NAG Library Routine Document

D02JAF

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

D02JAF solves a regular linear two-point boundary value problem for a single nth-order ordinary differential equation by Chebyshev series using collocation and least squares.

2 Specification

```
SUBROUTINE D02JAF (N, CF, BC, X0, X1, K1, KP, C, W, LW, IW, IFAIL)
INTEGER            N, K1, KP, LW, IW(K1), IFAIL
REAL (KIND=nag_wp) CF, X0, X1, C(K1), W(LW)
EXTERNAL           CF, BC
```

3 Description

D02JAF calculates the solution of a regular two-point boundary value problem for a single nth-order linear ordinary differential equation as a Chebyshev series in the interval (x_0, x_1). The differential equation

$$f_{n+1}(x)y^{(n)}(x) + f_n(x)y^{(n-1)}(x) + \cdots + f_1(x)y(x) = f_0(x)$$

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, $\mathrm{K1} - 1$, and the number of collocation points, KP. The routine sets up a system of linear equations for the Chebyshev coefficients, one equation 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 of the differential equation on an interval normalized to $(-1, 1)$.

E02AKF can be used to evaluate 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 its 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 differential equation.

Constraint: $\mathrm{N} \geq 1$.

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

CF defines the differential equation (see Section 3). It must return the value of a function $f_j(x)$ at a given point x, where, for $1 \leq j \leq n+1$, $f_j(x)$ is the coefficient of $y^{(j-1)}(x)$ in the equation, and $f_0(x)$ is the right-hand side.

The specification of CF is:

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

1: J – INTEGER *Input*

On entry: the index of the function f_j to be evaluated.

2: X – REAL (KIND=nag_wp) *Input*

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

CF must either be a module subprogram USEd by, or declared as EXTERNAL in, the (sub) program from which D02JAF 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 boundary conditions, each of which has the form $y^{(k-1)}(x_1) = s_k$ or $y^{(k-1)}(x_0) = s_k$. 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 boundary condition is $y^{(k-1)}(x_0) = s_k$, and to $+k$ if it is $y^{(k-1)}(x_1) = s_k$.

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: must be set to the value s_k.

BC must either be a module subprogram USEd by, or declared as EXTERNAL in, the (sub) program from which D02JAF 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 solution (hence the degree of the polynomial approximation is $\mathrm{K1} - 1$).

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

7: KP – INTEGER *Input*

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

Constraint: $\mathrm{KP} \geq \mathrm{K1} - \mathrm{N}$.

8: C(K1) – REAL (KIND=nag_wp) array *Output*

On exit: the computed Chebyshev coefficients; that is, the computed solution is:

$$\sum_{i=1}^{\mathrm{K1}}{}' \mathrm{C}(i) T_{i-1}(x)$$

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

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

10: LW – INTEGER *Input*

On entry: the dimension of the array W as declared in the (sub)program from which D02JAF is called.

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

11: IW(K1) – INTEGER array *Workspace*

12: 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: $\mathrm{IFAIL} = 0$ unless the routine detects an error or a warning has been flagged (see Section 6).

6 Error Indicators and Warnings

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

Errors or warnings detected by the routine:

$\mathrm{IFAIL} = 1$

On entry, $\mathrm{N} < 1$,
or $\mathrm{X0} \geq \mathrm{X1}$,
or $\mathrm{K1} < \mathrm{N} + 1$,
or $\mathrm{KP} < \mathrm{K1} - \mathrm{N}$.

$\mathrm{IFAIL} = 2$

On entry, $\mathrm{LW} < 2 \times (\mathrm{KP} + \mathrm{N}) \times (\mathrm{K1} + 1) + 7 \times \mathrm{K1}$ (insufficient workspace).

$\mathrm{IFAIL} = 3$

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 polynomial and the number of collocation points (see Section 9).

8 Parallelism and Performance

D02JAF 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.

D02JAF 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 D02JAF depends on the complexity of the differential equation, 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 $\mathrm{KP} = \mathrm{K1} - \mathrm{N}$, 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 $\mathrm{KP} \gg \mathrm{K1} - \mathrm{N}$. If the Chebyshev coefficients decrease rapidly (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 $(-1, 1)$.

Systems of regular linear differential equations can be solved using D02JBF. It is necessary before using D02JBF to write the differential equations as a first-order system. 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

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

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

10.1 Program Text

```
!   D02JAF Example Program Text
!   Mark 26 Release. NAG Copyright 2016.

    Module d02jafe_mod

!     D02JAF 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(j,x)

!       .. Function Return Value ..
        Real (Kind=nag_wp)             :: cf
!       .. Scalar Arguments ..
        Real (Kind=nag_wp), Intent (In) :: x
        Integer, Intent (In)           :: j
!       .. Executable Statements ..
        If (j==2) Then
          cf = 0.0E0_nag_wp
        Else
          cf = 1.0E0_nag_wp
        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 d02jafe_mod

    Program d02jafe

!     D02JAF Example Main Program
```

```
!     .. Use Statements ..
      Use nag_library, Only: d02jaf, e02akf, nag_wp
      Use d02jafe_mod, Only: bc, cf, nin, nout
!     .. Implicit None Statement ..
      Implicit None
!     .. Local Scalars ..
      Real (Kind=nag_wp)             :: dx, x, x0, x1, y
      Integer                        :: i, ia1, ifail, k1, k1max, kp, kpmax, &
                                        lw, m, n
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: c(:), w(:)
      Integer, Allocatable           :: iw(:)
!     .. Intrinsic Procedures ..
      Intrinsic                      :: real
!     .. Executable Statements ..
      Write (nout,*) 'D02JAF Example Program Results'
!     Skip heading in data file
      Read (nin,*)
!     n: order of the differential equation
!     k1: number of coefficients to be returned
!     kp: number of collocation points
      Read (nin,*) n, k1max, kpmax
      lw = 2*(kpmax+n)*(k1max+1) + 7*k1max
      Allocate (iw(k1max),c(k1max),w(lw))
!     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 d02jaf(n,cf,bc,x0,x1,k1,kp,c,w,lw,iw,ifail)

          Write (nout,99999) kp, k1, c(1:k1)
        End Do
      End Do
      k1 = 8
      m = 9
      ia1 = 1
      Write (nout,*)
      Write (nout,99998) 'Last computed solution evaluated at', m,          &
        ' equally spaced points'
      Write (nout,*)
      Write (nout,*) '      X         Y'
      dx = (x1-x0)/real(m-1,kind=nag_wp)
      x = x0
      Do i = 1, m
        ifail = 0
        Call e02akf(k1,x0,x1,c,ia1,k1max,x,y,ifail)

        Write (nout,99997) x, y
        x = x + dx
      End Do

99999 Format (1X,2(I3,1X),8F8.4)
99998 Format (1X,A,I5,A)
99997 Format (1X,2F10.4)
    End Program d02jafe
```

10.2 Program Data

```
D02JAF Example Program Data
   2 8 15             : n, k1max, kpmax
   -1.0  1.0          : x0, x1
```

10.3 Program Results

```
D02JAF Example Program Results

 KP  K1   Chebyshev coefficients
 10   4  -0.6108 -0.0000  0.3054  0.0000
 10   6  -0.8316 -0.0000  0.4246  0.0000 -0.0088 -0.0000
 10   8  -0.8325 -0.0000  0.4253  0.0000 -0.0092  0.0000  0.0001 -0.0000
 15   4  -0.6174 -0.0000  0.3087  0.0000
 15   6  -0.8316 -0.0000  0.4246  0.0000 -0.0088 -0.0000
 15   8  -0.8325 -0.0000  0.4253  0.0000 -0.0092 -0.0000  0.0001 -0.0000

Last computed solution evaluated at    9 equally spaced points

      X         Y
   -1.0000    0.0000
   -0.7500   -0.3542
   -0.5000   -0.6242
   -0.2500   -0.7933
    0.0000   -0.8508
    0.2500   -0.7933
    0.5000   -0.6242
    0.7500   -0.3542
    1.0000    0.0000
```

Example Program
Two-point Boundary-value Problem for ODE
by Chebyshev-series using Collocation and Least-squares