1 SUMMARY

This subroutine **solves a system of non-linear equations**

$$f_i(x_1, x_2, ..., x_n) = 0, \quad i=1,2,...,n$$

given an initial approximation. It is especially designed for the case where the Jacobian matrix **J** = $(\partial f_i/\partial x_j)$ is sparse. It uses a modified version of the 'dog-leg' algorithm of Powell (in 'Numerical methods for non-linear algebraic equations', ed. P.Rabinowitz, Gordon and Breach, 1970, 87-161). The problem should be reasonably well-scaled, that is the magnitudes of the components x_i of the solution should not vary widely, nor should the magnitudes of the functions f_i.

ATTRIBUTES — Version: 1.1.0. **Types:** Real (single, double). **Calls:** `FD15`. **Original date:** September 2001. **Remark:** `NS12` is a threadsafe version of `NS02`. **Origin:** N. Munksgaard (CE-DATA, Denmark) and J. K. Reid, Harwell.

2 HOW TO USE THE PACKAGE

2.1 Method of Use

'Reverse communication' is employed to permit the user to have full control over the storage format for the Jacobian matrix $\mathbf{J} = (\partial f_i / \partial x_j)$ and the way that it is calculated and manipulated. To use `NS12` he or she must call it repeatedly under the control of the parameter `IFLAG`. The initial call must be with `IFLAG` set to unity, the initial approximate solution placed in the array `X` and the corresponding function values placed in the array `F`. On a return with `IFLAG` > 0 code for evaluating **f** or performing some tasks with **J** (details in §2.3) must be executed and the subroutine recalled. A return with `IFLAG=0` indicates successful termination and error conditions are marked by `IFLAG` < 0.

The user must take full responsibility for the calculation of

$$\mathbf{J} = (\partial f_i / \partial x_j)$$

or an approximation to it. This is discussed in §2.6.

In this document we use the norm

$$\|\mathbf{x}\|_2 = \left(\sum_i x_i^2 \right)^{1/2}.$$

2.2 Initialization

The `NS12I/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 `NS12A/AD` entry and before any controls are set by the user, see Section 2.4.

The single precision version

```
CALL NS12I(ICNTL,CNTL,RKEEP)
```

The double precision version

```
CALL NS12ID(ICNTL,CNTL,RKEEP)
```

`ICNTL` is an `INTEGER` array of length 5 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.

RKEEP is a REAL (DOUBLE PRECISION in the D version) array of length 20 that need not be set by the user.

2.3 The Argument List and Calling Sequence

The single precision version

```
CALL NS12A(N,X,F,DMAX,ACC,DG,DV,G,V,DNORM,RNORM,IFLAG,ICNTL,CNTL,INFO,RKEEP)
```

The double precision version

```
CALL NS12AD(N,X,F,DMAX,ACC,DG,DV,G,V,DNORM,RNORM,IFLAG,ICNTL,CNTL,INFO,RKEEP)
```

N is an INTEGER variable which must be set by the user to the number n of unknowns. It is not altered by the subroutine. **Restriction:** $n > 0$.

X is a REAL (DOUBLE PRECISION in the D version) array of length n which must be set by the user to the initial approximation to the solution. On exit it contains the current approximation to the solution. If IFLAG $\neq 4$ it is the best solution so far found.

F is a REAL (DOUBLE PRECISION in the D version) array of length n and before initial entry and before re-entry with IFLAG = 4 it must be set by the user to contain the values of $f_i(\mathbf{x})$ at the point $\mathbf{x}$ specified in X. On exit with IFLAG $\neq 4$, F(I), I=1,N will contain the values of $f_i(\mathbf{x})$ at the point $\mathbf{x}$ given in X.

DMAX is a REAL (DOUBLE PRECISION in the D version) variable which must be set by the user to a generous overestimate for the distance $\|\mathbf{x}_0 - \mathbf{x}\|_2$ between the solution $\mathbf{x}$ and the initial approximation $\mathbf{x}_0$.

ACC is a REAL (DOUBLE PRECISION in the D version) variable which must be set by the user to the accuracy required. A normal return is made (IFLAG=0) when $\|\mathbf{f}\|_2 \leq$ ACC.

DG,DV,G,V are REAL (DOUBLE PRECISION in the D version) arrays of length n which must be set by the user on an IFLAG = 2 or 3 entry (see IFLAG specification). They need not be set on initial entry and should not be altered before an IFLAG=4 entry. The arguments DG, DV and G are altered by the subroutine, but V is not.

DNORM is a REAL (DOUBLE PRECISION in the D version) variable which need not be set by the user. It is used as a limit on the norm of the next change to X. On return with IFLAG = 2 or 3 it suggests a limit for $\|\mathbf{v}\|_2$ ($\mathbf{v}$ is explained in the description of IFLAG) and must not be altered between calls.

RNORM is a REAL (DOUBLE PRECISION in the D version) variable which need not be set by the user. On return with IFLAG = 2 or 3 it indicates a limit for $\|\mathbf{J}\mathbf{v}+\mathbf{f}\|_2$ ($\mathbf{v}$ is explained in the description of IFLAG) and must not be altered between calls.

IFLAG is an INTEGER variable which must be set by the user to the value 1 on the first entry to the subroutine. On return a negative value indicates an error (details in §2.5), the value 0 indicates that the requested accuracy has been achieved and positive values indicate that NS12 should be recalled after performing the following calculations without changing IFLAG or any other argument not explicitly mentioned.

IFLAG=2

Let $\mathbf{J}$ be the Jacobian matrix $(\partial f_i / \partial x_j)$ at the point (X(1),...,X(N)) or an approximation to the Jacobian matrix (see §2.6) and let $\mathbf{f}$=(F(1),...(F(N)).

(a) Place in array G the vector
$$\mathbf{g} = -\mathbf{J}^T\mathbf{f}.$$

(b) Place in array DG the vector $\mathbf{J}\mathbf{g}$.

(c) Place in array V an approximate solution to the equation
$$\mathbf{J}\mathbf{v} = -\mathbf{f}.$$
It must satisfy the error condition
$$\|\mathbf{f}+\mathbf{J}\mathbf{v}\|_2 \leq \text{RNORM}$$
and should preferably also satisfy the condition

$\|\mathbf{v}\|_2 \leq$ DNORM
(see further notes in §2.6).

(d) Place in array DV the vector **Jv**.

IFLAG=3

A return with IFLAG=3 may be treated by the user in exactly the same way as an IFLAG=2 return. However he or she may be able to save the work of step (c) because **x**, **f** and **v** are unchanged since the last IFLAG=2 entry. There will have meanwhile been an IFLAG=4 return. If an approximate **J** was revised during this return then all four vectors **v**, **g**, **Jv**, **Jg** must be recalculated, but otherwise only **g**, **Jv** and **Jg** need be reset.

IFLAG=4

Place in array F the vector **f** with components f_i(**x**), **x**=(X(1),...,X(N)). The previous vectors **x**, **f** are stored in DV and G, respectively, which may be useful if an approximate **J** is in use (see §2.6).

ICNTL is an INTEGER array of length 5 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.

RKEEP is the REAL (DOUBLE PRECISION in the D version) private workspace array of length 20 which must not be altered by the user while IFLAG > 1.

2.4 The control and information arrays

The arrays ICNTL and CNTL provide the user with some control over the operations of NS12A/AD. The controls are given default values by NS12I/ID and may then be optionally reset by the user. The integer value controls are:

ICNTL(1) (default = 6) is used as the unit number for the printing of error messages. This printing may be suppressed by setting ICNTL(1) to a non-positive value.

ICNTL(2) (default = 6) is used as the unit number for the printing of a few lines of monitoring information at each iteration. This printing may be suppressed by setting ICNTL(2) to a non-positive value.

ICNTL(3) to ICNTL(5) are not used at present and should not be altered.

The real value controls are:

CNTL(1) has a default value of $100\times\varepsilon$, where ε is the relative precision, If the routine decides that the non-linearities are so severe that stepsizes must be limited to EPS$\times\|\mathbf{x}\|_2$ then an error return is made. An incorrect calculation of **J** may also cause such an error return.

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 returned set to the number of function evaluations, i.e. the number of times when NS12A/AD is entered with IFLAG=4.

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

2.5 Error Messages

There are four error returns, indicated by the following values of IFLAG (and INFO(1)):

–1 The step-sizes have become very small. This may be because too much accuracy has been requested or be because of errors in calculating the vectors **v**, **g**, **Jv** and **Jg**.

–2 N or IFLAG out of range.

–3 X is near a nonzero stationary point of $\|\mathbf{f}\|_2$. This diagnostic may also occur if DMAX is set too small.

–4 The vectors set by the user when IFLAG = 2 or 3 do not satisfy the condition

$$\|\mathbf{f}+\mathbf{J}\mathbf{v}\|_2 \le \texttt{RNORM},$$

as judged by the size of $\mathbf{f}^T\mathbf{f}-2\mathbf{g}^T\mathbf{v}+(\mathbf{J}\mathbf{v})^T\mathbf{J}\mathbf{v}$.

2.6 Calculations Associated with the Jacobian

Often it will be economical and convenient to calculate

$$\mathbf{J}=(\partial f_i/\partial x_j)$$

analytically and solve the equation $\mathbf{J}\mathbf{v}=-\mathbf{f}$ directly. In such cases the user's code may check for **J** being singular or near enough to singularity for the inequality

$$\|\mathbf{v}\|_2 \ge \texttt{DNORM}$$

to hold and in this case solve instead

$$(\mathbf{J}+\mathbf{E})\mathbf{v}=-\mathbf{f}$$

where **E** is chosen so that $\|\mathbf{E}\mathbf{v}\|_2 \le \texttt{RNORM}$ and preferably so that $\|\mathbf{v}\|_2 \le \texttt{DNORM}$. This may sometimes be achieved by changing small pivots, or all diagonal elements, by quantities of size about RNORM/DNORM.

If an iterative method is used to solve $\mathbf{J}\mathbf{v}=-\mathbf{f}$ then it may be terminated when $\|\mathbf{J}\mathbf{v}+\mathbf{f}\|_2 \le \texttt{RNORM}$. If $\|\mathbf{v}\|_2$ is noticeably larger than DNORM then, as for direct solution, a perturbed **J** should be used.

If a finite-difference approximation is in use, then it may be updated, say by the method of Schubert (Math. Comp. **24** (1970), 27-30) on each return with IFLAG=4. Here the previous **x** and corresponding **f** are held in arrays DV and G, respectively. If this gives slow convergence or the error return with IFLAG=–1, we recommend a fresh start with a new finite-difference approximation.

3 GENERAL INFORMATION

Use of common: none.

Other routines called directly: FD15A/AD.

Input/output: diagnostic messages on unit ICNTL(2) (see §2.4) and error messages on unit ICNTL(1) (see §2.4).

Restrictions: $n>0$.

4 METHOD

The algorithm is based on the use of a spherical 'trust region' centred at the current point within which the linear function

$$\mathbf{L}(\delta)=\mathbf{f}(\mathbf{x})+\mathbf{J}(\mathbf{x})\delta$$

gives an adequate approximation to $\mathbf{f}(\mathbf{x}+\delta)$. The scalar μ that minimizes $\|\mathbf{L}(\mu\mathbf{g})\|_2$ and the scalars α and β that minimize $\|\mathbf{L}(\alpha\mathbf{g}+\beta\mathbf{v})\|_2$ are found and the 'dog-leg' trajectory joining **x** to $\mathbf{x}+\mu\mathbf{g}$ by a straight line and $\mathbf{x}+\mu\mathbf{g}$ to $\mathbf{x}+\alpha\mathbf{g}+\beta\mathbf{v}$ by a straight line is considered. The new iterate is taken as the point $\mathbf{x}+\alpha\mathbf{g}+\beta\mathbf{v}$ if this lies inside the

spherical trust region or otherwise as the point where the dogleg crosses the surface of this sphere. The new point $\mathbf{x}+\delta$ is accepted only if

$$\|\mathbf{f}(\mathbf{x}+\delta)\|_2 \leq \|\mathbf{f}(\mathbf{x})\|_2.$$

The size of the trust region (whose radius is held in DNORM) is adjusted according to how well $\|\mathbf{f}\|_2$ is reduced compared with expectations based on $\mathbf{L}(\delta)$.

If $\mathbf{v}$ is the Newton correction $-\mathbf{J}^{-1}\mathbf{f}$ then the best point $\alpha\mathbf{g}+\beta\mathbf{v}$ will be $\mathbf{v}$ itself, but if this is well outside the trust region it may not be very useful, and it is better to solve $\mathbf{J}\mathbf{v}+\mathbf{f}=\mathbf{0}$ only approximately but with a vector $\mathbf{v}$ that is inside or not far outside the trust region.

5 EXAMPLE OF USE

We consider the solution of the following set of non-linear equations

$$f_i(\mathbf{x}) = x_i + 0.5x_{i+1}^2 - 1.5 = 0, \quad i=1,2,...,n-1, \qquad f_n(\mathbf{x}) = x_n - 1.5 = 0.$$

Using the initial solution $\mathbf{x}=\mathbf{0}$, the following code suffices for the case $n=10$. Since the Jacobian for this set of equations is upper triangular the solution of $\mathbf{J}\mathbf{v}=-\mathbf{f}$ is found by back-substitution.

```
      DOUBLE PRECISION X(20), F(20), G(20), V(20), DG(20),
     * DV(20), ACC, DNORM, DMAX, RNORM
      DOUBLE PRECISION CNTL(5), RKEEP(20)
      INTEGER ICNTL(5), INFO(5)
      INTEGER N,I,IFLAG,ICALL,NMI
      DATA N, DMAX, ACC /10,10.0D0,1D-5/
      CALL NS12ID(ICNTL,CNTL,RKEEP)
      WRITE (6,99999) N, DMAX, ACC
C SUPPRESS MONITOR PRINTING
      ICNTL(2) = 0
C  SET INITIAL SOLUTION
      DO 10 I=1,N
        X(I) = 0.0D0
   10 CONTINUE
      IFLAG = 1
      DO 80 ICALL=1,20
        GO TO (20, 40, 40, 20), IFLAG
C       EVALUATE FUNCTION.
   20   DO 30 I=1,N-1
          F(I) = X(I) + 0.5D0*X(I+1)**2 - 1.5D0
   30   CONTINUE
        F(N) = X(N) - 1.5D0
        GO TO 70
C  COMPUTE V,G,DV AND DG.
   40   G(1) = -F(1)
        V(N) = -F(N)
        DO 50 I=2,N
          G(I) = -X(I)*F(I-1) - F(I)
          NMI = N - I + 1
          V(NMI) = -F(NMI) - X(NMI+1)*V(NMI+1)
   50   CONTINUE
        DV(N) = V(N)
        DG(N) = G(N)
        DO 60 I=1,N-1
          DV(I) = V(I) + X(I+1)*V(I+1)
          DG(I) = G(I) + X(I+1)*G(I+1)
   60   CONTINUE
C  ACTIVATE SUBROUTINE
   70   CALL NS12AD(N, X, F, DMAX, ACC, DG, DV, G, V, DNORM,
```

```
      *             RNORM,IFLAG,ICNTL,CNTL,INFO,RKEEP)
       IF (IFLAG.LE.0) GO TO 90
    80 CONTINUE
    90 WRITE (6,99998) IFLAG, (X(I),I=1,N)
       STOP
99999 FORMAT (' N = ', I3,' DMAX = ', F10.2,'   ACC =', D13.5)
99998 FORMAT (' IFLAG=', I2//' SOLUTION'/(5F9.5))
          END
```

This produces the following output

```
N =  10 DMAX =      10.00   ACC =  0.10000D-04
IFLAG= 0

SOLUTION
 0.61544  1.33008  0.58295  1.35429  0.53984
 1.38576  0.47800  1.42969  0.37500  1.50000
```