## 1 SUMMARY

Given functions $f_i(x_1, x_2,..., x_n)$, $i$=1,2,...,$m$, this subroutine **evaluates an approximation to the Jacobian matrix** $\mathbf{J}=\{\partial f_i/\partial x_j\}$ using finite differences. The routine is intended for the case when **J** is sparse or band structured and has additional entries which given the functions $f_i(\mathbf{x})$ construct the sparsity pattern for **J**.

Derivatives are estimated by

$$\partial f_i/\partial x_j \approx \frac{1}{2h_j}\{f_i(x_1,...,x_j+h_j,...,x_n - f_i(x_1,...,x_j-h_j,...,x_n)\}$$

where the steplengths $h_j$ are automatically chosen by the routine within bounds specified by the user.

The method is described by Curtis, Powell, and Reid, J. Inst. Maths. Applics. **13** (1974) 117-120, and Curtis and Reid, J. Inst. Maths. Applics. **13** (1974) 121-126.

**ATTRIBUTES — Version:** 1.0.0. **Types:** `TD12A`, `TD12AD`. **Calls:** `FD05`. **Remark:** This is a rewritten version of `TD02` and supersedes it. **Original date:** August 1990. **Origin:** J. K. Reid, Rutherford Appleton Laboratory.

## 2 HOW TO USE THE PACKAGE

### 2.1 Entries and argument lists

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.

'Reverse communication' is employed to permit the user to access any data needed to calculate the functions $f_i$. To use `TD12` 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 point at which evaluation is required placed in the array `X`, and the corresponding function values placed in the array `F`. On a return with `IFLAG` > `0`, code for placing in `F` the values of the functions at the point then held in `X` must be executed and the subroutine recalled. All other arguments and the data in the subroutine's common block must not be altered. A return with `IFLAG=0` indicates successful termination and the error condition is marked by `IFLAG` = `-1`.

There are three entries:

(a) `TD12A/AD` calculates the Jacobian matrix when the sparsity pattern is known.

(b) `TD12B/BD` constructs the sparsity pattern by returning to the user for $n$ different function values.

(c) `TD12C/CD` constructs the sparsity pattern of a band matrix.

#### 2.1.1 To calculate the approximate Jacobian

*The single precision version*

```
CALL TD12A(M,N,IRN,IP,H,X,F,HMAX,A,IG,W,IFLAG)
```

*The double precision version*

```
CALL TD12AD(M,N,IRN,IP,H,X,F,HMAX,A,IG,W,IFLAG)
```

`M` is an `INTEGER` variable set by the user to the number of functions $f_i(\mathbf{x})$, that is the number of rows in the Jacobian matrix. It is not altered by the subroutine.

`N` is an `INTEGER` variable set by the user to the number of variables $x_j$, that is the number of columns in the Jacobian matrix. It is not altered by the subroutine.

IRN is an INTEGER array of size the number of nonzeros in the Jacobian matrix. It must be set by the user to hold the row indices of these nonzeros, stored by columns, for example,

$$\frac{\partial f_1}{\partial x_1}, \frac{\partial f_3}{\partial x_1}, \ldots, \frac{\partial f_m}{\partial x_1}, \frac{\partial f_2}{\partial x_2}, \ldots$$

IRN is not altered by the subroutine.

IP is an INTEGER array of size N+1 that must be set by the user so that IP(J) is the position in IRN of the start of column J of the Jacobian matrix, J=1,2,...,N, and IP(N+1) is the position of the first unused location in IRN. It is not altered by the subroutine.

H is a REAL (DOUBLE PRECISION in the D version) array of size N required by TD12A/AD to specify the step lengths to be used when estimating the derivatives by finite differences (see §4). These step lengths are adjusted by TD12A/AD to give good accuracy and are left set on the assumption that TD12A/AD is likely to be called later with similar values for the variables X(J), J=l,2,...,N, in which case no resetting is necessary.

X is a REAL (DOUBLE PRECISION in the D version) array of size N that must be set initially by the user to the point $x_j$, $j$=1,2,...,$n$, at which the approximate Jacobian is required. It is altered by the subroutine between intermediate calls and is finally restored to its initial value.

F is a REAL (DOUBLE PRECISION in the D version) array of size M whose elements must be set by the user initially and before every intermediate call to $f_i(x_1, x_2, \ldots, x_n)$, $i$=1,2,...,$m$. It is not altered by the subroutine, except on the final return when it is restored to its initial value.

HMAX is a REAL (DOUBLE PRECISION in the D version) array of size N set by the user to upper bounds for $h_j$, $j$=1,2,...,$n$, but if HMAX(1)<0 then all bounds are taken as |HMAX(1)| and the size may be one. Lower bounds for $h_j$ are taken as EPS1 times the corresponding upper bound, where EPS1 is the common variable specifying the relative machine precision (see §2.3).

A is a REAL (DOUBLE PRECISION in the D version) array of size the number of nonzero derivatives into which the nonzero derivatives are placed.

IG is an INTEGER work array of size at least 2*N+1. IG(1) should be set to zero before a first entry to TD12A/AD for a particular sparsity structure; during this entry IG(1),...,IG(2*N+1) are found and are required for subsequent entries to TD12A/AD for this sparsity structure.

W is a REAL (DOUBLE PRECISION in the D version) array of size M+2*N used as workspace.

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, the value 0 indicates that the calculation is complete and positive values indicate that the subroutine should be recalled after calculating $f_i(x_1, x_2, \ldots, x_n)$, $i$=1,2,...,$m$, and placing them in F without changing other arguments.

### 2.1.2 To construct the sparsity pattern using function values

*The single precision version*

```
CALL TD12B(M,N,IRN,IP,H,X,F,W,IA,IFLAG)
```

*The double precision version*

```
CALL TD12BD(M,N,IRN,IP,H,X,F,W,IA,IFLAG)
```

M is an INTEGER variable set by the user to the number of functions $f_i(\mathbf{x})$, that is the number of rows in the Jacobian matrix. It is not altered by the subroutine.

N is an INTEGER variable set by the user to the number of variables $x_j$, that is the number of columns in the Jacobian matrix. It is not altered by the subroutine.

IRN,IP are INTEGER arrays as described in section 2.1.1. They need not be set on entry and on final return hold the

sparsity structure of the Jacobian matrix.

H is a REAL (DOUBLE PRECISION in the D version) array of size N required by TD12B/BD to specify the step lengths to be used when altering the components of **x** to find the dependencies. It is not altered by the subroutine.

X is a REAL (DOUBLE PRECISION in the D version) array of size N which must be set initially by the user to the point $x_j$, $j$=1,2,...,$n$, at which the approximate Jacobian is required. It is altered by the subroutine between intermediate calls and is finally restored to its initial value.

F is a REAL (DOUBLE PRECISION in the D version) array of size M whose elements must be set by the user initially and before every intermediate call to $f_i(x_1, x_2, ..., x_n)$, $i$=1,2,...,$m$. It is not altered by the subroutine.

W is a REAL (DOUBLE PRECISION in the D version) array of size M used as workspace.

IA is an INTEGER variable specifying the size of the arrays A and IRN. It is not altered by the subroutine.

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, the value –1 indicates the error of IA being too small, the value 0 indicates that the calculation is complete, and positive values indicate that the subroutine should be recalled after calculating $f_i(x_1, x_2, ..., x_n)$, $i$=1,2,...,$m$, and placing them in F without changing other arguments.

### 2.1.3 To construct the sparsity pattern of a band matrix

*The single precision version*

```
CALL TD12C(N,IRN,IP,IA,MBD,IFLAG)
```

*The double precision version*

```
CALL TD12CD(N,IRN,IP,IA,MBD,IFLAG)
```

N is an INTEGER variable set by the user to the number of variables $x_j$, that is the number of columns in the Jacobian matrix. It is not altered by the subroutine.

IRN,IP are INTEGER arrays as described in section 2.1.1. They need not be set on entry and on final return hold the sparsity structure of the Jacobian matrix.

IA is an INTEGER variable specifying the size of the arrays A and IRN. It is not altered by the subroutine.

MBD is an INTEGER variable which must be set by the user to the semi-bandwidth of the Jacobian, so that $\partial f_i/\partial x_j = 0$ if $|i-j| \geq$ MBD. It is not altered by the subroutine.

IFLAG is an INTEGER variable which need not be set by the user. On return, the value –1 indicates the error of IA being too small and the value 0 indicates that the calculation is complete.

### 2.2 Error return

If IA is found to be too small by TD12B/BD or TD12C/CD then a message is output on stream 6 (unless the common variable LP is changed, see §2.3) and IFLAG is set to –1. There are no other error returns.

### 2.3 Use of Comnon

The subroutines contain a common block called TD12D/DD, whose data must be preserved between entries. It has the form:

*The single precision version*

```
COMMON/TD12D/ UMIN,UAIM,UMAX,EPS,EPS1,LP,ADJUST,NGROUP
```

*The double precision version*

```
COMMON/TD12DD/ UMIN,UAIM,UMAX,EPS,EPS1,LP,ADJUST,NGROUP
```

The default values are set by the block data subprogram TD12E/ED.

UMIN is REAL (DOUBLE PRECISION in the D version) variable with default value 10. Further information is given in §4.

UAIM is REAL (DOUBLE PRECISION in the D version) variable with default value 100. Further information is given in §4.

UMAX is REAL (DOUBLE PRECISION in the D version) variable with default value 1000. Further information is given in §4.

EPS is REAL (DOUBLE PRECISION in the D version) variable that is set to the machine precision on an initial entry to TD12A/AD. Further information is given in §4.

EPS1 is REAL (DOUBLE PRECISION in the D version) variable that is set to the machine precision on an initial entry to TD12A/AD. Further information is given in §4.

LP is an INTEGER variable with default value 6. It specifies the Fortran unit number for messages.

ADJUST is a LOGICAL variable with default value .TRUE. . This allows the user, by setting ADJUST to .FALSE., to economise in the number of function calls. In that case, no estimation of errors is made, no adjustments are made to step sizes, and the less accurate one-sided difference approximation

$$\frac{\partial f_i}{\partial x_j} \approx \frac{f_i(x_1, x_2, ..., x_j + h_j, ..., x_n) - f_i(x_1, x_2, ..., x_n)}{h_j}$$

is used in place of the one given in §4.

NGROUP is an INTEGER variable that is set by TD12A/AD to the number of groups used (see §4).

TD12A/AD also contains a common block called TD12F/FD, whose data is really private to the subroutine. The data must be preserved between entries. It has the form:

*The single precision version*

```
COMMON/TD12F/ ADATA(6), IDATA(11), LDATA
```

*The double precision version*

```
COMMON/TD12FD/ ADATA(6), IDATA(11), LDATA
```

ADATA is a REAL (DOUBLE PRECISION in the D version) array.

IDATA is an INTEGER array.

LDATA is a LOGICAL variable.

TD12B/BD also contains a common block called TD12G/GD, whose data is really private to the subroutine. The data must be preserved between entries. It has the form:

*The single precision version*

```
COMMON/TD12G/ BDATA, JDATA(3)
```

*The double precision version*

```
COMMON/TD12GD/ BDATA, JDATA(3)
```

BDATA is a REAL (DOUBLE PRECISION in the D version) variable.

JDATA is an INTEGER array.

## 3 GENERAL INFORMATION

**Use of common:** uses common blocks called TD12D/DD, TD12F/FD, and TD12G/GD, see §2.3.

**Workspace:** passed as arguments, see IG and W.

**Other routines called directly:** there is a block data subprogram TD12E/ED and the function FD05A/FD05AD is called by TD12A/TD12AD.

**Input/output:** error messages, see LP in §2.3.

## 4 METHOD

On entry to TD12C/CD, the appropriate sparsity pattern is set up in a straightforward way and IA is checked.

On entry to TD12B/BD, the variable X(1) is changed to X(1)+H(1) and the corresponding changes to **f** are taken to indicate the nonzeros in the first column of the Jacobian matrix. X(1) is then restored to its original value and the same procedure is applied to each column. Note that it is important to avoid any special values (for example, zeros) of X(J), J=1,N, which cause freak zeros in the Jacobian matrix and it is important that the steps H(J) be large enough so that all the appropriate changes to **f** are noticeable.

On entry to TD12A/AD, unless ADJUST has the value .FALSE. (see §2.3), the derivatives are estimated by the difference formula.

$$\frac{\partial f_i}{\partial x_j} \approx \frac{f_i(x_1, x_2, ..., x_j + h_j, ..., x_n) - f_i(x_1, x_2, ..., x_j - h_j, ..., x_n)}{2h_j}.$$

The truncation error in this approximation is estimated using the second difference and the roundoff error is estimated from the effect on $f_i(\mathbf{x})$ of roundoff-level changes to $x_j$. Each $h_j$, is adjusted so that, if possible, the greatest ratio of roundoff error to truncation error for a column of the Jacobian is near UAIM and within the range (UMIN,UMAX). EPS is the greatest relative roundoff in a single operation and is needed for the roundoff error estimate. The steps $h_j$ are not permitted to leave the range

$$[\max(\texttt{EPS}*|x_j|, \texttt{EPS1}*\texttt{HMAX(J)}), \texttt{HMAX(J)}].$$

To economise on the number of function evaluations, the columns of the Jacobian are formed into groups such that no two columns of the same group have a nonzero in the same row so that all the changes $h_j$ corresponding to one group may be made together. The array IG is used to store store this grouping. For further details of TD12, see the three references given below.

### References

Curtis, A.R., Powell, M.J.D. and Reid, J.K. (1974). On the estimation of sparse Jacobian matrices. J. Inst. Maths. Applics. **13** (1974) 117-120.

Reid, J.K. (1972). Fortran subroutines for the solution of sparse systems of nonlinear equations. Harwell report R.7293.

Curtis, A.R. and Reid, J.K. (1972). The choice of step lengths when using differences to approximate Jacobian matrices. J. Inst. Maths. Applics. **13** (1974) 121-126.

## 5 EXAMPLE OF USE

As a simple example, suppose that we wish to estimate the Jacobian matrix of the functions

$$
\begin{aligned}
f_1(\mathbf{x}) &= x_1 x_2 \\
f_2(\mathbf{x}) &= x_1 + x_3^2 \\
f_3(\mathbf{x}) &= x_4 x_5 + x_6 \\
f_4(\mathbf{x}) &= x_3 - x_4 / x_5 \\
f_5(\mathbf{x}) &= 1 - 2x_6
\end{aligned}
$$

at the point $\mathbf{x} = (1, 2, \ldots, 6)^T$. The Jacobian has the sparsity pattern

$$
\begin{matrix}
\times & \times & 0 & 0 & 0 & 0 \\
\times & 0 & \times & 0 & 0 & 0 \\
0 & 0 & 0 & \times & \times & \times \\
0 & 0 & \times & \times & \times & 0 \\
0 & 0 & 0 & 0 & 0 & \times
\end{matrix} .
$$

We may then obtain the required Jacobian values using the following piece of code.

```
C     TEST OF TD12
      INTEGER M, N, IA, LIP, LW, LIG
      PARAMETER ( M = 5, N = 6, IA = 30 )
      PARAMETER ( LIP = N + 1, LW = M + 2*N, LIG = 2*N + 1 )
      INTEGER   I, IRN( IA ), IP( LIP ), IG( LIG ), IFLAG, J, J1, J2
      DOUBLE PRECISION X( N ), F( M ), A( IA ), W( LW ),
     *                 H( N ), HMAX( N )
      EXTERNAL         FUNC
      DATA        X / 1.0D+0, 2.0D+0, 3.0D+0, 4.0D+0, 5.0D+0, 6.0D+0 /
      DATA        H / 1.0D-1, 2.0D-1, 3.0D-1, 4.0D-1, 5.0D-1, 6.0D-1 /
      HMAX( 1 ) = - 1.0D+0
      IFLAG = 1
    5 CONTINUE
         CALL FUNC( X, F )
         CALL TD12BD( M, N, IRN, IP, H, X, F, W, IA, IFLAG)
      IF( IFLAG.GT.0) GO TO 5
      WRITE( 6, 2000 )
      DO 10 I = 1, N
         J1   = IP( I )
         J2   = IP( I + 1 ) - 1
         IF ( J1 .LE. J2 ) WRITE( 6, 2010 ) I, ( IRN( J ), J = J1, J2 )
   10 CONTINUE
      IG( 1 ) = 0
   15 CONTINUE
         CALL FUNC( X, F )
         CALL TD12AD( M, N, IRN, IP, H, X, F, HMAX, A, IG, W, IFLAG )
      IF( IFLAG.GT.0) GO TO 15
      WRITE( 6, 2020 )
      DO 20 I = 1, N
         J1   = IP( I )
         J2   = IP( I + 1 ) - 1
         IF ( J1 .LE. J2 ) WRITE( 6, 2030 ) I, ( A( J ), J = J1, J2 )
   20 CONTINUE
      STOP
 2000 FORMAT( /, ' On return from TD12BD :', / )
```

```
 2010 FORMAT( ' column ', I2, ' of the Jacobian has',
     *        ' nonzero(s) in row(s)', 4I2, ( 40I2 ) )
 2020 FORMAT( /, ' On return from TD12AD :', / )
 2030 FORMAT( ' value(s) of nonzero(s) in column', I2,
     *        ' of the Jacobian :',  1P, 2D9.1, ( 1P, 8D9.1 ) )
      END
      SUBROUTINE FUNC( X, F )
      DOUBLE PRECISION X( 6 ), F( 5 )
      F( 1 ) = X( 1 ) * X( 2 )
      F( 2 ) = X( 1 ) + X( 3 ) ** 2
      F( 3 ) = X( 4 ) * X( 5 ) + X( 6 )
      F( 4 ) = X( 3 ) - X( 4 ) / X( 5 )
      F( 5 ) = 1.0D+0 - 2.0D+0 * X( 6 )
      RETURN
      END
```

This produces the following output:

```
 On return from TD12BD :

 column  1 of the Jacobian has nonzero(s) in row(s) 1 2
 column  2 of the Jacobian has nonzero(s) in row(s) 1
 column  3 of the Jacobian has nonzero(s) in row(s) 2 4
 column  4 of the Jacobian has nonzero(s) in row(s) 3 4
 column  5 of the Jacobian has nonzero(s) in row(s) 3 4
 column  6 of the Jacobian has nonzero(s) in row(s) 3 5

 On return from TD12AD :

 value(s) of nonzero(s) in column 1 of the Jacobian :  2.0D+00  1.0D+00
 value(s) of nonzero(s) in column 2 of the Jacobian :  1.0D+00
 value(s) of nonzero(s) in column 3 of the Jacobian :  6.0D+00  1.0D+00
 value(s) of nonzero(s) in column 4 of the Jacobian :  5.0D+00 -2.0D-01
 value(s) of nonzero(s) in column 5 of the Jacobian :  4.0D+00  1.6D-01
 value(s) of nonzero(s) in column 6 of the Jacobian :  1.0D+00 -2.0D+00
```