NAG Library Routine Document

C06BAF

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

C06BAF accelerates the convergence of a given convergent sequence to its limit.

2 Specification

```
SUBROUTINE C06BAF (SEQN, NCALL, RESULT, ABSERR, WORK, LWORK, IFAIL)
INTEGER            NCALL, LWORK, IFAIL
REAL (KIND=nag_wp) SEQN, RESULT, ABSERR, WORK(LWORK)
```

3 Description

C06BAF performs Shanks' transformation on a given sequence of real values by means of the Epsilon algorithm of Wynn (1956). A (possibly unreliable) estimate of the absolute error is also given.

The routine must be called repetitively, once for each new term in the sequence.

4 References

Shanks D (1955) Nonlinear transformations of divergent and slowly convergent sequences *J. Math. Phys.* **34** 1–42

Wynn P (1956) On a device for computing the $e_m(S_n)$ transformation *Math. Tables Aids Comput.* **10** 91–96

5 Arguments

1: SEQN – REAL (KIND=nag_wp) *Input*

On entry: the next term of the sequence to be considered.

2: NCALL – INTEGER *Input/Output*

On entry: on the first call NCALL must be set to 0. Thereafter NCALL **must not** be changed between calls.

On exit: the number of terms in the sequence that have been considered.

3: RESULT – REAL (KIND=nag_wp) *Output*

On exit: the estimate of the limit of the sequence. For the first two calls, RESULT = SEQN.

4: ABSERR – REAL (KIND=nag_wp) *Output*

On exit: an estimate of the absolute error in RESULT. For the first three calls, ABSERR is set to a large machine-dependent constant.

5: WORK(LWORK) – REAL (KIND=nag_wp) array *Communication Array*

Used as workspace, but **must not** be changed between calls.

6: LWORK – INTEGER *Input*

On entry: the dimension of the array WORK as declared in the (sub)program from which C06BAF is called.

Suggested value: (maximum number of terms in the sequence) + 6. See Section 9.2.

Constraint: LWORK ≥ 7.

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, NCALL < 0.

IFAIL $= 2$

On entry, LWORK < 7.

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 of the absolute error estimate ABSERR varies considerably with the type of sequence to which the routine is applied. In general it is better when applied to oscillating sequences than to monotonic sequences where it may be a severe underestimate.

8 Parallelism and Performance

C06BAF is not threaded in any implementation.

9 Further Comments

9.1 Timing

The time taken is approximately proportional to the final value of NCALL.

9.2 Choice of LWORK

For long sequences, a 'window' of the last n values can be used instead of all the terms of the sequence. Tests on a variety of problems indicate that a suitable value is $n = 50$; this implies a value for LWORK of 56. You are advised to experiment with other values for your own specific problems.

9.3 Convergence

C06BAF will induce convergence in some divergent sequences. See Shanks (1955) for more details.

10 Example

This example attempts to sum the infinite series

$$\sum_{n=1}^{\infty} \frac{(-1)^{n+1}}{n^2} = \frac{\pi^2}{12}$$

by considering the sequence of partial sums

$$\sum_{n=1}^{1}, \sum_{n=1}^{2}, \sum_{n=1}^{3}, \ldots, \sum_{n=1}^{10}$$

10.1 Program Text

```
    Program c06bafe

!     C06BAF Example Program Text

!     Mark 26 Release. NAG Copyright 2016.

!     .. Use Statements ..
      Use nag_library, Only: c06baf, nag_wp, x01aaf
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: lwork = 16, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: abserr, ans, error, pi, r, result,   &
                                          seqn, sig
      Integer                          :: i, ifail, ncall
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: work(:)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: real
!     .. Executable Statements ..
      Write (nout,*) 'C06BAF Example Program Results'
      Write (nout,*)

      Allocate (work(lwork))

      pi = x01aaf(pi)
      ans = pi**2/12.0_nag_wp
      ncall = 0
      sig = 1.0_nag_wp
      seqn = 0.0_nag_wp
      Write (nout,99999) 'Estimated       Actual'
      Write (nout,99998) 'I       SEQN       RESULT', 'abs error        error'
      Write (nout,*)
      Do i = 1, 10
```

```
        r = real(i,kind=nag_wp)
        seqn = seqn + sig/(r**2)

!       ifail: behaviour on error exit
!              =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
        ifail = 0
        Call c06baf(seqn,ncall,result,abserr,work,lwork,ifail)

        error = result - ans
        sig = -sig
        If (i<=3) Then
!         First three calls of C06BAF return no error estimate
          Write (nout,99997) i, seqn, result, error
        Else
          Write (nout,99996) i, seqn, result, abserr, error
        End If
      End Do

99999 Format (36X,A)
99998 Format (3X,A25,8X,A)
99997 Format (1X,I4,2F12.4,3X,10X,'-   ',E14.2)
99996 Format (1X,I4,2F12.4,3X,2E14.2)
    End Program c06bafe
```

10.2 Program Data

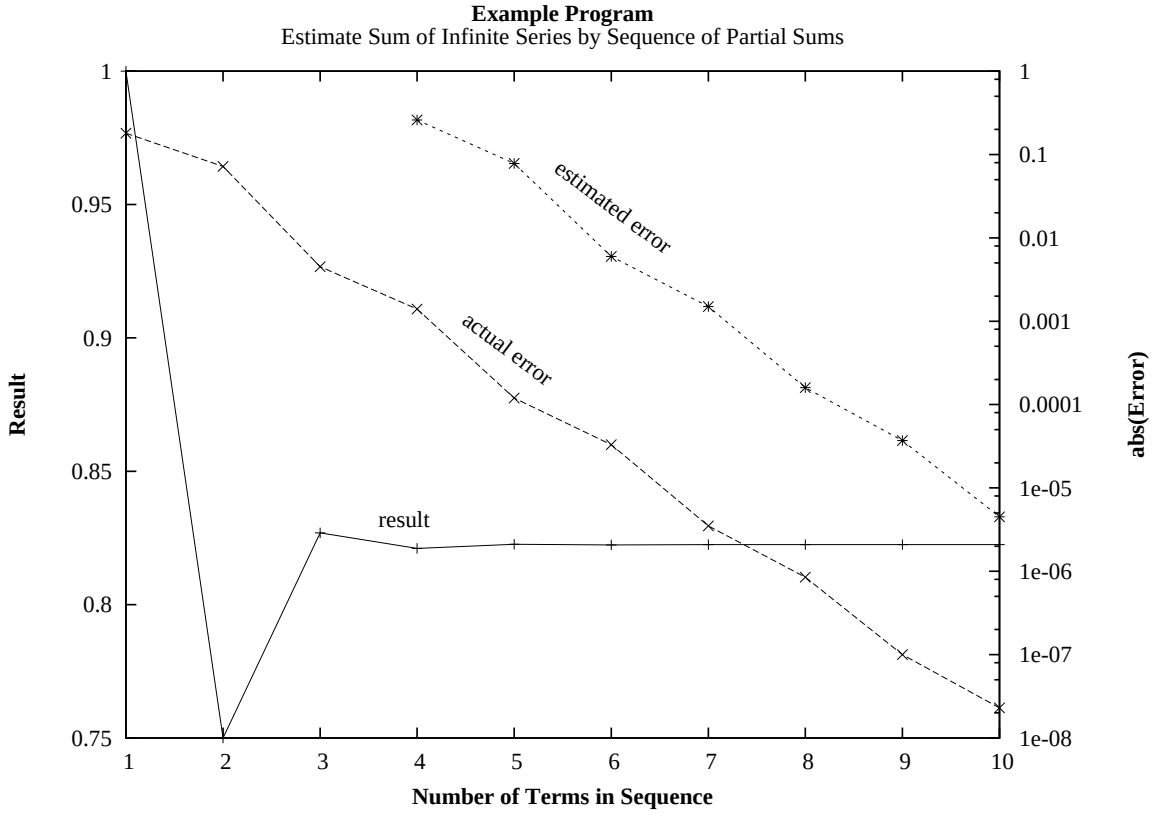
None.

10.3 Program Results

```
 C06BAF Example Program Results

                                  Estimated       Actual
   I       SEQN       RESULT      abs error        error

    1      1.0000      1.0000           -         0.18E+00
    2      0.7500      0.7500           -        -0.72E-01
    3      0.8611      0.8269           -         0.45E-02
    4      0.7986      0.8211      0.26E+00      -0.14E-02
    5      0.8386      0.8226      0.78E-01       0.12E-03
    6      0.8108      0.8224      0.60E-02      -0.33E-04
    7      0.8312      0.8225      0.15E-02       0.35E-05
    8      0.8156      0.8225      0.16E-03      -0.85E-06
    9      0.8280      0.8225      0.37E-04       0.10E-06
   10      0.8180      0.8225      0.45E-05      -0.23E-07
```


Example Program
Estimate Sum of Infinite Series by Sequence of Partial Sums
Result
abs(Error)
Number of Terms in Sequence
estimated error
actual error
result
1
0.95
0.9
0.85
0.8
0.75
0.1
0.01
0.001
0.0001
1e-05
1e-06
1e-07
1e-08
2
3
4
5
6
7
8
9
10