NAG Library Routine Document

D02UWF

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

D02UWF interpolates from a set of function values on a supplied grid onto a set of values for a uniform grid on the same range. The interpolation is performed using barycentric Lagrange interpolation. D02UWF is primarily a utility routine to map a set of function values specified on a Chebyshev Gauss–Lobatto grid onto a uniform grid.

2 Specification

```
SUBROUTINE D02UWF (N, NIP, X, F, XIP, FIP, IFAIL)
INTEGER            N, NIP, IFAIL
REAL (KIND=nag_wp) X(N+1), F(N+1), XIP(NIP), FIP(NIP)
```

3 Description

D02UWF interpolates from a set of $n+1$ function values, $f(x_i)$, on a supplied grid, x_i, for $i=0,1,\ldots,n$, onto a set of m values, $\hat{f}(\hat{x}_j)$, on a uniform grid, $\hat{x}_j$, for $j=1,2,\ldots,m$. The image $\hat{x}$ has the same range as x, so that $\hat{x}_j = x_{\text{min}} + ((j-1)/(m-1)) \times (x_{\text{max}} - x_{\text{min}})$, for $j=1,2,\ldots,m$. The interpolation is performed using barycentric Lagrange interpolation as described in Berrut and Trefethen (2004).

D02UWF is primarily a utility routine to map a set of function values specified on a Chebyshev Gauss–Lobatto grid computed by D02UCF onto an evenly-spaced grid with the same range as the original grid.

4 References

Berrut J P and Trefethen L N (2004) Barycentric lagrange interpolation *SIAM Rev.* **46(3)** 501–517

5 Arguments

1: N – INTEGER *Input*

On entry: n, where the number of grid points for the input data is $n+1$.

Constraint: $N > 0$ and N is even.

2: NIP – INTEGER *Input*

On entry: the number, m, of grid points in the uniform mesh $\hat{x}$ onto which function values are interpolated. If $\text{NIP} = 1$ then on successful exit from D02UWF, FIP(1) will contain the value $f(x_n)$.

Constraint: $\text{NIP} > 0$.

3: $\text{X}(\text{N}+1)$ – REAL (KIND=nag_wp) array *Input*

On entry: the grid points, x_i, for $i=0,1,\ldots,n$, at which the function is specified.

Usually this should be the array of Chebyshev Gauss–Lobatto points returned in D02UCF.

4: $\text{F}(\text{N}+1)$ – REAL (KIND=nag_wp) array *Input*

On entry: the function values, $f(x_i)$, for $i=0,1,\ldots,n$.

5: XIP(NIP) – REAL (KIND=nag_wp) array *Output*

On exit: the evenly-spaced grid points, $\hat{x}_j$, for $j = 1, 2, \ldots, m$.

6: FIP(NIP) – REAL (KIND=nag_wp) array *Output*

On exit: the set of interpolated values $\hat{f}(\hat{x}_j)$, for $j = 1, 2, \ldots, m$. Here $\hat{f}(\hat{x}_j) \approx f(x = \hat{x}_j)$.

7: 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 $= \langle value \rangle$.
Constraint: N > 0.

On entry, N $= \langle value \rangle$.
Constraint: N is even.

IFAIL $= 2$

On entry, NIP $= \langle value \rangle$.
Constraint: NIP > 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

D02UWF is intended, primarily, for use with Chebyshev Gauss–Lobatto input grids. For such input grids and for well-behaved functions (no discontinuities, peaks or cusps), the accuracy should be a small multiple of ***machine precision***.

8 Parallelism and Performance

D02UWF is not threaded in any implementation.

9 Further Comments

None.

10 Example

This example interpolates the function $x + \cos(5x)$, as specified on a 65-point Gauss–Lobatto grid on $[-1, 1]$, onto a coarse uniform grid.

10.1 Program Text

```
!   D02UWF Example Program Text
!   Mark 26 Release. NAG Copyright 2016.

    Module d02uwfe_mod

!     D02UWF 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 :: a = -1.0_nag_wp
      Real (Kind=nag_wp), Parameter, Public :: b = 1.0_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.
    Contains
      Function exact(x)

!       .. Function Return Value ..
        Real (Kind=nag_wp)              :: exact
!       .. Scalar Arguments ..
        Real (Kind=nag_wp), Intent (In) :: x
!       .. Intrinsic Procedures ..
        Intrinsic                       :: cos
!       .. Executable Statements ..
        exact = x + cos(5.0_nag_wp*x)
        Return
      End Function exact
    End Module d02uwfe_mod
    Program d02uwfe

!     D02UWF Example Main Program

!     .. Use Statements ..
      Use nag_library, Only: d02ucf, d02uwf, nag_wp, x02ajf
      Use d02uwfe_mod, Only: a, b, exact, nin, nout, reqerr, zero
!     .. Implicit None Statement ..
      Implicit None
!     .. Local Scalars ..
      Real (Kind=nag_wp)                :: uerr
      Integer                           :: i, ifail, iu, n, nip
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable   :: f(:), fip(:), x(:), xip(:)
!     .. Intrinsic Procedures ..
      Intrinsic                         :: abs, int, max
!     .. Executable Statements ..
```

```
      Write (nout,*) ' D02UWF Example Program Results '
      Write (nout,*)

      Read (nin,*)
      Read (nin,*) n, nip

      Allocate (f(n+1),fip(nip),xip(nip),x(n+1))

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

!     Set up problem right hand sides for grid
      Do i = 1, n + 1
        f(i) = exact(x(i))
      End Do

!     Map to an equally spaced grid
      ifail = 0
      Call d02uwf(n,nip,x,f,xip,fip,ifail)

!     Print solution
      Write (nout,*) ' Numerical solution F'
      Write (nout,*)
      Write (nout,99999)
      Write (nout,99998)(xip(i),fip(i),i=1,nip)

      If (reqerr) Then
        uerr = zero
        Do i = 1, nip
          uerr = max(uerr,abs(fip(i)-exact(xip(i))))
        End Do
        iu = 10*(int(uerr/10.0_nag_wp/x02ajf())+1)
        Write (nout,99997) iu
      End If

99999 Format (1X,T8,'X',T19,'F')
99998 Format (1X,F10.4,1X,F10.4)
99997 Format (/,/,1X,'F  is within a multiple ',I8,' of machine precision.')
    End Program d02uwfe
```

10.2 Program Data

```
D02UWF Example Program Data
  64  17          : N  NIP
```

10.3 Program Results

```
 D02UWF Example Program Results

 Numerical solution F

      X          F
   -1.0000    -0.7163
   -0.8750    -1.2060
   -0.7500    -1.5706
   -0.6250    -1.6249
   -0.5000    -1.3011
   -0.3750    -0.6745
   -0.2500     0.0653
   -0.1250     0.6860
    0.0000     1.0000
    0.1250     0.9360
    0.2500     0.5653
    0.3750     0.0755
```

```
      0.5000    -0.3011
      0.6250    -0.3749
      0.7500    -0.0706
      0.8750     0.5440
      1.0000     1.2837
```