

NAG Library Routine Document

D02ZAF

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

D02ZAF calculates the weighted norm of the local error estimate from inside a MONITR called from an integrator in sub-chapter D02M–N (e.g., see D02NBF).

2 Specification

```
double precision FUNCTION D02ZAF(NEQ, V, W, IFAIL)
INTEGER                                NEQ, IFAIL
double precision                    V(NEQ), W(NEQ)
```

3 Description

D02ZAF is for use with the forward communication integrators D02NBF, D02NCF, D02NDF, D02NGF, D02NHF and D02NJF and the reverse communication integrators D02NMF and D02NNF. It must be used only inside MONITR (if this option is selected) for the forward communication routines or on the equivalent return for the reverse communication routines. It may be used to evaluate the norm of the scaled local error estimate, $\|v\|$, where the weights used are contained in w and the norm used is as defined by an earlier call to the integrator setup routine (D02MVF, D02NVF or D02NWF). Its use is described under the description of MONITR in the specifications for the forward communication integrators mentioned above.

4 References

None.

5 Parameters

- 1: NEQ – INTEGER *Input*
On entry: the number of differential equations, as defined for the integrator being used.
- 2: V(NEQ) – **double precision** array *Input*
On entry: the vector, the weighted norm of which is to be evaluated by D02ZAF. V is calculated internally by the integrator being used.
- 3: W(NEQ) – **double precision** array *Input*
On entry: the weights, calculated internally by the integrator, to be used in the norm evaluation.
- 4: 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, 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).

Note: D02ZAF may return useful information for one or more of the following detected errors or warnings.

Errors or warnings detected by the routine:

IFAIL = 1

The value of the norm would either overflow or is close to overflowing. A value close to the square root of the largest number on the computer is returned.

7 Accuracy

The result is calculated close to *machine precision* except in the case when the routine exits with IFAIL = 1.

8 Further Comments

D02ZAF should only be used within MONITR associated with the integrators in sub-chapter D02M–N (e.g., see D02NBF). Its use and only valid calling sequence are fully documented in the description of MONITR in the routine documents for the integrators.

9 Example

This example solves the well-known stiff Robertson problem

$$\begin{aligned} a' &= -0.04a + 1.0E4bc \\ b' &= 0.04a - 1.0E4bc - 3.0E7b^2 \\ c' &= 3.0E7b^2 \end{aligned}$$

over the range $[0, 10]$ with initial conditions $a = 1.0$ and $b = c = 0.0$ using scalar error control (ITOL = 1) and computation of the solution at TOUT = 10.0 with TCRIT (e.g., see D02MVF) set to 10.0 (ITASK = 4). A BDF integrator (setup routine D02NVF) is used and a modified Newton method is selected. This example illustrates the use of D02ZAF within a monitor routine MONITR to output intermediate results during the integration. The same problem is solved in the example program for D02NBF where no monitoring was performed and so no intermediate solution information is output.

9.1 Program Text

```
*      D02ZAF Example Program Text
*      Mark 22 Release. NAG Copyright 2008.
*      .. Parameters ..
      INTEGER                NOUT
      PARAMETER              (NOUT=6)
      INTEGER                NEQ, NEQMAX, NRW, NINF, NWKJAC, MAXORD, SDYSAV,
+      MAXSTP, MXHNIL, LDYSAV
      PARAMETER              (NEQ=3, NEQMAX=NEQ, NRW=50+4*NEQMAX, NINF=23,
+      NWKJAC=NEQMAX*(NEQMAX+1), MAXORD=5,
+      SDYSAV=MAXORD+1, MAXSTP=200, MXHNIL=5, LDYSAV=NEQ)
      DOUBLE PRECISION       HO, HMAX, HMIN
      PARAMETER              (HO=0.0D0, HMAX=10.0D0, HMIN=1.0D-10)
      LOGICAL                PETZLD
      PARAMETER              (PETZLD=.FALSE.)
*      .. Local Scalars ..
      DOUBLE PRECISION       H, HU, T, TCRIT, TCUR, TOLSF, TOUT
      INTEGER                I, IFAIL, IMXER, ITASK, ITOL, ITRACE, NITER, NJE,
```

```

+          NQ, NQU, NRE, NST, OUTCHN
*
* .. Local Arrays ..
  DOUBLE PRECISION ATOL(NQ), CONST(6), RTOL(NQ), RWORK(NRW),
+          WKJAC(NWKJAC), Y(NQ), YDOT(NQ),
+          YSAV(LDYSAV,SDYSAV)
  INTEGER          INFORM(NINF)
  LOGICAL          ALGEQU(NQ)
*
* .. External Subroutines ..
  EXTERNAL          D02NBF, D02NSF, D02NVF, D02NYF, FCN, JAC, MONITR,
+          X04ABF
*
* .. Executable Statements ..
  WRITE (NOUT,*) 'D02ZAF Example Program Results'
  OUTCHN = NOUT
  CALL X04ABF(1,OUTCHN)
*
*
*   Set problem parameters
*
  T = 0.0D0
  TOUT = 10.0D0
*
*           Initialisation
*   Integrate to TOUT without passing TOUT.
*
  TCRIT = TOUT
  ITASK = 4
*
*   Use default values for the array CONST.
*
  DO 20 I = 1, 6
    CONST(I) = 0.0D0
20 CONTINUE
*
*   Use BDF formulae with modified Newton method.
*   Use averaged L2 norm for local error control.
*
  IFAIL = 1
  CALL D02NVF(NEQMAX,SDYSAV,MAXORD,'Newton',PETZLD,CONST,TCRIT,HMIN,
+          HMAX,HO,MAXSTP,MXHNIL,'Average-L2',RWORK,IFAIL)
  IF (IFAIL.NE.0) THEN
    WRITE (NOUT,99993) IFAIL
    GO TO 40
  END IF
*
*   Setup for using analytic Jacobian
*
  IFAIL = 0
  CALL D02NSF(NQ,NEQMAX,'Analytical',NWKJAC,RWORK,IFAIL)
*
  WRITE (NOUT,*)
  WRITE (NOUT,*) '   Analytic Jacobian'
  WRITE (NOUT,*)
*
*   Set tolerances.
*
  ITOL = 1
  RTOL(1) = 1.0D-4
  ATOL(1) = 1.0D-7
*
*   Initial values for Y.
*
  Y(1) = 1.0D0
  Y(2) = 0.0D0
  Y(3) = 0.0D0
  WRITE (NOUT,*) '      X          Y(1)          Y(2)          Y(3)'
  WRITE (NOUT,99999) T, (Y(I),I=1,NQ)
*
*   Solve the problem using MONITR to output intermediate results.
*
  ITRACE = 0
  IFAIL = 1
*

```

```

      CALL D02NBF(NEQ,LDYSAV,T,TOUT,Y,YDOT,RWORK,RTOL,ATOL,ITOL,INFORM,
+      FCN,YSAB,SDYSAV,JAC,WKJAC,NWKJAC,MONITR,ITASK,ITRACE,
+      IFAIL)
*
      IF (IFAIL.EQ.0) THEN
*
*       Get integration statistics.
*
        IFAIL = 0
        CALL D02NYF(NEQ,NEQMAX,HU,H,TCUR,TOLSF,RWORK,NST,NRE,NJE,NQU,
+      NQ,NITER,IMXER,ALGEQU,INFORM,IFAIL)
*
        WRITE (NOUT,*)
        WRITE (NOUT,99997) HU, H, TCUR
        WRITE (NOUT,99996) NST, NRE, NJE
        WRITE (NOUT,99995) NQU, NQ, NITER
        WRITE (NOUT,99994) ' Max Err Comp = ', IMXER
        WRITE (NOUT,*)
      ELSE
        WRITE (NOUT,*)
        WRITE (NOUT,99998) 'Exit D02NBF with IFAIL = ', IFAIL,
+      ' and T = ', T
      END IF
40 CONTINUE
*
99999 FORMAT (1X,F10.6,3(F13.7,2X))
99998 FORMAT (1X,A,I2,A,E12.5)
99997 FORMAT (1X,' HUSED = ',E12.5,' HNEXT = ',E12.5,' TCUR = ',E12.5)
99996 FORMAT (1X,' NST = ',I6,' NRE = ',I6,' NJE = ',I6)
99995 FORMAT (1X,' NQU = ',I6,' NQ = ',I6,' NITER = ',I6)
99994 FORMAT (1X,A,I4)
99993 FORMAT (1X,/1X,' ** D02NBF returned with IFAIL = ',I5)
      END
*
      SUBROUTINE FCN(NEQ,T,Y,R,IRES)
*
*       .. Scalar Arguments ..
      DOUBLE PRECISION T
      INTEGER          IRES, NEQ
*
*       .. Array Arguments ..
      DOUBLE PRECISION R(NEQ), Y(NEQ)
*
*       .. Executable Statements ..
      R(1) = -0.04D0*Y(1) + 1.0D4*Y(2)*Y(3)
      R(2) = 0.04D0*Y(1) - 1.0D4*Y(2)*Y(3) - 3.0D7*Y(2)*Y(2)
      R(3) = 3.0D7*Y(2)*Y(2)
      RETURN
      END
*
      SUBROUTINE JAC(NEQ,T,Y,H,D,P)
*
*       .. Scalar Arguments ..
      DOUBLE PRECISION D, H, T
      INTEGER          NEQ
*
*       .. Array Arguments ..
      DOUBLE PRECISION P(NEQ,NEQ), Y(NEQ)
*
*       .. Local Scalars ..
      DOUBLE PRECISION HXD
*
*       .. Executable Statements ..
      HXD = H*D
      P(1,1) = 1.0D0 - HXD*(-0.04D0)
      P(1,2) = -HXD*(1.0D4*Y(3))
      P(1,3) = -HXD*(1.0D4*Y(2))
      P(2,1) = -HXD*(0.04D0)
      P(2,2) = 1.0D0 - HXD*(-1.0D4*Y(3)-6.0D7*Y(2))
      P(2,3) = -HXD*(-1.0D4*Y(2))
*
*       Do not need to set P(3,1) since Jacobian preset to zero
*
      P(3,1) = - HXD*(0.0E0)
      P(3,2) = -HXD*(6.0D7*Y(2))
      P(3,3) = 1.0D0 - HXD*(0.0D0)
      RETURN
      END
*
      SUBROUTINE MONITR(NEQ,LDYSAV,T,HLAST,HNEXT,Y,YDOT,YSAB,R,ACOR,

```

```

+          IMON,INLN,HMIN,HMAX,NQU)
*
*   Fortran Library Implementation Wrapper routine for D02NBYN.
*   NAG COPYRIGHT 2004.
*
*   .. Parameters ..
INTEGER          NOUT
PARAMETER        (NOUT=6)
*   .. Scalar Arguments ..
DOUBLE PRECISION HLAST, HMAX, HMIN, HNEXT, T
INTEGER          IMON, INLN, LDYSAV, NEQ, NQU
*   .. Array Arguments ..
DOUBLE PRECISION ACOR(NEQ,*), R(*), Y(*), YDOT(*), YSAV(LDYSAV,*)
*   .. External Functions ..
DOUBLE PRECISION D02ZAF
EXTERNAL         D02ZAF
*   .. Local Scalars ..
DOUBLE PRECISION ERRLOC
INTEGER          I, IFAIL
*   .. Executable Statements ..
CONTINUE
IF (IMON.EQ.1) THEN
  IFAIL = 1
  ERRLOC = D02ZAF(NEQ,ACOR(1,2),ACOR(1,1),IFAIL)
  IF (IFAIL.NE.0) THEN
    IMON = -2
  ELSE IF (ERRLOC.GT.5.0D0) THEN
    WRITE (NOUT,99999) T, (Y(I),I=1,NEQ), ERRLOC
  ELSE
    WRITE (NOUT,99998) T, (Y(I),I=1,NEQ)
  END IF
END IF
*
RETURN
*
99999 FORMAT (1X,F10.6,3(F13.7,2X),/1X,' ** WARNING scaled local error',
+          ' = ',F13.5)
99998 FORMAT (1X,F10.6,3(F13.7,2X))
END

```

9.2 Program Data

None.

9.3 Program Results

D02ZAF Example Program Results

Analytic Jacobian

X	Y(1)	Y(2)	Y(3)
0.000000	1.0000000	0.0000000	0.0000000
0.000108	0.9999957	0.0000042	0.0000001
0.000215	0.9999914	0.0000083	0.0000003
0.000305	0.9999878	0.0000115	0.0000007
0.000394	0.9999842	0.0000145	0.0000013
0.000550	0.9999780	0.0000193	0.0000027
0.000707	0.9999717	0.0000234	0.0000049
0.000863	0.9999655	0.0000267	0.0000078
0.001134	0.9999546	0.0000308	0.0000145
0.001338	0.9999465	0.0000328	0.0000207
0.001541	0.9999384	0.0000342	0.0000274
0.001744	0.9999302	0.0000351	0.0000347
0.002024	0.9999190	0.0000357	0.0000453
0.002304	0.9999078	0.0000360	0.0000561
0.002584	0.9998966	0.0000362	0.0000671
0.002865	0.9998855	0.0000363	0.0000782
0.003252	0.9998700	0.0000364	0.0000936
0.003639	0.9998545	0.0000365	0.0001090
0.004026	0.9998390	0.0000365	0.0001245

0.005346	0.9997864	0.0000365	0.0001772
0.006665	0.9997337	0.0000365	0.0002298
0.011496	0.9995413	0.0000364	0.0004223
0.016328	0.9993492	0.0000364	0.0006144
0.027384	0.9989107	0.0000363	0.0010529
0.038440	0.9984742	0.0000362	0.0014896
0.049496	0.9980395	0.0000362	0.0019243
0.090362	0.9964493	0.0000359	0.0035149
0.131228	0.9948843	0.0000356	0.0050802
0.172093	0.9933438	0.0000353	0.0066209
0.256544	0.9902354	0.0000348	0.0097299
0.340995	0.9872231	0.0000342	0.0127427
0.425446	0.9843009	0.0000337	0.0156653
0.509897	0.9814639	0.0000332	0.0185029
0.647901	0.9769992	0.0000325	0.0229684
0.785904	0.9727312	0.0000318	0.0272371
0.923908	0.9686436	0.0000311	0.0313253
1.061912	0.9647221	0.0000305	0.0352474
1.315223	0.9579148	0.0000294	0.0420558
1.568533	0.9515559	0.0000285	0.0484156
1.821844	0.9455915	0.0000276	0.0543809
2.075154	0.9399766	0.0000268	0.0599966
2.328465	0.9346727	0.0000261	0.0653013
2.707880	0.9272386	0.0000251	0.0727363
3.087296	0.9203375	0.0000242	0.0796383
3.466712	0.9138972	0.0000234	0.0860794
3.846128	0.9078595	0.0000227	0.0921178
4.225543	0.9021762	0.0000220	0.0978018
4.812256	0.8939921	0.0000211	0.1059867
5.398969	0.8864397	0.0000203	0.1135400
5.985682	0.8794262	0.0000196	0.1205542
6.572395	0.8728779	0.0000190	0.1271031
7.159109	0.8667346	0.0000184	0.1332470
8.060884	0.8579697	0.0000176	0.1420126
8.962660	0.8499047	0.0000169	0.1500784
9.481330	0.8455416	0.0000166	0.1544418
10.000000	0.8413577	0.0000162	0.1586261

HUSED = 0.51867E+00 HNEXT = 0.51867E+00 TCUR = 0.10000E+02
NST = 55 NRE = 81 NJE = 17
NQU = 3 NQ = 3 NITER = 79
Max Err Comp = 3
