

NAG 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 an integration method specific setup routine which must be called prior to linear algebra setup routines and integrators from the SPRINT suite of routines, if the DASSL implementation of Backward Differentiation Formulae (BDF) is to be used. Note that this method is also available, independent from the SPRINT suite, using D02NEF

2 Specification

```

SUBROUTINE D02MVF (NEQMAX, SDYSAV, MAXORD, CON, TCRIT, HMIN, HMAX, HO,
1                MAXSTP, MXHNIL, NORM, RWORK, IFAIL)

INTEGER          NEQMAX, SDYSAV, MAXORD, MAXSTP, MXHNIL, IFAIL
double precision CON(3), TCRIT, HMIN, HMAX, HO, RWORK(50+4*NEQMAX)
CHARACTER*1      NORM

```

3 Description

An integrator setup routine must be called before the call to any linear algebra setup routine or integrator from the SPRINT suite of routines in this sub-chapter. This setup routine, D02MVF, makes the choice of the DASSL integrator and permits you to define options appropriate to this choice. Alternative choices of integrator from this suite are the BDF method and the BLEND method which can be chosen by initial calls to D02NVF or D02NWF respectively.

4 References

See the D02M–N sub-chapter Introduction.

5 Parameters

- 1: NEQMAX – INTEGER *Input*
On entry: a bound on the maximum number of differential equations to be solved.
Constraint: $\text{NEQMAX} \geq 1$.
- 2: SDYSAV – INTEGER *Input*
On entry: the second dimension of the array YSAV that will be supplied to the integrator, as declared in the (sub)program from which the integrator is called (e.g., see D02NBF).
Constraint: $\text{SDYSAV} \geq \text{MAXORD} + 3$.
- 3: MAXORD – INTEGER *Input*
On entry: the maximum order to be used for the BDF method. If $\text{MAXORD} = 0$ then $\text{MAXORD} = 5$ is assumed.
Constraint: $0 \leq \text{MAXORD} \leq 5$.

- 4: CON(3) – *double precision* array *Input/Output*
- On entry:* values to be used to control step size choice during integration. If any $\text{CON}(i) = 0.0$ on entry, it is replaced by its default value described below. In most cases this is the recommended setting.
- CON(1), CON(2), and CON(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 $\text{CON}(l) \times |h|$. The default value of CON(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 $\text{CON}(3) \times |h|$. 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 $\text{CON}(2) \times |h|$. The default values are 10.0 for CON(2) and 1000.0 for CON(3).
- Constraints:*
- These constraints must be satisfied after any zero values have been replaced by default values.
- $$0.0 < \text{CON}(1) < \text{CON}(2) < \text{CON}(3);$$
- $$\text{CON}(2) > 1.0;$$
- $$\text{CON}(3) > 1.0.$$
- On exit:* the values actually to be used by the integration routine.
- 5: TCRIT – *double precision* *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 (e.g., see D02NBF). A value, 0.0 say, must be specified even if ITASK subsequently specifies that TCRIT will not be used.
- 6: HMIN – *double precision* *Input*
- On entry:* the minimum absolute step size to be allowed. Set HMIN = 0.0 if this option is not required.
- 7: HMAX – *double precision* *Input*
- On entry:* the maximum absolute step size to be allowed. Set HMAX = 0.0 if this option is not required.
- 8: H0 – *double precision* *Input*
- On entry:* the step size to be attempted on the first step. Set H0 = 0.0 if the initial step size is 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 (e.g., see D02NBF). 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 , e.g., see D02NBF) 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.
- NORM = 'M'
Maximum norm.
- NORM = 'A'
Averaged L2 norm.

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

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

while for the maximum norm

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

If you wish 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, e.g., see D02NBF).

Only the first character to the actual parameter 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 = 'M', 'A' or 'D'.

- 12: RWORK(50 + 4 × NEQMAX) – **double precision** array *Communication Array*

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. If you are unfamiliar with this parameter you should refer to Section 3.3 in the Essential Introduction 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, if you are 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 SDYSAV < MAXORD + 3,
or MAXORD < 0,
or MAXORD > 5,
or invalid value for element of the array CON,
or NORM ≠ 'M', 'A' or 'D'.

7 Accuracy

Not applicable.

8 Further Comments

None.

9 Example

This example solves 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 solves 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

```
*      D02MVF Example Program Text
*      Mark 14 Release. NAG Copyright 1989.
*      .. Parameters ..
      INTEGER                NOUT
      PARAMETER              (NOUT=6)
      INTEGER                NEQ, NEQMAX, NRW, NINF, NWKJAC, MAXORD, SDYSAV,
+      MAXSTP, MXHNIL, LDYSAV
      PARAMETER              (NEQ=6, NEQMAX=NEQ, NRW=50+4*NEQMAX, NINF=23,
+      NWKJAC=NEQMAX*(NEQMAX+1), MAXORD=5,
+      SDYSAV=MAXORD+3, MAXSTP=5000, MXHNIL=5, LDYSAV=NEQ)
      DOUBLE PRECISION       HO, HMAX, HMIN
      PARAMETER              (HO=0.0D0, HMAX=0.0D0, HMIN=1.0D-10)
*      .. Local Scalars ..
      DOUBLE PRECISION       DUM, PI, T, TCRIT, TOUT
      INTEGER                I, IFAIL, ITASK, ITOL, ITRACE, OUTCHN
*      .. Local Arrays ..
      DOUBLE PRECISION       ATOL(NEQ), CON(3), RTOL(NEQ), RWORK(NRW),
+      WKJAC(NWKJAC), Y(NEQ), YDOT(NEQ),
+      YSAV(LDYSAV, SDYSAV)
      INTEGER                INFORM(NINF)
      LOGICAL                LDERIV(2)
*      .. External Functions ..
      DOUBLE PRECISION       X01AAF
      EXTERNAL               X01AAF
*      .. External Subroutines ..
      EXTERNAL               D02MVF, D02NBY, D02NGF, D02NSF, DAEJAC, DAERES,
+      X04ABF
*      .. Executable Statements ..
      WRITE (NOUT,*) 'D02MVF Example Program Results'
      OUTCHN = NOUT
      CALL X04ABF(1, OUTCHN)
      DO 20 I = 1, 3
         CON(I) = 0.0D0
20  CONTINUE
      ITRACE = 0
      PI = X01AAF(DUM)
      RTOL(1) = 1.0D-3
      ATOL(1) = 1.0D-6
      ITOL = 1
```

```

      T = 0.0D0
      TOUT = PI
*      Set initial values and derivatives
      Y(1) = 1.D0
      Y(2) = 0.0D0
      Y(3) = 0.D0
      Y(4) = 0.D0
      Y(5) = 0.0D0
      Y(6) = 0.0D0
      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.D0
      YDOT(5) = -3.D0*Y(4)
      YDOT(6) = 0.0D0
      ITASK = 4
      TCRIT = TOUT
      IFAIL = 1

*
      CALL D02MVF(NEQMAX,SDYSAV,MAXORD,CON,TCRIT,HMIN,HMAX,HO,MAXSTP,
+               MXHNIL,'AVERAGE-L2',RWORK,IFAIL)
*
      IF (IFAIL.EQ.0) THEN
        WRITE (NOUT,*)
        WRITE (NOUT,99999)
+       'Pendulum problem with relative tolerance = ', RTOL(1)
        WRITE (NOUT,99999)
+       ' and absolute tolerance = ', ATOL(1)
        WRITE (NOUT,*)
        WRITE (NOUT,*)
+       ' t          y1          y2          y3          y4          y5          y6'
        WRITE (NOUT,99998) T, (Y(I),I=1,6)
        IFAIL = 0
        CALL D02NSF(NEQ,NEQMAX,'ANALYTIC',NWKJAC,RWORK,IFAIL)
*
        IFAIL = 1
        LDERIV(1) = .TRUE.
        LDERIV(2) = .TRUE.
*
        CALL D02NGF(NEQ,LDYSAV,T,TOUT,Y,YDOT,RWORK,RTOL,ATOL,ITOL,
+               INFORM,DAERES,YSAB,SDYSAV,DAEJAC,WKJAC,NWKJAC,
+               D02NBY,LDERIV,ITASK,ITRACE,IFAIL)
*
        IF (IFAIL.NE.0) THEN
          WRITE (NOUT,99997) IFAIL, ' at T = ', T
        ELSE
          WRITE (NOUT,99998) T, (Y(I),I=1,NEQ)
        END IF
      ELSE
        WRITE (NOUT,99996) IFAIL
      END IF

*
99999 FORMAT (1X,A,1P,E7.1)
99998 FORMAT (1X,F6.4,3X,6(F8.4))
99997 FORMAT (1X,'Integration terminated with IFAIL = ',I2,A,F6.4)
99996 FORMAT (1X,/1X,' ** D02MVF returned with IFAIL = ',I5)
      END

*
      SUBROUTINE DAERES(NEQ,T,Y,YDOT,R,IRES)
*      .. Scalar Arguments ..
      DOUBLE PRECISION T
      INTEGER          IRES, NEQ
*      .. Array Arguments ..
      DOUBLE PRECISION 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) = 0.0D0

```

```

      R(6) = 0.0D0
    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.D0 - YDOT(4)
      R(5) = Y(1)*Y(3) + Y(2)*Y(4)
      R(6) = Y(1)**2 + Y(2)**2 - 1.0D0
    END IF
  RETURN
END
SUBROUTINE DAEJAC(NEQ,T,Y,YDOT,H,D,P)
* .. Parameters ..
DOUBLE PRECISION ONE, TWO
PARAMETER      (ONE=1.0D0,TWO=2.0D0)
* .. Scalar Arguments ..
DOUBLE PRECISION D, H, T
INTEGER          NEQ
* .. Array Arguments ..
DOUBLE PRECISION P(NEQ,NEQ), Y(NEQ), YDOT(NEQ)
* .. Local Scalars ..
DOUBLE PRECISION 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	-0.9856	-0.1694	-0.0986	0.5736	0.5080	0.0000

