NAG Fortran Library Routine Document

D02QZF

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

D02QZF interpolates components of the solution of a non-stiff system of first-order differential equations from information provided by the integrator routines D02QFF or D02QGF.

2 Specification

```
      SUBROUTINE D02QZF(NEQF, TWANT, NWANT, YWANT, YPWANT, RWORK, LRWORK,
     1                  IWORK, LIWORK, IFAIL)
      INTEGER           NEQF, NWANT, LRWORK, IWORK(LIWORK), LIWORK, IFAIL
      real              TWANT, YWANT(NWANT), YPWANT(NWANT), RWORK(LRWORK)
```

3 Description

D02QZF evaluates the first NWANT components of the solution of a non-stiff system of first-order ordinary differential equations at any point using the method of Watts and Shampine (1986) and information generated by D02QFF or D02QGF. D02QZF should not normally be used to extrapolate outside the current range of the values produced by the integration routine.

4 References

Watts H A and Shampine L F (1986) Smoother interpolants for Adams codes *SIAM J. Sci. Statist. Comput.* **7** 334–345

5 Parameters

1: NEQF – INTEGER *Input*

On entry: the number of first-order ordinary differential equations being solved by the integration routine. It must contain the same value as the parameter NEQF in a prior call to the setup routine D02QWF.

2: TWANT – ***real*** *Input*

On entry: the point at which components of the solution and derivative are to be evaluated. TWANT should not normally be an extrapolation point, that is TWANT should satisfy

$$\text{TOLD} \leq \text{TWANT} \leq \text{T},$$

or if integration is proceeding in the negative direction

$$\text{TOLD} \geq \text{TWANT} \geq \text{T},$$

where TOLD is the previous integration point and is, to within rounding, TCURR – HLAST (see D02QXF). Extrapolation is permitted but not recommended and an IFAIL value of 2 is returned whenever extrapolation is attempted.

3: NWANT – INTEGER *Input*

On entry: the number of components of the solution and derivative whose values at TWANT are required. The first NWANT components are evaluated.

Constraint: $1 \leq \text{NWANT} \leq \text{NEQF}$.

4: YWANT(NWANT) – ***real*** array *Output*

On exit: the calculated value of the ith component of the solution at TWANT, for $i = 1, 2, \ldots, \mathrm{NWANT}$.

5: YPWANT(NWANT) – ***real*** array *Output*

On exit: the calculated value of the ith component of the derivative at TWANT, for $i = 1, 2, \ldots, \mathrm{NWANT}$.

6: RWORK(LRWORK) – ***real*** array *Workspace*

This **must** be the same parameter RWORK as supplied to D02QWF and to D02QFF or D02QGF. It is used to pass information from these routines to D02QZF. Therefore its contents **must not** be changed prior to a call to D02QZF.

7: LRWORK – INTEGER *Input*

On entry: the dimension of the array RWORK as declared in the (sub)program from which D02QZF is called.

This must be the same parameter LRWORK as supplied to D02QWF.

8: IWORK(LIWORK) – INTEGER array *Workspace*

This **must** be the same parameter IWORK as supplied to D02QWF and to D02QFF or D02QGF. It is used to pass information from these routines to D02QZF. Therefore its contents **must not** be changed prior to a call to D02QZF.

9: LIWORK – INTEGER *Input*

On entry: the dimension of the array IWORK as declared in the (sub)program from which D02QZF is called.

This must be the same parameter LIWORK as supplied to D02QWF.

10: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. Users who are unfamiliar with this parameter should refer to Chapter P01 for details.

On exit: IFAIL $= 0$ unless the routine detects an error (see Section 6).

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, for users 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.**

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 $= 1$

An integration routine (D02QFF or D02QGF) has not been called, no integration steps have been taken since the last call to D02QWF with STATEF = 'S', one or more of the parameters LRWORK, LIWORK and NEQF does not match the same parameter supplied to D02QWF, or NWANT does not satisfy $1 \le \mathrm{NWANT} \le \mathrm{NEQF}$.

IFAIL = 2

D02QZF has been called for extrapolation. The values of the solution and its derivative at TWANT have been calculated and placed in YWANT and YPWANT before returning with this warning (see Section 7).

These error exits may be caused by overwriting elements of RWORK and IWORK.

7 Accuracy

The error in interpolation is of a similar order to the error arising from the integration. The same order of accuracy can be expected when extrapolating using D02QZF. However, the actual error in extrapolation will, in general, be much larger than for interpolation.

8 Further Comments

When interpolation for only a few components is required then it is more efficient to order the components of interest so that they are numbered first.

9 Example

We solve the equation

$$y'' = -y, \quad y(0) = 0, \quad y'(0) = 1$$

reposed as

$$y_1' = y_2$$

$$y_2' = -y_1$$

over the range $[0, \pi/2]$ with initial conditions $y_1 = 0$ and $y_2 = 1$ using vector error control (VECTOL = .TRUE.) and D02QFF in one-step mode (ONESTP = .TRUE.). D02QZF is used to provide solution values at intervals of $\pi/16$.

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.

```
*     D02QZF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NOUT
      PARAMETER        (NOUT=6)
      INTEGER          NEQF, NEQG, LATOL, LRTOL, LRWORK, LIWORK
      PARAMETER        (NEQF=2,NEQG=0,LATOL=NEQF,LRTOL=NEQF,
     +                 LRWORK=23+23*NEQF+14*NEQG,LIWORK=21+4*NEQG)
      real             TSTART, HMAX
      PARAMETER        (TSTART=0.0e0,HMAX=2.0e0)
*     .. Local Scalars ..
      real             PI, T, TCRIT, TINC, TOUT, TWANT
      INTEGER          I, IFAIL, J, MAXSTP, NWANT
      LOGICAL          ALTERG, CRIT, ONESTP, ROOT, SOPHST, VECTOL
      CHARACTER*1      STATEF
*     .. Local Arrays ..
      real             ATOL(LATOL), RTOL(LRTOL), RWORK(LRWORK), Y(NEQF),
     +                 YPWANT(NEQF), YWANT(NEQF)
      INTEGER          IWORK(LIWORK)
*     .. External Functions ..
      real             D02QFZ, X01AAF
      EXTERNAL         D02QFZ, X01AAF
*     .. External Subroutines ..
      EXTERNAL         D02QFF, D02QWF, D02QZF, FTRY03
*     .. Intrinsic Functions ..
```

```
      INTRINSIC        real
*     .. Executable Statements ..
      WRITE (NOUT,*) 'D02QZF Example Program Results'
      PI = X01AAF(0.0e0)
      STATEF = 'S'
      VECTOL = .TRUE.
      DO 20 I = 1, NEQF
         ATOL(I) = 1.0e-8
         RTOL(I) = 1.0e-4
   20 CONTINUE
      ONESTP = .TRUE.
      CRIT = .TRUE.
      TINC = 0.0625e0*PI
      TCRIT = 8.0e0*TINC
      TOUT = TCRIT
      MAXSTP = 500
      T = TSTART
      TWANT = TSTART + TINC
      NWANT = NEQF
      Y(1) = 0.0e0
      Y(2) = 1.0e0
      WRITE (NOUT,*)
      WRITE (NOUT,*) '  T         Y(1)      Y(2)'
      WRITE (NOUT,99999) T, Y(1), Y(2)
      IFAIL = -1
*
      CALL D02QWF(STATEF,NEQF,VECTOL,ATOL,LATOL,RTOL,LRTOL,ONESTP,CRIT,
     +            TCRIT,HMAX,MAXSTP,NEQG,ALTERG,SOPHST,RWORK,LRWORK,
     +            IWORK,LIWORK,IFAIL)
*
      J = 1
   40 IFAIL = -1
*
      CALL D02QFF(FTRY03,NEQF,T,Y,TOUT,D02QFZ,NEQG,ROOT,RWORK,LRWORK,
     +            IWORK,LIWORK,IFAIL)
*
      IF (IFAIL.EQ.0) THEN
   60    IF (TWANT.LE.T) THEN
            IFAIL = 0
*
            CALL D02QZF(NEQF,TWANT,NWANT,YWANT,YPWANT,RWORK,LRWORK,
     +                  IWORK,LIWORK,IFAIL)
*
            WRITE (NOUT,99999) TWANT, YWANT(1), YWANT(2)
            J = J + 1
            TWANT = TSTART + real(J)*TINC
            GO TO 60
         END IF
         IF (T.LT.TOUT) GO TO 40
      END IF
      STOP
*
99999 FORMAT (1X,F6.4,3X,2(F7.4,2X))
      END
*
      SUBROUTINE FTRY03(NEQF,T,Y,YP)
*     .. Scalar Arguments ..
      real              T
      INTEGER           NEQF
*     .. Array Arguments ..
      real              Y(NEQF), YP(NEQF)
*     .. Executable Statements ..
      YP(1) = Y(2)
      YP(2) = -Y(1)
      RETURN
      END
```

9.2 Program Data

None.

9.3 Program Results

```
 D02QZF Example Program Results

   T         Y(1)     Y(2)
 0.0000    0.0000   1.0000
 0.1963    0.1951   0.9808
 0.3927    0.3827   0.9239
 0.5890    0.5556   0.8315
 0.7854    0.7071   0.7071
 0.9817    0.8315   0.5556
 1.1781    0.9239   0.3827
 1.3744    0.9808   0.1951
 1.5708    1.0000  -0.0000
```