D05BYF – NAG Fortran Library Routine Document

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

D05BYF computes the fractional quadrature weights associated with the Backward Differentiation Formulae (BDF) of orders 4, 5 and 6. These weights can then be used in the solution of weakly singular equations of Abel type.

2 Specification

```
SUBROUTINE D05BYF(IORDER, IQ, LENFW, WT, SW, LDSW, WORK, LWK, IFAIL)
INTEGER           IORDER, IQ, LENFW, LDSW, LWK, IFAIL
real              WT(LENFW), SW(LDSW,2*IORDER-1), WORK(LWK)
```

3 Description

D05BYF computes the weights $W_{n,j}$ and ω_i for a family of quadrature rules related to a BDF method for approximating the integral:

$$\frac{1}{\sqrt{\pi}}\int_0^t \frac{\phi(s)}{\sqrt{t-s}}ds \simeq \sqrt{h}\sum_{j=0}^{2p-2} W_{n,j}\phi(jh) + \sqrt{h}\sum_{j=2p-1}^{n}\omega_{n-j}\phi(jh), \quad 0 \leq t \leq T, \tag{1}$$

with $t = nh$ ($n \geq 0$), for some given h. In (1),p is the order of the BDF method used and $W_{n,j}$, ω_i are the fractional starting and the fractional convolution weights respectively. The algorithm for the generation of ω_i is based on Newton's iteration. Fast Fourier transform (FFT) techniques are used for computing these weights and subsequently $W_{n,j}$ (see [1] and [2] for practical details and [3] for theoretical details). Some special functions can be represented as the fractional integrals of simpler functions and fractional quadratures can be employed for their computation (see [3]). A description of how these weights can be used in the solution of weakly singular equations of Abel type is given in Section 8.

4 References

[1] Baker C T H and Derakhshan M S (1987) Computational approximations to some power series *Approximation Theory* (ed L Collatz, G Meinardus and G Nürnberger) **81** 11–20

[2] Henrici P (1979) Fast Fourier methods in computational complex analysis *SIAM Rev.* **21** 481–529

[3] Lubich Ch (1986) Discretized fractional calculus *SIAM J. Math. Anal.* **17** 704–719

5 Parameters

1: IORDER — INTEGER *Input*

On entry: the order of the BDF method to be used, p.

Constraint: $4 \leq$ IORDER ≤ 6.

2: IQ — INTEGER *Input*

On entry: determines the number of weights to be computed. By setting IQ to a value, $2^{\text{IQ}+1}$ fractional convolution weights are computed.

Constraint: IQ ≥ 0.

3: LENFW — INTEGER *Input*

On entry: the length of the array WT.

Constraint: LENFW $\geq 2^{\mathrm{IQ}+2}$.

4: WT(LENFW) — ***real*** array *Output*

On exit: the first $2^{\mathrm{IQ}+1}$ elements of WT contains the fractional convolution weights ω_i, for $i = 0, 1, \ldots, 2^{\mathrm{IQ}+1} - 1$. The remainder of the array is used as workspace.

5: SW(LDSW,2*IORDER-1) — ***real*** array *Output*

On exit: SW$(n, j+1)$ contains the fractional starting weights $W_{n-1,j}$, for $n = 1, 2, \ldots, (2^{\mathrm{IQ}+1} + 2 \times \mathrm{IORDER} - 1)$; $j = 0, 1, \ldots, 2 \times \mathrm{IORDER} - 2$.

6: LDSW — INTEGER *Input*

On entry: the first dimension of the array SW as declared in the (sub)program from which D05BYF is called.

Constraint: LDSW $\geq 2^{\mathrm{IQ}+1} + 2 \times \mathrm{IORDER} - 1$.

7: WORK(LWK) — ***real*** array *Workspace*

8: LWK — INTEGER *Input*

On entry: the dimension of the array WORK as declared in the (sub)program from which D05BYF is called.

Constraint: LWK $\geq 2^{\mathrm{IQ}+3}$.

9: IFAIL — INTEGER *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (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 detected by the routine:

IFAIL = 1

On entry, IORDER < 4 or IORDER > 6,
or IQ < 0,
or LENFW $< 2^{\mathrm{IQ}+2}$,
or LDSW $< 2^{\mathrm{IQ}+1} + 2 \times \mathrm{IORDER} - 1$,
or LWK $< 2^{\mathrm{IQ}+3}$.

7 Accuracy

None.

8 Further Comments

Fractional quadrature weights can be used for solving weakly singular integral equations of Abel type. In this section, we propose the following algorithm which users may find useful in solving a linear weakly singular integral equation of the form

$$y(t) = f(t) + \frac{1}{\sqrt{\pi}} \int_0^t \frac{K(t,s)y(s)}{\sqrt{t-s}} ds, \quad 0 \leq t \leq T, \tag{2}$$

using D05BYF. In (2), $K(t,s)$ and $f(t)$ are given and the solution $y(t)$ is sought on a uniform mesh of size h such that $T = Nh$. Discretization of (2) yields

$$y_n = f(nh) + \sqrt{h} \sum_{j=0}^{2p-2} W_{n,j} K(nh, jh) y_j + \sqrt{h} \sum_{j=2p-1}^{n} \omega_{n-j} K(nh, jh) y_j, \tag{3}$$

where $y_n \simeq y(nh)$. We propose the following algorithm for computing y_n from (3) after a call to D05BYF:

(a) Set $N = 2^{\mathrm{IQ}+1} + 2 \times \mathrm{IORDER} - 2$ and $h = T/N$.

(b) Equation (3) requires $2 \times \mathrm{IORDER} - 2$ starting values, y_j, for $j = 1, 2, \ldots, 2 \times \mathrm{IORDER} - 2$, with $y_0 = f(0)$. These starting values can be computed by solving the system

$$y_n = f(nh) + \sqrt{h} \sum_{j=0}^{2\times\mathrm{IORDER}-2} \mathrm{SW}(n+1, j+1) K(nh, jh) y_j, \quad n = 1, 2, \ldots, 2 \times \mathrm{IORDER} - 2.$$

(c) Compute the inhomogeneous terms

$$\sigma_n = f(nh) + \sqrt{h} \sum_{j=0}^{2\times\mathrm{IORDER}-2} \mathrm{SW}(n+1, j+1) K(nh, jh) y_j, \quad n = 2\times\mathrm{IORDER}-1, 2\times\mathrm{IORDER}, \ldots, N.$$

(d) Start the iteration for $n = 2 \times \mathrm{IORDER} - 1, 2 \times \mathrm{IORDER}, \ldots, N$ to compute y_n from:

$$(1 - \sqrt{h}\mathrm{WT}(1) K(nh, nh)) y_n = \sigma_n + \sqrt{h} \sum_{j=2\times\mathrm{IORDER}-1}^{n-1} \mathrm{WT}(n-j+1) K(nh, jh) y_j.$$

Note that for nonlinear weakly singular equations, the solution of a nonlinear algebraic system is required at step (b) and a single nonlinear equation at step (d).

9 Example

The following example generates the first 16 fractional convolution and 23 fractional starting weights generated by the fourth-order BDF method.

9.1 Program Text

Note. The listing of the example program presented below uses bold italicised terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     D05BYF Example Program Text
*     Mark 16 Release. NAG Copyright 1992.
*     .. Parameters ..
      INTEGER          NOUT
      PARAMETER        (NOUT=6)
      INTEGER          IORDER, IQ, ITPMT, ITIQ, LENFW, LDSW, LWK
      PARAMETER        (IORDER=4,IQ=3,ITPMT=2*IORDER-1,ITIQ=2**(IQ+1),
     +                 LENFW=2*ITIQ,LDSW=ITIQ+ITPMT,LWK=4*ITIQ)
*     .. Local Scalars ..
      INTEGER          I, IFAIL, J
```

```
*     .. Local Arrays ..
      real              SW(LDSW,ITPMT), WORK(LWK), WT(LENFW)
*     .. External Subroutines ..
      EXTERNAL          D05BYF
*     .. Executable Statements ..
*
      WRITE (NOUT,*) 'D05BYF Example Program Results'
      WRITE (NOUT,*)
      IFAIL = 0
*
      CALL D05BYF(IORDER,IQ,LENFW,WT,SW,LDSW,WORK,LWK,IFAIL)
*
      WRITE (NOUT,*) 'Fractional convolution weights'
      WRITE (NOUT,*)
      DO 20 I = 1, ITIQ
         WRITE (NOUT,99999) I - 1, WT(I)
   20 CONTINUE
      WRITE (NOUT,*)
      WRITE (NOUT,*) 'Fractional starting weights'
      WRITE (NOUT,*)
      DO 40 I = 1, LDSW
         WRITE (NOUT,99999) I - 1, (SW(I,J),J=1,ITPMT)
   40 CONTINUE
*
      STOP
*
99999 FORMAT (1X,I5,7F9.4)
      END
```

9.2 Program Data

None.

9.3 Program Results

```
 D05BYF Example Program Results

 Fractional convolution weights

     0   0.6928
     1   0.6651
     2   0.4589
     3   0.3175
     4   0.2622
     5   0.2451
     6   0.2323
     7   0.2164
     8   0.2006
     9   0.1878
    10   0.1780
    11   0.1700
    12   0.1629
    13   0.1566
    14   0.1508
    15   0.1457
```

```
Fractional starting weights

      0   0.0000   0.0000   0.0000   0.0000   0.0000   0.0000   0.0000
      1   0.0565   2.8928  -6.7497  11.6491 -11.1355   5.5374  -1.1223
      2   0.0371   1.7401  -2.8628   6.5207  -6.4058   3.2249  -0.6583
      3   0.0300   1.3207  -2.4642   6.3612  -5.4478   2.7025  -0.5481
      4   0.0258   1.1217  -2.2620   5.3683  -3.7553   2.2132  -0.4549
      5   0.0230   0.9862  -2.0034   4.5005  -3.2772   2.7262  -0.4320
      6   0.0208   0.9001  -1.8989   4.2847  -3.5881   2.8201   0.2253
      7   0.0190   0.8506  -1.9250   4.4164  -4.0181   2.7932   0.1564
      8   0.0173   0.8177  -1.9697   4.5348  -4.2425   2.7458  -0.0697
      9   0.0160   0.7886  -1.9781   4.5318  -4.2769   2.6997  -0.2127
     10   0.0149   0.7603  -1.9548   4.4545  -4.2332   2.6541  -0.2620
     11   0.0140   0.7338  -1.9198   4.3619  -4.1782   2.6059  -0.2716
     12   0.0132   0.7097  -1.8842   4.2754  -4.1246   2.5544  -0.2767
     13   0.0125   0.6880  -1.8497   4.1933  -4.0662   2.5011  -0.2845
     14   0.0119   0.6681  -1.8153   4.1109  -4.0004   2.4479  -0.2915
     15   0.0114   0.6497  -1.7805   4.0279  -3.9304   2.3962  -0.2951
     16   0.0110   0.6327  -1.7461   3.9463  -3.8598   2.3466  -0.2958
     17   0.0105   0.6168  -1.7126   3.8677  -3.7907   2.2990  -0.2950
     18   0.0102   0.6020  -1.6804   3.7926  -3.7238   2.2536  -0.2935
     19   0.0098   0.5882  -1.6495   3.7209  -3.6589   2.2101  -0.2917
     20   0.0095   0.5752  -1.6199   3.6523  -3.5961   2.1686  -0.2895
     21   0.0093   0.5631  -1.5916   3.5867  -3.5356   2.1291  -0.2871
     22   0.0090   0.5517  -1.5644   3.5240  -3.4774   2.0914  -0.2844
```