1 SUMMARY

This subroutine solves the unconstrained minimization problem

$$\text{minimize} \quad F(\mathbf{x}), \quad \mathbf{x}=(x_1,x_2,...,x_n) .$$

The subroutine is especially effective on problems involving a large number of variables and does not require knowledge about the sparsity structure of the Hessian matrix. It uses the limited memory BFGS method, as described by Liu and Nocedal (On The Limited Memory BFGS Method For Large Scale Optimization, Technical Report NA-03, Department of Electrical Engineering and Computer Science, Northwestern University, 1988). In a typical iteration of this method an approximation H_k to the inverse of the Hessian

$$\left(\frac{\partial^2 F}{\partial x_i \partial x_j}\right)$$

is obtained by applying m BFGS updates to a diagonal matrix H_k^0, using information from the previous m steps. The user specifies the number m, which also determines the amount of storage required by the subroutine.

ATTRIBUTES — Version: 1.1.0. **Types:** Real (single, double). **Calls:** `FD15`, `_AXPY`, `_COPY`, `_DOT`. **Original date:** September 2001. **Remark:** `VA35` is a threadsafe version of `VA15`. **Origin:** Jorge Nocedal, Northwestern University, Illinois. **Licence:** A third-party licence for this package is available without charge.

2 HOW TO USE THE PACKAGE

Although there are both single and double precision versions of the routine available, the user is **strongly advised** to use the double precision version unless single precision on his or her machine actually means 8-byte arithmetic.

2.1 Method of Use

The user is required to calculate the function value F and its gradient

$$G=\left(\frac{\partial F}{\partial x_1},...,\frac{\partial F}{\partial x_n}\right)$$

In order to allow the user complete control over these computations, reverse communication is used. The subroutine must be called repeatedly under the control of the parameter `IFLAG`. Before the initial call, `IFLAG` must be set to zero, and the user must provide the starting point **x**, and the initial values of F and G. A return with `IFLAG=0` indicates successful termination; `IFLAG` < 0 indicates error conditions. On a return with `IFLAG=1` the function and gradient must be evaluated, and the subroutine recalled. The user may provide the diagonal matrices H_k^0 if not satisfied with the default choice (see section 4).

2.2 Initialization

The `VA35I/ID` entry should be called to initialize the user controls to default values and to ensure that the arrays of kept data used in the reverse communication are not unassigned. The initialization should be done before any calls are made to the `VA35A/AD` entry and before any controls are set by the user, see Section 2.4.

The single precision version

```
CALL VA35I(ICNTL,CNTL,KEEP,RKEEP)
```

The double precision version

```
CALL VA35ID(ICNTL,CNTL,KEEP,RKEEP)
```

`ICNTL` is an `INTEGER` array of length 10 that need not be set by the user. On return it contains default values.

CNTL is a REAL (DOUBLE PRECISION in the D version) array of length 5 that need not be set by the user. On return it contains default values.

KEEP is an INTEGER array of length 20 that need not be set by the user.

RKEEP is a REAL (DOUBLE PRECISION in the D version) array of length 40 that need not be set by the user.

2.3 The Argument List

The single precision version

```
CALL VA35A(N,M,X,F,G,DIAGCO,DIAG,EPS,W,IFLAG,ICNTL,CNTL,INFO,KEEP,RKEEP)
```

The double precision version

```
CALL VA35AD(N,M,X,F,G,DIAGCO,DIAG,EPS,W,IFLAG,ICNTL,CNTL,INFO,KEEP,RKEEP)
```

N is an INTEGER variable that must be set by the user to n, the number of components of **x**. It is not altered by the subroutine. Restrictions:N > 0.

M is an INTEGER variable that must be set by the user to m, the number of corrections used in the BFGS update. It is not altered by the subroutine. Values of M less than 3 are not recommended; large values of M will result in excessive storage and computing time. A value in the range 3 ≤ M ≤ 7 is recommended. Restriction: M > 0.

X is a REAL (DOUBLE PRECISION in the D version) array of length N. On initial entry it must be set by the user to the values of the initial estimate of the solution vector. On exit with IFLAG=0, it contains the values of the variables at the best point found (usually the solution).

F is a REAL (DOUBLE PRECISION in the D version) variable. Before initial entry and on a re-entry with IFLAG=1, it must be set by the user to contain the value of the function F evaluated at the point **x** specified in X.

G is a REAL (DOUBLE PRECISION in the D version) array of length N. Before initial entry and on a re-entry with IFLAG=1, it must be set by the user to contain the components of the gradient G evaluated at the point **x** specified in X.

DIAGCO is a LOGICAL variable that must be set to .TRUE. if the user wishes to provide the diagonal matrix H_k^0 at each iteration. Otherwise it should be set to .FALSE., in which case VA35A/AD will use a default value (see section 4). If DIAGCO is set to .TRUE. the subroutine will return at each iteration of the algorithm with IFLAG=2, and the diagonal matrix H_k^0 must be provided in the array DIAG.

DIAG is a REAL (DOUBLE PRECISION in the D version) array of length N. If DIAGCO=.TRUE., then on initial entry or on re-entry with IFLAG=2 DIAG must be set by the user to contain the values of the diagonal matrix H_k^0 Restriction: all elements of DIAG must be positive.

EPS is a REAL (DOUBLE PRECISION in the D version) variable that must be set by the user, and determines the accuracy with which the solution is to be found. The subroutine terminates when $||G|| <$ EPS $\max(1,||X||)$, where $||.||$ denotes the Euclidean norm.

W is a REAL (DOUBLE PRECISION in the D version) array of length N(2M+1)+2M used as workspace for VA35A/AD. This array must not be altered by the user.

IFLAG is an INTEGER variable that must be set to 0 on initial entry to the subroutine. A return with IFLAG < 0 indicates an error (see section 2.5). IFLAG=0 indicates that the subroutine has terminated without detecting errors. On a return with IFLAG=1, the user must evaluate the function F and gradient G. On a return with IFLAG=2, the user must provide the diagonal matrix H_k^0.

ICNTL is an INTEGER array of length 10 that contains integer control parameters, see Section 2.4. This argument is not altered by the subroutine.

CNTL is a REAL (DOUBLE PRECISION in the D version) array of length 5 that contains real control parameters, see Section 2.4. This argument is not altered by the subroutine.

INFO is an INTEGER array of length 5 that need not be set by the user. On return, it holds information on the calculation, see Section 2.4.

KEEP is the INTEGER private workspace array of length 20 initialized by VA35I/ID and must not be altered by the user while IFLAG > 0.

RKEEP is the REAL (DOUBLE PRECISION in the D version) private workspace array of length 40 initialized by VA35I/ID and must not be altered by the user while IFLAG > 0.

2.4 The control and information arrays

The arrays ICNTL and CNTL provide the user with some control over the operations of VA35A/AD. The controls are given default values by VA35I/ID and may then be optionally reset by the user. The integer value controls are:

ICNTL(1) (default = −1) specifies the frequency of the output produced by VA35:

ICNTL(1) < 0 : no output is generated,
ICNTL(1) = 0 : output only at first and last iteration,
ICNTL(1) > 0 : output every ICNTL(1) iterations.

ICNTL(2) (default = 0) specifies the type of output generated:

ICNTL(2) = 0 : iteration count, number of function calls, function value, norm of the gradient, and steplength,
ICNTL(2) = 1 : same as ICNTL(2)=0, plus vector of variables and gradient vector at the initial point,
ICNTL(2) = 2 : same as ICNTL(2)=1, plus vector of variables,
ICNTL(2) = 3 : same as ICNTL(2)=2, plus gradient vector.

ICNTL(3) (default = 6) is used as the unit number for the printing of error messages. This printing may be suppressed by setting ICNTL(3) to a non-positive value.

ICNTL(4) (default = 6) is used as the unit number for the printing of the monitoring information controlled by ICNTL(1) and ICNTL(2).

ICNTL(5) to ICNTL(10) are not used at present and should not be altered.

The real value controls are:

CNTL(1) (default = 0.9) controls the accuracy of the line search subroutine VA35E/ED. If the function and gradient evaluations are inexpensive with respect to the cost of the iteration (which is sometimes the case when solving very large problems) it may be advantageous to set CNTL(1) to a small value. Typical small values are 0.1 and 0.01. Restriction: CNTL(1) should be greater than 1.D-04. If CNTL(1) is less than 1.D-04 it is reset 0.01 and a message indicating this is printed.

CNTL(2) to ICNTL(5) are not used at present and should not be altered.

The INFO array argument which must be of length 5 is used to return information back to the user.

INFO(1) is used as an error indicator. It is set to zero when there are no errors otherwise it is set to one of the negative values listed in Section 2.5. Note that INFO(1) returns the same error indications as the argument IFLAG but returns zero when IFLAG > 0.

INFO(2) is used to indicate that the errors were in the line search phase (see section 2.5).

INFO(3) to INFO(5) are not used.

2.5 Error Messages

When the exit condition IFLAG (or INFO(1)) is less than 0 the subroutine has detected an error. The following negative values of IFLAG are possible:

IFLAG=-1 The line search subroutine failed. The parameter INFO(2) provides more detailed information and may be returned set to one of the following values:

0 Improper input parameters.

2 Relative width of the interval of uncertainty is too small.

3 More than 20 function evaluations have been required at the present iteration.

4 The step is too small.

5 The step is too large.

6 Rounding errors prevent further progress. There may not be a step which satisfies the sufficient decrease and curvature conditions.

`IFLAG=-2` The i-th diagonal element of the diagonal inverse Hessian approximation, given in `DIAG,` is not positive.

`IFLAG=-3` Improper input parameters for `VA35A/AD` (`N` or `M` are not positive).

2.6 Termination Criterion

Convergence is determined to have occurred when $||G|| <$ `EPS` $\max(1,||X||)$, where $||.||$ is the Euclidean norm and `EPS` is user supplied.

3 GENERAL INFORMATION

Use of common: none.

Workspace: see argument W.

Other routines called directly: `SDOT/DDOT, SAXPY/DAXPY, SCOPY/DCOPY, FD15A/AD, VA35B/BD, VA35E/ED, VA35F/FD.`

Input/output: No input; diagnostic messages on unit `ICNTL(4)` and error messages on unit `ICNTL(3).`

4 METHOD

The method implemented by `VA35A/AD` is the limited memory BFGS algorithm, as described in (1) (see also (3)). It is intended for problems with many variables. In this method quasi-Newton corrections are stored separately, and when the available storage is used up, the oldest correction is deleted to make space for the new one. The user specifies the number m of BFGS corrections that should be stored. `VA35A/AD` requires a total of $2m(n+1)+4n$ storage locations.

During the the first m iterations the method is identical to the BFGS method. Afterwards, an inverse Hessian approximation H_k is obtained by applying m BFGS updates to the diagonal matrix H_k^0, using information from the m previous iterations. The choice of H_k^0 influences greatly the performance of the method. In `VA35A/AD` the matrix H_k^0 is assumed to be diagonal, and is set to the identity matrix at the beginning, unless the user provides a different estimate. `VA35A/AD` allows the user to supply the value of H_k^0 at every iteration of the algorithm. If the user is unwilling to do so, the subroutine sets the default value $H_k^0 = y^T s/||y||^2 \ I$ where I is the identity matrix, s is the most recent step, and y is the most recent gradient difference. Some examples of formulae for the diagonal matrix H_k^0 are given in (1) and (2) . The reciprocals of the diagonal elements of the Hessian matrix of F are sometimes very good choices for H_k^0. Note, however, that these elements must be positive (and not too small) to be useful. The steplength is determined at each iteration by a line search subroutine (supplied by J.More) that enforces a sufficient decrease condition and a curvature condition.

References

(1) Liu, D.C. and Nocedal, J. 1988. On The Limited Memory BFGS Method For Large Scale Optimization. Technical Report NA-03, Department of Electrical Engineering and Computer Science Northwestern University.

(2) Gilbert, J.C. and Lemarechal C. 1988. Some Numerical Experiments with Variable Storage Quasi-Newton

Algorithms, INRIA, France.

(3) Nocedal, J. 1980. Updating quasi-Newton matrices with limited storage, Mathematics of Computation, Vol.24, No.151, 773-782.

5 EXAMPLE OF USE

We now consider the test problem,

$$\text{minimize} \sum_{j=1}^{n/2} \left(100\left(x_{2j} - x_{2j-1}^2\right)^2 + \left(1 - x_{2j-1}\right)^2\right)$$

where n is even. The solution to this problem is at the point (1,...,1) and the optimal function value is 0. We do not wish to provide the diagonal matrices H_k^0, and therefore set DIAGCO to FALSE. As mentioned in section 2.3, VA35C/CD is declared EXTERNAL.

```
      PROGRAM LBFGS
      INTEGER NDIM,MSAVE,NWORK
      PARAMETER(NDIM=2000,MSAVE=7,NWORK=NDIM*(2*MSAVE +1) +2*MSAVE)
      DOUBLE PRECISION X(NDIM),G(NDIM),DIAG(NDIM),W(NWORK)
      DOUBLE PRECISION F,EPS,T1,T2,CNTL(5),RKEEP(40)
      INTEGER IFLAG,ICALL,N,M,J,ICNTL(10),INFO(5),KEEP(20)
      LOGICAL DIAGCO
C
      CALL VA35ID(ICNTL,CNTL,KEEP,RKEEP)
      N=100
      M=5
      DIAGCO= .FALSE.
      EPS= 1.0D-5
      ICALL=0
      IFLAG=0
      DO 10 J=1,N,2
         X(J)=-1.2D0
         X(J+1)=1.D0
 10   CONTINUE
C
 20   CONTINUE
      F= 0.D0
      DO 30 J=1,N,2
        T1= 1.D0-X(J)
        T2= 1.D1*(X(J+1)-X(J)**2)
        G(J+1)= 2.D1*T2
        G(J)= -2.D0*(X(J)*G(J+1)+T1)
        F= F+T1**2+T2**2
 30   CONTINUE
      CALL VA35AD(N,M,X,F,G,DIAGCO,DIAG,EPS,W,IFLAG,ICNTL,CNTL,
     +           INFO,KEEP,RKEEP)
      IF(IFLAG.LE.0) GO TO 50
      ICALL=ICALL + 1
      IF(ICALL.GT.2000) GO TO 50
      GO TO 20
  50  CONTINUE
      IF(IFLAG.EQ.0) THEN
        WRITE(6, '( A, F5.2 )' ) ' Optimal function value = ', F
      ELSE
        WRITE(6, '( A, I4 )' ) ' Error return, IFLAG = ', IFLAG
      END IF
      END
```

The output is as follows:

```
   Optimal function value =  0.00
```