NAG Library Routine Document

D02JBF

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

D02JBF solves a regular linear two-point boundary value problem for a system of ordinary differential equations by Chebyshev series using collocation and least squares.

2 Specification

```
SUBROUTINE D02JBF (N, CF, BC, X0, X1, K1, KP, C, LDC, W, LW, IW, LIW,      &
                   IFAIL)

INTEGER             N, K1, KP, LDC, LW, IW(LIW), LIW, IFAIL
REAL (KIND=nag_wp) CF, X0, X1, C(LDC,N), W(LW)
EXTERNAL            CF, BC
```

3 Description

D02JBF calculates the solution of a regular two-point boundary value problem for a regular linear nth-order system of first-order ordinary differential equations as a Chebyshev series in the interval (x_0, x_1). The differential equation

$$y' = A(x)y + r(x)$$

is defined by CF, and the boundary conditions at the points x_0 and x_1 are defined by BC.

You specify the degree of Chebyshev series required, K1 $-$ 1, and the number of collocation points, KP. The routine sets up a system of linear equations for the Chebyshev coefficients, n equations for each collocation point and one for each boundary condition. The boundary conditions are solved exactly, and the remaining equations are then solved by a least squares method. The result produced is a set of coefficients for a Chebyshev series solution for each component of the solution of the system of differential equations on an interval normalized to $(-1, 1)$.

E02AKF can be used to evaluate the components of the solution at any point on the interval (x_0, x_1) – see Section 10 for an example. E02AHF followed by E02AKF can be used to evaluate their derivatives.

4 References

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 Arguments

1: N – INTEGER *Input*

On entry: n, the order of the system of differential equations.

Constraint: N $\geq$ 1.

2: CF – REAL (KIND=nag_wp) FUNCTION, supplied by the user. *External Procedure*

CF defines the system of differential equations (see Section 3). It must return the value of a coefficient function $a_{i,j}(x)$, of A, at a given point x, or of a right-hand side function $r_i(x)$ if J $= 0$.

The specification of CF is:

```
FUNCTION CF (I, J, X)
REAL (KIND=nag_wp) CF
INTEGER            I, J
REAL (KIND=nag_wp) X
```

1: I – INTEGER *Input*

2: J – INTEGER *Input*

On entry: indicate the function to be evaluated, namely $a_{i,j}(x)$ if $1 \leq \mathrm{J} \leq n$, or $r_i(x)$ if $\mathrm{J} = 0$.
$1 \leq \mathrm{I} \leq n$, $0 \leq \mathrm{J} \leq n$.

3: X – REAL (KIND=nag_wp) *Input*

On entry: the point at which the function is to be evaluated.

CF must either be a module subprogram USEd by, or declared as EXTERNAL in, the (sub) program from which D02JBF is called. Arguments denoted as *Input* must **not** be changed by this procedure.

3: BC – SUBROUTINE, supplied by the user. *External Procedure*

BC defines the n boundary conditions, which have the form $y_k(x_0) = s$ or $y_k(x_1) = s$. The boundary conditions may be specified in any order.

The specification of BC is:

```
SUBROUTINE BC (I, J, RHS)
INTEGER            I, J
REAL (KIND=nag_wp) RHS
```

1: I – INTEGER *Input*

On entry: the index of the boundary condition to be defined.

2: J – INTEGER *Output*

On exit: must be set to $-k$ if the ith boundary condition is $y_k(x_0) = s$, or to $+k$ if it is $y_k(x_1) = s$.

J must not be set to the same value k for two different values of I.

3: RHS – REAL (KIND=nag_wp) *Output*

On exit: the value s.

BC must either be a module subprogram USEd by, or declared as EXTERNAL in, the (sub) program from which D02JBF is called. Arguments denoted as *Input* must **not** be changed by this procedure.

4: X0 – REAL (KIND=nag_wp) *Input*

5: X1 – REAL (KIND=nag_wp) *Input*

On entry: the left- and right-hand boundaries, x_0 and x_1, respectively.

Constraint: $\mathrm{X1} > \mathrm{X0}$.

6: K1 – INTEGER *Input*

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

Constraint: K1 ≥ 2.

7: KP – INTEGER *Input*

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

Constraint: KP ≥ K1 − 1.

8: C(LDC, N) – REAL (KIND=nag_wp) array *Output*

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

$$y_k = \sum_{i=1}^{\mathrm{K1}} \mathrm{C}(i,k) T_{i-1}(x), \quad 1 \leq k \leq n$$

where $T_i(x)$ is the ith Chebyshev polynomial of the first kind, and $\sum$ denotes that the first coefficient, $\mathrm{C}(1,k)$, is halved.

9: LDC – INTEGER *Input*

On entry: the first dimension of the array C as declared in the (sub)program from which D02JBF is called.

Constraint: LDC ≥ K1.

10: W(LW) – REAL (KIND=nag_wp) array *Workspace*

11: LW – INTEGER *Input*

On entry: the dimension of the array W as declared in the (sub)program from which D02JBF is called.

Constraint: LW ≥ 2 × N × (KP + 1) × (N × K1 + 1) + 7 × N × K1.

12: IW(LIW) – INTEGER array *Workspace*

13: LIW – INTEGER *Input*

On entry: the dimension of the array IW as declared in the (sub)program from which D02JBF is called.

Constraint: LIW ≥ N × (K1 + 2).

14: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. If you are unfamiliar with this argument you should refer to Section 3.4 in How to Use the NAG Library and its Documentation for details.

For environments where it might be inappropriate to halt program execution when an error is detected, the value −1 or 1 is recommended. If the output of error messages is undesirable, then the value 1 is recommended. Otherwise, if you are not familiar with this argument, the recommended value is 0. **When the value −1 or 1 is used it is essential to test the value of IFAIL on exit.**

On exit: IFAIL = 0 unless the routine detects an error or a warning has been flagged (see Section 6).

6 Error Indicators and Warnings

If on entry IFAIL $= 0$ or -1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors or warnings detected by the routine:

IFAIL $= 1$

On entry, N < 1,
or X0 $\geq$ X1,
or K1 < 2,
or KP $<$ K1 $- 1$,
or LDC $<$ K1.

IFAIL $= 2$

On entry, LW $< 2 \times$ N $\times$ (KP $+ 1$) $\times$ (N $\times$ K1 $+ 1$) $+ 7 \times$ N $\times$ K1,
or LIW $<$ N $\times$ (K1 $+ 2$) (i.e., insufficient workspace).

IFAIL $= 3$

Either the boundary conditions are not linearly independent (that is, in BC the variable J is set to the same value k for two different values of I), 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 F04AMF).

IFAIL $= -99$

An unexpected error has been triggered by this routine. Please contact NAG.

See Section 3.9 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -399$

Your licence key may have expired or may not have been installed correctly.

See Section 3.8 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -999$

Dynamic memory allocation failed.

See Section 3.7 in How to Use the NAG Library and its Documentation for further information.

7 Accuracy

The Chebyshev coefficients are determined by a stable numerical method. The accuracy of the approximate solution may be checked by varying the degree of the polynomials and the number of collocation points (see Section 9).

8 Parallelism and Performance

D02JBF is not thread safe and should not be called from a multithreaded user program. Please see Section 3.12.1 in How to Use the NAG Library and its Documentation for more information on thread safety.

D02JBF makes calls to BLAS and/or LAPACK routines, which may be threaded within the vendor library used by this implementation. Consult the documentation for the vendor library for further information.

Please consult the X06 Chapter Introduction for information on how to control and interrogate the OpenMP environment used within this routine. Please also consult the Users' Note for your implementation for any additional implementation-specific information.

9 Further Comments

The time taken by D02JBF depends on the size and complexity of the differential system, the degree of the polynomial solution, and the number of matching points.

The collocation points in the interval (x_0, x_1) are chosen to be the extrema of the appropriate shifted Chebyshev polynomial. If $\mathrm{KP} = \mathrm{K1} - 1$, then the least squares solution reduces to the solution of a system of linear equations, and true collocation results.

The accuracy of the solution may be checked by repeating the calculation with different values of K1 and with KP fixed but $\mathrm{KP} \gg \mathrm{K1} - 1$. If the Chebyshev coefficients decrease rapidly for each component (and consistently for various K1 and KP), the size of the last two or three gives an indication of the error. If the Chebyshev coefficients do not decay rapidly, it is likely that the solution cannot be well-represented by Chebyshev series. Note that the Chebyshev coefficients are calculated for the interval $(-1, 1)$.

Linear systems of high-order equations in their original form, singular problems, and, indirectly, nonlinear problems can be solved using D02TGF.

10 Example

This example solves the equation

$$y'' + y = 1$$

with boundary conditions

$$y(-1) = y(1) = 0.$$

The equation is written as the first-order system

$$\begin{pmatrix} y_1' \\ y_2' \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

for solution by D02JBF and the boundary conditions are written

$$y_1(-1) = y_1(1) = 0.$$

We use $\mathrm{K1} = 4$, 6 and 8, and $\mathrm{KP} = 10$ and 15, so that the different Chebyshev series may be compared. The solution for $\mathrm{K1} = 8$ and $\mathrm{KP} = 15$ is evaluated by E02AKF at nine equally spaced points over the interval $(-1, 1)$.

10.1 Program Text

```
!   D02JBF Example Program Text
!   Mark 26 Release. NAG Copyright 2016.

    Module d02jbfe_mod

!     D02JBF Example Program Module:
!            Parameters and User-defined Routines

!     .. Use Statements ..
      Use nag_library, Only: nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Accessibility Statements ..
      Private
```

```
      Public                           :: bc, cf
!     .. Parameters ..
      Integer, Parameter, Public       :: nin = 5, nout = 6
    Contains
      Function cf(i,j,x)

!       .. Function Return Value ..
        Real (Kind=nag_wp)             :: cf
!       .. Parameters ..
        Integer, Parameter             :: n = 2
        Real (Kind=nag_wp), Parameter  :: a(n,n) = reshape((/0.0E0_nag_wp,     &
                                          -1.0E0_nag_wp,1.0E0_nag_wp,          &
                                          0.0E0_nag_wp/),(/n,n/))
        Real (Kind=nag_wp), Parameter  :: r(n) = (/0.0E0_nag_wp,1.0E0_nag_wp/)
!       .. Scalar Arguments ..
        Real (Kind=nag_wp), Intent (In) :: x
        Integer, Intent (In)           :: i, j
!       .. Intrinsic Procedures ..
        Intrinsic                      :: reshape
!       .. Executable Statements ..
        If (j>0) Then
          cf = a(i,j)
        End If
        If (j==0) Then
          cf = r(i)
        End If
        Return
      End Function cf

      Subroutine bc(i,j,rhs)

!       .. Scalar Arguments ..
        Real (Kind=nag_wp), Intent (Out) :: rhs
        Integer, Intent (In)           :: i
        Integer, Intent (Out)          :: j
!       .. Executable Statements ..
        rhs = 0.0E0_nag_wp
        If (i>1) Then
          j = -1
        Else
          j = 1
        End If
        Return
      End Subroutine bc
    End Module d02jbfe_mod

    Program d02jbfe

!     D02JBF Example Main Program

!     .. Use Statements ..
      Use nag_library, Only: d02jbf, e02akf, nag_wp
      Use d02jbfe_mod, Only: bc, cf, nin, nout
!     .. Implicit None Statement ..
      Implicit None
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: dx, x, x0, x1
      Integer                          :: i, ia1, ifail, j, k1, k1max, kp,     &
                                          kpmax, ldc, liw, lw, m, n
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: c(:,:), w(:), y(:)
      Integer, Allocatable             :: iw(:)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: real
!     .. Executable Statements ..
      Write (nout,*) 'D02JBF Example Program Results'
!     Skip heading in data file
      Read (nin,*)
!     n: order of the system of differential equations
!     k1: number of coefficients to be returned
!     kp: number of collocation points
```

```
      Read (nin,*) n, k1max, kpmax
      ldc = k1max
      liw = n*(k1max+2)
      lw = 2*n*(kpmax+1)*(n*k1max+1) + 7*n*k1max
      Allocate (iw(liw),c(ldc,n),w(lw),y(n))
!     x0: left-hand boundary, x1: right-hand boundary.
      Read (nin,*) x0, x1
      Write (nout,*)
      Write (nout,*) ' KP  K1   Chebyshev coefficients'
      Do kp = 10, kpmax, 5
        Do k1 = 4, k1max, 2

!         ifail: behaviour on error exit
!                =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
          ifail = 0
          Call d02jbf(n,cf,bc,x0,x1,k1,kp,c,ldc,w,lw,iw,liw,ifail)

          Write (nout,99999) kp, k1, c(1:k1,1)
          Write (nout,99998)(c(1:k1,j),j=2,n)
          Write (nout,*)
        End Do
      End Do
      k1 = 8
      m = 9
      ia1 = 1
      Write (nout,99997) 'Last computed solution evaluated at', m,             &
        '  equally spaced points'
      Write (nout,*)
      Write (nout,99996) '      X ', (j,j=1,n)
      dx = (x1-x0)/real(m-1,kind=nag_wp)
      x = x0
      Do i = 1, m
        Do j = 1, n

          ifail = 0
          Call e02akf(k1,x0,x1,c(1,j),ia1,ldc,x,y(j),ifail)

        End Do
        Write (nout,99995) x, y(1:n)
        x = x + dx
      End Do

99999 Format (1X,2(I3,1X),8F8.4)
99998 Format (9X,8F8.4)
99997 Format (1X,A,I5,A)
99996 Format (1X,A,2('      Y(',I1,')'))
99995 Format (1X,3F10.4)
    End Program d02jbfe
```

10.2 Program Data

```
D02JBF Example Program Data
   2 8 15             : n, k1max, kpmax
   -1.0  1.0          : x0, x1
```

10.3 Program Results

```
 D02JBF Example Program Results

  KP  K1   Chebyshev coefficients
  10   4  -0.7798  0.0000  0.3899 -0.0000
           0.0000  1.5751  0.0000 -0.0629

  10   6  -0.8326 -0.0000  0.4253  0.0000 -0.0090 -0.0000
          -0.0000  1.6290  0.0000 -0.0724 -0.0000  0.0009

  10   8  -0.8325 -0.0000  0.4253  0.0000 -0.0092 -0.0000  0.0001  0.0000
          -0.0000  1.6289  0.0000 -0.0724 -0.0000  0.0009  0.0000 -0.0000

  15   4  -0.7829  0.0000  0.3914 -0.0000
```

```
          0.0000  1.5778 -0.0000 -0.0631

  15   6  -0.8326 -0.0000  0.4253  0.0000 -0.0090  0.0000
          0.0000  1.6290  0.0000 -0.0724 -0.0000  0.0009

  15   8  -0.8325 -0.0000  0.4253  0.0000 -0.0092  0.0000  0.0001 -0.0000
          0.0000  1.6289  0.0000 -0.0724 -0.0000  0.0009  0.0000 -0.0000

 Last computed solution evaluated at    9  equally spaced points

       X       Y(1)      Y(2)
    -1.0000    0.0000   -1.5574
    -0.7500   -0.3542   -1.2616
    -0.5000   -0.6242   -0.8873
    -0.2500   -0.7933   -0.4579
     0.0000   -0.8508    0.0000
     0.2500   -0.7933    0.4579
     0.5000   -0.6242    0.8873
     0.7500   -0.3542    1.2616
     1.0000    0.0000    1.5574
```