1 SUMMARY

This subroutine evaluates an **approximation of the Hessian matrix**

$$\left\{ \frac{\partial^2 f}{\partial x_i \partial x_j} \right\}$$

for a real function $f(x_1, x_2, ..., x_n)$ of n variables whose gradient

$$\left\{ \frac{\partial f}{\partial x_i} \right\}$$

can be computed. The approximate Hessian matrix is generated by a method which uses finite differences in the gradient and takes advantage of the entries in the Hessian that are known to be zero. The user must specify the step sizes to be used in the finite difference calculation and supply a routine to evaluate the gradient.

The method is based on that of Powell and Toint, “On the Estimate of Sparse Hessian Matrices”, J. Numer. Anal., **16**, 6, December 1979.

ATTRIBUTES — **Version:** 1.0.0. **Types:** TD03A, TD03AD. **Original date:** December 1980. **Origin:** P. L. Toint (Univ. of Namur), modified by I. S. Duff, Harwell.

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 Calling sequence

To generate the approximate Hessian matrix, the user must first make one call to TD03A/TD03AD to analyze the sparsity pattern of the matrix and then call TD03B/TD03BD to compute the Hessian approximation. The computation of the Hessian at subsequent points $(x_1,..., x_n)$ or for any Hessian with the same sparsity pattern is accomplished by subsequent calls to TD03B/TB03BD only.

2.2 The analyze stage

Call TD03A/TD03AD to analyze the sparsity pattern of the matrix and to generate information which will be used in the numerical evaluation of the Hessian.

The single precision version

```
CALL TD03A (N,NZ,IROW,IDIAG,IANAL,NP,IWK,IERR)
```

The double precision version

```
CALL TD03AD(N,NZ,IROW,IDIAG,IANAL,NP,IWK,IERR)
```

N is an INTEGER variable set by the user to n the dimension of the Hessian matrix, i.e. the number of variables in the function f. It is unchanged by the subroutine. **Restriction:** N > 0.

NZ is an INTEGER variable set by the user to the number of nonzero entries on and below the diagonal of the Hessian matrix. It is unchanged by the subroutine. **Restriction:** N ≤ NZ ≤ N*(N+1)/2.

IROW is an INTEGER array of length NZ used to describe the sparsity structure of the Hessian matrix. It must be set by the user before entry to TD03A/TD03AD so that IROW(I), I=1,...,NZ contain the row numbers of the successive nonzero elements of the lower triangular part (including the diagonal) of the matrix when scanned column after column in the natural order. The diagonal entry must preceed the other entries in each column. The remaining entries may appear in any order. The array is unchanged by `TD03A/TD03AD`.

`IDIAG` is an `INTEGER` array of length `N` used to describe the sparsity structure of the Hessian matrix. It must be set by the user before entry to `TD03A/TD03AD` so that `IDIAG(I)`, `I=1,...,N` contain the position of the `I`-th diagonal of the matrix in the list held in `IROW`. The array is unchanged by `TD03A/TD03AD`. **Restriction:** `IROW(IDIAG(I)) = I` for `I=1,...,N`.

`IANAL` is an `INTEGER` array of length `2*NZ+3*N+1` used to store the information about the structure that is generated by `TD03A/TD03AD` for later reuse in `TD03B/TD03BD`. It must be left unchanged between the call to `TD03A/TD03AD` and subsequent calls to `TD03B/TD03BD`.

`NP` is an `INTEGER` variable which gives, on exit from `TD03A/TD03AD`, the number of gradient evaluations that are needed for one estimation of the Hessian. This must be left unchanged between the call to `TD03A/TD03AD` and subsequent calls to `TD03B/TD03BD`.

`IWK` is an `INTEGER` array of length `N` used by `TD03A/TD03AD` as work space.

`IERR` is an `INTEGER` error flag that has the value 0 if no error was detected in the call to `TD03A/TD03AD`. A positive value (+1, or +2) indicates a fatal error (see §2.5).

2.3 The estimation stage

Call `TD03B/TD03BD` to estimate the Hessian matrix using information produced by a previous call to `TD03A/TD03AD`.

The single precision version

```
CALL TD03B (N,NZ,IROW,IDIAG,IANAL,NP,X,GX,D,GRAD,B,WORK,IERR)
```

The double precision version

```
CALL TD03BD(N,NZ,IROW,IDIAG,IANAL,NP,X,GX,D,GRAD,B,WORK,IERR)
```

Arguments `N`, `NZ`, `IROW`, `IDIAG`, `IANAL`, `NP` are the same as those of the same name in the call to `TD03A/TD03AD` and must be unchanged since that call. They are not altered by `TD03B/TD03BD`.

`X` is a `REAL` (`DOUBLE PRECISION` in the D version) array of length `N` that is set by the user to the value of the variables at which the Hessian matrix of *f* is to be approximated. This argument is unchanged by the subroutine.

`GX` is a `REAL` (`DOUBLE PRECISION` in the D version) array of length `N` that is set by the user to the value of the gradient vector of *f* at the point defined by `X`. This argument is unchanged by the subroutine.

`D` is a `REAL` (`DOUBLE PRECISION` in the D version) array of length `N` that is set by the user to the values of the stepsizes that are to be used in the finite difference scheme. One can roughly say that `D(I)` is the step used to evaluate the `I`-th column of the Hessian. (Recommended values are between 10^{-7} and 10^{-3} times the corresponding component of `X`). This argument is unchanged by the subroutine.

`GRAD` is the name of a user supplied subroutine to compute the gradient vector for *f*. The subroutine has the argument list

```
SUBROUTINE GRAD(X,G,N,NZ,IROW,IDIAG)
```

`X` is a `REAL` (`DOUBLE PRECISION` in the D version) array of length `N` which will contain the values of the variables $(x_1,...,x_n)$.

`G` is a `REAL` (`DOUBLE PRECISION` in the D version) array of length `N` in which the subroutine must return the computed values of the gradient vector.

The arguments `N`, `NZ`, `IROW` and `IDIAG` are as described in §2.2. The routine's name must be declared `EXTERNAL` in the calling program.

`B` is a `REAL` (`DOUBLE PRECISION` in the D version) array of length `NZ` that is used for storing the nonzero elements of the estimated Hessian matrix in the order defined by the list stored in IROW. It contains the desired Hessian approximation on exit from TD03B/TD03BD.

WORK is a REAL (DOUBLE PRECISION in the D version) array of length 2*N that is used by TD03B/TD03BD as workspace. It need not be preserved between calls to TD03B/TD03BD.

IERR is an INTEGER variable which will have value 0 on successful completion of TD03B/TD03BD. Positive values (+2, +3) indicate an error has been detected (see §2.5).

2.4 Use of Common

Only one common block is used by TD03A/TD03AD and TD03B/TD03BD viz.

The single precision version

```
COMMON/TD03E / LP
```

The double precision version

```
COMMON/TD03ED/ LP
```

where

LP is an INTEGER variable which can be set by the user to the Fortran unit number for output of error messages. A default value of 6 is set in the block data subprogram TD03F/TD03FD and a value of 0 suppresses the printing of error messages.

2.5 Error returns

If some error in the input data is detected by the routines they return immediately to the calling program with the error flag IERR having a positive value. Possible values are:

0 no error was detected (TD03A/TD03AD and TD03B/TD03BD entries).

1 Error in sparsity pattern. Either IROW(IDIAG(I))$\neq$I or an entry in the upper triangle has been input (TD03A/TD03AD entry only).

2 Error in parameter N or NZ. One of the restrictions (see 2.2) has been violated (TD03A/TD03AD and TD03B/TD03BD entries).

3 The auxiliary information on the structure generated by TD03A/TD03AD in IANAL and NP is not present on a subsequent call to TD03B/TD03BD. It is likely that corruption has occurred or TD03A/TD03AD was not called (TD03B/TD03BD entry only).

For each positive value an informative message is output on unit number LP (see §2.4). This can be suppressed by setting LP to 0.

3 GENERAL INFORMATION

Use of common: TD03E/TD03ED (see §2.4).

Workspace: provided by the user through the arguments, IWK (INTEGER array of length N used by TD03A/TD03AD) and WORK (REAL (DOUBLE PRECISION in the D version) array of length 2*N used by TD03B/TD03BD).

Other routines called directly: internal subroutines TD03C/TD03CD and TD03D/TD03DD and the block data subprogram TD03F/TD03FD are used by TD03A/TD03AD and TD03B/TD03BD respectively. The user must also supply a subroutine to calculate gradients of the function (see §2.3).

Input/output: error messages are output on unit LP. This has default value 6 and output is suppressed if it is set to zero.

Restrictions: $0 < \texttt{N} \le \texttt{NZ} \le \texttt{N*(N+1)/2}$, `IROW(IDIAG(I))` = `I`.

4 METHOD

The routines use the “Lower triangular substitution algorithm” described in (1). *It assumes that no diagonal values are constrained by the sparsity requirements.*

The routine `TD03A/TD03AD` analyzes the sparsity pattern of the matrix in order to decide how many differences in gradient are needed for estimation and along what directions. For this purpose, it defines a symmetric permutation of the matrix that is held, with other information in `IANAL`.

The routine `TD03B/TD03BD` computes the differences in gradient that are required and then solves a consequent linear system by a substitution to obtain the entries of the approximate Hessian. For further details about the algorithms used see (1).

Once the pattern analysis is performed, one can approximate the Hessian of f at several different points. This is done by one call to `TD03A/TD03AD` followed by several calls to `TD03B/TD03BD` for different values of `X` and `GX`.

Reference

(1) M.J.D. Powell and Ph.L. Toint. On the estimation of sparse Hessian matrices. S.I.A.M. J. Numer. Anal. **16** (1979) 1060-1074.

5 EXAMPLE OF USE

As a simple example, suppose we wish to estimate the Hessian matrix of the function

$$f(x_1, \dots, x_5) = x_1^3 + x_2^3 + x_3^3 + x_4^3 + x_5^3 + x_1 x_4 + x_2 x_3 + x_3 x_4 + x_4 x_5$$

at the two points $x = (1,..., 1)^T$ and $x = (1, 2,..., 5)^T$. The Hessian has the sparsity pattern

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

Choosing uniform step lengths of size 10^{-6}, the following piece of code can be used to obtain the required approximations in the arrays `B1` and `B2` respectively:

```
      INTEGER I, IERR, N, NZ, NP
      DOUBLE PRECISION B1( 9 ), B2( 9 ), GX( 5 ), D( 5 ),
     *     WORK( 10 ), X1( 5 ), X2( 5 )
      INTEGER IANAL( 34 ), IDIAG( 5 ), IROW( 9 ), IWK( 5 )
      EXTERNAL DER1
      DATA IROW / 1, 4, 2, 3, 3, 4, 4, 5, 5 /
      DATA IDIAG / 1, 3, 5, 7, 9 /
      DATA X1 / 1.0D+0, 1.0D+0, 1.0D+0, 1.0D+0, 1.0D+0 /
      DATA X2 / 1.0D+0, 2.0D+0, 3.0D+0, 4.0D+0, 5.0D+0 /
      DATA D  / 1.0D-6, 1.0D-6, 1.0D-6, 1.0D-6, 1.0D-6 /
      N  = 5
      NZ = 9
      CALL TD03AD( N, NZ, IROW, IDIAG, IANAL, NP, IWK, IERR )
      CALL DER1( X1, GX, N, NZ, IROW, IDIAG )
      CALL TD03BD( N, NZ, IROW, IDIAG, IANAL,
     *            NP, X1, GX, D, DER1, B1, WORK, IERR )
      WRITE( 6, 2000 ) ( B1( I ), I = 1, 9 )
      CALL DER1( X2, GX, N, NZ, IROW, IDIAG )
      CALL TD03BD( N, NZ, IROW, IDIAG, IANAL,
     *            NP, X2, GX, D, DER1, B2, WORK, IERR )
```

```
        WRITE( 6, 2010 ) ( B2( I ), I = 1, 9 )
        STOP
   2000 FORMAT( /, ' AT 1ST POINT, NONZEROS IN HESSIAN MATRIX ARE ', /,
       *        ( 1P, 3D12.4 ) )
   2010 FORMAT( /, ' AT 2ND POINT, NONZEROS IN HESSIAN MATRIX ARE ', /,
       *        ( 1P, 3D12.4 ) )
        END
        SUBROUTINE DER1( X, G, N, NZ, IROW, IDIAG )
        INTEGER N, NZ
        DOUBLE PRECISION G( N ), X( N )
        INTEGER IDIAG( N ), IROW( NZ )
        G( 1 ) = 3.0D+0 * X( 1 ) ** 2 + X( 4 )
        G( 2 ) = 3.0D+0 * X( 2 ) ** 2 + X( 3 )
        G( 3 ) = 3.0D+0 * X( 3 ) ** 2 + X( 2 ) + X( 4 )
        G( 4 ) = 3.0D+0 * X( 4 ) ** 2 + X( 1 ) + X( 3 ) + X( 5 )
        G( 5 ) = 3.0D+0 * X( 5 ) ** 2 + X( 4 )
        RETURN
        END
```

This produces the following output:

```
   AT 1ST POINT, NONZEROS IN HESSIAN MATRIX ARE
    6.0000D+00  1.0000D+00  6.0000D+00
    1.0000D+00  6.0000D+00  1.0000D+00
    6.0000D+00  1.0000D+00  6.0000D+00

   AT 2ND POINT, NONZEROS IN HESSIAN MATRIX ARE
    6.0000D+00  1.0000D+00  1.2000D+01
    1.0000D+00  1.8000D+01  1.0000D+00
    2.4000D+01  1.0000D+00  3.0000D+01
```