NAG Fortran Library Routine Document

D02MVF

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

D02MVF is a setup routine which must be called by the user, prior to an integrator in Chapter D02M/N, if the DASSL implementation of Backward Differentiation Formulae (BDF) is to be used.

2 Specification

```
SUBROUTINE D02MVF(NEQMAX, NY2DIM, MAXORD, CONST, TCRIT, HMIN, HMAX, H0,
1                 MAXSTP, MXHNIL, NORM, RWORK, IFAIL)
INTEGER           NEQMAX, NY2DIM, MAXORD, MAXSTP, MXHNIL, IFAIL
real              CONST(3), TCRIT, HMIN, HMAX, H0, RWORK(50+4*NEQMAX)
CHARACTER*1       NORM
```

3 Description

An integrator setup routine must be called before the call to any integrator in this sub-chapter. The setup routine D02MVF makes the choice of the DASSL integrator and permits the user to define options appropriate to this choice.

4 References

None.

5 Parameters

1: NEQMAX – INTEGER *Input*

On entry: a bound on the maximum number of differential equations to be solved.

Constraint: NEQMAX ≥ 1.

2: NY2DIM – INTEGER *Input*

On entry: the second dimension of the array YSAVE that will be supplied to the integrator, as declared in the (sub)program from which the integrator is called.

Constraint: NY2DIM $\geq$ MAXORD $+ 3$.

3: MAXORD – INTEGER *Input*

On entry: the maximum order to be used for the BDF method. If MAXORD $= 0$ then MAXORD $= 5$ is assumed.

Constraint: $0 \leq$ MAXORD ≤ 5.

4: CONST(3) – ***real*** array *Input/Output*

On entry: values to be used to control step size choice during integration. If any CONST$(i) = 0.0$ on entry, it is replaced by its default value described below. In most cases this is the recommended setting.

CONST(1), CONST(2), and CONST(3) are factors used to bound step size changes. If the current step size h fails, then the modulus of the next step size is bounded by CONST$(l) \times |h|$. The default value of CONST(1) is 2.0. Note that the new step size may be used with a method of different

order to the failed step. If the initial step size is h, then the modulus of the step size on the second step is bounded by CONST(3) $\times |h|$. At any other stage in the integration, if the current step size is h, then the modulus of the next step size is bounded by CONST(2) $\times |h|$. The default values are 10.0 for CONST(2) and 1000.0 for CONST(3).

Constraints: the following constraints must be satisfied after any zero values have been replaced by default values:

$0.0 < \text{CONST}(1) < \text{CONST}(2) < \text{CONST}(3)$,
$\text{CONST}(2) > 1.0$,
$\text{CONST}(3) > 1.0$.

On exit: the values actually used by the routine.

5: TCRIT – ***real*** *Input*

On entry: a point beyond which integration must not be attempted. The use of TCRIT is described under the parameter ITASK in the specification for the integrator. A value, 0.0 say, must be specified even if ITASK subsequently specifies that TCRIT will not be used.

6: HMIN – ***real*** *Input*

On entry: the minimum absolute step size to be allowed. Set HMIN $= 0.0$ if this option is not required.

7: HMAX – ***real*** *Input*

On entry: the maximum absolute step size to be allowed. Set HMAX $= 0.0$ if this option is not required.

8: H0 – ***real*** *Input*

On entry: the step size to be attempted on the first step. Set H0 $= 0.0$ if the initial step size is to be calculated internally.

9: MAXSTP – INTEGER *Input*

On entry: the maximum number of steps to be attempted during one call to the integrator after which it will return with IFAIL $= 2$. Set MAXSTP $= 0$ if no limit is to be imposed.

10: MXHNIL – INTEGER *Input*

On entry: the maximum number of warnings printed (if ITRACE ≥ 0) per problem when $t + h = t$ on a step ($h =$ current step size). If MXHNIL ≤ 0, a default value of 10 is assumed.

11: NORM – CHARACTER*1 *Input*

On entry: indicates the type of norm to be used. Three options are available:

'M' maximum norm,

'A' averaged L2 norm,

'D' is the same as 'A'.

If VNORM denotes the norm of the vector v of length n, then for the averaged L2 norm

$$\text{VNORMB} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\frac{v_i}{w_i}\right)^2},$$

while for the maximum norm

$$\text{VNORM} = \max_{1\leq i\leq n}\left|\frac{v_i}{w_i}\right|.$$

If the user wishes to weight the maximum norm or the L2 norm, then RTOL and ATOL should be

scaled appropriately on input to the integrator (see under ITOL in the specification of the integrator for the formulation of the weight vector w_i from RTOL and ATOL).

Only the first character to the actual argument NORM is passed to D02MVF; hence it is permissible for the actual argument to be more descriptive, e.g., 'Maximum', 'Average L2' or 'Default' in a call to D02MVF.

Constraint: NORM must be one of 'M', 'A' or 'D'.

12: RWORK(50+4∗NEQMAX) – ***real*** array *Workspace*

This must be the same workspace array as the array RWORK supplied to the integrator. It is used to pass information from the setup routine to the integrator and therefore the contents of this array must not be changed before calling the integrator.

13: 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 $= 1$

On entry, NEQMAX < 1,
or NY2DIM $<$ MAXORD $+ 3$,
or MAXORD < 0,
or MAXORD > 5,
or invalid value for element of the array CONST,
or NORM $\neq$ 'M', 'A' or 'D'.

7 Accuracy

Not applicable.

8 Further Comments

None.

9 Example

We solve the plane pendulum problem defined by the equations:

$$\begin{aligned} x' &= u \\ y' &= v \\ u' &= -\lambda x \\ v' &= -\lambda y - 1 \\ x^2 + y^2 &= 1. \end{aligned}$$

The additional algebraic constraint $xu + yv = 0$ can be derived, and after appropriate substitution and

manipulation to avoid a singular Jacobian we solve the equations:

$$\begin{aligned} y_1' &= y_3 - y_6 y_1 \\ y_2' &= y_4 - y_6 y_2 \\ y_3' &= -y_5 y_1 \\ y_4' &= -y_5 y_2 - 1 \\ 0 &= y_1 y_3 + y_2 y_4 \\ 0 &= y_1^2 + y_2^2 - 1 \end{aligned}$$

with given initial conditions and derivatives.

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.

```
*     D02MVF Example Program Text
*     Mark 14 Release.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NOUT
      PARAMETER        (NOUT=6)
      INTEGER          NEQ, NEQMAX, NRW, NINF, NWKJAC, MAXORD, NY2DIM,
     +                 MAXSTP, MXHNIL
      PARAMETER        (NEQ=6,NEQMAX=NEQ,NRW=50+4*NEQMAX,NINF=23,
     +                 NWKJAC=NEQMAX*(NEQMAX+1),MAXORD=5,
     +                 NY2DIM=MAXORD+3,MAXSTP=5000,MXHNIL=5)
      real             H0, HMAX, HMIN
      PARAMETER        (H0=0.0e0,HMAX=0.0e0,HMIN=1.0e-10)
*     .. Local Scalars ..
      real             DUM, PI, T, TCRIT, TOUT
      INTEGER          I, IFAIL, ITASK, ITOL, ITRACE, MAXOD1
*     .. Local Arrays ..
      real             ATOL(NEQMAX), CONST(3), RTOL(NEQMAX), RWORK(NRW),
     +                 WKJAC(NWKJAC), Y(NEQMAX), YDOT(NEQMAX),
     +                 YSAVE(NEQMAX,NY2DIM)
      INTEGER          INFORM(NINF)
      LOGICAL          LDERIV(2)
*     .. External Functions ..
      real             X01AAF
      EXTERNAL         X01AAF
*     .. External Subroutines ..
      EXTERNAL         D02MVF, D02NBY, D02NGF, D02NSF, DAEJAC, DAERES,
     +                 X04ABF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'D02MVF Example Program Results'
      CALL X04ABF(1,NOUT)
      DO 20 I = 1, 3
         CONST(I) = 0.0e0
   20 CONTINUE
      ITRACE = 0
      PI = X01AAF(DUM)
      RTOL(1) = 1.0e-3
      ATOL(1) = 1.0e-6
      WRITE (NOUT,*)
      WRITE (NOUT,99999) 'Pendulum problem with relative tolerance = ',
     +  RTOL(1)
      WRITE (NOUT,99999) '                  and absolute tolerance = ',
     +  ATOL(1)
      ITOL = 1
      T = 0.0e0
      TOUT = PI
*     Set initial values and derivatives
      Y(1) = 1.e0
      Y(2) = 0.0e0
      Y(3) = 0.e0
      Y(4) = 0.e0
      Y(5) = 0.0e0
      Y(6) = 0.0e0
```

```
      YDOT(1) = Y(3) - Y(6)*Y(1)
      YDOT(2) = Y(4) - Y(6)*Y(2)
      YDOT(3) = -Y(5)*Y(1)
      YDOT(4) = -Y(5)*Y(2) - 1.e0
      YDOT(5) = -3.e0*Y(4)
      YDOT(6) = 0.0e0
      WRITE (NOUT,*)
      WRITE (NOUT,*)
     +  '  t          y1      y2      y3      y4      y5      y6'
      WRITE (NOUT,99998) T, (Y(I),I=1,6)
      ITASK = 4
      TCRIT = TOUT
      IFAIL = 0
      MAXOD1 = 0
*
      CALL D02MVF(NEQMAX,NY2DIM,MAXOD1,CONST,TCRIT,HMIN,HMAX,H0,MAXSTP,
     +            MXHNIL,'AVERAGE-L2',RWORK,IFAIL)
*
      CALL D02NSF(NEQ,NEQMAX,'ANALYTIC',NWKJAC,RWORK,IFAIL)
*
      IFAIL = -1
      LDERIV(1) = .TRUE.
      LDERIV(2) = .TRUE.
*
      CALL D02NGF(NEQ,NEQMAX,T,TOUT,Y,YDOT,RWORK,RTOL,ATOL,ITOL,INFORM,
     +            DAERES,YSAVE,NY2DIM,DAEJAC,WKJAC,NWKJAC,D02NBY,LDERIV,
     +            ITASK,ITRACE,IFAIL)
*
      IF (IFAIL.NE.0) THEN
         WRITE (NOUT,99997) 'Integration terminated with IFAIL = ',
     +     IFAIL, ' at T = ', T
      ELSE
         WRITE (NOUT,99998) T, (Y(I),I=1,NEQ)
      END IF
      STOP
*
99999 FORMAT (1X,A,1P,e7.1)
99998 FORMAT (1X,F6.4,3X,6(F8.4))
99997 FORMAT (1X,A,I2,A,F6.4)
      END
      SUBROUTINE DAERES(NEQ,T,Y,YDOT,R,IRES)
*     .. Scalar Arguments ..
      real              T
      INTEGER           IRES, NEQ
*     .. Array Arguments ..
      real              R(NEQ), Y(NEQ), YDOT(NEQ)
*     .. Executable Statements ..
      IF (IRES.EQ.-1) THEN
         R(1) = -YDOT(1)
         R(2) = -YDOT(2)
         R(3) = -YDOT(3)
         R(4) = -YDOT(4)
         R(5) = 0.0e0
         R(6) = 0.0e0
      ELSE
         R(1) = Y(3) - Y(6)*Y(1) - YDOT(1)
         R(2) = Y(4) - Y(6)*Y(2) - YDOT(2)
         R(3) = -Y(5)*Y(1) - YDOT(3)
         R(4) = -Y(5)*Y(2) - 1.e0 - YDOT(4)
         R(5) = Y(1)*Y(3) + Y(2)*Y(4)
         R(6) = Y(1)**2 + Y(2)**2 - 1.0e0
      END IF
      RETURN
      END
      SUBROUTINE DAEJAC(NEQ,T,Y,YDOT,H,D,P)
*     .. Parameters ..
      real              ONE, TWO
      PARAMETER         (ONE=1.0e0,TWO=2.0e0)
*     .. Scalar Arguments ..
      real              D, H, T
      INTEGER           NEQ
```

```
*     .. Array Arguments ..
      real                 P(NEQ,NEQ), Y(NEQ), YDOT(NEQ)
*     .. Local Scalars ..
      real                 HXD
*     .. Executable Statements ..
      HXD = H*D
      P(1,1) = (ONE+HXD*Y(6))
      P(1,3) = -HXD
      P(1,6) = HXD*Y(1)
      P(2,2) = (ONE+HXD*Y(6))
      P(2,4) = -HXD
      P(2,6) = HXD*Y(2)
      P(3,1) = HXD*Y(5)
      P(3,3) = ONE
      P(3,5) = HXD*Y(1)
      P(4,2) = HXD*Y(5)
      P(4,4) = ONE
      P(4,5) = HXD*Y(2)
      P(5,1) = -HXD*Y(3)
      P(5,2) = -HXD*Y(4)
      P(5,3) = -HXD*Y(1)
      P(5,4) = -HXD*Y(2)
      P(6,1) = -TWO*HXD*Y(1)
      P(6,2) = -TWO*HXD*Y(2)
      RETURN
      END
```

9.2 Program Data

None.

9.3 Program Results

```
 D02MVF Example Program Results

 Pendulum problem with relative tolerance = 1.0E-03
                   and absolute tolerance = 1.0E-06

   t          y1      y2      y3      y4      y5      y6
 0.0000     1.0000  0.0000  0.0000  0.0000  0.0000  0.0000
 3.1416    -0.9842 -0.1768 -0.1051  0.5852  0.5300  0.0001
```