NAG Fortran Library Routine Document

D02PXF

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

D02PXF computes the solution of a system of ordinary differential equations using interpolation anywhere on an integration step taken by D02PDF.

2 Specification

```
      SUBROUTINE D02PXF (TWANT, REQEST, NWANT, YWANT, YPWANT, F, WORK, WRKINT,
     1                   LENINT, IFAIL)
      INTEGER            NWANT, LENINT, IFAIL
      double precision   TWANT, YWANT(*), YPWANT(*), WORK(*), WRKINT(LENINT)
      CHARACTER*1        REQEST
      EXTERNAL           F
```

3 Description

D02PXF and its associated routines (D02PVF, D02PDF, D02PWF, D02PYF and D02PZF) solve the initial value problem for a first-order system of ordinary differential equations. The routines, based on Runge–Kutta methods and derived from RKSUITE (see Brankin *et al.* (1991)), integrate

$$y' = f(t,y) \quad \text{given} \quad y(t_0) = y_0$$

where y is the vector of n solution components and t is the independent variable.

D02PDF computes the solution at the end of an integration step. Using the information computed on that step D02PXF computes the solution by interpolation at any point on that step. It cannot be used if METHOD = 3 was specified in the call to setup routine D02PVF.

4 References

Brankin R W, Gladwell I and Shampine L F (1991) RKSUITE: A suite of Runge–Kutta codes for the initial value problems for ODEs *SoftReport 91-S1* Southern Methodist University

5 Parameters

1: TWANT – ***double precision*** *Input*

On entry: t, the value of the independent variable where a solution is desired.

2: REQEST – CHARACTER*1 *Input*

On entry: determines whether the solution and/or its first derivative are to be computed.

REQEST = 'S'

Compute the approximate solution only.

REQEST = 'D'

Compute the approximate first derivative of the solution only.

REQEST = 'B'

Compute both the approximate solution and its first derivative.

Constraint: REQEST = 'S', 'D' or 'B'.

3: NWANT – INTEGER *Input*

On entry: the number of components of the solution to be computed. The first NWANT components are evaluated.

Constraint: $1 \leq \text{NWANT} \leq n$, where n is specified by NEQ in the prior call to D02PVF.

4: YWANT($*$) – ***double precision*** array *Output*

Note: the dimension of the array YWANT must be at least NWANT if REQEST = 'S' or 'B' and at least 1 otherwise.

On exit: an approximation to the first NWANT components of the solution at TWANT if REQEST = 'S' or 'B'. Otherwise YWANT is not defined.

5: YPWANT($*$) – ***double precision*** array *Output*

Note: the dimension of the array YPWANT must be at least NWANT if REQEST = 'D' or 'B' and at least 1 otherwise.

On exit: an approximation to the first NWANT components of the first derivative at TWANT if REQEST = 'D' or 'B'. Otherwise YPWANT is not defined.

6: F – SUBROUTINE, supplied by the user. *External Procedure*

F must evaluate the functions f_i (that is the first derivatives y_i') for given values of the arguments t, y_i. It must be the same procedure as supplied to D02PDF.

Its specification is:

```
SUBROUTINE F (T, Y, YP)
double precision T, Y(n), YP(n)
```

where n is the value of NEQ in the call of D02PXF.

1: T – ***double precision*** *Input*

On entry: t, the current value of the independent variable.

2: Y(n) – ***double precision*** array *Input*

On entry: the current values of the dependent variables, y_i, for $i = 1, 2, \ldots, n$.

3: YP(n) – ***double precision*** array *Output*

On exit: the values of f_i, for $i = 1, 2, \ldots, n$.

F must be declared as EXTERNAL in the (sub)program from which D02PXF is called. Parameters denoted as *Input* must **not** be changed by this procedure.

7: WORK($*$) – ***double precision*** array *Input/Output*

Note: the dimension of the array WORK must be at least LENWRK (see D02PVF).

On entry: this **must** be the same array as supplied to D02PDF and **must** remain unchanged between calls.

On exit: contains information about the integration for use on subsequent calls to D02PDF or other associated routines.

8: WRKINT(LENINT) – ***double precision*** array *Input/Output*

On entry: must be the same array as supplied in previous calls, if any, and must remain unchanged between calls to D02PXF.

On exit: the contents are modified.

9: LENINT – INTEGER *Input*

On entry: the dimension of the array WRKINT as declared in the (sub)program from which D02PXF is called.

Constraints:

LENINT ≥ 1 if METHOD $= 1$ in the prior call to D02PVF;
LENINT $\geq n + 5 \times$ NWANT if METHOD $= 2$ and n is specified by NEQ in the prior call of D02PVF.

10: IFAIL – INTEGER *Input/Output*

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

On final 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, because for this routine the values of the output parameters may be useful even if IFAIL $\neq 0$ on exit, the recommended value is -1. **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 $= 1$

On entry, an invalid input value for NWANT or LENINT was detected or an invalid call to D02PXF was made, for example without a previous call to the integration routine D02PDF, or after an error return from D02PDF, or if D02PDF was being used with METHOD $= 3$. You cannot continue integrating the problem.

7 Accuracy

The computed values will be of a similar accuracy to that computed by D02PDF.

8 Further Comments

None.

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 $[0, 2\pi]$ with initial conditions $y_1 = 0.0$ and $y_2 = 1.0$. We use relative error control with threshold values of $1.0\text{D} - 8$ for each solution component. D02PDF is used to integrate the problem one step at a time and D02PXF is used to compute the first component of the solution and its derivative at intervals of length $\pi/8$ across the range whenever these points lie in one of those integration steps. We use a moderate order Runge–Kutta method ($\text{METHOD} = 2$) with tolerances $\text{TOL} = 1.0\text{D} - 3$ and $\text{TOL} = 1.0\text{D} - 4$ in turn so that we may compare the solutions. The value of π is obtained by using X01AAF.

Note that the length of WORK is large enough for any valid combination of input arguments to D02PVF and the length of WRKINT is large enough for any valid value of the parameter NWANT.

9.1 Program Text

```
*     D02PXF Example Program Text
*     Mark 17 Revised. NAG Copyright 1995.
*     .. Parameters ..
      INTEGER          NOUT
      PARAMETER        (NOUT=6)
      INTEGER          NEQ, NWANT, LENINT, LENWRK, METHOD
      PARAMETER        (NEQ=2,NWANT=1,LENINT=NEQ+5*NWANT,LENWRK=32*NEQ,
     +                 METHOD=2)
      DOUBLE PRECISION ZERO, ONE, TWO
      PARAMETER        (ZERO=0.0D0,ONE=1.0D0,TWO=2.0D0)
*     .. Local Scalars ..
      DOUBLE PRECISION HNEXT, HSTART, PI, TEND, TINC, TNOW, TOL, TSTART,
     +                 TWANT, WASTE
      INTEGER          I, IFAIL, J, L, NPTS, STPCST, STPSOK, TOTF
      LOGICAL          ERRASS
*     .. Local Arrays ..
      DOUBLE PRECISION THRES(NEQ), WORK(LENWRK), WRKINT(LENINT),
     +                 YNOW(NEQ), YPNOW(NEQ), YPWANT(NWANT),
     +                 YSTART(NEQ), YWANT(NWANT)
*     .. External Functions ..
      DOUBLE PRECISION X01AAF
      EXTERNAL         X01AAF
*     .. External Subroutines ..
      EXTERNAL         D02PDF, D02PVF, D02PXF, D02PYF, F
*     .. Executable Statements ..
      WRITE (NOUT,*) 'D02PXF Example Program Results'
*
*  Set initial conditions and input for D02PVF
*
      PI = X01AAF(ZERO)
      TSTART = ZERO
      YSTART(1) = ZERO
      YSTART(2) = ONE
      TEND = TWO*PI
      DO 20 L = 1, NEQ
         THRES(L) = 1.0D-8
   20 CONTINUE
      ERRASS = .FALSE.
      HSTART = ZERO
*
*  Set output control
*
      NPTS = 16
      TINC = TEND/NPTS
*
      DO 80 I = 1, 2
         IF (I.EQ.1) TOL = 1.0D-3
         IF (I.EQ.2) TOL = 1.0D-4
*
         IFAIL = 0
         CALL D02PVF(NEQ,TSTART,YSTART,TEND,TOL,THRES,METHOD,
     +               'Complex Task',ERRASS,HSTART,WORK,LENWRK,IFAIL)
```

```
*
         WRITE (NOUT,'(/A,E8.1)') ' Calculation with TOL = ', TOL
         WRITE (NOUT,'(/A/)') '    t         y1         y1'''
         WRITE (NOUT,'(1X,F6.3,2(3X,F8.4))') TSTART, (YSTART(L),L=1,NEQ)
*
         J = NPTS - 1
         TWANT = TEND - J*TINC
*
   40    CONTINUE
         IFAIL = -1
         CALL D02PDF(F,TNOW,YNOW,YPNOW,WORK,IFAIL)
*
         IF (IFAIL.EQ.0) THEN
   60       CONTINUE
            IF (TWANT.LE.TNOW) THEN
               IFAIL = 0
               CALL D02PXF(TWANT,'Both',NWANT,YWANT,YPWANT,F,WORK,
     +                     WRKINT,LENINT,IFAIL)
               WRITE (NOUT,'(1X,F6.3,2(3X,F8.4))') TWANT, YWANT(1),
     +           YPWANT(1)
               J = J - 1
               TWANT = TEND - J*TINC
               GO TO 60
            END IF
            IF (TNOW.LT.TEND) GO TO 40
         END IF
*
         IFAIL = 0
         CALL D02PYF(TOTF,STPCST,WASTE,STPSOK,HNEXT,IFAIL)
         WRITE (NOUT,'(/A,I6)')
     +     ' Cost of the integration in evaluations of F is', TOTF
*
   80 CONTINUE
*
      STOP
      END
      SUBROUTINE F(T,Y,YP)
*     .. Scalar Arguments ..
      DOUBLE PRECISION T
*     .. Array Arguments ..
      DOUBLE PRECISION Y(*), YP(*)
*     .. Executable Statements ..
      YP(1) = Y(2)
      YP(2) = -Y(1)
      RETURN
      END
```

9.2 Program Data

None.

9.3 Program Results

```
 D02PXF Example Program Results

 Calculation with TOL =  0.1E-02

    t         y1         y1'

  0.000     0.0000     1.0000
  0.393     0.3827     0.9239
  0.785     0.7071     0.7071
  1.178     0.9239     0.3826
  1.571     1.0000    -0.0001
  1.963     0.9238    -0.3828
  2.356     0.7070    -0.7073
  2.749     0.3825    -0.9240
  3.142    -0.0002    -0.9999
  3.534    -0.3829    -0.9238
  3.927    -0.7072    -0.7069
  4.320    -0.9239    -0.3823
  4.712    -0.9999     0.0004
  5.105    -0.9236     0.3830
  5.498    -0.7068     0.7073
  5.890    -0.3823     0.9239
  6.283     0.0004     0.9998

 Cost of the integration in evaluations of F is    68

 Calculation with TOL =  0.1E-03

    t          y1          y1'

  0.000     0.0000     1.0000
  0.393     0.3827     0.9239
  0.785     0.7071     0.7071
  1.178     0.9239     0.3827
  1.571     1.0000    -0.0000
  1.963     0.9239    -0.3827
  2.356     0.7071    -0.7071
  2.749     0.3827    -0.9239
  3.142    -0.0000    -1.0000
  3.534    -0.3827    -0.9239
  3.927    -0.7071    -0.7071
  4.320    -0.9239    -0.3827
  4.712    -1.0000     0.0000
  5.105    -0.9238     0.3827
  5.498    -0.7071     0.7071
  5.890    -0.3826     0.9239
  6.283     0.0000     1.0000

 Cost of the integration in evaluations of F is   105
```