MASTER

K-1685

DMIN1 AND DMIN2
TWO SUBROUTINES FOR THE UNCONSTRAINED
MINIMIZATION OF A FUNCTION
OF SEVERAL VARIABLES

RELEASED FOR ANNOUNCEMENT
IN NUCLEAR SCIENCE ABSTRACTS

AUTHOR:

G. W. Stewart, III

**UNION
CARBIDE**

UNION CARBIDE CORPORATION
NUCLEAR DIVISION

Operating the

OAK RIDGE GASEOUS DIFFUSION PLANT • OAK RIDGE NATIONAL LABORATORY
OAK RIDGE Y-12 PLANT • PADUCAH GASEOUS DIFFUSION PLANT

*For the Atomic Energy Commission
Under U.S. Government Contract W7405 eng 26*

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency Thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

Printed in the United States of America. Available from Clearinghouse for Federal
Scientific and Technical Information, National Bureau of Standards,
U.S. Department of Commerce, Springfield, Virginia 22151
Price: Printed Copy \$3.00; Microfiche \$0.65

LEGAL NOTICE

This report was prepared as an account of Government sponsored work. Neither the United States, nor the Commission, nor any person acting on behalf of the Commission:

- A. Makes any warranty or representation, expressed or implied, with respect to the accuracy, completeness, or usefulness of the information contained in this report, or that the use of any information, apparatus, method, or process disclosed in this report may not infringe privately owned rights; or
- B. Assumes any liabilities with respect to the use of, or for damages resulting from the use of any information, apparatus, method, or process disclosed in this report.

As used in the above, "person acting on behalf of the Commission" includes any employee or contractor of the Commission, or employee of such contractor, to the extent that such employee or contractor of the Commission, or employee of such contractor prepares, disseminates, or provides access to, any information pursuant to his employment or contract with the Commission, or his employment with such contractor.

H.C. \$ 3.00; MN .65

Report Date: January 19, 1967

Report Number: K-1685

Subject Category: Mathematics

and

Computers

RELEASED FOR ANNOUNCEMENT
IN NUCLEAR SCIENCE ABSTRACTS

DMIN1 AND DMIN2
TWO SUBROUTINES FOR THE UNCONSTRAINED
MINIMIZATION OF A FUNCTION
OF SEVERAL VARIABLES
G. W. Stewart, III

LEGAL NOTICE

This report was prepared as an account of Government sponsored work. Neither the United States, nor the Commission, nor any person acting on behalf of the Commission:
A. Makes any warranty or representation, expressed or implied, with respect to the accuracy, completeness, or usefulness of the information contained in this report, or that the use of any information, apparatus, method, or process disclosed in this report may not infringe privately owned rights; or
B. Assumes any liabilities with respect to the use of, or for damages resulting from the use of any information, apparatus, method, or process disclosed in this report.
As used in the above, "person acting on behalf of the Commission" includes any employee or contractor of the Commission, or employee of such contractor, to the extent that such employee or contractor of the Commission, or employee of such contractor prepares, disseminates, or provides access to, any information pursuant to his employment or contract with the Commission, or his employment with such contractor.

UNION CARBIDE CORPORATION
NUCLEAR DIVISION
Computer Sciences Center
Oak Ridge Gaseous Diffusion Plant
Oak Ridge, Tennessee

THIS PAGE
WAS INTENTIONALLY
LEFT BLANK

ABSTRACT

This report contains descriptions and listings of the FORTRAN IV subroutines DMIN1 and DMIN2 for the unconstrained minimization of a function of several variables. The programs are based on a method of Davidon.

THIS PAGE
WAS INTENTIONALLY
LEFT BLANK

Purpose

DMIN1 and DMIN2 are FORTRAN IV subroutines to find the minimum of a function with no constraints on its independent variables. DMIN1 is based on a method of Davidon [2] as modified by Fletcher and Powell [1]. It requires that the user furnish a subroutine to evaluate the gradient of the function to be minimized. DMIN2 approximates the gradient by difference quotients according to an algorithm given in [3].

Method

Both DMIN1 and DMIN2 are descent methods which proceed toward the minimum of the function by a sequence of linear minimizations. The directions along which the function is minimized are chosen so that if the function is quadratic in its n independent variables its minimum will be found after n linear searches. Specifically DMIN1 starts at a point x_0 by evaluating the gradient g_0 . The first direction s_0 is taken as $-H_0 g_0$, where H_0 is the identity matrix. A linear search is then performed and a minimum at $x_1 = x_0 + \sigma_0 s_0$ is found along s_0 . A new matrix is then calculated by setting $y_0 = g_1 - g_0$ and

$$H_1 = H_0 + \sigma_0 \frac{s_0 s_0^T}{s_0^T y_0} - \frac{H_0 y_0 y_0^T H_0}{y_0^T H_0 y_0}.$$

The entire process is then repeated from x_1 .

DMIN2 is essentially the same as DMIN1 except that the components of the gradient vector are computed using difference quotients. The step sizes for the differencing are chosen to balance off truncation and cancellation error.

DMIN1 and DMIN2 claim to have converged to a minimum when the components of both s_i and $\sigma_i s_i$ fall below a level specified by the user. The programs will also stop if some σ_i is zero, if some iteration fails to change the function value, if the norm of the gradient vector underflows the machine, or if a specified maximum number of linear minimizations is exceeded. These conditions are often caused by overly stringent convergence criteria. In fact, if the s_i are small and the σ_i are about unity in the last few iterations the method has probably converged, whatever the exit taken.

Usage

DMIN1 is called by the statement

```
CALL DMIN1(X,F,N,FO,EPS,EVAL,GRAD,NIT,IW,IC),
```

where

- X is a singly subscripted array which contains an initial approximation to the minimum. Upon return X contains the point at which the function is a minimum.
- F contains on return the function value at the minimum.
- N is the number of independent variables, i.e. the number of components of X.
- FO is a lower bound for the function value. It is used in computing step sizes for linear minimizations.
- EPS is an array containing the desired accuracy in the components of X on return.
- EVAL(X,F) is the name of a user coded subroutine which gives a point in the array X returns a function value in F. The name must be typed EXTERNAL.
- GRAD(X,G) is the name of a user coded subroutine which given a point in the array X returns a gradient in the array G. The name must be typed EXTERNAL.
- NIT is the maximum number of iterations to be performed.

IW =0, no write out.

 =1, write out the gradient vector, the direction of each linear minimization, and the function value before and after each linear minimization.

 =2, in addition to the above write out the function values during the course of the linear minimizations.

IC upon return IC.

 =1 normal convergence.

 =2 the derivative of the function along the direction for linear minimization was not negative.

 =3 no progress in the linear minimization.

 =4 the linear minimization failed to change the function value.

 =5 the maximum number of linear minimizations was exceeded.

DMIN2 is called by

CALL DMIN2(X,F,N,FO,EPS,DRV,EVAL,ETA,NIT,IW,IC)

where

X,F,N,FO,EPS,EVAL,NIT, and IC are the same as in the calling sequence for DMIN1.

IW =3 write out the function values calculated in evaluating the gradient in addition to the usual write outs for DMIN1.

DRV is an array containing initial step sizes for the difference approximations to derivatives.

ETA is an estimate of the relative accuracy of the function evaluations; i.e. if f is the true value and f^* is the computed value of the function, then

$$f^* = f(1+\epsilon), \quad |\epsilon| \leq \eta.$$

Restrictions

Both DMIN1 and DMIN2 must be used with the deck PARMIN. The programs are dimensioned to handle functions in up to twenty variables. The algorithm used by DMIN2 for computing derivatives requires an estimate of the number of significant figures contained in the floating point word of the machine on which the program is being run. This information is contained in the variable EM which is set at the beginning of the program. In the listing given here EM is set to 10^{-8} corresponding to the eight significant figures of the IBM 7090 floating point word.

References

1. W. C. Davidon (1959): "Variable Metric Method for Minimization" A.E.C. Research and Development Report, ANL-5990.
2. R. Fletcher and M. J. D. Powell (1963): "A Rapidly Convergent Descent Method for Minimization" Comp. J. V. 6, p. 163.
3. G. W. Stewart, III, "A Modification of Davidon's Minimization Method to Accept Difference Approximations to Derivatives" to appear in the Journal of the A.C.M.

OUTPUT FROM DMIN2

*	7.3177399E-01	1.6402380E-01	8.7605463E-03	1
*	7.3179433E-01	1.6403223E-01	8.7538242E-03	2
G		-4.8175057E-01	-3.6299582E 00	3
C		1.8921951E 01	1.8208964E 02	4
D		-1.4190306E-02	2.2530268E-02	5
F	7.3176993E-01	1.6403223E-01	8.7605463E-03	6
*	7.3058016E-01	1.0000000E 00		7
*	7.1287173E-01	5.0806538E-01		8
F	7.1287173E-01	5.0806538E-01		9
	56			10

LINES 1-2 FUNCTION VALUES AND COORDINATES USED IN COMPUTING THE GRADIENT
 LINE 3 COMPUTED GRADIENT
 LINE 4 ESTIMATES OF SECOND DERIVATIVES ALONG THE COORDINATE AXES
 LINE 5 DIRECTION FOR LINEAR MINIMIZATION
 LINE 6 STARTING FUNCTION VALUE AND POINT FOR THE LINEAR MINIMIZATION
 LINES 7-8 FUNCTION VALUES AND FRACTION OF THE DIRECTION TAKEN IN THE
 COURSE OF THE LINEAR MINIMIZATION
 LINE 9 FINAL FUNCTION VALUE AND FRACTION OF THE DIRECTION
 LINE 10 CUMULATIVE NUMBER OF FUNCTION EVALUATIONS

LISTING OF DMINI

```

$IBFTC DVDSNI
  SUBROUTINE DMINI(XD,FD,NN,FMIN,EPS,EVAL,GRAD,NLIN,WRITE,CONV)
  DIMENSION XD(1),EPS(1)
  DIMENSION H(2,2),X(20),G(20),GI(20),DEL(20),E(4),EE(4),F(4)
  LOGICAL IDENT
  INTEGER CONV,WRITE,COUNT
C
C   INITIALIZE THE PROGRAM
C
  N # NN
  FM # FMIN
  ILIN # 0
  COUNT # 1
  CALL INITPM(E,F,EE,A)
  CALL EVAL(XD,FD)
  CALL GRAD(XD,G)
  IF(WRITE.GT.0) WRITE(6,2004) FD,(XD(1),I#1,N)
C
C   SET H EQUAL TO THE IDENTITY MATRIX
C
  5 IDENT # .TRUE.
  DO 10 I#1,N
  DO 10 J#1,N
  10 H(I,J) # (I/J)*(J/I)
C
C   SET UP FOR A LINEAR MINIMIZATION
C
  15 D # 0.
  EP # 1.
  DO 25 I#1,N
  DEL(I) # 0.
  DO 20 J#1,N
  20 DEL(I) # DEL(I) - H(I,J)*G(J)
  IF(DEL(I).EQ.0.) GO TO 25
  EP # AMINI(EP,ABS(EPS(I)/DEL(I)))
  D # D + G(I)*DEL(I)
  25 CONTINUE
  EP # .05*EP
C
C   IF THE PREDICTED DIRECTION IS UPHILL, RESET H TO THE IDENTITY
C   MATRIX. IF H # I, RETURN.
C
  IF(D.LT.0.) GO TO 30
  IF(.NOT.IDENT) GO TO 5
  CONV # 2
  GO TO 500
  30 IF(FD.LE.FM) FM # -1.E+30
  E(2) # AMINI(1.,2.*(FM-FD)/D)
C
C   PERFORM THE LINEAR MINIMIZATION
C
  100 IF(WRITE.GT.0) WRITE(6,2005) (G(I),I#1,N)
  IF(WRITE.GT.0) WRITE(6,2000) EP,(DEL(I),I#1,N)
  IF(WRITE.GT.0) WRITE(6,2001) FD,(XD(1),I#1,N)
  F(1) # FD
  E(1) # 0.

```



```

      KKK # 0
103 DO 105 I#1,N
105 X(I) # X0(I) + E(2)*DEL(I)
      CALL EVAL(X,F(2))
      COUNT # COUNT + 1
      IF(WRITE.GT.1) WRITE(6,2002) F(2),E(2)
      ED # .5*D*E(2)**2/(D*E(2)+F(1)-F(2))
      IF(ED.LE.0.) ED # 2.*E(2)
      IF(F(2).LT.F(1)) GO TO 120
      E(2) # ED
      KKK # KKK + 1
      IF(KKK.LT.2) GO TO 103
      F(3) # F(2)
      F(2) # F0
      E(3) # E(2)
      E(2) # 0.
      E(1) # -E(3)
      DO 110 I#1,N
110 X(I) # X0(I) + E(1)*DEL(I)
      CALL EVAL(X,F(1))
      COUNT # COUNT + 1
      IF(WRITE.GT.1) WRITE(6,2002) F(1),E(1)
      GO TO 150
120 LOWEST # 2
      IF(ED.GT.3.*E(2)) ED # 3.*E(2)
      IF(ABS(E(2)-ED).LT.EP) ED # E(2) + 1.*EP
      IF(ABS(E(2)-ED).LT..03*ABS(E(2))) ED # 1.*E(2)
      DO 130 I#1,N
130 X(I) # X0(I) + ED*DEL(I)
      IF(ED.GT.F(2)) GO TO 140
      E(3) # E(2)
      E(2) # ED
      F(3) # F(2)
      CALL EVAL(X,F(2))
      COUNT # COUNT + 1
      IF(WRITE.GT.1) WRITE(6,2002) F(2),E(2)
      GO TO 150
140 E(3) # ED
      CALL EVAL(X,F(3))
      COUNT # COUNT + 1
      IF(WRITE.GT.1) WRITE(6,2002) F(3),E(3)
150 CALL PMIN(0)
160 LOWEST # 1
      DO 165 I#2,3
165 IF(F(I).LT.F(LOWEST)) LOWEST # I
      IE # 2. + SIGN(1.,EE(2))
      IF(A.LE.0.) IE # 4-IE
      IF(A.LE.0. .OR. ABS(EE(2)).GT.ABS(3.*EE(IE))) EE(2) # 3.*EE(IE)
      EEE # E(2) + EE(2)
      IF(ABS(EEE-E(LOWEST)).LT.EP) GO TO 250
      IF(ABS(EEE-E(LOWEST)).LT..03*ABS(E(LOWEST))) GO TO 250
      IF(EE(IE).LT.EE(2)) IE # IE + 1
      IF(IE.EQ.4) GO TO 180
      DO 170 LL#IE,3
      L # 3-LL+IE
      E(L+1) # E(L)
170 F(L+1) # F(L)
180 E(IE) # EEE

```

```

DO 190 I#1,N
190 X(I) # XD(I) + EEE*DEL(I)
CALL EVAL(X,F(IE))
COUNT # COUNT + 1
IF(WRITE.GT.1) WRITE(6,2002) F(IE),E(IE)
IF(IE.EQ.1) GO TO 150
KKK # 1
IF(IE.EQ.4) GO TO 220
IF(F(1).GT.F(4)) GO TO 200
CALL PMIN(0)
IF(E(2)+EE(2).LT.E(4) .AND. A.GT.0.) GO TO 160
GO TO 210
200 KKK # 2
CALL PMIN(1)
IF(E(3)+EE(2).GT.E(1) .AND. A.GT.0.) GO TO 220
210 KKK # 1
IF(F(2).LT.F(1) .AND. F(2).LE.F(3) .OR.
I F(2).LE.F(1) .AND. F(2).LT.F(3)) GO TO 150
220 DO 230 I#1,3
E(I) # E(I+1)
230 F(I) # F(I+1)
GO TO (150,160),KKK
C
C END OF THE LINEAR MINIMIZATION
C
250 IF(WRITE.GT.0) WRITE(6,2001) F(LOWEST),E(LOWEST)
IF(WRITE.GT.0) WRITE(6,2003) COUNT
C
C IF THERE WAS NO MOTION, RETURN.
C
IF(E(LOWEST).NE.0.) GO TO 260
CONV # 3
GO TO 500
C
C IF THE FUNCTION VALUE WAS NOT CHANGED, RETURN.
C
260 IF(F(LOWEST).NE.FD) GO TO 270
CONV # 4
GO TO 500
C
C TEST FOR CONVERGENCE
C
270 FD # F(LOWEST)
CONV # 1
ETEST # AMAX1(1.,ABS(E(LOWEST)))
DO 280 I#1,N
IF(ABS(ETEST*DEL(I)).GT.ABS(EPS(I))) CONV # 0
DEL(I) # E(LOWEST)*DEL(I)
XD(I) # XD(I) + DEL(I)
280 GI(I) # G(I)
IF(CONV.EQ.1) GO TO 500
C
C IF THE MAXIMUM NUMBER OF LINEAR MINIMIZATIONS HAS BEEN EXCEEDED,
C RETURN.
C
ILIN # ILIN + 1
CONV # 5
IF(ILIN.GE.NLIN) GO TO 500

```

```

      CALL GRAD(XD,G)
C
C   IF THE MINIMUM WAS FOUND ALONG -DEL, RESET H TO THE IDENTITY
C   MATRIX.
C
      IF(E(LOWEST).LT.0.) GO TO 5
C
C   MODIFY H AND GO BACK FOR ANOTHER ITERATION
C
290 IDENT # .FALSE.
      A # 0.
      DO 300 I#1,N
        G(I) # G(I) - G(I)
300 A # A + G(I)*DEL(I)
      B # 0.
      DO 320 I#1,N
        X(I) # 0.
        DO 310 J#1,N
310 X(I) # X(I) + H(I,J)*G(I)
320 B # B - X(I)*G(I)
      DO 330 I#1,N
        DO 330 J#1,N
          H(I,J) # H(I,J) + DEL(I)*DEL(J)/A + X(I)*X(J)/B
330 H(J,I) # H(I,J)
      GO TO 15
C
C   RETURN
C
500 IF(WRITE.GT.0) WRITE(6,2001) FD,(XD(I),I#1,N)
      IF(WRITE.GT.0) WRITE(6,2003) CONV
      RETURN
2001 FORMAT(3H DIPE15.7,2X,7E15.7/(3H 17X,7E15.7))
2002 FORMAT(3H FIPE15.7,2X,7E15.7/(3H 17X,7E15.7))
2003 FORMAT(3H *IPE15.7,2X,7E15.7/(3H 17X,7E15.7))
2004 FORMAT(3H *IPE15.7,2X,7E15.7/(3H 17X,7E15.7))
2005 FORMAT(3H G,17X,1P7E15.7/(3H 17X,7E15.7))
      END
$IBFTC PARAMN
      SUBROUTINE INITPM(E,F,EE,A)
      DIMENSION E(1),F(1),EE(1)
      RETURN
      ENTRY PMIN(I)
      EE(1) # E(I+1) - E(I+2)
      EE(3) # E(I+3) - E(I+2)
      DF1 # EE(1)*(F(I+3) - F(I+2))
      DF3 # EE(3)*(F(I+1) - F(I+2))
      EE(2) # .5*(EE(1)*DF1-EE(3)*DF3)/(DF1-DF3)
      A # (DF3-DF1)/(EE(1)*EE(3)*(EE(1)-EE(3)))
      RETURN
      END

```

LISTING OF DMIN2

```

$IBFTC DVDSN2
SUBROUTINE DMIN2(XD,FD,NN,FMIN,EPS,DRV,EVAL,ETA,NLIN,WRITE,CONV)
  DIMENSION XD(1),EPS(1),DRV(1)
  DIMENSION H(20,20),X(20),G(20),GI(20),Y(20),DEL(20),C(20),
  I      E(4),EE(4),F(4)
  LOGICAL IDENT
  INTEGER CONV,WRITE,COUNT

C
C   INITIALIZE THE PROGRAM
C
  EM # 1.E-8
  CALL INITPM(E,F,EE,A)
  FM # FMIN
  N # NN
  ILIN # 0
  COUNT # 1
  LOWEST # 1
  E(1) # 1.
  CALL EVAL(XD,FD)
  IF(WRITE.GT.0) WRITE(6,2000) FD,(XD(1),I#1,N)
  IF(WRITE.GT.2) WRITE(6,2007)
  DO 10 I#1,N
    X(I) # XD(I)
  5  XD(I) # X(I) + DRV(I)
    CALL EVAL(XD,FG)
    COUNT # COUNT + 1
    IF(WRITE.GT.2) WRITE(6,2001) FG,(XD(J),J#1,N)
    IF(FG.NE.FD) GO TO 7
    DRV(I) # 2.*DRV(I)
    GO TO 5
  7  G(I) # (FG - FD)/DRV(I)
  10 XD(I) # X(I)

C
C   SET H EQUAL TO THE IDENTITY MATRIX
C
  20 IDENT # .TRUE.
  DO 30 I#1,N
    DO 25 J#1,N
      25 H(I,J) # 0.
    H(I,I) # 1.
  30 C(I) # 1.
    IF(WRITE.GT.0) WRITE(6,2002) (G(I),I#1,N)
    IF(WRITE.GT.0) WRITE(6,2003) (C(I),I#1,N)

C
C   SET UP FOR A MINIMIZATION ALONG A LINE
C
  50 D # 0.
  EP # 1.
  DO 60 I#1,N
    DEL(I) # 0.
    DO 55 J#1,N
      55 DEL(I) # DEL(I) - H(I,J)*G(J)
    IF(DEL(I).EQ.0.) GO TO 60
    EP # AMIN1(EP,ABS(EPS(I)/DEL(I)))
    D # D + G(I)*DEL(I)
  60 CONTINUE

```

```

EP # .05*EP
IF(D.LT.0.) GO TO 70
IF(.NOT.IDENT) GO TO 20
CONV # 2
GO TO 500
70 IF(FD.LE.FM) FM # -1.E20
E(2) # AMIN1(1.,2.*(FM-FD)/D)
100 IF(WRITE.GT.0) WRITE(6,2004) EP,(DEL(I),I#1,N)
IF(WRITE.GT.0) WRITE(6,2005) FD,(X0(I),I#1,N)
F(1) # FD
E(1) # 0.
KKK # 0
103 DO 105 I#1,N
105 X(I) # X0(I) + E(2)*DEL(I)
CALL EVAL(X,F(2))
COUNT # COUNT + 1
IF(WRITE.GT.1) WRITE(6,2001) F(2),E(2)
IF(F(2).NE.F(1)) GO TO 107
E(2) # 2.*E(2)
GO TO 103
107 ED # .5*D*E(2)**2/(D*E(2)+(F(1)-F(2)))
IF(ED.LE.0.) ED # 2.*E(2)
IF(F(2).LT.F(1)) GO TO 120
E(2) # ED
KKK # KKK + 1
IF(KKK.LT.8) GO TO 103
F(3) # F(2)
F(2) # FD
E(3) # E(2)
E(2) # 0.
E(1) # -E(3)
DO 110 I#1,N
110 X(I) # X0(I) + F(1)*DEL(I)
CALL EVAL(X,F(1))
COUNT # COUNT + 1
IF(WRITE.GT.2) WRITE(6,2001) F(1),E(1)
GO TO 150
120 LOWEST # 2
IF(ED.GT.3.*E(2)) ED # 3.*E(2)
IF(ABS(E(2)-ED).LT.EP) ED # E(2) + 1.*FP
IF(ABS(E(2)-ED).LT..03*ABS(E(2))) ED # 1.*E(2)
DO 130 I#1,N
130 X(I) # X0(I) + ED*DEL(I)
IF(ED.GT.E(2)) GO TO 140
E(3) # E(2)
E(2) # ED
F(3) # F(2)
CALL EVAL(X,F(2))
COUNT # COUNT + 1
IF(WRITE.GT.1) WRITE(6,2001) F(2),E(2)
GO TO 150
140 E(3) # ED
CALL EVAL(X,F(3))
COUNT # COUNT + 1
IF(WRITE.GT.1) WRITE(6,2001) F(3),E(3)
150 CALL PMIN(0)
160 LOWEST # 1
DO 165 I#2,3

```

```

165 IF(F(1).LT.F(LOWEST)) LOWEST # I
    IE # 2. + SIGN(1.,EE(2))
    IF(A.LE.0.) IE # 4-IE
    IF(A.LE.0. .OR. ABS(EE(2)).GT.ABS(3.*EE(IE))) EE(2) # 3.*EE(IE)
    EEE # E(2) + EE(2)
    IF(ABS(EEE-E(LOWEST)).LT.EP) GO TO 250
    IF(ABS(EEE-E(LOWEST)).LT..03*ABS(E(LOWEST))) GO TO 250
    IF(EE(IE).LT.EE(2)) IE # IE + 1
    IF(IE.EQ.4) GO TO 180
    DO 170 LL#IE,3
    L # 3-LL+IE
    E(L+1) # E(L)
170 F(L+1) # F(L)
180 E(IE) # EEE
    DO 190 I#1,N
190 X(I) # X(I) + EEE*DEL(I)
    CALL EVAL(X,F(IE))
    COUNT # COUNT + 1
    IF(WRITE.GT.1) WRITE(6,2001) F(IE),E(IE)
    IF(IE.EQ.1) GO TO 150
    KKK # 1
    IF(IE.EQ.4) GO TO 220
    IF(F(1).GT.F(4)) GO TO 200
    CALL PMIN(0)
    IF(E(2)+EE(2).LT.E(4) .AND. A.GT.0.) GO TO 160
    GO TO 210
200 KKK # 2
    CALL PMIN(1)
    IF(E(3)+EE(2).GT.E(1) .AND. A.GT.0.) GO TO 220
210 KKK # 1
    IF(F(2).LT.F(1) .AND. F(2).LE.F(3) .OR.
    I - F(2).LE.F(1) .AND. F(2).LT.F(3)) GO TO 150
220 DO 230 I#1,3
    E(I) # E(I+1)
230 F(I) # F(I+1)
    GO TO (150,160),KKK
C
C   END OF MINIMIZATION ALONG DEL
C
250 IF(WRITE.GT.0) WRITE(6,2005) F(LOWEST),E(LOWEST)
    IF(WRITE.GT.0) WRITE(6,2006) COUNT
C
C   IF THERE WAS NO MOTION, RETURN.
C
    IF(E(LOWEST).NE.0.) GO TO 260
    CONV # 3
    GO TO 500
C
C   IF THE FUNCTION VALUE WAS NOT CHANGED, RETURN.
C
260 IF(F(LOWEST).NE.FD) GO TO 270
    CONV # 4
    GO TO 500
C
C   TEST FOR CONVERGENCE
C
270 FD # F(LOWEST)
    CONV # 1

```



```

      ETEST # AMAX1(1.,ABS(E(LOWEST)))
      DO 280 I#1,N
      IF(ABS(ETEST*DEL(I)).GT.ABS(EPS(I))) CONV # 0
      DEL(I) # E(LOWEST)*DEL(I)
      XD(I) # XD(I) + DEL(I)
280  G(I) # G(I)
      IF(CONV.EQ.1) GO TO 500

C
C   IF THERE HAVE BEEN TOO MANY MINIMIZATIONS ALONG A LINE, RETURN.
C
      ILIN # ILIN + 1
      CONV # 5
      IF(ILIN.GE.NLIN) GO TO 500

C
C   CALCULATE A NEW GRADIENT
C
      IF(WRITE.GT.2) WRITE(6,2007)
      DO 300 I#1,N
      X(I) # XD(I)
      IF(FD.EQ.0.) GO TO 285
      IF(IDENT) GO TO 285
      IF(G(I).EQ.0.) GO TO 285
      ETAM # AMAX1(ETA,ABS(EM*G(I)*XD(I)/FD))
      IF(G(I)**2.GT.C(I)*ABS(FD)*ETAM) GO TO 282
      DRV(I) # 2.*(ABS(FD)*ABS(G(I))*ETAM/C(I)**2)**.33333333
      DRV(I) # DRV(I)*(1. - ABS(G(I))/(1.5*C(I)*DRV(I) + 2.*ABS(G(I))))
      GO TO 283
282  DRV(I) # 2.*SQRT(ETAM*ABS(FD)/C(I))
      DRV(I) # DRV(I)*(1.-C(I)*DRV(I)/(3.*C(I)*DRV(I)+4.*ABS(G(I))))
283  DRV(I) # SIGN(DRV(I),G(I))
      IF(1.5*ABS(C(I)*DRV(I)/G(I)) .GT. .01) GO TO 295
285  XD(I) # X(I) + DRV(I)
      CALL EVAL(XD,FG)
      COUNT # COUNT + 1
      IF(WRITE.GT.2) WRITE(6,2001) FG,(XD(J),J#1,N)
      IF(FG.NE.FD) GO TO 290
      DRV(I) # 2.*DRV(I)
      GO TO 285
290  G(I) # (FG - FD)/DRV(I)
      GO TO 300
295  DRV(I) # 100.*ABS(FD*ETAM/G(I))
      DRV(I) # ABS(G(I)) + SQRT(G(I)**2 + 200.*ABS(FD)*C(I)*ETAM)
      DRV(I) # 100.*ABS(FD)*ETAM/DRV(I)
      XD(I) # X(I) + DRV(I)
      CALL EVAL(XD,FP)
      COUNT # COUNT + 1
      IF(WRITE.GT.2) WRITE(6,2001) FP,(XD(J),J#1,N)
      XD(I) # X(I) - DRV(I)
      CALL EVAL(XD,FMI)
      COUNT # COUNT + 1
      IF(WRITE.GT.2) WRITE(6,2001) FMI,(XD(J),J#1,N)
      G(I) # .5*(FP-FMI)/DRV(I)
300  XJ(I) # X(I)

C
C   IF THE MINIMUM WAS FOUND ALONG -DEL, GO SET H EQUAL TO C INVERSE
C
      IF(E(LOWEST).LT.0.) GO TO 20
C

```

```

C      MODIFY H AND GO BACK FOR ANOTHER ITERATION
C
      IDENT # .FALSE.
      A # 0.
      DO 310 I#1,N
      Y(I) # G(I) - G1(I)
310  A # A + Y(I)*DEL(I)
      IF(WRITE.GT.0) WRITE(6,2002) (G(I),I#1,N)
      AA # A/E(LOWEST)
      C1 # 1./A - D/AA**2
      C2 # 2./AA
      B # 0.
      DO 330 I#1,N
      C(I) # C(I) + C1*Y(I)**2 + C2*Y(I)*G1(I)
      X(I) # 0.
      DO 320 J#1,N
320  X(I) # X(I) + H(I,J)*Y(J)
330  B # B - X(I)*Y(I)
      IF(WRITE.GT.0) WRITE(6,2003) (C(I),I#1,N)
      DO 340 I#1,N
      IF(C(I).LE.0.) GO TO 20
      DO 340 J#1,N
      H(I,J) # H(I,J) + DEL(I)*DEL(J)/A + X(I)*X(J)/B
340  H(J,I) # H(I,J)
      WRITE(6,2007)
      GO TO 50
C
C      RETURN
C
500  IF(WRITE.GT.0) WRITE(6,2005) F0,(X0(I),I#1,N)
      IF(WRITE.GT.0) WRITE(6,2006) CONV
      RETURN
C
C
2000 FORMAT(3H1 *IPE15.7,2X,7E15.7/(3H 17X,7E15.7))
2001 FORMAT(3H1 *IPE15.7,2X,7E15.7/(3H 17X,7E15.7))
2002 FORMAT(3H0 G17X,1P7E15.7/(3H 17X,7E15.7))
2003 FORMAT(3H C17X,1P7E15.7/(3H 17X,7E15.7))
2004 FORMAT(3H D1PE15.7,2X,7E15.7/(3H 17X,7E15.7))
2005 FORMAT(3H F1PE15.7,2X,7E15.7/(3H 17X,7E15.7))
2006 FORMAT(1H 15)
2007 FORMAT(1H )
      END
$IBFTC PARMIN
      SUBROUTINE INITPM(E,F,EE,A)
      DIMENSION E(1),F(1),EE(1)
      RETURN
      ENTRY PMIN(I)
      EE(1) # E(I+1) - E(I+2)
      EE(3) # E(I+3) - E(I+2)
      DF1 # EE(1)*(F(I+3) - F(I+2))
      DF3 # EE(3)*(F(I+1) - F(I+2))
      EE(2) # .5*(EE(1)*DF1-EE(3)*DF3)/(DF1-DF3)
      A # (DF3-DF1)/(EE(1)*EE(3)*(EE(1)-EE(3)))
      RETURN
      END

```