NAG Library Routine Document

E02RBF

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

E02RBF evaluates a rational function at a user-supplied point, given the numerator and denominator coefficients.

2 Specification

```
SUBROUTINE E02RBF (A, IA, B, IB, X, ANS, IFAIL)

INTEGER            IA, IB, IFAIL
REAL (KIND=nag_wp) A(IA), B(IB), X, ANS
```

3 Description

Given a real value x and the coefficients a_j, for $j = 0, 1, \ldots, l$ and b_k, for $k = 0, 1, \ldots, m$, E02RBF evaluates the rational function

$$\frac{\sum_{j=0}^{l} a_j x^j}{\sum_{k=0}^{m} b_k x^k}.$$

using nested multiplication (see Conte and de Boor (1965)).

A particular use of E02RBF is to compute values of the Padé approximants determined by E02RAF.

4 References

Conte S D and de Boor C (1965) *Elementary Numerical Analysis* McGraw–Hill

Peters G and Wilkinson J H (1971) Practical problems arising in the solution of polynomial equations *J. Inst. Maths. Applics.* **8** 16–35

5 Parameters

1: A(IA) – REAL (KIND=nag_wp) array *Input*

On entry: A$(j+1)$, for $j = 1, 2, \ldots, l+1$, must contain the value of the coefficient a_j in the numerator of the rational function.

2: IA – INTEGER *Input*

On entry: the value of $l + 1$, where l is the degree of the numerator.

Constraint: IA ≥ 1.

3: B(IB) – REAL (KIND=nag_wp) array *Input*

On entry: B$(k+1)$, for $k = 1, 2, \ldots, m+1$, must contain the value of the coefficient b_k in the denominator of the rational function.

Constraint: if IB $= 1$, B$(1) \neq 0.0$.

4: IB – INTEGER *Input*

On entry: the value of $m+1$, where m is the degree of the denominator.

Constraint: IB ≥ 1.

5: X – REAL (KIND=nag_wp) *Input*

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

6: ANS – REAL (KIND=nag_wp) *Output*

On exit: the result of evaluating the rational function at the given point x.

7: 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.

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.**

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 $= 1$

The rational function is being evaluated at or near a pole.

IFAIL $= 2$

On entry, IA < 1,
or IB < 1,
or B$(1) = 0.0$ when IB $= 1$ (so the denominator is identically zero).

7 Accuracy

A running error analysis for polynomial evaluation by nested multiplication using the recurrence suggested by Kahan (see Peters and Wilkinson (1971)) is used to detect whether you are attempting to evaluate the approximant at or near a pole.

8 Further Comments

The time taken is approximately proportional to $l+m$.

9 Example

This example first calls E02RAF to calculate the $4/4$ Padé approximant to e^x, and then uses E02RBF to evaluate the approximant at $x = 0.1, 0.2, \ldots, 1.0$.

9.1 Program Text

```
    Program e02rbfe

!     E02RBF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: e02raf, e02rbf, nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: l = 4, m = 4, nout = 6
      Integer, Parameter               :: ia = l + 1
      Integer, Parameter               :: ib = m + 1
      Integer, Parameter               :: ic = ia + ib - 1
      Integer, Parameter               :: iw = ib*(2*ib+3)
      Logical, Parameter               :: plot = .False.
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: ans, tval, x
      Integer                          :: i, ifail, nx
!     .. Local Arrays ..
      Real (Kind=nag_wp)               :: a(ia), b(ib), cc(ic), w(iw)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: abs, exp, real
!     .. Executable Statements ..
      If (.Not. plot) Then
        Write (nout,*) 'E02RBF Example Program Results'
        nx = 10
      Else
        nx = 30
      End If

      cc(1) = 1.0E0_nag_wp

      Do i = 1, ic - 1
        cc(i+1) = cc(i)/real(i,kind=nag_wp)
      End Do

      ifail = 0
      Call e02raf(ia,ib,cc,ic,a,b,w,iw,ifail)

      If (.Not. plot) Then
        Write (nout,*)
        Write (nout,*) '    X         Pade          True'
      End If

      Do i = 1, nx
        x = real(i,kind=nag_wp)/10.0_nag_wp

        ifail = 0
        Call e02rbf(a,ia,b,ib,x,ans,ifail)

        tval = exp(x)

        If (plot) Then
          Write (nout,99999) x, ans, tval, abs(tval-ans)/tval
        Else
          Write (nout,99998) x, ans, tval
        End If
      End Do

99999 Format (1X,F6.1,4E19.9)
99998 Format (1X,F6.1,3E15.5)
    End Program e02rbfe
```

9.2 Program Data

None.

9.3 Program Results

```
E02RBF Example Program Results

     X          Pade          True
   0.1    0.11052E+01    0.11052E+01
   0.2    0.12214E+01    0.12214E+01
   0.3    0.13499E+01    0.13499E+01
   0.4    0.14918E+01    0.14918E+01
   0.5    0.16487E+01    0.16487E+01
   0.6    0.18221E+01    0.18221E+01
   0.7    0.20138E+01    0.20138E+01
   0.8    0.22255E+01    0.22255E+01
   0.9    0.24596E+01    0.24596E+01
   1.0    0.27183E+01    0.27183E+01
```