NAG Library Routine Document

D02UCF

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

D02UCF returns the Chebyshev Gauss–Lobatto grid points on $[a,b]$.

2 Specification

```
SUBROUTINE D02UCF (N, A, B, X, IFAIL)

INTEGER              N, IFAIL
REAL (KIND=nag_wp)  A, B, X(N+1)
```

3 Description

D02UCF returns the Chebyshev Gauss–Lobatto grid points on $[a,b]$. The Chebyshev Gauss–Lobatto points on $[-1,1]$ are computed as $t_i = -\cos\left(\frac{(i-1)\pi}{n}\right)$, for $i = 1,2,\ldots,n+1$. The Chebyshev Gauss–Lobatto points on an arbitrary domain $[a,b]$ are:

$$x_i = \frac{b-a}{2} t_i + \frac{a+b}{2}, \qquad i = 1,2,\ldots,n+1.$$

4 References

Trefethen L N (2000) *Spectral Methods in MATLAB* SIAM

5 Parameters

1: N – INTEGER *Input*

On entry: n, where the number of grid points is $n+1$. This is also the largest order of Chebyshev polynomial in the Chebyshev series to be computed.

Constraint: $\mathrm{N} > 0$ and N is even.

2: A – REAL (KIND=nag_wp) *Input*

On entry: a, the lower bound of domain $[a,b]$.

Constraint: $\mathrm{A} < \mathrm{B}$.

3: B – REAL (KIND=nag_wp) *Input*

On entry: b, the upper bound of domain $[a,b]$.

Constraint: $\mathrm{B} > \mathrm{A}$.

4: X($\mathrm{N}+1$) – REAL (KIND=nag_wp) array *Output*

On exit: the Chebyshev Gauss–Lobatto grid points, x_i, for $i = 1,2,\ldots,n+1$, on $[a,b]$.

5: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this parameter you should refer to Section 3.3 in the Essential Introduction for details.

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

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 parameter, the recommended value is 0. **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**

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, $\mathrm{N} \leq 0$ or N is odd.

IFAIL $= 2$

On entry, $\mathrm{A} \geq \mathrm{B}$.

7 Accuracy

Should be close to ***machine precision***.

8 Further Comments

The number of operations is of the order $n\log n$ and there are no internal memory requirements; thus the computation remains efficient and practical for very fine discretizations (very large values of n).

9 Example

This example solves the first-order problem $U_x + U = 1$ on $[0.0, 1.5]$ subject to the boundary condition $U(0) = 1 + e^{-1.0}$, using the Chebyshev integration formulation on a Chebyshev Gauss–Lobatto grid of order 16.

9.1 Program Text

```
!   D02UCF Example Program Text
!   Mark 23 Release. NAG Copyright 2011.

    MODULE d02ucfe_mod

!      D02UCF Example Program Module:
!             Parameters and User-defined Routines

!      .. Use Statements ..
       USE nag_library, ONLY : nag_wp
!      .. Implicit None Statement ..
       IMPLICIT NONE
!      .. Parameters ..
       REAL (KIND=nag_wp), PARAMETER       :: a = 0.0_nag_wp
       REAL (KIND=nag_wp), PARAMETER       :: b = 1.5_nag_wp
       REAL (KIND=nag_wp), PARAMETER       :: one = 1.0_nag_wp
       REAL (KIND=nag_wp), PARAMETER       :: zero = 0.0_nag_wp
       INTEGER, PARAMETER                  :: m = 1, nin = 5, nout = 6
       LOGICAL, PARAMETER                  :: reqerr = .FALSE.
    CONTAINS
       FUNCTION exact(x)
```

```
!          .. Implicit None Statement ..
           IMPLICIT NONE
!          .. Function Return Value ..
           REAL (KIND=nag_wp)                   :: exact
!          .. Scalar Arguments ..
           REAL (KIND=nag_wp), INTENT (IN)      :: x
!          .. Intrinsic Functions ..
           INTRINSIC                               exp
!          .. Executable Statements ..
           exact = exp(-x-one) + one
           RETURN
        END FUNCTION exact
        FUNCTION deriv(x)

!          .. Implicit None Statement ..
           IMPLICIT NONE
!          .. Function Return Value ..
           REAL (KIND=nag_wp)                   :: deriv
!          .. Scalar Arguments ..
           REAL (KIND=nag_wp), INTENT (IN)      :: x
!          .. Intrinsic Functions ..
           INTRINSIC                               exp
!          ..Executable Statements ..
           deriv = -exp(-x-one)
           RETURN
        END FUNCTION deriv
        SUBROUTINE bndary(m,y,bmat,bvec)

!          .. Implicit None Statement ..
           IMPLICIT NONE
!          .. Scalar Arguments ..
           INTEGER, INTENT (IN)                 :: m
!          .. Array Arguments ..
           REAL (KIND=nag_wp), INTENT (OUT)     :: bmat(m,m+1), bvec(m), y(m)
!          .. Executable Statements ..
!          Boundary condition of left side of domain
           y(1) = a
!          Set up Dirichlet condition using exact solution at x=a
           bmat(1:m,1:m+1) = zero
           bmat(1,1) = one
           bvec(1) = exact(a)

           RETURN
        END SUBROUTINE bndary
        SUBROUTINE pdedef(m,f)

!          .. Implicit None Statement ..
           IMPLICIT NONE
!          .. Scalar Arguments ..
           INTEGER, INTENT (IN)                 :: m
!          .. Array Arguments ..
           REAL (KIND=nag_wp), INTENT (OUT)     :: f(m+1)
!          .. Executable Statements ..
           f(1) = one
           f(2) = one

           RETURN
        END SUBROUTINE pdedef

     END MODULE d02ucfe_mod
     PROGRAM d02ucfe

!       D02UCF Example Main Program

!       .. Use Statements ..
        USE nag_library, ONLY : d02uaf, d02ubf, d02ucf, d02uef, x02ajf
        USE d02ucfe_mod, ONLY : a, b, bndary, deriv, exact, m, nag_wp, nin,     &
                                nout, one, pdedef, reqerr, zero
!       .. Implicit None Statement ..
        IMPLICIT NONE
!       .. Local Scalars ..
```

```
       REAL (KIND=nag_wp)                     :: resid, teneps, uerr, uxerr
       INTEGER                                :: i, ifail, n
!      .. Local Arrays ..
       REAL (KIND=nag_wp)                     :: bmat(m,m+1), bvec(1:m), f(m+1),  &
                                                 y(m)
       REAL (KIND=nag_wp), ALLOCATABLE        :: c(:), f0(:), u(:), uc(:,:),      &
                                                 ux(:), x(:)
!      .. Intrinsic Functions ..
       INTRINSIC                                 abs, int, max
!      .. Executable Statements ..
       WRITE (nout,*) ' D02UCF Example Program Results '
       WRITE (nout,*)

       READ (nin,*)
       READ (nin,*) n

       ALLOCATE (f0(n+1),c(n+1),u(n+1),ux(n+1),uc(n+1,m+1),x(n+1))

!      Set up problem boundary conditions and definition
       CALL bndary(m,y,bmat,bvec)
       CALL pdedef(m,f)

!      Set up solution grid
       ifail = 0
       CALL d02ucf(n,a,b,x,ifail)

!      Set up problem right hand sides for grid and transform
       f0(1:n+1) = one
       CALL d02uaf(n,f0,c,ifail)

!      Solve in coefficient space
       ifail = 0
       CALL d02uef(n,a,b,m,c,bmat,y,bvec,f,uc,resid,ifail)
!      Transform solution and derivative back to real space
       ifail = 0
       CALL d02ubf(n,a,b,0,uc(1,1),u,ifail)
       ifail = 0
       CALL d02ubf(n,a,b,1,uc(1,2),ux,ifail)

!      Print solution
       WRITE (nout,*) ' Numerical solution U and derivative Ux'
       WRITE (nout,*)
       WRITE (nout,99999)
       WRITE (nout,99998) (x(i),u(i),ux(i),i=1,n+1)

       IF (reqerr) THEN
          uerr = zero
          uxerr = zero
          DO i = 1, n + 1
             uerr = max(uerr,abs(u(i)-exact(x(i))))
             uxerr = max(uxerr,abs(ux(i)-deriv(x(i))))
          END DO
          teneps = 10.0_nag_wp*x02ajf()
          WRITE (nout,99997) 10*(int(uerr/teneps)+1)
          WRITE (nout,99996) 10*(int(uxerr/teneps)+1)
       END IF

99999  FORMAT (1X,T8,'X',T18,'U',T28,'Ux')
99998  FORMAT (1X,3F10.4)
99997  FORMAT (//1X,'U  is within a multiple ',I8,' of machine precision.')
99996  FORMAT (1X,'Ux is within a multiple ',I8,' of machine precision.')

    END PROGRAM d02ucfe
```

9.2 Program Data

```
D02UCF Example Program Data
  16              : N
```

9.3 Program Results

```
D02UCF Example Program Results

Numerical solution U and derivative Ux

     X         U         Ux
   0.0000    1.3679   -0.3679
   0.0144    1.3626   -0.3626
   0.0571    1.3475   -0.3475
   0.1264    1.3242   -0.3242
   0.2197    1.2953   -0.2953
   0.3333    1.2636   -0.2636
   0.4630    1.2315   -0.2315
   0.6037    1.2012   -0.2012
   0.7500    1.1738   -0.1738
   0.8963    1.1501   -0.1501
   1.0370    1.1304   -0.1304
   1.1667    1.1146   -0.1146
   1.2803    1.1023   -0.1023
   1.3736    1.0931   -0.0931
   1.4429    1.0869   -0.0869
   1.4856    1.0833   -0.0833
   1.5000    1.0821   -0.0821
```