User's Guide For DVERK -
a Subroutine for Solving Non-Stiff ODE's
T. E. Hull, W. H. Enright, K. R. Jackson

Technical Report No. 100
October, 1976

Department of Computer Science
University of Toronto
User's Guide for DVERK -
a Subroutine for Solving Non-stiff ODE's

T. E. Hull, W. H. Enright, K. R. Jackson

Summary

This document explains how to use DVERK, a double-precision subroutine for solving systems of first order ordinary differential equations. (The subroutine is based on Runge-Kutta formulas of orders 5 and 6 that were developed by Verner (4).)

The main purpose of the next two sections is to illustrate the use of the subroutine in a variety of different situations. The user is expected to refer to the first parts of the listing at the end of this report for detailed instructions on how to use the subroutine.

The first of the next two sections is devoted to the "basic use" of the subroutine, i.e., without options. The second illustrates the use of a variety of options. Then there follows a short section on when not to use the subroutine, and what alternatives should be considered instead. Finally, there is a brief section on machine dependencies, in case the program is to be modified for a machine other than the IBM 360/370, and a statement about how copies of the program and test driver may be obtained.
Basic Use

As can be seen from the listing, the arguments in the calling sequence are as follows:

N, FCN, X, Y, XEND, TOL, IND, C, NW, W

The first six items (N through TOL) are needed to describe the problem and must of course be specified by the user. If the user does not wish to select any of the available options, he need then do only the following: set IND=1, declare C and W, and specify NW.

The user specifies the initial values of X and Y, and the purpose of the subroutine is to "update" these values, i.e., to replace the value of X by the value of XEND, and the value of Y by its computed approximation to the value of Y at XEND. To have the calculations continued to a new value of XEND, the user need only specify the new value of XEND and re-enter the subroutine. This is illustrated in figure 1, where a program for printing out a table of solutions at equally spaced values of X is presented.

---

1 A tape containing DVERK, the test subroutines, the subroutines FCN1 and FCN2 that they need, and a driver program may be obtained from the authors, as explained near the end of the report. The driver program as well as the subroutines FCN1 and FCN2 are also listed there.

2 To condense listings in this report, we deviate from ANSI Fortran and use semi-colons as statement separators, but this device is not used on the tape.

3 The comment about checking IND is a reminder that this is the point where the user should test for error returns and make provision for whatever action is appropriate.
SUBROUTINE TEST1
INTEGER N, NW, IND, K
DOUBLE PRECISION X, XEND, Y(2), TOL, C(24), W(2,9)
EXTERNAL FCNI

C C
C Initialization for the predator-prey problem
C N = 2 ; X = 0.00 ; Y(1) = 1.00 ; IND = 1
C NW = 2 ; TOL = 1.0E-6 ; Y(2) = 3.00
C C
C Output heading and initial values
WRITE(6,1) X, Y(1), Y(2)
C 1 FORMAT(1H1, 17HOUTPUT FROM TEST1
C + / 1H0, 5X, 3H X, 5X, 4HY(1), 11X, 4HY(2)
C + / 1H0, 5X, F3.0, 1P2015,6)
C C
C Calculate and output the solution at X = 1, 2, ..., 10
DO 10 K = 1, 10
  XEND = DFLOAT(K)
  CALL OVRK(N, FCNI, X, Y, XEND, TOL, IND, C, NW, W)
  CHECK IND = EQ. 1 - OTHERWISE TAKE APPROPRIATE ACTION
  WRITE(6,2) X, Y(1), Y(2)
C 2 FORMAT(1H0, 5X, F3.0, 1P2015,6)
C 10 CONTINUE
C C
RETURN
END

OUTPUT FROM TEST1

<table>
<thead>
<tr>
<th>X</th>
<th>Y(1)</th>
<th>Y(2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.000</td>
<td>1.000000E+00</td>
<td>3.000000E+00</td>
</tr>
<tr>
<td>1.000</td>
<td>7.345283E-02</td>
<td>1.464445E+00</td>
</tr>
<tr>
<td>2.000</td>
<td>8.497744E-02</td>
<td>5.779538E-01</td>
</tr>
<tr>
<td>3.000</td>
<td>2.906960E-01</td>
<td>2.492533E-01</td>
</tr>
<tr>
<td>4.000</td>
<td>1.446599E+00</td>
<td>1.872189E-01</td>
</tr>
<tr>
<td>5.000</td>
<td>4.051460E+00</td>
<td>1.439479E+00</td>
</tr>
<tr>
<td>6.000</td>
<td>1.756181E-01</td>
<td>2.258599E+00</td>
</tr>
<tr>
<td>7.000</td>
<td>6.531117E-02</td>
<td>9.086004E-01</td>
</tr>
<tr>
<td>8.000</td>
<td>1.472268E+01</td>
<td>3.667180E-01</td>
</tr>
<tr>
<td>9.000</td>
<td>6.505960E-01</td>
<td>1.875748E+00</td>
</tr>
<tr>
<td>10.0</td>
<td>3.144327E+00</td>
<td>3.488260E-01</td>
</tr>
</tbody>
</table>

Figure 1 A table of solutions, at equally spaced values of X, is obtained for the predator-prey problem: 
\[ Y'_1 = 2Y_1(1-Y_2), \quad Y'_2 = Y_2(1-Y_1), \quad Y_1(0) = 1, \quad Y_2(0) = 3. \] (See footnotes on page 2.)
(A remark about NW might be appropriate at this stage. If only one system of equations is being solved, NW normally will have the same value as N. However, if more than one system is being handled, and they are to use a common workspace W, one after the other, the value of NW must be as large as the maximum value of the individual N's.)

The user should note that the subroutine may return to the calling program, if it runs into difficulties before reaching XEND. In such cases the value of IND is made negative, so that the calling program can test for such circumstances and take appropriate action. There are other situations (including re-entry with a negative value of IND) which leave the subroutine no reasonable course of action except to print out current information about the status of the calculation and then to stop. Further details about returns with negative IND, or simply stopping the calculation altogether, are given at the end of the part of the listing entitled USE.

We conclude this section on "basic use" with a brief discussion of the tolerance parameter TOL. The subroutine attempts to provide an approximate solution at XEND whose error is proportional to TOL. In other words, it is intended that the global error be proportional to TOL. The proportionality factor depends on the differential equations and the range of integration. (The proportionality can also be affected by the error control chosen by the user, if the user
should decide to choose an option other than the default provided by the subroutine.

The proportionality between global error and tolerance is illustrated in figure 2, where approximate solutions to a particular problem with a sequence of different values of TOL are presented. The norm of the error, and the proportionality factor have also been computed for each value of TOL to show how the proportionality can be quite steady. (As one might expect, the proportionality tends to be constant for the smaller values of TOL.)

We can think of TOL as being the "accuracy parameter". Making TOL smaller improves the accuracy. And the global error is kept, as closely as possible, proportional to TOL. Thus, more than one run, with different values of TOL, can be used in an attempt to estimate the global error.

The Use of Options

The part of the listing entitled OPTIONS explains in detail what is available to the user. Here we will only consider a number of examples.

If the user wishes to select some particular options at the beginning of a calculation, he proceeds exactly as with the "basic use" of the subroutine except that he sets IND=2 (rather than 1) and he must then assign values to C(1) through C(9). (With two of the error control options he will also have to lengthen the vector C, and assign values to C(31), C(32), ..., C(N+30).)
Figure 2 First, the "true" solution, YTRUE, to the predator-prey problem is computed by using DVERK with TOL=10^-12. Then the approximate solutions with TOL=10^-1, 10^-2, ..., 10^-9 are computed and output along with the max norm of the error vector and the ratio of this norm to TOL. This ratio, which is the proportionality factor between the global error and TOL, is roughly constant (especially for the smaller values of TOL).
The user would normally set the values of C(1) through C(9) to zero, which provides default values for the corresponding options, and then change only those values which correspond to the options he wants. In figures 3A and 3B we show two examples, one in which absolute error control has been selected, and another in which relative error control has been selected for the same problem.

Figures 4A and 4B illustrate the use of still another error control option. This time, different "floor values" are assigned to different components of the solution. (These examples also illustrate the way in which the vector C has to be extended to provide the individual floor values.)

The use of C(3), C(4) and C(6) to specify HMIN, HSTART and HMAX, respectively, is relatively straightforward, and both C(4) and C(6) do appear in later examples. We will therefore concentrate on a discussion of C(5), which corresponds to the optional specification of SCALE. By way of background for explaining the use of this parameter, we point out that the main reason for having HMAX, a bound on the magnitude of the step-size, is to ensure the reliability of a calculation. However, there is an objection to having the user specify HMAX for this purpose, and that is that the appropriate choice of HMAX depends on both the problem and the method. The user can be expected to provide information about his problem, but it is not so reasonable to expect him to be familiar with the method. The appropriate choice of HMAX for a particular problem will in general be different for different methods.
SUBROUTINE TEST3A
INTEGER N, NW, IND, K
DOUBLE PRECISION X, XEND, Y(1), TOL, C(24), W(1,9), RELERR
DATA C(1), C(2), C(3), C(4), C(5), C(6), C(7), C(8), C(9) /9=0.0D0/
EXTERNAL FCN2
C
CALCULATE THE SOLUTION TO THE PROBLEM Y' = Y, Y = 1 AT X = 0
C USING THE ABSOLUTE ERROR CONTROL OPTION
N = 1 ; X = 0.0D0 ; IND = 2 ; TOL = 1.0D-6
NW = 1 ; Y(1) = 1.0D0 ; C(1) = 1.0D0
C
OUTPUT HEADING
WRITE(6,1)
+ / 1H0, 20X, 22MAESOLUTE ERROR CONTROL
+ / 1H0, 5X, 3H X , 5X, 4HY(1) , 9X, 12HRELATIVE ERR, 5X,
+      12HNO FCN EVALS / )
C
CALCULATE AND OUTPUT THE SOLUTION, ITS RELATIVE ERROR AND THE
NUMBER OF FUNCTION EVALUATIONS USED AT X = 5, 10, ..., 50
C
LOOP
10 XEND = X + 5.0D0
    CALL DVERK(N, FCN2, X, Y, XEND, TOL, IND, C, NW, W)
    IF (IND .EQ. 3) GO TO 20
    WRITE(6,2) IND, X, Y(1)
2    FORMAT(1H0, 23HERROR RETURN WITH IND =, I3, 7H AT X =,
+      F7.3, 5H, Y =, 1P013.5)

.....EXIT LOOP
GO TO 30
20 CONTINUE
    RELERR = Y(1)/DEXP(X) - 1.0D0
    WRITE(6,3) X, Y(1), RELERR, C(24)
3    FORMAT(1H0, 5X, F3.0, 1P215.6, OPF12.0)
    IF (XEND .LT. 50.0D0) GO TO 10
C
END LOOP
30 CONTINUE
C
RETURN
END

OUTPUT FROM TEST3A

ABSOLUTE ERROR CONTROL

<table>
<thead>
<tr>
<th>X</th>
<th>Y(1)</th>
<th>RELATIVE ERR</th>
<th>FCN EVALS</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.</td>
<td>1.4841320E+02</td>
<td>-2.457570D-08</td>
<td>184.</td>
</tr>
<tr>
<td>10.</td>
<td>2.026470E+04</td>
<td>-2.471737D-08</td>
<td>616.</td>
</tr>
<tr>
<td>15.</td>
<td>3.2590170E+06</td>
<td>-2.471816D-08</td>
<td>1632.</td>
</tr>
</tbody>
</table>

ERROR RETURN WITH IND = -3 AT X = 16.212, Y = 1.0984330D+07

Figure 3A Absolute error control is used with the problem Y' = Y, Y(0)=1 and a tolerance of TOL=10^-6. DVERK was unable to satisfy this error requirement after X=16.212, and returned to the calling program with IND=-3. (The no. of function evaluations is available to the user in C(24).)
SURROUNTE TEST38

INTEGER N, NW, IND, K
DOUBLE PRECISION X, XEND, Y(1), TOL, C(24), W(1, 9), RELERR
DATA C(1), C(2), C(3), C(4), C(5), C(6), C(7), C(8), C(9) /9*0.D0/
EXTERNAL FCN2

C CALCULATE THE SOLUTION TO THE PROBLEM Y' = Y, Y = 1 AT X = 0
C USING THE RELATIVE ERROR CONTROL OPTION
C N = 1: X = 0.00: IND = 2: TOL = 1.0D-6
C NW = 1: Y(1) = 1.00: C(1) = 2.00
C

C OUTPUT HEADING
C WRITE(6,1) 1 FORMAT(1H1, 15HOUTPUT FROM TEST38 )
(+ /1HO: 20X, 22HRELATIVE ERROR CONTROL
(+ /1HO: 5X, 3H X, 5X, 4HY(1), 9X, 12HRELATIVE ERR, 5X,
(+ 12HNO FCN EVALS /
C
C CALCULATE AND OUTPUT THE SOLUTION, ITS RELATIVE ERROR AND THE
C NUMBER OF FUNCTION EVALUATIONS USED AT X = 5, 10, ..., 50
C DO 10 K = 5, 50, 5
C XEND = DFLOAT(K)
C CALL DVERK(N, FCN2, X, Y, XEND, TOL, IND, C, NW, W)
C CHECK IND = EQ. 3 - OTHERWISE TAKE APPROPRIATE ACTION
C RELEER = Y(1)/DEXP(X) - 1.00
C WRITE(6,2) X, Y(1), RELEER, C(24)
C 2 FORMAT(1H, 5X, F3.0, 1P2015.6, 0PF12.0)
C 10 CONTINUE
C
C RETURN
C END

OUTPUT FROM TEST38

RELATIVE ERROR CONTROL

<table>
<thead>
<tr>
<th>X</th>
<th>Y(1)</th>
<th>RELATIVE ERR</th>
<th>NO FCN EVALS</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>1.4841311D+02</td>
<td>-9.0521120-08</td>
<td>128</td>
</tr>
<tr>
<td>10</td>
<td>2.2026460D+04</td>
<td>-1.8630180-07</td>
<td>236</td>
</tr>
<tr>
<td>15</td>
<td>3.260150D+06</td>
<td>-2.8168220-07</td>
<td>386</td>
</tr>
<tr>
<td>20</td>
<td>4.851650D+08</td>
<td>-3.7742550-07</td>
<td>512</td>
</tr>
<tr>
<td>25</td>
<td>7.2004870D+10</td>
<td>-4.7298870-07</td>
<td>640</td>
</tr>
<tr>
<td>30</td>
<td>1.0686470D+13</td>
<td>-5.6855180-07</td>
<td>768</td>
</tr>
<tr>
<td>35</td>
<td>1.5860120D+15</td>
<td>-6.6411490-07</td>
<td>896</td>
</tr>
<tr>
<td>40</td>
<td>2.353851D+17</td>
<td>-7.5967800-07</td>
<td>1024</td>
</tr>
<tr>
<td>45</td>
<td>3.453424D+19</td>
<td>-8.5524120-07</td>
<td>1152</td>
</tr>
<tr>
<td>50</td>
<td>5.1847010D+21</td>
<td>-9.5080430-07</td>
<td>1280</td>
</tr>
</tbody>
</table>

Figure 3B Relative error control is used with the same problem that was considered in figure 3A. This time DVERK is able to complete the integration to X=50. As one would expect, fewer function evaluations are needed than over the corresponding intervals in figure 3A; in fact, with relative error control for this particular problem, the same number are needed for each interval of equal size.
SUBROUTINE TEST4A
INTEGER N, NW, IND, K
DOUBLE PRECISION X, XEND, Y(2), YTRUE(2), TOL, C(32), W(2,9),
+ ERR(2)
DATA C(1), C(2), C(3), C(4), C(5), C(6), C(7), C(8), C(9) /9*0.0D0/
+ EXTERNAL FCNI
C
C CALCULATE AN ACCURATE SOLUTION TO THE PREDATOR-PREY PROBLEM
C USING THE RELATIVE ERROR CONTROL OPTION
C
N = 2 ; X = 0.0D0 ; YTRUE(1) = 1.0D0
NW = 2 ; XEND = 20.0D0 ; YTRUE(2) = 7.0D0
IND = 2 ; TOL = 1.0-12 ; C(1) = 2.0D0
C
CALL DVERK(N, FCNI, X, YTRUE, XEND, TOL, IND, C, NW, W)
CHECK IND .EQ. 3 - OTHERWISE TAKE APPROPRIATE ACTION
C
C CALCULATE AN APPROXIMATE SOLUTION TO THE PREDATOR-PREY PROBLEM
C USING THE DEFAULT ERROR CONTROL OPTION
C
X = 0.0D0 ; Y(1) = 1.0D0 ; IND = 1
TOL = 1.0-3 ; Y(2) = 7.0D0
C
CALL DVERK(N, FCNI, X, Y, XEND, TOL, IND, C, NW, W)
CHECK IND .EQ. 3 - OTHERWISE TAKE APPROPRIATE ACTION
C
C CALCULATE THE ERRORS AND PRINT THE RESULTS
C
ERR(1) = YTRUE(1) - Y(1)
ERR(2) = YTRUE(2) - Y(2)
WRITE(6,1) Y(1), ERR(1), Y(2), ERR(2)
1 FORMAT(1H1, 15HOUTPUT FROM TEST4A
+ / 1H0, 7X, 4HY(1), 11X, 6HERR(1), 8X, 4HY(2), 11X, 6HERR(2)
+ / 1H0, 2(1P017.6, 1P012.3)
C
RETURN
END

OUTPUT FROM TEST4A

Y(1)    ERR(1)    Y(2)    ERR(2)
9.078123D-03 -8.909D-03  5.5825552D+00 -3.786D+00

Figure 4A The predator-prey problem is solved once more, but with different initial conditions: \textit{Y}_1(0)=1, \textit{Y}_2(0)=7. The value of \textit{Y}_1 is very small over significant intervals of the integration. A "true" solution, \textit{YTRUE}, to this problem is calculated using TOL=10^{-12} and relative error control; An approximate solution is then calculated using TOL=10^{-3} and the default error control. As can be seen from the errors printed, the solution is quite inaccurate.
SUBROUTINE TEST4B
INTEGER N, NW, IND, K
DOUBLE PRECISION X, XEND, Y(2), YTRUE(2), TOL, C(32), W(2,9),
+ ERR(2)
DATA C(1), C(2), C(3), C(4), C(5), C(6), C(7), C(8), C(9) /9*0.00/
EXTERNAL FCNI
C
C CALCULATE AN ACCURATE SOLUTION TO THE PREDATOR-PREY PROBLEM
C USING THE RELATIVE ERROR CONTROL OPTION
N = 2 ; X = 0.00 ; YTRUE(1) = 1.00
NW = 2 ; XEND = 20.00 ; YTRUE(2) = 7.00
IND = 2 ; TOL = 1.0-12 ; C(1) = 2.00
C
C CALL DVERK(N, FCNI, X, YTRUE, XEND, TOL, IND, C, NW, W)
CHECK IND *EQ. 3 - OTHERWISE TAKE APPROPRIATE ACTION
C
C CALCULATE AN APPROXIMATE SOLUTION TO THE PREDATOR-PREY PROBLEM
C USING ERROR CONTROL OPTION 4 WITH FLOOR VALUES 1.0-4 AND 1.00
X = 0.00 ; Y(1) = 1.00 ; IND = 2 ; C(31) = 1.0-4
TOL = 1.0-3 ; Y(2) = 7.00 ; C(1) = 4.00 ; C(32) = 1.00
C
C CALL DVERK(N, FCNI, X, Y, XEND, TOL, IND, C, NW, W)
CHECK IND *EQ. 3 - OTHERWISE TAKE APPROPRIATE ACTION
C
C CALCULATE THE ERRORS AND PRINT THE RESULTS
ERR(1) = YTRUE(1) - Y(1)
ERR(2) = YTRUE(2) - Y(2)
WRITE(5,1) Y(1), ERR(1), Y(2), ERR(2)
1 FORMAT(1H1, 13HOUTPUT FROM TEST4B
+ / 1H0, 7X, 4HY(1), 11X, 6HEPR(1), 8X, 4HY(2), 11X, 6HEPR(2)
+ / 1H0, 2H(1PD17,6, 1PD12,3) )
C
RETURN
END

OUTPUT FCN TEST4B

Y(1)    ERR(1)    Y(2)    ERR(2)
1.693960D-04  -5.4670-07  1.853155D+00  -5.6450-02

Figure 4B  This example is the same as the one in Figure 4A except that the approximate solution is calculated using error control option 4 with the floor values C(31)=10^-4 and C(32)=1. As can be seen from the errors printed, the solution is much more accurate in this example. In many similar problems, controlling the error in a component with a small value may be necessary. However, if one of the components of the solution passes through zero, using relative error control may lead to a division by zero (or a very small number). Using error control option 3 or 4 with appropriately chosen floor values may be the most suitable way to solve such problems.
It would be desirable to have a measure of the "scale" of a problem, from which each method can calculate an HMAX that is appropriate to itself, and the SCALE parameter, C(5), is an attempt to provide just such a measure. (Besides using C(5) to determine HMAX, DVERK also uses it to modify the acceptance criterion.)

DVERK's use of SCALE is based on a theoretical study of the application of DVERK's formulas to homogeneous linear equations with constant coefficients, where the appropriate value of SCALE is exactly the Lipschitz constant (using a max norm). The default value of SCALE is 1, from which the subroutine computes a default value of 2 for HMAX. The user can think of SCALE as a "reliability parameter" in much the same way that he can think of TOL as being an "accuracy parameter". Larger values of SCALE make the results more reliable (HMAX is proportional to the reciprocal of SCALE), just as smaller values of TOL make the results more accurate.

An example that illustrates the importance of using SCALE is given in figure 5.

The use of C(7) to impose a limit on the number of function evaluations to be allowed is straightforward. We illustrate by modifying the example of figure 1 to impose such a limit. As can be seen from the example in figure 6, the limit we chose was exceeded when X reached the value 2.70282, and the subroutine returns to the calling program with IND=-1.
SUBROUTINE TESTS
INTEGER N, NW, IND, I, K
DOUBLE PRECISION X, XEND, Y(1), TOL(3), C(24), W(1,9), RELERR(3)
+ SCALE(6), YTRUE
DATA SCALE / .125D0, .25D0, .50D0, 1.0D0, 2.0D0, 4.0D0 /,
+ TOL / 1.0D-5, 1.0D-7, 1.0D-9 /,
+ C(1), C(2), C(3), C(4), C(5), C(6), C(7), C(8), C(9) /9D0.0D0/
EXTERNAL FCN2

CALCULATE THE SOLUTION TO THE PROBLEM Y' = Y, Y = 1 AT X = 0
USING SEVERAL VALUES OF SCALE AND TOL
N = 1 : XEND = 10.0D0 : C(4) = 2.5D0
NW = 1 : YTRUE = DEXP(XEND)

OUTPUT HEADING
WRITE(6,1) (TOL(I), I = 1, 3)
1 FORMAT(1HL, 70HOUTPUT FROM TESTS - RELATIVE ERROR FOR SEVERAL VALU
+ES OF TOL AND SCALE
+ / 1H0. 5X, 5HSSCALE, 3(5X, 5HTOL =, 1P08.1) /

FOR EACH VALUE OF SCALE AND TOL CALCULATE THE SOLUTION TO THE
PROBLEM AND OUTPUT ITS RELATIVE ERROR
DO 20 K = 1, 6
C(5) = SCALE(K)
DO 10 I = 1, 3
X = 0.0D0 : Y(1) = 1.0D0 : IND = 2
CALL DVERK(N, FCN2, X, Y, XEND, TOL(I), IND, C, NW, W)
CHECK IND .EQ. 3 - OTHERWISE TAKE APPROPRIATE ACTION
RELERR(I) = (Y(1) - YTRUE) / YTRUE
10 CONTINUE
WRITE(6,2) C(5), (RELERR(I), I = 1, 3)
2 FORMAT(1H, F10.3, 1P3D15.6)
20 CONTINUE
RETURN
END

OUTPUT FROM TESTS - RELATIVE ERROR FOR SEVERAL VALUES OF TOL AND SCALE

<table>
<thead>
<tr>
<th>SCALE</th>
<th>TOL = 1.00-05</th>
<th>TOL = 1.00-07</th>
<th>TOL = 1.00-09</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.125</td>
<td>-4.9226950-03</td>
<td>-4.9095180-03</td>
<td>-4.9094040-03</td>
</tr>
<tr>
<td>0.250</td>
<td>-4.9156860-03</td>
<td>-4.9094590-03</td>
<td>-4.9094040-03</td>
</tr>
<tr>
<td>0.500</td>
<td>-4.9124830-03</td>
<td>-4.9094310-03</td>
<td>-4.9094040-03</td>
</tr>
<tr>
<td>1.000</td>
<td>-2.166140-06</td>
<td>-1.8362730-08</td>
<td>-1.6755720-10</td>
</tr>
<tr>
<td>2.000</td>
<td>-1.0161840-06</td>
<td>-9.0029360-09</td>
<td>-8.2376900-11</td>
</tr>
<tr>
<td>4.000</td>
<td>-4.1907140-07</td>
<td>-4.4511220-09</td>
<td>-4.1178760-11</td>
</tr>
</tbody>
</table>

Figure 5 The problem Y' = Y, Y(0) = 1 is integrated from
0 to 10 with several values of SCALE and TOL, and the relative
error in the integration is printed. For this problem, SCALE
should be greater than or equal to 1. When SCALE is less
than 1, the error control is extremely unreliable. When SCALE
is greater than 1, a modest improvement in the accuracy is
obtained. (Such dramatic unreliability is not likely to occur
with a poor choice of SCALE. However, this example demonstrates
that such behavior is possible. It happened here because we
deliberately chose a "bad" value for HSTART.)
SUBROUTINE TEST6
INTEGER N, NW, IND, K
DOUBLE PRECISION X, XEND, Y(2), TOL, C(24), W(2,9)
DATA C(1), C(2), C(3), C(4), C(5), C(6), C(7), C(8), C(9)/9*0.00/ EXTERNAL FCNI

C C
C CALCULATE THE SOLUTION TO THE PREDATOR-PREY PROBLEM BUT LIMIT
C THE NUMBER OF FUNCTION EVALUATIONS TO 100
N = 2 ; X = 0.00 ; XEND = 10.00
NW = 2 ; Y(1) = 1.00 ; TOL = 1.0-6
IND = 2 ; Y(2) = 3.00 ; C(7) = 100.00

C CALL DVERK(N, FCNI, X, Y, XEND, TOL, IND, C, NW, W)

C OUTPUT HEADING AND RESULTS
WRITE(6,1) IND, X, Y(1), Y(2), C(24)
1 FORMAT(1H1, 17HOUTPUT FROM TEST6
+ / 1HO, 5X, 37HTHE INTEGRATION TERMINATED WITH IND = -.13,
+ / 7HAT X =, F8.5
+ / 1H5, 5X, 10HAND Y(1) =, 1PD12.5, 8H Y(2) =, 1PD12.5
+ / 1HO, 5X, 43HTHE NUMBER OF FUNCTION EVALUATIONS USED WAS,
+ / DPF5.0)

C C
C RE-CALL DVERK TO CAUSE AN ABORT
CALL DVERK(N, FCNI, X, Y, XEND, TOL, IND, C, NW, W)

C RETURN
END

OUTPUT FROM TEST6

THE INTEGRATION TERMINATED WITH IND = -1 AT X = 2.70282
AND Y(1) = 1.895190-01 Y(2) = 3.127990-01

THE NUMBER OF FUNCTION EVALUATIONS USED WAS 103.

COMPUTATION STOPPED IN DVERK WITH THE FOLLOWING VALUES -

IND = -1
TOL = 1.000000006 X = 2.702816859719279D+00
N = 2
MMIN = 9.795879D-10 XEND = 1.000000000000000D+01
NW = 2
HMmax = 2.000000000000000D+00 PREV XEND = 0.0

NO OF SUCCESSFUL STEPS = 12.
NO OF SUCCESSIVE FAILURES = 0.
NO OF FUNCTION EVALS = 103.

THE COMPONENTS OF Y ARE
1.8951871850235560-01 3.1279867150403330-01

Figure 6 The predator-prey problem is integrated with the number of function evaluations limited to 100. The integration terminates the step after this number is exceeded. Then the subroutine is re-entered without changing IND (IND=-1) to force it to abort and print its error message.
To illustrate what happens when the subroutine decides to "abort" the calculation, it is re-entered with this negative value of IND. As mentioned earlier, the subroutine has no reasonable course of action when entered with such a value of IND, except to output current information that might be of interest (and that might also help locate the source of trouble), and then to stop. The second part of the output with figure 6 shows what can be expected under such circumstances.

We turn finally to the interrupt options. For these it is helpful to have in mind an "overview" of the subroutine, like the one presented on page 15. (A more detailed overview is contained in the listing.)

Notice that the first interrupt, which is associated with C(8), takes place just after the calculations have been made in preparation for taking a trial step, including in particular the calculation of the preliminary magnitude, H MAG, of the trial step-size. However, the final value of H MAG must still be determined; it could differ from the preliminary value only if the calculations have reached a point close to XEND. (If the subroutine cannot reach XEND exactly, with a step-size whose magnitude is less than or equal to H MAG, it will not attempt to go more than half-way to XEND. The corresponding value of H TRIAL may also differ from H MAG in sign.)
Initialization of options, counters, etc., and
the handling of re-entries after interrupts
Loop over the following, once for each trial step
Do preparatory calculations, including HMIN,
SCALE, HMAX and preliminary HMAG
Interrupt no. 1 if requested --- re-entry
Determine HMAG, XTRIAL, HTRIAL
Calculate YTRIAL
Calculate EST
Set IND=5 if acceptable, else =6
Interrupt no. 2 if requested --- re-entry
Accept or reject, update if necessary,
perhaps return to calling program, etc.
End loop

OVERVIEW "Overview" of subroutine to help understand
how to make use of the interrupt options.

The second interrupt, the one associated with C(9),
takes place immediately after the subroutine has decided
whether or not it would accept the most recent trial
calculation, and has set IND accordingly, but before it has
taken any action. This means that the user has all possible
information at his disposal (including the previous value
of Y, the new trial value of Y, and the error estimate), and
is also in a position to change the decision of the subroutine
(by forcing it to accept what it had planned to reject, or
vice versa).
An example illustrating the second interrupt is presented in figure 7, where DVERK is used to determine the value of X at which Y reaches a prescribed value.

One more remark about the use of options should be made before concluding this section. It should be pointed out that any option can be either initiated, or altered from a previous selection, prior to any re-entry to the subroutine. The other values of C(1) through C(9) do not need to be initialized, or re-initialized, and the value of IND need not be changed. For example, on a normal re-entry with IND=3, a new error control option could be selected. Or, on re-entry after an interrupt with IND=4, 5 or 6, a new value of HMAX or HMIN could be selected; the new value would not take effect instantly, of course, but only when the new internal value is determined in preparation for the next trial step. (E.g., a new HMAX was used in figure 7.)

When Not to Use

On the basis of both theoretical and experimental comparisons, we believe that the formulas on which this subroutine is based are the best of their kind that are currently available. They appear to be as efficient and reliable as the corresponding better-known formulas of the same order that were developed earlier by Fehlberg (which are reported on elsewhere (1)), but they do not have the
SUBROUTINE TEST
INTEGER N, NW, IND, K
DOUBLE PRECISION X, XEND, Y(1), TOL, C(24), W(1:9), XLOW, XHIGH,
                  EPS, ROOT
DATA C(1), C(2), C(3), C(4), C(5), C(6), C(7), C(8), C(9) /9*0.0D0/
EXTERNAL FCN2
C
N = 1 ; X = 0.0D0 ; TOL = 1.0D-6 ; EPS = 5.0D-4
NW = 1 ; XEND = 1.0D2 ; C(1) = 2.0D0
IND = 2 ; Y(1) = 1.0D0 ; C(9) = 1.0D0
C
LOOP UNTIL AN ACCEPTABLE VALUE OF YTRIAL IS GREATER-EQUAL 100
10 CALL DVERK(N, FCN2, X, Y, XEND, TOL, IND, C, NW, W)
   IF ((IND.EQ.5) .AND. (W(1:9).GE.100.0D0)) GO TO 20
   GO TO 10
20 CONTINUE
C
XLOW = X ; XHIGH = C(17) ; IND = 6
C
LOOP USING THE BISECTION METHOD UNTIL (XHIGH - XLOW) .LE. EPS
30 IF ((XHIGH - XLOW) .LE. EPS) GO TO 60
C
SET HMAX = (XHIGH - XLOW) / 2
C(6) = (XHIGH - XLOW) / 2.0D0
CALL DVERK(N, FCN2, X, Y, XEND, TOL, IND, C, NW, W)
   IF ((IND.NE.5) .OR. (W(1:9).LT.100.0D0)) GO TO 40
   YTRIAL ACCEPTABLE AND GE 100 - RESET XHIGH AND REJECT STEP
   XHIGH = C(17)
   IND = 6
40 CONTINUE
   IF ((IND.NE.5) .OR. (W(1:9).GE.100.0D0)) GO TO 50
   YTRIAL ACCEPTABLE AND LT 100 - RESET XLOW AND ACCEPT STEP
   XLOW = C(17)
50 CONTINUE
GO TO 30
60 CONTINUE
C
OUTPUT THE APPROXIMATE SOLUTION TO THE EQUATION EXP(X) = 100
ROOT = (XHIGH + XLOW) / 2.0D0
WRITE(6,1) ROOT
1 FORMAT(1H1, 1HOUTPUT FROM TEST
               + / 1H0, 5X, 45HTHE SOLUTION OF EXP(X) = 100 IS APPROXIMATELY,
               + 6X, 3 )
C
RETURN
END

OUTPUT FROM TEST

THE SOLUTION OF EXP(X) = 100 IS APPROXIMATELY 4.505

Figure 7. This example demonstrates the use of interrupt number 2. The problem Y'=Y, Y(0)=1 is integrated until YTRIAL, W(1,9), is greater than or equal to 100. Then the bisection method is used to approximate the solution to the equation exp(x)=100. This example also demonstrates the use of the HMAX specification option.
weakness with respect to differential equations of the simple form \( y' = f(x) \), which is common to all Fehlberg methods of order greater than 4. They also seem to have some other advantages as well (3). Some less extensive testing suggests that this subroutine is also able to cope with problems involving discontinuities.

However, there are three kinds of problems for which this subroutine would not be as efficient as currently available alternatives:

(a) **Stiff systems** It is well-known that explicit Runge-Kutta methods cannot compete with specially designed stiff methods on any but very mildly stiff systems. For further information about stiff methods see (2), or try DVOGER from the IMSL library, or GEAR or EPISODE, which are available from the Argonne Code Centre.

(b) **Interpolation** If output is required at very finely spaced values of \( X \), as for example in graphical output, the user should consider using a variable-order-Adams program, such as DVOGER from IMSL or DE/STEP from Argonne. (Gear and EPISODE have options to provide such Adams methods as well. Programs of this type have also been developed by Krogh at JPL, and Sedgwick at the University of Toronto.) These programs provide the information needed for interpolation between step points.

(c) **Expensive functions** If the function evaluations are expensive, so that most of the computer time is spent
in such evaluations, and the overhead of variable-order-Adams methods is not too important, an advantage in efficiency of perhaps 2 or 3 to 1 can be obtained with Adams methods.

Machine Dependencies

DVERK has been written to run in double precision on an IBM 360/370. A single precision version, SVERK, is also being developed.

In any event, the only 360/370 dependent part of the code of which we are aware appears in the machine constant RREB. RREB is the relative roundoff error bound. It is computed following statement 35, and assigned to C(10). As can be seen, the value for the 360/370 in double precision is 16**(-13). (In single precision on the same machine it would be 16**(-5).)

DWARF is a very small machine number. It is computed immediately after RREB and assigned to C(11). We have somewhat arbitrarily chosen the value to be 10**(-50). (We originally chose the smallest positive machine number for the 360/370, namely 16**(-65), but its computation overflowed; even 16**(-63) gave trouble with one of our compilers.)

The coefficients in the integration formulas have been given in the form of integers, with the lowest common denominator factored out. Our purpose in doing it this way was to help make the program more portable, and at the same
time to avoid possible inefficiencies that would exist if the
coefficients were left as fractions. On the other hand, the
number of digits is too great for single precision on some
machines (for example, the 360/370), and the program would
have to be modified accordingly.

Two changes in the declarations are needed in the
subroutine if it is to be run under a WATFIV compiler. It
is necessary to declare the dimension of C to be large enough,
namely 24 (unless C(1) is to be set equal to 4 or 5, in which
case the dimension would have to be at least N+30). Also,
it is necessary to declare DABS, DMAX1, DMIN1, and DSIGN to
be DOUBLE PRECISION.

Availability of Program

The subroutine DVERK, along with the subroutines
TEST1, TEST2, ..., TEST7 given in this report, the subroutines
FCN1 and FCN2 that they use, and a driver program, are available
on cards or tape. The driver program, FCN1 and FCN2 are as
follows:

```c
CALL TEST1
CALL TEST2
CALL TEST3A
CALL TEST3B
CALL TEST4A
CALL TEST4B
CALL TEST5
CALL TEST7
CALL TEST6
STOP
END
```
SUBROUTINE FCN1(N, X, Y, YP)
INTEGER N
DOUBLE PRECISION X, Y(N), YP(N)
YP(1) = 2.0D0 * Y(1) * (1.0D0 - Y(2))
YP(2) = Y(2) * (Y(1) - 1.0D0)
RETURN
END

SUBROUTINE FCN2(N, X, Y, YP)
INTEGER N
DOUBLE PRECISION X, Y(N), YP(N)
YP(1) = Y(1)
RETURN
END

(TEST6 is called last because it causes DVERK to stop execution.
The assignment statements in TEST1, etc., that are separated by semi-colons in this report appear on separate lines in what will be sent, so that the driver and tests are in standard Fortran; the semi-colons were used in the figures of this report only to save space.)

To cover handling and mailing, the cost for a card deck is $35.00 and for a tape $25.00. To obtain a copy on tape, it is necessary to send a tape (a mini tape is sufficient). The DVERK tape is a 9 track, 800 BPI source tape consisting of 2 files (the first contains the driver program, FCN1, FCN2, TEST1, TEST2, ..., TEST7 and the second contains DVERK). The data set attributes are DCB=(LRECL=80, BLKSIZE=800, RECFM=FB, DEN=2). The tape has no labels.
Requests for DVERK should be mailed to
DEPARTMENT OF COMPUTER SCIENCE
McLENNAN PHYSICAL LABORATORIES
UNIVERSITY OF TORONTO
TORONTO, ONTARIO, CANADA

ATTENTION: T. E. HULL

Cheques should be made payable to the University
of Toronto.

Bibliography
(1) W. H. Enright and T. E. Hull, Test Results on Initial
Value Methods for Non-Stiff Ordinary Differential
(2) W. H. Enright, T. E. Hull and B. Lindberg, Comparing
Numerical Methods for Stiff Systems of O.D.E:s, BIT 15,
(3) K. R. Jackson, W. H. Enright and T. E. Hull, A theoretical criterion for comparing Runge-Kutta methods, in
preparation.
(4) J. H. Verner, private communication.
DVERK Listing:

SUBROUTINE DVERK(N, FCN, X, Y, XEND, TOL, IND, C, NW, W)
INTEGER N, IND, NW, K
DOUBLE PRECISION X, Y(N), XEND, TOL, C(1), W(NW,9), TEMP

C*************************************************************
C PURPOSE - THIS IS A RUNGE-KUTTA SUBROUTINE BASED ON VERWER'S
C FIFTH AND SIXTH ORDER PAIR OF FORMULAS FOR FINDING APPROXIMATIONS TO
C THE SOLUTION OF A SYSTEM OF FIRST ORDER ORDINARY
C EQUATIONS WITH INITIAL CONDITIONS. IT ATTEMPTS TO KEEP THE GLOBAL
C ERROR PROPORTIONAL TO A TOLERANCE SPECIFIED BY THE USER.* (THE
C PROPORTIONALITY DEPENDS ON THE KIND OF ERROR CONTROL THAT IS USED.
C AS WELL AS THE DIFFERENTIAL EQUATION AND THE RANGE OF INTEGRATION.)*
C
C VARIOUS OPTIONS ARE AVAILABLE TO THE USER, INCLUDING DIFFERENT
C KINDS OF ERROR CONTROL, RESTRICTIONS ON STEP SIZES, AND INTERRUPTS
C WHICH PERMIT THE USER TO EXAMINE THE STATE OF THE CALCULATION (AND
C PERHAPS MAKE MODIFICATIONS) DURING INTERMEDIATE STAGES. *
C
C THE PROGRAM IS EFFICIENT FOR NON-STIFF SYSTEMS. HOWEVER, A GOOD
C VARIABLE-ORDER-ADAMS METHOD WILL PROBABLY BE MORE EFFICIENT IF THE
C FUNCTION EVALUATIONS ARE VERY COSTLY. SUCH A METHOD WOULD ALSO BE
C MORE SUITABLE IF ONE WANTED TO OBTAIN A LARGE NUMBER OF INTERMEDIATE
C SOLUTION VALUES BY INTERPOLATION, AS MIGHT BE THE CASE FOR EXAMPLE
C WITH GRAPHICAL OUTPUT.

MULL-ENRIGHT-JACKSON 1/10/76

C*************************************************************
C USE - THE USER MUST SPECIFY EACH OF THE FOLLOWING
C
C N  NUMBER OF EQUATIONS
C FCN NAME OF SUBROUTINE FOR EVALUATING FUNCTIONS - THE SUBROUTINE
C ITSELF MUST ALSO BE PROVIDED BY THE USER - IT SHOULD BE OF
C THE FOLLOWING FORM
C SUBROUTINE FCN(N, X, Y, YPRIME)
C INTEGER N
C DOUBLE PRECISION X, Y(N), YPRIME(N)
C *** ETC ***
C AND IT SHOULD EVALUATE YPRIME, GIVEN N, X AND Y
C
C X  INDEPENDENT VARIABLE - INITIAL VALUE SUPPLIED BY USER
C
C Y  DEPENDENT VARIABLE - INITIAL VALUES OF COMPONENTS Y(1), Y(2),
C    ... Y(N) SUPPLIED BY USER
C
C XEND VALUE OF X TO WHICH INTEGRATION IS TO BE CARRIED OUT - IT MAY
C BE LESS THAN THE INITIAL VALUE OF X
C
C TOL TOLERANCE - THE SUBROUTINE ATTEMPTS TO CONTROL A NORM OF THE
C LOCAL ERROR IN SUCH A WAY THAT THE GLOBAL ERROR IS
C PROPORTIONAL TO TOL. IN SOME PROBLEMS THERE WILL BE ENOUGH
C DAMPING OF ERRORS, AS WELL AS SOME CANCELLATION, SO THAT
C THE GLOBAL ERROR WILL BE LESS THAN TOL. ALTERNATIVELY, THE
C CONTROL CAN BE VIEWED AS ATTEMPTING TO PROVIDE A
C CALCULATED VALUE OF Y AT XEND WHICH IS THE EXACT SOLUTION
C
TO THE PROBLEM $y' = f(x, y) + e(x)$ WHERE THE NORM OF $e(x)$ * IS PROPORTIONAL TO TOL. (THE NORM IS A MAX NORM WITH * WEIGHTS THAT DEPEND ON THE ERROR CONTROL STRATEGY CHOSEN * BY THE USER, THE DEFAULT WEIGHT FOR THE $k$-TH COMPONENT IS * $1/\text{MAX}(1, \text{ABS}(y(k)))$ WHICH THEREFORE PROVIDES A MIXTURE OF * ABSOLUTE AND RELATIVE ERROR CONTROL.)

INDICATOR - ON INITIAL ENTRY IND MUST BE SET EQUAL TO EITHER * 1 OR 2. IF THE USER DOES NOT WISH TO USE ANY OPTIONS, HE * SHOULD SET IND TO 1 - ALL THAT REMAINS FOR THE USER TO DO * THEN IS TO DECLARE C AND W, AND TO SPECIFY NW, THE USER * MAY ALSO SELECT VARIOUS OPTIONS ON INITIAL ENTRY BY * SETTING IND = 2 AND INITIALIZING THE FIRST 9 COMPONENTS OF * C AS DESCRIBED IN THE NEXT SECTION. HE MAY ALSO RE-ENTER * THE SUBROUTINE WITH IND = 3 AS MENTIONED AGAIN BELOW. IN ANY EVENT, THE SUBROUTINE RETURNS WITH IND EQUAL TO * 3 AFTER A NORMAL RETURN * 4, 5, OR 6 AFTER AN INTERRUPT (SEE OPTIONS C(8), C(9)) * -1, -2, OR -3 AFTER AN ERROR CONDITION (SEE BELOW)

C COMMUNICATIONS VECTOR - THE DIMENSION MUST BE GREATER THAN OR * EQUAL TO 24, UNLESS OPTION C(1) = 4 OR 5 IS USED, IN WHICH * CASE THE DIMENSION MUST BE GREATER THAN OR EQUAL TO N+30.

NW FIRST DIMENSION OF WORKSPACE W - MUST BE GREATER THAN OR * EQUAL TO N

W WORKSPACE MATRIX - FIRST DIMENSION MUST BE NW AND SECOND MUST * BE GREATER THAN OR EQUAL TO 9

THE SUBROUTINE WILL NORMALLY RETURN WITH IND = 3, HAVING * REPLACED THE INITIAL VALUES OF X AND Y WITH, RESPECTIVELY, THE VALUE * OF X(0) AND AN APPROXIMATION TO Y AT X=0. THE SUBROUTINE CAN BE * CALLED REPEATEDLY WITH NEW VALUES OF X AND Y WITHOUT HAVING TO CHANGE ANY OTHER ARGUMENT, HOWEVER, CHANGES IN TOL, OR ANY OF THE OPTIONS * DESCRIBED BELOW, MAY ALSO BE MADE ON SUCH A RE-ENTRY IF DESIRED.

THREE ERROR RETURNS ARE ALSO POSSIBLE, IN WHICH CASE X AND Y * WILL BE THE MOST RECENTLY ACCEPTED VALUES - * WITH IND = -3 THE SUBROUTINE WAS UNABLE TO SATISFY THE ERROR * REQUIREMENT WITH A PARTICULAR STEP-SIZE THAT IS LESS THAN OR * EQUAL TO HMIN, WHICH MAY MEAN THAT TOL IS TOO SMALL * WITH IND = -2 THE VALUE OF HMIN IS GREATER THAN HMAX, WHICH * PROBABLY MEANS THAT THE REQUESTED TOL (WHICH IS USED IN THE * CALCULATION OF HMIN) IS TOO SMALL * WITH IND = -1 THE ALLOWED MAXIMUM NUMBER OF FCN EVALUATIONS HAS * BEEN EXCEEDED, BUT THIS CAN ONLY OCCUR IF OPTION C(7), AS * DESCRIBED IN THE NEXT SECTION, HAS BEEN USED

THERE ARE SEVERAL CIRCUMSTANCES THAT WILL CAUSE THE CALCULATIONS * TO BE TERMINATED, ALONG WITH OUTPUT OF INFORMATION THAT WILL HELP * THE USER DETERMINE THE CAUSE OF THE TROUBLE. THESE CIRCUMSTANCES * INVOLVE ENTRY WITH ILLEGAL OR INCONSISTENT VALUES OF THE ARGUMENTS, * SUCH AS ATTEMPTING A NORMAL RE-ENTRY WITHOUT FIRST CHANGING THE * VALUE OF X(0), OR ATTEMPTING TO RE-ENTER WITH IND LESS THAN ZERO.

*******************************************************************************
OPTIONS - IF THE SUBROUTINE IS ENTERED WITH IND = 1, THE FIRST 9
* COMPONENTS OF THE COMMUNICATIONS VECTOR ARE INITIALIZED TO ZERO AND *
* THE SUBROUTINE USES ONLY DEFAULT VALUES FOR EACH OPTION. IF THE *
* SUBROUTINE IS ENTERED WITH IND = 2, THE USER MUST SPECIFY EACH OF *
* THESE 9 COMPONENTS - NORMALLY HE WOULD FIRST SET THEM ALL TO ZERO, *
* AND THEN MAKE NON-ZERO THOSE THAT CORRESPOND TO THE PARTICULAR *
* OPTIONS HE WISHES TO SELECT. IN ANY EVENT, OPTIONS MAY BE CHANGED ON *
* RE-ENTRY TO THE SUBROUTINE - BUT IF THE USER CHANGES ANY OF THE *
* OPTIONS, OR TOL, IN THE COURSE OF A CALCULATION HE SHOULD BE CAREFUL *
* ABOUT HOW SUCH CHANGES AFFECT THE SUBROUTINE - IT MAY BE BETTER TO *
* RESTART WITH IND = 1 OR 2. (COMPONENTS 10 TO 24 OF C ARE USED BY THE *
* PROGRAM - THE INFORMATION IS AVAILABLE TO THE USER, BUT SHOULDN'T *
* NORMALLY BE CHANGED BY HIM.)
*
C C(1) ERROR CONTROL INDICATOR - THE NORM OF THE LOCAL ERROR IS THE *
MAX NORM OF THE WEIGHTED ERROR ESTIMATE VECTOR, THE *
WEIGHTS BEING DETERMINED ACCORDING TO THE VALUE OF C(1) - *
IF C(1)=1 THE WEIGHTS ARE 1/ABS(Y(K)) (ABSOLUTE ERROR CONTROL) *
IF C(1)=2 THE WEIGHTS ARE 1/ABS(Y(K)) (RELATIVE ERROR CONTROL) *
IF C(1)=3 THE WEIGHTS ARE 1/MAX(ABS(C(2))ABS(Y(K))) *
(RELATIVE ERROR CONTROL. UNLESS ABS(Y(K)) IS LESS THAN THE FLOOR VALUE, ABS(C(2)) ) *
IF C(1)=4 THE WEIGHTS ARE 1/MAX(ABS(C(K+30))ABS(Y(K))) *
(HERE INDIVIDUAL FLOOR VALUES ARE USED) *
IF C(1)=5 THE WEIGHTS ARE 1/ABS(C(K+30)) *
FOR ALL OTHER VALUES OF C(1), INCLUDING C(1) = 0, THE *
DEFAULT VALUES OF THE WEIGHTS ARE TAKEN TO BE *
1/MAX(1,ABS(Y(K))), AS MENTIONED EARLIER *
(IN THE TWO CASES C(1) = 4 OR 5 THE USER MUST DECLARE THE *
DIMENSION OF C TO BE AT LEAST N+30 AND MUST INITIALIZE THE *
COMPONENTS C(31), C(32), .... C(N+30)).
*
C C(2) FLOOR VALUE - USED WHEN THE INDICATOR C(1) HAS THE VALUE 3 *
*
C C(3) HMIN SPECIFICATION - IF NOT ZERO, THE SUBROUTINE CHOOSES HMIN *
TO BE ABS(C(3)) - OTHERWISE IT USES THE DEFAULT VALUE *
10**MAX(DWARF,RREB)*MAX(WEIGHTED NORM Y/TCL,ABS(X)) *
WHERE DWARF IS A VERY SMALL POSITIVE MACHINE NUMBER AND *
RREB IS THE RELATIVE ROUNDOFF ERROR BOUND *
*
C C(4) HSTART SPECIFICATION - IF NOT ZERO, THE SUBROUTINE WILL USE *
AN INITIAL HMAX EQUAL TO ABS(C(4)), EXCEPT OF COURSE FOR *
THE RESTRICTIONS IMPOSED BY HMIN AND HMAX - OTHERWISE IT *
USES THE DEFAULT VALUE OF HMAX**TCL**(1/6) *
*
C C(5) SCALE SPECIFICATION - THIS IS INTENDED TO BE A MEASURE OF THE *
SCALE OF THE PROBLEM - LARGER VALUES OF SCALE TEND TO MAKE *
THE METHOD MORE RELIABLE. FIRST BY POSSIBLY RESTRICTING *
HMAX (AS DESCRIBED BELOW) AND SECOND, BY TIGHTENING THE *
ACCEPTANCE REQUIREMENT - IF C(5) IS ZERO, A DEFAULT VALUE *
OF 1 IS USED. FOR LINEAR HOMOGENEOUS PROBLEMS WITH *
CONSTANT COEFFICIENTS, AN APPROPRIATE VALUE FOR SCALE IS A *
NORM OF THE ASSOCIATED MATRIX. FOR OTHER PROBLEMS, AN *
APPROXIMATION TO AN AVERAGE VALUE OF A NORM OF THE *
JACOBIAN ALONG THE TRAJECTORY MAY BE APPROPRIATE *
*
C C(6) HMAX SPECIFICATION - FOUR CASES ARE POSSIBLE
IF C(6),NE.0 AND C(5),NE.0, HMAX IS TAKEN TO BE
MIN(ABS(C(6)),2/ABS(C(5)))
IF C(6),NE.0 AND C(5),EQ.0, HMAX IS TAKEN TO BE ABS(C(6))
IF C(6),EQ.0 AND C(5),NE.0, HMAX IS TAKEN TO BE
2/ABS(C(5))
IF C(6),EQ.0 AND C(5),EQ.0, HMAX IS GIVEN A DEFAULT VALUE
OF 2

C(7) MAXIMUM NUMBER OF FUNCTION EVALUATIONS - IF NOT ZERO, AN
ERROR RETURN WITH IND = -1 WILL BE CAUSED WHEN THE NUMBER
OF FUNCTION EVALUATIONS EXCEEDS ABS(C(7))

C(8) INTERRUPT NUMBER 1 - IF NOT ZERO, THE SUBROUTINE WILL
INTERRUPT THE CALCULATIONS AFTER IT HAS CHOSEN ITS
PRELIMINARY VALUE OF HMAG, AND JUST BEFORE CHOOSING MTRIAL
AND XTRIAL IN PREPARATION FOR TAKING A STEP (MTRIAL MAY
DIFFER FROM HMAG IN SIGN, AND MAY REQUIRE ADJUSTMENT IF
XEND IS NEAR) - THE SUBROUTINE RETURNS WITH IND = 4, AND
WILL RESUME CALCULATION AT THE POINT OF INTERRUPTION IF
RE-ENTERED WITH IND = 4

C(9) INTERRUPT NUMBER 2 - IF NOT ZERO, THE SUBROUTINE WILL
INTERRUPT THE CALCULATIONS IMMEDIATELY AFTER IT HAS
DECIDED WHETHER OR NOT TO ACCEPT THE RESULT OF THE MOST
RECENT TRIAL STEP, WITH IND = 5 IF IT PLANS TO ACCEPT, OR
IND = 6 IF IT PLANS TO REJECT - Y(*) IS THE PREVIOUSLY
ACCEPTED RESULT, WHILE W(*,9) IS THE NEWLY COMPUTED TRIAL
VALUE, AND W(*,2) IS THE UNWEIGHTED ERROR ESTIMATE VECTOR.
THE SUBROUTINE WILL RESUME CALCULATIONS AT THE POINT OF
INTERRUPTION ON RE-ENTRY WITH IND = 5 OR 6. (THE USER MAY
CHANGE IND IN THIS CASE IF HE WISHES, FOR EXAMPLE TO FORCE
ACCEPTANCE OF A STEP THAT WOULD OTHERWISE BE REJECTED, OR
VICE VERSA. HE CAN ALSO RESTART WITH IND = 1 OR 2.)

******************************************************************************

C SUMMARY OF THE COMPONENTS OF THE COMMUNICATIONS VECTOR

PRESCRIBED AT THE OPTION OF THE USER

C(1) ERROR CONTROL INDICATOR
C(2) LOCA L VALUE
C(3) MINIMUM SPECIFICATION
C(4) MSTART SPECIFICATION
C(5) SCALE SPECIFICATION
C(6) MAXIMUM SPECIFICATION
C(7) MAX NO OF FCN EVALS
C(8) INTERRUPT NO 1
C(9) INTERRUPT NO 2

DETERMINED BY THE PROGRAM

C(10) RRED(REAL ROUNDOFF ERR BND)
C(11) DAWF(VERY SMALL MACH NO)
C(12) WIGHTED NORM Y
C(13) MIN
C(14) HMAG
C(15) SCALE
C(16) HMAG
C(17) XTRIAL
C(18) MTRIAL
C(19) EST
C(20) PREVIOUS XEND
C(21) FLAG FOR XEND
C(22) NO OF SUCCESSFUL STEPS
C(23) NO OF SUCCESSIVE FAILURES
C(24) NO OF FCN EVALS

C IF C(1) = 4 OR 5, C(31), C(32), ..., C(N+30) ARE FLOOR VALUES
AN OVERVIEW OF THE PROGRAM

BEGIN INITIALIZATION, PARAMETER CHECKING, INTERRUPT RE-ENTRIES

****** ABORT IF N OF FCN EVALS > TOL

CASES - INITIAL ENTRY, NORMAL RE-ENTRY, INTERRUPT RE-ENTRIES

CASE 1 - INITIAL ENTRY (IND = EQ. 1 OR 2)

IF INITIAL ENTRY WITHOUT OPTIONS (IND = EQ. 1)

SET C(1) TO C(9) EQUAL TO ZERO

ELSE INITIAL ENTRY WITH OPTIONS (IND = EQ. 2)

MAKE C(1) TO C(9) NON-NEGATIVE

MAKE FLOOR VALUES NON-NEGATIVE IF THEY ARE TO BE USED

END IF

INITIALIZE PREB, DWF, PREV XEND, FLAG, COUNTS

CASE 2 - NORMAL RE-ENTRY (IND = EQ. 3)

****** ABORT IF XEND REACHED, AND EITHER X CHANGED OR XEND NOT

RE-INITIALIZE FLAG

CASE 3 - RE-ENTRY FOLLOWING AN INTERRUPT (IND = EQ. 4 TO 6)

TRANSFER CONTROL TO THE APPROPRIATE RE-ENTRY POINT

END CASES

END INITIALIZATION, ETC.

LOOP THROUGH THE FOLLOWING 4 STAGES, ONCE FOR EACH TRIAL STEP

STAGE 1 - PREPARE

****** ERROR RETURN (WITH IND=-1) IF N OF FCN EVALS TOO GREAT

CALC SLOPE (ADDING 1 TO NO OF FCN EVALS) IF IND NE 6

CALC HMIN, SCALE, HMAX

****** ERROR RETURN (WITH IND=-2) IF HMIN GT HMAX

CALC PRELIMINARY HMAG

****** INTERRUPT NO 1 (WITH IND=4) IF REQUESTED

RE-ENTRY

CALC HMAG, XTRIAL AND HTRIAL

END STAGE

STAGE 2 - CALC YTRIAL (ADDING 7 TO NO OF FCN EVALS)

STAGE 3 - CALC THE ERROR ESTIMATE

STAGE 4 - MAKE DECISIONS

SET IND=5 IF STEP ACCEPTABLE, ELSE SET IND=6

****** INTERRUPT NO 2 IF REQUESTED

RE-ENTRY

IF STEP ACCEPTED (IND = EQ. 5)

UPDATE X, Y FROM XTRIAL, YTRIAL

ADD 1 TO NO OF SUCCESSFUL STEPS

SET NO OF SUCCESSIVE FAILURES TO ZERO

****** RETURN (WITH IND=3, XEND SAVED, FLAG SET) IF X EQ. XEND

ELSE STEP NOT ACCEPTED (IND = EQ. 6)

ADD 1 TO NO OF SUCCESSIVE FAILURES

****** ERROR RETURN (WITH IND=-3) IF HMAG LE HMIN

END IF

END STAGE

END LOOP

BEGIN ABORT ACTION

OUTPUT APPROPRIATE MESSAGE ABOUT STOPPING THE CALCULATIONS,
ALONG WITH VALUES OF IND, N, NW, TOL, HMIN, HMAX, X, XEND,
PREVIOUS XEND, NO OF SUCCESSFUL STEPS, NO OF SUCCESSIVE
FAILURES, NO OF FCN EVALS, AND THE COMPONENTS OF Y

STOP

END ABORT ACTION
C******************************************************************************
C  * BEGIN INITIALIZATION, PARAMETER CHECKING, INTERRUPT RE-ENTRIES *
C******************************************************************************

C       ABORT IF IND OUT OF RANGE 1 TO 6
C       IF (IND.LT.1 .OR. IND.GT.6) GO TO 500

C      CASES - INITIAL ENTRY, NORMAL RE-ENTRY, INTERRUPT RE-ENTRIES
C      GO TO (5, 5, 45, 1111, 2222, 2222), IND
C      CASE 1 - INITIAL ENTRY (IND .EQ. 1 OR 2)
C            ABORT IF N.GT.NW OR TOL.LE.0
C 5      IF (N.GT.NW .OR. TOL.LE.0.D0) GO TO 500
C            IF (IND.EQ. 2) GO TO 15
C      INITIAL ENTRY WITHOUT OPTIONS (IND .EQ. 1)
C      SET C(1) TO C(9) EQUAL TO 0
C 10     DO 10 K = 1, 9
C      C(K) = 0.D0
C      CONTINUE
C      GO TO 35
C      CONTINUE
C      INITIAL ENTRY WITH OPTIONS (IND .EQ. 2)
C      MAKE C(1) TO C(9) NON-NEGATIVE
C      DO 20 K = 1, 9
C      C(K) = DABS(C(K))
C      CONTINUE
C      MAKE FLOOR VALUES NON-NEGATIVE IF THEY ARE TO BE USED
C      IF (C(1).NE.4.D0 .AND. C(1).NE.5.D0) GO TO 30
C      DO 25 K = 1, N
C      C(K+30) = DABS(C(K+30))
C 25     CONTINUE
C      CONTINUE
C      CONTINUE
C      INITIALIZE RREB, DWARF, PREV XEND, FLAG, COUNTS
C      C(10) = 16.00**(-13)
C      C(11) = 1.D-50
C      SET PREVIOUS XEND INITIALLY TO INITIAL VALUE OF X
C      C(20) = X
C      DO 40 K = 21, 24
C      C(K) = 0.D0
C 40     CONTINUE
C      GO TO 50
C      CASE 2 - NORMAL RE-ENTRY (IND .EQ. 3)
C       ABORT IF XEND REACHED, AND EITHER X CHANGED OR XEND NOT
C 45     IF (C(21).NE.0.D0 .AND.
C                  .AND.  
C                  (X.XNE.C(20) .OR. XEND.EQ.C(20))) GO TO 500
C      RE-INITIALIZE FLAG
C      C(21) = 0.D0
C      GO TO 50
C      CASE 3 - RE-ENTRY FOLLOWING AN INTERRUPT (IND .EQ. 4 TO 6)
C      TRANSFER CONTROL TO THE APPROPRIATE RE-ENTRY POINT***********
C      THIS HAS ALREADY BEEN HANDLED BY THE COMPUTED GO TO
C      END CASES
C 50     CONTINUE
C
C      END INITIALIZATION, ETC.
C C***********************************************************************
C * LOOP THROUGH THE FOLLOWING 4 STAGES, ONCE FOR EACH TRIAL STEP *
C * UNTIL THE OCCURRENCE OF ONE OF THE FOLLOWING *
C * (A) THE NORMAL RETURN (WITH IND .EQ. 3) ON REACHING XEND IN *
C * STAGE 4 *
C * (B) AN ERROR RETURN (WITH IND .LT. 0) IN STAGE 1 OR STAGE 4 *
C * (C) AN INTERRUPT RETURN (WITH IND .EQ. 4, 5 OR 6), IF *
C * REQUESTED, IN STAGE 1 OR STAGE 4 *
C***********************************************************************

99999 CONTINUE
C C***********************************************************************
C * STAGE 1 - PREPARE - DO CALCULATIONS OF HMIN, HMAX, ETC., *
C * AND SOME PARAMETER CHECKING, AND END UP WITH SUITABLE *
C * VALUES OF HMAG, XTRIAL AND HTRIAL IN PREPARATION FOR TAKING *
C * AN INTEGRATION STEP. *
C***********************************************************************
C C***********************************************************************
C ERROR RETURN (WITH IND=-1) IF NO OF FCN EVALS TOO GREAT
C IF (C(7) .EQ. 0 .OR. C(24) .LT. C(7)) GO TO 100
C IND = -1
C RETURN
C 100 CONTINUE
C C CALCULATE SLOPE (ADDING 1 TO NO OF FCN EVALS) IF IND .NE. 6
C IF (IND .NE. 6) GO TO 105
C CALL FCN(N, X, Y, W(1:1))
C C(24) = C(24) + 1.D0
C 105 CONTINUE
C C CALCULATE HMIN - USE DEFAULT UNLESS VALUE PRESCRIBED
C C(13) = C(13)
C IF (C(3) .NE. 0.D0) GO TO 165
C CALCULATE DEFAULT VALUE OF HMIN
C FIRST CALCULATE WEIGHTED NORM Y = C(12) - AS SPECIFIED
C BY THE ERROR CONTROL INDICATOR C(1)
C TEMP = 0.D0
C IF (C(1) .NE. 1.D0) GO TO 115
C ABSOLUTE ERROR CONTROL - WEIGHTS ARE 1
C DO 110 K = 1, N
C TEMP = DMAX1(TEMP, DABS(Y(K)))
C 110 CONTINUE
C C(12) = TEMP
C GO TO 160
C 115 IF (C(1) .NE. 2.D0) GO TO 120
C RELATIVE ERROR CONTROL - WEIGHTS ARE 1/DABS(Y(K)) SO
C WEIGHTED NORM Y IS 1
C C(12) = 1.D0
C GO TO 160
C 120 IF (C(1) .NE. 3.D0) GO TO 130
C WEIGHTS ARE 1/MAX(C(2),ABS(Y(K))).
C DO 125 K = 1, N
C TEMP = DMAX1(TEMP, DABS(Y(K))/C(2))
C 125 CONTINUE
C C(12) = DMIN1(TEMP, 1.D0)
GO TO 160
130 IF (C(1) .NE. 4.D0) GO TO 140
   WEIGTHS ARE 1/MAX(C(K+30),ABS(Y(K)))
   DO 135 K = 1, N
       TEMP = DMAX1(TEMP, DABS(Y(K))/C(K+30))
   CONTINUE
C(12) = DMIN1(TEMP, 1.D0)
GO TO 160
140 IF (C(1) .NE. 5.D0) GO TO 150
   WEIGTHS ARE 1/C(K+30)
   DO 145 K = 1, N
       TEMP = DMAX1(TEMP, DABS(Y(K))/C(K+30))
   CONTINUE
C(12) = TEMP
GO TO 160
150 CONTINUE
C(12) = DMIN1(TEMP, 1.D0)
160 CONTINUE
C(13) = 10.D0*DMAX1(C(11),C(10)*DMAX1(C(12)/TOL,DABS(X)))
165 CONTINUE
C CALCULATE SCALE - USE DEFAULT UNLESS VALUE PRESCRIBED
C(15) = C(5)
IF (C(5) .EQ. 0.D0) C(15) = 1.D0
C CALCULATE HMAX - CONSIDER 4 CASES
C CASE 1 BOTH HMAX AND SCALE PRESCRIBED
IF (C(5) .NE. 0.D0 .AND. C(5) .NE. 0.D0)
   C(16) = DMIN1(C(6), 2.D0/C(5))
C CASE 2 - HMAX PRESCRIBED, BUT SCALE NOT
IF (C(5) .NE. 0.D0 .AND. C(5) .EQ. 0.D0) C(16) = C(5)
C CASE 3 - HMAX NOT PRESCRIBED, BUT SCALE IS
IF (C(6) .EQ. 0.D0 .AND. C(5) .NE. 0.D0) C(16) = 2.D0/C(5)
C CASE 4 - NEITHER HMAX NOR SCALE IS PROVIDED
IF (C(6) .EQ. 0.D0 .AND. C(5) .EQ. 0.D0) C(16) = 2.D0
C
C************ERROR RETURN (WITH IND=-2) IF HMIN .GT. HMAX
IF (C(13) .LE. C(16)) GO TO 170
IND = -2
RETURN
170 CONTINUE
C CALCULATE PRELIMINARY HMAG - CONSIDER 3 CASES
IF (IND .GT. 2) GO TO 175
C CASE 1 - INITIAL ENTRY - USE PRESCRIBED VALUE OF HSTART, IF
ANY, ELSE DEFAULT
C(14) = C(4)
IF (C(4) .EQ. 0.D0) C(14) = C(16)*TOL**(1./6.)
GO TO 185
175 IF (C(23) .GT. 1.D0) GO TO 180
C CASE 2 - AFTER A SUCCESSFUL STEP, OR AT MOST ONE FAILURE,
C USE MIN(2, .9*(TOL/EST)***(1/6.))*HMAG, BUT AVOID POSSIBLE
C OVERFLOW. THEN AVOID REDUCTION BY MORE THAN HALF.
   TEMP = 2.D0*C(14)
IF (TOL .LT. (2.0D0/900)**6*C(19))
\[ \text{TEMP} = .9D0*(TOL/C(19))**(1./6.)*C(14) \]
C(14) = DMAX1(TEMP, .5D0*C(14))
GO TO 185

180 CONTINUE
C CASE 3 - AFTER TWO OR MORE SUCCESSIVE FAILURES
C C(14) = .5D0*C(14)
185 CONTINUE
C CHECK AGAINST HMAX
C C(14) = DMINT(C(14), C(16))
C CHECK AGAINST HMIN
C C(14) = DMAX1(C(14), C(13))
C
C************ INTERRUPT NO 1 (WITH IND=4) IF REQUESTED
IF (C(9) .GE. EQ.0.0D0) GO TO 1111
IND = 4
RETURN
1111 CONTINUE
C RESUME HERE ON RE-ENTRY WITH IND .EQ. 4 .......RE-ENTRY....
C CALCULATE HMAG, XTRIAL - DEPENDING ON PRELIMINARY HMAG, XEND
IF (C(14) .GE. DABS(XEND - X)) GO TO 190
DO NOT STEP MORE THAN HALF WAY TO XEND
C(14) = DMINT(C(14), .5D0*DABS(XEND - X))
C(17) = X + DSIGN(C(14), XEND - X)
GO TO 195
190 CONTINUE
HIT XEND EXACTLY
C(14) = DABS(XEND - X)
C(17) = XEND
195 CONTINUE
C CALCULATE HTrial
C C(18) = C(17) - X
C
END STAGE 1
C
C*****************************************************************************
C * STAGE 2 - CALCULATE YTRIAL [ADDDING 7 TO NO OF FCN EVALS]. *
C * W(*)2), ... W(*)8) HOLD INTERMEDIATE RESULTS NEEDED IN *
C * STAGE 3. W(*)9) IS TEMPORARY STORAGE UNTIL FINALLY IT HOLDS *
C * YTRIAL. *
C*****************************************************************************
C TEMP = C(18)/13981690800000.0D0
DO 200 K = 1, N
W(K,9) = Y(K) + TEMP*W(K,1)*233028180000.0D0
200 CONTINUE
CALL FCN(N, X + C(18)/6.0D0, W(1,9), W(1,2))
C DO 205 K = 1, N
W(K,9) = Y(K) + TEMP*( W(K,1)*74569017600.0D0
\[ + W(K,2)*298276070400.0D0 \])
205 CONTINUE
CALL FCN(N, X + C(18)*(4.0D0/15.0D0), W(1,9), W(1,3))
C
DO 210 K = 1, N
   W(K,9) = Y(K) + TEMP* (  W(K,1)*1.165140900000.D0
      +  W(K,2)*3.728450880000.D0
      +  W(K,3)*5.495422700000.D0 )
+  CONTINUE
210
CALL FCN(N, X + C(18)*(2.0D0/3.0D0), W(1,9), W(1,4))
C
DO 215 K = 1, N
   W(K,9) = Y(K) + TEMP* (  - W(K,1)*3.6046545659375.D0
      +  W(K,2)*1.2816549900000.D0
      -  W(K,3)*9.284716546875.D0
      +  W(K,4)*1.237962206250.D0 )
+  CONTINUE
215
CALL FCN(N, X + C(18)*(5.0D0/6.0D0), W(1,9), W(1,5))
C
DO 220 K = 1, N
   W(K,9) = Y(K) + TEMP* (  W(K,1)*3.355605792000.D0
      -  W(K,2)*1.118535264000.D0
      +  W(K,3)*9.172628850000.D0
      -  W(K,4)*4.27218330000.D0
      +  W(K,5)*4.82505408000.D0 )
+  CONTINUE
220
CALL FCN(N, X + C(18), W(1,9), W(1,6))
C
DO 225 K = 1, N
   W(K,9) = Y(K) + TEMP* (  - W(K,1)*7.70204740536.D0
      +  W(K,2)*2.311639545600.D0
      -  W(K,3)*1.322092233000.D0
      -  W(K,4)*4.53006781920.D0
      +  W(K,5)*3.26875481855.D0 )
+  CONTINUE
225
CALL FCN(N, X + C(18)/15.D0, W(1,9), W(1,7))
C
DO 230 K = 1, N
   W(K,9) = Y(K) + TEMP* (  W(K,1)*2.84592438900.D0
      +  W(K,2)*9.75468000000.D0
      +  W(K,3)*7.89711037500.D0
      -  W(K,4)*1.9208266000.D0
      +  W(K,5)*4.0002987600.D0
      +  W(K,7)*2.0158600000.D0 )
+  CONTINUE
230
CALL FCN(N, X + C(18), W(1,9), W(1,8))
C
CALCULATE YTIAL, THE EXTRAPOLATED APPROXIMATION AND STORE
IN W(*,9)
C
DO 235 K = 1, N
   W(K,9) = Y(K) + TEMP* (  W(K,1)*1.04862681000.D0
      +  W(K,3)*5.45186250000.D0
      +  W(K,4)*4.46637345000.D0
      +  W(K,5)*1.88806464000.D0
      +  W(K,7)*1.507687500.D0
      +  W(K,8)*9.759946500.D0 )
+  CONTINUE
235
C
ADD 7 TO THE NO OF FCN EVALS
C(24) = C(24) + 7.D0
C
C
END STAGE 2
**CALCULATE THE UNWEIGHTED ABSOLUTE ERROR ESTIMATE VECTOR**

DO 300 K = 1, N
    W(K,2) = ( W(K,1) + W(K,3) + W(K,4) + W(K,5) + W(K,6) + W(K,7) + W(K,8) ) / N

300 CONTINUE

**CALCULATE THE WEIGHTED MAX NORM OF W(*,2) AS SPECIFIED BY THE ERROR CONTROL INDICATOR C(1)**

TEMP = 0.D0

IF (C(1) .NE. 1.D0) GO TO 310

ABSOLUTE ERROR CONTROL

DO 305 K = 1, N
    TEMP = DMAX1(TEMP, DABS(W(K,2)))

305 CONTINUE

GO TO 360

310 IF (C(1) .NE. 2.D0) GO TO 320

RELATIVE ERROR CONTROL

DO 315 K = 1, N
    TEMP = DMAX1(TEMP, DABS(W(K,2)/Y(K)))

315 CONTINUE

GO TO 360

320 IF (C(1) .NE. 3.D0) GO TO 330

WEIGHTS ARE 1/ABS(Y(K))

DO 325 K = 1, N
    TEMP = DMAX1(TEMP, DABS(W(K,2)) / DMAX1(C(2), DABS(Y(K))))

325 CONTINUE

GO TO 360

330 IF (C(1) .NE. 4.D0) GO TO 340

WEIGHTS ARE 1/ABS(Y(K))

DO 335 K = 1, N
    TEMP = DMAX1(TEMP, DABS(W(K,2)) / DMAX1(C(K+30), DABS(Y(K))))

335 CONTINUE

GO TO 360

340 IF (C(1) .NE. 5.D0) GO TO 350

WEIGHTS ARE 1/C(K+30)

DO 345 K = 1, N
    TEMP = DMAX1(TEMP, DABS(W(K,2)/C(K+30)))

345 CONTINUE

GO TO 360
350 CONTINUE
C DEFAULT CASE – WEIGHTS ARE 1/\text{MAX}(1,\text{ABS}(Y(K)))
DO 355 K = 1, N
  TEMP = DMAX1(TEMP, DABS(W(K,2)))
  TEMP = DMAX1(TEMP, DABS(Y(K)))
+ CONTINUE
360 CONTINUE
C
C CALCULATE EST = (THE WEIGHTED MAX NORM OF W(*,2))*HMAG*SCALE
C – EST IS INTENDED TO BE A MEASURE OF THE ERROR PER UNIT
C STEP IN YTRIAL
C(19) = TEMP*C(14)*C(15)
C
C END STAGE 3
C
******************************************************************************
C * STAGE 4 – MAKE DECISIONS *
******************************************************************************
C
SET IND=5 IF STEP ACCEPTABLE, ELSE SET IND=6
IND = 5
IF (C(19) .GT. TOL) IND = 6
C
C********** INTERRUPT NO 2 IF REQUESTED
C IF (C(9) .EQ. 0.00) G0 TO 2222
C RETURN
C
C 2222 RESUME HERE ON RE-ENTRY WITH IND .EQ. 5 OR 6 ...
C
C CONTINUE
C
C IF (IND .EQ. 6) G0 TO 410
C
C STEP ACCEPTED (IND .EQ. 5), SO UPDATE X, Y FROM XTRIAL,
C XTRIAL, ADD 1 TO THE NO OF SUCCESSFUL STEPS, AND SET
C THE NO OF SUCCESSIVE FAILURES TO ZERO
C X = C(17)
DO 400 K = 1, N
  Y(K) = W(K,9)
400 CONTINUE
C(22) = C(22) + 1.00
C(23) = 0.00
C
C************ RETURN(WITH IND=3, XEND SAVED, FLAG SET) IF X .EQ. XEND
C IF (X .NE. XEND) G0 TO 405
C
C 405 CONTINUE
C GO TO 420
C
C CONTINUE
C
C STEP NOT ACCEPTED (IND .EQ. 6), SO ADD 1 TO THE NO OF
C SUCCESSIVE FAILURES
C C(23) = C(23) + 1.00
C
C************ ERROR RETURN (WITH IND=-3) IF HMAG .LE. HMIN
C IF (C(14) .GT. C(13)) G0 TO 415
C
C RETURN
C
C 415 CONTINUE
C 420 CONTINUE
C
C END STAGE 4
C
GO TO 99999
C
END LOOP
C
C BEGIN ABORT ACTION
500 CONTINUE
C
WRITE(6,505) IND, TOL, X, N, C(13), XEND, NW, C(16), C(20),
+ C(22), C(23), C(24), (Y(K), K = 1, N)
505 FORMAT( /* 1H0, 5BHCOMPUTATION STOPPED IN DVERK WITH THE FOLLOWING
* G VALUES —
+ / 1H0, 5HIND =, I4, 5X, 6HTOL =, 1PD13.6, 5X, 11HX =
+ 1PD22.15
+ / 1H + 5HN =, I4, 5X, 5HHMIN =, 1PD13.6, 5X, 11HSEND =
+ 1PD22.15
+ / 1H + 5HNW =, I4, 5X, 6HHMAX =, 1PD13.6, 5X, 11HMREV XEND =
+ 1PD22.15
+ / 1H0, 14X, 27HND OF SUCCESSFUL STEPS =, 0PF8.0
+ / 1H + 14X, 27HND OF SUCCESSIVE FAILURES =, 0PF8.0
+ / 1H + 14X, 27HND OF FUNCTION EVALS =, 0PF8.0
+ / 1H0, 23HTHE COMPONENTS OF Y ARE
+ /* (1H, 1P5D24.15)
C
STOP
C
C END ABORT ACTION
C
END