D02TGF – NAG Fortran Library Routine Document

Note. Before using this routine, please read the Users' Note for your implementation to check the interpretation of bold italicised terms and other implementation-dependent details.

1 Purpose

D02TGF solves a system of linear ordinary differential equations by least-squares fitting of a series of Chebyshev polynomials using collocation.

2 Specification

```
 SUBROUTINE D02TGF(N, M, L, X0, X1, K1, KP, C, IC, COEFF, BDYC, W,
1                  LW, IW, LIW, IFAIL)
 INTEGER           N, M(N), L(N), K1, KP, IC, LW, IW(LIW), LIW,
1                  IFAIL
 real              X0, X1, C(IC,N), W(LW)
 EXTERNAL          COEFF, BDYC
```

3 Description

The routine calculates an approximate solution of a linear or linearised system of ordinary differential equations as a Chebyshev-series. Suppose there are n differential equations for n variables $y_1, y_2, \ldots, y_n$, over the range (x_0, x_1). Let the ith equation be

$$\sum_{j=1}^{m_i+1} \sum_{k=1}^{n} f_{kj}^i(x) y_k^{(j-1)}(x) = r^i(x)$$

where $y_k^{(j)}(x) = \frac{d^j y_k(x)}{dx^j}$. The routine COEFF provided by the user evaluates the coefficients $f_{kj}^i(x)$ and the right-hand side $r^i(x)$ for each i, $1 \leq i \leq n$, at any point x. The boundary conditions may be applied either at the end-points or at intermediate points; they are written in the same form as the differential equations, and specified by the routine BDYC. For example the jth boundary condition out of those associated with the ith differential equation takes the form

$$\sum_{j=1}^{l_i+1} \sum_{k=1}^{n} f_{kj}^{ij}\left(x^{ij}\right) y_k^{(j-1)}\left(x^{ij}\right) = r^{ij}\left(x^{ij}\right),$$

where x^{ij} lies between x_0 and x_1. It is assumed in this routine that certain of the boundary conditions are associated with each differential equation. This is for the user's convenience; the grouping does not affect the results.

The degree of the polynomial solution must be the same for all variables. The user specifies the degree required, $k_1 - 1$, and the number of collocation points, k_p, in the range. The routine sets up a system of linear equations for the Chebyshev coefficients, with n equations for each collocation point and one for each boundary condition. The collocation points are chosen at the extrema of a shifted Chebyshev polynomial of degree $k_p - 1$. The boundary conditions are satisfied exactly, and the remaining equations are solved by a least-squares method. The result produced is a set of Chebyshev coefficients for the n functions $y_1, y_2, \ldots, y_n$, with the range normalised to $[-1, 1]$.

E02AKF can be used to evaluate the components of the solution at any point on the range $[x_0, x_1]$ (see Section 9 for an example). E02AHF and E02AJF may be used to obtain Chebyshev-series representations of derivatives and integrals (respectively) of the components of the solution.

4 References

[1] Picken S M (1970) Algorithms for the solution of differential equations in Chebyshev-series by the selected points method *Report Math. 94* National Physical Laboratory

5 Parameters

1: N — INTEGER *Input*

On entry: the number of differential equations in the system, n.

Constraint: $N \geq 1$.

2: M(N) — INTEGER array *Input*

On entry: M(i) must be set to the highest order derivative occurring in the ith equation, for $i = 1, 2, \ldots$,N.

Constraint: $M(i) \geq 1$, for $i = 1, 2, \ldots, n$.

3: L(N) — INTEGER array *Input*

On entry: L(i) must be set to the number of boundary conditions associated with the ith equation, for $i = 1, 2, \ldots, n$.

Constraint: $L(i) \geq 0$, for $i = 1, 2, \ldots, n$.

4: X0 — ***real*** *Input*

On entry: the left-hand boundary, x_0.

5: X1 — ***real*** *Input*

On entry: the right-hand boundary, x_1.

Constraint: $X1 > X0$.

6: K1 — INTEGER *Input*

On entry: the number of coefficients, k_1, to be returned in the Chebyshev-series representation of the solution (hence, the degree of the polynomial approximation is K1−1).

Constraint: $K1 \geq 1 + \max_{1 \leq i \leq N} M(i)$.

7: KP — INTEGER *Input*

On entry: the number of collocation points to be used, k_p.

Constraint: $N \times KP \geq N \times K1 + \sum_{i=1}^{N} L(i)$.

8: C(IC,N) — ***real*** array *Output*

On exit: the kth column of C contains the computed Chebyshev coefficients of the kth component of the solution, y_k; that is, the computed solution is:

$$y_k = {\sum_{i=1}^{k_1}}' C(i,k) T_{i-1}(x), 1 \leq k \leq n,$$

where $T_i(x)$ is the Chebyshev polynomial of the first kind and Σ' denotes that the first coefficient, $C(1,k)$, is halved.

9: IC — INTEGER *Input*

On entry: the first dimension of the array C as declared in the (sub)program from which D02TGF is called.

Constraint: $IC \geq K1$.

10: COEFF — SUBROUTINE, supplied by the user. *External Procedure*

COEFF defines the system of differential equations (see Section 3). It must evaluate the coefficient functions $f_{kj}^i(x)$ and the right-hand side function $r^i(x)$ of the ith equation at a given point. Only non-zero entries of the array A and RHS need be specifically assigned, since all elements are set to zero by D02TGF before calling COEFF.

Its specification is:

```
SUBROUTINE COEFF(X, I, A, IA, IA1, RHS)
INTEGER          I, IA, IA1
real             X, A(IA,IA1), RHS
```

Important: the dimension declaration for A must contain the variable IA, not an integer constant.

1: X — ***real*** *Input*

On entry: the point x at which the functions must be evaluated.

2: I — INTEGER *Input*

On entry: the equation for which the coefficients and right-hand side are to be evaluated.

3: A(IA,IA1) — ***real*** array *Input/Output*

On entry: all elements of A are set to zero.

On exit: $\mathrm{A}(k,j)$ must contain the value $f_{kj}^i(x)$, for $1 \le k \le n$, $1 \le j \le m_i + 1$.

4: IA — INTEGER *Input*
5: IA1 — INTEGER *Input*

On entry: the first and second dimensions of A, respectively.

6: RHS — ***real*** *Input/Output*

On entry: RHS is set to zero.

On exit: it must contain the value $r^i(x)$.

COEFF must be declared as EXTERNAL in the (sub)program from which D02TGF is called. Parameters denoted as *Input* must **not** be changed by this procedure.

11: BDYC — SUBROUTINE, supplied by the user. *External Procedure*

BDYC defines the boundary conditions (see Section 3). It must evaluate the coefficient functions f_{kj}^{ij} and right-hand side function r^{ij} in the jth boundary condition associated with the ith equation, at the point x^{ij} at which the boundary condition is applied. Only non-zero entries of the array A and RHS need be specifically assigned, since all elements are set to zero by D02TGF before calling BDYC.

Its specification is:

```
SUBROUTINE BDYC(X, I, J, A, IA, IA1, RHS)
INTEGER         I, J, IA, IA1
real            X, A(IA,IA1), RHS
```

Important: the dimension declaration for A must contain the variable IA, not an integer constant.

1: X — ***real*** *Output*

On exit: the value x^{ij} at which the boundary condition is applied.

2: I — INTEGER *Input*

On entry: the differential equation with which the condition is associated.

3: J — INTEGER *Input*

On entry: the boundary condition for which the coefficients and right-hand side are to be evaluated.

4: A(IA,IA1) — ***real*** array *Input/Output*

On entry: all elements of A are set to zero.

On exit: the value $f_{kj}^{ij}\left(x^{ij}\right)$ for $1 \le k \le n$, $1 \le j \le m_i + 1$.

5: IA — INTEGER *Input*
6: IA1 — INTEGER *Input*

On entry: the first and second dimensions of A, respectively.

7: RHS — ***real*** *Input/Output*

On entry: RHS is set to zero.

On exit: the value $r^{ij}\left(x^{ij}\right)$.

BDYC must be declared as EXTERNAL in the (sub)program from which D02TGF is called. Parameters denoted as *Input* must **not** be changed by this procedure.

12: W(LW) — ***real*** array *Workspace*

13: LW — INTEGER *Input*

On entry: the dimension of the array W as declared in the (sub)program from which D02TGF is called.

Constraint: LW $\geq 2 \times (\mathrm{N}\times\mathrm{KP}+\mathrm{NL}) \times (\mathrm{N}\times\mathrm{K1}+1) + 7 \times \mathrm{N} \times \mathrm{K1}$,

where $\mathrm{NL} = \sum_{i=1}^{n} \mathrm{L}(i)$.

14: IW(LIW) — INTEGER array *Workspace*

15: LIW — INTEGER *Input*

On entry: the dimension of the array IW as declared in the (sub)program from which D02TGF is called.

Constraint: LIW $\geq \mathrm{N} \times \mathrm{K1} + 1$.

16: IFAIL — INTEGER *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6 Error Indicators and Warnings

Errors detected by the routine:

IFAIL = 1

On entry, $\mathrm{N} < 1$,
or $\mathrm{M}(i) < 1$ for some i,
or $\mathrm{L}(i) < 0$ for some i,
or $\mathrm{X0} \geq \mathrm{X1}$,
or $\mathrm{K1} < 1 + \mathrm{M}(i)$ for some i,
or $\mathrm{N} \times \mathrm{KP} < \mathrm{N} \times \mathrm{K1} + \sum_{i=1}^{n} \mathrm{L}(i)$,
or $\mathrm{IC} < \mathrm{K1}$.

IFAIL =2

On entry, LW is too small (see Section 5),
or $\mathrm{LIW} < \mathrm{N} \times \mathrm{K1}$.

IFAIL = 3

Either the boundary conditions are not linearly independent, or the rank of the matrix of equations for the coefficients is less than the number of unknowns. Increasing KP may overcome this latter problem.

IFAIL = 4

The least-squares routine F04AMF has failed to correct the first approximate solution (see NAG Fortran Library document F04AMF). Increasing KP may remove this difficulty.

7 Accuracy

Estimates of the accuracy of the solution may be obtained by using the checks described in Section 8. The Chebyshev coefficients are calculated by a stable numerical method.

8 Further Comments

The time taken by the routine depends on the complexity of the system of differential equations, the degree of the polynomial solution and the number of matching points.

If the number of matching points k_p is equal to the number of coefficients k_1 minus the average number of boundary conditions $\frac{1}{n}\sum_{i=1}^{n} l_i$, then the least-squares solution reduces to simple solution of linear equations and true collocation results. The accuracy of the solution may be checked by repeating the calculation with different values of k_1. If the Chebyshev coefficients decrease rapidly, the size of the last two or three gives an indication of the error. If they do not decrease rapidly, it may be desirable to use a different method. Note that the Chebyshev coefficients are calculated for the range normalised to $[-1,1]$.

Generally the number of boundary conditions required is equal to the sum of the orders of the n differential equations. However, in some cases fewer boundary conditions are needed, because the assumption of a polynomial solution is equivalent to one or more boundary conditions (since it excludes singular solutions).

A system of **nonlinear** differential equations must be linearised before using the routine. The calculation is repeated iteratively. On each iteration the linearised equation is used. In the example in Section 9, the y variables are to be determined at the current iteration whilst the z variables correspond to the solution determined at the previous iteration, (or the initial approximation on the first iteration). For a starting approximation, we may take, say, a linear function, and set up the appropriate Chebyshev coefficients before starting the iteration. For example, if $y_1 = ax + b$ in the range (x_0, x_1), we set B, the array of coefficients,

B(1,1) $= a \times (x_0 + x_1) + 2 \times b$,

B(1,2) $= a \times (x_1 - x_0)/2$,

and the remainder of the entries to zero.

In some cases a better initial approximation may be needed and can be obtained by using E02ADF or E02AFF to obtain a Chebyshev-series for an approximate solution. The coefficients of the current iterate must be communicated to COEFF and BDYC, e.g., in COMMON. (See the example in Section 9). The convergence of the (Newton) iteration cannot be guaranteed in general, though it is usually satisfactory from a good starting approximation.

9 Example

To solve the nonlinear system

$$y_1' + \left(y_2^2 - 1\right) y_1 + y_2 = 0,$$
$$2y_2'' - y_1' = 0,$$

in the range $(-1,1)$, with $y_1 = 0$, $y_2 = 3$, $y_2' = 0$ at $x = -1$.

Suppose an approximate solution is z_1, z_2 such that $y_1 \sim z_1$, $y_2 \sim z_2$: then the first equation gives, on linearising,

$$2y_1' + \left(z_2^2 - 1\right) y_1 + \left(2z_1z_2 + 1\right) y_2 = 2z_1z_2^2.$$

The starting approximation is taken to be $z_1 = 0$, $z_2 = 3$. In the program below, the array B is used to hold the coefficients of the previous iterate (or of the starting approximation). We iterate until the Chebyshev coefficients converge to 5 figures. E02AKF is used to calculate the solution from its Chebyshev coefficients.

9.1 Program Text

Note. The listing of the example program presented below uses bold italicised terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     D02TGF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          N, MIMAX, K1, IC, KP, LSUM, LW, LIW
      PARAMETER        (N=2,MIMAX=8,K1=MIMAX+1,IC=K1,KP=15,LSUM=3,
     +                 LW=2*(N*KP+LSUM)*(N*K1+1)+7*N*K1,LIW=N*K1)
      INTEGER          NOUT
      PARAMETER        (NOUT=6)
*     .. Scalars in Common ..
      real             X0, X1
*     .. Arrays in Common ..
      real             B(K1,N)
*     .. Local Scalars ..
      real             EMAX, X
      INTEGER          I, IA1, IFAIL, ITER, J, K
*     .. Local Arrays ..
      real             C(IC,N), W(LW), Y(N)
      INTEGER          IW(LIW), L(N), M(N)
*     .. External Subroutines ..
      EXTERNAL         BDYC, COEFF, D02TGF, E02AKF
*     .. Intrinsic Functions ..
      INTRINSIC        ABS, MAX, real
*     .. Common blocks ..
      COMMON           /ABC/B, X0, X1
*     .. Executable Statements ..
      WRITE (NOUT,*) 'D02TGF Example Program Results'
      X0 = -1.0e0
      X1 = 1.0e0
      M(1) = 1
      M(2) = 2
      L(1) = 1
      L(2) = 2
      DO 40 J = 1, N
         DO 20 I = 1, K1
            B(I,J) = 0.0e0
   20    CONTINUE
   40 CONTINUE
      B(1,2) = 6.0e0
      ITER = 0
   60 ITER = ITER + 1
      WRITE (NOUT,*)
      WRITE (NOUT,99999) ' Iteration', ITER,
     +  ' Chebyshev coefficients are'
      IFAIL = 1
*
```

```
      CALL D02TGF(N,M,L,X0,X1,K1,KP,C,IC,COEFF,BDYC,W,LW,IW,LIW,IFAIL)
*
      IF (IFAIL.EQ.0) THEN
         DO 80 J = 1, N
            WRITE (NOUT,99998) (C(I,J),I=1,K1)
   80    CONTINUE
         EMAX = 0.0e0
         DO 120 J = 1, N
            DO 100 I = 1, K1
               EMAX = MAX(EMAX,ABS(C(I,J)-B(I,J)))
               B(I,J) = C(I,J)
  100       CONTINUE
  120    CONTINUE
         IF (EMAX.LT.1.0e-5) THEN
            K = 9
            IA1 = 1
            WRITE (NOUT,*)
            WRITE (NOUT,99999) 'Solution evaluated at', K,
     +        '  equally spaced points'
            WRITE (NOUT,*)
            WRITE (NOUT,99997) '      X ', (J,J=1,N)
            DO 160 I = 1, K
               X = (X0*real(K-I)+X1*real(I-1))/real(K-1)
               DO 140 J = 1, N
                  IFAIL = 0
*
                  CALL E02AKF(K1,X0,X1,C(1,J),IA1,K1,X,Y(J),IFAIL)
*
  140          CONTINUE
               WRITE (NOUT,99996) X, (Y(J),J=1,N)
  160       CONTINUE
         ELSE
            GO TO 60
         END IF
      ELSE
         WRITE (NOUT,*)
         WRITE (NOUT,99999) 'D02TGF fails with IFAIL =', IFAIL
      END IF
      STOP
*
99999 FORMAT (1X,A,I3,A)
99998 FORMAT (1X,9F8.4)
99997 FORMAT (1X,A,2('      Y(',I1,')'))
99996 FORMAT (1X,3F10.4)
      END
*
      SUBROUTINE COEFF(X,I,A,IA,IA1,RHS)
*     .. Parameters ..
      INTEGER          N, MIMAX, K1
      PARAMETER        (N=2,MIMAX=8,K1=MIMAX+1)
*     .. Scalar Arguments ..
      real             RHS, X
      INTEGER          I, IA, IA1
*     .. Array Arguments ..
      real             A(IA,IA1)
*     .. Scalars in Common ..
      real             X0, X1
```

```
*     .. Arrays in Common ..
      real             B(K1,N)
*     .. Local Scalars ..
      real             Z1, Z2
      INTEGER          IFAIL
*     .. External Subroutines ..
      EXTERNAL         E02AKF
*     .. Common blocks ..
      COMMON           /ABC/B, X0, X1
*     .. Executable Statements ..
      IF (I.LE.1) THEN
         IA1 = 1
         IFAIL = 0
*
         CALL E02AKF(K1,X0,X1,B(1,1),IA1,K1,X,Z1,IFAIL)
         CALL E02AKF(K1,X0,X1,B(1,2),IA1,K1,X,Z2,IFAIL)
*
         A(1,1) = Z2*Z2 - 1.0e0
         A(1,2) = 2.0e0
         A(2,1) = 2.0e0*Z1*Z2 + 1.0e0
         RHS = 2.0e0*Z1*Z2*Z2
      ELSE
         A(1,2) = -1.0e0
         A(2,3) = 2.0e0
      END IF
      RETURN
      END
*
      SUBROUTINE BDYC(X,I,J,A,IA,IA1,RHS)
*     .. Scalar Arguments ..
      real            RHS, X
      INTEGER         I, IA, IA1, J
*     .. Array Arguments ..
      real            A(IA,IA1)
*     .. Executable Statements ..
      X = -1.0e0
      A(I,J) = 1.0e0
      IF (I.EQ.2 .AND. J.EQ.1) RHS = 3.0e0
      RETURN
      END
```

9.2 Program Data

None.

9.3 Program Results

```
 D02TGF Example Program Results

  Iteration  1 Chebyshev coefficients are
  -0.5659 -0.1162  0.0906 -0.0468  0.0196 -0.0069  0.0021 -0.0006  0.0001
   5.7083 -0.1642 -0.0087  0.0059 -0.0025  0.0009 -0.0003  0.0001  0.0000

  Iteration  2 Chebyshev coefficients are
  -0.6338 -0.1599  0.0831 -0.0445  0.0193 -0.0071  0.0023 -0.0006  0.0001
   5.6881 -0.1792 -0.0144  0.0053 -0.0023  0.0008 -0.0003  0.0001  0.0000
```

```
 Iteration  3 Chebyshev coefficients are
 -0.6344 -0.1604  0.0828 -0.0446  0.0193 -0.0071  0.0023 -0.0006  0.0001
  5.6880 -0.1793 -0.0145  0.0053 -0.0023  0.0008 -0.0003  0.0001  0.0000

 Iteration  4 Chebyshev coefficients are
 -0.6344 -0.1604  0.0828 -0.0446  0.0193 -0.0071  0.0023 -0.0006  0.0001
  5.6880 -0.1793 -0.0145  0.0053 -0.0023  0.0008 -0.0003  0.0001  0.0000

Solution evaluated at  9  equally spaced points

      X       Y(1)      Y(2)
   -1.0000    0.0000    3.0000
   -0.7500   -0.2372    2.9827
   -0.5000   -0.3266    2.9466
   -0.2500   -0.3640    2.9032
    0.0000   -0.3828    2.8564
    0.2500   -0.3951    2.8077
    0.5000   -0.4055    2.7577
    0.7500   -0.4154    2.7064
    1.0000   -0.4255    2.6538
```