) GO TO 40                                             KFNLF    1359
000242          NM=2                                                             KFNLF    1360
```

```
000242          IR=-K+IXC1+1                                                    KFNLF   1361
000244          IF (IR.LT.1.OR.IR.GT.MMAX) GO TO 40                             KFNLF   1362
000253          IS=IXC2-IX1(K)-IX2(K)                                           KFNLF   1363
000256          IF (IS.GT.MMAX) IS=MMAX                                         KFNLF   1364
000262          IF (IS.LT.1) IS=1                                               KFNLF   1365
000266          IS=(IS-1)*MMAX+IR                                               KFNLF   1366
000271          IL=IS+IX2(K)*MMAX                                               KFNLF   1367
000275          IF (IL.GT.IR+MM1) IL=IR+MM1                                     KFNLF   1368
000301          GO TO 10                                                        KFNLF   1369
000302       40 CONTINUE                                                        KFNLF   1370
000310          C=C+W1                                                          KFNLF   1371
000311          Y1=Y1+W1*Y(1)                                                   KFNLF   1372
000313          Y2=Y2+W1*Y(2)                                                   KFNLF   1373
000315          S11=S11+W1*Y(1)*Y(1)                                            KFNLF   1374
000317          S12=S12+W1*Y(1)*Y(2)                                            KFNLF   1375
000321          S22=S22+W1*Y(2)*Y(2)                                            KFNLF   1376
000323          W(IJ)=W1                                                        KFNLF   1377
000325       50 IJ=IJ+1                                                         KFNLF   1378
000333          RETURN                                                          KFNLF   1379
000333          END                                                             KFNLF   1380
```

Reproduced from best available copy.

```
SUBPROGRAM LENGTH
000402

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
10      -  000175     20      -  000227     30      -  000249     40      -  000303

BLOCK NAMES AND LENGTHS
CM1     -  000311     CM2     -  000320     CM4     -  000316     CM5     -  000006
CM6     -  000007

VARIABLE ASSIGNMENTS
A11     -  000 03C01  A12     -  000004C01  A22     -  000005C01  C       -  000C00C04
H2      -  000700     H3      -  000001     I       -  000366     IJ      -  000364
IL      -  000376     IR      -  000374     IS      -  000375     IXC1    -  000367
IXC2    -  000373     IX1     -  000002     IX2     -  000003     J       -  000365
K       -  000372     M       -  000001C01  MMAX    -  000002C01  MM1     -  000363
NM      -  000373     NPQ     -  000007C01  NSD     -  000001C01  PS      -  000006C05
Q       -  000401     RLAMN1  -  000012C02  RLAMN2  -  000002C02  RUN1    -  000014C02
RUN2    -  000 04C02  S11     -  000003C04  S12     -  000004C04  S22     -  000005C04
TT      -  000 01C05  W1      -  000371     XC      -  000351     XCN     -  000010C02
XK      -  000 12C03  XMN     -  000361     XM1     -  000362     XP      -  000353
XT      -  000355     XZ      -  000 03C03  Y       -  000357     YN1     -  000010C02
YN2     -  000 01C02  Y1      -  000010C 4  Y2      -  000002C04  Z1      -  000377
Z2      -  000400

START OF CONSTANTS
000336

START OF TEMPORARIES
000341

START OF INDIRECTS
000351
```

GENERALIZED TWO-DIMENSIONAL DISCRETE NONLINEAR PREDICTOR

CAPTAIN KENNETH D. SENNE
FRANK J. SEILER RESEARCH LABORATORY
AIR FORCE SYSTEMS COMMAND
U.S. AIR FORCE ACADEMY, CO 80840
FEBRUARY 1971

MULTIMODAL RANGING TEST 28 MARCH 1971

PROBLEM PARAMETERS

MAXIMUM MONTE CARLO	SAMPLES PER PATH
1	100

GRID SIZE	NO. GRID SIGMAS	NO. TRACKING ELLIPSE SIGMAS
M = 15	NSD = 6	NSDC = 16

NUMBER OF INPUTS	NUMBER OF OUTPUTS	INITIAL RANDOM NUMBER
NP = 2	NQ = 1	91526457325

PROCESS NOISE
COVARIANCE Q

.05J	.025
.025	.050

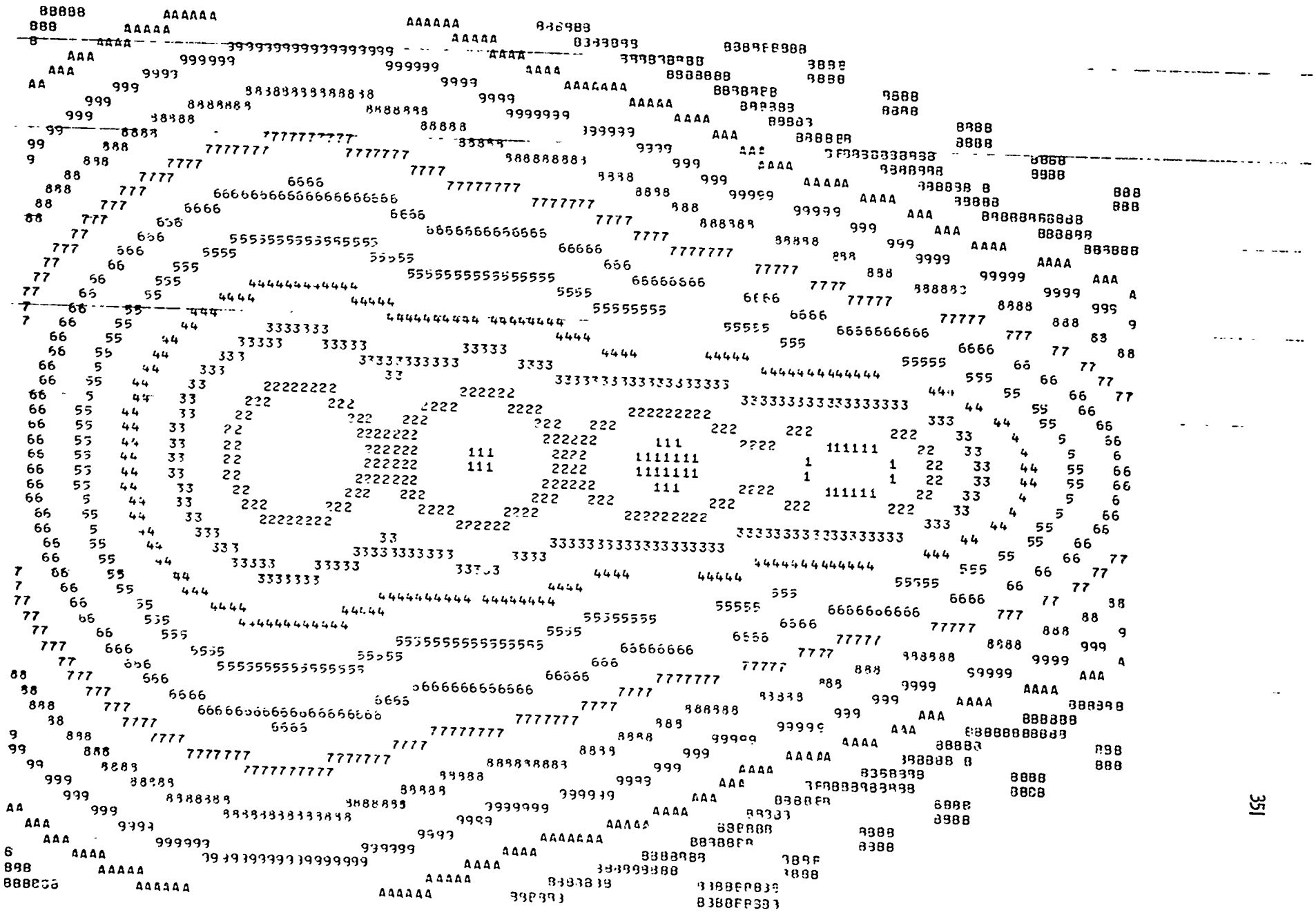
MEASUREMENT NOISE
COVARIANCE R

.01J

LINEAR PARAMETERS

F	1.001	2.000
	0.000	1.001
SIGMA	1.000	0.000
	0.000	1.000

```
QWIKPLOT OUTPUT WITH     10 LEVELS
X LIMITS-HORIZONTAL- -6.0000000E+00  6.0000000E+00
Y LIMITS-VERTICAL-   -6.0000000E+00  6.0000000E+00
Z LIMITS-             4.1991411E-65  4.2466288E-01
```



SAMPLE PATH INFORMATION FOR MONTE CARLO NUMBER 1

N	V(N-1)	Z(N-1)	U(N-1)	X(N)	XHAT(N/N-1)	SIGMA(N/N-1)	XHAT(N/N-2)	SIGMA(N/N-2)	SQTLAMU	NPTS PHI
0				7.[illegible]E+0[illegible]	1.000E+01	1.000E+00	0.	1.000E+01	1.000E+00	0.	1.000E+00	
				1.414E-01	0.	0.	2.700E-01	0.	0.	2.700E-01	5.196E-01	180
LINEARIZED CHECK- 0 ITERATION(S)...					1.000E+01	1.000E+00	0.	1.000E+01	1.000E+00	0.		
					0.	0.	2.700E-01	0.	0.	2.700E-01		

QWIKPLOT OUTPUT WITH 11 LEVELS

X LIMITS-HORIZONTAL-	-6.000000[illegible]E+00	6.[illegible]E+00
Y LIMITS-VERTICAL-	-6.000000E+00	6.000000E+00
Z LIMITS-	0.	2.4593272E-01

Additional Appendix B.

A Gaussian-Sum Program for the Passive Receiver Problem

This program was written for solving the "new problem" referred to in Appendix B of Chapter VI.

The program was compiled on one of the Kirtland AFB CDC 6600's using the RUN compiler. The random number generator is identical to the one used in the program of Additional Appendix A.

Programmer: D. L. Alspach

(Reproduced by permission)

Preceding page blank

```
                PROGRAM  KANLF(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)
      000002    DIMENSION X0(2)                                                  00003000
      000002    COMMON/GS0/ CPRI(2,150)),DPRI(2,2,1500),BETA(1500),NPRI          00004000
      000002    COMMON/GS1/ AM(1),2),A_PM(10),BM(10,2,2),NM                      00005000
      000002      COMMON/ONE/XSPRI(2,3)),VSPRI(2,2,30)                           00006000
      000002    COMMON/K2/EPK1(30),EPK2(30),EPS1(30),EPS2(30),AK(30),AS(30)      00007000
                1,EPK    ),EPK2P(30),EPS1P(30),EPS2P(30),AKP(30),ASP(30)       00008000
                1,NP     ,APR(30),XPNT(2,30),ZPNT(30)                          00009000
      000002    COMM     KPOST(2),VKPOST(2,2),XKPRI(2),VKPRI(2,2),ZK(1)        00010000
      000002    COMMO    PRI(2,2),H(1,2),PHIT(2,2),HT(2),Z(1),Q(2,2),R(1),NSTATE,00011000
                1 NMEAS,BE    EPS,EP                                            00012000
      000002    COMMON/K8/1ST MEAS,TOTIM,TGSPRI,TGSPST,TKAL,TDIF,T0              00013000
                1,KMAX,RMIN,R    BET,BETO,NRAND,KK                             00014000
      000002    COMMON/K9/ A(3    ),AP(3),10),B(30,6),BP(30,6)                  00015000
      000002    COMMON/S19/X(2),ZZZZ,K,XSPOST(2,30),VSPOST(2,2,30)             00016000
      000002    NAMELIST/ONE/KK,N,NMAX,NRAND,KMAX,TMAX,NSKIP                   00017000
            C                                                                  00018000
      000002    READ(5,ONE)                                                    00019000
      000065  97 FORMAT(1H-,*KK=*,    N=*,I4,* NMAX=*,I5,*NRAND=*,I5,*TMAX*,E15.5)00020000
      000005    TMAX=TMAX-8.                                                   00021000
      000007    TST=0.$TMEAS=0.$TOTI    ).$TGSPRI=0.$TGSPST=0.$TKAL=0.$TDIF=0.$EP=0.00022000
      000015    WRITE(6,97) KK,N,NMAX,NRAND,TMAX                               00023000
            C                                                                  00024000
      000031    DO 2  I=1,KMAX $DO 2 J=1,6                                     00025000
      000042    BP(I,J)=0.                                                     00026000
      000043  2 B(I,J)=0.                                                      00027000
      000047    CALL SECOND(T0)                                                00028000
      000050    CALL SECOND(TO) $WRITE(6,101) T0                               00029000
      000060  101 FORMAT(1H ,*T0=*,E16.6)                                      00030000
      000062    DO 2 I=1,KMAX $ DO 2 J=1,10                                    00031000
      000071    AP(I,J)=0.                                                     00032000
      000072  2 A(I,J)=0.                                                      00033000
      000076    IF (NSKIP .GT. 100) GO TO 1                                    00034000
      000101    IF (NSKIP.LE. 0) GO TO 1                                       00035000
      000102    LL1=NSKIP*(3*KMAX+4)                                           00036000
            C                                                                  00037000
            C                                                                  00038000
      000105  1 N=N+1                                                         00039000
      000107    NSTATE=2                                                       00040000
      000107    NMEAS=1                                                        00041000
      000110    RMIN=0.                                                        00042000
      000111    RMAX=3.                                                        00043000
      000113    NM=6                                                           00044000
      000114    EPS=.0005                                                      00045000
      000115    DELL=.0005                                                     00046000
      000116    BET=1.                                                         00047000
      000120    BETJ=1.                                                        00048000
      000120    X0(1)=3.
      000121    X0(2)=3.
      000122    XKPRI(1)=3.
      000123    XKPRI(2)=3.
      000124    VKPRI(1,1)=1.                                                  00053000
      000125    VKPRI(1,2)=0.                                                  00054000
      000126    VKPRI(2,1)=0.                                                  00055000
      000127    VKPRI(2,2)=1.                                                  00056000
      000127    R(1)=.01
      000131    Q(1,1)=.1                                                      00058000
      000132    Q(1,2)=.05                                                     00059000
```

Reproduced from best available copy.

```
000134          Q(2,1)=.05                                                      00060000
000134          Q(2,2)=.1                                                       00061000
000135          NPRI=1                                                          00062000
000136          NM=5                                                            00063000
000137          BETA(1)=1.                                                      00064000
000141          DO 200 I=1,2                                                    00065000
000143          CPRI(I,1)=XKPRI(I)                                              00066000
000144           XSPRI(I,1)=XKPRI(I)                                            00067000
000146          DO 200 J=1,2                                                    00068000
000156          JPRI(I,J,1)=VKPRI(I,J)                                          00069000
000157            VSPRI(I,J,1)=VKPRI(I,J)                                       00070000
000161    20    CONTINUE                                                        00071000
000163          K=1                                                             00072000
000164          PHI(1,1)=1.
000165          PHI(1,2)=0.                                                     00074000
000166          PHI(2,1)=0.                                                     00075000
000167          PHI(2,2)=1.                                                     00076000
000170          CALL SECOND(T1)                                                 00077000
000171          CALL FILGSV(X0,NM)                                              00078000
000173          CALL SECOND(T2) $ TOTIM=TOTIM+T2-T1                             00079000
000200          CALL SRTGS(KMAX)                                                00080000
        C                                                                       00081000
000202          CALL SECOND(TC) $ IF(T).GT.TMAX)GO TO 5                         00082000
000210          IF(N .LT. NMAX) GO TO 1                                         00083000
000212        5 WRITE(6,101) T0                                                 00084000
000220          WRITE(6,97) KK,N,NMAX,NRAND,TMAX                                00085000
000236          WRITE(6,89) TST,TMEAS,TOTIM,TGSPRI,TGSPST,TKAL,TDIF,T0,EP       00086000
000264       89 FORMAT(1H-,9F11.5,/)                                            00087000
000264          CALL OUTPGS(N,KMAX)                                             00088000
000266          STOP                                                            00089000
000270          END                                                             00090000
```

Reproduced from best available copy.

PROGRAM LENGTH INCLUDING I/O BUFFERS
004447

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

1	-	000116	5	-	000213	89	-	000356	97	-	000333
101	-	000345									

BLOCK NAMES AND LENGTHS

GSJ	-	02446	GS1	-	000107	ONE	-	000264	K2	-	000776
K5	-	000 17	S7	-	000027	K8	-	000017	K9	-	001700
S19	-	00027									

VARIABLE ASSIGNMENTS

A	-	000 6 01	AK	-	000170004	AKP	-	000454004	ALPM	-	000024002
AM	-	000 6 02	AP	-	000454010	AS	-	000226004	ASP	-	000512004
B	-	000113 01	BET	-	000013007	BETA	-	021450001	BETO	-	000014007
BM	-	000 36002	BP	-	001414010	CPRI	-	000000001	DELL	-	000024006
DPRI	-	000 57 01	EP	-	000025006	EPK1	-	000000004	EPK1P	-	000264004
EPK2	-	000 36004	EPK2P	-	000322004	EPS	-	000025006	EPS1	-	000074004
EPS1P	-	000 36 04	EPS2	-	000132004	EPS2P	-	000416004	H	-	000004006
HT	-	000 12006	I	-	000376	J	-	000377	K	-	000003011
KK	-	000 10007	KMAX	-	000100007	LL1	-	000400	N	-	000372

NM	-	000166C2	NMAX	-	000373	NMEAS	-	000023C06	NPO	-	000550C04
NPR	-	000516C4	NPRI	-	024404C11	NRAND	-	000015C07	NSKIP	-	000375
NSTATE	-	000.22C06	PHI	-	000000C16	PHIT	-	000006C06	Q	-	000015C06
R	-	000 210.6	RMAX	-	000012C17	RMIN	-	000011C07	TDIF	-	000006C07
TGSPRI	-	000013C07	TGSPST	-	000004C17	TKAL	-	000C05C07	TMAX	-	000374
TMEAS	-	000.01C07	TOTIM	-	000002C17	TST	-	000000C07	T0	-	000007C07
T1	-	000401	T2	-	000402	VKPOST	-	000002C05	VKPRI	-	000010C05
VSPOST	-	0000 C11	VSPRI	-	000740C13	X	-	000000C11	XKPOST	-	000000C05
XKPRI	-	000 66C.5	XPNT	-	000644C14	XSPOST	-	000004C11	XSPRI	-	000000C03
X0	-	000376	Z	-	000314C16	ZK	-	000014C05	ZPNT	-	000740C04

START OF CONSTANTS
000322

START OF TEMPORARIES
000362

START OF INDIRECTS
00.366

UNUSED COMPILER SPACE
05530.

```
                SUBROUTINE MEAS(Z,RMIN,RMAX,BET, R)
000007          COMMON/GS1/ AM(10,2),ALPM(10),BM(10,2,2),NM
000007          SR=SQRT(R)
000014          TOT=0.
000014           DD=(RMAX-RMIN)/FLOAT(NM)
000020          SZ=SIN(Z)
000021                          CZ=COS(Z)
000026          SB=SIN(BET)
000033                            CB=COS(BET)
000041          SZ2=SZ**2
000042                          CZ2=CZ**2
000043          Q2I=1./(.7*DD)**2
000046                                  R2I=1./SR**2
000051          DO 30 L=1,NM
000111          I=L
000112          RN=FLOAT(L)*DD+RMIN
000114          IF (NM .EQ. 1) RN=(RMAX-RMIN)*.5
000120          A1=CZ/RN
000122                          A2=-SZ/RN
000123          XH=SB+RN*SZ
000126                            YH=CB+RN*CZ
000130          A12=A1**2
000131                          A22=A2**2
000133          A=A12*R2I+SZ2*Q2I
000136          D=A22*R2I+CZ2*Q2I
000140          B=-2.*((A12*XH+A1*A2*YH)*R2I+(SZ2*SB+CB*SZ*CZ+RN*SZ)*Q2I)
000152          E=-2.*((A22*YH+A1*A2*XH)*R2I+(CZ2*CB+SB*SZ*CZ+RN*CZ)*Q2I)
000164          C=2.*(R2I*A1*A2+Q2I*SZ*CZ)
000170          BM(I,2,2)=A
000171            BM(I,1,1)=D
000174           BM(I,1,2)=C*.5
000176           BM(1,2,1)=C*.5
000201          DET=A*D-BM(I,1,2)**2
000202          AM(I,2)=-.5*(D*B-E*C*.5)/DET
000207          AM(I,1)=-.5*(-B*.5*C+A*E)/DET
000215           ALPM(I)=1.
000220          TOT=TOT+ALPM(I)
000221       3  CONTINUE
000225          TOTM1=1./TOT
000227          DO 31 I=1,NM
000240      31  ALPM(I)=ALPM(I)*TOTM1
000242          RETURN
000242          END
```

SUBPROGRAM LENGTH
000400

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

BLOCK NAMES AND LENGTHS
GS1 - 000107

VARIABLE ASSIGNMENTS
A - 000371 ALPM - 000024C01 AM - 000000C01 A1 - 000363

A12	=	000367	A2	=	000364	A22	=	000370	B2	=	000373
BM	=	000376	C	=	000375	CB	=	000353	CZ	=	000351
CZ2	=	000355	D	=	000372	DD	=	000347	DET	=	000376
E	=	000374	I	=	000361	L	=	000360	NM	=	000106C01
Q2I	=	000356	RN	=	000352	R2I	=	000357	SB	=	000352
SR	=	000345	SZ	=	000350	SZ2	=	000354	TOT	=	000346
TOTM1	=	000377	XM	=	000365	YM	=	000366			

START OF CONSTANTS
000245

START OF TEMPORARIES
001254

START OF INDIRECTS
001337

JNJSED COMPILER SPACE
155200

```
                SUBROUTINE FILGSV(X0,N4)
000004            COMMON/ONE/XSPRI(2,30),VSPRI(2,2,30)
000004          COMMON/GS0/ CPRI(2,1500),DPRI(2,2,1500),BETA(1500),NPRI
000004          DIMENSION X0(2)
000004          COMMON/C15/ W(2),V(1)
000004          COMMON/S19/X(2),ZZZZ,K,XSPOST(2,30),VSPOST(2,2,30)
000004          COMMON/K5/XKPOST(2),VKPOST(2,2),XKPRI(2),VKPRI(2,2),ZK(1)
000004          COMMON/S7/PHI(2,2),H(1,2),PHIT(2,2),HT(2),Z(1),Q(2,2),R(1),NSTATE,
               1 NMEAS,DELL,EPS,EP
000004          COMMON/K8/TST,TMEAS,TOTIM,TGSPRI,TGSPST,TKAL,TDIF,T0
               1,KMAX,RMIN,RMAX,BET,BETO,NRAND,KK
          C     INITIALIZE        ***************************************
000004          AA1=SQRT(Q(1,1))
000006          AA2=Q(1,2)/AA1
000010          AA3=SQRT(Q(2,2)-AA2*AA2)
          C
000015          DO 201 I=1,NSTATE
000020          DO 201 J=1,NRAND
000021      201 CALL RANSS(KK,W(I))
000033          A1=SQRT(VKPRI(1,1))
000035          A2=VKPRI(1,2)/A1
000037          A3=SQRT(VKPRI(2,2)-A2*A2)
000043          W(2)=W(1)*A2+W(2)*A3  $ W(1)=W(1)*A1
000050          X(1)=X0(1)+W(1)  $  X(2)=X0(2)+W(2)
          C     INITIAL STATE DETERMINED   AND  SET  TO START MAJOR LOOP
000055        1 CONTINUE
000055          Z(1)=0.
000056          IF (K.GE.KMAX) GO TO 205
          C     TAKE MEASUREMENT
000061          SB=SIN(BET)  $  CB=COS(BET)
000064          Z(1)=ATAN2(X(2)-SB,X(1)-CB)
000074          DO 203 I=1,NRAND
000077      203 CALL RANSS(KK,V(1))
000105          V(1)=V(1)*SQRT(R(1))  $  Z(1)=Z(1)+V(1)
000114      205 CONTINUE
000114          ZZZZ=Z(1)
000116          CALL BUFFGS
000117          IF (NPRI*NM.GT.1500) GO TO 195
000125          CALL SECOND(T1)
000126          SZ=SIN(Z(1))  $ CZ=COS(Z(1))
000133          RMID=(XSPRI(2,K)-SB)*SZ+(XSPRI(1,K)-CB)*CZ
000144          IF (RMID .LE. 0.) RMID=0.
000150          SIG=SQRT(VSPRI(1,1,K)+VSPRI(2,2,K))
000157          RMAX=RMID+3.*SIG $ RMIN=RMID-2.*SIG
000164          IF (RMIN .LE. 0.) RMIN=0.
000170          ZZ=Z(1)
000171          RR=R(1)
000173          CALL MEAS(ZZ,RMIN,RMAX,BET,RR)
000177          CALL SECOND(T2) $ TMEAS=TMEAS+T2-T1
000204          CALL SECOND(T1)
000206          CALL GSPOST
000207          CALL SECOND(T2) $ TGSPST=TGSPST+T2-T1
000214          CALL SECOND(T1)
000216          CALL DROP2
000217          CALL SECOND(T2) $ EP=EP+T2-T1
000224          CALL SECOND(T1)
000226          CALL COMBN3
000227          CALL SECOND(T2) $ TDIF=TDIF+T2-T1
```

```
000234          CALL SECOND(T1)                                          00189000
000236           CALL BUF2                                               00190000
000237          IF (K .GE. KMAX) GO TO 5
000243          CALL SECOND(T2) $ TST=TST+T2-T1
              C UPDATE TIME INDIX                                        00191000
000250          K=K+1                                                    00192000
000252          BET=BET+BETO                                             00193000
000254          DO 3 I=1,NSTATE                                          00194000
000257          DO 3 J=1,NRAND                                           00195000
000260        3 CALL RANSS(KK,W(I))                                      00196000
000272          W(2)=W(1)*AA2+W(2)*AA3 $ W(1)=W(1)*AA1                   00197000
000276          TEMP2=X(1)  $X(1)=PHI(1,1)*X(1)+PHI(1,2)*X(2)+W(1)       00198000
000303          X(2)=PHI(2,1)*TEMP2+PHI(2,2)*X(2)+W(2)                   00199000
000307          CALL SECOND(T1)                                          00200000
000311          CALL GSPRI(PHI,Q)                                        00202000
000313          CALL SECOND(T2) $ TGSPRI=TGSPRI+T2-T1                    00203000
000320          GO TO 1                                                  00204000
000322        5 CONTINUE                                                 00205000
              C                                                          00206000
              C                                                          00207000
000322          RETURN                                                   00208000
000323      195 STOP                                                     00209000
000325          END                                                      00210000
                                                                         00211000
```

SUBPROGRAM LENGTH
000404

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

1	-	000056	5	-	000323	195	-	000324	205	-	000115

BLOCK NAMES AND LENGTHS

ONE	-	000264	GS0	-	024405	C15	-	000003	S19	-	000270
K5	-	000015	S7	-	000327	K8	-	000017			

VARIABLE ASSIGNMENTS

AA1	-	000361	AA2	-	000362	AA3	-	000363	A1	-	000366
A2	-	000367	A3	-	000370	BET	-	000013C07	BETA	-	021450C02
BETO	-	000 14C07	C8	-	000372	CPRI	-	000000C02	CZ	-	000375
DPRI	-	000567 002	EP	-	000260006	H	-	000004C06	HT	-	000012C06
I	-	000364	J	-	000365	K	-	000003C04	KK	-	000016C07
KMAX	-	000 1.C07	NPRI	-	024404C02	NRAND	-	000015C07	NSTATE	-	000022C06
PHI	-	000 1.C06	PHIT	-	000006C06	Q	-	000015C06	R	-	000021C06
R1AX	-	000 12C07	RMID	-	000376	RMIN	-	000011C07	RR	-	000401
S8	-	000371	SIG	-	000377	SZ	-	000374	TDIF	-	000006C07
TEMP2	-	000403	TGSPRI	-	000003C07	TGSPST	-	000004C07	TMEAS	-	000001C07
TST	-	000.0 C07	T1	-	000373	T2	-	000402	V	-	000002C03
VKPOST	-	000 02C05	VKPRI	-	000010C05	VSPOST	-	000100C04	VSPRI	-	000074C01
W	-	000.0 C03	X	-	000000C04	XKPOST	-	000000C05	XKPRI	-	000006C05
X3POST	-	000 04C04	XSPRI	-	000000C01	Z	-	000014C06	ZK	-	000014C05
ZZ	-	00040	ZZZZ	-	000002C04						

START OF CONSTANTS
000330

```
      SUBROUTINE BUFFGS
        COMMON/ONE/XSPRI(2,30),VSPRI(2,2,30)
      COMMON/GS4/CPOST(2,1500),DPOST(2,2 ,1500)   ,ALPHA(1500),NPOST
      COMMON/GS0/ CPRI(2,1500),DPRI(2,2,1500),BETA(1500),NPRI
      COMMON/K2/EPK1(30),EPK2(30),EPS1(30),EPS2(30),AK(30),AS(30)
     1,EPK1P(30),EPK2P(30),EPS1P(30),EPS2P(30),AKP(30),ASP(30)
     1,NPO(30),NPR(30),XPNT(2,30),ZPNT(30)
      COMMON/K5/XKPOST(2),VKPOST(2,2),XKPRI(2),VKPRI(2,2),ZK(1)
      COMMON/S19/X(2),ZZZZ,K,XSPOST(2,30),VSPOST(2,2,30)
      AA=1.E-10
      XPNT(1,K)=X(1)$XPNT(2,K)=X(2)$ZPNT(K)=ZZZZ
      NPO(K)=NPOST$NPR(K)=NPRI
      DO30 I=1,2$XSPRI(I,K)=0.$XSPOST(I,K)=0.$DO30 J=1,2$VSPRI(I,J,K)=0.
    3 VSPOST(I,J,K)=0.
      DO 60 I=1,NPRI $ DO 60 J=1,2
      XSPRI(J,K)=BETA(I)*CPRI(J,I)+XSPRI(J,K)
      DO 61 L=1,2
   61 VSPRI(J,L,K)=VSPRI(J,L,K)+BETA(I)*(DPRI(J,L,I)+CPRI(J,I)*CPRI(L,I)
     1)
   60 CONTINUE
      DO 62 J=1,2$DO 62 L=1,2
   62 VSPRI(J,L,K)=VSPRI(J,L,K)-XSPRI(J,K)*XSPRI(L,K)
      RETURN
      ENTRY BUF2
      DO 70 I=1,NPOST$ DO 70 J=1,2
      XSPOST(J,K)=ALPHA(I)*CPOST(J,I)+XSPOST(J,K)
      DO 75 L=1,2
   75 VSPOST(J,L,K)=VSPOST(J,L,K)+ALPHA(I)*(DPOST(J,L,I)+CPOST(J,I)*CPOS
     1T(L,I))
    7 CONTINUE
      DO 72 J=1,2$DO 72 L=1,2
   72 VSPOST(J,L,K)=VSPOST(J,L,K)-XSPOST(J,K)*XSPOST(L,K)
      EPS1(K)=X(1)-XSPOST(1,K)
      EPS1P(K)=X(1)-XSPRI(1,K)
      EPS2(K)=X(2)-XSPOST(2,K)
      EPS2P(K)=X(2)-XSPRI(2,K)
      AS(K)= EPS1(K)**2*VSPOST(2,2,K)-2.*VSPOST(1,2,K)*EPS1(K)*EPS2(K)+
     1EPS2(K)**2*VSPOST(1,1,K)
      ASP(K)=EPS1P(K)**2*VSPRI(2,2,K)-2.*VSPRI(1,2,K)*EPS1P(K)*EPS2P(K)+
     1EPS2P(K)**2*VSPRI(1,1,K)
      DET=VSPOST(1,1,K)*VSPOST(2,2,K)-VSPOST(1,2,K)**2
      IF(DET.LT. AA*AS(K)) GO TO 71
      IF(DET.LE. 0.)GO TO   71
      AS(K)=AS(K)/DET
      DET=VSPRI(1,1,K)*VSPRI(2,2,K)-VSPRI(1,2,K)**2
      IF(DET.LT. AA*AS(K)) GO TO 71
      IF(DET.LE. 0.)GO TO   71
      ASP(K)=ASP(K)/DET
      RETURN
   71 WRITE(6,73) DET,AS(K),VSPOST(1,1,K),VSPOST(1,2,K),VSPOST(2,2,K)
      WRITE(6,73) DET,ASP(K),VSPRI (1,1,K),VSPRI (1,2,K),VSPRI (2,2,K)
      WRITE(6,50)EPS1P(K),EPS2P(K),EPS1(K),EPS2(K),ASP(K),AS(K),K
    50 FORMAT(1H ,*EPK1=*,E12.5,2X,E12.5,2X,*EPS=*E12.5,2X,E12.5
     1          ,2X,*AK=*,E12.5,2X,*AS=*,E12.5,2X,*K=*,I4)
   73 FORMAT(1H ,*ERROR IN VSPOST*,2X,/, 7E12.4)
      END
```

SUBPROGRAM LENGTH
000474

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
50 - 000462 71 - 000305 73 - 000416

BLOCK NAMES AND LENGTHS
ONE - 000364 GS4 - 024405 GS0 - 024405 K2 - 000776
K5 - 000012 S19 - 000270

VARIABLE ASSIGNMENTS
AA - 000467 AK - 000170C04 AKP - 000454C04 ALPHA - 021450C02
AS - 000620C04 ASP - 000512C04 BETA - 021450C03 BUF2 - 000466
DPOST - 000600C02 DPRI - 000600C03 DET - 000473 DPOST - 005570C02
DPRI - 000570C03 EPK1 - 000300C04 EPK1P - 000264C04 EPK2 - 000036C04
EPK2P - 000220C04 EPS1 - 000374C04 EPS1P - 000360C04 EPS2 - 000132C04
EPS2P - 000160C04 I - 000477 J - 000471 K - 000003C06
L - 000472 NPD - 000550C04 NPOST - 024404C02 NPR - 000506C04
NPRI - 024400C03 VKPOST - 000020C05 VKPRI - 000010C05 VSPOST - 000100C06
VSPRI - 000740C01 X - 000300C06 XKPOST - 000000C05 XKPRI - 000006C05
XPNT - 000440C04 XSPOST - 000004C06 XSPRI - 000000C01 ZK - 000014C05
ZPNT - 000740C04 ZZZZ - 000020C06

START OF CONSTANTS
000375

START OF TEMPORARIES
000423

START OF INDIRECTS
000437

UNUSED COMPILER SPACE
055300

Reproduced from best available copy.

```
                SUBROUTINE SRTGS(KMAX)                                          00268000
000002          COMMON/K2/EPK1(30),EPK2(30),EPS1(30),EPS2(30),AK(30),AS(30)     00269000
               1,EPK1P(30),EPK2P(30),EPS1P(30),EPS2P(30),AKP(30),ASP(30)        00270000
               1,NPO(30),NPR(30),XPNT(2,30),ZPNT(30)                            00271000
000002          COMMON/K9/ A(30,10),AP(30,10),B(30,6),BP(30,6)
000002          WRITE(6,89)                                                     00273000
000036       89 FORMAT(1H-,*----------*,/)                                      00274000
000037          DO 3 I=1,KMAX                                                   00275000
000011          A(I,3)= A(I,3)+EPS1(I)$A(I,4)=A(I,4)+EPS2(I)                    00276000
000015          A(I,7)=A(I,7)+ABS(EPS1(I))$A(I,8)=A(I,8)+ABS(EPS2(I))           00277000
000023          A(I,10)=A(I,10)+AS(I)                                           00278000
000026          B(I,4)=B(I,4)+EPS1(I)**2$B(I,6)=B(I,6)+EPS2(I)**2               00279000
000032          B(I,5)=B(I,5)+EPS1(I)*EPS2(I)                                   00280000
000036          AP(I,3)=AP(I,3)+EPS1P(I) $ AP(I,4)=AP(I,4)+EPS2P(I)             00281000
000042          AP(I,7)=AP(I,7)+ABS(EPS1P(I)) $ AP(I,8)=AP(I,8)+ABS(EPS2P(I))   00282000
000052          AP(I,10)=AP(I,10)+ASP(I)                                        00283000
000062          BP(I,4)=BP(I,4)+EPS1P(I)**2 $ BP(I,6)=BP(I,6)+EPS2P(I)**2       00284000
000056          BP(I,5)=BP(I,5)+EPS1P(I)*EPS2P(I)                               00285000
000062        3 CONTINUE                                                        00286000
000064          WRITE(6,93)(I,EPS1P(I),EPS2P(I),ASP(I),EPS1(I),EPS2(I),AS(I),NPR(I00287000
               1),XPNT(1,I),XPNT(2,I),ZPNT(I),I=1,KMAX)                         00288000
000126       93 FORMAT(1H ,I3,6F10.5,I4,3F10.5)                                 00289000
000127          RETURN                                                          00290000
000130          END                                                             00291000
```

Reproduced from best available copy.

SUBPROGRAM LENGTH
000231

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
89 - 000136 93 - 000141

BLOCK NAMES AND LENGTHS
K2 - 000770 K9 - 001760

VARIABLE ASSIGNMENTS
A - 000000C02 AK - 000170C01 AKP - 000454C01 AP - 000454C02
AS - 000260C01 ASP - 000512C01 B - 001130C02 BP - 001414C02
EPK1 - 000000C01 EPK1P - 000264C01 EPK2 - 000036C01 EPK2P - 000322C01
EPS1 - 000740C01 EPS1P - 000360C01 EPS2 - 000132C01 EPS2P - 000416C01
I - 000003 NPO - 000550C01 NPR - 000606C01 XPNT - 000644C01
ZPNT - 000740C01

START OF CONSTANTS
000133

START OF TEMPORARIES
000145

START OF INDIRECTS
000166

UNUSED COMPILER SPACE
055400

```
                SUBROUTINE GSPOST                                                  00292000
000001          COMMON/GS1/ AM(10,2),ALPM(10),BM(10,2,2),NM                        00293000
000001          COMMON/GS6/ CPRI(2,1500),DPRI(2,2,1500),BETA(1500),NPRI
000001          COMMON/GS4/CPOST(2,1500),DPOST(2,2 ,1500)   ,ALPHA(1500),NPOST
000001          DIMENSION B(10,2,2),S(2,2),D(2)  ,PM1(2,2),WK(2,2),QM1(2,2)        00296000
000001              TOT=0. $ NPOST=NM*NPRI $L=0 $ DO 1 I=1,NM                      00297000
000007          DET=BM(I,1,1)*BM(I,2,2)-BM(I,1,2)**2  $  DETM1=1./DET              00298000
000013          B(I,1,1)=BM(I,2,2)*DETM1 $ B(I,2,2)=BM(I,1,1)*DETM1                00299000
000020          B(I,1,2)=-BM(I,1,2)*DETM1 $ B(I,2,1)=B(I,1,2)                      00300000
000023        1 CONTINUE                                                           00301000
000026          DO 30 J=1,NPRI                                                     00302000
000027          DET=DPRI(1,1,J)*DPRI(2,2,J)-DPRI(1,2,J)**2  $ DETM1=1./DET         00303000
000035          PM1(1,1)=DETM1*DPRI(2,2,J) $PM1(2,2)=DETM1*DPRI(1,1,J)             00304000
000042          PM1(1,2)=-DETM1*DPRI(1,2,J) $PM1(2,1)=PM1(1,2)                     00305000
000046          DO 30 I=1,NM                                                       00306000
000047          L=(I-1)*NPRI+J                                                     00307000
000053          DO 2 L1=1,2  $  DO 2 L2=1,2                                        00308000
000066        2 QM1(L1,L2)=PM1(L1,L2)+BM(I,L1,L2)                                  00309000
000073          DETQ=QM1(1,1)*QM1(2,2)-QM1(1,2)**2  $  DETQM1=1./DETQ              00310000
000077          DPOST(1,1,L)=QM1(2,2)*DETQM1 $DPOST(2,2,L)=QM1(1,1)*DETQM1         00311000
000104          DPOST(1,2,L)=-QM1(1,2)*DETQM1  $DPOST(2,1,L)=DPOST(1,2,L)          00312000
000112          DO 3L1=1,2$DO3L2=1,2$WK(L1,L2)=0.                                  00313000
000116          DO 3 L3=1,2                                                        00314000
000135        3 WK(L1,L2)=WK(L1,L2)+DPOST(L1,L3,L)*BM(I,L2,L3)                     00315000
000145          DO 4 L1=1,2                                                        00316000
000156        4 D(L1)=AM(I,L1)-CPRI(L1,J)                                          00317000
000161          DO 5 L1=1,2                                                        00318000
000170        5 CPOST(L1,L)=CPRI(L1,J)+WK(L1,1)*D(1)+WK(L1,2)*D(2)                 00319000
000177          S(1,1)=B(I,2,2)+DPRI(2,2,J)                                        00320000
000202          S(2,2)=B(I,1,1)+DPRI(1,1,J)                                        00321000
000205          S(1,2)=-B(I,1,2)-DPRI(1,2,J)                                       00322000
000210          DETM1=1./(S(1,1)*S(2,2)-S(1,2)**2)                                 00323000
000213          S(1,1)=S(1,1)*DETM1 $ S(2,2)=S(2,2)*DETM1 $ S(1,2)=S(1,2)*DETM1    00324000
000216          WT1=D(1)**2*S(1,1)+D(2)**2*S(2,2)+2.*D(1)*D(2)*S(1,2)              00325000
000225          ALPHA(L)=BETA(J)*ALPM(I)*EXP(-.5*WT1)                              00326000
000236        3 TOT=TOT+ALPHA(L)                                                   00327000
000245          TOTM1=1./TOT                                                       00328000
000247          DO 31 I=1,NPOST                                                    00329000
000255       31 ALPHA(I)=ALPHA(I)*TOTM1                                            00330000
000257          RETURN                                                             00331000
000257          END                                                                00332000
```

Reproduced from best available copy.

```
SUBPROGRAM LENGTH
000435

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

BLOCK NAMES AND LENGTHS
GS1    -  000107    GS6    -  024405    GS4    -  024405

VARIABLE ASSIGNMENTS
ALPHA  -  02145 C03 ALPM   -  000024C01 AM     -  000000C01 B      -  000326
BETA   -  02145 C 2 BM     -  000036C01 CPOST  -  000000C03 CPRI   -  000000C02
D      -  000402    DET    -  000423    DETM1  -  000424    DETQ   -  000430
```

```
GETQM1 -  000431    OPOST  -  005670C03 OPRI   -  005670C02 I      -  000422
J      -  000425    L      -  000421    L1     -  000426    L2     -  000427
L3     -  000432    NM     -  005166C01 NPOST  -  024404C03 NPRI   -  024404C02
PM1    -  000414    QM1    -  000414    S      -  000376    TOT    -  000420
TOTM1  -  000434    WK     -  000410    WT1    -  000433

START OF CONSTANTS
000262

START OF TEMPORARIES
000266

START OF INDIRECTS
000300

UNUSED COMPILER SPACE
055700
```

```
                SUBROUTINE GSPRI(PHI,Q)                                          00333000
000004          DIMENSION PHI(2,2),Q(2,2),T(2,2),C(2)                            00334000
000034          COMMON/GS0/ CPRI(2,1500),DPRI(2,2,1500),BETA(1500),NPRI
          C     ONE STAGE PREDICTED DENSITY  ***  ALSO  INITIAL STATE PDF        00336000
000004          DIMENSION TEMP4(2,2)                                             00337000
          C     A POSTERIORI DENSITY                                             00338000
000004          COMMON/GS4/CPOST(2,1500),DPOST(2,2 ,1500)  ,ALPHA(1500),NPOST
          C     SYSTEM FORMULATION AND SET UP                                    00340000
000004          NPRI=NPOST                                                       00341000
000005          DO 3 I=1,NPRI                                                    00342000
000007          BETA(I)=ALPHA(I)                                                 00343000
000011          DO 4 J=1,2 $ DO 4 L=1,2                                          00344000
000023        4 TEMP4(J,L)=DPOST(J,L,I)                                          00345000
000031            T(1,1)=PHI(1,1)**2*TEMP4 (1,1)+2.*PHI(1,1)*PHI(1,2)*TEMP4 (1,200346000
                2)+PHI(1,2)**2*TEMP4 (2,2)                                       00347000
                  T(2,2)=PHI(2,1)**2*TEMP4 (1,1)+2.*PHI(2,2)*PHI(1,2)*TEMP4 (1,200348000
                2)+PHI(2,2)**2*TEMP4 (2,2)                                       00349000
000047            T(1,2)=PHI(1,1)*(PHI(2,1)*TEMP4 (1,1)+PHI(2,2)*TEMP4 (2,1))+   00350000
                1PHI(2,2)*PHI(1,2)*TEMP4 (2,2)+PHI(2,1)*PHI(1,2)*TEMP4 (1,2)     00351000
                  T(2,1)=T(1,2)                                                  00352000
000061            DO 5 J=1,2$DO 5 L=1,2                                          00353000
000074        5 DPRI(J,L,I)=T(J,L) +Q(J,L)                                       00354000
000103            DO 2 J=1,2                                                     00355000
000111        2 C(J)=CPOST(J,I)                                                  00356000
000116            C(1)=PHI(1,1)*C(1)+PHI(1,2)*C(2)                               00357000
000122            C(2)=PHI(2,1)*CPOST(1,I)+PHI(2,2)*C(2)                         00358000
000129            DO 3 J=1,2                                                     00359000
000133        3 CPRI(J,I)=C(J)                                                   00360000
000141          CONTINUE                                                         00361000
000141            RETURN                                                         00362000
000142            END                                                            00363000
```

Reproduced from best available copy.

```
SUBPROGRAM LENGTH
000212

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
1       -  000146

BLOCK NAMES AND LENGTHS
GS0     -  024403     GS4     -  024405

VARIABLE ASSIGNMENTS
ALPHA   -  021450 C02 BETA    -  021450C01 C       -  000201    CPOST   -  000000C02
CPRI    -  000000 C01 DPOST   -  005670C02 DPRI    -  005670C01 I       -  000207
J       -  000201     L       -  000211    NPOST   -  024404C02 NPRI    -  024404C01
T       -  000175     TEMP4   -  000203

START OF CONSTANTS
000145

START OF TEMPORARIES
000147

START OF INDIRECTS
```

```
                SUBROUTINE DROP2                                                00364000
000001          COMMON/GS4/C(2,150),D(2,2,150),W(1500),N                        00365000
000001          COMMON/S7/PHI(2,2),H(1,2),PHIT(2,2),HT(2),Z(1),Q(2,2),R(1),NSTATE,00366000
               1 NMEAS,DELL,EPS,EP                                              00367000
000001          QQQ=EPS                                                         00368000
000002          AMAX=AMAX(W,N)*.1                                               00369000
000006          IF (WMAX .GT. EPS) GO TO 4                                      00370000
000012          EPS=WMAX                                                        00371000
000013          WRITE (6,90) EPS                                                00372000
000020       90 FORMAT(1H-,50X,*++++++++++++++++++++*,/,50X,*--EPS=*,E15.5,*++++*,/00373000
               1 50X,*....................*)                                   00374000
000026          M=1 $ WTOT=0.                                                   00375000
000022          DO 1 J=1,N                                                      00376000
000023          IF(W(J).LT. EPS) GO TO 1                                        00377000
000026          M=M+1  $  W(M)=W(J)  $  WTOT=WTOT+W(M)                          00378000
000033          DO 2 I=1,NSTATE                                                 00379000
000035          C(I,M)=C(I,J)                                                   00380000
000041          DO 2 L=1,NSTATE                                                 00381000
000053        2 D(I,L,M)=D(I,L,J)                                               00382000
000061        1 CONTINUE                                                        00383000
000063          WTOTM1=1./WTOT                                                  00384000
000065          DO 3 J=1,M                                                      00385000
000072        3 W(J)=W(J)*WTOTM1                                                00386000
000074          EPS=QQQ                                                         00387000
000075          N=M                                                             00388000
000077          RETURN                                                          00389000
000077          END                                                             00390000
```

Reproduced from best available copy.

SUBPROGRAM LENGTH
000146

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

1	- 000 61	4	- 000321	90	- 000165		

BLOCK NAMES AND LENGTHS

GS4	- 024460	S7	- 000327

VARIABLE ASSIGNMENTS

C	- 000 6C.1	D	- 000307C01	EPS	- 000325C02	H	-	000004C02
HT	- 000 12C.2	I	- 000135	J	- 000134	L	-	000136
M	- 000.3C	N	- 024464C01	NSTATE	- 000022C02	PHI	-	000000C02
PHIT	- 000 6C.2	Q	- 000315C02	QQQ	- 000130	R	-	001021C02
W	- 02145 C.1	WMAX	- 000 31	WTOT	- 000133	WTOTM1	-	000137
Z	- 000 14C.2							

START OF CONSTANTS
000102

START OF TEMPORARIES
000126

START OF INDIRECTS
000124

```
      SUBROUTINE COMBN3
      DIMENSION DMN(2,2),AMN(2)
      COMMON/GS4/CP(2,1500),DP(2,2,1500),ALP(1500),NP
      COMMON/C6/WA(2),TEMP(1,2),TEMP1(1),TEMP2(2),TEMP4(2,2),C(2),A(1)
     1,AT(2),AA(2)
      COMMON/S7/PHI(2,2),H(1,2),PHIT(2,2),HT(2),Z(1),Q(2,2),R(1),NSTATE,
     1 NMEAS,DELL,EPS,EP
      CALL SECOND(T1)
      N=NSTATE $ M=NMEAS
      ZZ=Z(1)
      DO 11 I=1,NP
      AMN(2)=CP(2,I) $L=I
      DO 12 J=I,NP
      IF(CP(2,J)-AMN(2))13,12,12
   13 CONTINUE
      AMN(2)=CP(2,J) $ L=J
   12 CONTINUE
      BMN=ALP(L)
      ALP(L)=ALP(I) $ALP(I)=BMN
      AMN(1)=CP(1,L)
      DO 14 K1=1,2
      CP(K1,L)=CP(K1,I) $CP(K1,I)=AMN(K1)
      DO 14 K2=1,2
      DMN(K1,K2)=DP    (K1,K2,L)
      DP(K1,K2,L)=DP(K1,K2,I)
      DP(K1,K2,I)=DMN(K1,K2)
   14 CONTINUE
   11 CONTINUE
      M=0
      DO 40 K=1,NP
      WK=ALP(K)
      LNC=0 $ L1=1
   43 IF(L1 .EQ. K) GO TO 3
      J=K-L1
      Z(1)=0.
      WJ=ALP(J)
      IF(WJ .EQ.0.)GOTO 41
      WM1=1./(WK+WJ)
      A(1)=WJ*WK*WM1
      DELLS=(DELL/A(1))**2
      DO 2 I=1,N
      Z(1)=Z(1)+(CP(I,K)-CP(I,J))**2
      IF(Z(1) .GT. DELLS) GO TO 42
      DO 2 L=I,N
      Z(1)=Z(1)+ABS(DP(L,I,J)-DP(L,I,K))
      IF(Z(1).GT.DELLS) GO TO 42
    2 CONTINUE
      DO 6 I=1,N
      CP(I,J)=(CP(I,J)*WJ+WK*CP(I,K))*WM1
      DO 6 L=1,N
      DP(I,L,J)=(DP(I,L,J)*WJ+WK*DP(I,L,K)+A(1)*(CP(I,J)-CP(I,K))*(CP(L,
     1J)-CP(L,K)))*WM1
    6 CONTINUE
      ALP(K)=0.
      ALP(J)=WK+WJ
      GO TO 40
    3 M=M+1
      IF(K.EQ.M)GO TO 40
```

Reproduced from best available copy.

```
                ALP(M)=ALP(K)$ALP(K)=0.                                  00449000
                DO 5 I=1,N                                               00450000
                CP(I,M)=CP(I,K)                                          00451000
                DO 5 L=1,N                                               00452000
                DP(I,L,M)=DP(I,L,K)                                      00453000
              5 CONTINUE                                                 00454000
                GO TO 46                                                 00455000
             42 LNC=LNC+1                                                00456000
                IF (LNC .GT.2) L1=K-1                                    00457000
             41 L1=L1+1                                                  00458000
                GO TO 43                                                 00459000
              4 CONTINUE                                                 00460000
                NP=M                                                     00461000
                Z(1)=ZZ                                                  00462000
                RETURN                                                   00463000
                END                                                      00464000
```

Reproduced from best available copy.

SUBPROGRAM LENGTH
000466

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

2	-	000216	3	-	000231	12	-	000025	13	-	000021
43	-	000270	41	-	000270	42	-	000263	43	-	000100

BLOCK NAMES AND LENGTHS

GS4	-	000466	C6	-	000022	S7	-	000027			

VARIABLE ASSIGNMENTS

A	-	000202	AA	-	000202	ALP	-	021450C01	AMN	-	000355
AT	-	000206	BMN	-	000366	C	-	000013C02	CP	-	000000C01
DELL	-	000240	DELLS	-	000377	DMN	-	000351	DP	-	005570C01
H	-	000246	HT	-	000312C03	I	-	000363	J	-	000365
K	-	000372	K1	-	000367	K2	-	000370	L	-	000364
LNC	-	000373	L1	-	000374	M	-	000361	N	-	000360
NMEAS	-	000230	NP	-	024404C01	NSTATE	-	000022C03	PHI	-	000000C03
PHIT	-	000206	Q	-	000315C03	R	-	000021C03	TEMP	-	000002C02
TEMP1	-	000244	TEMP2	-	000305C02	TEMP4	-	000037C02	T1	-	000357
WA	-	000202	WJ	-	000375	WK	-	000372	WM1	-	000376
Z	-	000140	ZZ	-	000362						

START OF CONSTANTS
000302

START OF TEMPORARIES
000304

START OF INDIRECTS
000323

UNUSED COMPILER SPACE
000570

```
                      FUNCTION ARMX(X,N)                                00465000
000004                DIMENSION X(N)                                    00466000
000004                ARMX=X(1)                                         00467000
000005                DO 1 J=2,N                                        00468000
000011                IF(X(J)-ARMX)1,1,2                                00469000
000013           2    ARMX=X(J)                                         00470000
000014           1    CONTINUE                                          00471000
000015                RETURN                                            00472000
000017                END                                               00473000
```

SUBPROGRAM LENGTH
000031

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
1 - 000015 2 - 000014

BLOCK NAMES AND LENGTHS

VARIABLE ASSIGNMENTS
ARMX - 000027 J - 000036

START OF CONSTANTS
000022

START OF TEMPORARIES
000023

START OF INDIRECTS
000025

UNUSED COMPILER SPACE
057100

Reproduced from best available copy.

```
                SUBROUTINE  RANSS(KK,W)                                 00474000
000004          COMMON  / CM6/ PI, TWOPI                                00475000
000014          COMMON  / CM10/ IST, IR(2), X(2)                        00476000
000034          DATA  (IT=1)
000004          IR(2)=KK                                                00478000
000004          NN=0                                                    00479000
000005          IF (IT.NE.1) GO TO 10                                   00480000
000010          IT=0                                                    00481000
000016          PI=4.*ATAN(1.)                                          00482000
000015          TWOPI=2.*PI                                             00483000
000016          IST=KK                                                  00484000
000017          IR(1)=0                                                 00485000
000020          N=1                                                     00486000
000022      1   W=GRN01(NN)                                             00487000
000026          KK=IR(2)                                                00488000
000027          RETURN                                                  00489000
000027          END                                                     00490000


SUBPROGRAM LENGTH
000044

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
10      -  000023

BLOCK NAMES AND LENGTHS
CM6     -  000002    CM10    -  000005

VARIABLE ASSIGNMENTS
IR      -  000001C2 IST     -  000001C2 IT      -  000042    NN      -  000043
PI      -  000001C1 TWOPI   -  000001C1 X       -  000003C2

START OF CONSTANTS
000032

START OF TEMPORARIES
000036

START OF INDIRECTS
000042

UNUSED COMPILER SPACE
057100
```

Reproduced from best available copy.

```
                FUNCTION  GRN01(NN)
000002          COMMON  / CM6/ PI, TWOPI                                  00491000
000002          COMMON  /CM10/ IST, IR(2), X(2)                           00492000
000002          DATA  (IM=5437558138 87), (JM=2513271)                    00493000
000002          J=IR(1)
000003          IF (NN.NE.0) GO TO 1,                                     00495000
000005          GO TO (20,40), J                                          00496000
000012      1   IR(2)=IST                                                 00497000
000014      2   IR(1)=2                                                   00498000
000015          DO 3, I=1,2                                               00499000
000025          IR(2)=MOD(IR(2)*JM,IM)                                    00500000
000032      3   X(I)=FLOAT(IR(2))/FLOAT(IM)                               00501000
000035          X1=SQRT(ABS(-2.*ALOG(X(1))))                              00502000
000043          X(2)=TWOPI*X(2)                                           00503000
000045          X(1)=X1*SIN(X(2))                                         00504000
000050          X(2)=X1*COS(X(2))                                         00505000
000053          GRN01=X(1)                                                00506000
000055          RETURN                                                    00507000
000056      4   IR(1)=1                                                   00508000
000057          GRN01=X(2)                                                00509000
000061          RETURN                                                    00510000
000061          END                                                       00511000
                                                                          00512000
```

Reproduced from best available copy.

```
SUBPROGRAM LENGTH
000126

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
10      -  000013    20      -  000015    40      -  000057

BLOCK NAMES AND LENGTHS
CM6     -  000002    CM10    -  000005

VARIABLE ASSIGNMENTS
GRN01   -  000012    I       -  000124    IM      -  000121    IR      -  000001C02
IST     -  000001C02 J       -  000123    JM      -  000122    TWOPI   -  000001C01
X       -  000001C02 X1      -  000125

START OF CONSTANTS
000064

START OF TEMPORARIES
000066

START OF INDIRECTS
000111

UNUSED COMPILER SPACE
057000
```

```
                SUBROUTINE OUTPGS(N,KMAX)                                       00542000
000004          COMMON/K97/A(30,10),AP(30,10),B(30,6),BP(30,6)
000004          COMMON/R10/C(10)                                                00544000
000004          GO TO 99                                                        00545000
000004          WRITE(6,98) C                                                   00546000
***NP ********************** OUTPGS
000012       98 FORMAT(1H-,10F11.2)                                             00547000
000014          FFF=C(4)                                                        00548000
000019          DO  7 I=1,10                                                    00549000
000023        7 C(I)=C(I)/FFF                                                   00550000
000025          WRITE(6,99) C                                                   00551000
000032       99 FORMAT(1H-,10F11.7)                                             00552000
000034        1 CONTINUE                                                        00553000
000034          PRINT 95                                                        00554000
000040       95 FORMAT(1H1)                                                     00555000
000042          FNM1=1./FLOAT(N)                                                00556000
000044          DO 4 I=1,KMAX & DO 4 J=1,10                                     00557000
000054          AP(I,J)=AP(I,J)*FNM1                                            00558000
000065        4 A(I,J)=A(I,J)*FNM1                                              00559000
000063          DO 41 I=1,KMAX&DO 41 J=1,6                                      00560000
000073          BP(I,J)=BP(I,J)*FNM1                                            00561000
000074       41 B(I,J)=B(I,J)*FNM1                                              00562000
000102          WRITE(6,93) (A(I,3),A(I,4),B(I,4),B(I,5),B(I,6),A(I,7),A(I,8),  00563000
               1 A(I,10),I=1,KMAX)                                              00564000
      C                                                                         00565000
      C                                                                         00566000
000135       93 FORMAT(1H ,2F11.7,2X,3F11.7,5X,2F11.7,2X,F11.7)                 00567000
000137      100 FORMAT(8F10.6)                                                  00568000
000137          PRINT 95                                                        00569000
000142     1003 FORMAT(1H-,*APRIORI*,/)                                         00570000
000144          WRITE(6,1003)                                                   00571000
000147          WRITE(6,93) (AP(I,3),AP(I,4),BP(I,4),BP(I,5),BP(I,6),AP(I,7),   00572000
               1AP(I,8),AP(I,10),I=1,KMAX)                                      00573000
      C                                                                         00574000
      C                                                                         00575000
000214          RETURN                                                          00576000
000215          END                                                             00577000
```

Reproduced from best available copy.

SUBPROGRAM LENGTH
000253

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

10	-	000036	93	-	000226	95	-	000223	98	-	000213
99	-	000017	100	-	000234	1003	-	000236			

BLOCK NAMES AND LENGTHS

K9	-	000176	R10	-	000012

VARIABLE ASSIGNMENTS

A	-	000100	AP	-	000454001	B	-	001139C01	BP	-	001414C01
C	-	000000	FFF	-	000247	FNM1	-	000251	I	-	000250
J	-	000252									

START OF CONSTANTS

```
JJ(21J

START OF TEMPORARIES
JJJ241

START OF INDIRECTS
JJJ245

JNJSED COMPIL R SPACE
J562JJ

VP********NO PATH TJ THIS STATEMENT
    JJJJJ5
```

```
                SUBROUTINE  SECOND(T)                                          00578000
000002          T=0.                                                           00579000
000003          RETURN                                                         00580000
000003          END                                                            00581000
```

SUBPROGRAM LENGTH
000011

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

BLOCK NAMES AND LENGTHS

VARIABLE ASSIGNMENTS

START OF CONSTANTS
000006

START OF TEMPORARIES
000007

START OF INDIRECTS
000011

UNUSED COMPILER SPACE
057200

KK= 91526487325N= 0 NMAX= 100NRAND= 1TMAX 20000E+01
TO= 0.

1	.15197	.81644	.68958	.76019	1.13585	1.26342	1	3.15197	3.81644	.83210
2	.94347	1.52777	1.98831	.38987	.77459	1.64335	6	3.33525	4.20837	.69240
3	.37335	1.14051	3.32706	.17058	.71078	2.02732	17	3.31873	4.57429	.83475
4	.03213	.38081	.51056	-.19660	.23056	.79088	35	3.18027	4.24433	.85345
5	-.32233	.43342	1.74059	.43750	.77662	.98888	47	3.05455	4.44718	1.20653
6	.34744	.69755	.70438	.17091	.57945	.72456	46	2.96449	4.36811	1.12711
7	.49749	.57851	.47332	.39526	.73332	.72285	39	3.29107	4.36718	.86540
8	.69593	1.31733	1.92029	.54902	.60948	.52273	41	3.59174	4.95119	.94448
9	1.07977	.91737	1.61469	1.00205	.80433	1.70658	54	4.12249	5.25908	.78750
10	.81439	.40579	1.20142	.88399	.30179	2.49081	51	3.93483	4.86055	.96542
11	1.06340	.69751	1.85398	.83379	.57949	1.55351	50	4.11424	5.25627	1.00103
12	1.01469	.52375	2.08127	.21600	.43194	.25177	43	4.29514	5.20053	.80437
13	-.66558	-.02827	1.57507	-.75892	-.09814	2.72146	41	3.41356	4.74032	.91400
14	-.96607	-.60841	1.71211	-.96595	-.97113	1.80063	47	3.20642	4.23005	.88408
15	-1.08257	-1.02958	1.90486	-.97123	-1.45379	3.14467	53	3.08980	4.17160	.93784
16	-1.66255	-1.46085	4.46792	-1.75876	-1.50702	5.78900	50	2.39847	4.16454	.83599
17	-1.81325	-.99659	5.73059	-.80783	-.80188	1.49017	43	2.34398	4.67296	1.33625
18	-.79844	-.94413	1.39578	-.26593	-.75022	1.10686	45	2.35337	4.53072	1.36059
19	-.14036	-.29226	.13299	.04668	-.25616	.29565	37	2.47164	4.98868	1.33405
20	-.23043	-.32966	.21111	-.23805	-.37986	.29101	35	2.19452	4.91518	1.13350
21	-.37399	-.47697	.45226	.26622	2.56094	41.35426	39	2.05858	4.81807	0.00000

Additional Appendix C.

A Gauss-Hermite Program for Implementing
the Two-Dimensional Phase Demodulator

This program was written to test the Gauss-Hermite method. The numerical results of this program are given in Appendix B of Chapter VII.

The program was compiled on one of the Kirtland AFB CDC 6600's using the FTN compiler. The code-optimization algorithm was employed. The random number generator is identical to the one described in Chapter IV.

Programmer: C. Hecht

Preceding page blank

```
      PROGRAM MAIN(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)
      COMMON/NAME/T(20),WT(20),GAMA(9,9),HH(9,9),BINOM(17,17)
      COMMON/NAME6/NUM1,NO1,NO2
      COMMON/NAME8/A11,A22,Q22,R11,DELT,JSEED
      COMMON/NAME9/Y1EST,Y2EST
      COMMON /RN/ NZZZ(3), XNZ(2)
      COMMON /GN/ DZZZ1, JGAUSS, XZZZ(2)
      JGAUSS=J
  100 CONTINUE
      READ(5,651)Y1EST,Y2EST,P110,DELF,Q22C,NUM1,NO1,NO2,JSEED
      IF(EOF(5))102,101
  101 CONTINUE
      P110=P110**2
      QQ=Q22C**(.25)
      RR=(P110/(SQRT(2.0)*QQ))**(4.0/3.0)
      FTC=SQRT(2.0)* RR**(.25) /QQ
      DELT=DELF*FTC
      Q22=Q22C*DELT
      R11=RR/DELT
      P220=P110*SQRT(Q22C/RR)
      A11=    P110
      A22=    P220
      WRITE(6,652) P110,FTC,DELT,Q22,R11
      WRITE(6,657) A11,A22
  657 FORMAT(* *,30X,*VARIANCE OF INIT. POS. AND VEL.*,1P2E15.6)
      WRITE(6,653) Q22C,RR
  653 FORMAT(*0*,*CONTINUOUS DRIVING VARIANCE =*,1PE14.6/
     1* CONTINUOUS NOISE VARIANCE =*,1PE14.6)
  651 FORMAT(5E10.4,3I5,I10)
  652 FORMAT(*1*,*EQUILIBRIUM CONTINUOUS POSITION VARIANCE =*,1PE14.6/
     1 * FILTER TIME CONSTANT =*                            ,1PE14.6/
     1 * DELTA =*                                           ,1PE14.6/
     1 * DISCRETE DRIVING VARIANCE =*                       ,1PE14.6/
     1 * DISCRETE NOISE VARIANCE =*                         ,1PE14.6)
      CALL INITAL
      CALL CYCLE
      IF(NO2.LT.511) GO TO 100
  102 CONTINUE
      STOP
      END
```

SYMBOLIC REFERENCE MAP

ENTRY POINTS		DEF LINE	REFERENCES
4052	MAIN	1	

VARIABLES	SN	TYPE		RELOCATION								
0	A11	REAL		NAME8	REFS	4	24	DEFINED	21			
1	A22	REAL		NAME8	REFS	4	24	DEFINED	22			
312	BINOM	REAL	ARRAY	NAME	REFS	2						
4261	DELF	REAL			REFS	17	DEFINED	10				
4	DELT	REAL		NAME8	REFS	4	18	19	23	DEFINED	17	
0	DZZZ1	REAL		GN	REFS	7						
4265	ETC	REAL			REFS	17	23	DEFINED	16			
50	GAMA	REAL	ARRAY	NAME	REFS	2						
171	IN	REAL	ARRAY	NAME	REFS	2						
1	IGAUS,	INTEGER		GN	REFS	7	DEFINED	8				
3	ISEED	INTEGER		NAME8	REFS	4	DEFINED	10				
1	NO1	INTEGER		NAME6	REFS	3	DEFINED	10				
2	NO2	INTEGER		NAME6	REFS	3	37	DEFINED	10			
0	NUM1	INTEGER		NAME6	REFS	3	DEFINED	10				
6	NZZZ	INTEGER	ARRAY	RN	REFS	6						
4260	P110	REAL			REFS	13	15	20	21	22		
					DEFINED	10	13					
4266	P220	REAL			REFS	22	DEFINED	24				
4263	QQ	REAL			REFS	15	16	DEFINED	14			
2	Q2?	REAL		NAME8	REFS	4	23	DEFINED	18			
4262	Q220	REAL			REFS	14	18	20	26	DEFINED	18	
4264	RR	REAL			REFS	16	19	20	26	DEFINED	15	
3	R11	REAL		NAME8	REFS	4	23	DEFINED	19			
0	T	REAL	ARRAY	NAME	REFS	2						
24	WT	REAL	ARRAY	NAME	REFS	2						
3	XNZ	REAL	ARRAY	RN	REFS	6						
2	XZZZ	REAL	ARRAY	GN	REFS	7						
6	Y1EST	REAL		NAME9	REFS	5	DEFINED	10				
1	Y2EST	REAL		NAME9	REFS	5	DEFINED	10				

FILE NAMES	MODE				
0	INPUT				
2022	OUTPUT				
0	TAPE5	FMT	READS	10	
2022	TAPE6	FMT	WRITES	23	24 26

EXTERNALS	TYPE	ARGS		REFERENCES		
CYCLE		0		36		
EOF	REAL	1		11		
INITAL		0		35		
SQRT	REAL	1	LIBRARY	15	16	20

STATEMENT LABELS			DEF LINE	REFERENCES
4054	100		9	37
0	101	INACTIVE	12	11
4174	102		38	11
4222	651	FMT	29	10
4225	652	FMT	30	23

STATEMENT LABELS

			DEF LINE	REFERENCES
4210	653	FMT	27	26
4202	657	FMT	25	24

COMMON BLOCKS

	LENGTH	MEMBERS - BIAS	NAME	(LENGTH)
NAME	491	0	T	(20)
		121	HM	(81)
NAME6	3	1	NUM1	(1)
NAME8	6	0	A11	(1)
		3	R11	(1)
NAME9	2	0	Y1EST	(1)
RN	5	0	NZZZ	(3)
GN	4	1	DZZZ1	(1)
		20	NT	(20)
		202	BINOM	(289)
		1	NO1	(1)
		1	A22	(1)
		4	DELT	(1)
		1	Y2EST	(1)
		3	XNZ	(2)
		1	JGAUSS	(1)
		40	GAMA	(81)
		2	NO2	(1)
		2	Q22	(1)
		5	JSEED	(1)
		2	XZZZ	(2)

STATISTICS

PROGRAM LENGTH	2258	149
BUFFER LENGTH	40448	2084
COMMON LENGTH	7778	511

```
      SUBROUTINE INITAL
      COMMON/NAME/T(20),WT(20),GAMA(9,9),HM(9,9),BINOM(17,17)
      COMMON/NAME6/NUM1,NO1,NO2
      CALL HERMIT(NUM1,T,WT)
5     CALL COEF(NO1,HM)
      CALL CMBTRL(NO1,BINOM)
      CALL GAM22(NO1,GAMA)
      RETURN
      END
```

SYMBOLIC REFERENCE MAP

ENTRY POINTS		DEF LINE	REFERENCES
1	INITAL	1	8

VARIABLES	SN	TYPE	RELOCATION							
312	BINOM	REAL	ARRAY	NAME	REFS	2	6			
50	GAMA	REAL	ARRAY	NAME	REFS	2	7			
171	HM	REAL	ARRAY	NAME	REFS	2	5			
1	NO1	INTEGER		NAME6	REFS	3	5	6	7	
2	NO2	INTEGER		NAME6	REFS	3				
0	NUM1	INTEGER		NAME6	REFS	3	4			
0	T	REAL	ARRAY	NAME	REFS	2	4			
24	WT	REAL	ARRAY	NAME	REFS	2	4			

EXTERNALS	TYPE	ARGS	REFERENCES
CMBTRL		2	6
COEF		2	5
GAM22		2	7
HERMIT		3	4

COMMON BLOCKS	LENGTH	MEMBERS - BIAS	NAME	(LENGTH)							
NAME	491	0	T	(20)	20	WT	(20)	40	GAMA	(61)	
		121	HM	(61)	202	BINOM	(289)				
NAME6	3	0	NUM1	(1)	1	NO1	(1)	2	NO2	(1)	

STATISTICS

PROGRAM LENGTH	308	24
COMMON LENGTH	7566	494

```
      SUBROUTINE HERMIT (NO,T,WT)
      DIMENSION T(20),WT(20)
      DIMENSION  T10(10),WT10(10)
      DIMENSION                T10(5),WT10(5)
      DATA T10/.3423013272,1.0366108297,1.7566836492,2.5327316742,
     1 3.4361591188/
      DATA WT10/6.1086263373E-1,2.4013861108E-1,3.3874394455E-2,
     1  1.3436457467E-3,7.6404328552E-6/
      DATA T20/0.24534070,0.73747373,1.23407621,1.73853771,2.25497400,
     X 2.78880606,3.34785457,3.94476404,4.60368245,5.38748089/
      DATA WT20/4.62243669E-1,2.866755.5E-1,1.0901726GE-1,2.48105209E-2,
     X 3.24377334E-3,2.28338636E-4,7.8.255048E-6,1.08606937E-7,
     X 4.39934099E-10,2.22939365E-13/
      IF(NO.EQ.20) GO TO 120
  11  DO 22 I=1,5
      T(I)=-T10(6-I)
   22 WT(I)=WT10(6-I)
      DO 23 I=6,10
      T(I)=T10(I-5)
   23 WT(I)=WT10(I-5)
      GO TO 150
  12  DO 18 I=1,10
      T(I)=-T20(11-I)
   18 WT(I)=WT20(11-I)
      DO 19 I=11,20
      T(I)=T20(I-10)
   19 WT(I)=WT20(I-10)
  150 CONTINUE
      RETURN
      END
```

```
REGISTER ALLOCATION
  1 REGISTERS ASSIGNED OVER THE LOOP BEGINNING AT LINE     15
  1 REGISTERS ASSIGNED OVER THE LOOP BEGINNING AT LINE     22
```

SYMBOLIC REFERENCE MAP

ENTRY POINTS		DEF LINE	REFERENCES
2	HERMIT	1	29

VARIABLES	SN	TYPE		RELOCATION								
62	I	INTEGER			REFS	2*16	2*17	2*19	2*20	2*23	2*24	2*26
					2*27	DEFINED	15	18	22	25		
0	NO	INTEGER		F.P.	REFS	14	DEFINED	1				
0	T	REAL	ARRAY	F.P.	REFS	2	DEFINED	1	16	19	23	26
107	T10	REAL	ARRAY		REFS	4	16	19	DEFINED	5		
63	T20	REAL	ARRAY		REFS	3	23	26	DEFINED	9		
0	WT	REAL	ARRAY	F.P.	REFS	2	DEFINED	1	17	20	24	27
114	WT10	REAL	ARRAY		REFS	4	17	20	DEFINED	7		
75	WT20	REAL	ARRAY		REFS	3	24	27	DEFINED	11		

STATEMENT LABELS			DEF LINE	REFERENCES
0	18		24	22
0	19		27	25
0	22		17	15
0	23		20	18
0	110	INACTIVE	15	
37	120		22	14
57	150		28	21

LOOPS	LABEL		INDEX	FROM-TO	LENGTH	PROPERTIES
23	22	*	I	15 17	48	INSTACK
34	23		I	18 20	38	INSTACK
43	18	*	I	22 24	48	INSTACK
54	19		I	25 27	38	INSTACK

STATISTICS

PROGRAM LENGTH	137B	95

```
                      SUBROUTINE COEF(N1,HG)
                      DIMENSION L( · , 9),HG(9,9)
                      L(1,1) = 1
                      L(2,2)=2
     5                L(2,1)=0
                      L(3,3)=4
                      L(3,2)=0
                      L(3,1)=-2
                      NDEG = N1 - 1
    10                DO 50 N=3,NDEG
                      NN=N+1
                      L(NN,NN)=2*L(NN-1,NN-1)
                      L(NN,NN-1)=0
                      L(NN,1)=-2*(N-1)*L(NN-2,1)
    15                NM=N-1
                      DO50 K=2,NM
                      L(NN,NN-K)=2*L(NN-1,NN-1-K) - 2*(N-1)*L(NN-2,NN-K)
                   50 CONTINUE
                      DO 60 K=1,N1
    20                DO 60 N=1,N1
                      HG(K,N)=L(K,N)
                   60 CONTINUE
                      WRITE(6,64)
                      DO 63 I=1,N1
    25                WRITE(6,65)(HG(I,J),J=1,I)
                   63 CONTINUE
                   64 FORMAT(*0*,*    HERMITE COEFICIENTS*)
                   65 FORMAT(*0*,9F10.0)
                      RETURN
    30                END
```

```
SUMMARY OF CHANGES MADE BY THE OPTIMIZER
  52 WORDS OF INVARIANT RLIST REMOVED FROM THE LOOP STARTING AT LINE    16
```

SYMBOLIC REFERENCE MAP

ENTRY POINTS	DEF LINE	REFERENCES
2 COEF	1	29

VARIABLES	SN	TYPE	RELOCATION										
0	HG	REAL	ARRAY	F.P.	REFS	2	25	DEFINED	1	21			
145	I	INTEGER			REFS	2*25	DEFINED	24					
146	J	INTEGER			REFS	25	DEFINED	25					
144	K	INTEGER			REFS	3*17	2*21	DEFINED	16	19			
147	L	INTEGER	ARRAY		REFS	2	12	14	2*17	21			
					DEFINED	3	4	5	6	7	8	12	
						13	14	17					
141	N	INTEGER			REFS	11	14	15	17	2*21			
					DEFINED	10	20						
140	NDEG	INTEGER			REFS	10	DEFINED	9					
143	NM	INTEGER			REFS	16	DEFINED	15					
142	NM	INTEGER			REFS	4*12	2*13	2*14	6*17	DEFINED	11		
0	N1	INTEGER		F.P.	REFS	9	19	20	24	DEFINED	1		

FILE NAMES	MODE			
TAPE6	FMT	WRITES	23	25

STATEMENT LABELS		DEF LINE	REFERENCES	
3	50	18	10	16
0	60	22	19	20
0	63	26	24	
131	64 FMT	27	23	
135	65 FMT	28	25	

LOOPS	LABEL	INDEX	FROM-TO	LNGTH	PROPERTIES		
24	50	* N	10 16	446		NOT INNER	
51	50	* K	16 18	148	OPT		
71	60	* K	19 22	118		NOT INNER	
75	60	N	20 22	28	INSTACK		
106	63	* I	24 26	158		EXT REFS	NOT INNER
111		* J	25	78		EXT REFS	

STATISTICS

PROGRAM LENGTH 2778 191

```
SUBROUTINE  CMBTRL                                    CDC 6600 FTN V3.0-P213 OPT=2  03/19/72  00.54.44.      PAGE      1

                  SUBROUTINE CMBTRL(K,A)
                  DIMENSION A(17,17)
                  KN=K-1
                  DO 20 N=1,K
                  A(N,1)=1.0
                  A(N,N)=1.0
               20 CONTINUE
                  DO 21   N=2,KN
                  DO 21 M=2,N
                  A(N+1,M) = A(N,M) + A(N,M-1)
               21 CONTINUE
                  WRITE (6,51)
                  DO 23 I=1,K
                  WRITE(6,50) (A(I,J), J=1,I)
               23 CONTINUE
               50 FORMAT(* *,12F9.0)
               51 FORMAT(*1*,5X,*BINOMIAL COEFFICIENTS*/)
                  RETURN
                  END

REGISTER ALLOCATION
  1 REGISTERS ASSIGNED OVER THE LOOP BEGINNING AT LINE     4
```

```
      SUBROUTINE  CMBTRL                                  CDC 6600 FTN V3.0-P213 OPT=2  03/19/72  00.54.44.      PAGE      2

      SYMBOLIC REFERENCE MAP

ENTRY POINTS      DEF LINE      REFERENCES
    2  CMBTRL          1          18

VARIABLES      SN  TYPE            RELOCATION
    0  A           REAL        ARRAY      F.P.        REFS          2        2*10        14    DEFINED        1          5          6
                                                        10
   76  I           INTEGER                            REFS       2*14    DEFINED        13
   77  J           INTEGER                            REFS         14    DEFINED        14
    0  K           INTEGER                F.P.        REFS          3          4        13    DEFINED        1
   73  KN          INTEGER                            REFS          8    DEFINED         3
   75  M           INTEGER                            REFS       3*10    DEFINED         9
   74  N           INTEGER                            REFS          5        2*6         9       3*10    DEFINED        4          8

FILE NAMES         MODE
      TAPES         FMT                    WRITES       12         14

STATEMENT LABELS                DEF LINE    REFERENCES
    0  20                           7          4
    0  21                          11          8          9
    0  23                          15         13
   62  50       FMT                16         14
   65  51       FMT                17         12

LOOPS  LABEL     INDEX      FROM-TO     LENGTH     PROPERTIES
   22  20         N            4 7          2B      INSTACK
   25  21       * N            8 11        11B                NOT INNER
   31  21         M            9 11         2B      INSTACK
   42  23       * I           13 15        16B                EXT REFS  NOT INNER
   45           * J           14            7B                EXT REFS

STATISTICS
  PROGRAM LENGTH     112B       74
```

```
SUBROUTINE  GAM22                                    CDC 6600 FTN V3.0-P213 OPT=2  03/19/72  00.54.44.      PAGE      1

                  SUBROUTINE GAM22(N,AAMA)
                  DIMENSION            AI( 9),FCTRL( 9),AAMA(9,9)
                  DO 153 I=2,N
                  AI(1)=1.0
                  DO 152 K=2,I
                  AK=K
                  AI(K)=AI(K-1)*AK
              152 CONTINUE
                  FCTRL(I)=AI(I)
              153 CONTINUE
                  FCTRL(1)=1.0
                  DO 81 I=2,N
                  DO 81 J=2,N
                  AAMA(I,J)=2.0**(I-1)*2.0**(J-1)*FCTRL(I-1)*FCTRL(J-1)
                  AAMA(I,J)=1.0/AAMA(I,J)
               81 CONTINUE
                  AAMA(1,1)=1.0
                  DO 82 K=2,N
                  AAMA(K,1)=2.0**(K-1)*FCTRL(K-1)
                  AAMA(K,1)=1.0/AAMA(K,1)
                  AAMA(1,K)=AAMA(K,1)
               82 CONTINUE
                  WRITE(6,50)
               50 FORMAT (*0*,5X,*NORMALIZING COEFFICIENTS*/)
                  WRITE(6,51)((AAMA(I,J),I=1,N ),J=1,N )
               51 FORMAT(*0*,1P9E14.6)
                  RETURN
                  END

SUMMARY OF CHANGES MADE BY THE OPTIMIZER
  13 WORDS OF INVARIANT RLIST REMOVED FROM THE LOOP STARTING AT LINE     13

REGISTER ALLOCATION
  2 REGISTERS ASSIGNED OVER THE LOOP BEGINNING AT LINE      5
```

SYMBOLIC REFERENCE MAP

ENTRY POINTS		DEF LINE	REFERENCES
2	GAM22	1	27

VARIABLES	SN	TYPE	RELOCATION								
0	AAMA	REAL	ARRAY F.P.	REFS	2	15	20	21	23		
				DEFINED	1	14	16	17	19	20	21
135	AI	REAL	ARRAY	REFS	2	7	9	DEFINED	4	7	
133	AK	REAL		REFS		DEFINED	6				
146	FCTRL	REAL	ARRAY	REFS	1	2*14	19	DEFINED	9	11	
131	I	INTEGER		REFS	5	2*9	3*14	2*15	25		
				DEFINED	3	12	25				
134	J	INTEGER		REFS	3*14	2*15	25	DEFINED	13	25	
132	K	INTEGER		REFS	6	2*7	3*19	2*20	2*21		
				DEFINED	5	18					
0	N	INTEGER	F.P.	REFS	3	12	13	18	2*25		
				DEFINED	1						

FILE NAMES	MODE			
TAPE6	FMT	WRITES	23	25

STATEMENT LABELS			DEF LINE	REFERENCES	
117	50	FMT	24	23	
124	51	FMT	26	25	
0	81		16	12	13
0	82		22	18	
0	152		8	5	
0	153		10	3	

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES		
15	153	* I	3 10	16B		NOT INNER	
23	152	* K	5 8	3B	INSTACK		
36	81	* I	12 16	25B		EXT REFS	NOT INNER
41	81	* J	13 16	20B		EXT REFS	
66	82	* K	18 22	13B		EXT REFS	
104		* J	25	6B		EXT REFS	

STATISTICS

PROGRAM LENGTH	1768	126

```
      SUBROUTINE EIGEN(B,FKS)
      DIMENSION B(3),FKS(4),A(2,2),D(2,2),T(2,2)
      PI=3.14159266
      P2=PI*0.5
      IF(ABS(B(2))._T.1.0E-7) GO TO 15
      IF(ABS(B(1)-B(3)).LT.1.0E-7) GO TO 16
      PHI=0.5*ATAN(2.0*B(2)/(B(1)-B(3)))
      GO TO 20
   15 PHI=0.0
      GO TO 20
   16 PHI=0.5*SIGN(P2,B(2))
   20 CONTINUE
      A(1,1)=B(1)
      A(2,1)=B(2)
      A(1,2)=B(2)
      A(2,2)=B(3)
      T(1,1)=COS(PHI)
      T(2,1)=SIN(PHI)
      T(1,2)=-T(2,1)
      T(2,2)=T(1,1)
      FKS(1)=T(1,1)
      FKS(2)=T(2,1)
      FKS(3)=T(1,2)
      FKS(4)=T(2,2)
      DO 25 I=1,2
      DO 25 J=1,2
      D(I,J)=0.0
      DO 25 K=1,2
      D(I,J)=A(I,K)*T(K,J) + D(I,J)
   25 CONTINUE
      DI=T(1,2)
      T(1,2)=T(2,1)
      T(2,1)=DI
      DO 35 I=1,2
      DO 35 J=1,2
      A(I,J)=0.0
      DO 35 K=1,2
      A(I,J)=T(I,K)*D(K,J) + A(I,J)
   35 CONTINUE
      B(1)=A(1,1)
      B(3)=A(2,2)
      RETURN
      END
```

SYMBOLIC REFERENCE MAP

ENTRY POINTS DEF LINE REFERENCES
2 EIGEN 1 42

VARIABLES	SN	TYPE	RELOCATION								
144	A	REAL	ARRAY		REFS	2	29	38	40	41		
					DEFINED	13	14	15	16	36	38	
0	B	REAL	ARRAY	F.P.	REFS	2	5	2*6	3*7	11	13	14
						15	16	DEFINED	1	40	41	
150	D	REAL	ARRAY		REFS	2	29	38	DEFINED	27	29	
143	D1	REAL			REFS	33	DEFINED	31				
0	FKS	REAL	ARRAY	F.P.	REFS	2	DEFINED	1	21	22	23	24
140	I	INTEGER			REFS	27	3*29	36	3*38	DEFINED	25	34
141	J	INTEGER			REFS	27	3*29	36	3*38	DEFINED	26	35
142	K	INTEGER			REFS	2*29	2*38	DEFINED	28	37		
137	PHI	REAL			REFS	17	18	DEFINED	7	9	11	
135	PI	REAL			REFS	4	DEFINED	3				
136	P2	REAL			REFS	11	DEFINED	4				
154	T	REAL	ARRAY		REFS	2	19	20	21	22	23	24
						29	31	32	38	DEFINED	17	18	19
						20	32	33				

EXTERNALS	TYPE	ARGS	REFERENCES
ATAN	REAL	1 LIBRARY	7
COS	REAL	1 LIBRARY	17
SIN	REAL	1 LIBRARY	18

INLINE FUNCTIONS	TYPE	ARGS		DEF LINE	REFERENCES	
ABS	REAL	1	INTRIN		5	6
SIGN	REAL	2	INTRIN		11	

STATEMENT LABELS	DEF LINE	REFERENCES		
33 15	9	5		
35 16	11	6		
41 20	12	8	10	
0 25	30	25	26	28
0 35	39	34	35	37

LOOPS	LABEL		INDEX	FROM-TO	LENGTH	PROPERTIES
60	25	*	I	25 33	208	NOT INNER
61	25	*	J	26 30	158	NOT INNER
70	25		K	28 30	38	INSTACK
104	35	*	I	34 39	208	NOT INNER
105	35	*	J	35 39	158	NOT INNER
114	35		K	37 39	38	INSTACK

STATISTICS
PROGRAM LENGTH 2048 132

```
      SUBROUTINE CYCLE
      DIMENSION              BTA( 9),                    AR( 9),BL( 9),
     1 Y1(20,20),Y2(20,20), XKS(3),EKS(4),
     2 TAURK(20,20),TAUKL(20,20),HRK(8),HKL(16),
     3 BRL(9,9),Y1P(20),Y2P(20),
     4                        CENMOM( 9,9 ),ALPHA1( 9),ALPHA2( 9),
     5 DENS(20,20),HT2(20,20), HHH(20,20) ,BIN2(9,9),CENM(17,17)
      DIMENSION TT1PH(9)    ,TT2PH(9)
      DIMENSION                FKS(4),GKS(4)
      DIMENSION AA11(9),AA21(9),AA12(9),AA22(9)
      REAL K1,K2,M1,M2,MOM(3,3)
      REAL M18AR,M23AR
      DIMENSION XDAT(230,3)
      COMMON/NAME7/T(20),HT(20),GAMA(9,9),HH(9,9),BINOM(17,17)
      COMMON/NAME6/NUM1,NO1,NO2
      COMMON/NAME8/A11,A22,Q22,R11,DELT,JSEED
      COMMON/NAME9/XONE,XTWO
      WRITE (6,652)
      WRITE(6,653) A11,A22,Q22,R11,DELT,NUM1,NO1
      PI=3.1415926536
      PI2=2.0*PI
      NOM=NO1-1
      NSAMP=0
      SUMP1=0.0
    2 CONTINUE
      KOUNT = 1
      DELSQ = DELT**2
      CENMOM(1,1)=1.0
      CENMOM(1,2)=0.0
      CENMOM(2,1)=0.0
      CENMOM(2,2)=0.0
      CENM(1,1)=1.0
      CENM(2,1)=0.0
      CENM(1,2)=0.0
      AA11(1)=1.0
      AA21(1)=1.0
      AA12(1)=1.0
      AA22(1)=1.0
      BTA(1) = 1.0
      AR(1) = 1.0
      BL(1) = 1.0
      ALPHA1(1)=1.0
      ALPHA2(1)=1.0
      TT1PH(1)  =1.0
      TT2PH(1)  =1.0
          DO 30 I=1,NO1
          DO 30 J=1,NO1
   30 BRL(I,J)=0.0
      DO 35 I=1,NUM1
      DO 35 J=1,NUM1
   35 HT2(I,J)=HT(I)*HT(J)
      DEV1=SQRT(A11)
      DEV2=SQRT(A22)
      DEV3=SQRT(R11)
      Y1ST=XONE
```

```
      Y2EST=XTWO
      CALL GAUSS(JSEED,DEV1,Y1EST,X1)
      CALL GAUSS(JSEED,DEV2,Y2EST,X2)
      XDAT(KOUNT,1)=X1
60    DEV2=SQRT(Q22)
      COSX1=COS(X1)
      SINX1=SIN(X1)
      CALL GAUSS(JSEED,DEV3,COSX1,Z1)
      CALL GAUSS(JSEED,DEV3,SINX1,Z2)
65    C ********  INITIALIZE PARAMETERS  **********************************
      DO 41 I=1,NO1
      DO 41 J=1,NO1
   41 BRL(I,J)=0.0
      BRL(1,1)=1.0
70    C ********  COMPUTE INITIAL MEAN AND VARIANCE BY LINEARIZING  ******
      K1=1.0/A11 + 1.0/R11
      K1=1.0/K1
      K2=A22
      XKS(1)=K1
75    XKS(2)=0.
      XKS(3)=K2
      CALL EIGEN(XKS,EKS)
      DEVK1=SQRT(2.0*XKS(1))
      DEVK2=SQRT(2.0*XKS(3))
80    DO 51 I=1,NUM1
       Y1P(I)  =     DEVK1*T(I)
       Y2P(I)  =     DEVK2*T(I)
   51 CONTINUE
      COSY1=COS(Y1EST)
85    SINY1=SIN(Y1EST)
      BZ1=(Z1*SINY1-Z2*COSY1-Y1EST)/R11
      M1= -BZ1*K1 + K1/A11*Y1EST
      M2=Y2EST
      DO 56 I=1,NUM1
90    DO 56 J=1,NUM1
      MMM(I,J)=1.0
      Y1(I,J)=EKS(1)*Y1P(I)    + EKS(3)*Y2P(J)    + M1
      Y2(I,J) = EKS(2)*Y1P(I)    + EKS(4)*Y2P(J)    + M2
   56 CONTINUE
95    GO TO 425
C
C     ******  MAIN RECYCLE POINT  ***************************************
C
   70 CONTINUE
100   TT1=-1.0/(EKS(1)*DELT - EKS(2))
      TT1SQ=TT1**2
      TT2= EKS(1)*TT1
      TT2SQ=TT2**2
      TT3=1.0/(EKS(4)-EKS(3)*DELT)
105   TT3SQ=TT3**2
      TT4=EKS(3)*TT3
      AA=Q22 + TT1SQ*S11
      BB=TT1SQ*S11/AA
      BETA= -Q22/(S11*TT1SQ)
110   CSQ= TT3SQ*S22 + S11*TT1SQ*(1.0 - BB)
```

```
      CC = SQRT(CSQ)
      BSQCSQ=BB**2/CSQ
      V4 = TT1*V1BAR
      V5 =  -BETA*TT1*V1BAR - TT3/BB*V2BAR
115   V6=TT4/BB + BETA*TT2
      AR(2)= SQRT(BB)
      AR(3)=BB
      C3=1.0/(S22*Q22*TT3SQ + TT1SQ*TT3SQ*S11*S22 + S11*Q22*TT1SQ)
      BL(3)=S22*TT3SQ*C3*AA
120   BL(2)= SQRT(BL(3))
      IF(TT1.LT.0.0)AR(2)=-AR(2)
      IF(TT3.LT.0.0)BL(2)=-BL(2)
      BTA(3) = Q22**2*C3
      BTA(2)=-SQRT(BTA(3))
125   DO 120 K=4,NO1
      BTA(K) = BTA(K - 1)*BTA(2)
      AR(K) = AR(K - 1)*AR(2)
      BL(K) = BL(K - 1)*BL(2)
  120 CONTINUE
130   A2RT = 1.0/SQRT(2.0*AA)
      BCRT = BB/(SQRT(2.0)*CC)
C
      X1=X1 + X2*DELT
      XDAT(KOUNT,1)=X1
135   CALL GAUSS(JSEED,DEV2,X2,X2)
      COSX1=COS(X1)
      SINX1=SIN(X1)
      CALL GAUSS(JSEED,DEV3,COSX1,Z1)
      CALL GAUSS(JSEED,DEV3,SINX1,Z2)
140   Y1EST=ETA1
      COSY1=COS(Y1EST)
      SINY1=SIN(Y1EST)
      BZ1=(Z1*SINY1-Z2*COSY1-Y1EST)/R11
      C11 = +TT2/AA + BSQCSQ*V6
145   C22 = 1.0/AA + BSQCSQ
      C33 = 1.0/R11 + TT2SQ/AA + BSQCSQ*V6**2
      C44= BZ1                                    - V4*TT2/AA+ BSQCSQ*V5*V6
      C55 = -V4/AA       + BSQCSQ*V5
      K1=1.0/C33
150   K2 = 1.0/C22
      RTK1 = SQRT(K1)
      RTK2 = SQRT(K2)
      RHO = -C11*RTK1*RTK2
      RHOIN = 1.0/(1.0 - RHO**2)
155   XKS(1)=K1*RHOIN
      XKS(3)=K2*RHOIN
      XKS(2) = SQRT(XKS(1)*XKS(3))*RHO
      M1=-RHOIN*(RTK1*RTK2*RHO*C55    +    K1*C44)
      M2 =-RHOIN*(RTK1*RTK2*RHO*C44 + K2*C55)
160   CALL EIGEN(XKS,EKS)
C  ********  GENERATE NEW GRID  ****************************************
C
      DEVK1=SQRT(2.0*XKS(1))
      DEVK2=SQRT(2.0*XKS(3))
165   200 DO 211 I=1,NUM1
```

```
      Y1P(I)    = DEVK1*T(I)
      Y2P(I)    = DEVK2*T(I)
  211 CONTINUE
      DO 212 I=1,NUM1
      DO 212 J=1,NUM1
      Y1(I,J)=EKS(1)*Y1P(I)    + EKS(3)*Y2P(J)    + M1
      Y2(I,J) = EKS(2)*Y1P(I)    + EKS(4)*Y2P(J)    + M2
  212 CONTINUE
      DO 215 I=1,NUM1
      DO 215 J=1,NUM1
      TAURK(I,J) = -(TT2*Y1(I,J) - TT1*V1BAR)
      TAUKL(I,J) = BCRT*(Y2(I,J) - BETA*TAURK(I,J) + (TT4*Y1(I,J) -
     1 TT3*V2BAR)/BB)
      TAURK(I,J) = A2RT*(Y2(I,J) - TAURK(I,J))
  215 CONTINUE
      DO 301 I=1,NUM1
      DO 301 J=1,NUM1
      HHH(I,J) = 0.0
  301 CONTINUE
C
C  ****COMPUTATION OF POLYNOMIAL FUNCTION  ******************************
C
      DO 251 IR=1,NO1
      DO 251 K=1,IR
  251 BIN2(IR,K)=BTA(K)*BINOM(IR,K)
      DO 255 L=1,NO1
      ITT=NO1+1-L
      DO 255 IR=1,ITT
  255 BRL(IR,L)=BRL(IR,L)*AR(IR)*BL(L)
      DO 311 I=1,NU41
      DO 311 J=1,NUM1
      HRK(1     ) = 2.0*TAURK(I,J)
      HKL(1     ) = 2.0*TAUKL(I,J)
      HRK(2     )=4.0*TAURK(I,J)**2 - 2.0
      HKL(2     )=4.0*TAUKL(I,J)**2 - 2.0
      DO 226 K=3,NOM
      XK = K -1
      HRK(K     ) = 2.0*TAURK(I,J)*HRK(K-1)      - 2.0*XK*HRK(K-2)
      HKL(K     ) = 2.0*TAUKL(I,J)*HKL(K-1)      - 2.0*XK*HKL(K-2)
  226 CONTINUE
      DO 310 L=1,NO1
      ITT=NO1+1-L
      DO 310 IR=1,ITT
          IF(BRL(IR,L).EQ.0.0) GO TO 310
      HHK=0.0
      DO 308 K=1,IR
      IN1= IR - K
      IN2 = K + L -2
      IF(IN1.EQ.0.AND.IN2.EQ.0) GO TO 304
      IF(IN1.EQ.0) GO TO 305
      IF(IN2.EQ.0) GO TO 306
      HH1 = HRK(IN1)    *HKL(IN2)
      HH1=HH1*BIN2(IR,K)
      GO TO 307
  304 HH1=1.0
```

```
      GO TO 307
  305 HH1=HKL(IN2)     *BIN2(IR,K)
      GO TO 307
  306 HH1=HRK(IN1)     *BIN2(IR,K)
  307 CONTINUE
      HHK = HH1 + HHK
  308 CONTINUE
      HH2 = HHK*BRL(IR,L)
      HHH(I,J) = HHH(I,J) + HH2
  310 CONTINUE
      IF(HHH(I,J).LT.0.0)  HHH(I,J)=0.
  311 CONTINUE
  425 CONTINUE
C
C     ***  COMPUTATION OF EXPONENTIAL FUNCTION  ***************************
C
      SUMDEN = 0.0
      Z1R=Z1/R11
      Z2R=Z2/R11
      ZSQ=.5/R11
      Y1ES2=Y1EST**2
      Z1RS=Z1R*(COSY1+Y1EST*SINY1)+Z2R*(SINY1-Y1EST*COSY1)
      DO 411 I=1,NUM1
      DO 411 J=1,NUM1
      EXPON=EXP(Z1R*COS(Y1(I,J)) + Z2R*SIN(Y1(I,J))
     1 +BZ1*Y1(I,J) +ZSQ*(Y1(I,J)**2+Y1ES2) - Z1RS )
      DENS(I,J) = EXPON*HHH(I,J)
      SUMDEN = DENS(I,J)*WT2(I,J)     + SUMDEN
  411 CONTINUE
      SUMDI=1.0/SUMDEN
      DO 421 I=1,NUM1
      DO 421 J=1,NUM1
      DENS(I,J) = DENS(I,J)*SUMDI
  421 CONTINUE
      DO 426 I=1,NUM1
      DO 426 J=1,NUM1
      DENS(I,J) = DENS(I,J)*WT2(I,J)
  426 CONTINUE
C     *********      COMPUTE MEANS   ************************
C     ****  RAISE GRID POINTS TO POWERS   *********
      DO 477 J=1,NUM1
      TAURK(1,J)=1.0
      TAUKL(1,J)=1.0
      TAURK(2,J)=Y1P(J)
      TAUKL(2,J)=Y2P(J)
  477 CONTINUE
C**********           COMPUTE MOMENTS          **********
      NK1=NO1
      DO 478 K=3,NK1
      DO 478 J=1,NUM1
      TAURK(K,J)=TAURK(K-1,J)*TAURK(2,J)
      TAUKL(K,J)=TAUKL(K-1,J)*TAUKL(2,J)
  478 CONTINUE
      DO 491 MM=1,NK1
      ITT=NK1+1-MM
```

```
      DO 491 NN=1,ITT
      CENMOM(MM,NN)=0.0
      DO 491 I=1,NU41
      DO 491 J=1,NU41
280   CENMOM(MM,NN)=TAURK(MM,I)*TAUKL(NN,J)*DENS(I,J) +CENMOM(MM,NN)
  491 CONTINUE
      MOM(2,1)=CENMOM(2,1)
      MOM(1,2)=CENMOM(1,2)
      M18AR=MOM(2,1)*EKS(1)+MOM(1,2)*EKS(3)
285   M28AR=MOM(2,1)*EKS(2)+MOM(1,2)*EKS(4)
      ETA1=M1+M18AR
      ETA2=M2+M28AR
      XDAT(KOUNT,2)=ETA1
C**********          MOMENTS FROM CENTRAL MOMENTS          **********
290   TT1PH(2)  =-CENMOM(2,1)
      TT2PH(2)  =-CENMOM(1,2)
      DO 503 K=3,NK1
      TT1PH(K)  =-CENMOM(2,1)*TT1PH(K-1)
      TT2PH(K)  =-CENMOM(1,2)*TT2PH(K-1)
295   IF(ABS(TT1PH(K  )).LT.1.0E-40) TT1PH(K  )=0.0
      IF(ABS(TT2PH(K  )).LT.1.0E-40) TT2PH(K  )=0.0
  503 CONTINUE
      DO 461 M=1,NK1
      ITT=NK1+1-M
300   DO 461 N=1,ITT
      CENM(M,N)=0.0
      DO 461 I=1,M
      DO 461 J=1,N
      CENM(M,N)=BINOM(M,I)*BINOM (N,J)* TT1PH(M-I+1  )*
305  1 TT2PH(N-J+1  )          *CENMOM(I,J) + CENM(M,N)
  461 CONTINUE
C  *****    COMPUTE COVARIANCE MATRIX   ********************
      XKS(1)=CENM(3,1)
      XKS(2)=CENM(2,2)
310   XKS(3)=CENM(1,3)
                  CALL EIGEN(XKS,FKS)
      GKS(1)=EKS(1)*FKS(1) + EKS(3)*FKS(2)
      GKS(2)=EKS(2)*FKS(1) + EKS(4)*FKS(2)
      GKS(3)=EKS(1)*FKS(3) + EKS(3)*FKS(4)
315   GKS(4)=EKS(2)*FKS(3) + EKS(4)*FKS(4)
      DO 473 I=1,4
  473 EKS(I)=GKS(I)
      V18AR=ETA1*EKS(1)+ETA2*EKS(2)
      V28AR=ETA1*EKS(3)+ETA2*EKS(4)
320   M1=ETA1
      M2=ETA2
      S11=XKS(1)
      S22=XKS(3)
      P11 = EKS(1)**2*S11 + EKS(3)**2*S22
325   P22 = EKS(2)**2*S11 + EKS(4)**2*S22
      P12 = EKS(1)*EKS(2)*S11 + EKS(3)*EKS(4)*S22
      WRITE (6,622) KOUNT
      WRITE(6,621)Z1,X1,ETA1,P11,P12,S11,EKS(1),EKS(3)
      WRITE(6,617)
330   WRITE (6,621) Z2,X2,ETA2,P12,P22,S22,EKS(2),EKS(4)
```

```
C**********          COORDINAT ROTATION OF MOMENTS      **********
      AA11(2)=FKS(1)
      AA21(2)=FKS(3)
      AA12(2)=FKS(2)
      AA22(2)=FKS(4)
      DO 492 J=3,NO1
      AA11(J)=AA11(J-1)*FKS(1)
      AA21(J)=AA21(J-1)*FKS(3)
      AA12(J)=AA12(J-1)*FKS(2)
      AA22(J)=AA22(J-1)*FKS(4)
      IF(ABS(AA11(J)).LT.1.0E-40)AA11(J)=0.0
      IF(ABS(AA21(J)).LT.1.0E-40)AA21(J)=0.0
      IF(ABS(AA12(J)).LT.1.0E-40)AA12(J)=0.0
      IF(ABS(AA22(J)).LT.1.0E-40)AA22(J)=0.0
  492 CONTINUE
  500 CONTINUE
      CENMOM(1,1)=1.0
      CENMOM(2,1)=0.0
      CENMOM(1,2)=0.0
      CENMOM(2,2)=0.0
      CENMOM(3,1)=XKS(1)
      CENMOM(1,3)=XKS(3)
      IF(NO1.EQ.3) GO TO 496
      DO 495 MM=1,NO1
      ISS=5-MM
      IF(ISS.LT.1)ISS=1
      ITT=NO1+1-MM
      DO 495 NN=ISS,ITT
      CENMOM(MM,NN)=0.0
      J1=MM+NN
      DO 495 IR=1,MM
      DO 495 IS=1,NN
      J2=IR+IS
      CENMOM(MM,NN)=BINOM(MM,IR)*BINOM(NN,IS)*AA11(MM-IR+1)*AA12(IR)*
     1 AA21(NN+1-IS)*AA22(IS)*CENM(J1-J2+1,J2-1) + CENMOM(MM,NN)
  495 CONTINUE
  496 CONTINUE
      ALPHA1(2)=1.0/SQRT(2.0*S11)
      ALPHA2(2)=1.0/SQRT(2.0*S22)
      DO 505 I=3,NO1
      ALPHA1(I)=ALPHA1(I-1)*ALPHA1(2)
      ALPHA2(I)=ALPHA2(I-1)*ALPHA2(2)
  505 CONTINUE
      ALP=ALPHA1(2)*ALPHA2(2)/3.1415926
C
C  ****  COMPUTATION OF COEFFICIENTS FOR EXPANSION  ******************
C
      BRL(1,1)=ALP
      DO 511 IR=1,NO1
      ISS=5-IR
      IF(ISS.LT.1) ISS=1
      ITT=NO1+1-IR
      DO 511 L=ISS,ITT
      DX=0.0
      DO 510 I=1,IR
```

```
                      DO 510 J=1,L
                      IF(HH(IR,I).EQ.0.0) GO TO 510
                      IF(HH( L,J).EQ.0.0) GO TO 510
                      DX=DX + HH(IR,I)*HH(L,J)*ALPHA1(I)*ALPHA2(J)*GENMOM(I,J)
  390              510 CONTINUE
                      BRL(IR,L)=ALP*GAMA(IR,L)*DX
                   511 CONTINUE
                      IF(KOUNT.EQ.NO2) GO TO 550
                      KOUNT = KOUNT + 1
  395                 GO TO 70
                   652 FORMAT (*1*,5X,*A11*,11X,*A22*,11X,*Q22*,10X,*R11*,11X,*DELT*,8X,
                     1 *NO. OF INTG. PTS.*,2X,*NO. OF TERMS* )
                   653 FORMAT(* *,1P5E14.6,2I14 /)
                   617 FORMAT(* *,12X,*Z2*, 9X,*X2*,11X,*ETA2*,14X,*P12*,11X,*P22* )
  400              621 FORMAT(* *,5X,1P3E14.6, 4X,1P2E14.6,4X,1PE14.6,4X,1P2E14.6)
                   622 FORMAT(*0*,*TIME =*,I3,3X, *Z1*  9X,*X1*,11X,*ETA1*,14X,*P11*,11X,
                     1 *P12*,
                     1 14X,*EIGENVALUES*,11X,*EIGENVECTORS*)
                   550 CONTINUE
  405                 SUMP=0.0
                      DO 1501 I=31,NO2
                      XD=ABS(XDAT(I,1)-XDAT(I,2))
                  1498 CONTINUE
                      IF(XD.GT.PI ) GO TO 1499
  410                 GO TO 1500
                  1499 XD=XD-PI2
                      GO TO 1498
                  1500 CONTINUE
                      SUMP=(XD)**2 + SUMP
  415             1501 CONTINUE
                      X=NO2-30
                      SUMP=SUMP/X
                      NNS=NSAMP +1
                      WRITE(6,1511)NNS, SUMP
  420             1511 FORMAT(*0*,*SAMPLE PATH*,I4,* STATISTICAL VARIANCE =*,1PE13.6)
                      XNSAMP=NSAMP
                      XAA=XNSAMP+1.0
                      SUMP1=(SUMP + XNSAMP*SUMP1)/XAA
                      DBSUM=ALOG10(SUMP1)*10.0
  425                 WRITE(6,1521)NNS,SUMP1,DBSUM
                  1521 FORMAT(* *,*CUMULATIVE AVERAGE FOR*,I4,* SAMPLE PATHS.  STATISTICA
                     1L VARIANCE =*,1PE13.6,4X,* = *,1PE13.6,*DB*)
                      NSAMP=NSAMP+1
                      IF(NSAMP.EQ.200) GO TO 1750
  430                 GO TO 2
                  1750 CONTINUE
                      RETURN
                      END

SUMMARY OF CHANGES MADE BY THE OPTIMIZER
  28 WORDS OF INVARIANT RLIST REMOVED FROM THE LOOP STARTING AT LINE   175
  19 WORDS OF INVARIANT RLIST REMOVED FROM THE LOOP STARTING AT LINE   303
  32 WORDS OF INVARIANT RLIST REMOVED FROM THE LOOP STARTING AT LINE   362

REGISTER ALLOCATION
  2 REGISTERS ASSIGNED OVER THE LOOP BEGINNING AT LINE    80
  2 REGISTERS ASSIGNED OVER THE LOOP BEGINNING AT LINE    90
```

```
3 REGISTERS ASSIGNED OVER THE LOOP BEGINNING AT LINE   125
2 REGISTERS ASSIGNED OVER THE LOOP BEGINNING AT LINE   155
3 REGISTERS ASSIGNED OVER THE LOOP BEGINNING AT LINE   170
3 REGISTERS ASSIGNED OVER THE LOOP BEGINNING AT LINE   175
3 REGISTERS ASSIGNED OVER THE LOOP BEGINNING AT LINE   251
1 REGISTERS ASSIGNED OVER THE LOOP BEGINNING AT LINE   252
1 REGISTERS ASSIGNED OVER THE LOOP BEGINNING AT LINE   261
1 REGISTERS ASSIGNED OVER THE LOOP BEGINNING AT LINE   303
2 REGISTERS ASSIGNED OVER THE LOOP BEGINNING AT LINE   362
2 REGISTERS ASSIGNED OVER THE LOOP BEGINNING AT LINE   370
```

SYMBOLIC REFERENCE MAP

ENTRY POINTS DEF LINE REFERENCES
1 CYCLE 1 432

VARIABLES SN TYPE RELOCATION

2101	AA	REAL			REFS	108	119	130	144	145	146	147
						148	DEFINED	107				
11025	AA11	REAL	ARRAY		REFS	10	337	341	364	DEFINED	35	332
						337	341					
11047	AA12	REAL	ARRAY		REFS	10	339	343	364	DEFINED	37	334
						339	343					
11036	AA21	REAL	ARRAY		REFS	10	338	342	364	DEFINED	36	333
						338	342					
11060	AA22	REAL	ARRAY		REFS	10	340	344	364	DEFINED	38	335
						340	344					
2172	ALP	REAL			REFS	378	391	DEFINED	374			
5707	ALPHA1	REAL	ARRAY		REFS	2	2*371	374	389	DEFINED	42	368
						371						
5720	ALPHA2	REAL	ARRAY		REFS	2	2*372	374	389	DEFINED	43	369
						372						
2214	AR	REAL	ARRAY		REFS	2	121	2*127	194	DEFINED	40	116
						117	121	127				
0	A11	REAL		NAME8	REFS	16	19	52	71	87		
2120	A2RT	REAL			REFS	179	DEFINED	130				
1	A22	REAL		NAME8	REFS	16	19	53	73			
2103	B8	REAL			REFS	110	112	114	115	116	117	131
						177	DEFINED	108				
2121	BCRT	REAL			REFS	177	DEFINED	131				
2104	BETA	REAL			REFS	114	115	177	DEFINED	109		
312	BINOM	REAL	ARRAY	NAME	REFS	14	190	2*304	2*364			
10211	BIN2	REAL	ARRAY		REFS	2	218	222	224	DEFINED	190	
2225	BL	REAL	ARRAY		REFS	2	120	122	2*128	194		
					DEFINED	41	119	120	122	128		
5375	BRL	REAL	ARRAY		REFS	2	194	209	228	DEFINED	48	68
						69	194	378	391			
2110	BSQCSQ	REAL			REFS	144	145	146	147	148		
					DEFINED	112						
2203	BTA	REAL	ARRAY		REFS	2	124	2*126	190	DEFINED	39	123
						124	126					
2071	BZ1	REAL			REFS	87	147	245	DEFINED	86	143	
2107	[illegible]	REAL			REFS	131	DEFINED	111				
10332	CEIGM	REAL	ARRAY		REFS	2	304	308	309	310	364	
					DEFINED	32	33	34	301	304		
2[illegible]	CEIGNUM	REAL	ARRAY		REFS	2	280	282	288	290	291	293
						294	304	364	389	DEFINED	28	29	30
						31	277	280	347	348	349	350	351
						352	359	364				
2061	COSX1	REAL			REFS	63	138	DEFINED	61	136		
2067	COSY1	REAL			REFS	86	143	2*242	DEFINED	84	141	
2105	CSQ	REAL			REFS	111	112	DEFINED	110			
2123	C11	REAL			REFS	153	DEFINED	144				
2124	C22	REAL			REFS	150	DEFINED	145				
2116	C3	REAL			REFS	119	123	DEFINED	118			

VARIABLES	SN	TYPE	RELOCATION								
2125	C33	REAL		REFS	149	DEFINED	146				
2126	C44	REAL		REFS	158	159	DEFINED	147			
2127	C55	REAL		REFS	158	159	DEFINED	148			
2202	DBSUM	REAL		REFS	425	DEFINED	424				
2047	DELSQ	* REAL		DEFINED	27						
4	DELT	REAL	NAME8	REFS	16	19	27	100	104	133	
5731	DENS	REAL	ARRAY	REFS	2	248	253	257	280		
				DEFINED	247	253	257				
2065	DEVK1	REAL		REFS	81	166	DEFINED	78	163		
2066	DEVK2	REAL		REFS	82	167	DEFINED	79	164		
2052	DEV1	REAL		REFS	57	DEFINED	52				
2053	DEV2	REAL		REFS	58	135	DEFINED	53	60		
2054	DEV3	REAL		REFS	63	64	138	139	DEFINED	54	
2173	DX	REAL		REFS	389	391	DEFINED	384	389		
3701	EKS	REAL	ARRAY	REFS	2	77	2*92	2*93	2*100	102	2*104
					106	160	2*171	2*172	2*284	2*285	2*312	2*313
					2*314	2*315	2*318	2*319	2*324	2*325	4*326	2*328
					2*330	DEFINED	317				
2122	ETA1	REAL		REFS	140	288	318	319	320	328	
				DEFINED	286						
2160	ETA2	REAL		REFS	318	319	321	330	DEFINED	287	
2153	EXPON	REAL		REFS	247	DEFINED	245				
11015	FKS	REAL	ARRAY	REFS	9	311	2*312	2*313	2*314	2*315	332
					333	334	335	337	338	339	340	
50	GAMA	REAL	ARRAY NAME	REFS	14	391					
11021	GKS	REAL	ARRAY	REFS	9	317	DEFINED	312	313	314	315
7371	HHH	REAL	ARRAY	REFS	2	229	231	247	DEFINED	91	183
					229	231					
2140	HHK	REAL		REFS	226	228	DEFINED	210	226		
2143	HH1	REAL		REFS	218	226	DEFINED	217	218	220	222
					224						
2144	HH2	REAL		REFS	229	DEFINED	228				
5355	HKL	REAL	ARRAY	REFS	2	2*204	217	222	DEFINED	198	200
					204						
171	HM	REAL	ARRAY NAME	REFS	14	387	388	2*389			
5345	HRK	REAL	ARRAY	REFS	2	2*203	217	224	DEFINED	197	199
					203						
2050	I	INTEGER		REFS	48	2*51	68	2*81	2*82	91	2*92
					2*93	2*166	2*167	2*171	2*172	2*176	4*177	3*179
					183	197	198	199	200	203	204	2*229
					2*231	4*245	2*247	2*248	2*253	3*257	2*280	3*304
					2*317	2*371	2*372	387	3*389	2*407		
				DEFINED	46	49	66	80	89	165	169	
					174	181	195	243	251	255	278	302
					316	370	385	406				
2141	IN1	INTEGER		REFS	214	215	217	224	DEFINED	212		
2142	IN2	INTEGER		REFS	214	216	217	222	DEFINED	213		
2134	IR	INTEGER		REFS	189	2*190	3*194	209	211	212	218	
					222	224	228	363	3*364	380	382	385
					387	389	2*391	DEFINED	188	193	208	361
					379						
2170	IS	INTEGER		REFS	363	3*364	DEFINED	362			
2166	ISS	INTEGER		REFS	356	358	381	383	DEFINED	355	356	
					380	381					

VARIABLES	SN	TYPE		RELOCATION							
2136	ITT	INTEGER			REFS	193	208	276	300	358	383	
					DEFINED	192	207	275	299	357	382	
2051	J	INTEGER			REFS	48	2*51	68	91	2*92	2*93	2*171
						2*172	2*176	4*177	3*179	183	197	198	199
						200	203	204	2*229	2*231	4*245	2*247	2*248
						2*253	3*257	262	263	2*264	2*265	3*271	3*272
						2*280	3*304	2*337	2*338	2*339	2*340	2*341	2*342
						2*343	2*344	388	3*389	DEFINED	47	50	67
						90	170	175	182	196	244	252	256
						261	270	279	303	336	386		
5	JSEED	INTEGER		NAME8	REFS	16	57	58	63	64	135	138	
						139							
2167	J1	INTEGER			REFS	364	DEFINED	360					
2171	J2	INTEGER			REFS	2*364	DEFINED	363					
2117	K	INTEGER			REFS	2*126	2*127	2*128	3*190	202	3*203	3*204	
						212	213	218	222	224	2*271	2*272	2*293
						2*294	2*295	2*296	DEFINED	125	189	201	211
						269	292						
2046	KOUNT	INTEGER			REFS	59	134	288	327	393	394		
					DEFINED	26	394						
2033	K1	REAL			REFS	11	72	74	2*87	151	155	158	
					DEFINED	71	72	149					
2034	K2	REAL			REFS	11	76	152	156	159			
					DEFINED	73	150						
2135	L	INTEGER			REFS	192	3*194	207	209	213	228	386	
						388	389	2*391	DEFINED	191	206	383	
2161	M	INTEGER			REFS	299	301	302	4*304	DEFINED	298		
2156	MM	INTEGER			REFS	275	277	3*280	355	357	359	360	
						361	4*364	DEFINED	274	354			
11071	MOM	REAL	ARRAY		REFS	11	2*284	2*285	DEFINED	282	283		
2035	M1	REAL			REFS	11	92	171	286	DEFINED	87	158	
						320							
2037	M1BAR	REAL			REFS	12	286	DEFINED	284				
2036	M2	REAL			REFS	11	93	172	287	DEFINED	88	159	
						321							
2040	M2BAR	REAL			REFS	12	287	DEFINED	285				
2162	N	INTEGER			REFS	301	303	4*304	DEFINED	300			
2155	NK1	INTEGER			REFS	269	274	275	292	298	299		
					DEFINED	268							
2157	NN	INTEGER			REFS	277	3*280	359	360	362	4*364		
					DEFINED	276	358						
2177	NNS	INTEGER			REFS	419	425	DEFINED	418				
2043	NOM	INTEGER			REFS	201	DEFINED	22					
1	NO1	INTEGER		NAME6	REFS	15	19	22	46	47	66	67	
						125	188	191	192	206	207	268	336
						353	354	357	370	379	382		
2	NO2	INTEGER		NAME6	REFS	15	393	406	416				
2044	NSAMP	INTEGER			REFS	418	421	428	429	DEFINED	23	428	
0	NUM1	INTEGER		NAME6	REFS	15	19	49	50	80	89	90	
						165	169	170	174	175	181	182	195
						196	243	244	251	252	255	256	261
						270	278	279					
2041	PI	REAL			REFS	21	409	DEFINED	22				
2042	PI2	REAL			REFS	411	DEFINED	21					

VARIABLES	SN	TYPE	RELOCATION									
2163	P11	REAL			REFS	328	DEFINED	324				
2165	P12	REAL			REFS	328	330	DEFINED	326			
2164	P22	REAL			REFS	330	DEFINED	325				
2	Q22	REAL		NAME8	REFS	16	19	60	107	109	2*118	123
2132	RHO	REAL			REFS	154	157	158	159	DEFINED	153	
2133	RHOIN	REAL			REFS	155	156	158	159	DEFINED	154	
2130	RTK1	REAL			REFS	153	158	159	DEFINED	151		
2131	RTK2	REAL			REFS	153	158	159	DEFINED	152		
3	R11	REAL		NAME8	REFS	16	19	54	71	86	143	146
						238	239	240				
2062	SINX1	REAL			REFS	64	139	DEFINED	62	137		
2070	SINY1	REAL			REFS	86	143	2*242	DEFINED	85.	142	
2145	SUMDEN	REAL			REFS	248	250	DEFINED	237	248		
2154	SUMDI	REAL			REFS	253	DEFINED	250				
2174	SUMP	REAL			REFS	414	417	419	423	DEFINED	405	414
						417						
2045	SUMP1	REAL			REFS	423	424	425	DEFINED	24	423	
2102	S11	REAL			REFS	107	108	109	110	2*118	324	325
						326	328	368	DEFINED	322		
2106	S22	REAL			REFS	110	2*118	119	324	325	326	330
						369	DEFINED	323				
0	T	REAL	ARRAY	NAME	REFS	14	81	82	166	167		
4525	TAUKL	REAL	ARRAY		REFS	2	198	200	204	2*272	280	
					DEFINED	177	263	265	272			
3705	TAURK	REAL	ARRAY		REFS	2	177	179	197	199	203	2*271
						280	DEFINED	176	179	262	264	271
2072	TT1	REAL			REFS	101	102	113	114	121	176	
					DEFINED	100						
10773	TT1PH	REAL	ARRAY		REFS	8	293	295	304	DEFINED	44	290
						293	295					
2073	TT1SQ	REAL			REFS	107	108	109	110	2*118		
					DEFINED	101						
2074	TT2	REAL			REFS	103	115	144	147	176		
					DEFINED	102						
11004	TT2PH	REAL	ARRAY		REFS	8	294	296	304	DEFINED	45	291
						294	296					
2075	TT2SQ	REAL			REFS	146	DEFINED	103				
2076	TT3	REAL			REFS	105	106	114	122	177		
					DEFINED	104						
2077	TT3SQ	REAL			REFS	110	2*118	119	DEFINED	105		
2100	TT4	REAL			REFS	115	177	DEFINED	106			
2112	V1BAR	REAL			REFS	113	114	176	DEFINED	318		
2114	V2BAR	REAL			REFS	114	177	DEFINED	319			
2111	V4	REAL			REFS	147	148	DEFINED	113			
2113	V5	REAL			REFS	147	148	DEFINED	114			
2115	V6	REAL			REFS	144	146	147	DEFINED	115		
24	WT	REAL	ARRAY	NAME	REFS	14	2*51					
6551	WT2	REAL	ARRAY		REFS	2	248	257	DEFINED	51		
2176	X	REAL			REFS	417	DEFINED	416				
2201	XAA	REAL			REFS	423	DEFINED	422				
2175	XD	REAL			REFS	409	411	414	DEFINED	407	411	
11102	XDAT	REAL	ARRAY		REFS	13	2*407	DEFINED	59	134	288	
2137	XK	REAL			REFS	203	204	DEFINED	202			
3676	XKS	REAL	ARRAY		REFS	2	77	78	79	2*157	160	163

VARIABLES	SN	TYPE	RELOCATION							
				164	311	322	323	351	352	
				DEFINED	74	75	76	155	156	157	308
				309	310					
2200	XNSAMP	REAL		REFS	422	423	DEFINED	421		
0	XONE	REAL	NAME9	REFS	17	55				
1	XTWO	REAL	NAME9	REFS	17	56				
2057	X1	REAL		REFS	57	59	61	62	133	134	136
				137	328	DEFINED	133			
2060	X2	REAL		REFS	58	133	2*135	330		
2236	Y1	REAL	ARRAY	REFS	2	176	177	4*245	DEFINED	92	171
2055	Y1EST	REAL		REFS	57	84	85	86	87	141	142
				143	241	2*242	DEFINED	55	140		
2151	Y1ES2	REAL		REFS	245	DEFINED	241				
5516	Y1P	REAL	ARRAY	REFS	2	92	93	171	172	264	
				DEFINED	81	166					
3056	Y2	REAL	ARRAY	REFS	2	177	179	DEFINED	93	172	
2056	Y2EST	REAL		REFS	58	88	DEFINED	56			
5542	Y2P	REAL	ARRAY	REFS	2	92	93	171	172	265	
				DEFINED	82	167					
2150	ZSQ	REAL		REFS	245	DEFINED	240				
2063	Z1	REAL		REFS	63	86	138	143	238	328	
2146	Z1R	REAL		REFS	242	245	DEFINED	238			
2152	Z1RS	REAL		REFS	245	DEFINED	242				
2064	Z2	REAL		REFS	64	86	139	143	239	330	
2147	Z2R	REAL		REFS	242	245	DEFINED	239			

FILE NAMES	MODE								
TAPE6	FMT	WRITES	18	19	327	328	329	330	419	425

EXTERNALS	TYPE	ARGS	REFERENCES							
ALOG10	REAL	1 LIBRARY	424							
COS	REAL	1 LIBRARY	61	84	136	141	245			
EIGEN		2	77	160	311					
EXP	REAL	1 LIBRARY	245							
GAUSS		4	57	58	63	64	135	138	139	
SIN	REAL	1 LIBRARY	62	85	137	142	245			
SQRT	REAL	1 LIBRARY	52	53	54	60	78	79	111	116	120
			124	130	131	151	152	157	163	164	368
			369							

INLINE FUNCTIONS	TYPE	ARGS	DEF LINE	REFERENCES						
ABS	REAL	1 INTRIN		295	296	341	342	343	344	407

STATEMENT LABELS		DEF LINE	REFERENCES		
34	2	25	430		
0	30	48	46	47	
0	35	51	49	50	
0	41	68	66	67	
0	51	83	80		
0	56	94	89	90	
224	70	99	395		
0	120	129	125		
0	200	INACTIVE	165		
0	211	168	165		

STATEMENT	LABELS		DEF LINE	REFERENCES			
0	212		173	169	170		
0	215		180	174	175		
0	226		205	201			
0	251		190	188	189		
0	255		194	191	193		
0	301		184	181	182		
670	304		220	214			
672	305		222	215			
674	306		224	216			
676	307		225	219	221	223	
0	308		227	211			
713	310		230	206	208	209	
0	311		232	195	196		
0	411		249	243	244		
0	421		254	251	252		
734	425		233	95			
0	426		258	255	256		
0	461		306	298	300	302	303
0	473		317	316			
0	477		266	261			
0	478		273	269	270		
0	491		281	274	276	278	279
0	492		345	336			
0	495		366	354	358	361	362
1510	496		367	353			
0	500	INACTIVE	346				
0	503		297	292			
0	505		373	370			
1564	510		390	385	386	387	388
0	511		392	379	383		
1610	550		404	393			
1753	617	FMT	399	329			
1761	621	FMT	400	328	330		
1767	622	FMT	401	327			
1736	652	FMT	396	18			
1750	653	FMT	398	19			
1617	1498		408	412			
1621	1499		411	409			
1623	1500		413	410			
0	1501		415	406			
2001	1511	FMT	420	419			
2010	1521	FMT	426	425			
1670	1750		431	429			

LOOPS	LABEL		INDEX	FROM-TO	LENGTH	PROPERTIES
52	30	*	I	46 48	108	NOT INNER
56	30		J	47 48	28	INSTACK
63	35	*	I	49 51	118	NOT INNER
67	35		J	50 51	28	INSTACK
124	41	*	I	66 68	108	NOT INNER
130	41		J	67 68	28	INSTACK
161	51		I	80 83	38	INSTACK
202	56	*	I	89 94	218	NOT INNER
210	56		J	90 94	118	OPT

LOOPS	LABEL		INDEX	FROM-TO	LENGTH	PROPERTIES
325	120		K	125 129	38	INSTACK
453	211		I	165 168	38	INSTACK
457	212	*	I	169 173	208	NOT INNER
465	212		J	170 173	78	INSTACK
500	215	*	I	174 180	278	NOT INNER
513	215		J	175 180	118	OPT
530	301	*	I	181 184	168	NOT INNER
534	301		J	182 184	28	INSTACK
541	251	*	IR	188 190	118	NOT INNER
545	251		K	189 190	38	INSTACK
553	255	*	L	191 194	138	NOT INNER
561	255		IR	193 194	38	INSTACK
567	311	*	I	195 232	1458	NOT INNER
570	311	*	J	196 232	1428	NOT INNER
617	226	*	K	201 205	118	OPT
632	310	*	L	206 230	668	NOT INNER
636	310	*	IR	208 230	608	NOT INNER
652	308	*	K	211 227	308	OPT
750	411	*	I	243 249	478	EXT REFS NOT INNER
751	411	*	J	244 249	448	EXT REFS
1022	421	*	I	251 254	118	NOT INNER
1027	421		J	252 254	28	INSTACK
1034	426	*	I	255 258	128	NOT INNER
1041	426		J	256 258	28	INSTACK
1053	477		J	261 266	48	INSTACK
1061	478	*	K	269 273	148	NOT INNER
1067	478		J	270 273	38	INSTACK
1076	491	*	MM	274 281	348	NOT INNER
1102	491	*	NN	276 281	268	NOT INNER
1106	491	*	I	279 281	178	NOT INNER
1117	491		J	279 281	48	INSTACK
1153	503		K	292 297	128	OPT
1166	461	*	M	298 306	448	NOT INNER
1172	461	*	N	300 306	358	NOT INNER
1176	461	*	I	302 306	278	NOT INNER
1212	461	*	J	303 306	78	INSTACK
1255	473		I	316 317	28	INSTACK
1373	492		J	336 345	238	OPT
1430	495	*	MM	354 366	608	NOT INNER
1441	495	*	NN	358 366	448	NOT INNER
1446	495	*	IR	361 366	338	NOT INNER
1462	495	*	IS	362 366	158	OPT
1523	505		I	370 373	38	INSTACK
1532	511	*	IR	379 392	518	NOT INNER
1543	511	*	L	383 392	356	NOT INNER
1545	510	*	I	385 390	238	NOT INNER
1554	510		J	386 390	128	OPT
1615	1501		I	406 415	118	OPT

COMMON BLOCKS	LENGTH	MEMBERS - BIAS NAME(LENGTH)							
NAME	491	0 T	(20)	20 WT	(20)	40 GAMA	(81)		
		121 HH	(81)	202 BINOM	(289)				
NAME6	3	0 NUM1	(1)	1 NO1	(1)	2 NO2	(1)		
NAME8	6	0 A11	(1)	1 A22	(1)	2 Q22	(1)		

COMMON BLOCKS LENGTH MEMBERS - BIAS NAME(LENGTH)

NAME9 2 3 RII (1) 4 DELT (1) 5 JSEED (1)
0 XONE (1) 1 XTWO (1)

STATISTICS
PROGRAM LENGTH 12364B 5364
COMMON LENGTH 766B 502

```
      FUNCTION RANF(NS,MODE)
      DIMENSION  NS(2), NC(2)
      COMMON /RN/ N1, N2, MP, T1, T2
      DATA  M1, M2/244734, 158551/
C        MODE=0 TO CONTINUE, OTHERWISE RESTART WITH
C        INTEGER NUMBER NS(1)*2**18+NS(2)
      IF (MODE) 10, 100, 10
   10 N1=NS(1)
      N2=NS(2)
      T1=2.**(-18)
      T2=2.**(-36)
      MP=2**18
  100 DO 200 I=1,2
      GO TO (110,120),I
  110 K=M2*N2
      GO TO 190
  120 K=M1*N2+M2*N1+KD
  190 KD=K/MP
  200 NC(I)=K-KD*MP
      N1=NC(2)
      N2=NC(1)
      XN1=N1
      XN2=N2
      RANF=XN1*T1+XN2*T2
      RETURN
      END
```

SYMBOLIC REFERENCE MAP

ENTRY POINTS

		DEF LINE	REFERENCES
2	RANF	1	25

VARIABLES

		SN	TYPE	RELOCATION								
76	I		INTEGER		REFS	14	19	DEFINED	13			
77	K		INTEGER		REFS	18	19	DEFINED	15	17		
100	KD		INTEGER		REFS	17	19	DEFINED	18			
0	MODE		INTEGER	F.P.	REFS	7	DEFINED	1				
2	MP		INTEGER	RN	REFS	3	18	19	DEFINED	12		
72	M1		INTEGER		REFS	17	DEFINED	4				
73	M2		INTEGER		REFS	15	17	DEFINED	4			
103	NC		INTEGER	ARRAY	REFS	2	20	21	DEFINED	19		
0	NS		INTEGER	ARRAY F.P.	REFS	2	8	9	DEFINED	1		
0	N1		INTEGER	RN	REFS	3	17	22	DEFINED	8	20	
1	N2		INTEGER	RN	REFS	3	15	17	23	DEFINED	9	21
75	RANF		REAL		DEFINED	24						
3	T1		REAL	RN	REFS	3	24	DEFINED	10			
4	T2		REAL	RN	REFS	3	24	DEFINED	11			
101	XN1		REAL		REFS	24	DEFINED	22				
102	XN2		REAL		REFS	24	DEFINED	23				

STATEMENT LABELS

			DEF LINE	REFERENCES
0	10	INACTIVE	8	2*7
27	100		13	7
41	110		15	14
44	120		17	14
51	190		18	16
0	200		19	13

LOOPS

LOOPS	LABEL		INDEX	FROM-TO	LENGTH	PROPERTIES
33	200	*	I	13 19	258	OPT

COMMON BLOCKS

	LENGTH	MEMBERS - BIAS NAME(LENGTH)					
RN	5	3 N1	(1)	1 N2	(1)	2 MP	(1)
		3 T1	(1)	4 T2	(1)		

STATISTICS

PROGRAM LENGTH	1128	74
COMMON LENGTH	58	5

```
      SUBROUTINE GAUSS(JS,SD,XM,X)
      DIMENSION  NST(2)
      COMMON /RN/ N1, N2, MC, T1, T2
      COMMON /GN/ TWOPI, J, XR(2)
      IF (J) 10, 10, 20
   10 J=2
      TWOPI=8.*ATAN(1.)
      NST(1)=244734
      NST(2)=158551
      XR(1)=RANF(NST,1)
      GO TO 35
   20 GO TO (30,40), J
   30 J=2
      XR(1)=RANF(NST,0)
   35 XR(2)=RANF(NST,0)
      X1=SQRT(ABS(-2.*ALOG(XR(1))))
      XR(2)=TWOPI*XR(2)
      XR(1)=X1*SIN(XR(2))
      XR(2)=X1*COS(XR(2))
      X=XR(1)*SD+XM
      RETURN
   40 J=1
      X=XR(2)*SD+XM
      RETURN
      END
```

```
      SUBROUTINE  GAUSS                                    CDC 6600 FTN V3.0-P213 OPT=2  03/19/72  00.54.44.      PAGE      2

      SYMBOLIC REFERENCE MAP

ENTRY POINTS      DEF LINE      REFERENCES
    2  GAUSS            1          21         24

VARIABLES      SN  TYPE            RELOCATION
    1  J           INTEGER              GN            REFS          4          5         12    DEFINED         6         13         22
    0  JS          INTEGER   *UNUSED     F.P.       DEFINED          1
    2  MC          INTEGER              RN            REFS          3
  100  NST         INTEGER    ARRAY                   REFS          2         10         14         15    DEFINED         8          9
    0  N1          INTEGER              RN            REFS          3
    1  N2          INTEGER              RN            REFS          3
    0  SD          REAL                  F.P.         REFS         20         23    DEFINED         1
    0  TWOPI       REAL                 GN            REFS          4         17    DEFINED         7
    3  T1          REAL                 RN            REFS          3
    4  T2          REAL                 RN            REFS          3
    0  X           REAL                  F.P.       DEFINED          1         20         23
    0  XM          REAL                  F.P.         REFS         20         23    DEFINED         1
    2  XR          REAL       ARRAY     GN            REFS          4         16         17         18         19         20         23
                                                    DEFINED         10         14         15         17         18         19
   77  X1          REAL                               REFS         18         19    DEFINED        16

EXTERNALS           TYPE    ARGS        REFERENCES
      ALOG        REAL        1 LIBRARY      16
      ATAN        REAL        1 LIBRARY       7
      COS         REAL        1 LIBRARY      19
      RANF        REAL        2              10         14         15
      SIN         REAL        1 LIBRARY      18
      SQRT        REAL        1 LIBRARY      16

INLINE FUNCTIONS    TYPE    ARGS         DEF LINE  REFERENCES
      ABS         REAL        1  INTRIN                 16

STATEMENT LABELS                DEF LINE    REFERENCES
    0  10           INACTIVE        6          2*5
   17  20                          12           5
   25  30                          13          12
   31  35                          15          11
   54  40                          22          12

COMMON BLOCKS    LENGTH      MEMBERS - BIAS NAME(LENGTH)
      RN             5                0 N1      (1)                  1 N2      (1)                  2 MC      (1)
                                      3 T1      (1)                  4 T2      (1)
      GN             4                0 TWOPI   (1)                  1 J       (1)                  2 XR      (2)

STATISTICS
  PROGRAM LENGTH     1028       66
   COMMON LENGTH      118        9
```

EQUILIBRIUM CONTINUOUS POSITION VARIANCE = 1.000000E-04
FILTER TIME CONSTANT = 2.714418E-01
DELTA = 2.714418E-02
DISCRETE DRIVING VARIANCE = 2.714418E-04
DISCRETE NOISE VARIANCE = 5.000000E-04
VARIANCE OF INIT. POS. AND VEL. 1.000000E-04 2.714418E-03

CONTINUOUS DRIVING VARIANCE = 1.000000E-02
CONTINUOUS NOISE VARIANCE = 1.357209E-05

HERMITE COEFICIENTS

1.

0. 2.

-2. 0. 4.

0. -12. 0. 8.

12. 0. -48. 0. 16.

BINOMIAL COEFFICIENTS

```
1.
1.      1.
1.      2.      1.
1.      3.      3.      1.
1.      4.      6.      4.      1.
```

NORMALIZING COEFFICIENTS

```
1.000000E+00  5.000000E-01  1.250000E-01  2.083333E-02  2.604167E-03  5.000000E-01  2.500000E-01  6.250000E-02  1.041667E-02
1.302083E-03  1.250000E-01  6.250000E-02  1.562500E-02  2.604167E-03  3.255208E-04  2.083333E-02  1.041667E-02  2.604167E-03
4.340278E-04  5.425347E-05  2.604167E-03  1.302083E-03  3.255208E-04  5.425347E-05  6.781684E-06
```

A11	A22	Q22	R11	DELT	NO. OF INTG. PTS.	NO. OF TERMS
1.000000E-04	2.714418E-03	2.714418E-04	5.000000E-04	2.714418E-02	10	5

Reproduced from best available copy.

TIME = 1	Z1	X1	ETA1	P11	P12	EIGENVALUES	EIGENVECTORS	
	9.865661E-01	1.360673E-02	5.187855E-03	8.351885E-05	0.	8.351885E-05	1.000000E+00	0.
	Z2	X2	ETA2	P12	P22			
	3.105892E-02	7.803262E-03	1.215371E-16	0.	2.714418E-03	2.714418E-03	0.	1.000000E+00
TIME = 2	Z1	X1	ETA1	P11	P12	EIGENVALUES	EIGENVECTORS	
	1.017557E+00	1.381854E-02	7.643203E-03	7.283972E-05	6.275732E-05	7.148395E-05	9.997667E-01	2.159844E-02
	Z2	X2	ETA2	P12	P22			
	2.213402E-02	2.799281E-03	2.115449E-03	6.275732E-05	2.976448E-03	2.977804E-03	-2.159844E-02	9.997667E-01
TIME = 3	Z1	X1	ETA1	P11	P12	EIGENVALUES	EIGENVECTORS	
	9.799179E-01	1.389452E-02	8.731258E-03	6.798722E-05	1.244222E-04	6.307236E-05	9.992207E-01	3.947070E-02
	Z2	X2	ETA2	P12	P22			
	1.512597E-02	1.143504E-02	4.001543E-03	1.244222E-04	3.212883E-03	3.217798E-03	-3.947070E-02	9.992207E-01
TIME = 4	Z1	X1	ETA1	P11	P12	EIGENVALUES	EIGENVECTORS	
	9.601708E-01	1.420492E-02	1.027156E-02	6.715242E-05	1.843345E-04	5.702656E-05	9.984917E-01	5.493331E-02
	Z2	X2	ETA2	P12	P22			
	1.914697E-02	2.019664E-02	7.936894E-03	1.843345E-04	3.409401E-03	3.419537E-03	-5.493331E-02	9.984917E-01
TIME = 5	Z1	X1	ETA1	P11	P12	EIGENVALUES	EIGENVECTORS	
	1.015889E+00	1.475314E-02	1.644970E-02	6.857135E-05	2.382763E-04	5.232364E-05	9.976832E-01	6.803053E-02
	Z2	X2	ETA2	P12	P22			
	5.413595E-02	1.598442E-02	2.865091E-02	2.382763E-04	3.546701E-03	3.562946E-03	-6.803053E-02	9.976832E-01
TIME = 6	Z1	X1	ETA1	P11	P12	EIGENVALUES	EIGENVECTORS	
	9.826545E-01	1.518702E-02	2.103715E-02	7.218435E-05	2.870813E-04	4.916449E-05	9.968005E-01	7.992927E-02
	Z2	X2	ETA2	P12	P22			
	4.331333E-02	2.539935E-02	4.330244E-02	2.870813E-04	3.662936E-03	3.682385E-03	-7.992927E-02	9.968005E-01
TIME = 7	Z1	X1	ETA1	P11	P12	EIGENVALUES	EIGENVECTORS	
	9.927956E-01	1.587647E-02	1.772359E-02	7.667975E-05	3.269138E-04	4.702248E-05	9.959103E-01	9.034794E-02
	Z2	X2	ETA2	P12	P22			
	-7.298512E-03	2.572762E-02	2.486627E-02	3.269138E-04	3.650610E-03	3.680268E-03	-9.034794E-02	9.959103E-01
TIME = 8	Z1	X1	ETA1	P11	P12	EIGENVALUES	EIGENVECTORS	
	1.031279E+00	1.657482E-02	1.849123E-02	8.098804E-05	3.549052E-04	4.557225E-05	9.958801E-01	9.907418E-02
	Z2	X2	ETA2	P12	P22			
	1.958564E-02	2.691373E-02	2.504391E-02	3.549052E-04	3.610165E-03	3.645501E-03	-9.907418E-02	9.958801E-01
TIME = 9	Z1	X1	ETA1	P11	P12	EIGENVALUES	EIGENVECTORS	
	1.007272E+00	1.714251E-02	2.152361E-02	8.518057E-05	3.751475E-04	4.490443E-05	9.942862E-01	1.067473E-01
	Z2	X2	ETA2	P12	P22			
	3.371209E-02	2.806250E-02	3.584514E-02	3.751475E-04	3.539174E-03	3.579450E-03	-1.067473E-01	9.942862E-01
TIME = 10	Z1	X1	ETA1	P11	P12	EIGENVALUES	EIGENVECTORS	
	1.013337E+00	1.776852E-02	2.334114E-02	8.871063E-05	3.865011E-04	4.473295E-05	9.935888E-01	1.130546E-01
	Z2	X2	ETA2	P12	P22			
	1.018557E-02	2.949414E-02	2.631838E-02	3.865011E-04	3.441526E-03	3.485504E-03	-1.130546E-01	9.935888E-01
TIME = 11	Z1	X1	ETA1	P11	P12	EIGENVALUES	EIGENVECTORS	
	9.687519E-01	1.840625E-02	1.976813E-02	9.218325E-05	3.941984E-04	4.511312E-05	9.929463E-01	1.185649E-01
	Z2	X2	ETA2	P12	P22			
	1.345221E-02	3.153893E-02	2.054795E-02	3.941984E-04	3.346408E-03	3.393478E-03	-1.185649E-01	9.929463E-01
TIME = 12	Z1	X1	ETA1	P11	P12	EIGENVALUES	EIGENVECTORS	
	1.004325E+00	1.926235E-02	2.446346E-02	9.409966E-05	3.932947E-04	4.559258E-05	9.924799E-01	1.224077E-01
	Z2	X2	ETA2	P12	P22			
	4.240717E-02	2.498145E-02	3.784114E-02	3.932947E-04	3.234420E-03	3.282927E-03	-1.224077E-01	9.924799E-01

Additional Appendix D.

A Point-Mass Program for Implementing the Interpolating Version of the Cyclic Phase Demodulator

This program simulates the cyclic interpolating demodulator, described in Appendix C of Chapter VII.

The program was compiled on one of the Kirtland AFB CDC 6600's using the FTN compiler. The code-optimization algorithm was employed. The random number generator is identical to the one described in Chapter IV.

Programmer: C. Hecht

```
      PROGRAM MAIN(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)
      COMMON DENS1(21,105),DENS2(21,105),KLOW(21,105),KUP(21,105),
     1 PROP(21,105)
      COMMON /RN/ NZZZ(3), XNZ(2)
      COMMON /GN/ DZZZ1, JGAUSS, XZZZ(2)
      DIMENSION XDAT(1000,5),XHAT(2)
      DIMENSION
     1 Y1(81),Y2(287),SINY(81),COSY(81),EXDON(30),EXDON(15)
      DIMENSION XDATM(1000,5)
      DIMENSION Y3(15)
      DIMENSION SENS(81)
      DIMENSION Q(2,2),PBAR(2,2),PN(2,2),AN(2),F(2,2),PDUMY(2,2),
     1 PDUMY2(2,2),          EIGVEC(2,2),      PNF(2,2)
      DIMENSION XDEN(21)
      LOGICAL LOW,JP
      LOGICAL SPLST, SPL
      DATA NBL, NAS/1H , 1H*/
    2 C      UE
      J      =0
      READ(5,651)Y1EST,Y2EST,ALP110,DELF,Q22C,NUM1,NUM2,NO2,JSEED
      SPL=.FALSE.
      IF (JSEED.GT.0) GO TO 6666
      SPL=.TRUE.
      JSEED=-JSEED
 6666 CONTINUE
      P110=10.**(ALP110/10.)
      QQ=Q22C**(.25)
      RX=(P110/(SQRT(2.0)*QQ))**(4.0/3.0)
      FTC=SQRT(2.0)* RX**(.25) /QQ
      DELT=DELF*FTC
      Q22=Q22C*DELT
      R11=RX/DELT
      A11=10.**((ALP110+1.4)/10.)
      A22=2.*P110/(FTC*FTC)
      WRITE (6,652) P110,ALP110,FTC,DELT,Q22,R11
      WRITE(6,653) Q22C,RX
  653 FORMAT(*0*,*CONTINUOUS DRIVING VARIANCE =*,1PE14.6/
     1* CONTINUOUS NOISE VARIANCE =*,1PE14.6)
  651 FORMAT(5E10.4,4I5)
  652 FORMAT(*1*,*EQUILIBRIUM CONTINUOUS POSITION VARIANCE =*,1PE14.6
     1 * (*1PE14.6* DB)*/
     1 * FILTER TIME CONSTANT =*                                ,1PE14.6/
     1 * DELTA =*                                               ,1PE14.6/
     1 * DISCRETE DRIVING VARIANCE =*                           ,1PE14.6/
     1 * DISCRETE NOISE VARIANCE =*                             ,1PE14.6)
      XMCPN=0.
      XMCCN=0.
      XMCCSN=0.
      XMCPP=0.
      XMCCP=0.
      XMCCSP=0.
      NMC=1
      DO 8999 I=1,NO2
      DO 8999 J=1,5
 8999 XDATM(I,J)=0.
```

```
      900  KOUNT=1
           SPLST=.FALSE.
           IF (NMC.EQ.JSEED.OR.SPL) SPLST=.TRUE.
           DELSQ=DELT**2
   60      PI=3.1415926536
           PI2=2.0*PI
           PIDLT=PI/DELT
           PINV=1.0/PI
           PI2DLT=2.0*PIDLT
   65      U1=NUM1
           U2=NUM2
           Q(1,1)=Q11
           Q(2,2)=Q22
           XHAT(1)=Y1EST
   70      XHAT(2)=Y2EST
           XHAT1=Y1EST
           XHAT2=Y2EST
           PN(1,1)=A11
           PN(2,2)=A22
   75      PN(1,2)=0.
           PN(2,1)=0.
           R=R11
           DEN =1.0/(PN(1,1) + R)
           F(1,1)=1.0
   80      F(1,2)=DELT
           F(2,1)=0.
           F(2,2)=1.0
           A=DELT*PINV*SQRT(10.0*Q22)
           IA=A+0.5
   85      IY2=U2/PI2DLT*SQRT(50.0*Q22) + .5
           IF (NMC.LE.1) WRITE (6,901) IA,IY2
      901  FORMAT(*0*,10X,*NUMBER OF CYCLES ON EACH SIDE =*,I4,6X,
          1 *NUMBER OF POINTS ON EACH SIDE =*,I4)
     C*****      GRID Y1 AND Y2      *****
   90      EDG1=PI/U1
           EDG2=PIDLT/U2
           DO 49 I=1,NUM1
           X=I-1
           X=X/U1
   95   49 Y1(I)=-PI + X*PI2 +EDG1
           DO 50 I=1,NUM2
           X=I-1
           X=X/U2
   50   50 Y2(I)=-PIDLT + X*PI2DLT + EDG2
  100      DO 51 I=1,IY2
           X=I
           X=X/U2
   51   51 Y3(I)=X*PI2DLT
           IF (NMC.GT.1) GO TO 9689
  105      WRITE(6,913)
           WRITE(6,912)(Y1(I),I=1,NUM1)
           WRITE(6,914)
           WRITE(6,912)(Y2(I),I=1,NUM2)
     9689  CONTINUE
  110   912 FORMAT(* *,1P10E12.4)
```

```
      913 FORMAT(*0*,*Y1 GRID*)
      914 FORMAT(*0*,*Y2 GRID*)
          DO 55 I=1,NUM1
          COSY(I)=COS(Y1(I))
115    55 SINY(I)=SIN(Y1(I))
C*****      SUM OF EXPONENTIALS      *****
          IB=2*IA +1
          IF(IY2.EQ.0) GO TO 153
          DO 150 I=1,IY2
120       EXPON(I)=0.0
          DO 150 IR=1,IB
          IC=-IA-1+IR
          R=IC
          R=PI2OLT*R
125   15. EXPON(I)=EXPON(I) + EXP(-0.5/Q22*(Y2(I)+R)**2)
          DO 152 I=1,IY2
          EXPON(I)=EXPON(IY2+1-I)
          EXPON(I+IY2+1)=EXPON(I)
      152 CONTINUE
130       153 CONTINUE
          EXPON(IY2+1)=1.0
          IYY=2*IY2 + 1
          WRITE(6,912)(EXPON(I),I=1,IYY )
C*****      INITIAL DENSITY      *****
135       SUM=0.0
          DO 160 I=1,NJM1
          DO 160 J=1,NJM2
          DENS1(I,J)=EXP(-0.5/A11*Y1(I)**2-.5/A22*Y2(J)**2)
      16. SUM=SUM+DENS1(I,J)
140       SUMI=1.0/SUM
          DO 165 I=1,NJM1
          DO 165 J=1,NJM2
      165 DENS1(I,J)=SUMI*DENS1(I,J)
C*****      INTEGER ARRANGEMENT      *****
145       RATIO=U1/U2
          DO 250 I=1,NUM1
          AI=I
          DO 250 J=1,NJM2
          AJ=J
150       AK=AI-RATIO*AJ + .5*(RATIO+U1)
          KL=AK
          IF(AK.LT.0.0)KL=KL-1
          KU=KL+1
          REALK=KL
155       PROP(I,J)=AK-REALK
          KLOW(I,J)=KL
          IF(KL.LT.1)KLOW(I,J)=KL+NUM1
          IF(KL.GT.NUM1)KLOW(I,J)=KL-NUM1
          KUP(I,J)=KU
160       IF(KU.LT.1)KUP(I,J)=KU+NJM1
          IF(KU.GT.NUM1)KUP(I,J)=KU-NUM1
      25. CONTINUE
          DEV1= SQRT(A11)
          DEV2= SQRT(A22)
165       DEV3= SQRT(R11)
```

```
                CALL GAUSS(JSEED,DEV1,Y1EST,X1  )
                XDAT(KOUNT,1)=X1
                CALL GAUSS(JSEED,DEV2,Y2EST,X2  )
                CALL GAUSS(JSEED,DEV3,COS(X1  ),Z1)
170             CALL GAUSS(JSEED,DEV3,SIN(X1  ),Z2)
                DEVQ2= SQRT(Q22)
                R=R11
                NUNNL=0
                NUNPL=0
175             CS=.75*PI
                ENL=ABS(XDAT(1,1)-XDAT(1,2))
                EPL=ABS(XDAT(1,1)-XDAT(1,4))
                IF (ENL.GT.CS) NUNNL=NUNNL+1
                IF (EPL.GT.CS) NUNPL=NUNPL+1
180             DO 5656 I=1,NJM1
                XDEN(I)=0.
                DO 5656 J=1,NJM2
 5656           XDEN(I)=XDEN(I)+DENS1(I,J)
                GO TO 470
185              450 CONTINUE
                X1=X1 + X2*DELT
                XDAT(KOUNT,1)=X1
                CALL GAUSS(JSEED,DEVQ2,X2,X2)
                CALL GAUSS(JSEED,DEV3,COS(X1  ),Z1)
190             CALL GAUSS(JSEED,DEV3,SIN(X1  ),Z2)
C  ***  RICCATI EQUATION UPDATE  *******
                PDUMY(1,1)=(R*(PN(1,1)+2.*PN(1,2)*DELT)-PN(1,2)**2*DELSQ)*DEN
               1  + PN(2,2)*DELSQ
                PDUMY(1,2)=PN(1,2)*(R-PN(1,2)*DELT)*DEN + PN(2,2)*DELT
195             PDUMY(2,2)=-PN(1,2)**2*DEN + PN(2,2) + Q(2,2)
                PN(1,1)=PDUMY(1,1)
                PN(1,2)=PDUMY(1,2)
                PN(2,2)=PDUMY(2,2)
                 PN(2,1) = PN(1,2)
200             DEN =1.0/(PN(1,1) + R)
                470 CONTINUE
C*****      UPDATE DENSITY FUNCTION      *****
                SUM=0.0
                DO 285 I=1,NJM1
205              285 SENS(I)=EXP(-.5/R11*((Z1-COSY(I))**2+( Z2-SINY(I))**2) )
                DO 310 J=1,NJM2
                LOW=.FALSE.
                UP=.FALSE.
                K1=J-IY2
210             K2=J+IY2
                IF(K1.LT.1)LOW=.TRUE.
                IF(K2.GT.NUM2)UP=.TRUE.
                IF(LOW) K1=K1+NUM2
                IF(UP)K2=K2-NJM2
215             DO 311 I=1,NJM1
                MM=0
                DUM=0.0
                IF(LOW) GO TO 301
                IF(UP) GO TO 305
220             DO 300 L=K1,K2
```

```
            MM=MM+1
            KL=KLOW(I,L)
            KU=KUP(I,L)
            DUM=EXPON(MM)*(DENS1(KL,L)*(1.-PROP(KL,L))+DENS1(KU,L)*PROP(KL,L))
225        1 + DUM
          300 CONTINUE
            GO TO 315
          301 CONTINUE
            DO 303 L=K1,NUM2
230         MM=MM+1
            KL=KLOW(I,L)
            KU=KUP(I,L)
            DUM=EXPON(MM)*(DENS1(KL,L)*(1.-PROP(KL,L))+DENS1(KU,L)*PROP(KL,L))
           1 + DUM
235       303 CONTINUE
            DO 304 L=1,K2
            MM=MM+1
            KL=KLOW(I,L)
            KU=KUP(I,L)
240         DUM=EXPON(MM)*(DENS1(KL,L)*(1.-PROP(KL,L))+DENS1(KU,L)*PROP(KL,L))
           1 + DUM
          304 CONTINUE
            GO TO 315
          305 CONTINUE
245         DO 306 L=K1,NUM2
            MM=MM+1
            KL=KLOW(I,L)
            KU=KUP(I,L)
            DUM=EXPON(MM)*(DENS1(KL,L)*(1.-PROP(KL,L))+DENS1(KU,L)*PROP(KL,L))
250        1 + DUM
          306 CONTINUE
            DO 307 L=1,K2
            MM=MM+1
            KL=KLOW(I,L)
255         KU=KUP(I,L)
            DUM=EXPON(MM)*(DENS1(KL,L)*(1.-PROP(KL,L))+DENS1(KU,L)*PROP(KL,L))
           1 + DUM
          307 CONTINUE
          315 CONTINUE
260         IF(DUM       .LT.1.0E-30) GO TO 308
            DUM=DUM*SENS(I)
            GO TO 309
          308 DUM=0.0
          309 SUM=DUM + SUM
265         DENS2(I,J)=DUM
          310 CONTINUE
            SUMI=1.0/SUM
            DO 312 I=1,NJM1
            DO 312 J=1,NJM2
270         DENS1(I,J)=SUMI*DENS2(I,J)
          312 CONTINUE
            SHAT=0.0
            CHAT=0.0
            DO 360 I=1,NJM1
275         DO 360 J=1,NJM2
```

```
                SHAT=SHAT+DENS1(I,J)*SINY (I)
                CHAT=CHAT+DENS1(I,J)*COSY(I)
             36. CONTINUE
                CXHAT  =ATAN(SHAT/CHAT)
280             IF(CHAT.GT.0.0) GO TO 367
                IF(SHAT.GT.0.0) CXHAT=CXHAT + PI
                IF(SHAT.LT.0.0) CXHAT=CXHAT-PI
             367 PLOSS=2.0*(1.)-SQRT(SHAT**2+CHAT**2))
                XDAT(KOUNT,2)=CXHAT
285             XDAT(KOUNT,3)=PLOSS
                PNF(1,1)=PN(1,1)*R*DEN
                PNF(1,2)=PN(1,2)*R*DEN
                PNF(2,1)=PNF(1,2)
                PNF(2,2)=PN(2,2) - PN(1,2)**2*DEN
290          C  ***    FILTER UPDATE    ****
                SINF1=SIN(XHAT1)
                COSF1=COS(XHAT1)
                XHAT(1)=XHAT1+DEN*(-PN(1,1)*SINF1*Z1+PN(1,1)*COSF1*Z2)
                XHAT(2)=XHAT2+DEN*(-PN(1,2)*SINF1*Z1+PN(1,2)*COSF1*Z2)
295             X1FF=XHAT(1)
              84 CONTINUE
                IF(X1FF.GT.PI) GO TO 88
                IF(X1FF.LT.-PI) GO TO 89
                GO TO 90
300           88 X1FF=X1FF-PI2
                GO TO 84
              89 X1FF=X1FF+PI2
                GO TO 84
              90 CONTINUE
305          C   ***  PREDICTOR UPDATE  **
                XHAT1=XHAT(1) + DELT*XHAT(2)
                XHAT2=XHAT(2)
                XDAT(KOUNT,4)=XHAT(1)
                XDAT(KOUNT,5)=PNF(1,1)
310             ENL=ABS(XDAT(KOUNT,1)-XDAT(KOUNT,2))
                EPL=ABS(XDAT(KOUNT,1)-XDAT(KOUNT,4))
                IF (ENL.GT.CS) NUNNL=NUNNL+1
                IF (EPL.GT.CS) NUNPL=NUNPL+1
                DO 5676 I=1,NUM1
315             XDEN(I)=0.
                DO 5676 J=1,NUM2
             5676 XDEN(I)=XDEN(I)+DENS1(I,J)
                X1MOD=X1
              184 CONTINUE
320             IF(X1MOD.GT.PI) GO TO 188
                IF(X1MOD.LT.-PI) GO TO 189
                GO TO 190
              188 X1MOD=X1MOD-PI2
                GO TO 184
325           189 X1MOD=X1MOD+PI2
                GO TO 184
              19. CONTINUE
              222 IF (KOUNT.EQ.NO2) GO TO 505
                KOUNT=KOUNT + 1
330             GO TO 450
```

```
      207 FORMAT(*1*,1P6E15.6)
      505 CONTINUE
          IF (SPLST) WRITE (6,90303) NMC
     9000 FORMAT(*1*35X*MONTE CARLO SAMPLE PATH AVERAGES FOR PATH NO. *I5/
     1 *0*2X*N*8X*N_E*11X*AVNLE*10X*AVNLV*5X*NLCS*6X*PLLE*10X*AVPLLE*
     2 9X*AVPLLV*8X*AVCPLLV*9X*PLCS*//)
          CS=.75*PI
          CSN=0.
          CSP=0.
          DO 6200 I=1,NO2
          CALL MOD2PI(XDAT(I,1)-XDAT(I,2),ERRN)
          ICSNLF=NBL
          IF (ABS(ERRN).LE.CS) GO TO 6000
          ICSNLF=NAS
          CSN=CSN+1./NO2
 6000     XDATM(I,1)=AVG(XDATM(I,1),ERRN,NMC)
          XDATM(I,2)=AVG(XDATM(I,2),ERRN*ERRN,NMC)
          CALL MOD2PI(XDAT(I,1)-XDAT(I,4),ERRP)
          ICSPLL=NBL
          IF (ABS(ERRP).LE.CS) GO TO 6100
          ICSPLL=NAS
          CSP=CSP+1./NO2
 6100     XDATM(I,3)=AVG(XDATM(I,3),ERRP,NMC)
          XDATM(I,4)=AVG(XDATM(I,4),ERRP*ERRP,NMC)
          XDATM(I,5)=AVG(XDATM(I,5),XDAT(I,5),NMC)
          SMN=XDATM(I,1)**2
          SMP=XDATM(I,3)**2
          XDMSN=XDATM(I,2)-SMN
          XDMSP=XDATM(I,4)-SMP
          IF (SPLST) WRITE (6,90100) I,ERRN,XDATM(I,1),XDMSN,
         1 ICSNLF,ERRP,XDATM(I,3),XDMSP,          XDATM(I,5),ICSPLL
     9010 FORMAT(* *I5,3E15.7,2X,A1,2X,4E15.7,2X,A1)
 6200 CONTINUE
          IF (SPLST) WRITE (6,90200) CS, CSN, CSP
     9020 FORMAT(*0*29X*CYCLE SLIP FRACTIONS WITH THRESHOLD = *E15.7
         1 * RADIANS.*/30X*NONLINEAR -*E15.7*, PHASE-LOCKED LOOP -*E15.7//)
          XMCCSN=XMCCSN+CSN
          XMCPLL=XMCPLL+CSP
          XMCPN=0.
          XMCPP=0.
          XMCCN=0.
          XMCCP=0.
          XMCCVP=0.
          H=NO2-30
          DO 6300 I=31,NO2
          XMCPN=XMCPN+XDATM(I,1)
          XMCPP=XMCPP+XDATM(I,3)
 6300     XMCCVP=XMCCVP+XDATM(I,5)
          XMCPN=XMCPN/H
          XMCPP=XMCPP/H
          XMCCVP=XMCCVP/H
          SMN=XMCPN*XMCPN
          SMP=XMCPP*XMCPP
          DO 6400 I=31,NO2
          XMCCN=XMCCN+XDATM(1,2)
```

```
                5400 XMCCP=XMCCP+XDATM(I,4)
                     XMCCN=XMCCN/M-SMN
                     XMCCP=XMCCP/M-SMP
                     ALXN=10.*ALOG10(XMCCN)
      390            ALXP=10.*ALOG10(XMCCP)
                     WRITE (6,9030) NMC,XMCPN,XMCCN,XMCPP,XMCCP,XMCCVP,ALXN,ALXP
               9030  FORMAT(*0MONTE CARLO NO.*I5,10X*ENLF*12X*VNLF*12X*EPLL*12X
                    1 *VPLL*12X*CVPLL*/* CUMULATIVE TIME AVERAGES*5E15.7/* DECIBELS*
                    2 31X,E15.7,15X,E15.7)
      395            NMC=NMC+1
                     IF (NMC.GT.JSEED) STOP
                     GO TO 9000
                     END
```

SUMMARY OF CHANGES MADE BY THE OPTIMIZER

```
  27 WORDS OF INVARIANT RLIST REMOVED FROM THE LOOP STARTING AT LINE   121
  13 WORDS OF INVARIANT RLIST REMOVED FROM THE LOOP STARTING AT LINE   126
  28 WORDS OF INVARIANT RLIST REMOVED FROM THE LOOP STARTING AT LINE   137
  14 WORDS OF INVARIANT RLIST REMOVED FROM THE LOOP STARTING AT LINE   204
```

REGISTER ALLOCATION

```
  4 REGISTERS ASSIGNED OVER THE LOOP BEGINNING AT LINE    92
  4 REGISTERS ASSIGNED OVER THE LOOP BEGINNING AT LINE    96
  4 REGISTERS ASSIGNED OVER THE LOOP BEGINNING AT LINE   100
  1 REGISTERS ASSIGNED OVER THE LOOP BEGINNING AT LINE   126
  1 REGISTERS ASSIGNED OVER THE LOOP BEGINNING AT LINE   142
  3 REGISTERS ASSIGNED OVER THE LOOP BEGINNING AT LINE   220
  3 REGISTERS ASSIGNED OVER THE LOOP BEGINNING AT LINE   229
  3 REGISTERS ASSIGNED OVER THE LOOP BEGINNING AT LINE   236
  3 REGISTERS ASSIGNED OVER THE LOOP BEGINNING AT LINE   245
  1 REGISTERS ASSIGNED OVER THE LOOP BEGINNING AT LINE   252
  2 REGISTERS ASSIGNED OVER THE LOOP BEGINNING AT LINE   269
  3 REGISTERS ASSIGNED OVER THE LOOP BEGINNING AT LINE   275
  2 REGISTERS ASSIGNED OVER THE LOOP BEGINNING AT LINE   384
```

SYMBOLIC REFERENCE MAP

ENTRY POINTS DEF LINE REFERENCES
4052 MAIN 1

VARIABLES SN TYPE RELOCATION

6173 A REAL
REFS 84 DEFINED 83
6210 AI REAL
REFS 150 DEFINED 147
6211 AJ REAL
REFS 150 DEFINED 149
6212 AK REAL
REFS 151 152 155 DEFINED 150
6125 ALP110 REAL
REFS 26 33 35 DEFINED 25
6265 ALXN REAL
REFS 391 DEFINED 389
6266 ALXP REAL
REFS 391 DEFINED 390
33164 AN REAL *UNDEF
REFS 12
6143 A11 REAL
REFS 73 138 163 DEFINED 33
6144 A22 REAL
REFS 74 138 164 DEFINED 34
6241 CHAT REAL
REFS 277 279 286 283 DEFINED 273 277
6245 COSF1 REAL
REFS 293 294 DEFINED 292
21002 COSY REAL ARRAY
REFS 7 205 277 DEFINED 114
6230 CS REAL
REFS 178 179 312 313 343 353 364
DEFINED 175 337
6250 CSN REAL
REFS 345 364 367 DEFINED 338 345
6251 CSP REAL
REFS 352 364 368 DEFINED 339 352
6242 CXHAT REAL
REFS 281 282 284 DEFINED 279 281 282
6126 DELF REAL
REFS 30 DEFINED 20
6157 DELSQ REAL
REFS 2*192 DEFINED 59
6140 DELT REAL
REFS 31 32 35 59 62 80 83
186 192 2*194 306 DEFINED 30
6172 DEN REAL
REFS 192 194 195 286 287 289 293
294 DEFINED 78 206
0 DENS1 REAL ARRAY / /
REFS 2 139 143 183 2*224 2*233 2*240
2*249 2*256 276 277 317 DEFINED 138 143
274
4235 DENS2 REAL ARRAY / /
REFS 2 270 DEFINED 265
6225 DEVQ2 REAL
REFS 188 DEFINED 171
6216 DEV1 REAL
REFS 166 DEFINED 163
6217 DEV2 REAL
REFS 168 DEFINED 164
6220 DEV3 REAL
REFS 169 170 189 190 DEFINED 165
6236 DUM REAL
REFS 224 233 240 249 256 261
264 265 DEFINED 217 224 233 240 249
256 251 263
0 DZZZ1 REAL GN
REFS 5
6176 EDG1 REAL
REFS 95 DEFINED 90
6177 EDG2 REAL
REFS 99 DEFINED 91
33202 EIGVEC REAL *UNDEF
REFS 12
6231 ENL REAL
REFS 178 312 DEFINED 176 310
6232 EPL REAL
REFS 179 313 DEFINED 177 311
6252 ERRN REAL
REFS 341 343 346 2*347 360
6254 ERRP REAL
REFS 348 350 353 2*354 360
21161 EXDON REAL ARRAY
REFS 7 125 127 128 DEFINED 120 125
21123 EXPON REAL ARRAY
REFS 7 133 224 233 240 249 256
DEFINED 127 128 131
33166 F REAL ARRAY
REFS 12 DEFINED 79 80 81 82
6137 FTC REAL
REFS 33 2*34 35 DEFINED 29

VARIABLES	SN	TYPE	RELOCATION							
6264	H	REAL		REFS	379	380	381	387	388	
				DEFINED	374					
6154	I	INTEGER		REFS	55	93	95	97	99	101	103
					106	108	2*114	2*115	120	3*125	2*127	2*128
					133	2*138	139	2*143	147	155	156	157
					158	159	160	161	181	3*183	3*205	222
					223	231	232	238	239	247	248	254
					259	261	265	2*270	2*276	2*277	315	3*317
					2*341	2*346	2*347	2*348	2*353	2*354	3*355	356
					357	358	359	4*360	376	377	378	385
					386	DEFINED	53	92	96	100	106	108
					113	119	126	133	136	141	146	180
					204	215	268	274	314	340	375	384
6174	IA	INTEGER		REFS	86	117	122	DEFINED	84			
6201	IB	INTEGER		REFS	121	DEFINED	117					
6203	IC	INTEGER		REFS	123	DEFINED	122					
6253	ICSNLF	INTEGER		REFS	368	DEFINED	342	344				
6255	ICSPLL	INTEGER		REFS	363	DEFINED	349	351				
6202	IR	INTEGER		REFS	122	DEFINED	121					
6204	IYY	INTEGER		REFS	133	DEFINED	132					
6175	IY2	INTEGER		REFS	86	100	118	119	126	127	128	
					131	132	209	210	DEFINED	85		
6155	J	INTEGER		REFS	55	2*138	139	2*143	149	155	156	
					157	158	159	160	161	183	209	210
					265	2*270	276	277	317	DEFINED	54	137
					142	148	182	206	269	275	316	
1	JGAUSS	INTEGER		GN	REFS	5	DEFINED	19				
6133	JSEED	INTEGER			REFS	22	24	58	166	168	169	170
					188	189	190	396	DEFINED	20	24	
6213	KL	INTEGER			REFS	152	153	154	156	2*157	2*158	3*224
					3*233	3*240	3*249	3*256	DEFINED	151	152	222
					231	238	247	254				
14472	KLOW	INTEGER	ARRAY / /		REFS	2	222	231	238	247	254	
					DEFINED	156	157	158				
6156	KOUNT	INTEGER			REFS	187	187	284	285	308	309	2*310
					2*311	328	329	DEFINED	56	329		
6214	KU	INTEGER			REFS	159	2*160	2*161	224	233	240	249
					256	DEFINED	153	223	232	239	248	255
14727	KUP	INTEGER	ARRAY / /		REFS	2	223	232	239	248	255	
					DEFINED	159	160	161				
6233	K1	INTEGER			REFS	211	213	220	229	245		
					DEFINED	209	213					
6234	K2	INTEGER			REFS	212	214	220	236	252		
					DEFINED	210	214					
6237	L	INTEGER			REFS	222	223	4*224	231	232	4*233	238
					239	4*240	247	248	4*249	254	255	4*256
					DEFINED	220	229	236	245	252		
6117	LOW	LOGICAL			REFS	15	213	218	DEFINED	207	211	
6235	MM	INTEGER			REFS	221	224	230	233	237	240	246
					249	253	256	DEFINED	216	221	230	237
					246	253						
5725	NAS	INTEGER			REFS	344	351	DEFINED	17			
5724	NBL	INTEGER			REFS	342	349	DEFINED	17			
6153	NMC	INTEGER			REFS	58	86	104	333	346	347	353

VARIABLES	SN	TYPE		RELOCATION								
6132	NO2	INTEGER			354	355	391	395	396	DEFINED	52	395
					REFS	53	328	340	345	352	374	375
					384	DEFINED	20					
6130	NUM1	INTEGER			REFS	65	92	106	113	136	141	146
					157	2*158	160	2*161	180	204	215	268
					274	314	DEFINED	20				
6131	NUM2	INTEGER			REFS	66	96	108	137	142	148	182
					206	212	213	214	229	245	269	275
					316	DEFINED	20					
6226	NUNNL	INTEGER			REFS	178	312	DEFINED	173	178	312	
6227	NUNPL	INTEGER			REFS	179	313	DEFINED	174	179	313	
0	NZZZ	INTEGER	ARRAY	RN	REFS	4						
33154	PBAR	REAL	*UNDEF		REFS	12						
33172	POUMY	REAL	ARRAY		REFS	12	196	197	198	DEFINED	192	194
					195							
33176	POUMY2	REAL	*UNDEF		REFS	12						
6160	PI	REAL			REFS	61	62	63	90	95	175	281
					282	297	298	320	321	337		
					DEFINED	60						
6162	PIDLT	REAL			REFS	64	91	99	DEFINED	62		
6163	PINV	REAL			REFS	83	DEFINED	63				
6161	PI2	REAL			REFS	95	300	302	323	325		
					DEFINED	61						
6164	PI2DLT	REAL			REFS	85	99	103	124	DEFINED	64	
6243	PLOSS	REAL			REFS	285	DEFINED	283				
33160	PN	REAL	ARRAY		REFS	12	78	4*192	3*194	2*195	199	266
					286	287	2*289	2*293	2*294	DEFINED	73	74
					75	76	196	197	198	199		
33206	PNF	REAL	ARRAY		REFS	12	288	309	DEFINED	286	287	288
					289							
21164	PROP	REAL	ARRAY	/ /	REFS	2	2*224	2*233	2*240	2*249	2*256	
					DEFINED	155						
6134	P110	REAL			REFS	28	34	35	DEFINED	26		
33150	Q	REAL	ARRAY		REFS	12	195	DEFINED	67	68		
6135	QQ	REAL			REFS	28	29	DEFINED	27			
6141	Q22	REAL			REFS	35	68	83	85	125	171	
					DEFINED	31						
6127	Q22C	REAL			REFS	27	31	36	DEFINED	20		
6171	R	REAL			REFS	78	124	125	192	194	200	286
					287	DEFINED	77	123	124	172		
6207	RATIO	REAL			REFS	2*150	DEFINED	145				
6215	REALK	REAL			REFS	155	DEFINED	154				
6136	RX	REAL			REFS	29	32	36	DEFINED	28		
6142	R11	REAL			REFS	35	77	165	172	20		
					DEFINED	32						
33027	SENS	REAL	ARRAY		REFS	11	261	DEFINED	265			
6240	SHAT	REAL			REFS	276	279	281	282	283		
					DEFINED	272	276					
6244	SINF1	REAL			REFS	293	294	DEFINED	291			
20661	SINY	REAL	ARRAY		REFS	7	205	276	DEFINED	115		
6256	SMN	REAL			REFS	358	387	DEFINED	356	382		
6257	SMP	REAL			REFS	359	388	DEFINED	357	383		
6122	SPL	LOGICAL			REFS	16	58	DEFINED	21	23		
6121	SPLST	LOGICAL			REFS	16	333	360	364	DEFINED	57	58

PROGRAM MAIN

VARIABLES	SN	TYPE	RE-LOCATION		
6205	SUM	REAL			
6206	SUMI	REAL			
6120	UP	LOGICAL			
6165	U1	REAL			
6166	U2	REAL			
6200	X	REAL			
6267	XDAT	REAL	ARRAY		
21200	XDATM	REAL	ARRAY		
33212	XDEN	REAL	ARRAY		
6260	XDMSN	REAL			
6261	XDMSP	REAL			
20077	XHAT	REAL	ARRAY		
6167	XHAT1	REAL			
6170	XHAT2	REAL			
6146	XMCGN	REAL			
6151	XMCGP	REAL			
6147	XMCGSN	REAL			
6152	XMCGSP	REAL			
6263	XMCGVP	REAL			
6262	XMCPLL	REAL			
6145	XMCPN	REAL			
6150	XMCPP	REAL			
3	XNZ	REAL	ARRAY	RN	
2	XZZZ	REAL	ARRAY	GN	
6221	X1	REAL			
6246	X1FF	REAL			
6247	X1MOD	REAL			
6222	X2	REAL			
20161	Y1	REAL	ARRAY		
6123	Y1EST	REAL			
20222	Y2	REAL	ARRAY		
6124	Y2EST	REAL			
33010	Y3	REAL	ARRAY		
6223	Z1	REAL			
6224	Z2	REAL			

REFS	139	140	264	267	DEFINED	135	139
203	264						
REFS	143	270	DEFINED	140	267		
REFS	15	214	219	DEFINED	208	212	
REFS	93	94	145	150	DEFINED	65	
REFS	85	91	98	102	145		
DEFINED	66						
REFS	94	95	98	99	102	103	
DEFINED	93	94	97	98	101	102	
REFS	6	2*176	2*177	2*310	2*311	2*341	2*348
355	DEFINED	167	187	284	285	308	309
REFS	9	346	347	353	354	355	356
357	358	359	3*360	376	377	378	385
386	DEFINED	55	346	347	353	354	355
REFS	14	183	317	DEFINED	181	183	315
317							
REFS	360	DEFINED	358				
REFS	360	DEFINED	359				
REFS	6	295	2*306	307	308		
DEFINED	69	70	293	294			
REFS	291	292	293	DEFINED	71	306	
REFS	294	DEFINED	72	307			
REFS	385	387	389	391	DEFINED	47	371
385	387						
REFS	386	388	390	391	DEFINED	50	372
386	388						
REFS	367	DEFINED	48	367			
DEFINED	51						
REFS	378	381	391	DEFINED	373	378	381
REFS	368	DEFINED	368				
REFS	376	379	2*382	391	DEFINED	46	369
376	379						
REFS	377	380	2*383	391	DEFINED	49	370
377	380						
REFS	4						
REFS	5						
REFS	166	167	169	170	186	187	189
190	318	DEFINED	186				
REFS	297	298	300	302	DEFINED	295	300
302							
REFS	320	321	323	325	DEFINED	318	323
325							
REFS	168	186	2*188				
REFS	7	106	114	115	138		
DEFINED	95						
REFS	69	71	166	DEFINED	20		
REFS	7	108	138	DEFINED	99		
REFS	70	72	168	DEFINED	20		
REFS	10	125	DEFINED	103			
REFS	169	189	205	293	294		
REFS	170	190	205	293	294		

FILE NAMES		MODE									
0	INPUT										
2022	OUTPUT										
0	TAPE5	FMT		READS	20						
2022	TAPE6	FMT		WRITES	35	36	86	105	106	107	108	133
					333	360	364	391				

EXTERNALS	TYPE	ARGS		REFERENCES							
ALOG10	REAL	1	LIBRARY	389	390						
ATAN	REAL	1	LIBRARY	279							
AVG	REAL	3		346	347	353	354	355			
COS	REAL	1	LIBRARY	114	169	189	292				
EXP	REAL	1	LIBRARY	125	138	205					
GAUSS		4		166	168	169	170	188	189	190	
MOD2PI		2		341	348						
SIN	REAL	1	LIBRARY	115	170	190	291				
SQRT	REAL	1	LIBRARY	28	29	83	85	163	164	165	171	283

INLINE FUNCTIONS	TYPE	ARGS	DEF LINE	REFERENCES					
ABS	REAL	1	INTRIN	176	177	310	311	343	350

STATEMENT LABELS			DEF LINE	REFERENCES	
0	2	INACTIVE	18		
0	40		95	92	
0	50		99	96	
0	51		103	100	
0	55		115	113	
5266	84		296	301	303
5273	88		300	297	
5276	89		302	298	
5301	90		304	299	
0	150		125	119	121
0	152		129	126	
4445	153		130	118	
0	160		139	136	137
0	165		143	141	142
5340	184		319	324	326
5345	188		323	320	
5350	189		325	321	
5353	190		327	322	
6013	207	FMT NO REFS	331		
0	222	INACTIVE	328		
0	250		162	146	148
0	285		205	204	
0	300		226	220	
5007	301		228	218	
0	303		235	229	
0	304		242	236	
5063	305		244	219	
0	306		251	245	
0	307		258	252	
5143	308		263	260	
5144	309		264	262	
0	310		266	206	215
0	312		271	268	269

STATEMENT LABELS			DEF LINE	REFERENCES			
5136	315		259	227	243		
0	360		278	274	275		
5231	367		283	280			
4653	450		185	330			
4714	470		201	184			
5360	505		332	328			
5740	651	FMT	39	20			
5743	652	FMT	40	35			
5726	653	FMT	37	36			
5770	901	FMT	87	86			
6002	912	FMT	110	106	108	133	
6005	913	FMT	111	105			
6010	914	FMT	112	107			
0	5656		183	180	182		
0	5676		317	314	316		
5410	6000		346	343			
5436	6100		353	350			
0	6200		363	360			
0	6300		378	375			
0	6400		386	384			
4100	6666		25	22			
0	8999		55	53	54		
4203	9000		56	397			
4366	9689		109	104			
6016	90000	FMT	334	333			
6037	90100	FMT	362	360			
6044	90200	FMT	365	364			
6061	90300	FMT	392	391			

LOOPS	LABEL		INDEX	FROM-TO	LENGTH	PROPERTIES	
4175	8999	*	I	53 55	68	NOT INNER	
4177	8999		J	54 55	28	INSTACK	
4305	40	*	I	92 95	58	INSTACK	
4322	50	*	I	96 99	58	INSTACK	
4337	51	*	I	100 103	38	INSTACK	
4367	55	*	I	113 115	78	EXT REFS	
4405	150	*	I	119 125	248	EXT REFS	NOT INNER
4413	150	*	IR	121 125	148	EXT REFS	
4437	152	*	I	126 129	58	INSTACK	
4461	160	*	I	136 139	248	EXT REFS	NOT INNER
4466	160	*	J	137 139	158	EXT REFS	
4510	165	*	I	141 143	118	NOT INNER	
4515	165		J	142 143	28	INSTACK	
4524	250	*	I	146 162	568	NOT INNER	
4534	250	*	J	148 162	368	OPT	
4640	5656	*	I	180 183	138	NOT INNER	
4646	5656		J	182 183	28	INSTACK	
4720	265	*	I	204 205	118	EXT REFS	
4732	310	*	J	216 266	2248	NOT INNER	
4754	310	*	I	215 266	2208	NOT INNER	
4772	300		L	220 226	138	OPT	
5021	303		L	229 235	138	OPT	
5046	304		L	236 242	138	OPT	
5075	306		L	245 251	138	OPT	

LOOPS	LABEL		INDEX	FROM-TO	LENGTH	PROPERTIES
5122	307		L	252 258	138	OPT
5161	312	*	I	268 271	128	NOT INNER
5167	312		J	269 271	28	INSTACK
5176	360	*	I	274 278	168	NOT INNER
5205	360		J	275 278	38	INSTACK
5324	5676	*	I	314 317	138	NOT INNER
5332	5676		J	316 317	28	INSTACK
5373	6200	*	I	340 363	1228	EXT REFS
5545	6360		I	375 378	48	INSTACK
5565	6400		I	384 386	38	INSTACK

COMMON BLOCKS	LENGTH	MEMBERS - BIAS NAME(LENGTH)							
/ /	11025	0	DENS1	(2205)	2205	DENS2	(2205)	4410	KLOW (2205)
		6615	KUP	(2205)	9820	PROP	(2205)		
RN	5	0	NZZZ	(3)	3	XNZ	(2)		
GN	4	0	DZZZ1	(1)	1	JGAUSS	(1)	2	XZZZ (2)

STATISTICS

PROGRAM LENGTH	271758	11901
BUFFER LENGTH	40448	2084
COMMON LENGTH	118	9
BLANK COMMON	254218	11025

```
      SUBROUTINE MOD2PI(X,XM)
      DATA ISW/0/
      IF (ISW) 20, 10, 20
   10 PI=4.*ATAN(1.)
      TWOPI=2.*PI
      ISW=999
   20 XM=X
   30 IF (XM.GT.PI) GO TO 40
      IF (XM.LT.-PI) GO TO 50
      RETURN
   40 XM=XM-TWOPI
      GO TO 30
   50 XM=XM+TWOPI
      GO TO 30
      END
```

SUBROUTINE MOD2PI

CDC 6600 FTN V3.0-P213 OPT=2 03/22/72 16.47.23. PAGE 2

SYMBOLIC REFERENCE MAP

ENTRY POINTS DEF LINE REFERENCES
2 MOD2PI 1 10

VARIABLES SN TYPE RELOCATION

31	ISW	INTEGER		REFS	3	DEFINED	2	6				
35	PI	REAL		REFS	5	8	9	DEFINED	4			
36	TWOPI	REAL		REFS	11	13	DEFINED	5				
0	X	REAL	F.P.	REFS	7	DEFINED	1					
6	XM	REAL	F.P.	REFS	8	9	11	13	DEFINED	1	7	
				11	13							

EXTERNALS TYPE ARGS REFERENCES
ATAN REAL 1 LIBRARY 4

STATEMENT LABELS DEF LINE REFERENCES

0	10	INACTIVE	4	3
12	20		7	2*3
14	30		8	12 14
23	40		11	8
26	50		13	9

STATISTICS
PROGRAM LENGTH 378 31

FUNCTION AVG

CDC 6600 FTN V3.0-P213 OPT=2 03/22/72 16.47.23. 1

```
      FUNCTION AVG(AV,X,I)
      AVG=((I-1.)*AV+X)/I
      RETURN
      END
```

FUNCTION AVG CDC 6600 FTN V3.0-P213 OPT=2 03/22/72 16.47.23. PAGE 2

SYMBOLIC REFERENCE MAP

ENTRY POINTS		DEF LINE	REFERENCES
2	AVG	1	3

VARIABLES	SN	TYPE	RELOCATION				
0	AV	REAL	F.P.	REFS	2	DEFINED	1
14	AVG	REAL		DEFINED	2		
0	I	INTEGER	F.P.	REFS	2*2	DEFINED	1
0	X	REAL	F.P.	REFS	2	DEFINED	

STATISTICS

PROGRAM LENGTH	15B	13

```
      FUNCTION RANF(NS,MODE)
      DIMENSION  NS(2), NC(2)
      COMMON /RN/ N1, N2, MP, T1, T2
      DATA  M1, M2/244734, 158551/
C        MODE=0 TO CONTINUE, OTHERWISE RESTART WITH
C        INTEGER NUMBER NS(1)*2**18+NS(2)
      IF (MODE) 10, 100, 10
   10 N1=NS(1)
      N2=NS(2)
      T1=2.**(-18)
      T2=2.**(-36)
      MP=2**18
  100 DO 200 I=1,2
      GO TO (110,120),I
  110 K=M2*N2
      GO TO 190
  120 K=M1*N2+M2*N1+KD
  190 KD=K/MP
  200 NC(I)=K-KD*MP
      N1=NC(2)
      N2=NC(1)
      XN1=N1
      XN2=N2
      RANF=XN1*T1+XN2*T2
      RETURN
      END
```

SYMBOLIC REFERENCE MAP

ENTRY POINTS DEF LINE REFERENCES
2 RANF 1 25

VARIABLES SN TYPE RELOCATION

76	I	INTEGER			REFS	14	19	DEFINED	13			
77	K	INTEGER			REFS	18	19	DEFINED	15	17		
100	KD	INTEGER			REFS	17	19	DEFINED	18			
0	MODE	INTEGER		F.P.	REFS	7	DEFINED	1				
2	MP	INTEGER		RN	REFS	3	18	19	DEFINED	12		
72	M1	INTEGER			REFS	17	DEFINED	4				
73	M2	INTEGER			REFS	15	17	DEFINED	4			
103	NC	INTEGER	ARRAY		REFS	2	20	21	DEFINED	19		
0	NS	INTEGER	ARRAY	F.P.	REFS	2	8	9	DEFINED	1		
0	N1	INTEGER		RN	REFS	3	17	22	DEFINED	8	20	
1	N2	INTEGER		RN	REFS	3	15	17	23	DEFINED	9	21
75	RANF	REAL			DEFINED	24						
3	T1	REAL		RN	REFS	3	24	DEFINED	10			
4	T2	REAL		RN	REFS	3	24	DEFINED	11			
101	XN1	REAL			REFS	24	DEFINED	22				
102	XN2	REAL			REFS	24	DEFINED	23				

STATEMENT LABELS DEF LINE REFERENCES

0	10	INACTIVE	8	2*7
27	100		13	7
41	110		15	14
44	120		17	14
51	190		18	16
0	200		19	13

LOOPS LABEL INDEX FROM-TO LENGTH PROPERTIES
33 200 * I 13 19 258 OPT

COMMON BLOCKS LENGTH MEMBERS - BIAS NAME(LENGTH)
RN 5 0 N1 (1) 1 N2 (1) 2 MP (1)
3 T1 (1) 4 T2 (1)

STATISTICS

PROGRAM LENGTH	1128	74
COMMON LENGTH	58	5

```
      SUBROUTINE GAUSS(JS,SD,XM,X)
      DIMENSION  NST(2)
      COMMON /RN/ N1, N2, MC, T1, T2
      COMMON /GN/ TWOPI, J, XR(2)
      IF (J) 10, 15, 20
   10 J=2
      TWOPI=8.*ATAN(1.)
      NST(1)=244734
      NST(2)=158551
      NST(1)=102943
      NST(2)=185617
      XR(1)=RANF(NST,1)
      GO TO 35
   20 GO TO (30,40), J
   30 J=2
      XR(1)=RANF(NST,J)
   35 XR(2)=RANF(NST,0)
      X1=SQRT(ABS(-2.*ALOG(XR(1))))
      XR(2)=TWOPI*XR(2)
      XR(1)=X1*SIN(XR(2))
      XR(2)=X1*COS(XR(2))
      X=XR(1)*SD+XM
      RETURN
   40 J=1
      X=XR(2)*SD+XM
      RETURN
      END
```

SYMBOLIC REFERENCE MAP

ENTRY POINTS		DEF LINE	REFERENCES									
2	GAUSS	1	23	26								

VARIABLES	SN	TYPE		RELOCATION									
1	J	INTEGER		GN	REFS	4	5	14	DEFINED	6	15	24	
0	JS	INTEGER	*UNUSED	F.P.	DEFINED	1							
2	MC	INTEGER		RN	REFS	3							
100	NST	INTEGER	ARRAY		REFS	2	12	16	17	DEFINED	8	9	
						10	11						
0	N1	INTEGER		RN	REFS	3							
1	N2	INTEGER		RN	REFS	3							
0	SD	REAL		F.P.	REFS	22	25	DEFINED	1				
0	TWOPI	REAL		GN	REFS	4	19	DEFINED	7				
3	T1	REAL		RN	REFS	3							
4	T2	REAL		RN	REFS	3							
0	X	REAL		F.P.	DEFINED	1	22	25					
0	XM	REAL		F.P.	REFS	22	25	DEFINED	1				
2	XR	REAL	ARRAY	GN	REFS	4	18	19	20	21	22	25	
					DEFINED	12	16	17	19	20	21		
77	X1	REAL			REFS	20	21	DEFINED	18				

EXTERNALS	TYPE	ARGS		REFERENCES			
ALOG	REAL	1	LIBRARY	18			
ATAN	REAL	1	LIBRARY	7			
COS	REAL	1	LIBRARY	21			
RANF	REAL	2		12	16	17	
SIN	REAL	1	LIBRARY	20			
SQRT	REAL	1	LIBRARY	18			

INLINE FUNCTIONS	TYPE	ARGS		DEF LINE	REFERENCES
ABS	REAL	1	INTRIN		18

STATEMENT LABELS			DEF LINE	REFERENCES
0	10	INACTIVE	6	2*5
16	20		14	5
24	30		15	14
30	35		17	13
53	40		24	14

COMMON BLOCKS	LENGTH	MEMBERS - BIAS	NAME	(LENGTH)							
RN	5	0	N1	(1)	1	N2	(1)	2	MC	(1)	
		3	T1	(1)	4	T2	(1)				
GN	4	3	TWOPI	(1)	1	J	(1)	2	XR	(2)	

STATISTICS

PROGRAM LENGTH	128	66
COMMON LENGTH	118	9

```
EQUILIBRIUM CONTINUOUS POSITION VARIANCE =  5.011872E-01 ( -3.000000E+00 DB)
FILTER TIME CONSTANT =  4.645260E+00
DELTA =  4.645260E-01
DISCRETE DRIVING VARIANCE =  4.645260E-03
DISCRETE NOISE VARIANCE =  2.505936E+00

CONTINUOUS DRIVING VARIANCE =  1.000000E-02
CONTINUOUS NOISE VARIANCE =  1.164072E+00

          NUMBER OF CYCLES ON EACH SIDE =   0      NUMBER OF POINTS ON EACH SIDE =   4

Y1 GRID
 -2.9920E+00 -2.6928E+00 -2.3936E+00 -2.0944E+00 -1.7952E+00 -1.4960E+00 -1.1968E+00 -8.9760E-01 -5.9840E-01 -2.9920E-01
 -2.4869E-14  2.9920E-01  5.9840E-01  8.9760E-01  1.1968E+00  1.4960E+00  1.7952E+00  2.0944E+00  2.3936E+00  2.6928E+00
  2.9920E+00

Y2 GRID
 -6.6986E+00 -6.5698E+00 -6.4410E+00 -6.3121E+00 -6.1833E+00 -6.0545E+00 -5.9257E+00 -5.7969E+00 -5.6680E+00 -5.5392E+00
 -5.4104E+00 -5.2816E+00 -5.1528E+00 -5.0239E+00 -4.8951E+00 -4.7663E+00 -4.6375E+00 -4.5087E+00 -4.3799E+00 -4.2510E+00
 -4.1222E+00 -3.9934E+00 -3.8646E+00 -3.7358E+00 -3.6069E+00 -3.4781E+00 -3.3493E+00 -3.2205E+00 -3.0917E+00 -2.9628E+00
 -2.8340E+00 -2.7052E+00 -2.5764E+00 -2.4476E+00 -2.3187E+00 -2.1899E+00 -2.0611E+00 -1.9323E+00 -1.8035E+00 -1.6746E+00
 -1.5458E+00 -1.4170E+00 -1.2882E+00 -1.1594E+00 -1.0306E+00 -9.0173E-01 -7.7292E-01 -6.4410E-01 -5.1528E-01 -3.8646E-01
 -2.5764E-01 -1.2882E-01 -5.1514E-14  1.2882E-01  2.5764E-01  3.8646E-01  5.1528E-01  6.4410E-01  7.7292E-01  9.0173E-01
  1.0306E+00  1.1594E+00  1.2882E+00  1.4170E+00  1.5458E+00  1.6746E+00  1.8035E+00  1.9323E+00  2.0611E+00  2.1899E+00
  2.3187E+00  2.4476E+00  2.5764E+00  2.7052E+00  2.8340E+00  2.9628E+00  3.0917E+00  3.2205E+00  3.3493E+00  3.4781E+00
  3.6069E+00  3.7358E+00  3.8646E+00  3.9934E+00  4.1222E+00  4.2510E+00  4.3799E+00  4.5087E+00  4.6375E+00  4.7663E+00
  4.8951E+00  5.0239E+00  5.1528E+00  5.2816E+00  5.4104E+00  5.5392E+00  5.6680E+00  5.7969E+00  5.9257E+00  6.0545E+00
  6.1833E+00  6.3121E+00  6.4410E+00  6.5698E+00  6.6986E+00
  3.8767E-13  1.0435E-07  7.8907E-04  1.6760E-01  1.0000E+00  1.6760E-01  7.8907E-04  1.0435E-07  3.8767E-13

MONTE CARLO NO.    1          ENLF              VNLF              EPLL              VPLL              CVPLL
CUMULATIVE TIME AVERAGES  -.1614760E+00   .4776409E+00  -.1578502E+00   .5175799E+00   .4545867E+00
DECIBELS                                  -.3208985E+01                   -.2860226E+01
  3.8767E-13  1.0435E-07  7.8907E-04  1.6760E-01  1.0000E+00  1.6760E-01  7.8907E-04  1.0435E-07  3.8767E-13

MONTE CARLO NO.    2          ENLF              VNLF              EPLL              VPLL              CVPLL
CUMULATIVE TIME AVERAGES  -.3952044E+00   .5823950E+00  -.3791377E+00   .6228713E+00   .4545867E+00
DECIBELS                                  -.2347824E+01                   -.2056017E+01
  3.8767E-13  1.0435E-07  7.8907E-04  1.6760E-01  1.0000E+00  1.6760E-01  7.8907E-04  1.0435E-07  3.8767E-13

MONTE CARLO NO.    3          ENLF              VNLF              EPLL              VPLL              CVPLL
CUMULATIVE TIME AVERAGES  -.4265772E+00   .5794005E+00  -.4015112E+00   .6516728E+00   .4545867E+00
DECIBELS                                  -.2370211E+01                   -.1859704E+01
  3.8767E-13  1.0435E-07  7.8907E-04  1.6760E-01  1.0000E+00  1.6760E-01  7.8907E-04  1.0435E-07  3.8767E-13

MONTE CARLO NO.    4          ENLF              VNLF              EPLL              VPLL              CVPLL
CUMULATIVE TIME AVERAGES  -.2799451E+00   .9129866E+00  -.2507063E+00   .1247824E+01   .4545867E+00
DECIBELS                                  -.3953561E+00                    .9615348E+00
  3.8767E-13  1.0435E-07  7.8907E-04  1.6760E-01  1.0000E+00  1.6760E-01  7.8907E-04  1.0435E-07  3.8767E-13

MONTE CARLO NO.    5          ENLF              VNLF              EPLL              VPLL              CVPLL
CUMULATIVE TIME AVERAGES  -.3158369E+00   .8119111E+00  -.3308181E+00   .1128453E+01   .4545867E+00
DECIBELS                                  -.9049152E+00                    .5248344E+00
  3.8767E-13  1.0435E-07  7.8907E-04  1.6760E-01  1.0000E+00  1.6760E-01  7.8907E-04  1.0435E-07  3.8767E-13

MONTE CARLO NO.    6          ENLF              VNLF              EPLL              VPLL              CVPLL
CUMULATIVE TIME AVERAGES  -.2447678E+00   .7583713E+00  -.2688081E+00   .1014110E+01   .4545867E+00
DECIBELS                                  -.1201181E+01                    .6084969E-01
  3.8767E-13  1.0435E-07  7.8907E-04  1.6760E-01  1.0000E+00  1.6760E-01  7.8907E-04  1.0435E-07  3.8767E-13
```

Additional Appendix E.

A Fourier Series Implementation
of the Cyclic Phase Demodulator

The Fourier Series experiment, described in Appendix D of Chapter VII is included here.

The program was compiled on one of the Kirtland AFB CDC 6600's using the FTN compiler. The code-optimization algorithm was employed. The random number generator is identical to the one described in Chapter IV.

Programmer: C. Hecht

Preceding page blank

```
      PROGRAM MAIN(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)
      COMPLEX SSK( 82),AA1(41,41),AA2(41,41)
      COMPLEX CMPX,CMPX2,CMPX3,CMPX4
      DIMENSION BSF2(100),EMSL( 82)
      DIMENSION XDAT( 136,3)
      DIMENSION Q(2,2),PBAR(2,2),PN(2,2),AN(2),F(2,2),PDUMY(2,2),
     1 PDUMY2(2,2),XHAT(2),EIGVEC(2,2),X(2),PNF(2,2)
      COMMON /RN/ NZZZ(3), XNZ(2)
      COMMON /GN/ UZZZ1, JGAUSS, XZZZ(2)
    1 CONTINUE
      JGAUSS=0
      READ(5,651)Y1EST,Y2EST,ALP11 ,DELF,Q22C,NUM1,NO1 ,NO2,JSEED
      IF(EOF(5)) 1799,1791
 1791 CONTINUE
      P11C=10.**(ALP11C/10.)
      PI=3.141 926 36
      PI2=2.0*PI
      PDB=ALOG10(P11C)*10.
      QQ=Q22C**(.25)
      RX=(P11C/(SQRT(2.0)*QQ))**(4./3.)
      FTC=SQRT(2.0)* RX**(.25) /QQ
      DELT=DELF*FTC
      Q22=Q22C*DELT
      R11=RX/DELT
      P22C=P11C*SQRT(Q22C/RX)
      A11=    P11C
      A11=10.**((ALP11C+10.)/10.)
      A22=    P22C
      WRITE(6,652) P11C,PDB,FTC,DELT,Q22,R11
      WRITE(6,653) Q22C,RX
      WRITE(6,654)A11,A22
  654 FORMAT(* *,*VARIANCE OF INITIAL POSITION AND VELOC.*,1P2.16.6)
  653 FORMAT(* *,*CONTINUOUS DRIVING VARIANCE =*,1PE14.6/
     1* CONTINUOUS NOISE VARIANCE =*,1PE14.6)
  651 FORMAT(5E10.0,3I5,I10)
  652 FORMAT(*1*,*EQUILIBRIUM CONTINUOUS POSITION VARIANCE =*,1PE14.6,
     1 * = *, 1PE14.6,* DB* /
     1 * FILTER TIME CONSTANT =*                              ,1PE14.6/
     1 * DELTA =*                                             ,1PE14.6/
     1 * DISCRETE DRIVING VARIANCE =*                         ,1PE14.6/
     1 * DISCRETE NOISE VARIANCE =*                           ,1PE14.6)
      ISAMP=1
      NSAMP=3
      M=NO2 - 30
      SUMP1=0.
      SUMP2=0.
C)))))))))   NO1 IS NUMBER OF TERMS OF EXPANSION   ))))))))))))))))))
      NTERM=NO1*2
      NTERM1=NTERM+1
      NTERM2=NO1+1
      NT=NTERM1+NTERM2
      NTERM3=NO1*4+1
      DELSQ=DELT**2
      QEXP=-0.5*Q22*DELSQ
      DO 451 J=1,NO1
```

```
                AJ=J**2
                ARG=EXP(QEXP*AJ)
                EMSL(NTERM2+J)=ARG
                EMSL(NTERM2-J)=ARG
  60          451 CONTINUE
                EMSL(NTERM2)=1.0
                PC1=-.5*A11
                PC2=-.5*A22
                DO 452 I=1,NTERM1
  65            DO 452 J=1,NTERM1
                AI=(I-NTERM2)**2
                AJ=(J-NTERM2)**2
                XAA=EXP(PC1*AI + PC2*AJ)
                AA1(I,J)=CMPLX(XAA,0.0)
  70            AA2(I,J)=( .,).
              452 CONTINUE
                Q(1,1)=0.0
                Q(2,2)=Q22
                R=R11
  75          2 CONTINUE
                KOUNT=1
                XHAT(1)=Y1EST
                XHAT(2)=Y2EST
                XHAT1=Y1EST
  80            XHAT2=Y2EST
                PN(1,1)=A11
                PN(2,2)=A22
                PN(1,2)= .
                PN(2,1)= .
  85            DEN =1.0/(PN(1,1) + R)
                F(1,1)=1.0
                F(1,2)=DELT
                F(2,1)=0.
                F(2,2)=1.0
  90            DEV1= SQRT(A11)
                DEV2= SQRT(A22)
                DEV3= SQRT(R)
                CALL GAUSS(JSEED,DEV1,Y1EST,X(1))
                CALL GAUSS(JSEED,DEV2,Y2EST,X(2))
  95            XDAT(KOUNT,1)=X(1)
                CALL GAUSS(JSEED,DEV3,COS(X(1)),Z1)
                CALL GAUSS(JSEED,DEV3,SIN(X(1)),Z2)
                DEVQ1= SQRT(Q(1,1))
                DEVQ2= SQRT(Q(2,2))
 100            GO TO 70
              5, CONTINUE
                X(1) =X(1) + X(2)*DELT
                XDAT(KOUNT,1)=X(1)
                CALL GAUSS(JSEED,DEVQ2,X(2),X(2))
 105            CALL GAUSS(JSEED,DEV3,COS(X(1)),Z1)
                CALL GAUSS(JSEED,DEV3,SIN(X(1)),Z2)
C  ***  RICCATI EQUATION UPDATE  ********
                PDUMY(1,1)=(R*(PN(1,1)+2.0*PN(1,2)*DELT)-PN(1,2)**2*DELSQ)*DEN
               1  + PN(2,2)*DELSQ
 110            PDUMY(1,2)=PN(1,2)*(R-PN(1,2)*DELT)*DEN + PN(2,2)*DELT
```

Reproduced from best available copy.

```
      PDUMY(2,2)=-PN(1,2)**2*DEN + PN(2,2) + D(2,2)
      PN(1,1)=PDUMY(1,1)
      PN(1,2)=PDUMY(1,2)
      PN(2,2)=PDUMY(2,2)
      PN(2,1) = PN(1,2)
      DEN =1.0/(PN(1,1) + R)
    7 CONTINUE
      OBMAG=SQRT(Z1**2 + Z2**2)/R
      OBPH=ATAN2(Z2,Z1)
      CALL IO(OBMAG,BSF)
      CALL INUO(OBMAG,NTERM,BSF,BSF2)
      DO 53  K=1,NTERM
      AK=K
      CMPX3=CMPLX(0.,OBPH*AK)
      CMPX4=CMPLX(0.,-OBPH*AK)
      SSK(NTERM1-K)=CMPLX(BSF2(K),0.)*CEXP(CMPX3)
      SSK(K+NTERM1)=CMPLX(BSF2(K),0. )*CEXP(CMPX4)
   53 CONTINUE
      SSK(NTERM1)=CMPLX(BSF,0. )
      CMPX2=(0.,0.)
      DO 54  NU=1,NTERM1
      CMPX=SSK(-NU+ NTE)*AA1(NU,NU)
      CMPX2=CMPX2+CMPX
   54 CONTINUE
      CMPX3=CMPX2
      DO 56  L=1,NTERM1
      CMPX4=CMPLX(EMSL(L),0.)/CMPX3
      DO 56  M=1,NTERM1
      CMPX2=(0.,0.)
      DO 55  NU=1,NTERM1
      IND=L+NU-NTERM2
      IF(IND.LT.1.OR.IND.GT.NTERM1) GO TO 55
      CMPX=SSK(M-NU+NTERM1)*AA1(NU,IND)
      CMPX2=CMPX2+CMPX
   55 CONTINUE
      AA2(M,L)=CMPX4*CMPX2
   56 CONTINUE
 4122 FORMAT(* *,1P10E11.3)
      DO 57  M=1,NTERM1
      DO 57  L=1,NTERM1
      AA1(M,L)=AA2(M,L)
   57 CONTINUE
      N1=NTERM2-1
      N =NTERM2
      AA11=AIMAG(AA1(N1,N ))
      AA22=REAL(AA1(N1,N ))
      X1NL=ATAN2(AA11,AA22)
      XDAT(KOUNT,3)=X1NL
      PNF(1,1)=PN(1,1)*R*DEN
      PNF(1,2)=PN(1,2)*R*DEN
      PNF(2,1)=PNF(1,2)
      PNF(2,2)=PN(2,2) - PN(1,2)**2*DEN
C  ***     FILTER UPDATE    ****
      SINF1=SIN(XHAT1)
      COSF1=COS(XHAT1)
```

```
      XHAT(1)=XHAT1+DEN*(-PN(1,1)*SINF1*Z1+PN(1,1)*COSF1*Z2)
      XHAT(2)=XHAT2+DEN*(-PN(1,2)*SINF1*Z1+PN(1,2)*COSF1*Z2)
C     ***  PREDICTOR UPDATE  **
      XHAT1=XHAT(1) + DELT*XHAT(2)
      XHAT2=XHAT(2)
      XDAT(KOUNT,2)=XHAT(1)
      IF(KOUNT.EQ.NO2)
     1WRITE(6,1510) KOUNT,X(1),XHAT(1),X1NL
 1510 FORMAT(* *,I6,5X,*POSITION =*,1PE12.4,4X,*PHASE LOCK ESTIMATE =*,
     11PE12.4,4X,*NONLINEAR ESTIMATE =*,1PE12.4)
      IF(KOUNT .EQ. NO2) GO TO 305
      KOUNT=KOUNT + 1
      GO TO 50
  305 CONTINUE
      SUMP=0.0
      DO 1601 I= 31,KOUNT
      XD=ABS(XDAT(I,1)-XDAT(I,2))
 1698 CONTINUE
      IF(XD.GT.PI ) GO TO 1699
      GO TO 1700
 1699 XD=XD-PI2
      GO TO 1698
 1700 CONTINUE
      SUMP=(XD)**2 + SUMP
 1601 CONTINUE
      SUMP=SUMP/H
      RUMP=SUMP
      XNSAMP=NSAMP
      XAA=XNSAMP+1.0
      SUMP1=(SUMP + XNSAMP*SUMP1)/XAA
      DSUMP1=ALOG1 (SUMP1)*10.
      SUMP= .0
      DO 2601 I= 31,KOUNT
      XD=ABS(XDAT(I,1)-XDAT(I,3))
 2698 CONTINUE
      IF(XD.GT.PI ) GO TO 2699
      GO TO 2700
 2699 XD=XD-PI2
      GO TO 2698
 2700 CONTINUE
      SUMP=(XD)**2 + SUMP
 2601 CONTINUE
      SUMP=SUMP/H
      SUMP2=(SUMP + XNSAMP*SUMP2)/XAA
      DSUMP2=ALOG10 (SUMP2)*10.
      WRITE(6,1511)ISAMP,RUMP,SUMP1,DSUMP1,SUMP,SUMP2,DSUMP2
 1511 FORMAT(* *,[illegible] PATH *,I[illegible],4X,*PHASE LOCK *1P3E12.4,*DB*,
     [illegible]
      NSAMP=NSAMP+1
      IF(NSAMP.EQ.200 ) GO TO 1800
      ISAMP=ISAMP+1
      GO TO 2
 1800 CONTINUE
      GO TO 1
 1700 CONTINUE
```

Reproduced from best available copy.

```
      STOP
      END
```

SUMMARY OF CHANGES MADE BY THE OPTIMIZER
14 WORDS OF INVARIANT RLIST REMOVED FROM THE LOOP STARTING AT LINE 65

SYMBOLIC REFERENCE MAP

ENTRY POINTS		DEF LINE	REFERENCES
4052	MAIN	1	

VARIABLES	SN	TYPE	RELOCATION							
2545	AA1	COMPLEX	ARRAY	REFS	2	132	143	155	156	
				DEFINED	69	151				
2206	AA11	REAL		REFS	157	DEFINED	155			
14207	AA2	COMPLEX	ARRAY	REFS	2	151	DEFINED	70	146	
2267	AA22	REAL		REFS	157	DEFINED	156			
5235	AI	REAL		REFS	68	DEFINED	66			
5230	AJ	REAL		REFS	57	68	DEFINED	56	67	
2257	AK	REAL		REFS	124	125	DEFINED	123		
5167	ALP110	REAL		REFS	15	27	DEFINED	12		
23761	AN	REAL	*UNDEF	REFS	6					
5231	ARG	REAL		REFS	58	59	DEFINED	57		
5211	A11	REAL		REFS	31	62	81	90	DEFINED	26	27
5212	A22	REAL		REFS	31	63	82	91	DEFINED	28	
5255	BSF	REAL		REFS	12?	121	129				
22651	BSF2	REAL	ARRAY	REFS	4	121	126	127			
5155	CMPX	COMPLEX		REFS	3	133	144	DEFINED	132	143	
5157	CMPX2	COMPLEX		REFS	3	133	145	144	146		
				DEFINED	130	133	139	144			
5161	CMPX3	COMPLEX		REFS	3	126	137	DEFINED	124	135	
5163	CMPX4	COMPLEX		REFS	3	127	146	DEFINED	125	137	
5272	COSF1	REAL		REFS	166	167	DEFINED	165			
5170	DELF	REAL		REFS	22	DEFINED	12				
5225	DELSQ	REAL		REFS	54	2*108	DEFINED	53			
5205	DELT	REAL		REFS	23	24	29	53	87	102	108
				2*110	169	DEFINED	22				
5243	DEN	REAL		REFS	108	110	111	159	161	162	166
				167	DEFINED	85	116				
2251	DEVQ1	* REAL		DEFINED	98						
2252	DEVQ2	REAL		REFS	104	DEFINED	99				
5244	DEV1	REAL		REFS	93	DEFINED	90				
5245	DEV2	REAL		REFS	94	DEFINED	91				
5246	DEV3	REAL		REFS	96	97	195	196	DEFINED	92	
5277	DSUMP1	REAL		REFS	211	DEFINED	196				
5300	DSUMP2	REAL		REFS	211	DEFINED	210				
6	DZZZ1	REAL	GN	REFS	9						
24001	EIGVEC	REAL	*UNDEF	REFS	6						
23015	EMSL	REAL	ARRAY	REFS	4	137	DEFINED	58	59	61	
23763	F	REAL	ARRAY	REFS	6	DEFINED	86	87	88	89	
2214	FTC	REAL		REFS	22	29	DEFINED	21			
5215	H	REAL		REFS	191	208	DEFINED	44			
5234	I	INTEGER		REFS	66	69	70	2*182	2*199		
				DEFINED	64	181	198				
5263	IND	INTEGER		REFS	2*142	143	DEFINED	141			
5213	ISAMP	INTEGER		REFS	211	215	216	DEFINED	42	216	
5227	J	INTEGER		REFS	56	58	59	67	69	71	
				DEFINED	55	65					
1	JGAUSS	INTEGER	GN	REFS	9	DEFINED	11				
75	JSLED	INTEGER		REFS	93	94	96	97	104	105	106

VARIABLES	SN	TYPE	RELOCATION								
				DEFINED	12						
5256	K	INTEGER		REFS	123	2*126	2*127	DEFINED	122		
5240	KOUNT	INTEGER		REFS	95	103	158	171	2*172	176	177
				181	198	DEFINED	76	177			
5261	L	INTEGER		REFS	137	141	146	2*151	DEFINED	136	150.
5262	M	INTEGER		REFS	143	146	2*151	DEFINED	138	149	
5173	NO1	INTEGER		REFS	48	50	52	55	DEFINED	12	
5174	NO2	INTEGER		REFS	44	172	176	DEFINED	12		
5214	NSAMP	INTEGER		REFS	193	214	DEFINED	43	214		
5223	NTE	INTEGER		REFS	132	DEFINED	51				
5220	NTERM	INTEGER		REFS	49	121	122	DEFINED	48		
5221	NTERM1	INTEGER		REFS	51	64	65	126	127	129	131
				136	138	140	142	143	149	150	
				DEFINED	49						
5222	NTERM2	INTEGER		REFS	51	58	59	61	66	67	141
				153	154	DEFINED	50				
5224	NTERM3	* INTEGER		DEFINED	52						
5260	NU	INTEGER		REFS	3*132	141	2*143	DEFINED	131	142	
5172	NUM1	* INTEGER		DEFINED	12						
0	NZZZ	INTEGER	ARRAY RN	REFS	8						
5265	N,	INTEGER		REFS	155	156	DEFINED	154			
5264	N1	INTEGER		REFS	155	156	DEFINED	153			
5253	OBMAG	REAL		REFS	120	121	DEFINED	118			
5254	OBPH	REAL		REFS	124	125	DEFINED	119			
23751	PBAR	REAL	*UNDEF	REFS	6						
5232	PC1	REAL		REFS	68	DEFINED	52				
5233	PC2	REAL		REFS	68	DEFINED	63				
5211	PDB	REAL		REFS	29	DEFINED	18				
23767	PDUMY	REAL	ARRAY	REFS	6	112	113	114	DEFINED	108	110
				111							
23773	PDUMY2	REAL	*UNDEF	REFS	6						
5177	PI	REAL		REFS	17	184	201	DEFINED	16		
5200	PI2	REAL		REFS	186	203	DEFINED	17			
23755	PN	REAL	ARRAY	REFS	6	85	4*108	3*110	2*111	115	116
				159	160	2*162	2*166	2*167	DEFINED	81	82
				83	84	112	113	114	115		
24007	PNF	REAL	ARRAY	REFS	6	161	DEFINED	159	160	161	162
5176	PI.0	REAL		REFS	18	29	25	26	29		
				DEFINED	15						
5210	P220	REAL		REFS	28	DEFINED	25				
23745	Q	REAL	ARRAY	REFS	6	98	99	111	DEFINED	72	73
5226	QEXP	REAL		REFS	57	DEFINED	54				
5202	QQ	REAL		REFS	20	21	DEFINED	19			
5216	Q22	REAL		REFS	29	54	73	DEFINED	23		
5171	Q22C	REAL		REFS	19	23	25	30	DEFINED	12	
5237	R	REAL		REFS	85	92	108	110	116	118	159
				160	DEFINED	74					
5275	RUMP	REAL		REFS	211	DEFINED	192				
5203	RX	REAL		REFS	21	24	25	30	DEFINED	20	
5207	R11	REAL		REFS	29	74	DEFINED	24			
5271	SINF1	REAL		REFS	166	167	DEFINED	164			
5301	SSK	COMPLEX	ARRAY	REFS	2	132	1-3	DEFINED	126	127	129
5273	SUMP	REAL		REFS	189	191	192	195	206	208	209
				211	DEFINED	180	189	191	197	206	208

VARIABLES	SN	TYPE	RELOCATION									
5216	SUMP1	REAL			REFS	195	196	211	DEFINED	45	195	
5217	SUMP2	REAL			REFS	209	210	211	DEFINED	46	209	
24605	X	REAL	ARRAY		REFS	6	93	94	95	96	97	2*102
						103	2*104	105	106	172	DEFINED	102
5236	XAA	REAL			REFS	69	195	209	DEFINED	68	194	
5274	XD	REAL			REFS	184	186	199	201	203	206	
					DEFINED	182	186	199	203			
23137	XDAT	REAL	ARRAY		REFS	5	2*182	2*199	DEFINED	95	103	158
						171						
23777	XHAT	REAL	ARRAY		REFS	6	2*169	170	171	172		
					DEFINED	77	78	166	167			
5241	XHAT1	REAL			REFS	164	165	166	DEFINED	79	169	
5242	XHAT2	REAL			REFS	167	DEFINED	80	170			
5276	XNSAMP	REAL			REFS	194	195	209	DEFINED	193		
3	XNZ	REAL	ARRAY	RN	REFS	8						
2	XZZZ	REAL	ARRAY	GN	REFS	9						
5270	X1NL	REAL			REFS	158	172	DEFINED	157			
5165	Y1EST	REAL			REFS	77	79	93	DEFINED	12		
5166	Y2EST	REAL			REFS	78	80	94	DEFINED	12		
5247	Z1	REAL			REFS	96	105	118	119	166	167	
5250	Z2	REAL			REFS	97	106	118	119	166	167	

FILE NAMES		MODE										
0	INPUT											
2022	OUTPUT											
	TAPE5	FMT		READS	12							
2022	TAPE6	FMT		WRITES	29	30	31	172	211			

EXTERNALS	TYPE	ARGS		REFERENCES							
ALOG	REAL	1	LIBRARY	18	196	217					
ATAN2	REAL	2	LIBRARY	119	157						
CEXP	COMPLEX	1	LIBRARY	126	127						
COS	REAL	1	LIBRARY	96	105	165					
EOF	REAL	1		13							
EXP	REAL	1	LIBRARY	57	68						
GAUSS		4		93	94	96	97	154	115	106	
INUE		4		121							
IO		2		12							
SIN	REAL	1	LIBRARY	97	106	164					
SQRT	REAL	1	LIBRARY	2	21	25	91	91	92	98	99	118

INLINE FUNCTIONS	TYPE	ARGS		DEF LINE	REFERENCES						
ABS	REAL	1	INTRIN		182	199					
AIMAG	REAL	1	INTRIN		155						
CMPLX	COMPLEX	2	INTRIN		69	124	125	126	127	129	137
REAL	REAL	1	INTRIN		156						

STATEMENT LABELS		DEF LINE	REFERENCES
4053	1	1	219
4331	2	75	217
4345	50	101	178
4436	73	117	181
4664	305	179	176
0	451	6	55

STATEMENT LABELS				DEF LINE	REFERENCES	
0	452			71	64	65
0	230			126	122	
0	545			134	131	
4542	550			145	14.	142
0	560			147	136	138
0	570			152	149	15?
5101	651	FMT		35	12	
5064	652	FMT		36	29	
5047	653	FMT		33	3?	
5040	654	FMT		32	31	
5114	1510	FMT		174	172	
5127	1511	FMT		212	211	
0	1601			19?	181	
4673	1698			183	187	
4675	1699			130	184	
4677	1700			188	185	
0	1731		INACTIVE	14	13	
4770	1799			22	13	
4767	1800			218	219	
0	2601			207	198	
4722	2698			20.	204	
4724	2699			213	201	
4726	2700			215	202	
5111	4122	FM	REFS	148		

LOOPS	LABEL	IN	FROM-TO	LENGTH	PROPERTIES
4225	451	* J	55 60	128	EXT REFS
4246	452	* I	64 71	278	EXT REFS NOT INNER
4251	452	* J	5 71	228	EXT REFS
4421	530	* K	120 128	3 8	EXT REFS
4464	545	* NU	131 134	138	OPT
4503	560	* L	136 147	508	NOT INNER
4513	560	* M	138 147	448	NOT INNER
4521	551	* NU	141 [illegible]	238	OPT
4562	570	* M	149 [illegible]	148	NOT INNER
4571	570	L	151 [illegible]	38	INSTACK
4671	1601	I	181 19[illegible]	118	OPT
4720	2601	I	198 20[illegible]	118	OPT

COMMON BLOCKS	LENGTH	MEMBERS - BIAS NAME(LENGTH)		
RN	5	0 NZZZ (3)	3 XNZ (2)	
GN	4	3 JZZZ1 (1)	1 JGAUSS (1)	2 XZZZ (2)

STATISTICS		
PROGRAM LENGTH	177518	8169
BUFFER LENGTH	40648	2084
COMMON LENGTH	118	9

454

```
C
C     ..................................................................
C
C        SUBROUTINE I0
C
C        PURPOSE
C            COMPUTE THE MODIFIED BESSEL FUNCTION I OF ORDER ZERO
C
C        USAGE
C            CALL I0(X,RI0)
C
C        DESCRIPTION OF PARAMETERS
C            X    -GIVEN ARGUMENT OF THE BESSEL FUNCTION I OF ORDER
C            RI0  -RESULTANT VALUE OF THE BESSEL FUNCTION I OF ORDER
C
C        REMARKS
C            LARGE VALUES OF THE ARGUMENT MAY CAUSE OVERFLOW IN THE
C            BUILTIN EXP-FUNCTION
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           POLYNOMIAL APPROXIMATIONS GIVEN BY E.E. ALLEN ARE USED FOR
C           CALCULATION.
C           FOR REFERENCE SEE
C           M. ABRAMOWITZ AND I.A. STEGUN,*HANDBOOK OF MATHEMATICAL
C           FUNCTIONS*, U.S. DEPARTMENT OF COMMERCE, NATIONAL BUREAU OF
C           STANDARDS APPLIED MATHEMATICS SERIES, 1966, P.378.
C
C     ..................................................................
C
      SUBROUTINE I0(X,RI0)
      RI =ABS(X)
      IF(RI -3.75)1,1,2
    1 Z=X*X*7.111111E-2
      RI =(((((4.5813E-3*Z+3.60768E-2)*Z+2.659732E-1)*Z+1.206749E0)*Z
     1+3.089942E0)*Z+3.515623E0)*Z+1.
      RETURN
    2 Z=3.75/RI0
      RI = EXP(RI0)/SQRT(RI0)*((((((((3.92377E-3*Z-1.647633E-2)*Z
     1+2.635537E-2)*Z-2.577[illegible]6E-2)*Z+9.16281E-3)*Z-1.57565E-3)*Z
     2+2.25319E-3)*Z+1.328592E-2)*Z+3.989423E-1)
      RETURN
      END
```

SYMBOLIC REFERENCE MAP

ENTRY POINTS		DEF LINE	REFERENCES							
2	IJ	33	39	44						

VARIABLES	SN	TYPE	RELOCATION								
0	RIJ	REAL	F.P.	REFS	35	43	2*41	DEFINED	33	34	37
J	X	REAL		REFS	34	2*36	DEFINED	33			
73	Z	REAL	F.P.	REFS	6*37	8*41	DEFINED	36	40		

EXTERNALS	TYPE	ARGS		REFERENCES
EXP	REAL	1	LIBRARY	41
SQRT	REAL	1	LIBRARY	41

INLINE FUNCTIONS	TYPE	ARGS		DEF LINE	REFERENCES
ABS	REAL	1	INTRIN		34

STATEMENT LABELS			DEF LINE	REFERENCES
0	1	INACTIVE	36	2*35
25	2		4	35

STATISTICS

PROGRAM LENGTH 748 6

456

```
      C                                                                      INUE  10
      C     ******************************************************************INUE  20
      C                                                                      INUE  30
      C     SUBROUTINE INUE                                                  INUE  40
      C                                                                      INUE  50
      C     PURPOSE                                                          INUE  60
      C        COMPUTE THE MODIFIED BESSEL FUNCTIONS I FOR ORDERS 1 TO N     INUE  70
      C                                                                      INUE  80
      C     USAGE                                                            INUE  90
      C        CALL INUE(X,N,ZI,RI)                                          INUE 100
      C                                                                      INUE 110
      C     DESCRIPTION OF PARAMETERS                                        INUE 120
      C        X     -GIVEN ARGUMENT OF THE BESSEL FUNCTIONS I               INUE 130
      C        N     -GIVEN MAXIMUM ORDER OF BESSEL FUNCTIONS I              INUE 140
      C        ZI    -GIVEN VALUE OF BESSEL FUNCTION I OF ORDER ZERO         INUE 150
      C               FOR ARGUMENT X                                         INUE 160
      C        RI    -RESULTANT VECTOR OF DIMENSION N, CONTAINING THE        INUE 170
      C               VALUES OF THE FUNCTIONS I FOR ORDERS 1 TO N            INUE 180
      C                                                                      INUE 190
      C     REMARKS                                                          INUE 200
      C        THE VALUE OF ZI MAY BE CALCULATED USING SUBROUTINE I0.        INUE 210
      C        USING A DIFFERENT VALUE HAS THE EFFECT THAT ALL VALUES OF     INUE 220
      C        BESSEL FUNCTIONS I ARE MULTIPLIED BY THE  FACTOR ZI/I(0,X)    INUE 230
      C        WHERE I(0,X) IS THE VALUE OF I FOR ORDER 0 AND ARGUMENT X.    INUE 240
      C        THIS MAY BE USED DISADVANTAGEOUSLY IF ONLY THE RATIOS OF I    INUE 250
      C        FOR DIFFERENT ORDERS ARE REQUIRED.                            INUE 260
      C                                                                      INUE 270
      C     SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                    INUE 280
      C        NONE                                                          INUE 290
      C                                                                      INUE 300
      C     METHOD                                                           INUE 310
      C        THE VALUES ARE OBTAINED USING BACKWARD RECURRENCE RELATION    INUE 320
      C        TECHNIQUE. THE RATIO I(N+1,X)/I(N,X) IS OBTAINED FROM A       INUE 330
      C        CONTINUED FRACTION.                                           INUE 340
      C        FOR REFERENCE SEE                                             INUE 350
      C        G. BLANCH,*NUMERICAL EVALUATION OF CONTINUED FRACTIONS*,       INUE 360
      C        SIAM REVIEW, VOL.6,NO.4,1964,PP.383-421.                      INUE 370
      C                                                                      INUE 380
      C     ******************************************************************INUE 390
      C                                                                      INUE 400
            SUBROUTINE INUE(X,N,ZI,RI)                                       INUE 410
            DIMENSION RI(1)                                                  INUE 420
            IF(N)10,10,1                                                     INUE 430
          1 FN=N                                                             INUE 440
            Q1=X/4.                                                          INUE 450
            IF(ABS(X)-3.E-4)6,6,2                                            INUE 460
          2 A0=1.                                                            INUE 470
            A1=0.                                                            INUE 480
            B0=0.                                                            INUE 490
            B1=1.                                                            INUE 500
            FI=FN                                                            INUE 510
          3 FI=FI+2.                                                         INUE 520
            AN=FI/ABS(X)                                                     INUE 530
            A=AN*A1+A0                                                       INUE 540
            B=AN*B1+B0                                                       INUE 550
```

```
                A1=A1                                                   INUE 560
                B1=B1                                                   INUE 570
                A1=A                                                    INUE 580
                B1=B                                                    INUE 590
      60        Q1=Q1                                                   INUE 600
                Q1=A/B                                                  INUE 610
                IF(ABS((Q1-Q )/Q1)-1.E-6)4,4,3                          INUE 620
              4 IF(X)5,6,6                                              INUE 630
              5 Q1=-Q1                                                  INUE 640
      65      6 K=N                                                     INUE 650
              7 Q1=X/(FN+X*Q1)                                          INUE 660
                RI(K)=Q1                                                INUE 670
                FN=FN-2.                                                INUE 680
                K=K-1                                                   INUE 690
      70        IF(K)8,8,7                                              INUE 700
              8 FI=ZI                                                   INUE 710
                DO 9 I=1,N                                              INUE 720
                FI=FI*RI(I)                                             INUE 730
              9 RI(I)=FI                                                INUE 740
      75     10 RETURN                                                  INUE 750
                END                                                     INUE 760
```

REGISTER ALLOCATION
3 REGISTERS ASSIGNED OVER THE LOOP BEGINNING AT LINE 66
1 REGISTERS ASSIGNED OVER THE LOOP BEGINNING AT LINE 72

SYMBOLIC REFERENCE MAP

ENTRY POINTS		DEF LINE	REFERENCES
2	INUE	41	75

VARIABLES	SN	TYPE		RELOCATION								
121	A	REAL			REFS	58	61	DEFINED	54			
120	AN	REAL			REFS	54	55	DEFINED	53			
113	A0	REAL			REFS	54	DEFINED	47	56			
114	A1	REAL			REFS	54	56	DEFINED	48	58		
122	B	REAL			REFS	59	61	DEFINED	55			
115	B0	REAL			REFS	55	DEFINED	49	57			
116	B1	REAL			REFS	55	57	DEFINED	50	59		
117	FI	REAL			REFS	52	53	73	74	DEFINED	51	52
						71	73					
111	FN	REAL			REFS	45	51	66	68	DEFINED	44	68
125	I	INTEGER			REFS	73	74	DEFINED	72			
124	K	INTEGER			REFS	67	69	70	DEFINED	65	69	
0	N	INTEGER		F.P.	REFS	43	2*44	65	72	DEFINED	41	
123	Q0	REAL			REFS	62	DEFINED	60				
112	Q1	REAL			REFS	61	2*62	64	66	67		
					DEFINED	45	61	64	66			
0	R1	REAL	ARRAY	F.P.	REFS	42	73	DEFINED	41	67	74	
0	X	REAL		F.P.	REFS	45	46	53	63	2*66		
					DEFINED	41						
0	ZI	REAL		F.P.	REFS	71	DEFINED	41				

INLINE FUNCTIONS	TYPE	ARGS		DEF LINE	REFERENCES	
ABS	REAL	1	INTRIN		46	53 62

STATEMENT LABELS		DEF LINE	REFERENCES	
0 1	INACTIVE	44	43	
0 2	INACTIVE	47	46	
31 3		52	62	
0 4	INACTIVE	63	2*62	
0 5	INACTIVE	64	63	
56 6		65	2*46	2*63
61 7		66	73	
0 8	INACTIVE	71	2*70	
0 9		74	72	
101 10		75	2*43	

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
76	9	I	72 74	28	INSTACK

STATISTICS

PROGRAM LENGTH 1418 97

FUNCTION RANF CDC 6600 FTN V3.0-P213 OPT=2 03/26/72 01.12.49. PAGE 1

```
      FUNCTION RANF(NS,MODE)
      DIMENSION  NS(2), NC(2)
      COMMON /RN/ N1, N2, MP, T1, T2
      DATA  M1, M2/244734, 158551/
C        MODE=0 TO CONTINUE, OTHERWISE RESTART WITH
C        INTEGER NUMBER NS(1)*2**18+NS(2)
      IF (MODE) 10, 100, 10
   10 N1=NS(1)
      N2=NS(2)
      T1=2.**(-18)
      T2=2.**(-36)
      MP=2**18
    1 DO 200 I=1,2
      GO TO (110,120),I
  110 K=M2*N2
      GO TO 19
  120 K=M1*N2+M2*N1+KD
  190 KD=K/MP
  200 NC(I)=K-KD*MP
      N1=NC(2)
      N2=NC(1)
      XN1=N1
      XN2=N2
      RANF=XN1*T1+XN2*T2
      RETURN
      END
```

SYMBOLIC REFERENCE MAP

ENTRY POINTS		DEF LINE	REFERENCES
2	RANF	1	25

VARIABLES	SN	TYPE		RELOCATION								
76	I	INTEGER			REFS	14	19	DEFINED	13			
77	K	INTEGER			REFS	18	19	DEFINED	15	17		
100	KO	INTEGER			REFS	17	19	DEFINED	18			
0	MODE	INTEGER		F.P.	REFS	7	DEFINED	1				
2	MP	INTEGER		RN	REFS	3	18	19	DEFINED	12		
72	M1	INTEGER			REFS	17	DEFINED	4				
73	M2	INTEGER			REFS	15	17	DEFINED	4			
103	NC	INTEGER	ARRAY		REFS	2	20	21	DEFINED	19		
0	NS	INTEGER	ARRAY	F.P.	REFS	2	8	9	DEFINED	1		
0	N1	INTEGER		RN	REFS	3	17	22	DEFINED	8	20	
1	N2	INTEGER		RN	REFS	3	15	17	23	DEFINED	9	21
75	RANF	REAL			DEFINED	24						
3	T1	REAL		RN	REFS	3	24	DEFINED	10			
4	T2	REAL		RN	REFS	3	24	DEFINED	11			
101	XN1	REAL			REFS	24	DEFINED	22				
102	XN2	REAL			REFS	24	DEFINED	23				

STATEMENT LABELS			DEF LINE	REFERENCES
0	10	INACTIVE	8	2*7
27	100		13	7
41	11		15	14
44	120		17	14
51	190		18	16
0	200		19	13

LOOPS	LABEL	INDEX	FROM-TO	LENGTH	PROPERTIES
33	200	* I	13 19	258	OPT

COMMON BLOCKS	LENGTH	MEMBERS - BIAS NAME(LENGTH)							
RN	5	0 N1	(1)	1 N2	(1)	2 MP	(1)		
		3 T1	(1)	4 T2	(1)				

STATISTICS		
PROGRAM LENGTH	1128	74
COMMON LENGTH	58	5

```
      SUBROUTINE GAUSS(JS,SD,XM,X)
      DIMENSION NST(2)
      COMMON /RN/ N1, N2, MC, T1, T2
      COMMON /GN/ TWOPI, J, XR(2)
      IF (J) 10, 1, 20
   10 J=2
      TWOPI=8.*ATAN(1.)
      NST(1)=244734
      NST(2)=158551
      NST(1)=102943
      NST(2)=185617
      XR(1)=RANF(NST,1)
      GO TO 35
   20 GO TO (30,40), J
   30 J=2
      XR(1)=RANF(NST,J)
   35 XR(2)=RANF(NST,J)
      X1=SQRT(ABS(-2.*ALOG(XR(1))))
      XR(2)=TWOPI*XR(2)
      XR(1)=X1*SIN(XR(2))
      XR(2)=X1*COS(XR(2))
      X=XR(1)*SD+XM
      RETURN
   40 J=1
      X=XR(2)*SD+XM
      RETURN
      END
```

SYMBOLIC REFERENCE MAP

ENTRY POINTS		DEF LINE	REFERENCES
2	GAUSS	1	23 20

VARIABLES	SN	TYPE		RELOCATION									
1	J	INTEGER		GN	REFS	4	5	14	DEFINED	6	15	24	
0	JS	INTEGER	*UNUSED	F.P.	DEFINED	1							
2	MC	INTEGER		RN	REFS	3							
100	NST	INTEGER	ARRAY		REFS	2	12	16	17	DEFINED	8	9	
0	N1	INTEGER			DEFINED	11							
1	N2	INTEGER		RN	REFS	3							
0	SO	REAL		RN	REFS	3							
0	TWOPI	REAL		F.P.	REFS	22	25	DEFINED	1				
3	T1	REAL		GN	REFS	4	19	DEFINED	7				
4	T2	REAL		RN	REFS	3							
0	X	REAL		RN	REFS	3							
0	XM	REAL		F.P.	DEFINED	1	22	25					
2	XR	REAL	ARRAY	F.P.	REFS	22	25	DEFINED	1				
				GN	REFS	4	18	19	20	21	22	25	
77	X1	REAL			DEFINED	12	16	17	19	20	21		
					REFS	20	21	DEFINED	18				

EXTERNALS	TYPE	ARGS	REFERENCES		
ALOG	REAL	1 LIBRARY	18		
ATAN	REAL	1 LIBRARY	7		
COS	REAL	1 LIBRARY	21		
RANF	REAL	2	12	10	17
SIN	REAL	1 LIBRARY	2		
SQRT	REAL	1 LIBRARY	18		

INLINE FUNCTIONS	TYPE	ARGS	DEF LINE	REFERENCES
ABS	REAL	1	INTRIN	18

STATEMENT LABELS		DEF LINE	REFERENCES
0 10	INACTIVE	6	2*5
16 20		14	5
24 30		15	14
30 35		17	13
53 40		24	14

COMMON BLOCKS	LENGTH	MEMBERS - BIAS NAME(LENGTH)							
RN	5	0 N1	(1)						
		3 T1	(1)	4 N2	(1)		2 MC	(1)	
GN	4	0 TWOPI	(1)	4 T2	(1)				
				1 J	(1)		2 XR	(2)	

STATISTICS

PROGRAM LENGTH	1028	66
COMMON LENGTH	118	9

Additional Appendix F.

Some Unpublished Conference Papers Referenced by this Report

The papers in this Appendix are referenced frequently in this report. Because of their relative inaccessibility to the readers of the report they are reproduced for reference here.

Preceding page blank

Reprinted from Proc. Second Symp. on Nonlinear Estimation Theory and Its Applications, San Diego, Sept. 1971, 51-58.

REALIZATION OF NON-LINEAR FILTERS*

R. S. Bucy**

Abstract

The concept of numerically exact non-linear filter synthesis is defined and shown to be of importance. Confidence intervals for error performance are described, which allow one to make meaningful statistical inferences about filter performance. Further, some recent synthesis results for non-linear filters are discussed.

1. INTRODUCTION

In the past eleven years, a rather amazingly complete theory of non-linear filtering has emerged, see [1], [2], [3], and [4]. In fact, the representation theorem, see [4], has provided a solution to a general class of non-linear filtering problems, albeit the solution depends on the evaluation of a function space integral. Unfortunately, the applications of this theory have been almost non-existent. The reason for this state of affairs is paradoxical; the success and wide-spread use of the Kalman-Bucy linear theory, see [4] and [5]. The situation is reminiscent of the popularity of the frequency domain techniques in the mid-50's, which delayed the state space viewpoint for some years.

In this paper, I would like to review some of the efforts at synthesis of optimal filters for some specific problems, and at the same time point out some slightly paradoxical things I have learned over the past three years. In particular, the meaning of Monte Carlo results will be discussed in some detail, as it is now commonplace to design systems on the basis of Monte Carlo runs, too often only 25 to 100 Monte Carlo runs are made. Of course, as we shall show, such a small number of runs produces unbelievably large uncertainties. Finally, we will discuss in detail a parameter estimation problem considered by Licht in [6] and the phase lock study in [7].

2. NUMERICALLY EXACT REALIZATIONS

The basic problem of machine realization of optimal non-linear filters consists in the iteration of a non-linear convolution equation of the form

$$J_n(\underline{x}) = \int_{-\infty}^{\infty} T_n(\underline{x}, \underline{\xi})\, J_{n-1}(\underline{\xi})\, d\underline{\xi} \qquad (2.1)$$

with $J_n(x)$ the conditional density of the signal process at time n, given the observations up to $n-1$ and T_n a non-linear function of the last piece of data. Of course, (2.1) is nothing but a discrete form of Bayes' rule which can be written as (2.1), when the signal process is Markovian and the plant noise as well as the observation noise are independent white noise sequences. For details on how $J_n(x)$ can be represented as a set of $(2M+1)^{d***}$ weights on a moving grid, see [11] and [8]. The important point is that the point

*This research was partially supported by the Air Force Office of Scientific Research, Office of Aerospace Research, United States Air Force, under Grant AF-AFOSR-1244-67.

**University of Southern California, Los Angeles.

*** d is the state dimension

Preceding page blank

mass representation is numerically exact, that is the sup of the difference between the true conditional probability of a compact set and the approximating probability converges to zero as the number of point masses gets large.

To illustrate the importance of numerical exactness, consider the following problem:

$$x_n = \alpha x_{n-1} + u_{n-1}\ ,\ x_o = c$$
$$z_n = x_n^3 + v_n \qquad (2.2)$$

with u_n and v_n independent zero mean Gaussian white noise sequences and c a mutually independent zero mean Gaussian white noise sequence. This problem was considered in [11] and [10]. The behavior of $\hat{x}_{n/n}(M) = Ex_n|z_n, \ldots, z_o)$ as computed as function of M, the number of weights in the approximation to $J_n(x)$ is quite remarkable. As long as none of the observations are large in comparison to the observation noise standard deviation, an M of 48 produces $\hat{x}_{n/n}(M) \approx \hat{x}_{n/n}(2M)$ to four places. However, if z_n is large, then $\hat{x}_{n/n}(256) = z_n^{1/3}$. What happens is that in the case of z_n large, the filter density approximates a delta function and the weight approximation to the filter density must contain a large number of weights to be accurate. The phenomenon is quite interesting as it shows that bandwidth is irrelevant for non-linear filters. Because we used a numerically exact realization scheme, we found this phenomenon, which, by the way is easy to explain after it was observed by looking at the error of the estimate $z_n^{1/3}$.

Without a numerically exact scheme, there is no way of knowing when the approximation to the optimal filter is adequate. We note in passing that the sub-optimal schemes, such as [12] are not numerically exact, as T_n is replaced by a truncated series approximation.

3. MONTE CARLO ERROR EVALUATION

Suppose that a random variable x is estimated as $\hat{x}$ and that x_i and $\hat{x}_i$ denote the i independent trial and the statistic S_n^2 is computed as

$$S_n^2 = \frac{1}{n}\sum_{i=1}^{n} \tilde{x}_i^2$$

with $\tilde{x}_i = x_i - \hat{x}_i$, then

$$\frac{N S_N^2 - N\mu_2}{\sqrt{N}\sqrt{\mu_4 - \mu_2^2}} \quad \text{with } \mu_\alpha = E\tilde{x}^\alpha \qquad (3.1)$$

is asymptotically normal of mean zero and variance 1 in view of the central limit theorem. In other words, S_n^2 is normal mean μ_2 and variance $\frac{(\mu_4 - \mu_2^2)}{n}$.

For Monte Carlo error analysis then S_n^2 is the estimate of the mean square error, and with probability .9972

$$-3\sqrt{\frac{\mu_4 - \mu_2^2}{n}} \le S_n^2 - \mu_2 \le 3\sqrt{\frac{\mu_4 - \mu_2^2}{n}} \qquad (3.2)$$

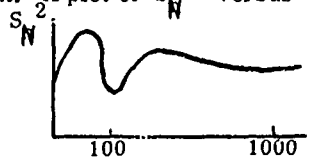
if $\mu_4 \approx 3\mu_2^2$ as it is when $\tilde{x}$ is Gaussian, then with probability .9972

$$\frac{S_n^2}{1+3\sqrt{\frac{2}{n}}} \le \mu_2 \le \frac{S_n^2}{1-3\sqrt{\frac{2}{n}}} \qquad (3.3)$$

In Figure I, we have plotted $3\sqrt{\frac{2}{n}}$ versus N. Note how slowly $3\sqrt{\frac{2}{n}}$ goes to zero. The net effect is that in order to achieve a 3σ confidence interval of overall length .2 of S_n^2, 2,000 Monte Carlo's are needed; under the assumption $\mu_4 \approx 3\mu_2^2$. Of course, if $\mu_4 << 3\mu_2^2$, the situation improves, but if $\mu_4 > 3\mu_2^2$, the situation is worse.

In view of these results, the Monte Carlo analysis of systems which is often made with 16 to 100 runs, seems curious at best. Of course, another advantage of synthesis methods, which are numerically exact and carry the distribution, is that μ_4 can be at least approximately determined, so that with (3.3) the confidence of an estimate can be more exactly determined. We used 2,000 Monte Carlo runs to evaluate the performance of the phase-lock loop in [7] while 500 Monte Carlo runs were used in [8], as we

proved order of magnitude mean square error betterment. A plot of S_N^2 versus N can look like



and if such a graph is made for a particular problem, it will quickly convince one that no inference should be made on μ_2 without a bare minimum of about 500 Monte Carlo runs.

In order to illustrate that the problems raised above are real, we consider the following problem considered first by Licht in [6]

$$\frac{dx}{dt} = -x \quad , \quad x(o) = c$$
$$dz = (x + x^3)dt + dv \tag{3.4}$$

with c Gaussian mean one variance 1 and the dv process white noise of spectral density R. Licht used the Representation Theorem and obtained the experimental results shown in Figures II, III and IV, which he disclosed to me in [13], after I had seen the results of [6] and suggested he simulate the sensor orbit filter. The sensor orbit filter, see [4] and [9], is the wide sense Kalman-Bucy filter for the system

$$\frac{dy_1}{dt} = -y_1 \quad \text{and} \quad \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} \in N\left(\begin{pmatrix} 1 \\ 4 \end{pmatrix}, \begin{pmatrix} 1 & 6 \\ 6 & 60 \end{pmatrix}\right)^*$$
$$\frac{dy_2}{dt} = -3y_2 \tag{3.5}$$
$$dz = (y_1 + y_2)dt + dv$$

where $y_1 = x$, $y_2 = x^3$. Note that (3.5) is just (3.4) in new coordination. The true error performance of the wide-sense filter for (3.5) is easily evaluated by solving a Riccati equation and is shown on the figures. Notice first if we assume that $\mu_4 \geq 3\mu_2^2$ that the three plots are inconclusive about the relative error performance of the optimal versus the extended Kalman-Bucy filter as the confidence intervals are too large. In fact, none of the experimental results are precise enough to draw any conclusions**. However, the true sensor orbit performance is significantly higher than the experimental extended Kalman-Bucy filter error, on the basis of Figure II, only. The conclusion that we may draw then is the following, the extended K-B filter performance depends on the coordination of the signal process, not too surprising. However, even more significant is the following: The performance of the extended K. B. filter may be worse in sensor orbit coordinates. (N. B. the wide-sense sensor orbit filter coincides with the extended K. B. filter for (3.5)).

This result surprised me, as intuitively I expected the extended K. B. filter to perform better in sensor orbit coordinates. I feel it necessary to remark that Licht's thesis is an excellent piece of work; its only defect consists in drawing conclusions from too small a number of Monte Carlo runs. This defect is shared by many papers, for example [12], where 50 Monte Carlo's are used.

3. PHASE LOCK LOOP DESIGN

A problem which has an interesting history as well as major technological importance is that of phase detection. An idealized model of which is the following:

$$d\phi = d\beta \; ; \; \phi_o = c$$
$$dz_1 = \cos\phi\, dt + dv_1 \tag{4.1}$$
$$dz_2 = \sin\phi\, dt + dv_2$$

where $d\beta$, dv_1, dv_2 are mutually independent white noise processes of spectral densities q, 2r, and 2r, while c is a Gaussian random variable of zero mean and variance A. A more realistic model, ϕ the output of two integrators is being

* $y \in N(\mu, P)$ denotes that y is normally distributed mean μ covariance P.

** The confidence intervals are too large.

studied by Hecht in [10], however, I will confine myself to a discussion of the model (4.1). The interested reader should consult [7] for the detailed analysis of the optimal filter for (4.1).

It can easily be shown that the steady-state extended K-B filter for the model (4.1) is given by

$$d\hat{\phi} = \sqrt{\frac{q}{2r}}\,(dz_2 \cos\hat{\phi} + dz_2 \sin\hat{\phi}) \qquad (4.2)$$

with $\hat{\phi}(o) = 0$.

Now $\hat{\phi}$ given by (4.2) is the output of the classical 1 dimensional phase-lock loop and for small $\phi - \hat{\phi}$ the error is $R = \sqrt{2rq}$.

Since for an important class of problems only the relative phase is of interest, in [7] we considered the loss function $L(\phi - \phi^*) = 2(1 - \cos(\phi - \phi^*))$ which is quadratic for small $\phi - \phi^*$, but cyclic otherwise, in other words, ϕ^* was chosen so that

$$\begin{aligned} &E\, L(\phi_n - \phi_n^*) \mid z_i(j) \;\; i = 1, 2;\; j = 1 \ldots n-1) \\ &\leq E\, L(\phi_n - \psi) \mid z_i(j) \;\; i=1, 2;\; j = 1, \ldots n-1 \;) \end{aligned} \qquad (4.3)$$

for the discrete version of the problem. It is easily seen that ϕ_n^* is given by

$$\phi_n^* = \tan^{-1} \frac{\hat{S}_n}{\hat{C}_n} \qquad (4.4)$$

with $\hat{S}_n = E \sin \phi_n \mid z_i(j) \;\; i=1, 2;\; j=1, \ldots, n-1)$

$\hat{C}_n = E \cos \phi_n \mid z_i(j)\; i=1, 2;\; j=1, \ldots, n-1).$

Let us agree to call ϕ_n^* the cyclic non-linear estimator. It is easily seen that

$$\begin{aligned} &E\, L(\phi_n - \phi_n^*) \mid z_i(j) \;\; i=1, 2;\; j=1 \ldots n-1) \\ &= 2\left(1 - \sqrt{\hat{C}_n^2 + \hat{S}_n^2}\;\right) \end{aligned} \qquad (4.5)$$

In Figure V, the experimental results are given together with the theoretical error curve derived by Viterbi in [14]. The 3σ confidence interval extends .4 dB above and below the non-linear error performance curve, so that we may conclude that the optimal filter achieves a mean square error improvement of between .6 dB and 1.4 dB. However, since the lower curve represents a lower bound, the maximum achievable error improvement would be 2 dB. The optimal filter achieves 1/3 to 2/3 of the error performance difference of the filter which senses ϕ directly versus the classical phase-lock loop. The maximum achievable error improvement for the more realistic two-dimensional phase-lock loop (i.e., the phase is doubly integrated white noise) is about 6 db, so that one might hope for a 3 db error betterment for the optimal filter for this problem, and this would certainly affect the way on how phase-lock loops are built in the future. Unfortunately, the current economic doldrums have adversely affected NASA's ability to support such a study.

4. CONCLUSIONS

In the past three years, non-linear optimal filters have been built, using digital computers, and have been evaluated, using Monte Carlo results. The purpose of this paper has been to indicate the amount of Monte Carlo runs necessary to make valid statistical inferences as to filter performance, as well as reviewing some of the results of recent studies of non-linear filters. I hope this paper will have the effect of stimulating meaningful work in the area of non-linear filtering.

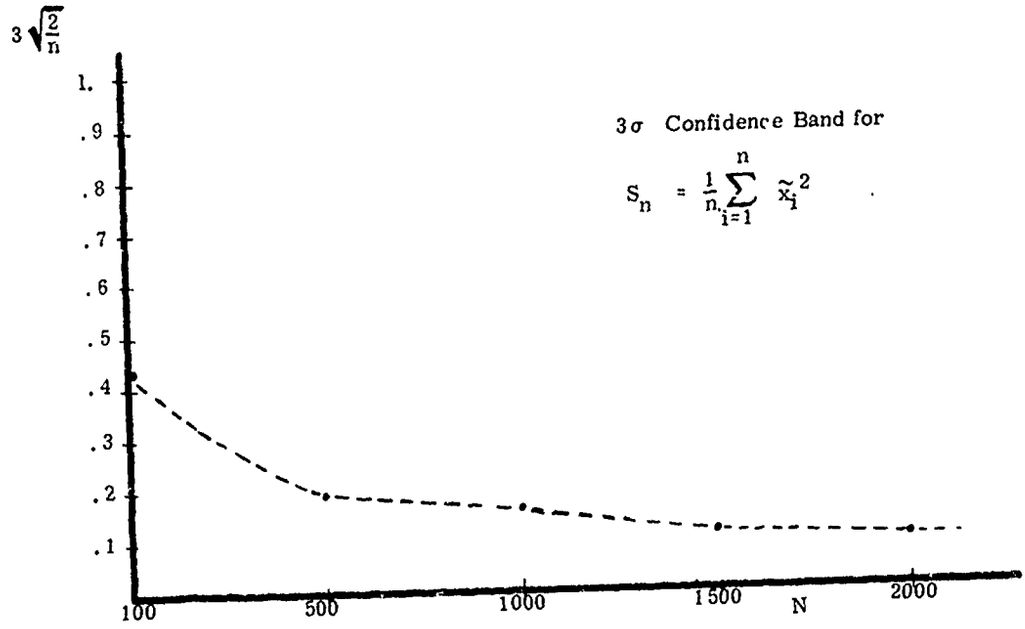


FIGURE I

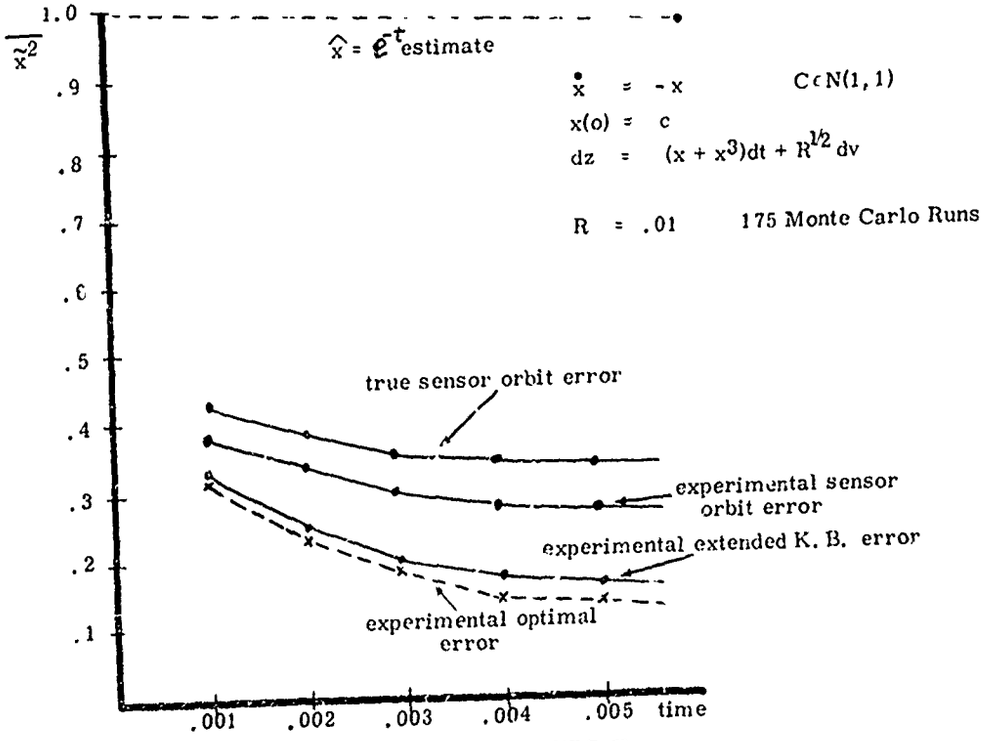


FIGURE II

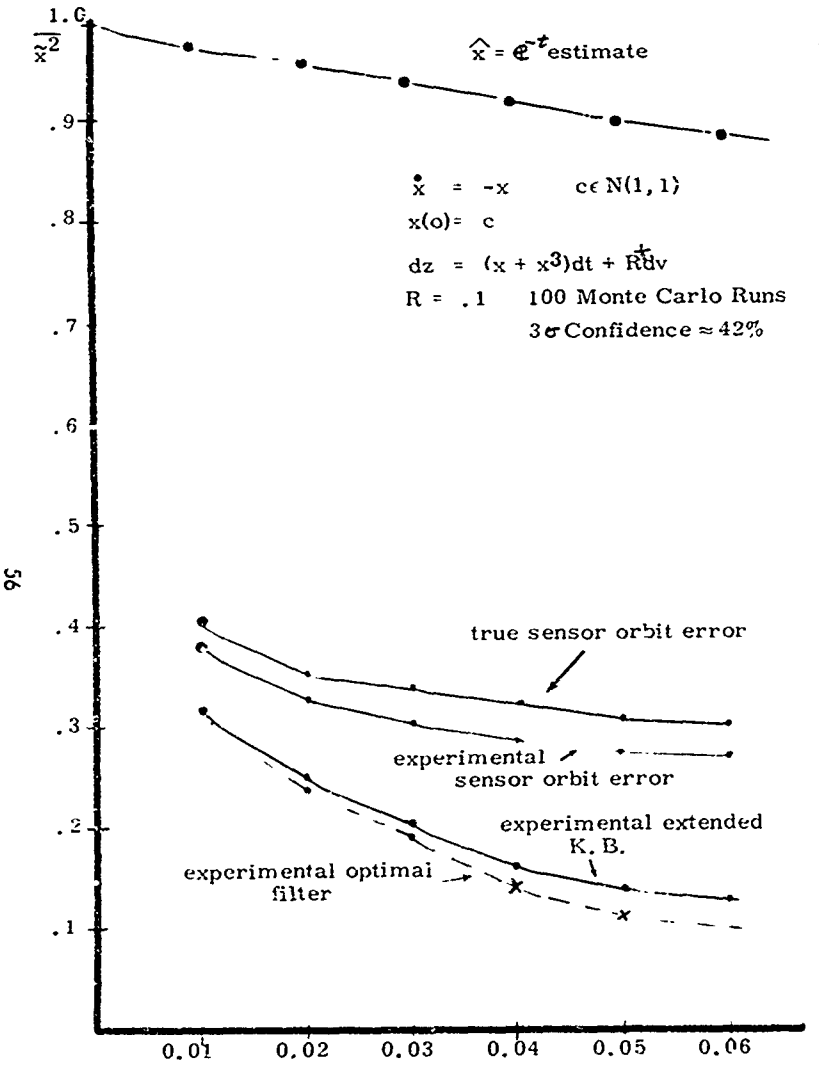


FIGURE III

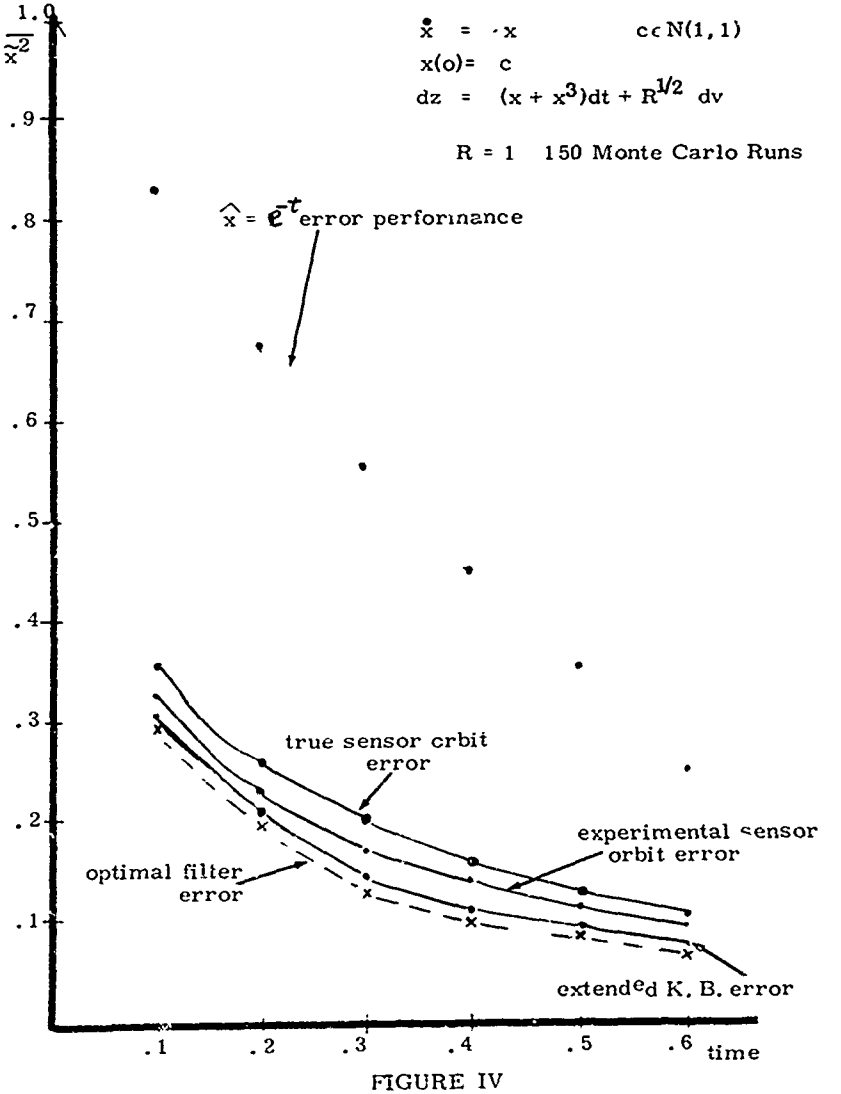


FIGURE IV

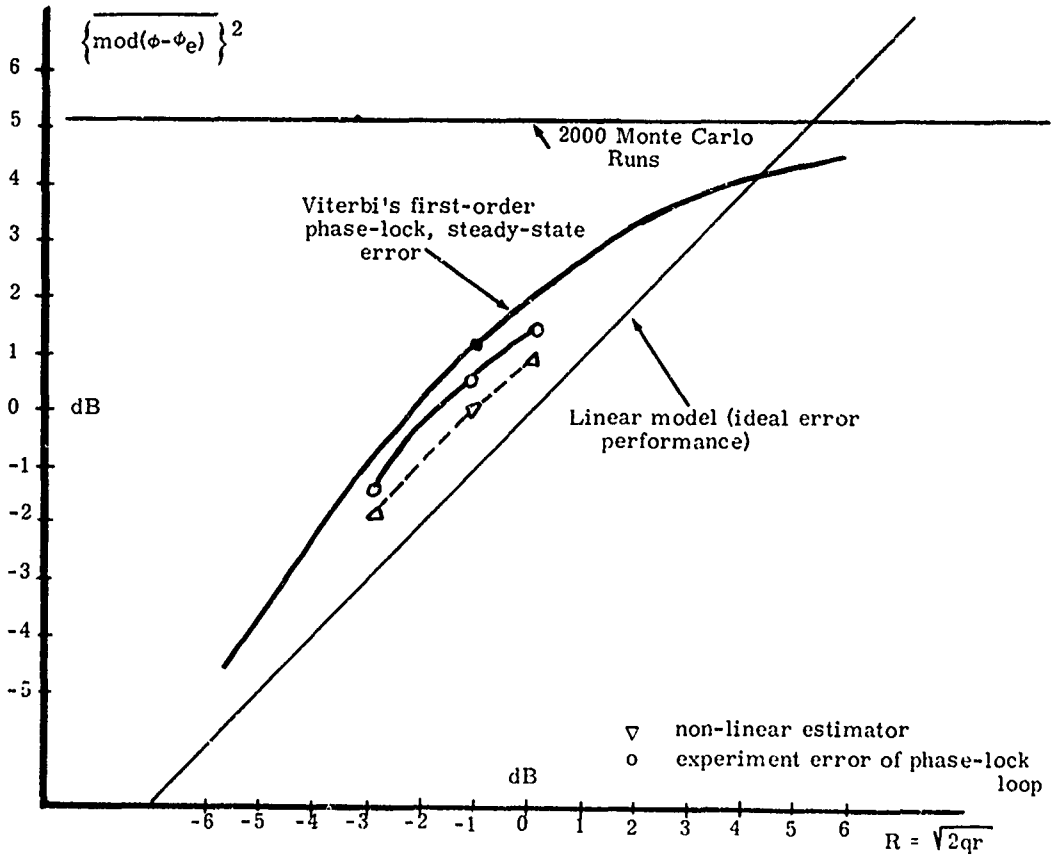


FIGURE V

REFFRENCES

[1] R. L. Stratonovich "Conditional Markov Processes", Theory of Probability and Applications, 1960.

[2] H. J. Kushner, "On the differential equations satisfied by conditional probability densities of Markov processes with applications," J. SIAM Control, Vol. 2, pp. 106-119, 164.

[3] R. S. Bucy" Nonlinear filtering," IEEE Trans. Automatic Control (Correspondence), vol. AC-10, p. 198, April 1965.

[4] R. S. Bucy and P. D. Joseph, "Filtering for Stochastic Processes with Applications to Guidance" New York:Interscience, 1968.

[5] R. E. Kalman and R. S. Bucy, "New results in linear filtering and prediction theory," Trans. ASME, J. Basic Engrg., ser. D, vol. 83, pp. 95-108, 1961.

[6] B W. Licht, thesis, Systems Research Center, Case Institute of Tech., 1970.

[7] A. J. Mallinchrodt, R. S. Bucy, S. Y. Cheng, "A design study for an Optimal Non-linear Receiver/Demodulator, Final Report to N. A. S. A. Goddard Space Flight Center, Contract No. NAS5-10789, August 1970.

[8] R. S. Bucy, K. D. Senne, "Digital Synthesis of Non-linear Filters", Automatica, 7, 1971, pp. 287-298.

[9] J. T. Lo, "Finite Dimensional Sensor Orbits and Non-linear Filtering," U. S. C. Aerospace Engineering Report AE-114, August 1969, to appear I. E. E. E. Proc. on Information Theory.

[10] Calvin Hecht, Thesis, Department of Aerospace Engineering, U. S. C., Nov. 1971.

[11] R. S. Bucy, "Bayes Theorem and Digital Realizations of Non-linear Filters" J. A. A. S., 17, 1970, 80-94.

[12] D. L. Alspach and H. W. Sorenson, "Approximation of Density Functions by a Sum of Gaussians for Non-linear Bayes Estimation," Proc. Sym. on Non-Linear Est., Western Periodicals.

REFERENCES (Cont'd)

[13] E. Licht, private communication, August 1969.

[14] A. J. Viterbi, "Phase-Locked Loop Dynamics in the Presence of Noise by Fokker-Plank Techniques," Proc. IEEE, 51, (1963), 1737-1953.

Reprinted from Proc. Second Symp. on Nonlinear Estimation Theory and Its Applications, San Diego, Sept. 1971, 59-87.

HYBRID COMPUTER SYNTHESIS of OPTIMAL DISCRETE NONLINEAR FILTERS*

R. S. Bucy, M. J. Merritt and D. S. Miller**

Abstract

The total number of digital operations required to generate an estimate using a discrete optimal nonlinear filter is extremely large. This paper delineates the computer task and introduces an algorithm for mechanizing an optimal 1-step predictor using a hybrid computer to reduce the computation time. The advantages and limitations of the hybrid mechanization are described. Hybrid computer predictions (as simulated by a continuous system simulation language) are compared to those generated by an all digital mechanization. Timing estimates, accuracy estimates and computer equipment requirements are given. Extensions of the algorithm to multidimensional systems and observations are discussed. Use of a parallel digital multiprocessor to perform the algorithm is demonstrated. The use of interactive computer graphics for debugging and control of the algorithm is discussed.

1. INTRODUCTION

It has been known for a long time that the optimal, discrete time, nonlinear estimator for systems whose plant dynamics, sensor characteristics and signal statistics are known is just the algorithm obtained using Bayes' Rule to sequentially update the conditional probability density based on the latest data. However, until recently researchers have avoided this numerically exact procedure, preferring instead to develop suboptimal Taylor Series estimators, as for example the extended Kalman-Bucy filter and the minimum least-squares error second order filter. The reason for this is clear. A true optimal filter is a computational juggernaut. In the past few years the availability of 3rd generation high speed digital processors has led to several papers by Bucy (3), Bucy and Senne (4), (5), and Bucy, Senne and Geesey (6) which describe digital computer mechanizations for the required iteration of the conditional probability density function. Both one and two state variable nonlinear problems with Gaussian statistics have appeared in the literature. It is quite clear that an extremely simple 4 state variable filtering problem with a real time data rate of 1 sample per second pushes contemporary high speed sequential processors to the limits of their capabilities.

The hard core of the problem in computing the (n+1)st estimate is the need to re-evaluate the d-dimensional conditional probability density function $J_{n+1}(\underline{y})$ for each piece of data. This requires the evaluation of a d-fold convolution integral:

$$J_{n+1}(\underline{y}) = \int^{R^d} T_n(\underline{y},\underline{\xi}) \cdot J_n(\underline{\xi}) d\underline{\xi} \qquad (1)$$

*This research was partially supported by the U. S. Air Force Office of Scientific Research under Grant AF-AFOSR-1244-67, the U. S. Army Research Office under Grant DA-ARO-D31-124-71-637 and the National Institutes of Health under Grant GM-16197-04.

**The authors are with the University of Southern California, Los Angeles.

If the statistics are Gaussian, $T_n(\underline{y},\underline{\xi})$ is an exponential with vector-matrix argument. $J_n(\underline{\xi})$ is the previous conditional density function evolved to time t_n. If the J_n's are M-ly discretized in all directions then $J_n(\underline{\xi})$ and $J_{n+1}(\underline{y})$ are each defined on a grids containing M^d points. The computation requires M^d evaluations of the integrand for each of the M^d points in the domain of $J_{n+1}(\underline{y})$. Thus one iteration of the algorithm requires $(M^d)^2 = M^{2d}$ exponentiations each of which entails the computation of a multitermed quadratic form. The integration and statistical computations which follow the convolution consume considerably less computer time proportionally. Even for low dimensionality filters: 2 state variable (2-D) plant with linear additive plant and observation noise sequences, Bucy and Senne, (4), introduced rotated coordinates and ellipsoidal mode tracking to achieve a real time data rate of one prediction per two seconds. Suppose the conditional density at time t_n: $J_n(\underline{\xi})$, and the corresponding element of the discrete data sequence $\underline{z}_n$, are available. The computation may be speeded up by generating the exponential portions of the integrand both simultaneously and continuously. This is easily done on an analog computer. However, the analog computer is notably poor at storage of time varying functions of several variables. Analog computers are not well adapted to arithmetic computations, the solution of algebraic equations, etc. Hybrid computers were developed to overcome these problems by combining the digital computer's arithmetic, storage and control functions with the analog computer's ability to solve large numbers of simultaneous differential equations rapidly. As might be expected, hybrid computation introduces new problems. Some of the most important of these are:

(1) How to partition equations between analog and digital computers,

(2) Development of analog representations of filter variables,

(3) Setting of initial conditions,

(4) Effects of limited bandwidth of analog computer components,

(5) Effects of digital to analog and analog to digital conversion times,

(6) Limitations imposed by maximum digital computer I/O data rates,

(7) Delays caused by analog computer mode switching and reset times,

(8) The effect of limited analog computer dynamic range.

These problems are considered in later sections of the paper, and estimates of performance of a high speed hybrid computer (containing a contemporary analog computer and a 3rd generation high speed digital processor) are given.*

The sections below will describe:

(1) The computer task in optimal filter synthesis,

(2) Review the digital computer algorithm proposed by Bucy (Ref. 3) for a one state variable (1-D) problem with linear plant and cubed sensor,

(3) Describe the hybrid algorithm, illustrating its application to the same cubed sensor problem. The basic theory and flow charts are presented.

(4) Some of the details which lead to a discussion of the hybrid algorithm's speed

*In mid-1971 a contemporary analog computer might be an Electronic Associates EAI 680, Applied Dynamics AD-4 or Comcor Ci-5000. The high speed digital processor might be a Control Data CDC 6600 or IBM 360/85.

and accuracy are described.

(5) The latest results obtained for the cubed sensor problem (simulating the hybrid algorithm with a digital continuous system simulation language) are presented.

(6) We will briefly discuss some of the interesting extensions of the above: 2-D and multi-D problems, the effects of nonlinear plant on grid spacing and grid floating; and the use of a parallel digital multiprocessor such as the ILIAC IV to implement the hybrid algorithm.

(7) Preliminary timing estimates for 1-D and 2-D filters for selected hardware configurations are given and compared against similar estimates for the all digital algorithm.

(8) The use of interactive computer graphics for display and debugging of optimal filters and their density functions is described.

(9) Finally, we summarize the advantages and disadvantages of the hybrid algorithm as it compares to the purely digital optimal filter realization and conclude with a description of work in progress.

2. THE COMPUTER TASK IN OPTIMAL FILTER SYNTHESIS

Optimal discrete nonlinear filters are mechanized as an iterative process wherein the same basic procedure is recomputed for each new piece of data. During each iteration, a new conditional density function and a new conditional mean are evolved. Many substeps and numerous calculations are involved. For convenience and comparative purposes, only the one-step predictor is considered here. This section reviews the all digital algorithm of Bucy (3) and its application to a one-dimensional (1-D) cubed sensor problem. Subsequent sections describe the hybrid mechanization.

In the all digital algorithm, for computational purposes, the conditional probability density function is represented as a set of point masses, i.e., a set of M dirac delta functions appropriately positioned and weighted, see Figure (1). In order to obtain the weighting to be assigned to <u>each</u> point in the new grid space, a convolution must be performed over <u>every</u> point carrying positive mass in the old grid space. This convolution is, by far, the most time consuming portion of the calculation and, in essence, is "THE COMPUTER TASK" in optimal filter synthesis. If the running variables of the conditional density function J_n are finely discretized over a large domain, then the computation can become burdensome. If the grid is allowed to "float" at the beginning of each iteration, it need not be of such high resolution nor broad span as would otherwise be required. This leads to a considerable reduction in computer workload. In (3), the grid for J_{n+1} is centered at the mean of J_n, $\hat{x}_n$, and adjusted to span $6\hat{\sigma}_n$ or $8\hat{\sigma}_n$ to either side, where $\hat{\sigma}_n$ is the standard deviation of J_n.

2.1 THE ONE DIMENSIONAL CUBED SENSOR PROBLEM*

The model:

$$\begin{aligned} x_n &= ax_{n-1} + u_{n-1} \\ x_0 &= c \\ z_n &= x_n^3 + v_n \end{aligned} \qquad (2.1)$$

The signal process $\{x_n\}_{n=1,\ldots,m,\ldots,}$ is a discrete time, indexed set of scalar valued random variables with $a > 0$.

The stochastic process $\{u_n\}_{n=1,\ldots,m,\ldots,}$ is a set of independent, normal, identically distributed scalar valued random variables, with variance, q. Hence the conditional density of the $(n+1)$st state, y, given the nth state is:

$$f_{y|x_n=\xi}(\xi) = \frac{1}{\sqrt{2\pi q}} e^{-\frac{1}{2q}[y-a\xi]^2} \qquad (2.2)$$

*A more detailed development of this problem is given by Bucy in (3).

The random variable c is a scalar independent of the u_n process and having a density

$$f_c(\xi) = \frac{1}{\sqrt{2\pi k}} e^{-\frac{1}{2k}(\xi)^2} \tag{2.3}$$

The observation process $\{z_n\}_{n=1,\ldots,m,\ldots,}$ is a scalar valued data sequence. The stochastic process $\{v_n\}_{n=1,\ldots,m,\ldots,}$ is a set of independent, normal, identically distributed scalar random variables, with variance, r, and independent of the u_n process and the random variable c. Hence the conditional density of the nth observation given the nth state is:

$$f_{z_n|x_n=\xi^3}(\xi) = \frac{1}{\sqrt{2\pi r}} e^{-\frac{1}{2r}(z_n-\xi^3)^2} \tag{2.4}$$

Based on this description of the signal and observation processes, a sequential application of Bayes' Rule leads to the following update relationship for the probability density function representing x_{n+1} at time t_{n+1} conditioned on data $z_n,\ldots,z_0$ which we abbreviate as $J_{n+1}(y)$.

$$J_{n+1}(y) \triangleq \int_{-\infty}^{\infty} e^{-\frac{1}{2q}(y-a\xi)^2} e^{-\frac{1}{2r}(z_n-\xi')^2} \times J_n(\xi)d\xi \tag{2.5}$$

where $\triangleq$ means equality to a normalization constant dependent on $z_{n-1},\ldots,z_0$.

There are essentially two problems associated with implementing an optimal nonlinear estimator:

(1) It is necessary to represent the nonparametric conditional density in some practical storage media - the representation problem.

(2) Evaluation of the integral, equation (2.5), - the convolution problem.

2.2 REPRESENTATION

Bucy (3) describes the probability density function as M ordered pairs $\{J_n[\xi_n(i)], \xi_n(i)\}$, $i = 1,\ldots,M$ with: $\xi_n(i)$ a map from the set $\{1,2,\ldots,M\}$ to the reals; $J_n[\xi_n(i)]$ is the corresponding value of J_n a $\xi_n(i)$. Considering the components of J_n as nonnegative masses, $\xi_n(i)$ is the point in R^1 which carries the ith mass $J_n[\xi_n(i)]$.

The continuous probability density function $J_n(\cdot)$, is thus discretely represently by M delta functions of weight $J_n[\xi_n(i)]$ located at $\xi_n(i)$, i=1,...,M.

Now define

$$y_i \triangleq \xi_{n+1}(i)$$
$$\xi_j \triangleq \xi_n(j) \tag{2.6}$$

and (2.5) can be rewritten as a discrete density function:

$$J_{n+1}(y_i) \triangleq \sum_{j=1}^{M} e^{-\frac{1}{2q}(y_i-a\xi_j)^2} e^{-\frac{1}{2r}(z_n-\xi^3)^2} J_n(\xi_j) \tag{2.7}$$

with i = 1, ..., M

where $\sum_{i=1}^{M} J_{n+1}(y_i) \triangleq 1$.

For an initial condition from (2.3),

$$J_0(\xi_j) = e^{-\frac{1}{2k}\xi_j^2} \quad j=1,\ldots,m. \tag{2.8}$$

The gridding $\xi_n(j)$ is centered at mean $\hat{x}_{n-1}$ and spans $N\cdot\hat{\sigma}_{n-1}$ standard deviations to either side where N is determined empirically. Hence, we have

$$\xi_{n+1}(i) = \hat{x}_n - N\hat{\sigma}_n + \frac{N\hat{\sigma}_n}{(M-1)/2}\cdot(i-1)$$

with $1 \le i \le M$ (2.9)

and $\xi_1(i) = \xi_0(i)$ with (from (2.3)) $\hat{x}_0 = 0$ and $\hat{\sigma}_0 = \sqrt{k}$.

With the representation (2.6)-(2.9), the convolution (2.5) is solved for $J_{n+1}(y)$. After normalization, its mean and variance are computed.

2.3 CONVOLUTION

Given the representation (2.6)-(2.9), (2.5) is solved for $J_{n+1}^{U}(y)$ in M sequential integrations each

consisting of M integrand evaluations, where $J^{U}_{n+1}(y)$ is $J_{n+1}(y)$ prior to normalization.

2.4 ESTIMATE COMPUTATION

After normalization to obtain $J_{n+1}(y)$, computation of its mean, $\hat{x}_{n+1}$, and variance $\hat{\sigma}^2_{n+1}$ are straightforward:

$$\hat{x}_{n+1} = \sum_{i=1}^{M} y_i J_{n+1}(y_i)$$

$$\hat{\sigma}^2_{n+1} = \sum_{i=1}^{M} [y_i^2 J_{n+1}(y_i)] - (\hat{x}_{n+1})^2$$

An overall flow chart of the digital algorithm for this problem is shown in figure (2). The integrand evaluation and computation of estimates are shown in figures (3) and (4). An expanded flow chart, figure (5), shows the integrand evaluation for a 2-D optimal 1-step predictor and is included as a further illustration that convolution, and in particular, integrand evaluation is the problem in discrete optimal filtering.

3. THE HYBRID ALGORITHM

The preceding discussion has pinpointed the major obstacle to high speed real time filtering. This obstacle is the need to calculate the conditional density functions and their corresponding argument values M^{2d} times for every piece of new data. For example, in the cubed sensor problem the integrand must be computed M^2 times during the M evaluation of the discrete convolution

$$J_{n+1}(y_I) \cong \sum_{j=1}^{M} e^{-\frac{1}{2q}[y_I - a\xi_j]^2} e^{-\frac{1}{2r}[z_n - \xi_j^3]^2} \times J_n(\xi_j) \qquad (3.1a)$$

which is a discrete approximation to the continuous convolution

$$J_{n+1}(y) \cong \int_{-\infty}^{\infty} e^{-\frac{1}{2q}(y-a\xi)^2} e^{-\frac{1}{2r}(z_n - \xi^3)^2} \times J_n(\xi)d\xi \qquad (3.1b)$$

The hybrid algorithm derives its speed advantage by using the analog computer to perform most of this computation in a parallel and continuous manner.

3.1 THEORY

This section describes the mathematical basis for the hybrid algorithm and its architectures. Particular attention is given to the problems associated with: (1) partitioning the computational tasks, (2) analog and digital representation of variables, (3) analog computer techniques for exponential teneration, and (4) convolution on the analog computer. For convenience, two assumptions are made:

(1) All analog and hybrid components are perfect, i.e., noise, drift static and dynamic errors do not occur. The effects of these errors are treated in Section 4 below.

(2) Analog computer components are represented by their block structural continuous system simulation language counterparts, which do not invert and which do not require amplitude or time scaling.

3.1.1 Partitioning of the Computational Tasks

Before the computational tasks can be assigned to either the digital or the analog computer, a number of important properties of equation 3.1 must be examined:

(1) The integrand of the discrete convolution, 3.1a, may be partitioned into three pieces:

(a) $Y_D(y_I, \xi_j) = e^{-\frac{1}{2q}(y_I - a\xi_j)^2}$

(b) $Z_D(\xi_j) = e^{-\frac{1}{2r}(z_n - \xi_j^3)^2}$

(c) $J_n(\xi_j)$

the first of which depends on y_I while the others do not.

(2) A similar partitioning may be made for the continuous convolution, equation 3.1b:

(a) $Y_C(y,\xi) = e^{-\frac{1}{2q}(y-a\xi)^2}$

(b) $Z_C(\xi) = e^{-\frac{1}{2r}(z_n-\xi^3)^2}$

(c) $J_n(\xi)$

Again, only the first component depends on y.

(3) If r and q are of the same order of magnitude, then the exponential, $Z_C(\xi)$, will vary sharply when $|\xi^3-z_n|$ is small. This will not be true of $Y_C(\xi)$ if $a \leq 1$.

(4) $J_n(\xi)$ is the result of an arithmetic computation at the conclusion of the preceding iteration, or an initial condition. Further $J_n(\xi)$ is stored in tabular form on a grid which will not, in general, be suitable for the next iterative cycle. The values of $J_n(\xi)$ may be obtained by interpolation (or occasionally extrapolation) on the tabular data.

These observations suggest the following partitioning of tasks:

(1) Use the digital computer to compute and store the terms $Z_D(\xi_j)\, J_n(\xi_j)$ for $j = 1,2,\ldots, M$.

(2) Use M similar analog computer circuits to generate the M functions $Y_C(y_I,\xi)$ where $I = 1,2,\ldots,M$ in parallel, where ξ becomes a continuous variable corresponding to $\{\xi_j\}_{j=1,2,\ldots,M}$.

(3) Transmit $Z_D(\xi_j)J_n(\xi_j)$ from the digital computer to the analog computer maintaining the correspondence between ξ_j and ξ.

(4) On the analog computer multiply the M $Y_C(y_I,\xi)$ terms by a piecewise continuous function corresponding to $Z_D(\xi_j)\, J_n(\xi_j)$.

(5) On the analog computer perform a parallel continuous integration of the product obtained in (4) to obtain M unnormalized points of the new conditional density function, $J^u_{n+1}(y)$.

(6) When ξ (on the analog computer) has covered an interval corresponding to the domain of $Z_D(\xi_j)\, J_n(\xi_j)$, halt the analog computation and read the outputs of the M integrators, described in (5) above, into the digital computer.

(7) In the digital computer, normalize and store the new discrete conditional density function $J_{n+1}(y)$ and compute its mean $\hat{x}_{n+1}$ and variance $\hat{\sigma}_{n+1}$.

For the case of Gaussian plant noise, the analog computer's tasks may be separated into three pieces: exponential generation, integrand formation, and integration. The relationships between these three pieces are shown in Figure 6.

3.1.2 Analog Computer Representation of Variables

The representation of $J_n(\xi_j)$ as a set of discrete pairs was described in Section 2, above, and is illustrated in Figure 1. The function $Z_D(\xi_j)$ is stored similarly. $Y_C(y,\xi)$ is, of course, an infinite set of functions, that is, there is a function $Y_C(y_I,\cdot)$ for every fixed value $y = y_I$. M of these functions, $Y_C(y_I,\xi)_{I = 1,2,\ldots,M}$, are generated as continuous functions of time on the analog computer. Four typical $Y_C(y_I,\xi)$ terms are shown in Figure 7. In order to form an integrand corresponding to the discrete terms of (3.1a) $\{Y_D(y_I,\xi_j) \cdot Z_D(\xi_j) \cdot J_n(\xi_j)\}_{J = 1,2,\ldots,M}$ for $I = 1,2,\ldots,M$ or to the continuous terms of (3.1b) $Y_C(y,\xi)\, Z_C(\xi)J_n(\xi)$ the digitally stored terms

must appear in the analog computer as functions of time. Define

$$D_j = \{[Z_D(\xi_j) \cdot J_n(\xi_j)], \xi_j\}_{j = 1,2,\ldots,M} \tag{3.2}$$

to be a set of discrete pairs stored in tabular form in the digital computer. D_j may be converted into a piecewise continuous function of t by introducing

$$Z'_{PC}(t) = \sum_{j=1}^{M} Z_D(\xi_j) J_n(\xi_j) [u(t-t_1)-u(t-t_2)] \tag{3.3}$$

where $t_1 \le t < t_2$

and $t_1 = \Psi_n(\xi_j)$

$t_2 = \Psi_n(\xi_{j+1})$

where the digital representation's running variable ξ_j is mapped into the analog computer's independent variable time, t, by the mapping

$$\xi \xrightarrow{\Psi'_n} t$$

which consists of a scaling and translation. That is:

$$t = \Psi'_n(\xi) = b_n(\xi + c_n) \tag{3.4a}$$

where

$$b_n = t/[2N\hat{\sigma}_{n-1} + 2N\hat{\sigma}_{n-1}/(M-1)] \tag{3.4b}$$

$$c_n = N\hat{\sigma}_{n-1} - \hat{x}_{n-1} \tag{3.4c}$$

For convenience below, the subscripts on b and c will be omitted, and it will be understood that b and c are functions of the step number, n. A typical $Z'_{PC}(t)$ resulting from this mapping is seen in Figure 8. The output process of the digital computer is characterizable as a zero order hold (ZOH) process. The transfer function of the ZOH may be derived from the Laplace transform of its impulse response:

$$G_{ZOH}(s) = \frac{1 - e^{-Ts}}{s}$$

If the sampling interval T (seconds/sample) is small compared to the period of the highest frequency in the signals being processed, then it can be shown*

$$G_{ZOH}(s) \cong e^{-Ts/2}$$

Thus, the use of a ZOH reconstruction at the digital-analog interface results in an almost pure delay of $\frac{1}{2}$ T. For a more detailed discussion of this, see Miura and Iwata [9]. Since $Z_D(\xi_j)$ is known beforehand, it is easy to translate it forward by this amount, cancelling the delay and eliminating a possible source of errors. This translation is accomplished by representing the set of discrete products D_j on the analog computer by the time function

$$Z_{PC}[t] = \sum_{j=1}^{M} Z_D(\xi_j) J_n(\xi_j)\, U(t) \tag{3.5}$$

with

$U[t] = u(t-t_1) - u(t-t_2)$

where $t_1 \le t < t_2$

with $t_1 = \Psi_n(\xi_j - \frac{\Delta\xi}{2})$

$t_2 = \Psi_n(\xi_j + \frac{\Delta\xi}{2})$

$\Delta\xi = \xi_{j+1} - \xi_j$

as shown in Figure 9 with the map

$$\xi \xrightarrow{\Psi_n} t \ .$$

That is,

$$t = \Psi_n(\xi) = b(\xi + c) \tag{3.6}$$

*A factor of 1/T has been eliminated for convenience. The original sampling process contained a constant, T, which cancels it out.

with

$$b = T/[2N\hat{\sigma}_{n-1} + \frac{2N\hat{\sigma}_{n-1}}{(M-1)}]$$

$$c = [N\hat{\sigma}_{n-1} + \frac{N\hat{\sigma}_{n-1}}{(M-1)}] - \hat{x}_{n-1}$$

where b and c are in general, functions of n. This tranformation considers the probability mass represented in the digital computer by a point mass located at ξ_j to be represented on the analog computer by the area under a step function located on an interval whose midpoint is at $t_j = \Psi_n(\xi_j)$. For future use we add

$$\Psi_n^{-1}: \ \xi(t) = \frac{t}{b} - c \qquad (3.7)$$

with b and c as in (3.6).

These definitions differ from those given previously (Eq. 3.4). All subsequent references to Ψ_n, b and c will refer to Eqs. 3.5 and 3.6. As a final point, we note that the analog computer's total run time T, which determines the constants in (3.6), is chosen as the smallest value for which accurate hybrid computation can be performed by a given computer system for a particular filter, see Section 4 below.

3.1.3 Exponential Generation

Having determined that the exponentiation required to develop the Y_C terms of Eq. (3.1) will occur on the analog computer and that a suitable mapping of the variable ξ into time t, is given by (3.6), we turn to a discussion of the actual generation:

$$Y_C[y_I, \xi(t)] \triangleq e^{-\frac{1}{2q}[y_I - a\xi(t)]^2} \qquad (3.8)$$

with I = 1,2,...,M

where $\xi(t)$ is given by (3.7). This is depicted in Section I of Figure 6. For convenience we define

$$e^{W_I(t)} \triangleq e^{-\frac{1}{2q}[y_I - a\xi(t)]^2} \qquad (3.9)$$

and

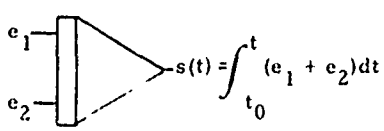
$$W_I(t) \triangleq -\frac{1}{2q}[y_I - a\xi(t)]^2 \qquad (3.10)$$

Generalized integration. Generalized integration is a name given to a method often used to generate explicit composite functions on the analog computer (10), (11). This technique requires:

(1) The derivative of the argument with respect to t be available during the generaoi the function.

(2) The initial condition for the function be supplied for $t = t_0$, where t_0 is the time at which function generation starts.

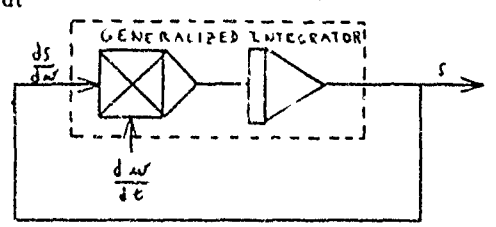
Let s(t) be the output of an analog integration element:



we desire $s(t) = e^w$ differentiating once $\frac{ds}{dw} = e^w$ or $\frac{ds}{dw} - s = 0$. The solution of this auxiliary equation is the desired continuous function. Since M of these functions must be generated simultaneously, and all will depend on their arguments, y_I, it is not possible to esta' lish a correspondence between w and t. In ead, generalized integration is involved which depends on the following application of the chain rule for differentiation of composite functions

$$\int \frac{ds}{dw} \cdot dw = \int \frac{ds}{dw} \frac{dw}{dt} dt \qquad (3.11)$$

If $\frac{dw}{dt}$ is available in the analog computer, then



Application to the state update exponentials. Applying the method to 3.9

$$\dot{W}_I(t) \triangleq \frac{dW_I(t)}{dt} = \frac{dW_I[\xi(t)]}{d\xi} \cdot \frac{d\xi(t)}{dt} \qquad (3.12)$$

from (3.7), $\frac{d\xi}{dt} = \frac{1}{b}$

$$\dot{W}_I(t) = \frac{a}{bq}[y_I - a\xi] \qquad (3.13)$$

This can be expanded to

$$\dot{W}_I(t) = \frac{a}{bq}[y_I - a(\frac{t}{b} - c)] \qquad (3.14)$$

As mentioned above, b and c are functions of n, as are the y_I. To obtain the initial condition $e^{W_I[\xi(t_0)]}$ expand (3.9) using (3.7) with $t = t_0$

$$e^{W_I[\xi(t_0)]} = e^{-\frac{1}{2q}[y_I - a(\frac{t_0}{b} - c)]^2} \qquad (3.15)$$

For the case $t_0 = 0$ this reduces to

$$e^{W_I[\xi(0)]} = e^{-\frac{1}{2q}[y_I - ac]^2} \qquad (3.16)$$

Hardware constraints often preclude starting the generation of the e^{W_I} terms at $t_0 = 0$. This is discussed in Section 3.2.2. The derivative terms $\dot{W}_I(t)$, I = 1,2,...,M shown in Section I of Figure 6 are also generated by the analog computer. The constants necessary for production of these derivative terms, as well as the initial conditions, $e^{W_I[\xi(t_0)]}$, are sent from the digital computer prior to starting the analog solution. See the discussion in Section 3.2 for additional implementation details.

3.1.4 Integrand Formation

Returning to Figure 6, we note that the M terms $\{e^{W_I(t)}\}_{I = 1,2,...,M}$ generated in Section I are each multiplied in Section II by the same term. The output of the digital to analog converter (DAC) in Figure 6 is set to the proper amplitude every Δt seconds under digital computer timing control. The change in representation from the set of M discrete pairs D_j stored in the digital computer to the piecewise continuous function $Z_{PC}[t]$ during a run on the analog computer is based on the map Y_n and constants b and c which are derived from the conditional mean and variance estimates of the previous iterative cycle.

Subsequent sections discuss the manner in which this digital to analog data transmission is performed and some of the problems and limitations associated with it. Assuming $Z_{PC}(t)$ of proper form, the analog computer performs M parallel continuous multiplications as shown in Section II of Figure 6.

3.1.5 Integration

The last step required by the convolution of (3.1) is the M parallel integrations of $\{e^{W_I(t)} Z_{PC}(t)\}_{I = 1,2,...,M}$. The integrations proceed as indicated in Section III of Figure 6, continuously, in parallel and simultaneous with the generation of the $e^{W_I(t)}$ terms and the injection of $Z_{PC}(t)$ which (see Table I) is identical to $J_{n|n}(t)$.

If the observation, z_n, is too far from $\hat{x}_n$, or r is too small, severe scaling problems will occur in outputing $Z_{PC}(t)$ to the analog computer. These rather unlikely events are treated in Section 3.2.2 below. We note also in Section III of Figure 6, that in addition to

$$J^u_{n+1}(y_I) = \int_0^T e^{W_I} Z_{PC}(t)\, dt \qquad (3.18)$$

the analog computer is able to generate

$$y_I J^u_{n+1}(y_I) = \int_0^T y_I e^{W_I} Z_{PC}(t)\, dt \qquad (3.19)$$

and

$$y_I^2 J^u_{n+1}(y_I) = \int_0^T y_I^2 e^{W_I} Z_{PC}(t)\, dt \qquad (3.20)$$

for I = 1,2,..., M

at the expense of some additional equipment. The judgment on whether to do this depends considerably on the computing power of the digital computer and on the analog equipment available, see Section for additional discussion.

3.2 REALIZATION OF THE HYBRID ALGORITHM

There are two steps required to produce an operating hybrid optimal filter: (1) definition of the equations to be solved by each computer, and (2) development of software to link the two computers together realistically. In the process of partitioning, the first requirement was satisfied. The reader familiar with hybrid computation will recognize the myriad of details which must be carefully treated in linking an analog and digital computer together.

A flow chart of the hybrid algorithm (for the cubed sensor problem) is given in Figure 10. The information paths within the hybrid computer systems are shown in Figure 11. The analog computer diagrams for the convolution integral are presented in Figures 12 and 13.

3.2.1 Information Flow in the Hybrid Filter

The hybrid filter is best described by progressing through the flow chart of Figure 10, with occasional references to the flow charts of the purely digital 1-step predictor, Figures 2, 3 and 4. The hybrid flow chart, Figure 10, assumes a single random sequence for its input. In general these will be noisy observations of an on-line stochastic system. The functions performed within each box shown in Figure 10 are:

BOX 1. The digital computer initializes parameters: q - plant noise variance; r - observation noise variance; k - initial state estimate variance; M - grid discretization; N - number of estimated standard deviations, $\hat{\sigma}_{n-1}$, about the (n-1)st estimated mean, $\hat{x}_{n-1}$, spanned by the nth grid (see Figure 1)*; and J_0 - the assumed initial conditional density function. The zeroth grid base is laid out using J_0 as a reference. The observation number, n, is set to -1. The processing performed here is identical to that of BOX 1 in the all digital filter algorithm of Figure 2. The digital computer calculates and transmits to the analog M initial values of y_I - the points at which $J_{n+1}(y)$ will be computed. The M initial conditions $e^{W_I(t_0)}$ are stored in the analog computer's sample and hold elements, as are the initial values of the scaling constant, b_0, and the translation constant c_0. Because of analog computer limited dynamic range (see Section 3.2.2), some initial conditions will be too small to be adequately set on the analog computer at time t = 0. The digital computer determines the time in the solution, t_{0_I}, at which each of the M exponential generators can be accurately started. When this time is reached, the analog gate shown in Figure 12 will be closed and the exponential generation $e^{W_I(t)}$ begun.

BOX 2. The sample sequence number n is incremented. All iterative cycles begin here.

BOX 3. For a simulated system the digital computer performs a model and density update identical to that represented by BOX 3 of Figure 2. In the case of an on-line stochastic input, past data corresponding to the incremented count is assumed to be available. However, it need not actually arrive until required by the computation at BOX 7.

BOX 4. The analog computer is placed in the initial condition (IC) mode. Initial conditions are set into the analog computer integrators. The analog gates in Figure 12 are closed only for those integrators at which $t_{0_I} = 0$. Other circuit configurations may be more desirable. The important

*As indicated in the discussion at BOX 5, the grid base is not changed between iterations unless the computation requires it. Hence, in general, N is the number of estimated standard deviations, $\hat{\sigma}_{n-r}$, about the $(n-r)^{th}$ estimated mean $\hat{x}_{n-r}$ spanned by the nth grid where r is the number of times since the grid was last changed.

point is that limited analog computer dynamic range requires that some integrators start "later" than others because their initial value does not reach an accurately settable value until $t \geq t_{0_I}$ where t_{0_I} is a grid base and problem dependent number (see Section 3.2.2).

BOX 5. A check is made to determine whether the grid should be floated, i.e., whether the mesh or location of the grid base on which the (n+1)st probability density function $J_{n+1}(y)$ is to be constructed should be changed from that on which the nth probability density function, $J_n(\xi)$, was placed. Although much time can be saved in both purely digital and hybrid filtering if the grid base does not change, the percentage savings in hybrid computation are significantly greater. One must, of course, be very careful that the grid mesh is not mislocated or too narrow, causing significant amounts of "probability mass" to be located off the end of the grid. Likewise, a grid which is too course will not be able to properly resolve the probability mass. BOX 5 compares $\hat{x}_n$ with $\hat{x}_{n-r}$ and $\hat{\sigma}_n$ with $\hat{\sigma}_{n-r}$, where r is the number of observations processed since the last grid float, and branches to BOX 6 if either of the differences is too great.

BOX 6. The grid is floated by recomputing the scale factor, b, and translation factor, c, followed by computing M new values for y_I and $e^{W_I(t_0)}$. This information is DAC'd to the analog computer. M new values of t_{0_I}, the time of application of the Ith initial condition $e^{W_I(t_0)}$ to the Ith exponential generator, are computed and stored. Interpolation and possibly extrapolation are required for $J_n[\xi(j)]$ because values of $\xi(j)$ at which its M constituent delta functions are located should correspond with the new values for y_I. For a "smooth" $J_n[\xi]$ a second set of interpolations is required to produce (k-1) x M intermediate values. The k xM discrete pairs representing $J_n(\xi)$, $\{J_n[\xi(\ell)], \xi(\ell)\}_{\ell = 1,2,\ldots,M}$, are stored for future use during the current analog solution as well as for all future solutions on the same grid base. For interpolated values, $J_n(\xi_\ell)$ and $Z_D(\xi_\ell)$, define a set of discrete pairs

$$D_\ell = \{[J_{n|n}(\xi_\ell)], \xi_\ell\}_{\ell = 1, 2, \ldots, k \times M} = \{[Z_D(\xi_\ell) J_n(\xi_\ell)], \xi_\ell\}_{\ell = 1, 2, \ldots, k \times M} \quad (3.2a)$$

stored in the digital computer corresponding to D_J (see Eq. 3.2). If k is chosen odd, the mapping (3.6) may be maintained and the time function appearing on the analog will be

$$Z_{PC}^{(\ell)}(t) = \sum_{\ell=1}^{k \times M} Z_D(\xi_\ell) J_n(\xi_\ell) U_1(t) \quad (3.5a)$$

where U_1 is defined as in 3.5, but with running subscript ℓ. As $Z_{PC}^{(\ell)}(t)$ is identical in form to $Z_{PC}(t)$ shown in Figure 9 (but with increased discretization), the superscript will be suppressed in this sequel. For this kind of intermittent grid float, the span and center of ξ and y are made identical in order that each succeeding $J_{n+1}(y)$ is developed on a grid base suitable for its injection as $J_n(\xi)$ on the next iteration.

BOX 7. For an on-line system there may be a wait for the nth data sample, z_n. Using z_n, $Z_D(\xi_1)$ is computed. The product $Z_D(\xi_1) \cdot J_n[\xi(1)]$ is formed. Under certain conditions (see Section 3.2.2) those values, for $\ell > 1$ (corresponding to $t > 0$), may be computed in parallel with the operations performed in BOXES 8-14 as long as they are ready before the analog computer requires them.

BOX 8. The running index, ℓ, of the piecewise continuous function $Z_{PC}(t)$ (see Eq. (3.5a)), is set to 1. $Z_D(\xi_1)J_n(\xi_1)$, or what is equivalent, $J_{n|n}(\xi_1)$ is DAC'd from the digital to the analog computer.

BOX 9. The analog computer is placed in the COMPUTE mode. This is time t_0, or, as is the custom, $t_0 = 0$.

BOX 10. The analog computer simultaneously and continuously generates $e^{W_I[t]} \triangleq e^{-\frac{1}{2q}[y_I - a\xi(\cdot)]^2}$ for I = 1 to M, where $\xi(t) = \frac{t}{b} + c$, see Eq. (3.7) and Figure 7. In addition, it performs a continuous integration of the product $e^{W_I(t)} Z_{PC}(t)$. When $t = \forall[\xi_\ell + \frac{\Delta\xi_\ell}{2}]$, i.e., the value of t corresponding to the time of injection of the next digitally stored value of $Z_{PC}(t)$, BOX 11 is entered.

BOX 11. A test is made to determine whether the $(k \times M)^{th}$ value of $J_{n|n}(\xi_\ell)$ has been processed.

BOX 12. If $\ell < k \times M$ the index ℓ is incremented by one.

BOX 13. The next digitally stored value of $J_{n|n}(\xi)$, $J_{n|n}(\xi_\ell)$ is DAC'd to the analog computer.

The analog computer remains in COMPUTE while ℓ is incremented from 1 to k x M. During this time, it integrates $\{e^{W_I(t)} \cdot e^{W_Z(t)} \cdot J_n(t)\}_{I = 1, 2, \ldots, M}$. This operation corresponds to the one taking place in BOX 4 of Figure 2, labeled Integral Evaluation. BOX 4 of Figure 2 is expanded in Figure 3. It is the difference in nature of the computation being performed by the hybrid filter in BOXes 9-13 of Figure 12, from that being performed by the digital filter in BOX 4 of Figure 2, that leads to the major differences in speed and accuracy between the two optimal filters.

BOX 14. If the answer to the question posed in BOX 11 is positive indicating that k x M time subintervals have elapsed (or equivalently that t = T), the analog computer is put into HOLD (i.e., integrator inputs are disconnected).

BOX 15. The M unnormalized (n+1)st probability density function points, $J_{n+1}(y_I)$, are read from analog to digital converters (ADC'd).

BOX 16. The digital computer performs normalization to produce $J_{n+1}(y)$, followed by computation of $\hat{x}_{n+1}$ and $\hat{\sigma}_{n+1}$. This calculation is identical to that performed in BOX 5 of Figure 2 and expanded in Figure 4.

BOX 17. A check is made to see if the last piece of data, $z_{n_{MAX}}$, has been processed. If $n < n_{MAX}$, n is incremented (BOX 2) and another iteration of the algorithm occurs. If $n = n_{MAX}$, the exit is taken.

3.2.2 Further Details of the Hybrid Algorithm

The discussion below provides additional insight into the functioning of the hybrid algorithm. Analog and hybrid errors are treated briefly.

Grid floating. If the plant is unstable (a > 1 in Eq. 2.1), or if successive estimates do not converge in phase space, then it is difficult to discretize the conditional density function $J_{n+1}(y)$. If the range on y is made large, then an unreasonable number of discretization intervals results. If the discretization is made coarser, then the accuracy is impaired. The solution to the impasse is to float the grid (see References 3 and 4). The floated grid permits the accuracy of a finely discretized grid to be obtained with a minimum of computer time, by placing the discretization where it will do the most good. In the purely digital

computation if the grid were fixed (i.e., not floated), then the computations described above could be simplified considerably. The functions

$$e^{-\frac{1}{2q}(y_I - \xi_j)^2}$$

$$e^{-\frac{1}{2r}(z_n - \xi_j^3)^2}$$

could be precomputed and stored in large arrays. Some interpolation will be required for the function which contains z_n and the arrays may be quite large. Equivalently, in the hybrid algorithm, the exponential generators, $Y_C(y_I, \xi)$, could be established for the entire observation sequence. The sample and hold elements shown at the output of the digital to analog converters (DAC's) in Figure 13 would not be necessary. When the range of y in $J_{n+1}(y)$ is known in advance and is not too large, acceptable accuracy has been obtained using both floated and nonfloated grids. As the range of y increases, the difference in computation time for each estimate on the floated and nonfloated grid becomes larger, at equivalent accuracy. However, those problems which can be represented by a cyclical, or modulo-2π density function not requiring high local resolution such as the phase lock loop error, see (12), can be represented on a fixed grid.

Filter errors at low data rates. Analog computer components are subject to a variety of low frequency and static errors

(a) noise
(b) offset
(c) drift
(d) nonlinear behavior

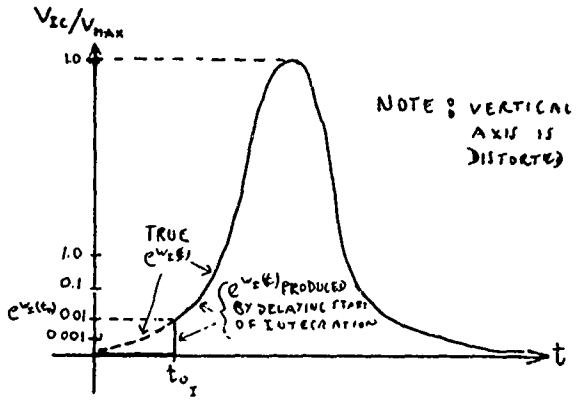
Contemporary, 3rd generation analog computers(19) are available with low frequency and static accurancies on the order of 0.01%. As the number of components used in a problem increases, these errors are compounded. Further, the range of useful computation on such analog computers is $\pm$ 10 volts. If the noise level is approximately 1 to 10 millivolts, then the useable dynamic range of an analog computer is 1 part in 10^4 or 10^3.

This limited dynamic range should be contrasted with 1 part in 10^{150} available in the IBM 360 series machines, and the even greater dynamic range available from CDC computers. The density function of a normally distributed random variable drops three decades in 3.2 standard deviations either side of the mean. It is not uncommon for an optimal filter to span 12 or 16 standard deviations. Digital computers easily generate density functions spanning 72 standard deviations ($\pm$ 36 deviations on either side of the mean). Consider the exponential generator in Figure 6.

To generate a "bell" on the analog computer, an initial condition is needed which corresponds to a point on one of its "skirts". If we were to attempt to set all initial conditions at the beginning of an analog computer run for the 1-D cubed sensor problem with a = 0.5, fully one half of all these initial values correspond to points three or more standard deviations from the mean. That is, half of all initial conditions would have to be set to an accuracy below the analog computer's dynamic range. To avoid large errors in the bells, the start of integration is delayed until $Y_C(y_I, \xi)$ reaches a value which can be accurately set at the integrator's output, approximately 10 MV in a 10 V full scale analog computer. This value is stored in the sample and hold device until the time corresponding to the initial condition is reached, see Figure 12. The time of application t_0 of an IC is calculated using Eq. 3.15 with

$$e^{W_I[\xi(t_0)]} = .001.$$

A signal $\lambda_{\xi_i}(t)$ from the Ith digital output point (DO) in the discrete data control unit (see Figure 11) releases the integrator. If all of the decades of precision were visible in a graph, then the exponential function and its corresponding analog computer equivalent might appear as follows:



In Section 5 we report results of a hybrid optimal filter simulation in which the ratio of V_{IC} to V_{MAX} is varied. At the output circuit the principle source of low frequency errors is located at the integrator input where small observation noise variance or a data point z_n corresponding to an x_n far from $\hat{x}_n$ lead to a needlelike $J_{n|n}(t)$. Multipliers are built such that they produce full scale output when full scale input is applied to both inputs. In terms of threshold voltages (V_{TH}), if the full scale voltage is 10^4 V_{TH} then the product of two inputs, each 100 times V_{TH}, produces only a threshold voltage output. Since contemporary integrators have excellent equivalent input drift and noise characteristics, the primary problem encountered in this circuit is to obtain multiplier outputs which are above V_{TH} for significant inputs. Assuming that the exponential generators are always set to achieve full scale voltage at $a\xi = y_I$, only the scaling or method of multiplying by $J_{n|n}(t)$ may be varied, to avoid loss of accuracy. The set D_ℓ can be totally precomputed and rescaled (at BOX 7 of Figure 10) or alternatively multiplying DACs instead of multipliers can be used for output multiplication. Both slow the computation down and the latter increases cost as well. It is also possible to bypass the analog computer and perform a simple analytical computation corresponding to a stochastic state update for the remote data point case. See (13) for a discussion of the case where $x_n \approx z_n^{\frac{1}{3}}$.

Filter Errors Associated with High Speed Hybrid Computation. Having discussed some of the problems which arise if we can assume the analog computer has an amplitude accuracy of 1 part in 10^3 or 10^4, it is important to determine when this assumption can be made. Above a certain data rate, finite analog computer component bandwidths and nonzero data transmission and mode switching times will so deteriorate analog computer performance as to invalidate accuracy analysis based on purely static errors. Component bandwidth limits the fidelity with which the exponential generators can produce $Y_C(y_I, \xi)$ and the output multipliers can produce

$$e^{W_I(t)} J_{n|n}(t).$$

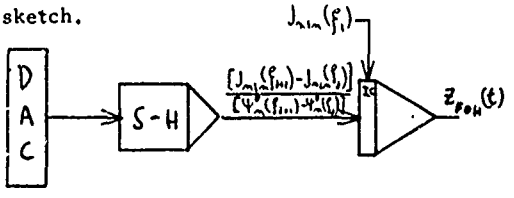
In Section 4 we present analytical results in which the exponential integrator is replaced by a second order system representing its high frequency equivalent circuit. The output multiplier again may be approached more directly. In Section 4 we present analytical results considering two limiting cases for the form of $J_{n|n}(t)$.

Intercomputer maximum data rates and hybrid system switching times are a second source of errors in high frequency hybrid filtering. The hybrid optimal filter has the advantage that for low dimensional problems, i.e., $d \leq 3$, $J_{n|n}(\xi_\ell)$ may be computed in its entirety by the digital computer prior to convolution on the analog computer. That is, the two computers can operate essentially sequentially. This can be seen in Figure 10 where for $n \geq 0$ the digital computer performs computations during BOXes 16, 17, 2, 3, 5, (6), and 7 followed by the analog performing the computation in BOX 10. The linkage system and the digital computer do not have to work very hard while the analog computer solves simultaneous differential equations at high speed. This situation does not prevail in closed loop hybrid computation. Maximum channel data transfer rates place an obvious upper bound on filter data rate. For example, to produce $J^u_{n|n}(y)$ on the analog computer in 1 MS with $\ell = 100$ $J_{n|n}(\xi_\ell)$ must be transmitted at a 100 KH rate. That is, every 10 μs, a word located in computer memory must be transmitted to a register at an interface (see Figure 11) and converted to an

equivalent stable analog voltage available at the DAC output shown in Figures 6 and 12. Although higher speed equipment is available, maximum channel data rates appear to set 1 MS as a lower bound for a single convolution on the analog if $Z_{PC}(t)$ is of the form shown in Figure 9.

Even though the analog computer performs the convolution in an essentially open loop manner, at high repetition rates, phase shifts caused by timing imprecision (i.e.,'earliness or lateness) in operation of integrators, sample and hold mode control switches, and analog gates are an additional cause of inaccuracy.* Specifically (see Figures 11, 12, and 13), these errors are caused by the limits on timing precision that the $\lambda_{E_I}(t)$ signals can close the analog gates, and the high speed DAC can supply $J_n(t)e^{W_Z(t)}$ with respect to each other and with respect to the time origin set by the COMPUTE mode control switch which starts the time base integrator. We must add to this timing skew and the effects of deteriorated signal waveforms of the pulses transmitted through the linkage system to initiate these operations. It can be shown, that a timing imprecision of only 16 nanoseconds produces a .01% F.S. amplitude error for a sine wave with 1 MS period. We defer detailed examination of this point until Section 4 and mention only that timing imprecision is a not well advertised hybrid computer characteristic and that a 300 nanosecond switching grey band is a more reasonable value to expect for contemporary analog computers with 900 ns - 1 μs total switching times.

Interpolation of $J_{n|n}(\xi)$. As a final detail we discuss the advantages and difficulties of injected function interpolation. First note that no matter how it is represented in the digital or analog computers $Z_D(\xi)J_n(\xi)$ or $J_{n|n}(\xi)$ is a continuous function. The $Y_C(y_I,\xi)$ terms generated on the analog are also continuous. Therefore interpolation of $J_n(\xi_j)$, given accurate values at ξ_j will increase the accuracy of computation. Results given in Section 5 demonstrate this. However interpolation lengthens digital computation time and overlapping the computation leads to output scaling problems discussed above. In addition, as mention, the greater the number, $\ell = k \times M$, the more data which must be transferred between computers per analog run and the larger the minimum T as determined by maximum channel data rate must be. Another approach is the use of a first order hold FOH. This device (see (16)) essentially performs linear interpolation on the data after transmission It does not require a $\frac{\Delta\xi_j}{2}$ advance and requires less transmitted data points for equivalent amplitude accuracy. Only a simple circuit is required, see sketch.



FIRST ORDER HOLD

4. THE EFFECTS OF HIGH DATA RATES ON HYBRID FILTER ACCURACY

As mentioned, there is some data rate, above which, analog computer static errors are no longer reliable guides to solution accuracy. In this section we consider for the scalar filter just what the tradeoffs are. Again the cubed sensor problem serves as an example.

4.1 ANALOG COMPUTER COMPONENT BANDWIDTH

4.1.1 Exponential Generator

Under certain reasonable assumptions an operation amplifier can be represented by the following

*For a discussion of this point see (14) and (15), where the nature and effects of H. Y. Brid's law also called the "Magic Equation" are discussed.

transfer function

$$G(s) = -\frac{z_f}{z_i} \cdot \frac{1}{1 + \frac{1}{A\beta}} \qquad (4.1)$$

where A = open loop amplifier GAIN

$\beta = z_i / (z_i + z_f)$

z_f = feedback network short circuit transfer impedance

z_i = input network short circuit transfer impedance

For A >> 1, the usual approximation

$$G(s) \approx -\frac{z_f}{z_i}$$

results. At high frequencies it is necessary to replace A with $A(j\omega)$ to evaluate operational amplifier performance. $A(j\omega)$ is, in general, a complex transfer function with 3 or more break points. Simplified analysis valid for some operational amplifiers assumes a single pole transfer function

$$\frac{A_0}{\tau_2 s + 1} \qquad (4.2)$$

Substitution of 4.2 into 4.1 and setting $\frac{z_f}{z_i} = \frac{1}{\tau_1 s}$

$\left(\text{WHERE } \begin{matrix} z_f = \frac{1}{Cs} \\ z_i = R \end{matrix} \Rightarrow \tau_1 = RC \right)$

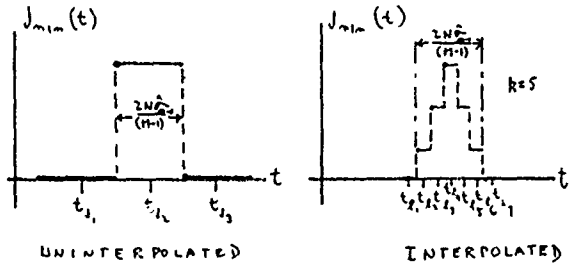
for an integrator yields a second order transfer function containing two widely separated poles. The low frequency pole represents the physical fact that the amplifier gain is not infinite at DC.

The location of the high frequency pole determines the effect of limited bandwidth on hybrid filter performance. For $A_0 = 10^8$, $\tau_1 = 1/2000$, $\tau_2 = 10$, $N = 8$, $T = 4$ MS, $\hat{\sigma}_n^2 \approx q$, $\hat{x}_n \approx 0$

It can be shown that the highest frequency of interest in $\overset{\circ}{W} e^{w_I(t)}$ will not produce an error in excess of .01%. Since the exponential generators are nonlinear, this analysis can only be considered a rough guide. See (26) for results of a simulation of amplifier limited bandwidth in an exponential generator.

4.1.2 Output Multiplier

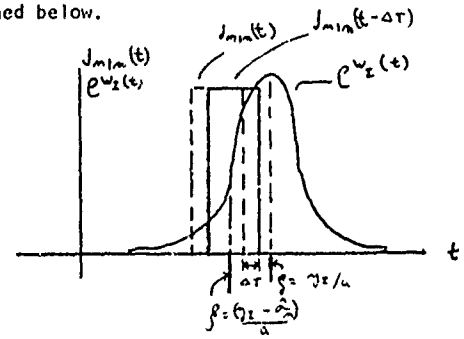
At the output multiplier $e^{w_I(t)}$ is multiplied by $J_{m|m}(t)$ which has two possible worst case configurations



If we assume the fundamental and third harmonic contain all terms contributing significant mass to $J^u_{m+1}(y_I)$ and use manufacturer's multiplier specifications, a 4 MS convolution time, T, leads to errors less than .02%. Results of a simulation of this portion of the circuitry will also appear in (26).

4.2 HYBRID SYSTEM TIMING IMPRECISION

This limitation may be examined by considering the mass shift in $J_{m+1}(y)$ caused by the worst case sketched below.



The boxcar function corresponds to the uninterpolated $J_{m|m}(t)$ worst case considered above; the bell is $e^{w_I(t)}$. $de^{w_I(t)}/d\hat{\sigma}_n$ can be obtained from normalized tables. For the cubed sensor problem with T = 4 MS, a = 0.5, $\hat{\sigma}_n^2 \approx q$, ΔT in sketch above, of 300 nanoseconds (see section 3.2.2) we again find that the errors in $J^u_{m+1}(y_I)$ are less than .02%.

A similar sort of analysis in which an ideal $J^u_{m+1}(y_I)$ is compared against the $J^u_{m+1}(y_I)$ produced when $\lambda_{t_I}(t)$'s delayed by 300 NS also yields errors in $J^u_{m+1}(y_I)$ of less than .02%.

5. SIMULATION RESULTS

A simulation of the hybrid algorithm for optimal discrete one step prediction using a modified version of MOBSSL, a block structured continuous system simulation language developed at the University of Southern California (17), (18) is underway. The modifications to MOBSSL provide an ability to simulate analog computer limited dynamic range and make double precision arithmetic available for highly accurate simulation. Two types of results are available at present.

5.1 VERIFICATION OF THE ALGORITHM

Table 2 compares for 6 data points the estimated means and variances produced by a simulated hybrid filter against results obtained for the all digital filter on the cubed sensor problem. For both filters: $k = q = r = 1.0$, $a = 0.5$ and $N = 8$. For the hybrid filter $M = 29$ and $k \times M = 319$. A dynamic range of $10^{1'}$ .t was used in this simulation. Digital filter ... discretizations were $M = 29$ and $M =$. Digital filter runs of the form described by Bucy (13) to obtain $\hat{x}_n(M)$ and $\hat{\sigma}_n^2(M)$, have shown that for this problem, $M = 513$ is a sufficiently great discretization to produce estimates which do not differ in the 4th place past the decimal point. Note that the hybrid filter estimates are closer to the highly discretized digital filter estimates than those of the digital filter with $M = 29$.

5.2 EFFECTS OF ANALOG COMPUTER LIMITED DYNAMIC RANGE

Results from a simulation in which analog computer dynamic range was varied are presented in Table 3. Mean and variance estimates obtained from the hybrid filter and a highly discretized all digital filter for three data points are shown. The ratio of V_{MAX} to V_{TH} is varied between 10^2 and 10^{10}. At dynamic ranges of 10^3 or greater the simulated hybrid filter estimates are in close agreement with the purely digital estimates. For both filters: $k = q = r = 1$, $a = 0.5$, $N = 8$. For the all digital filters $M = 513$ and $M = 29$. The hybrid filter grid discretization was $M = 29$ with no interpolation, i.e., $k \times M = 29$.

6. USE OF INTERACTIVE GRAPHICS IN OPTIMAL FILTERING

The USC System Simulation Laboratory contains a Adage AGT-10 graphic computer and IBM-360-44 medium speed scientific computer and a Beckman Ease 2132 Analog Computer. A hybrid linkage connects the computers. Current software permits the graphic computer to be used standalone or in conjunction with the 360 in any of several modes.

At present programs have been written which read a sequence of stored 2-D conditional density function from disk and display them in various ways on the graphic terminal screen. Most of these programs are written in the AGNOS graphic programming language (20).

These display programs can be used to evaluate filter performance by indicating temporary loss of observation data, singularities in the model equations, grid inadequacy, poor modeling, or very poor prior knowledge of state. The display for one of these programs is shown in Figure 14. Any of the 2-D density functions on the periphery may be displayed in the center. A sequence of mounds may be displayed at center, using a "joystick" to control the sequencing rate. The density functions may be blended from one to the next. The graphic output of a program capable of displaying any preselected density function is given in Figure 15. The graphic computer screen, its function switches and joystick are foreground. The IBM-360 is in background.

7. SUMMARY AND CONCLUSIONS

The hybrid algorithm for optimal nonlinear discrete filtering has been presented. Timing estimates indicate that the hybrid algorithm will require 10 seconds for one state estimate in a 4-D task. This compares to 3 seconds for 49 CDC 6600's operating in parallel. Thus, one hybrid computer with 250 integrators and multipliers operates at speeds equivalent to 49 CDC 6600's. The relative disadvantage is accuracy. High speed hybrid filter estimates will generally be within 1%.

The effects of analog and hybrid errors are discussed and a more thorough treatment is in progress [Reference 26].

Application areas for the hybrid optimal filter are:

(1) high data rate, moderate accuracy tasks, i.e., .1% to 1% accuracy is acceptable,
(2) continued illumination of optimal nonlinear filtering with a view towards more general theory,
(3) establishing estimates of moderate accuracy as a reference for suboptimal filter estimates in high dimensional tasks.

ACKNOWLEDGMENT

The authors wish to express their gratitude to Dr. R. E. Kaplan and M. Asa for system programming support and programming assistance at all levels and to R. Klement, W. Liles and A. Vreeland for assistance in the graphic portions and to J. Roth who typed the manuscript.

TABLE 1 HYBRID FILTER NOTATION*

The Bayesian Sequential Estimator:

$J_{a|b}(\cdot) \triangleq$ Conditional Density of random variable x_a given observations $z_0, z_1, \ldots, z_b$

The following notational simplification is used:

If a = b + 1 conditional subscript b is suppressed, e.g., $J_{n+1}(\cdot) \triangleq J_{n+1|n}(\cdot)$

$$J_{n+1}(y_I) = \sum_{j=1}^{M} Y_D(y_I, \xi_j)\, Z_D(\xi_j)\, J_n(\xi_j)$$

$$J_{n+1}(y) = \int_{-\infty}^{\infty} Y_C(y, \xi)\, Z_1(\xi)\, J_n(\xi)\, d\xi$$

1	$Y_C(y,\xi) = e^{-\frac{1}{2q}(y-a\xi)^2}$	Unnormalized Conditional Density of x_{n+1} given x_n
1.1	$Y_C(y_I,\xi) = e^{-\frac{1}{2q}(y_I-a\xi)^2}$	Analog Representation of 1 for fixed $y = y_I$
1.2	$Y_D(y_I,\xi_j) = e^{-\frac{1}{2q}(y_I-a\xi_j)^2}$	Discrete Representation of 1
2.	$Z_D(\xi) = e^{-\frac{1}{2r}(z_n-\xi^3)^2}$	Unnormalized Conditional Density of z_n given x_n
2.1	$Z_D(\xi_j) = e^{-\frac{1}{2r}(z_n-\xi_j^3)^2}$	Discrete Representation of 2
3	$J_n(\xi)$	Conditional Density of Random Variable x_n given Observation $z_0, z_1, \ldots, z_{n-1}$ a continuous function
3.1	$J_n(\xi_j)$	Discrete Representation of 3
4	$D_j = \{[Z_D(\xi_j) \cdot J_n(\xi_j)], \xi_j\}_{j=1,2,\ldots,M}$	
4.1	$D_\ell = \{[Z_D(\xi_\ell) \cdot J_n(\xi_\ell)], \xi_\ell\}_{\ell=1,2,\ldots,k\times M}$	
5	$Z_{PC}(t) = \sum_{j=1}^{M} Z_D(\xi_j) J_n(\xi_j) U(t)$	Piecewise Continuous Function Corresponding to 4
	with $U(t) = u(t-t_1)-u(t-t_2)$ where $t_1 \le t < t_2$ $t_1 = \Psi_n(\xi_j - \frac{\Delta\xi}{2})$, $t_2 = \Psi_n(\xi_j + \frac{\Delta\xi}{2})$, $\Delta\xi = \xi_{j+1}-\xi_j$, Ψ_n defined in (3.6)	
5.1	$Z_{PC}^{(\ell)}(t) = \sum_{\ell=1}^{k\times M} Z_D(\xi_\ell) J_n(\xi_\ell)\, U(t)$	Piecewise Continuous Function Corresponding to 4.1
	For k odd define U(t) as in 5 with j replaced by ℓ	
6	$J_{n\|n}(\xi) \triangleq Z_D(\xi)\, J_n(\xi)$	(See Reference 8, Eq. 2.5)
6.1	$J_{n\|n}(\xi_j) \triangleq Z_D(\xi_j)\, J_n(\xi_j)$	Discrete Representation of 6

*Unless otherwise specified in this table or text, the notation is that used in (3), (4) and (8).

7	$e^{W_I(t)} \triangleq Y_C(y_I, \xi)$	Alternate form of 1.1
8	$e^{W_Z(t)} \triangleq \sum_{j=1}^{M} Z_D(\xi_j)\, U(t)$	Piecewise Continuous Function Corresponding to 2.1
9	$J_n(t) \triangleq \sum_{j=1}^{M} J_n(\xi_j)\, U(t)$	Piecewise Continuous Function Corresponding to 3.1
10	$J_{n\mid n}(t) \triangleq \sum_{j=1}^{M} Z_D(\xi_j)\, J_n(\xi_j)\, U(t)$	Piecewise Continuous Function Corresponding to 6.1
	Note: $J_{n\mid n}(t) \triangleq Z_{PC}(t)$	
11	$J^u_{n+1}(y_I) = \int_0^T Y_C Z_{PC}\, dt$	Unnormalized I^{TH} Point of (n+1)st Conditional Density Function $J_{n+1}(y)$

TABLE 1 (CONTINUED)

TABLE 2 COMPARISON OF ESTIMATES FROM TWO ALL DIGITAL OPTIMAL FILTERS WITH THOSE FROM SIMULATION OF AN IDEAL HYBRID OPTIMAL FILTER

k	ESTIMATED MEANS $\hat{x}_{k+1}$			ESTIMATED VARIANCES $\hat{\sigma}^2_{k+1}$		
	DIGITAL		HYBRID	DIGITAL		HYBRID
	M = 29	M = 513	M = 29	M = 29	M = 513	M = 29
0	.855	.825	.825	1.005	1.004	1.004
1	-.026	-.022	-.022	1.093	1.092	1.092
2	.334	.333	.333	1.106	1.097	1.096
3	-.110	-.100	-.100	1.099	1.095	1.095
4	.145	.159	.159	1.089	1.096	1.095
5	-.012	-.015	-.015	1.082	1.085	1.084

TABLE 3 EFFECT OF ANALOG COMPUTER LIMITED DYNAMIC RANGE ON HYBRID FILTER ACCURACY

k	ESTIMATED MEANS $\hat{x}_{k+1}$						ESTIMATED VARIANCES $\hat{\sigma}^2_{k+1}$					
	DIGITAL		HYBRID				DIGITAL		HYBRID			
	M=513	M=29	10^2:1	10^3:1	10^4:1	10^{10}:1	M=513	M=29	10^2:1	10^3:1	10^4:1	10^{10}:1
0	.825	.855	.845	.854	.844	.854	1.004	1.001	.869	1.003	1.009	0.989
1	-.022	-.026	-.036	-.039	-.045	-.020	1.092	1.0[illegible]	1.005	1.090	1.091	1.091
2	.333	.334	.389	.321	.320	.342	1.097	1.106	.979	1.098	1.096	1.104

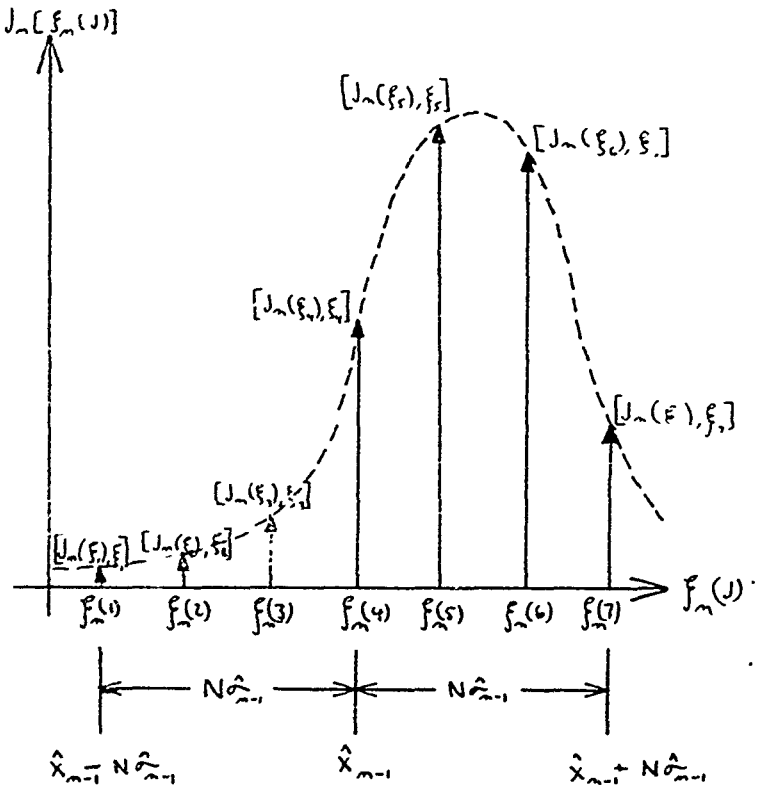
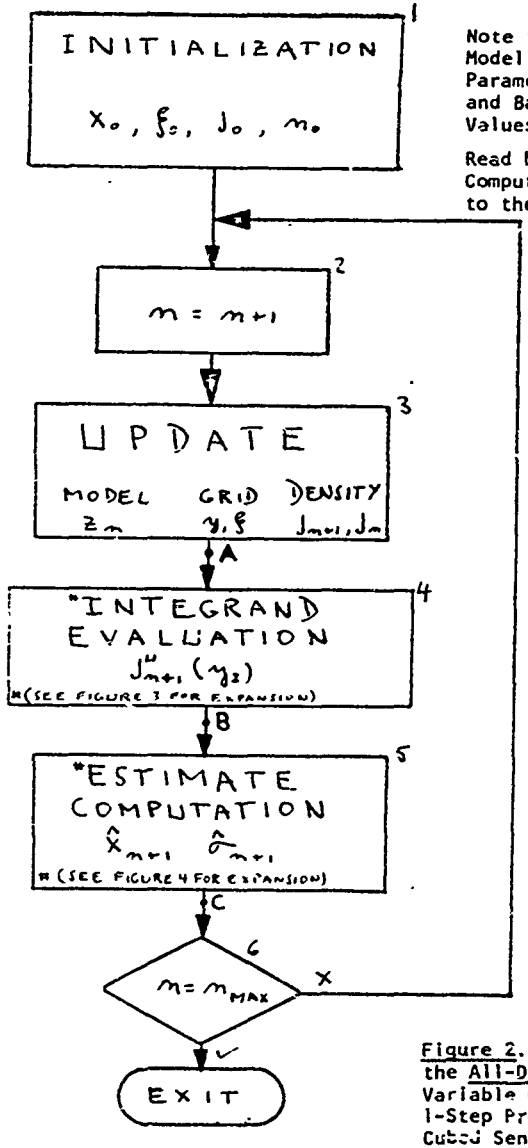


Figure 1. Representation of the nth Conditional Density Function $J_m[f_m(J)]$ by a Set of M Dirac Delta Functions on a Grid Spanning $2N\hat{\sigma}_{m-1}$ Centered at $\hat{x}_{m-1}$ for the Case M=7 for a 1-State Variable Problem.



Note to BOX 1:
Model and Statistics Parameters - a,k,q,r and Basic Computation Values - N,M,n_{MAX} are Read by the Digital Computer as Data Prior to the Processing in BOX 1.

Figure 2. Flow Chart of the All-Digital 1-State Variable Optimal Discrete 1-Step Predictor for the Cubed Sensor Problem.

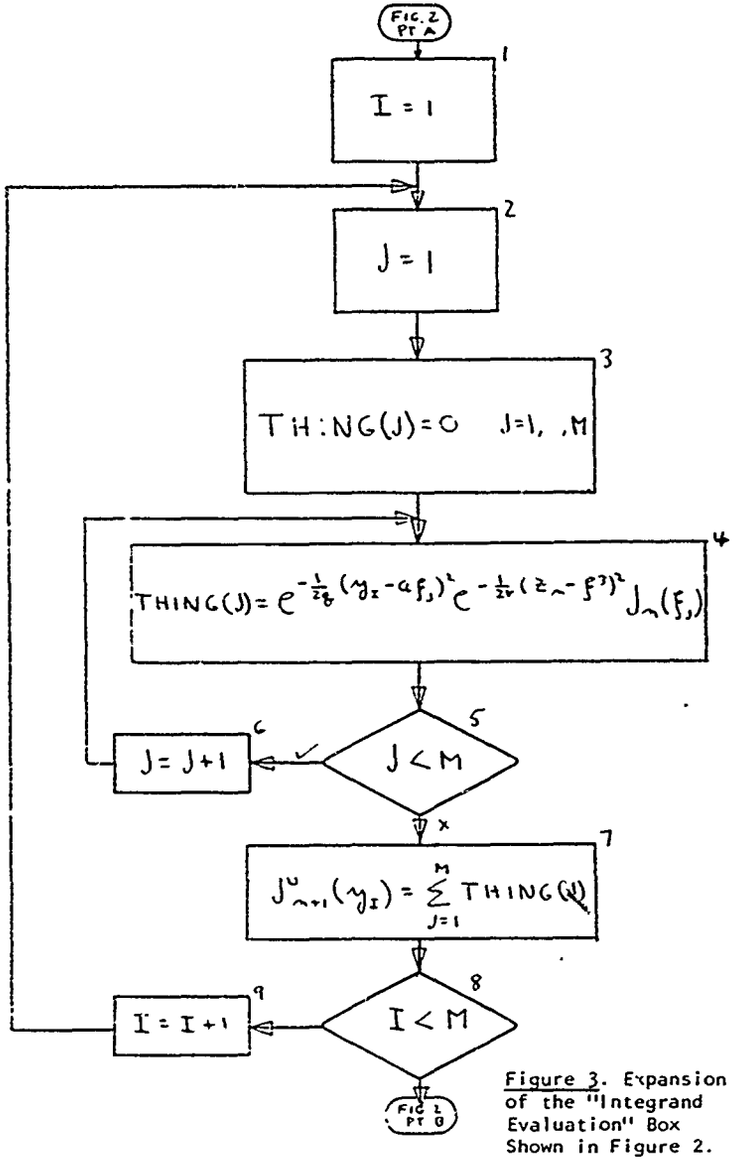


Figure 3. Expansion of the "Integrand Evaluation" Box Shown in Figure 2.

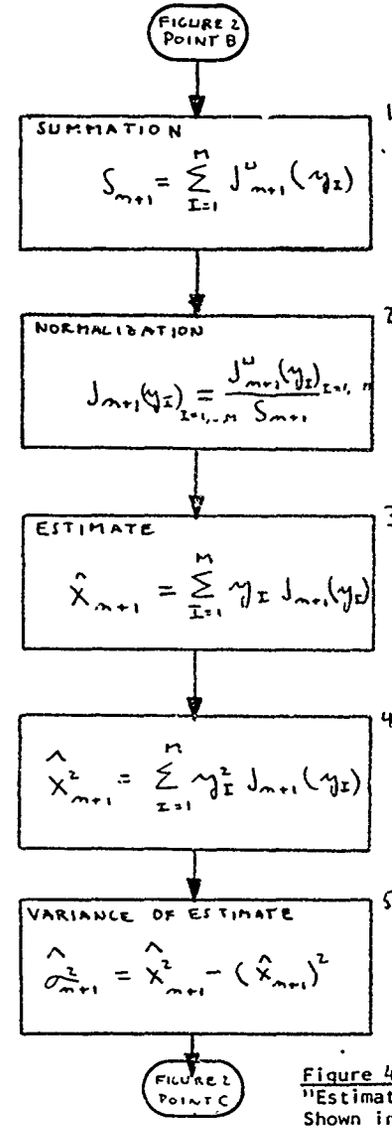


Figure 4. Expansion of the "Estimate Computation" Box Shown in Figure 2.

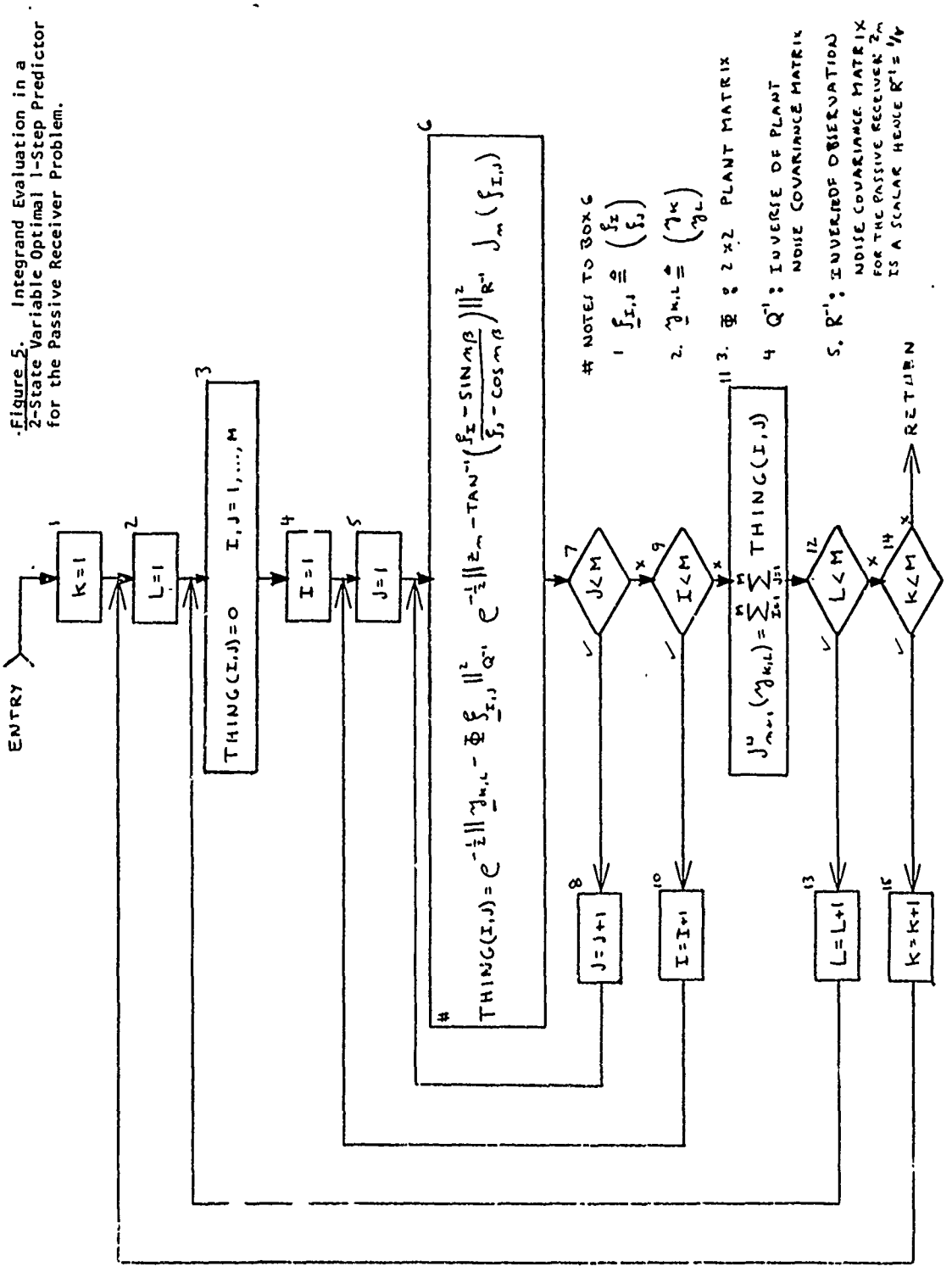


Figure 5. Integrand Evaluation in a 2-State Variable Optimal 1-Step Predictor for the Passive Receiver Problem.

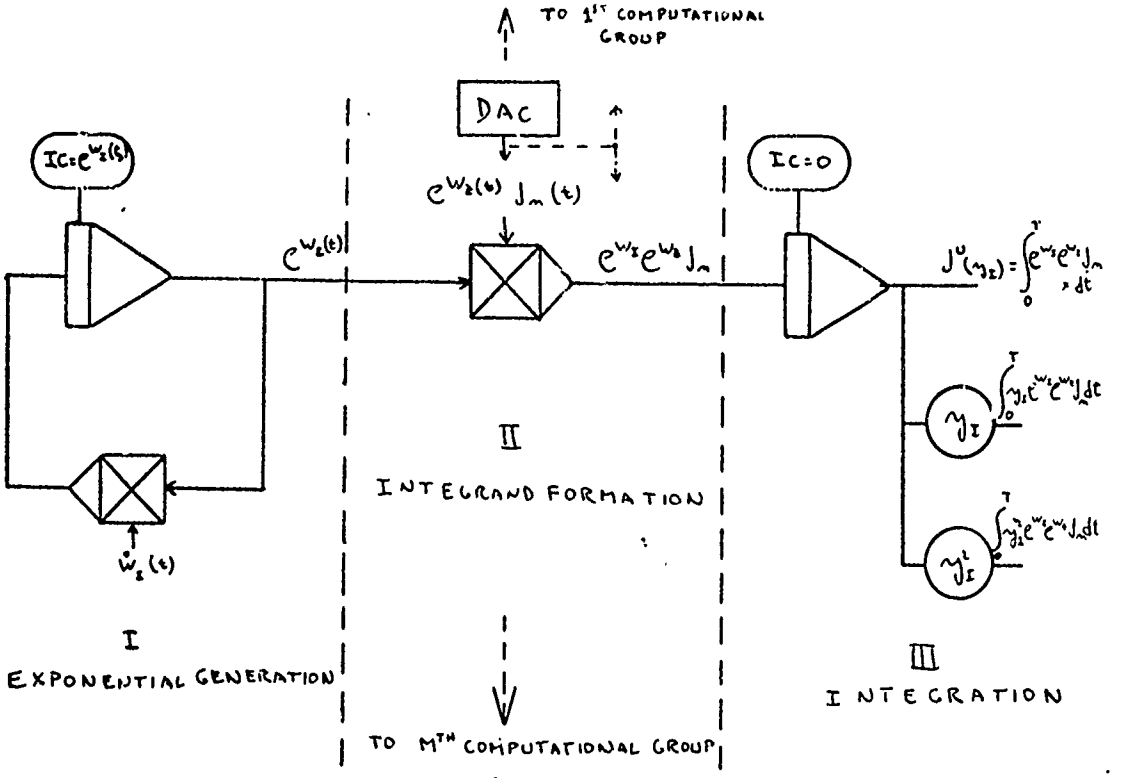


Figure 6. The Three Analog Computer Tasks in a Hybrid Optimal Filter: Exponential Generation (Section I), Integrand Formation(Section II), and Integration(Section III).

Reproduced from best available copy.

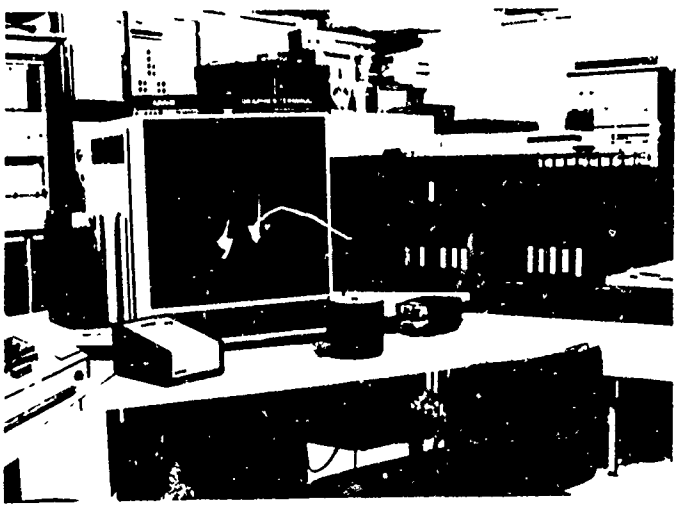


Figure 15. Adage Graphic Terminal, IBM 360-44 and a Density Function under Consideration as a Prior Distribution.

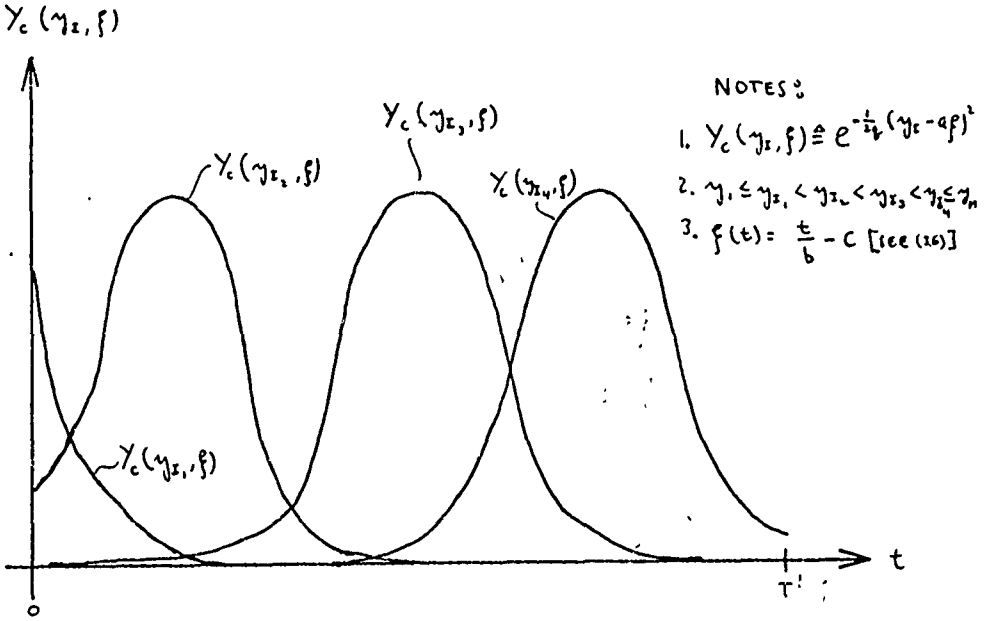


Figure 7. Exponentials Generated on an Analog Computer Corresponding to Four Typical $Y_c(\eta_z, \xi)$ Terms, each of which Represents the Contribution of Model State Update Equation to the lth Point of the $(n+1)^{st}$ Conditional Density Function.

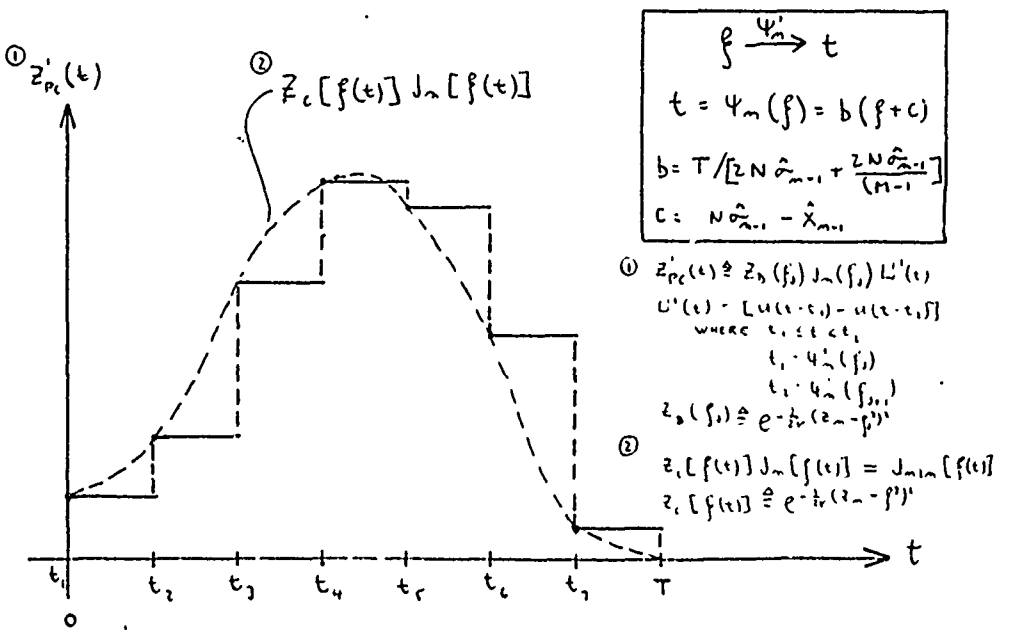


Figure 8. Piecewise Continuous Function $Z'_{Pc}(t)$ Appearing at Analog Computer as a Result of Digital to Analog Conversion in Accordance with the Map Ψ'_m of Equation (3.4) for the Case M=7. Dotted Curve Represents a Mapping by Ψ'_m of the Function $Z_c(\xi) J_m(\xi)$ which Coincides with $Z_{Pc}(t)$ at the M Computed Points.

Reproduced from best available copy.

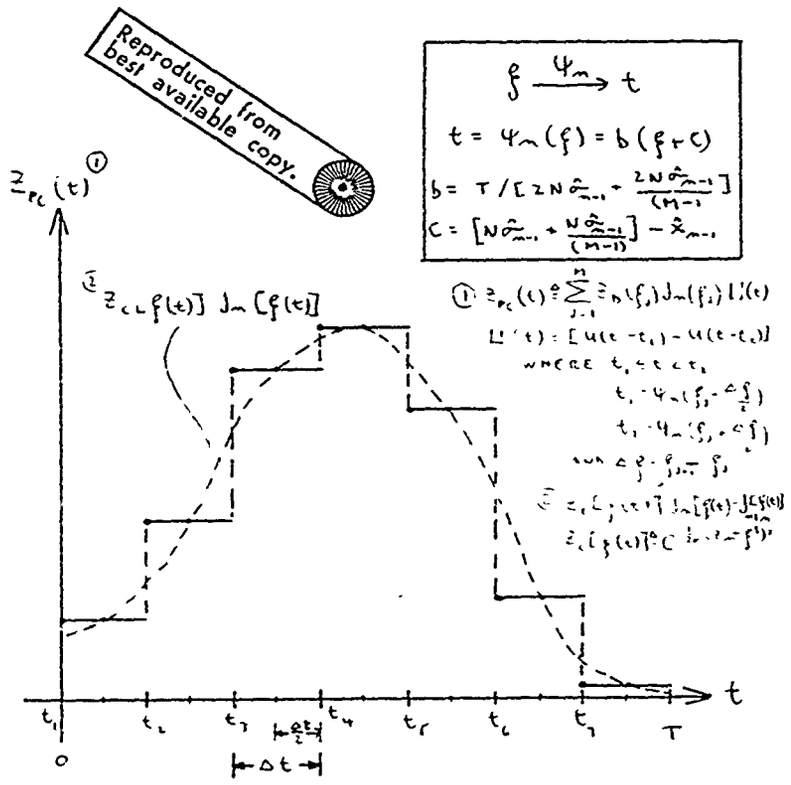


Figure 9. Piecewise Continuous Function $Z_{p_c}(t)$ Appearing at Analog Computer as a Result of Analog to Digital Conversion in Accordance with the Map Ψ_m of Equation (3.6) for the Case M=7. Dotted Curve Represents a Mapping by Ψ_m of the Function $Z_c(\xi)J_m(\xi)$ which Coincides with $Z_{p_c}(t)$ at the M Computed Points.

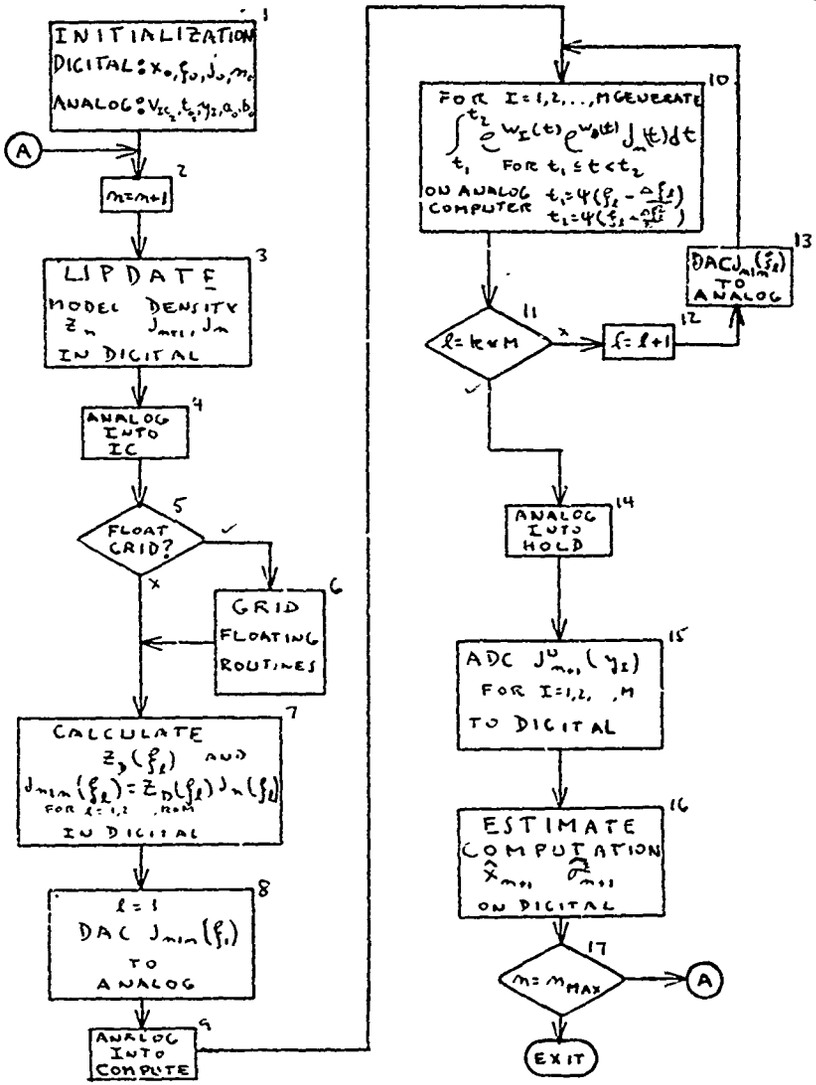


Figure 10. Flow Chart of the Hybrid Algorithm for Realization of an Optimal Discrete Nonlinear Filter as Applied to the Cubed Sensor Problem.

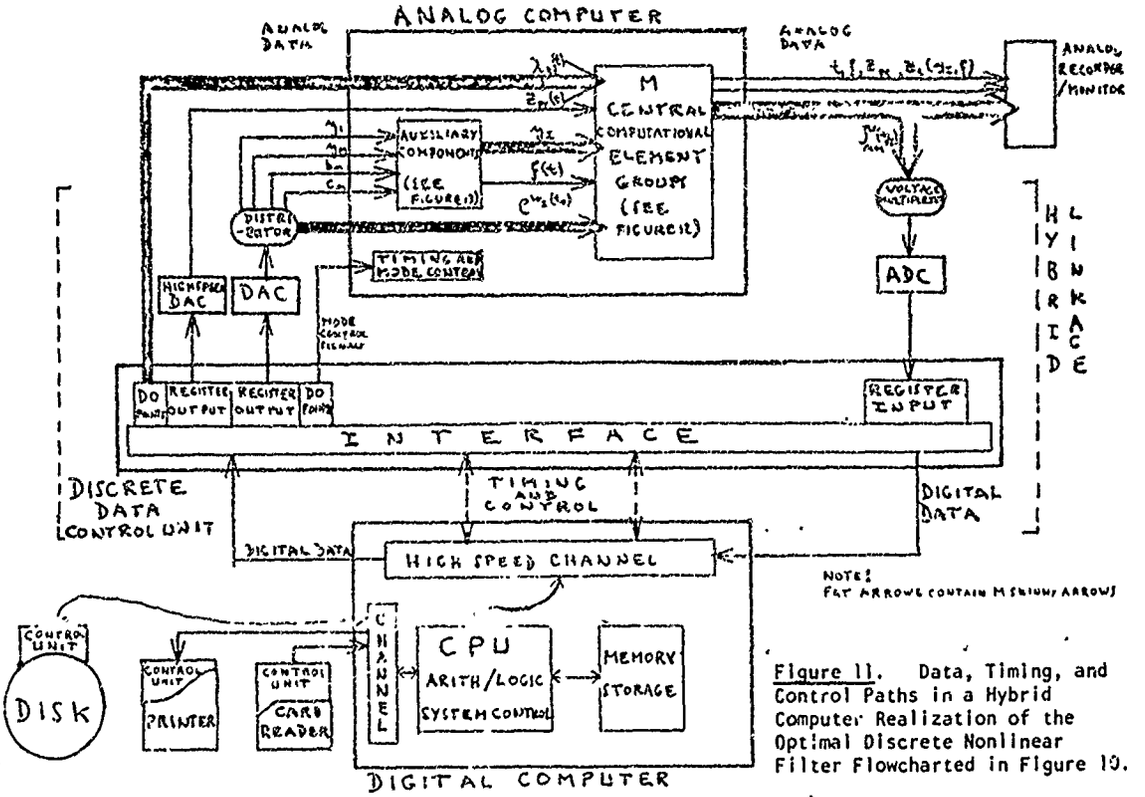


Figure 11. Data, Timing, and Control Paths in a Hybrid Computer Realization of the Optimal Discrete Nonlinear Filter Flowcharted in Figure 10.

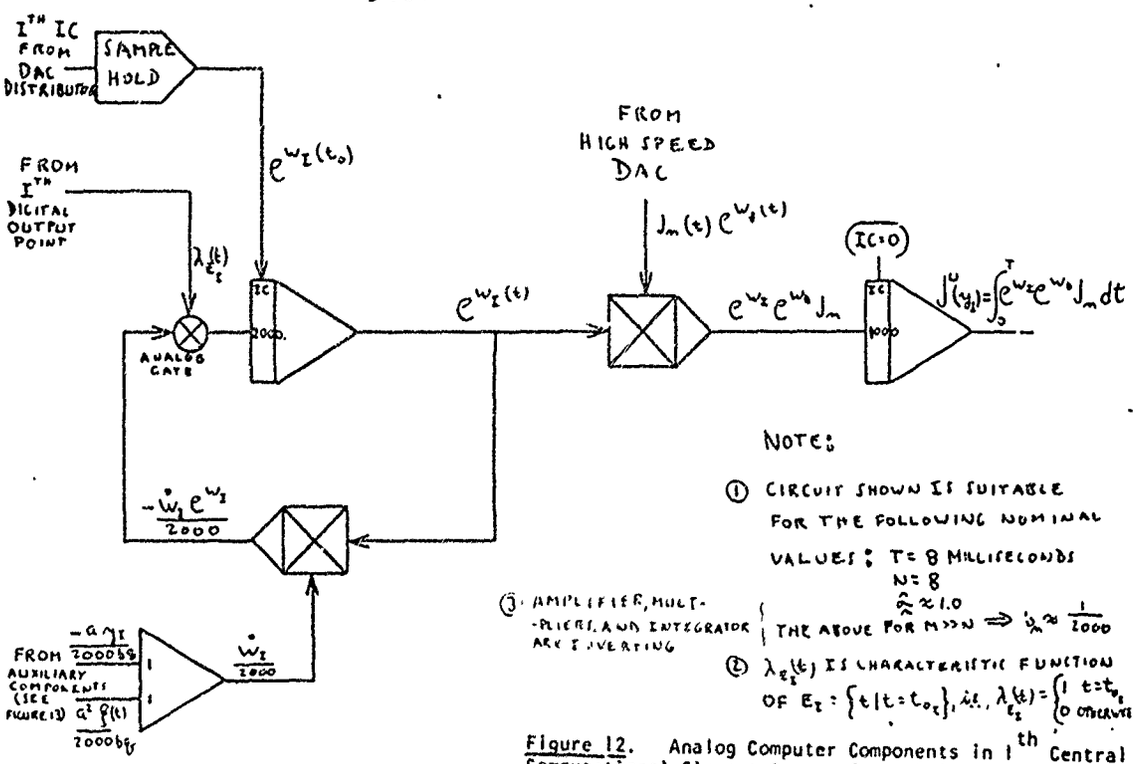


Figure 12. Analog Computer Components in I^{th} Central Computational Element Group of a Hybrid Optimal Discrete Filter for the Cubed Sensor Problem.

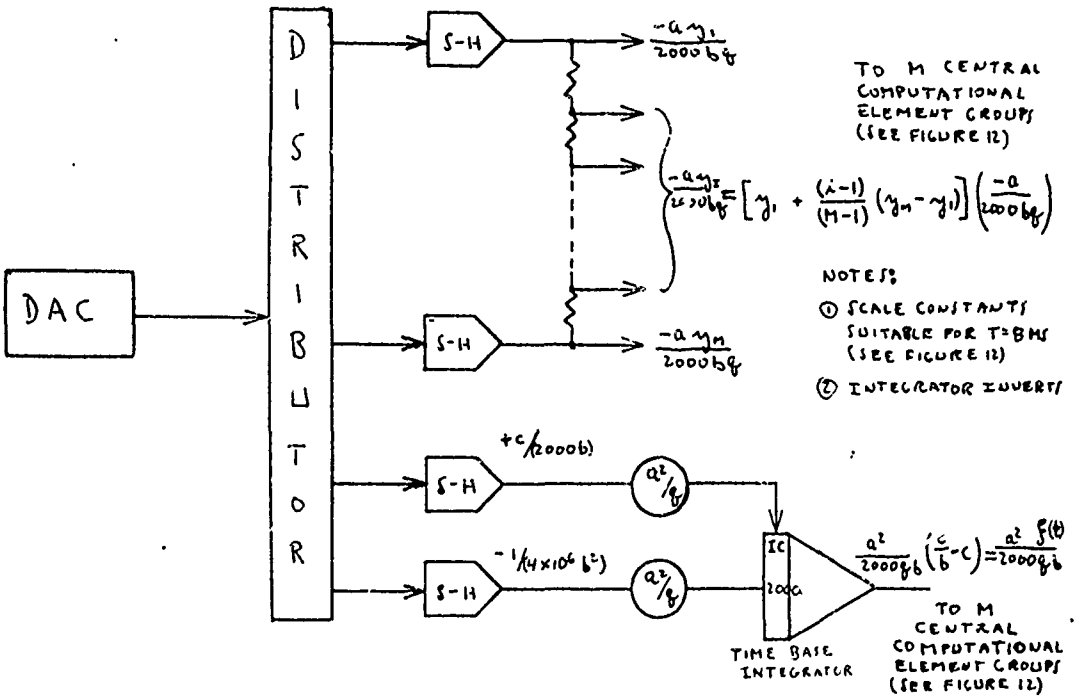


Figure 13. Analog Computer Components which Provide $\mathring{W}_i(t)$ Terms for Feedback Multiplier in i^{th} Central Computational Element Group for Cubed Sensor Problem Shown in Figure 12.

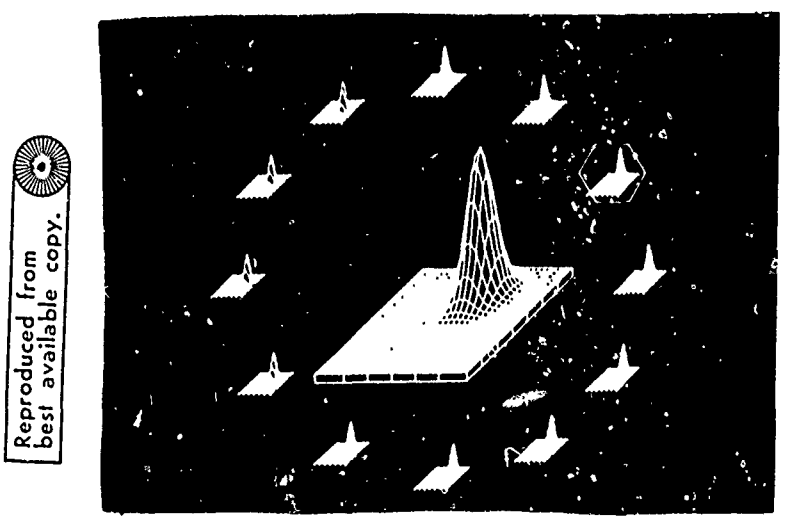

Reproduced from best available copy.

Figure 14. Multiple 2-D Density Function Display. Density function encircled by the hexagon is enlarged in the center of the screen.

REFERENCES

(1) Wald, A., Statistical Decision Functions, New York: Wiley, 1950.

(2) Ho, Y. C. and R. C. K. Lee, "A Bayesian Approach to Problems in Stochastic Estimation and Control," IEEE Transactions on Automatic Control, IX (October 1964), pp. 333-339.

(3) Bucy, R. S., "Bayes' Theorem and Digital Realization for Nonlinear Filters," The Journal of the Astronautical Sciences, XVII. No. 2 (1969), pp. 80-94.

(4) Bucy, R. S. and K. D. Senne, "Digital Synthesis of Non-linear Filters," Automatica, VII (May 1971), pp. 287-298.

(5) Senne, K. D. and R. S. Bucy, "Digital Realization of Optimal Discrete-Time Nonlinear Estimators," Proceedings of the Fourth Annual Princeton Conference on Systems Science (1970), pp. 280-284.

(6) Bucy, R. S., R. A. Geesey and K. D. Senne, "Passive Receiver Design Via Non-Linear Filtering," Proceedings of the Third Hawaii International Conference on Systems Sciences, I (1970), pp. 477-480.

(7) Senne, K. D., "Computer Experiments with Nonlinear Estimators," Proceedings of the Second Symposium on Nonlinear Estimation Theory, (1971).

(8) Hecht, C., "Digital Realization of Nonlinear Filters," Proceedings of the Second Symposium on Nonlinear Estimation Theory, (1971).

(9) Miura, T. and Iwata, J., "Effects of Digital Execution Time in a Hybrid Computer," Proceedings Fall Joint Computer Conference, Vol. 23 (1963), pp. 251-265.

(10) Johnson, C. L., Analog Computer Technics, 2nd Edition, McGraw-Hill, New York, 1963.

(11) Levine, L., Methods for Solving Engineering Problems Using Analog Computers, McGraw Hill, New York, 1964.

(12) Mallinckrodt, A. J., Bucy, R. S. and Cheng, S. Y., "A Design Study for an Optimal Non-Linear Receiver/Demodulator," Final Report to National Aeronautics and Space Administration, Goddard Space Flight Center, Contract No. NAS5-10789, (August 1970).

(26) Miller, D. S., "Hybrid Synthesis of Optimal Discrete Nonlinear Filters," Ph.D. Dissertation, University of Southern California, 1971.

(13) Bucy, R. S., "Realization of Non-Linear Filters," Proceedings of Second Symposium on Nonlinear Estimation Theory, (1971).

(14) Korn, G. A., "Progress of Analog/Hybrid Computation," Proceedings of the IEEE, Vol. 54, No. 12 (1966), pp. 1935-1849.

(15) Conant, B. K., "A New Solid State Electronic Iterative Differential Analyzer Making Maximum Use of Integrated Circuits," AFIPS Conference Proceedings, Fall Joint Computer Conference, Vol. 33 (1968), pp. 1233-1249.

(16) Kuo, B. C., Analysis and Synthesis of Sampled-Data Control System, Prentice-Hall, New Jersey, 1963, pp. 46-49.

(17) Merritt, M. J. and Miller, D. S., MOBSSL User's Manual, USCEE Report 380, University of Southern California, Electronic Sciences Laboratory, September 1969.

(18) Merritt, M. J. and Miller, D. S., "MOBSSL-UAF - An Augmented Block Structured Continuous System Simulation Language for Digital and Hybrid Computers," AFIPS Conference Proceedings, Fall Joint Computer Conference, Vol. 35 (1969), pp. 255-274.

(19) Rubin, A. I., "Analog/Hybrid-What it was, What it is, What it may be," AFIPS Conference Proceedings, Fall Joint Computer Conference, Vol. 37 (1970), pp. 641-652.

(20) Klement, R. E., Lukes, W., Merritt, M. J. and Vreeland, A., "The AGNOS Language," University of Southern California, Electrical Engineering Technical Report No. 71-19, April 1971.

(21) Barnes, C. H., Brown, R. M., Kato, M., Kuck, D. J., Slotnick, D. L., and Stokes, R. S., "The ILIAC IV Computer," IEEE Transactions on Computers, Vol. C-17, No. 8 (1968), pp. 746-757.

(22) Kuck, D. J., "ILIAC IV Software and Application Programming," IEEE Transactions on Computers, Vol. C-17, No. 8 (1968), pp. 758-770.

(23) Davis, R. L., "The ILIAC Processing Element," IEEE Transactions on Computers, Vol. C-18, No. 9 (1969), pp. 800-816.

(24) White, H. J. and Yu, E.K.C., "Use of Read Only Memory in ILIAC IV," AFIPS Conf. Pro., Spring Joint Computer Conf., Vol. 36 (1970), pp. 197-204.

(25) Graham, W.R., "The Parallel and the Pipeline Computers," Datamation (Dec 1970), pp. 68-71.

Reprinted from Proc. Second Symp. on Nonlinear Estimation Theory and Its Applications, San Diego, Sept. 1971, 152-158.

DIGITAL REALIZATION OF NON-LINEAR FILTERS*

Calvin Hecht

University of Southern California
Los Angeles, California

Abstract

A method is presented for solving the discrete-time non-linear estimation problem using Gauss-Hermite numerical integration and Hermite expansions of the conditional density function. Numerical results are given to demonstrate the effectiveness of the method.

1. INTRODUCTION

Equations have been developed which give recursive relations for the density function of the state vector conditioned on the observations (Bucy[2]). The equations are general and allow non-linear plant and observation processes, and non-Gaussian plant and observation noise terms. The equations involve integration over d-dimension space, and, except for d = 1, and in some cases 2, the equations have had no practical solutions.

Various approaches involving approximation techniques and the treatment of problems less general than the complete non-linear problem have been proposed. These approaches include schemes for approximating the non-Gaussian density function, methods of linearizing the system and treatment of problems with a non-changing state vector. Although the methods have had different levels of success, a good generalized method has not been produced.

In this paper a 2-dimensional problem is solved using a technique which appears capable of application to a six-dimension problem. The capability of the method depends on the extent of the non-linearities. The technique can be used for severe non-linearities, but the required number of points for numerical integration and the number of terms for the expansion of the density function become large, restricting the use for higher dimension problems.

2. BAYES LAW COMPUTATION

The equations for the conditional density of the state vector conditioned on the observation process using Bayesian estimation were derived in Bucy[4]. The relationships were for discrete time processes. The equations are also given in a slightly different form (avoidance of the pseudo-inverse) in Hecht[6]. The model, notation, and equations as given in the second reference are given in Tables 2.1 and 2.2.

The relations as given in Table 2.2 are generally very difficult to solve, either in closed form or numerically. Most of the recent effort in non-linear estimation has been toward solving the recurrence equations for specific applications. Several papers at the Symposium on Non-Linear Estimation Theory of 1970 [12] dealt with numerical techniques or other schemes to obtain practical solutions. It can be seen from equation (2.4), for example, that in order to construct a density function $J_{n/n}(y)$ defined on a set of points y_i, it would be required to integrate over the entire d-dimension space R^d. If the prior density function were defined on a domain which consisted of k-points for each component of x, i would be equal to k^d and the number of computations for one update of the density function would be $(k^d)^2$. For k and d larger than approximately 20 and 2, respectively, the computation would be impossible with contemporary computers.

Bucy and Senne[3] have devised a method of defining the function and performing the integration by selecting the domain in a special way. By the efficient selection of the points on which to define the density function they were able to solve a two-dimensional problem using only $(15)^2$ points and achieved an accuracy equivalent to approximately $(56)^2$ points, with a corresponding decrease in computation time. Bucy, Miller and Merritt[5] have proposed reducing computation time by performing the integrations in parallel on an analog computer as part of a hybrid computation arrangement. Lo[9] and Alspach and Sorenson[1] represent the density by a sum of Gaussian functions as another approach, although the true non-linear estimation is not achieved in the latter paper because only a linearized system is considered. Kizner [8] and others represent the density function with Hermite function expansions to solve the non-linear problem, but also fail to demonstrate the Bayes rule recurrence relation with a non-linear system and plant driving terms.

In the following development, several techniques are employed for practical solutions to the equations of Table 2.2. The next section gives the

*This research was supported by the Air Force Office of Scientific Research, United States Air Force, Under Grant AF-AFOSR-1244-67.

Preceding page blank

analytic details of a two-dimension example where the observation process, $h_n(x)$, is non-linear and the plant is linear. Both the plant and observation noise are Gaussian. By making a Hermite expansion of the non-linear prior density function, the a posteriori function is computed in closed-form. To obtain the Hermite coefficients for the new density function requires numerical integration, but the Gauss-Hermite numerical integration formulae are utilized. The Gauss-Hermite formula has been demonstrated in Hecht to be eight times more effective than rectangular integration for a one-dimension example. That is, equivalent accuracy was achieved using one-eighth the number of points to define the function to be integrated.

The technique for the two-dimension case can be extended to higher dimension examples. In the higher dimension case, computation time is reduced by performing a portion of the integrations analytically, with a potential of reducing the computation time for a d-dimension problem to the equivalent of a d/2-dimension problem.

Table 2.1 - Model for Bayes Rule Conditional Density Recursion Formula

$$x_n = \Phi_n(x_{n-1}) + \sigma_{n-1}(x_{n-1})\, u_{n-1} \qquad (2.1)$$

$$x_o = c \qquad (2.2)$$

$$z_n = h_n(x_n) + v_n \qquad (2.3)$$

Notation

- x_n = a sequence of d-dimension random vectors
- n = time index
- $\Phi_n(x_{n-1})$ = a function from R^d to R^d
- $\sigma_{n-1}(x_{n-1})$ = a function from R^d to d×r matrices
- $\{u_n\}$ = a process of independent r-dimension random vectors with density $p_{u_n}(\omega)$.
- c = a d-dimension random vector independent of the u_n process and having density $p_c(x)$.
- z_n = a sequence of s-dimension random vectors
- $h_n(x_n)$ = a function from R^d to R^s
- $\{v_n\}$ = a process of independent s-dimension random vectors with density $p_{v_n}(\phi)$, independent of c and the u_n process.

Table 2.2 - Conditional Density Recursion Formulae for Bayesian Estimation

Recurrence Relation	Equation for Conditional Density	
$J_{n-1/n-1}(y) \rightarrow J_{n/n}(y)$	$J_{n/n}(y) = \frac{1}{k(n)}\, p_{v_n}(z_n - h(y)) \int \cdots d \bullet \int p_{\sigma u}(y-\Phi(x))\, J_{n-1/n-1}(x)\, dx$	(2.4)
$J_{n/n-1}(y) \rightarrow J_{n/n}(y)$	$J_{n/n}(y) = \frac{1}{k(n)}\, p_v\ (z_n - h(y))\, J_{n/n-1}(y)$	(2.5)
$J_{n/n-1}(y) \rightarrow J_{n+1/n}(y)$	$J_{n+1/n}(y) = \frac{1}{k(n)} \int \cdots d \bullet \int p_{\sigma u}(y - \Phi(x))\, p_v(z_n - h(x)\, J_{n/n-1}(x)\, dx$	(2.6)
$J_{n/n}(y) \rightarrow J_{n-1/n}(y)$	$J_{n+1/n}(y) = \frac{1}{k(n)} \int \cdots d \bullet \int p_{\sigma u}(y - \Phi(x))\, J_{n/n}(x)\, dx$	(2.7)

Note: The index n has been omitted in some places for notational simplicity.

Notation:

- $J_{a/b}(y)$ = conditional density of the random vector x_a given observation vectors $z_o \ldots z_b$
- $k(n)$ = a normalizing constant to make $J(\bullet)$ a density function
- $\int \cdots d \bullet \int (\bullet) dx$ = d- fold integration with respect to the variables x_i, components of the vector x.
- $p_{\sigma u}(\cdot)$ = density of the d-dimension random vector $\sigma_{n-1}(x_{n-1})\, u_{n-1}$

3. TWO-DIMENSIONAL REALIZATION

A non-linear filter was synthesized to solve a phase angle receiver/demodular problem. The model for the system consisted of position and velocity. The receiver, described in Viterbi (13), and Mallinckrodt, Bucy, Cheng (10), was modeled as observing the cosine and sine of the position coordinate.

The equations to describe the signal process, or plant equations, were

$$\left.\begin{aligned} \dot{x}_1 &= x_2 + \epsilon\,\omega_1 \\ \dot{x}_2 &= \omega_2 \end{aligned}\right\} \qquad (3.1)$$

where $\epsilon\omega_1$ and ω_2 were uncorrelated Gaussian white noise processes. These equations were converted to discrete time and took the form

$$\left.\begin{aligned} x_1(n) &= x_1(n-1) + x_2(n-1)\Delta + \epsilon u_1(n-1)\Delta \\ x_2(n) &= x_2(n-1) + u_2(n-1)\Delta \end{aligned}\right\}. \qquad (3.2)$$

Hecht (6) gives a summary of modeling with discrete time systems and the conversions between discrete and continuous time models. Δ is the time between samples.

Rewriting equation (3.2),

$$x(n+1) = Fx(n) + G\,u(n) \qquad (3.3)$$

with $F = \begin{bmatrix} 1 & \Delta \\ 0 & 1 \end{bmatrix}$ $\quad G = \begin{bmatrix} \epsilon & 0 \\ 0 & 1 \end{bmatrix}$

$$x = \begin{bmatrix} x_1(n) \\ x_2(n) \end{bmatrix} \qquad u = \begin{bmatrix} u_1(n) \\ u_2(n) \end{bmatrix}$$

The observation process was

$$z(n) = H(x_1(n),\ x_2(n)) + v(n) \qquad (3.4)$$

with

$$H = \begin{bmatrix} \cos x_1(n) \\ \sin x_1(n) \end{bmatrix}$$

$$z(n) = \begin{bmatrix} z_1(n) \\ z_2(n) \end{bmatrix} \qquad v(n) = \begin{bmatrix} v_1(n) \\ v_2(n) \end{bmatrix}$$

The density functions, using the notation of Table 2.2, were

$$p_{Gu}(\xi) = \frac{1}{(2\pi)^{1/2}|Q|^{1/2}} \exp\left\{-\tfrac{1}{2}||\xi||^2_{Q^{-1}}\right\}$$

$$p_v(\xi) = \frac{1}{(2\pi)^{1/2}|R|^{1/2}} \exp\left\{-\tfrac{1}{2}||\xi||^2_{R^{-1}}\right\} \qquad (2.5)$$

where

$$Q = \begin{bmatrix} E(\epsilon^2 u_1^2) & 0 \\ 0 & E(u_2^2) \end{bmatrix} = \begin{bmatrix} \epsilon^2 q & 0 \\ 0 & q \end{bmatrix} \qquad (3.6)$$

$$R = \begin{bmatrix} E[v_1^2] & 0 \\ 0 & E[v_2^2] \end{bmatrix} = \begin{bmatrix} r & 0 \\ 0 & r \end{bmatrix}$$

In the signal process, the position is a deterministic function of velocity, which is accounted for by letting $\epsilon \to 0$.

The filter problem was solved by applying equations (2.5) and (2.7) of Table 2.2:

$$J_{n+1/n}(y) = \frac{1}{K(n)} \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} p_{Gu}(y-Fx)\, J_{n/n}(x)\, dx_1\, dy_2 \qquad (3.7)$$

$$J_{n/n}(y) = \frac{1}{K(n)}\, p_v(z_n - H(y))\, J_{n/n-1}(y) \qquad (3.8)$$

In the expression $p_{Gu}(\cdot)$ of equation (3.7) we let $\epsilon \to 0$ to obtain

$$\lim_{\epsilon\to 0} p_{Gu}(y-Fx) = c_o\,\delta\left((y_1 - x_2\Delta) - x_1\right)\exp\left\{-\frac{(y_2-x_2)^2}{2q}\right\} \qquad (3.9)$$

where $c_o = \dfrac{1}{(2\pi)^{1/2} q^{1/2}}$

$\delta(\cdot)$ = dirac delta function.

Equation (3.9) was put into (3.7).

$$J_{n+1/n}(y) = c_1 \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \delta\left((y_1 - x_2\Delta) - x_1\right)\exp\left\{-\frac{1}{2q}(y_2-x_2)^2\right\} J_{n/n}(x)\, dx_1\, dx_2$$

with c_1 a generalized constant which includes $k(n)$ and c_o; in general, we let c_i be a generalized constant which may include contributions from several sources.

The integration with respect to x_1 was performed by taking advantage of the δ- function to obtain

$$J_{n+1/n}(y) = c_1 \int_{-\infty}^{\infty} \exp\left\{-\frac{1}{2q}(y_2-x_2)^2\right\} J_{n/n}\begin{pmatrix} y_1 - x_2\Delta \\ x_2 \end{pmatrix} dx_2 \qquad (3.10)$$

$$J_{n/n}\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = c_2 \exp\left\{-\frac{1}{2r}\left[(z_1-\cos y_1)^2 + (z_2-\sin y_1)^2\right]\right\} J_{n/n-1}\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} \qquad (3.11)$$

The pair of equations, (3.10) and (3.11) give the required conditional density filter update relations. The original two-dimension problem, involving the estimation of a 2-vector, was reduced to a single integration due to the fact that only one of the states was driven by a random process.

In general, when the signal process is a dynamic system with the elements of the state vector representing position and velocity, the covariance matrix will be singular since the only driving terms are acceleration. Thus to solve a problem in three-dimension physical space requires a six-dimension state vector, but the problem can be reduced to a triple integration in the same manner as above. Bucy and Senne (3) mention the computational simplification resulting from the singular covariance matrix. It is the above simplification to which they are referring.

The integration of equation (3.10) is by no means an easy task. Normally, when numerical integration is applied, a posteriori density function $J_{n/n}\begin{pmatrix} y_1 \\ y_2 \end{pmatrix}$ is computed on a set of points $y_1(i)$, $y_2(i)$. This function becomes the prior density function for the subsequent iteration. What is needed for equation (3.10), however, is

$J_{n/n}\begin{bmatrix} y_1 - x_2 \\ x_2 \end{bmatrix}$. That is, the function is needed with the first argument different than the one for which the function was computed.

By use of Hermite polynomial expansions the above difficulty is surmounted and an explicit formula was derived for the new density function which involves no integrations.

Because of space limitations in this paper the detailed steps of the development are not given. An outline of the steps follows with equation (3.12) the final result. We assume a set of quantities are available at a particular time which completely characterizes the conditional density function, and demonstrate how these same quantities are computed for the following time increment.

1. From the prior stage we have the means, covariance matrix, characteristic values and vectors of the covariance matrix, and a series expansion of the conditional density in characteristic vector coordinates in terms of Hermite polynomials.

2. Series form of the density is put into equation (3.10) and then one takes the Fourier transform of both sides.

3. Rework the result of Step 2 to take the inverse Fourier transform to obtain $J_{n+1/n}$.

4. Multiply $J_{n/n-1}$ by the observation process, equation (3.11), to obtain $J_{n/n}$.

5. Compute moments for means, covariance matrix, and for the coefficients of the new expansion.

$$J_{n/n}\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = c_3 \cdot \exp\left\{-\frac{1}{2r}\left[(z_1-\cos y_1)^2 + (z_2-\sin y_1)^2\right]\right\} \cdot \exp\left\{-\frac{1}{2A}\left(y_2-(T_2 y_1 - T_1 \bar{v}_1)\right)^2\right\} \cdot \exp\left\{-\frac{1}{2}\frac{B^2}{C^2}\left[y_2-\beta(T_1\bar{v}_1 - T_2 y_1) + \frac{1}{B}(T_4 y_1 - T_3 \bar{v}_2)\right]^2\right\} \cdot \sum_{r=0}^{m_1}\sum_{\ell=0}^{m_2} b_{r\ell}\,(\text{sign } T_1)^r (\text{sign } T_2)^\ell \, (B)^{\frac{r+1}{2}} \left(\frac{\lambda_2 T_3^2 A}{\lambda_2 T_3^2 q + \lambda_1\lambda_2 T_1^2 T_2^2 + \lambda_1 T_1^2 q}\right)^{\frac{\ell+1}{2}} \cdot \sum_{k=0}^{r}\binom{r}{k}(-1)^k \left[\frac{q^2}{\lambda_2 T_3^2 q + \lambda_1\lambda_2 T_1^2 T_3^2 + \lambda_1 T_1^2 q}\right]^{k/2} H_{r-k}\left(\frac{1}{\sqrt{2A}}\left(y_2-(T_1\bar{v}_1 - T_2 y_1)\right)\right) \cdot H_{k+\ell}\left(\frac{B}{\sqrt{2C}}\left(y_2-\beta(T_2 y_1 - T_1\bar{v}_1) + \frac{1}{B}(T_4 y_1 - T_3\bar{v}_2)\right)\right). \quad (3.12)$$

where all of the undefined terms are functions of the quantities listed in Step 1, and c_3 is a normalizing constant to make $J_{n/n}$ a density function.

$J_{n/n}$, as given by equation (3.12) is the required equation to update the conditional density function for each sampling time. To complete the cycle it is necessary to approximate the density function with a new Hermite series, as characterized by a new set of coefficients, $b_{r\ell}$. This operation is shown in the next section.

4. HERMITE APPROXIMATION AND NUMERICAL INTEGRATION

Under the proper conditions a function of two variables, $f(x_1 x_2)$ can be approximated by a series expansion of Hermite polynomials of the form

$$f(x_1 x_2) \approx y(x_1, x_2) = e^{-\alpha_1^2 x_1^2} e^{-\alpha_2^2 x_2^2} \sum_{r=0}^{m_1}\sum_{\ell=0}^{m_2} b_{r\ell}\, H_r(\alpha_1 x_1) \cdot H_\ell(\alpha_2 x_2) \quad (4.1)$$

with the coefficients, $b_{r\ell}$, determined by

$$b_{r\ell} = \frac{\alpha_1\alpha_2}{\pi\, r!\, 2^\ell \ell!\pi} \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x_1 x_2) H_r(\alpha_1 x_1) H_\ell(\alpha_1 x_2)\, dx_1\, dx_2 \quad (4.2)$$

The approximation is the best in the sense that

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} e^{\alpha_1^2 x_1^2} e^{\alpha_2^2 x_2^2}\left[f(x_1 x_2) - y(x_1 x_2)\right]^2 dx_1 dx_2 = \text{minimum} \quad (4.3)$$

By letting $f(x_1 x_2)$ be the density function of the previous section with

$$\alpha_1^2 = \frac{1}{2\sigma_1^2}\,, \quad \alpha_2^2 = \frac{1}{2\sigma_2^2}\,, \quad \sigma_1^2, \sigma_2^2$$

the characteristic values of the covariance matrix, the conditions for the approximation to be valid are easily satisfied (Hildebrand (7)).

Let $f(x_1 x_2)$ be a density function we wish to approximate, and let $m_1 = m_2 = 0$ in equation (4.1). Then

$$b_{00} = \frac{\alpha_1\alpha_2}{\pi}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x_1 x_2)\, dx_1\, dx_2 = \frac{\alpha_1\alpha_2}{\pi} \quad (4.4)$$

$$y(x_1 x_2) = \frac{\alpha_1\alpha_2}{\pi}\, e^{-\alpha_1^2 x_1^2} e^{-\alpha_2^2 x_2^2} \quad (4.5)$$

Let σ_1^2 and σ_2^2 be the variances of $f(x_1 x_2)$, and let

$$\alpha_1^2 = \frac{1}{2\sigma_1^2}$$

$$\alpha_2^2 = \frac{1}{2\sigma_2^2}$$

Equation (4.5) is

$$y(x_1 x_2) = \frac{1}{2\pi\sqrt{\sigma_1\sigma_2}} e^{-\frac{x_1^2}{2\sigma_1^2}} e^{-\frac{x_2^2}{2\sigma_2^2}} \quad (4.6)$$

By taking only the zero'th term of the expansion we can get a Gaussian fit to the true density function.

In addition to the above assumptions, except let m_1 and $m_2 > 0$, let η_1 and η_2 be the means of $f(x_1 x_2)$ and μ_{ij} the i - j'th central moment.

$$\mu_{ij} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}(x_1 - \eta_1)^i (x_2 - \eta_2)^j f(x_1 x_2)\, dx_1 dx_2 \quad (4.7)$$

The product of the Hermite polynomials when expanding about the expected values, can be written

$$H_r(\alpha_1(x_1-\eta_1)) \cdot H_\ell(\alpha_2(x_2-\eta_2)) = \sum_{i=0}^{r}\sum_{j=0}^{\ell} a_j(r)\, a_j(\ell)\alpha_1^{\,i}\,\alpha_2^{\,j}(x_1-\eta_1)^i (x_2-\eta_2)^j \quad (4.8)$$

Equation (4.2) is then

$$b_{r\ell} = \frac{\alpha_1\,\alpha_2}{2^r r!\, 2^\ell \ell!\pi}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x_1 x_2) \sum_{i=0}^{r}\sum_{j=0}^{\ell} a_{ij}\,\alpha_1^{\,i}\,\alpha_2^{\,j}(x_1-\eta_1)^i (x_2-\eta_2)^j\, dx_1 dx_2$$

$$= \frac{\alpha_1\,\alpha_2}{2^r r!\, 2^\ell \ell!\pi}\sum_{i=0}^{r}\sum_{j=0}^{\ell} a_i(r)a_j(\ell)\alpha_1^{\,i}\,\alpha_2^{\,j}\,\mu_{ij} \quad (4.9)$$

The coefficients $a_i(r)\, a_j(\ell)$ in (4.9) are functions of the Hermite polynomial coefficients, and need to be computed only one time in advance and stored.

The recurrence formula for the Hermite polynomials is (Hildebrand [7]):

$$\left.\begin{aligned} H_{r+1}(x) &= 2x\, H_r(x) - 2r\, H_{r-1}(x) \\ H_0(x) &= 1 \qquad H_1(x) = 2x \end{aligned}\right\} \quad (4.10)$$

Designating c_{mn} as the coefficient of x^n in $H_m(x)$, $a_i(r)\, a_j(\ell)$ in equation (4.9) is determined from

$$a_i(r)\, a_j(\ell) = c_{ri}\, c_{\ell j} \quad (4.11)$$

c_{mn} was generated from a recurrence formula which was derived from (4.10).

Comparing equation (4.1) with the conditional density function in eigenvector coordinates from the previous section, we see the approximating polynomial expansion can be accomplished by letting:

$$\alpha_1 = \frac{1}{2\lambda_1} \qquad \alpha_2 = \frac{1}{2\lambda_2}$$

$$\eta_1 = \bar{v}_1 \qquad \eta_2 = \bar{v}_2$$

and using (4.9) to compute the coefficients $b_{r\ell}$.

Gauss-Hermite numerical integration was used to perform the integrations for the moments of equation (4.7). The conditional density function was given by equation (3.12).

Gauss-Hermite integration is performed according to the following formula:

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} e^{-x_1^2 - x_2^2} f(x_1\, x_2)\, dx_1\, dx_2 \cong \sum_{i=1}^{n}\sum_{j=1}^{n} \omega_i\, \omega_j\, f(x_{1_i}, x_{2_j}) \quad (4.12)$$

with x_{1_i} and x_{2_j} the roots of $H_n(x)$ and ω_i and ω_j the corresponding weighting functions. The numbers ω_i and x_i are tabulated in several places for a variety of values of n; Shao, Chen, Frank [11] for example.

The a posteriori density function of equation (3.12) consists of a product of an exponential function and a polynomial function. The exponential function has two contributions, the non-linear observation part with the trigonometrical functions and a Gaussian part.

To perform the numerical integration of equation (3.12) effectively, it was found desirable to consider two situations and to treat them differently. 1) When the contribution of the observation process was small (r large), the non-Gaussian exponential was combined with the polynomial function to construct $f(x_1, x_2)$ of equation (4.12). The characteristic values and vectors of the remaining Gaussian part were found and used to formulate the exponential part of the integrand of (4.12). 2) When the contribution of the observation process was significant (r small), the non-Gaussian exponent was linearized. The linearized exponent was combined with the other Gaussian exponents and treated in the same way as in 1). The difference between the non-linear and linear exponent was combined with the polynomial function to construct $f(x_1, x_2)$ of equation (4.12).

5. NUMERICAL RESULTS

The equations of the previous sections were programmed and a limited number of runs were made. An extended Kalman-Bucy filter (extended K-B filter) was programmed for the same problem using the same noise sequence for comparison purposes.

Runs were made using the non-linear filter with the coefficients, $b_{r\ell}$ of equation (4.12), up to b_{88}. It was found that equal accuracy could be obtained by eliminating higher order cross-

coefficents, so by adjusting m_1 and m_2 in equation (3.12) so that $r+\ell \leq 8$, a reduction in running time without loss of accuracy was obtained.

The numerical integration was performed using 10-point and 20-point integrations. For good signal-to-noise ratios, the lower number of points was adequate, whereas with poor ratios more accurate integration was needed. This was to be expected since for good signal/noise the equations are close to being Gaussian and the Gauss-Hermite integration formulae are almost perfect. The basic independent parameter for the numerical trials was the continuous linear model error variance of x_1, designated $p_{11}(o)$. The discrete linearized variance of the one-step prediction is designated $p_{11}(\Delta)$, and in the limit $p_{11}(\Delta) \rightarrow p_{11}(o)$. In Hecht[6] it is shown that for Δ small compared to the filter time constant, F,

$$p_{11}(\Delta) \cong p_{11}(o)\ (1 + \frac{\Delta}{F}) .$$

All trials were made with $\Delta/F = 0.1$.

Viterbi[13] shows $p_{11}(o)$ is the inverse of the effective signal/noise ratio, so $p_{11}(o)$ has the double meaning for the continuous linearized model:

1) $p_{11}(o) = E\left[(x_1 - \hat{x}_1)^2\right]$

2) $p_{11}(o) = \frac{1}{S/N} = \frac{N}{S}$

Trials were made with sample path of 100 points (10 filter time constants). The variance of the initial state was set equal to r in every case. The signal term, q, was set to 0.01 for all trials.

For very small $p_{11}(o)$ the extended K-B filter and the non-linear filter performed almost identically. For $p_{11}(o) = .0001$ the sequence of estimates for the same signal and noise sequences agreed to within 6 decimal places.

As the parameter $p_{11}(o)$ was increased, the non-linear filter was also tested using only one coefficient, b_{oo}. The reason for this is that one would expect better performance from the one-coefficient filter than the extended K-B filter due to the integration of the non-Gaussian term. It was desired to determine the performance of the nine-coefficient filter since all of the non-linear information is carried by the coefficients not including b_{oo}. By checking both the one and nine-coefficient filter, the additional improvement of the higher terms was evaluated.

For $p_{11}(o) \leq .04$ (S/N = 13 db) the average variance for a 100 sample run was almost the same for the extended K-B, the one-, and the nine-coefficient filters. For $p_{11}(o) = .16$ (S/N = 8 db), the one-coefficient filter shows an improvement over the extended K-B, and the nine-coefficient filter shows an equal improvement over the one-coefficient filter.

For $p_{11}(o) = .25$ (S/N = 6 db), the non-linear filter appears substantially better than the extended K-B. These results, although for a very small sampling, do indicate an encouraging trend. They are summarized in Table 5.1.

Table 5.1 - Average Variance for Extended Kalman-Bucy, One-Coefficient, and Nine-Coefficient Non-Linear Filters.

Sample Path = 100 points

$p_{11}(o)$	Q_D	R_D	Δ	K-B	1-Coef.	9-Coef.
.01	.00126	0.05	0.126	6.6852 E-3	7.0754 E-3	7.0782 E-3
.04	.00200	0.2	0.200	6.1703 E-2	6.1681 E-2	6.1711 E-2
.16	.00317	0.8	0.317	2.0403 E-1	1.9556 E-1	1.8349 E-1
.25	.00368	1.25	0.368	7.2118 E-1	2.9438 E-1	2.7058 E-1

REFERENCES

(1) Alspach, D. L. and Sorenson, H. W., "Approximation of Density Functions by a Sum of Gaussians for Non-Linear Estimation," Symposium on Nonlinear Estimation Theory and its Applications, San Diego, Calif., Sept. 21-23, 1970.

(2) Bucy, R. S., and Joseph, P. D., "Filtering for Stochastic Proccesses with Applications to Guidance," Wiley-Intersciences, New York, 1968.

(3) Bucy, R. S., and Senne, K. D., "Digital Synthesis of Non-linear Filters," Autonatica, Vol. 7, pp. 278-298, Pergamon Press, 1971.

(4) Bucy, R. S., "Bayes Theorem and Digital Realizations for Non-Linear Filters," Journ. of the Astronautical Sciences, Vol. XVII, No. 2, Sept.-Oct. 1969.

(5) Bucy, R. S, Merritt, M. J., and Miller, D. S. "Hybrid Computer Synthesis of Optimal Discrete Non-linear Filters," Second Symp. on Nonlinear Estimation Theory, San Diego, Calif, Sept. 1971.

(6) Hecht, C., "Synthesis and Realization of Nonlinear Filters," Ph. D. Dissertation, Univ. of Southern California, Los Angeles, 1971.

(7) Hildebrand, F. B., "Introduction to Numerical Analysis," McGraw-Hill, New York, 1956.

(8) Kizner, W., "Optimal Estimation Based on Orthogonal Expansions," Techn. Rep. 32-1366, Jet Propulsion Laboratory, Pasadena, Cal. April, 1969.

(9) Lo, J. T., "Finite Dimension Sensor Orbits and Optimal Nonlinear Filtering," Ph. D. Dissertation, Univ. of So. Cal., Los Angeles, Aug., 1969.

(10) Mallinckrodt, Bucy, Cheng, "Final Report for a Design Study for an Optimal Nonlinear Receiver/Demondulator," Goddard Space Flight Ctr. Greenbelt, Maryland, August, 1970.

(11) Shao, Chen, Frank, "Tables of Zeros and Weights of Certain Associated Laguerre Polynomials and the Related Generalized Hermite

Polynomials," Mathematics of Computation, page 604, Oct., 1964.
(12) Symposium on Nonlinear Estimation Theory and its Applications, San Diego, Calif., Sept. 21-23, 1970.
(13) Viterbi, A.J., "Principles of Coherent Communications," McGraw-Hill, New York, 1966.
(14) Wiener, N., "The Fourier Integral and Certain of its Applications," Dover Public, Inc. New York, 1933.

Corrected Printing:

Original misprinted in Proc. Second Symp. on Nonlinear Estimation Theory and Its Applications, San Diego, Sept. 1971, 314-324.

COMPUTER EXPERIMENTS WITH NONLINEAR ESTIMATORS*

Kenneth D. Senne
Frank J. Seiler Research Laboratory
Air Force Systems Command
U.S. Air Force Academy, Colorado 80840

Abstract

Elaborate and extensive experiments with various digital implementations of discrete-time nonlinear estimators reveals some interesting and important characteristics of nonlinear estimators. The experiments include extensive Monte Carlo comparisons of alternative techniques, sample path comparisons of optimal and approximate estimators and associated graphical illustration of relevant considerations and details.

1. INTRODUCTION

In the proceedings of the first Symposium on Nonlinear Estimation Theory, Bucy and Senne proposed a digital realization for nonlinear estimators (henceforth referred to as the Floating Grid Method)[1,2]. The basis for the Floating Grid Method is a numerical approximation to the general Bayes-optimal solution to the nonlinear estimation problem[3]. Thus the Floating Grid is one of a variety of possible applications of numerical techniques to calculate the Bayes integral recursion formula[4,5]. The alternatives to the numerical techniques, which involve only density representation and integral approximations, are the various analytical techniques, which involve functional approximations of the underlying physical problem equations and/or moment series expansions of the conditional densities[3,6,7]. Common analytical approximations include linearized (extended) Kalman-Bucy filters, quasi-moment filters, second order techniques, etc. The obvious advantages of analytical approximations to the estimation problem are the relative simplicity of the computations, compared with the numerical approximations. On the other hand, improvements to analytical techniques are relatively hard to obtain since generalizations come at the expense of extensive analysis and/or engineering experimentation, whereas the accuracy of a numerical approximation can be directly related to computation time, with little or no modification to the computational algorithm itself. Thus, while practical applications will frequently demand the simplicity of analytical approximations, the answers to questions relating to the behavior

* The views expressed herein are those of the author and do not necessarily reflect the views of the United States Air Force or the Department of Defense.

Preceding page blank

of optimal estimates which can give considerable insight into the selection of practical estimators may best be obtained by studying a suitable numerical approximation.

In this presentation a report is given describing extensive additional experimentation with the Floating Grid Method and comparisons with analytical approximates including the linearized filter and the generalized linearized filter of Alspach and Sorenson[5]. All techniques have been exposed to the same data records both for sample path studies and elaborate Monte Carlo performance comparisons. As a result of the simulations, many conclusions have been made concerning the behavior of optimal estimates, the "engineering modifications" to analytic approximations which improve performance, and experimental methods relating to evaluation of estimators. Sample path studies are exemplified by a movie illustrating the evolution of the conditional probability densities for a passive tracking problem. Monte Carlo studies are given to compare the performance of the Floating Grid Method, the linearized estimator, the "modified" linearized estimator, and the generalized linearized (gaussian-sum) estimator on a similar tracking problem. The results illustrate conclusively that additional efforts on sophisticated numerical techniques will result in considerable additional insight and experience with optimal nonlinear estimators.

2. PROBLEM DESCRIPTION

Consider the non-linear dynamical system described by

$$\underline{x}_n = \underline{f}_n(\underline{x}_{n-1}) + \underline{\sigma}_n(\underline{x}_{n-1})\underline{u}_{n-1}, \qquad \underline{x}_o = \underline{c}$$
$$\underline{z}_n = \underline{h}_n(\underline{x}_n) + \underline{v}_n \qquad (1)$$

with

$$\underline{f}_n(\cdot) : R^d \to R^d$$
$$\underline{\sigma}_n(\cdot) : R^d \to R^d \times R^p$$
$$\underline{h}_n(\cdot) : R^d \to R^q$$

and $\underline{u}_n$, $\underline{v}_n$ are zero-mean white Gaussian sequences with covariances Q(n), R(n), respectively. The initial state vector $\underline{c}$ is distributed on R^d according to the density $\underline{J}_o(\cdot)$. If the conditional density of $\underline{x}_n$, given measurements $\underline{z}_o, \underline{z}_1, \ldots, \underline{z}_{n-1}$, is denoted by

$$\underline{J}_n(y)dy \triangleq \text{prob}(\underline{x}_n \in y+dy \mid \underline{z}_o, \underline{z}_1, \ldots, \underline{z}_{n-1}), \qquad (2)$$

then an application of the discrete representation theorem gives the sequential formula formula for obtaining $\underline{J}_{n+1}(y)$ from $\underline{J}_n(y)$ as follows:

$$C_n \underline{J}_{n+1}(\underline{y}) = \int_{R^d} T_n(\underline{y},\underline{\zeta})\underline{J}_n(\underline{\zeta})d\underline{\zeta}, \qquad (3)$$

where C_n is a normalizing factor, and T_n depends on $z_n - \underline{h}_n(\underline{\zeta})$ and $\underline{y} - \underline{f}_n(\underline{\zeta})$ in accordance with the assumed Gaussian density functions[2]. The approximation problem is involved with representing the conditional densities (2) numerically and calculating the d- dimensional integral (3). In many cases of interest the conditional mean is the desired estimate of $\underline{x}_n$.

3. PROPOSED IMPLEMENTATIONS

Some early attempts to implement approximate optimal nonlinear estimators resulted in the application of power series expansions of the system nonlinearities and/or moment expansions of the conditional density function[1,5,7]. Such techniques may be referred to by the term "analytical approximations", since the underlying problem is represented by a simpler analytic description. Recently some attention has been turned toward the problem of representing the Bayes integral solution to discrete-time estimation problems by a numerical approximation on a digital computer. Numerical techniques consist basically of two interrelated subproblems - density representation and Bayes-integral computation. For density representation various investigators[1,8] have proposed orthogonal series representations, floating point grid representations[1,2], and sums of gaussians[3]. The Bayes integral calculation is usually tied very closely to the type of density representation. For example,

orthogonal series representations usually lead to some form of fourier transform calculations for the Bayes integral, either explicitly[4], or implicitly[8]. Alternatively, the floating point grid representations typically lead to simple quadrature formulas for the Bayes integral. To date, however, the gaussian-sum representation has required the use of analytical approximations such as local linearization, in order to effectively implement the Bayes law computation, which really classifies the technique as a generalized linearized (analytical) technique.

4. AN EXAMPLE PROBLEM

In order to compare the performance of the optimal estimator with that of currently popular approximate estimators we have chosen a simple two-dimensional example which is based on a passive ranging experiment[1,2].

$$\underline{x}(n)=\underline{F}\underline{x}(n-1)+\underline{u}(n-1), \qquad \underline{x}(0)=\underline{c}$$

$$\underline{z}(n)=\underline{h}_n[\underline{x}(n)]+\underline{v}(n), \tag{4}$$

where

$$\underline{h}_n[\underline{x}(n)]=\tan^{-1}\left[\frac{x_2(n)-\sin\beta n}{x_1(n)-\cos\beta n}\right]$$

where the transition matrix F may be taken as the identity in order to simulate random walk, $\underline{u}$ is distributed zero-mean Gaussian with covariance Q, $\underline{v}$ is zero-mean Gaussian with covariance R and β is the angular rotation rate of the sensor. The mathematical model (4) arises in connection with tracking geometries of Figure 1 and Figure 2. In the case of Figure 1, an aircraft sensor S is orbiting about a radarship target T which is undergoing random walk motions from the ocean waves. In this case the initial distribution on $\underline{x}$, the position of the ship, is taken to be zero-mean Gaussian with identity covariance Σ_0. The geometry of Figure 1 more closely describes the situation studied previously than that of Figure 2, which illustrates the more important (and obviously more difficult) problem of locating a target T which is separated by a significant distance from the sensor's circular orbit. The problem described by (4) is in either case one of locating a target by using only bearing sensor information (passive reception). It turns out that the previously studied problem of Figure 1 is considerably easier than originally suspected, and a new experiment comparing all previously proposed methods on the same sample trajectories reveals this fact, as described in the next section.

5. EXPERIMENTAL RESULTS

In order to adequately study the passive receiver problem a number of increasingly difficult cases must be considered. The first case, referred to as the "old problem," is the situation illustrated in Figure 1, with $F=\begin{bmatrix}0.5 & 0.0\\ 0.0 & 1.0\end{bmatrix}$, $\beta=1$, $Q=\begin{bmatrix}0.10 & 0.05\\ 0.05 & 0.10\end{bmatrix}$, $R=0.1$, and $\underline{J}_0 \sim N(\underline{0},I)$. The second problem, called the "new problem," is identical to the old problem, with the important exceptions that $\underline{J}_0 \sim N\left(\begin{bmatrix}3\\3\end{bmatrix},I\right)$, $R=0.01$, and $F=I$. Finally, an example with a multimodal (i.e. nongaussian) initial density, described in Figures 3 and 4, is presented to illustrate what happens if the Gaussian approximation is radically wrong.

Reproduced from best available copy.

5.1 THE OLD PROBLEM

It turns out that the sensor is allowed to have considerable new information with each sample if it orbits about the target at the high rate of $\beta=1$ radian per sample. Thus it is not surprising to find that it is relatively easy to design an approximate estimator which performs very close to optimum. A straightforward linearized predictor performs miserably, it happens, because of the large fluctuations in the innovations process $\underline{z}_n-\underline{h}[\hat{\underline{x}}(n)]$ as the sensor completes each orbit. In fact, as previously observed[1,2], the linearized filter is asymptotically unstable - the mean square error oscillates with ever increasing amplitude. This fact is illustrated in Table 1, where in the second column the Monte Carlo average of the sum squared errors for 100 sample paths is presented for an iterated linearized predictor which is linearized as usual about the current filtered estimate and the filter update is iterated until the sensor linearizations have stabilized. If the

calculations are continued for more samples in each trajectory then the instability becomes easily evident. The behavior of the linearized predictor is so poor, in fact, that it is worse than the "blind predictor" which is initialized at the mean of the state vector and ignores the data entirely thereafter. The theoretical mean-square error performance of the mean-state predictor is illustrated in column one of Table 1.
If the geometry of the old problem is exploited, it is possible to "fix up" the linearized predictor so that its performance is almost optimal. The fixed linearized predictor is identical to the standard linearized predictor with the addition of logic to calculate the innovations $\underline{Z}_n - \underline{h}_n[\hat{\underline{x}}(n)]$ modulo 2π in the range $[-\pi,\pi]$. The sensational improvement in performance of the fixed predictor is illustrated in column three of Table 1.
Finally the numerical techniques of gaussian sums[5], and floating grids[1], are included in the last two columns of Table 1. If allowances for the fact that the one standard deviation confidence level in a Monte Carlo estimate of N tracks is approximately $2\sigma^4/N$, where σ^2 is the true variance and the densities are assumed Gaussian, then for 100 Monte Carlos the resulting variances are only determined within one standard deviation to ±17.32 percent of the true σ^2. Thus, to within the experimental accuracy available in 100 Monte Carlos, the performances illustrated in the last three columns of Table 1 are essentially equivalent. We conclude, therefore, that the problem is not sufficiently difficult to resolve the differences between the various estimation techniques.

5.2 THE NEW PROBLEM

If the geometry of Figure 1 is replaced with that of Figure 2 then it no longer is possible to fix up the performance of the linearized predictor by a simple application of modular arithmetic. Furthermore, the performance of the numerical techniques is decidedly better than the linearized predictor as illustrated in Table 2. The Gaussian sum predictor experienced an unexplained transient error in the first estimate on about the fiftieth Monte Carlo which was so large that the resulting Monte Carlo estimate is clearly in the transient phase in the table for the first estimate. The recovery is stable, however, so that the remaining values in column three are reliable.

5.3 MULTIMODAL INITIAL CONDITIONS

In the previous two cases the relative success of the linearized-type predictor can be related to the validity of the Gaussian approximation to the conditional density functions. The simplest way to destroy the validity of the Gaussian assumption is to provide a nongaussian initial density function. Consider, for example the detector geometry illustrated in Figure 3. If there were a sequence of known reflecting ionospheric layers above the aircraft observer and we were given an apriori distribution on the transmitter's power, then it is conceivable that we might want to integrate over all elevations to maximize detectability, thus introducing a multimodal range ambiguity as shown in the figure. Probabilistically, the initial condition would be obtained by taking the product of the multimodal range ambiguity density with the bearing ambiguity density. The result might look as in Figure 4, where no particular scale is intended. As soon as the aircraft's detector circuits obtain a reliable detection, the aircraft banks left into a circle of unit radius and activates a high sensitivity receiver which is tuned to one elevation. The resulting problem parameters used for this experiment are $F=\begin{bmatrix}1.001 & 0.000\\ 0.000 & 1.001\end{bmatrix}$, $Q=\begin{bmatrix}0.050 & 0.025\\ 0.025 & 0.050\end{bmatrix}$, $R=.01$, $S=.01$, and $\underline{J}_o$ was chosen with one cross range standard deviation corresponding to 0.25 radian in bearing and a down range standard deviation in each mode of 0.25. If the densities are plotted isometrically as viewed from the direction shown in Figure 4, then some of the more interesting examples from the movie are shown in Figure 5, with sample numbers as given. As seen in Figure 5 the density first peaks up on the wrong range when very little information is available from the measurements, and later makes a dramatic change to the correct mode and stabilizes. This behavior is clearly illustrated in the solid

plot of Figure 6 of the resulting prediction error magnitude. In contrast the dotted curve in Figure 6 shows the performance for the same sample path of the linearized predictor, which is almost identical to the mean state estimate for the problem. Figure 6 illustrates two interesting phenomena. The linearized approximation can be made useless by only a large uncertainty in initial conditions. On the other hand, the nonlinear estimator, which contains the more detailed information about the conditional densities may still perform satisfactorily. In addition, the optimum system creates a certain amount of optimism for itself initially and calculates a relatively high confidence incorrect estimate. Experience with several such sample paths has resulted in the conclusion that the situation in Figure 6 is just about the worst possible case, that on the average the performance is without much improvement for about half the iterations (0.5 radian of sensor rotations), and then proceeds to converge at about the same rate as the previous examples. The linearized predictor, on the other hand, without a suitable initial condition doesn't ever converge on the average.

6. CONCLUSIONS

The examples have shown the wealth of experimental experience available to the estimation engineer as a result of numerical studies of the problem. It is possible, for example, to assess the validity of given analytical approximations and the value of unused information regarding the structure of the estimation problem. In addition it is possible to compute practical lower bounds on performance for nonlinear estimators in some small dimensional problems, a fact which may have some important engineering significance to designers of sensors, for example, who must know the theoretical limitations on the performance of a given design. It is anticipated that future experiments along these lines will substantiate this conjecture. In the meantime there appears to be many possible application areas to explore.

7. BIBLIOGRAPHY

1. R.S. Bucy and K.D. Senne, "Realization of Optimum Discrete-Time Nonlinear Estimators" Proc. Symposium on Nonlinear Estimation Theory 6-17 (1970).

2. R.S. Bucy and K.D. Senne, "Digital Synthesis of Non-linear Filters," Automatica 7, 287-298 (1971).

3. R.S. Bucy and P.D. Joseph, Filtering for Stochastic Processes with Applications to Guidance, Wiley Interscience, New York, 1968.

4. H.W. Sorenson and A.R. Stubberud, "Non-linear Filtering by Approximation of the A Posteriori Density," International J. Control 8, 33-51 (1968).

5. H.W. Sorenson and D.L. Alspach, "Recursive Bayesian Estimation using Gaussian Sums," Automatica (1971).

6. R.W. Bass, V.D. Norum, and L. Schwartz, "Optimal Multichannel Nonlinear Filtering," J. Math. Anal. Appl. 16, 152-164 (1966).

7. H.J. Kushner, "Approximations to Optimal Nonlinear Filters," IEEE Trans. Automatic Control 12, 546 -556 (1967).

8. C. Hecht, "Digital Realization of Nonlinear Filters," Proc. Second Symposium on Nonlinear Estimation Theory (1971).

Sample	Mean-State Predictor	Iterated Linearized Predictor	"Fixed" Linearized Predictor	Gaussian Sum Predictor	Floating Grid Predictor
1	1.250	4.334	1.468	0.757	0.866
2	1.388	1.678	0.974	0.596	0.674
3	1.447	0.651	0.562	0.457	0.631
4	1.537	0.535	0.550	0.473	0.408
5	1.634	0.828	0.617	0.518	0.403
6	1.734	3.187	0.619	0.531	0.464
7	1.833	4.927	0.540	0.471	0.674
8	1.933	4.912	0.618	0.553	0.454
9	2.033	4.248	0.564	0.578	0.377
10	2.133	3.763	0.619	0.641	0.443
	Unstable	Unstable	Stable	Stable	Stable

Table 1. Monte Carlo Averaged Sum Squared Error Performance for Predictors - Old Problem

Sample	Mean-State Predictor	Linearized Predictor	Gaussian Sum Predictor	Floating Grid Predictor
1	2.200	1.665	11.436*	0.955
2	2.400	1.928	1.723	1.020
3	2.600	2.187	1.495	1.006
4	2.800	2.229	1.321	1.083
5	3.000	2.190	1.208	1.118
6	3.200	2.145	1.456	1.359
7	3.400	1.830	1.625	1.521
8	3.600	2.105	1.437	1.370
9	3.800	2.136	1.423	1.132
10	4.000	2.349	1.286	1.316
11	4.200	2.370	1.431	1.493
12	4.400	2.114	1.533	1.515
13	4.600	1.883	1.475	1.345
14	4.800	1.752	1.398	1.268
15	5.000	1.752	1.553	1.441
16	5.200	1.850	1.617	1.534
17	5.400	1.627	1.275	1.222
18	5.600	1.561	1.271	1.238
19	5.800	1.620	1.630	1.314
20	6.000	1.688	1.803	1.556

* transient

Table 2. Monte Carlo Averaged Sum-Squared Error Performance for Predictors - New Problem

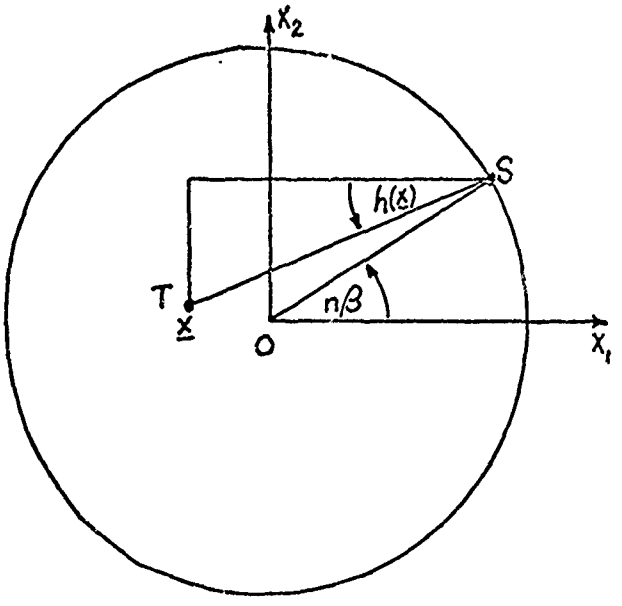


Figure 1. Sensor Geometry - Old Problem

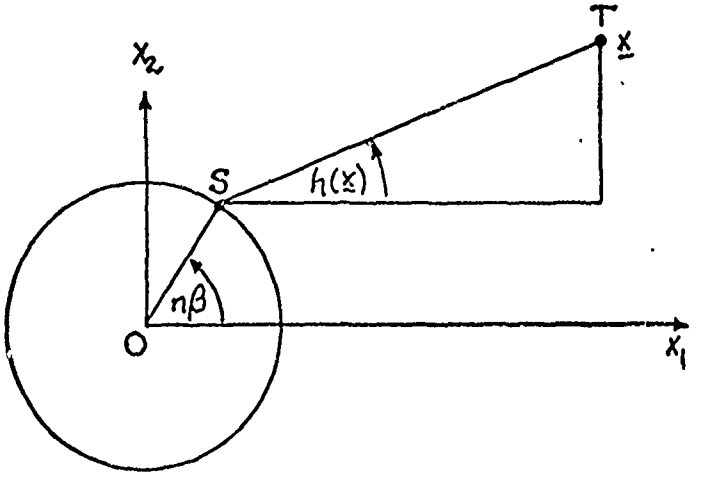


Figure 2. Sensor Geometry - New Problem

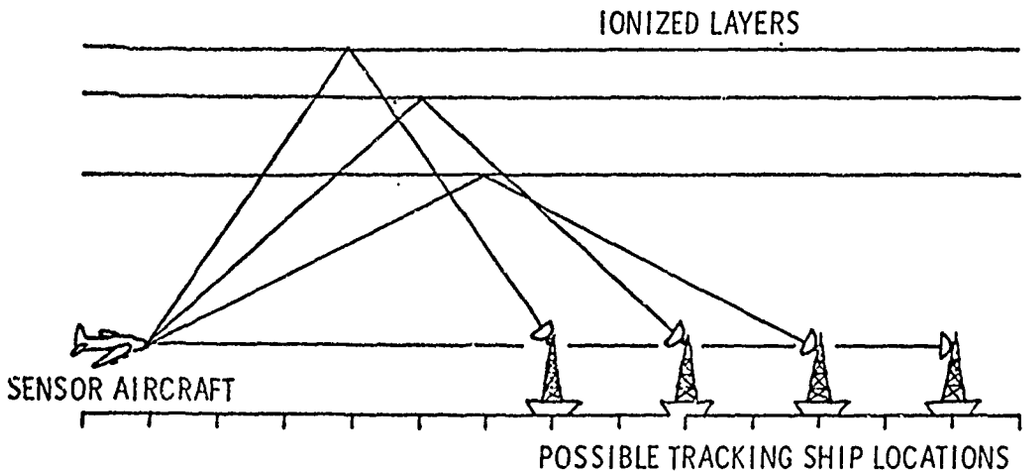


Figure 3. Multimodal Range Ambiguity - Detector Geometry

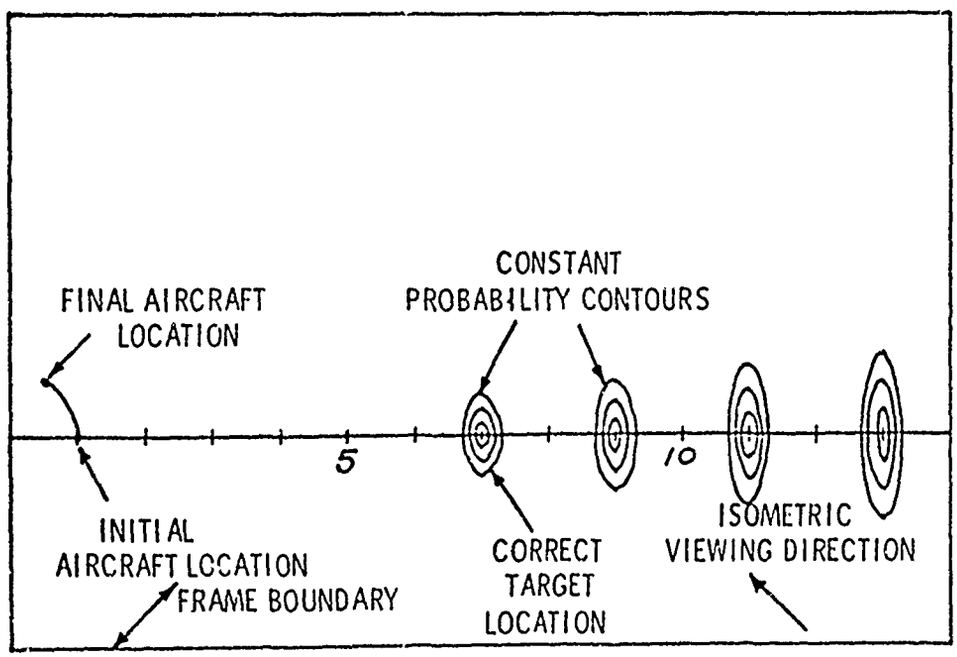


Figure 4. Multimodal Range Ambiguity - Estimator Geometry

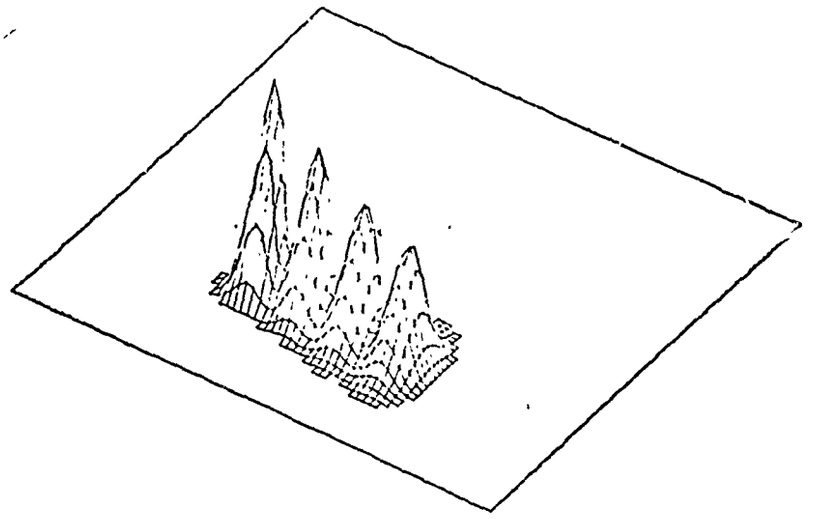
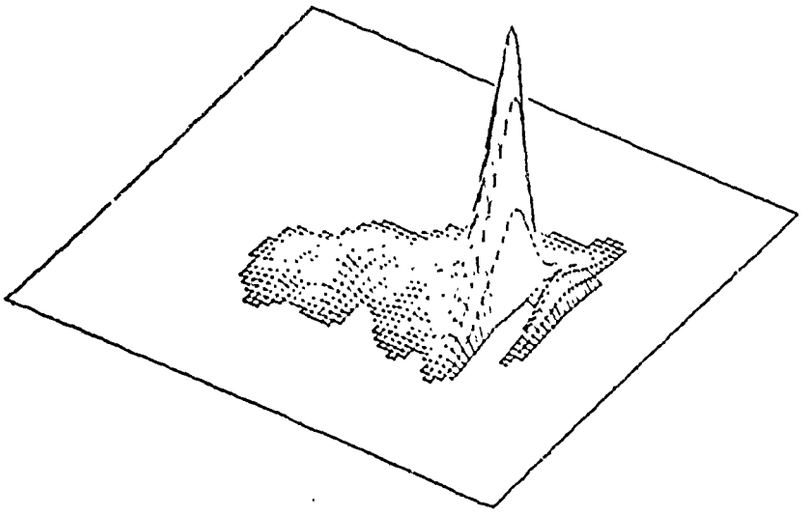
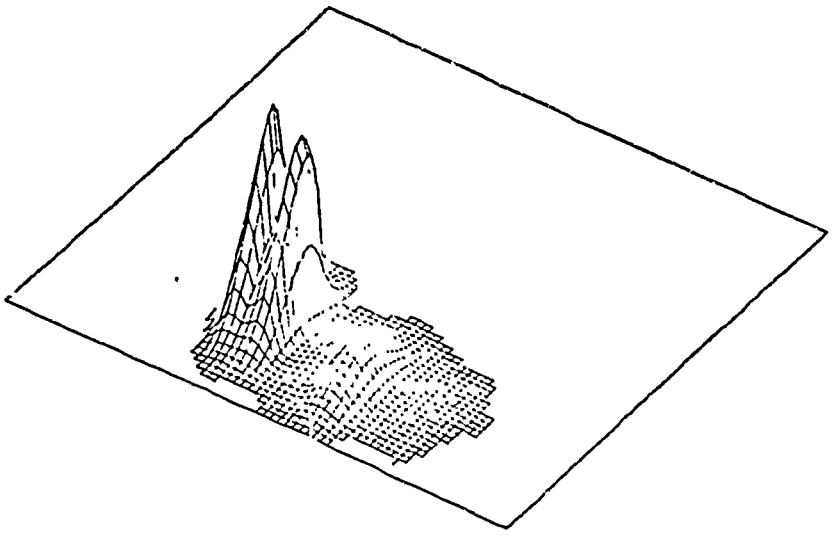
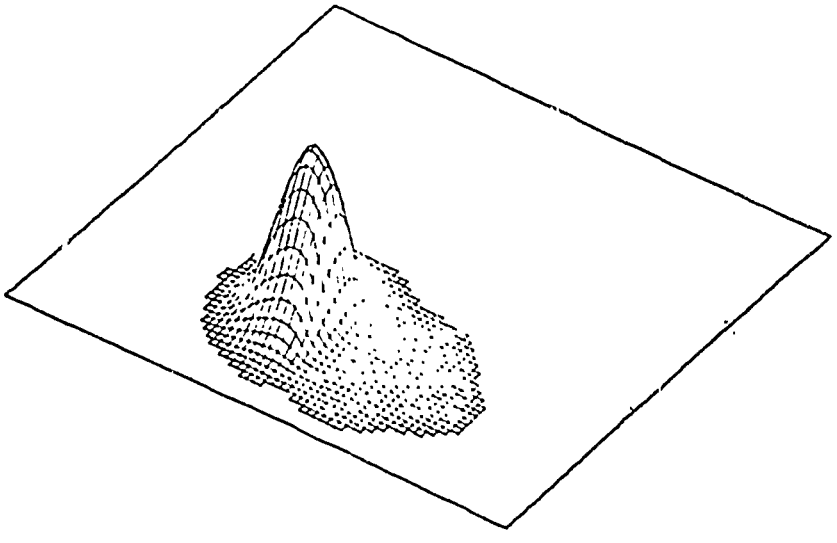


Figure 5. Multimodal Conditional Densities - Typical Examples:
a. Initial Density b. First Solution
c. Solution Transition d. Final Solution







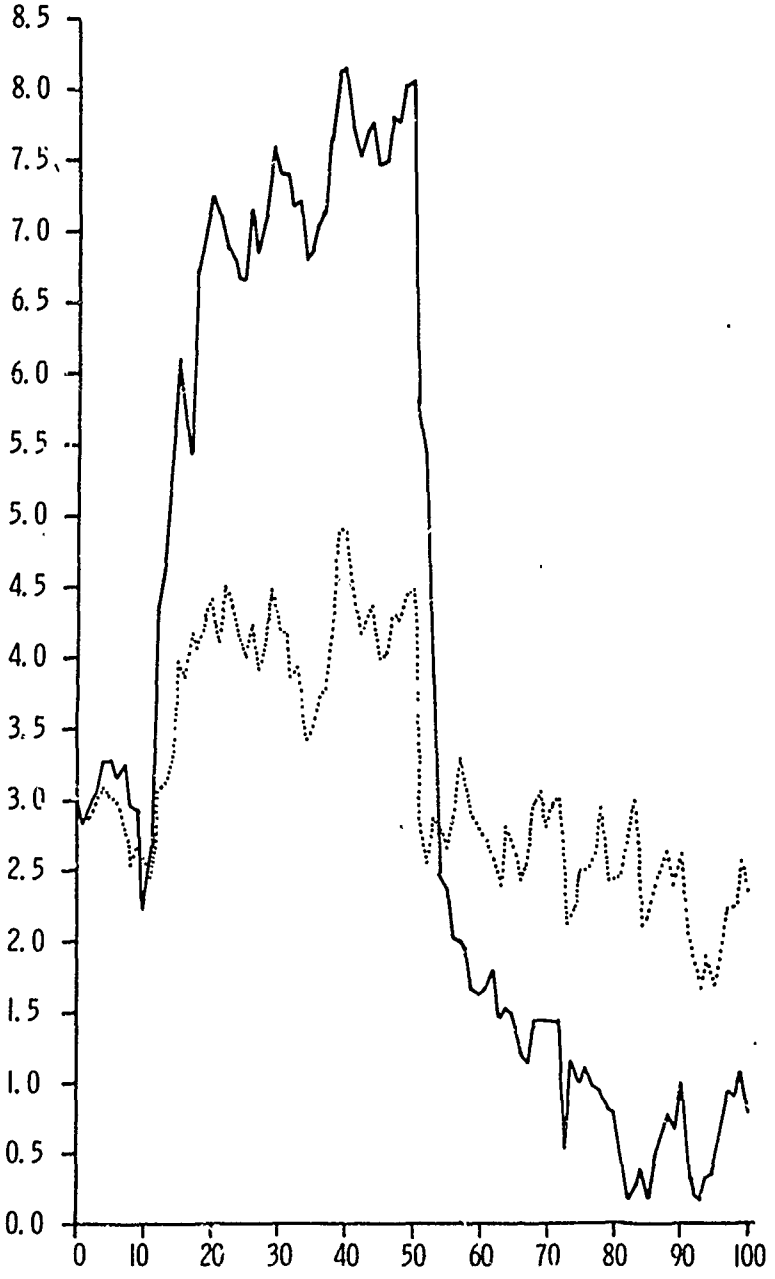


Figure 6. Absolute Error Performance of Optimal and Linearized Predictors for Multimodal Problem