NAG Library Routine Document

D02UZF

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

D02UZF returns the value of the kth Chebyshev polynomial evaluated at a point $x \in [-1, 1]$. D02UZF is primarily a utility routine for use by the Chebyshev boundary value problem solvers.

2 Specification

```
SUBROUTINE D02UZF (K, X, T, IFAIL)
INTEGER            K, IFAIL
REAL (KIND=nag_wp) X, T
```

3 Description

D02UZF returns the value, T, of the kth Chebyshev polynomial evaluated at a point $x \in [-1, 1]$; that is, $T = \cos(k \times \arccos(x))$.

4 References

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

5 Arguments

1: K – INTEGER *Input*

On entry: the order of the Chebyshev polynomial.

Constraint: $\mathrm{K} \geq 0$.

2: X – REAL (KIND=nag_wp) *Input*

On entry: the point at which to evaluate the polynomial.

Constraint: $-1.0 \leq \mathrm{X} \leq 1.0$.

3: T – REAL (KIND=nag_wp) *Output*

On exit: the value, T, of the Chebyshev polynomial order k evaluated at x.

4: 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 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, K $= \langle value \rangle$.
Constraint: K ≥ 0.

IFAIL $= 2$

On entry, X $= \langle value \rangle$.
Constraint: $-1.0 \leq$ X ≤ 1.0.

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 accuracy should be close to ***machine precision***.

8 Parallelism and Performance

D02UZF is not threaded in any implementation.

9 Further Comments

None.

10 Example

A set of Chebyshev coefficients is obtained for the function $x + \exp(-x)$ defined on $[-0.24 \times \pi, 0.5 \times \pi]$ using D02UCF. At each of a set of new grid points in the domain of the function D02UZF is used to evaluate each Chebshev polynomial in the series representation. The values obtained are multiplied to the Chebyshev coefficients and summed to obtain approximations to the given function at the new grid points.

10.1 Program Text

```
!   D02UZF Example Program Text
!   Mark 26 Release. NAG Copyright 2016.

    Module d02uzfe_mod

!     D02UZF 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                           :: exact
!     .. Parameters ..
      Real (Kind=nag_wp), Parameter, Public :: one = 1.0_nag_wp
      Real (Kind=nag_wp), Parameter, Public :: two = 2.0_nag_wp
      Real (Kind=nag_wp), Parameter, Public :: zero = 0.0_nag_wp
      Integer, Parameter, Public       :: nin = 5, nout = 6
      Logical, Parameter, Public       :: reqerr = .False.
!     .. Local Scalars ..
      Real (Kind=nag_wp), Public, Save :: a, b
    Contains
      Function exact(x)

!       .. Function Return Value ..
        Real (Kind=nag_wp)             :: exact
!       .. Scalar Arguments ..
        Real (Kind=nag_wp), Intent (In) :: x
!       .. Intrinsic Procedures ..
        Intrinsic                      :: exp
!       .. Executable Statements ..
        exact = x + exp(-x)
        Return
      End Function exact
    End Module d02uzfe_mod
    Program d02uzfe

!     D02UZF Example Main Program

!     .. Use Statements ..
      Use nag_library, Only: d02uaf, d02ucf, d02uzf, nag_wp, x01aaf, x02ajf
      Use d02uzfe_mod, Only: a, b, exact, nin, nout, one, reqerr, two, zero
!     .. Implicit None Statement ..
      Implicit None
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: deven, dmap, fseries, pi, t, teneps, &
                                          uerr, xeven, xmap
      Integer                          :: i, ifail, k, m, n
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: c(:), f(:), x(:)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: abs, int, max, min, real
!     .. Executable Statements ..
      Write (nout,*) ' D02UZF Example Program Results '
      Write (nout,*)

      Read (nin,*)
      Read (nin,*) n, m

      Allocate (f(n+1),c(n+1),x(n+1))

!     Set up problem boundary conditions and definition
      pi = x01aaf(pi)
      a = -0.24_nag_wp*pi
      b = pi/two

!     Set up Chebyshev grid
      ifail = 0
      Call d02ucf(n,a,b,x,ifail)

!     Evaluate function on grid and get interpolating Chebyshev coefficients.
      Do i = 1, n + 1
        f(i) = exact(x(i))
      End Do
      ifail = 0
      Call d02uaf(n,f,c,ifail)
```

```
!     Evaluate Chebyshev series manually by evaluating each Chebyshev
!     polynomial in turn at new equispaced (m+1) grid points.
!     Chebyshev series on [-1,1] map of [a,b].
      xmap = -one
      dmap = two/real(m-1,kind=nag_wp)
      xeven = a
      deven = (b-a)/real(m-1,kind=nag_wp)

      Write (nout,99999)
      uerr = zero
      Do i = 1, m
        fseries = zero
        Do k = 0, n
          ifail = 0
          Call d02uzf(k,xmap,t,ifail)
          fseries = fseries + c(k+1)*t
        End Do
        uerr = max(uerr,abs(fseries-exact(xeven)))
        Write (nout,99998) xmap, xeven, fseries
        xmap = min(one,xmap+dmap)
        xeven = xeven + deven
      End Do

      If (reqerr) Then
        teneps = 10.0_nag_wp*x02ajf()
        Write (nout,'(//)')
        Write (nout,99997) 10*(int(uerr/teneps)+1)
      End If

99999 Format (1X,T6,'x_even',T17,'x_map',T28,'Sum')
99998 Format (1X,3F10.4)
99997 Format (1X,'Error in coefficient sum is < ',I8,' * machine precision.')

    End Program d02uzfe
```

10.2 Program Data

```
D02UZF Example Program Data
  16, 9              : N, M
```

10.3 Program Results

```
  D02UZF Example Program Results

     x_even     x_map      Sum
    -1.0000   -0.7540    1.3715
    -0.7500   -0.4634    1.1261
    -0.5000   -0.1728    1.0158
    -0.2500    0.1178    1.0067
     0.0000    0.4084    1.0731
     0.2500    0.6990    1.1961
     0.5000    0.9896    1.3613
     0.7500    1.2802    1.5582
     1.0000    1.5708    1.7787
```