1 SUMMARY

To calculate the minimum of a general function of \( n \) variables when values of both the first and second derivatives with respect to the variables can be provided.

The method is based on the Newton method and is described by M.D. Hebden in AERE TP.515.

The user can specify the absolute accuracies required in each variable and must give an initial estimate of the solution and provide subroutines to evaluate \( f(x) \) and its first and second derivatives.


2 HOW TO USE THE PACKAGE

2.1 Argument lists

The single precision version

\[
\text{CALL VA21A(FUNCT,DERIV,N,X,F,G,GG,EPS,WORK,IWORK,IPRINT,}\nonumber \\
\quad \text{MAXFN,LP,PASS,IPASS)}
\]

The double precision version

\[
\text{CALL VA21AD(FUNCT,DERIV,N,X,F,G,GG,EPS,WORK,IWORK,IPRINT,}\nonumber \\
\quad \text{MAXFN,LP,PASS,IPASS)}
\]

\text{FUNCT} is the name of a subroutine which must be written by the user and must be declared in an \texttt{EXTERNAL} statement in the subroutine calling VA21A/AD. Its form is specified in §2.2.

\text{DERIV} is the name of a subroutine which must be written by the user and must be declared in an \texttt{EXTERNAL} statement in the subroutine calling VA21A/AD. Its form is specified in §2.3.

\text{N} is an \texttt{INTEGER} variable which must be set by the user to \( n \), the number of variables. It is not altered.

\textbf{Restriction:} \( n \geq 1 \).

\text{X} is a \texttt{REAL} (DOUBLE PRECISION in the D version) array of length \( n \) in which the user must set an initial estimate of the solution. On return it holds the best estimate of the solution found by the subroutine.

\text{F} is a \texttt{REAL} (DOUBLE PRECISION in the D version) variable which the user need not set. On return it holds the smallest value of \( f(x) \) found.

\text{G} is a \texttt{REAL} (DOUBLE PRECISION in the D version) array of length \( n \) which the user need not set. It is used to hold the first derivatives \( \frac{\partial f}{\partial x_i} \).

\text{GG} is a \texttt{REAL} (DOUBLE PRECISION in the D version) array of length \( n^2 \) which the user need not set. It is used to hold the second derivatives \( \frac{\partial^2 f}{\partial x_i \partial x_j} \).

\text{EPS} is a \texttt{REAL} (DOUBLE PRECISION in the D version) array of length \( n \) which the user must set to the absolute accuracies required for the variables. It is not altered by the subroutine.

\text{WORK} is a \texttt{REAL} (DOUBLE PRECISION in the D version) array of length \( 5n \) which the user need not set. It is used as workspace.

\text{IWORK} is an \texttt{INTEGER} array of length \( n \) which the user need not set. It is used as workspace.

\text{IPRINT} is an \texttt{INTEGER} variable which must be set by the user to control the printing required. If IPRINT=0 no
printing is done; otherwise printing takes place every \(|\text{IPRINT}|\) iterations and before return. The iteration number, number of function evaluations, value of \(f\) and value of \(x\) are printed, and if \(\text{IPRINT}<0\) the first and second derivatives are printed too. It is not altered.

\(\text{MAXFN}\) is an \texttt{INTEGER} variable which must be set by the user to a limit on the number of function evaluations to be performed (usually one function evaluation is performed each iteration). The subroutine returns after \(\text{MAXFN}\) function evaluations. It is not altered.

\(\text{LP}\) is an \texttt{INTEGER} variable which must be set by the user to a unit number for the printing or to a non-positive number if printing is to be suppressed.

\(\text{PASS}\) is a \texttt{REAL} (\texttt{DOUBLE PRECISION} in the D version) array of any length which may be used to pass information from the calling subroutine to \texttt{FUNCT} and \texttt{DERIV}. It is not altered by \texttt{VA21A/AD}.

\(\text{IPASS}\) is an \texttt{INTEGER} array of any length which may be used to pass information from the calling subroutine to \texttt{FUNCT} and \texttt{DERIV}. It is not altered by \texttt{VA21A/AD}.

2.2 The subroutine \texttt{FUNCT}

The subroutine \texttt{FUNCT} must be written by the user to specify the function to be minimized. It must have the form

\[
\text{SUBROUTINE FUNCT}(N,X,F,\text{IFL},\text{PASS},\text{IPASS})
\]

\(\text{N}\) is an \texttt{INTEGER} variable which will have the value \(n\), the number of variables. It must not be altered.

\(\text{X}\) is a \texttt{REAL} (\texttt{DOUBLE PRECISION} in the D version) array of length \(n\) in which the current point \(x\) is stored. It must not be altered.

\(\text{F}\) is a \texttt{REAL} (\texttt{DOUBLE PRECISION} in the D version) variable in which the value \(f(x)\) must be returned, unless \(f\) does not have a value at \(x\).

\(\text{IFL}\) is an \texttt{INTEGER} variable which has the value 0 on entry. If \(f\) has no value at \(x\), \(\text{IFL}\) should be returned with the value 1. Otherwise \(\text{IFL}\) should be left with the value 0.

\(\text{PASS}\) is a \texttt{REAL} (\texttt{DOUBLE PRECISION} in the D version) array of any length which may be used to pass information from the calling subroutine. It is not altered by \texttt{VA21A/AD}.

\(\text{IPASS}\) is an \texttt{INTEGER} array of any length which may be used to pass information from the calling subroutine. It is not altered by \texttt{VA21A/AD}.

2.3 The subroutine \texttt{DERIV}

The subroutine \texttt{DERIV} must be written by the user to specify the first and second derivatives of the function to be minimized. It must have the form

\[
\text{SUBROUTINE DERIV}(N,X,G,GG,\text{PASS},\text{IPASS})
\]

\(\text{N}\) is an \texttt{INTEGER} variable which will have the value \(n\), the number of variables. It must not be altered.

\(\text{X}\) is a \texttt{REAL} (\texttt{DOUBLE PRECISION} in the D version) array of length \(n\) in which the current point \(x\) is stored. It must not be altered.

\(\text{G}\) is a \texttt{REAL} (\texttt{DOUBLE PRECISION} in the D version) variable in which the values of the first derivatives \(\partial f/\partial x_i\), \(i=1,2,...,n\) must be returned.

\(\text{GG}\) is a \texttt{REAL} (\texttt{DOUBLE PRECISION} in the D version) array of dimensions \((n,n)\) in which the values of the second derivatives \(\partial^2 f/\partial x_i\partial x_j\), \(i=1,2,...,n\), \(j=1,2,...,i\) must be returned. There is no need to set \(\text{GG}(i,j), j>i\).

\(\text{PASS}\) is a \texttt{REAL} (\texttt{DOUBLE PRECISION} in the D version) array of any length which may be used to pass information from the calling subroutine. It is not altered by \texttt{VA21A/AD}.

\(\text{IPASS}\) is an \texttt{INTEGER} array of any length which may be used to pass information from the calling subroutine. It is not altered by \texttt{VA21A/AD}.
3 GENERAL INFORMATION

Workspace: Provided in the arguments WORK and IWORK.

Use of common: None.

Other routines called directly: _DOT, the private subroutines VA21Y/YD and VA212/ZD, and the user-written subroutines FUNCT and DERIV are called.

Input/output: Output is under the control of arguments IPRINT and LP.

Restrictions: \( n \geq 1 \).

4 METHOD

The method is based on the Newton method and is described by M.D. Hebden in AERE TP.515. The code is a modification of the code VA11 of Hebden (1971).

Software modifications bring the code into line with Fortran 77, give more control over the printing, make the printing clearer, and allow information in variable-length arrays to be passed to the functions FUNCT and DERIV.

The algorithmic alterations principally concern the treatment (by VA21Y/YD) of second derivative matrices that are not positive definite. Hebden’s procedure is explained in §5 of AERE TP.515. It is unnecessarily complicated, implicitly assumes that the second derivatives are of size about unity, and provokes underflow interrupts if they are very small. Instead perturbations \( \kappa_j \) are chosen to make pivot \( j \) as large as any off-diagonal element in row \( j \), \( j=1,2,...,n-1 \) and to make pivot \( n \) non-negative. If the matrix is not positive definite, this makes the last pivot have the value zero; we have retained Hebden’s procedure for calculating a vector in the null space of the matrix actually factorized.

5 EXAMPLE OF USE

As a very simple example, the following code finds the minimum of the function

\[ (x_1-\alpha)^2 + (x_1-\beta x_2)^2 \]

when \( \alpha=1 \) and \( \beta=2 \).

```fortran
INTEGER IWORK(2), IPASS(1)
DOUBLE PRECISION F, X(2), WORK(10), G(2), GG(2,2), EPS(2), PASS(2)
EXTERNAL FUNCT, DERIV
EPS(1) = 0.0005D0
EPS(2) = 0.0005D0
MAXFN = 100
IPRINT = -20
N=2
LP=6
X(1) = 0.0D0
X(2) = 0.0D0
PASS(1) = 1.0D0
PASS(2) = 2.0D0
CALL VA21AD(FUNCT, DERIV, N, X, F, G, GG, EPS, WORK, IWORK, IPRINT, MAXFN, *
* LP, PASS, IPASS)
STOP
END

C SUBROUTINE FUNCT(N,X,F,IFL,PASS,IPASS)
DOUBLE PRECISION X(1), F, S, PASS(2), ALPHA, BETA
INTEGER IPASS(1)
ALPHA = PASS(1)
BETA = PASS(2)
F = (X(1)-ALPHA)**2 + (X(1)-BETA*X(2))**2
RETURN
END
```
SUBROUTINE DERIV(N,X,G,GG,PASS,IPASS)
DOUBLE PRECISION X(1),G(1),GG(N,1),PASS(2),ALPHA,BETA
INTEGER IPASS(1)
ALPHA = PASS(1)
BETA = PASS(2)
G(1) = 2.0D0*(X(1)-ALPHA) + 2.0D0*(X(1)-BETA*X(2))
G(2) = -2.0D0*(X(1)-BETA*X(2))*BETA
GG(1,1) = 4.0D0
GG(1,2) = -2.0D0*BETA
GG(2,1) = -2.0D0
GG(2,2) = 2.0D0*BETA**2
RETURN
END

This produces the following output

ENTRY TO VA21A

ITERATION 1, 1 FUNCTION EVALUATIONS, F = 1.0000000D 00
X = 0.00000000D-01 0.00000000D-01
G = -2.00000000D 00 0.00000000D-01
GG( 1,.) = 4.00000000D 00
GG( 2,.) = -2.00000000D 00 8.00000000D 00

ITERATION 14, 14 FUNCTION EVALUATIONS, F = 5.2523442D-07
X = 9.9933043D-01 4.9952655D-01
G = -7.8445291D-04 -1.1093552D-03
GG( 1,.) = 4.00000000D 00
GG( 2,.) = -2.00000000D 00 8.00000000D 00