NAG Library Routine Document

E02RAF

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

E02RAF calculates the coefficients in a Padé approximant to a function from its user-supplied Maclaurin expansion.

2 Specification

```
SUBROUTINE E02RAF (IA, IB, C, IC, A, B, W, JW, IFAIL)
INTEGER            IA, IB, IC, JW, IFAIL
REAL (KIND=nag_wp) C(IC), A(IA), B(IB), W(JW)
```

3 Description

Given a power series

$$c_0 + c_1x + c_2x^2 + \cdots + c_{l+m}x^{l+m} + \cdots$$

E02RAF uses the coefficients c_i, for $i = 0, 1, \ldots, l+m$, to form the $[l/m]$ Padé approximant of the form

$$\frac{a_0 + a_1x + a_2x^2 + \cdots + a_lx^l}{b_0 + b_1x + b_2x^2 + \cdots + b_mx^m}$$

with b_0 defined to be unity. The two sets of coefficients a_j, for $j = 0, 1, \ldots, l$, and b_k, for $k = 0, 1, \ldots, m$, in the numerator and denominator are calculated by direct solution of the Padé equations (see Graves–Morris (1979)); these values are returned through the argument list unless the approximant is degenerate.

Padé approximation is a useful technique when values of a function are to be obtained from its Maclaurin expansion but convergence of the series is unacceptably slow or even nonexistent. It is based on the hypothesis of the existence of a sequence of convergent rational approximations, as described in Baker and Graves–Morris (1981) and Graves–Morris (1979).

Unless there are reasons to the contrary (as discussed in Chapter 4, Section 2, Chapters 5 and 6 of Baker and Graves–Morris (1981)), one normally uses the diagonal sequence of Padé approximants, namely

$$\{[m/m], m = 0, 1, 2, \ldots\}.$$

Subsequent evaluation of the approximant at a given value of x may be carried out using E02RBF.

4 References

Baker G A Jr and Graves–Morris P R (1981) Padé approximants, Part 1: Basic theory *encyclopaedia of Mathematics and its Applications* Addison–Wesley

Graves–Morris P R (1979) The numerical calculation of Padé approximants *Padé Approximation and its Applications. Lecture Notes in Mathematics* (ed L Wuytack) **765** 231–245 Adison–Wesley

5 Arguments

1: IA – INTEGER *Input*
2: IB – INTEGER *Input*

On entry: IA must specify $l+1$ and IB must specify $m+1$, where l and m are the degrees of the numerator and denominator of the approximant, respectively.

Constraint: IA ≥ 1 and IB ≥ 1.

3: C(IC) – REAL (KIND=nag_wp) array *Input*

On entry: C(i) must specify, for $i = 1, 2, \ldots, l+m+1$, the coefficient of x^{i-1} in the given power series.

4: IC – INTEGER *Input*

On entry: the dimension of the array C as declared in the (sub)program from which E02RAF is called.

Constraint: IC $\geq$ IA + IB $-$ 1.

5: A(IA) – REAL (KIND=nag_wp) array *Output*

On exit: A($j+1$), for $j = 1, 2, \ldots, l+1$, contains the coefficient a_j in the numerator of the approximant.

6: B(IB) – REAL (KIND=nag_wp) array *Output*

On exit: B($k+1$), for $k = 1, 2, \ldots, m+1$, contains the coefficient b_k in the denominator of the approximant.

7: W(JW) – REAL (KIND=nag_wp) array *Workspace*
8: JW – INTEGER *Input*

On entry: the dimension of the array W as declared in the (sub)program from which E02RAF is called.

Constraint: JW $\geq$ IB $\times$ (2 $\times$ IB + 3).

9: 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, JW $<$ IB $\times$ (2 $\times$ IB + 3),
or IA < 1,

or IB < 1,
or IC $<$ IA + IB $- 1$

(so there are insufficient coefficients in the given power series to calculate the desired approximant).

IFAIL $= 2$

The Padé approximant is degenerate.

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 solution should be the best possible to the extent to which the solution is determined by the input coefficients. It is recommended that you determine the locations of the zeros of the numerator and denominator polynomials, both to examine compatibility with the analytic structure of the given function and to detect defects. (Defects are nearby pole-zero pairs; defects close to $x = 0.0$ characterise ill-conditioning in the construction of the approximant.) Defects occur in regions where the approximation is necessarily inaccurate. The example program calls C02AGF to determine the above zeros.

It is easy to test the stability of the computed numerator and denominator coefficients by making small perturbations of the original Maclaurin series coefficients (e.g., c_l or c_{l+m}). These questions of intrinsic error of the approximants and computational error in their calculation are discussed in Chapter 2 of Baker and Graves–Morris (1981).

8 Parallelism and Performance

E02RAF is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

E02RAF 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 is approximately proportional to m^3.

10 Example

This example calculates the [4/4] Padé approximant of e^x (whose power-series coefficients are first stored in the array C). The poles and zeros are then calculated to check the character of the [4/4] Padé approximant.

10.1 Program Text

```
    Program e02rafe

!     E02RAF Example Program Text

!     Mark 26 Release. NAG Copyright 2016.

!     .. Use Statements ..
      Use nag_library, Only: c02agf, e02raf, 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               :: jw = ib*(2*ib+3)
      Logical, Parameter               :: scale = .True.
!     .. Local Scalars ..
      Integer                          :: i, ifail
!     .. Local Arrays ..
      Real (Kind=nag_wp)               :: a(ia), b(ib), c(ic), dd(ia+ib),      &
                                          w(jw), work(2*(l+m+1)), z(2,l+m)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: real
!     .. Executable Statements ..
      Write (nout,*) 'E02RAF Example Program Results'

!     Power series coefficients in C

      c(1) = 1.0E0_nag_wp

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

      ifail = 0
      Call e02raf(ia,ib,c,ic,a,b,w,jw,ifail)

      Write (nout,*)
      Write (nout,*) 'The given series coefficients are'
      Write (nout,99999) c(1:ic)
      Write (nout,*)
      Write (nout,*) 'Numerator coefficients'
      Write (nout,99999) a(1:ia)
      Write (nout,*)
      Write (nout,*) 'Denominator coefficients'
      Write (nout,99999) b(1:ib)

!     Calculate zeros of the approximant using C02AGF
!     First need to reverse order of coefficients

      dd(ia:1:-1) = a(1:ia)

      ifail = 0
      Call c02agf(dd,l,scale,z,work,ifail)

      Write (nout,*)
      Write (nout,*) 'Zeros of approximant are at'
      Write (nout,*)
      Write (nout,*) '    Real part    Imag part'
      Write (nout,99998)(z(1,i),z(2,i),i=1,l)
```

```
!     Calculate poles of the approximant using C02AGF
!     Reverse order of coefficients

      dd(ib:1:-1) = b(1:ib)

      ifail = 0
      Call c02agf(dd,m,scale,z,work,ifail)

      Write (nout,*)
      Write (nout,*) 'Poles of approximant are at'
      Write (nout,*)
      Write (nout,*) '    Real part    Imag part'
      Write (nout,99998)(z(1,i),z(2,i),i=1,m)

99999 Format (1X,5E13.4)
99998 Format (1X,2E13.4)
    End Program e02rafe
```

10.2 Program Data

None.

10.3 Program Results

```
 E02RAF Example Program Results

 The given series coefficients are
    0.1000E+01   0.1000E+01   0.5000E+00   0.1667E+00   0.4167E-01
    0.8333E-02   0.1389E-02   0.1984E-03   0.2480E-04

 Numerator coefficients
    0.1000E+01   0.5000E+00   0.1071E+00   0.1190E-01   0.5952E-03

 Denominator coefficients
    0.1000E+01  -0.5000E+00   0.1071E+00  -0.1190E-01   0.5952E-03

 Zeros of approximant are at

     Real part    Imag part
   -0.5792E+01   0.1734E+01
   -0.5792E+01  -0.1734E+01
   -0.4208E+01   0.5315E+01
   -0.4208E+01  -0.5315E+01

 Poles of approximant are at

     Real part    Imag part
    0.5792E+01   0.1734E+01
    0.5792E+01  -0.1734E+01
    0.4208E+01   0.5315E+01
    0.4208E+01  -0.5315E+01
```