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[1,2,3] 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 FCN1
C
C     INITIALIZATION FOR THE PREDATOR-PREY PROBLEM
      N  = 2 ;    X    = 0.D0  ;    Y(1) = 1.D0 ;    IND = 1
      NW = 2 ;    TOL  = 1.D-6 ;    Y(2) = 3.D0
C
C     OUTPUT HEADING AND INITIAL VALUES
      WRITE(6,1) X, Y(1), Y(2)
    1 FORMAT(1H1, 17HOUTPUT FROM TEST1
     +       / 1H0, 5X, 3H X , 5X, 4HY(1), 11X, 4HY(2)
     +       / 1H0, 5X, F3.0, 1P2D15.6)
C
C     CALCULATE AND OUTPUT THE SOLUTION AT X = 1, 2, ...., 10
      DO 10 K = 1, 10
         XEND = DFLOAT(K)
         CALL DVERK(N, FCN1, X, Y, XEND, TOL, IND, C, NW, W)
C        CHECK IND .EQ. 3 - OTHERWISE TAKE APPROPRIATE ACTION
         WRITE(6,2) X, Y(1), Y(2)
    2    FORMAT(1H , 5X, F3.0, 1P2D15.6)
   10 CONTINUE
C
      RETURN
      END
```

```
OUTPUT FROM TEST1

      X      Y(1)           Y(2)

      0.     1.000000D+00   3.000000D+00
      1.     7.734528D-02   1.464449D+00
      2.     8.497744D-02   5.779538D-01
      3.     2.908908D-01   2.492533D-01
      4.     1.446599D+00   1.872189D-01
      5.     4.051460D+00   1.439479D+00
      6.     1.756181D-01   2.258599D+00
      7.     6.531117D-02   9.088004D-01
      8.     1.472268D-01   3.667180D-01
      9.     6.505956D-01   1.875748D-01
     10.     3.144327D+00   3.488206D-01
```

Figure 1 A table of solutions, at equally spaced values of X, is obtained for the predator - prey problem: $Y_1'=2Y_1(1-Y_2)$, $Y_2'=Y_2(1-Y_1)$, $Y_1(0)=1$, $Y_2(0)=3$. (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).)

```
      SUBROUTINE TEST2
      INTEGER N, NW, IND, K
      DOUBLE PRECISION X, XEND, Y(2), YTRUE(2), TOL, C(24), W(2,9),
     +       ERR, RATIO
      EXTERNAL FCN1
C
C     CALCULATE AN ACCURATE SOLUTION TO THE PREDATOR-PREY PROBLEM
      N  = 2 ;   X    = 0.D0  ;   YTRUE(1) = 1.D0 ;   TOL = 1.D-12
      NW = 2 ;   XEND = 10.D0 ;   YTRUE(2) = 3.D0 ;   IND = 1
C
      CALL DVERK(N, FCN1, X, YTRUE, XEND, TOL, IND, C, NW, W)
C     CHECK IND .EQ. 3 - OTHERWISE TAKE APPROPRIATE ACTION
C
C     OUTPUT HEADING
      WRITE(6,1)
    1 FORMAT(1H1, 17HOUTPUT FROM TEST2
     +     / 1H0, 7X, 3HTOL, 7X, 4HY(1), 14X, 4HY(2), 12X
     +        8HNORM ERR, 4X, 5HRATIO / )
C
C     CALCULATE AND OUTPUT THE SOLUTION, THE MAX NORM OF THE ERROR,
C     AND THE RATIO OF THIS NORM TO TOL FOR SEVERAL VALUES OF TOL
      DO 10 K = 1, 9
         X   = 0.D0 ;   Y(1) = 1.D0 ;   TOL = 10.D0**(-K)
         IND = 1    ;   Y(2) = 3.D0
         CALL DVERK(N, FCN1, X, Y, XEND, TOL, IND, C, NW, W)
C        CHECK IND .EQ. 3 - OTHERWISE TAKE APPROPRIATE ACTION
         ERR = DMAX1(DABS(YTRUE(1) - Y(1)), DABS(YTRUE(2) - Y(2)))
         RATIO = ERR / TOL
         WRITE(6,2) TOL, Y(1), Y(2), ERR, RATIO
    2    FORMAT(1H , 5X, 1PD7.1, 1P2D18.9, 1PD12.3, 0PF8.1)
   10 CONTINUE
C
      RETURN
      END
```

```
OUTPUT FROM TEST2

      TOL         Y(1)                Y(2)              NORM ERR    RATIO

    1.0D-01   2.772217043D+00   1.507458648D+00   1.159D+00    11.6
    1.0D-02   3.402005791D+00   5.996577983D-01   2.577D-01    25.8
    1.0D-03   3.084290867D+00   3.381031955D-01   6.005D-02    60.0
    1.0D-04   3.145229495D+00   3.492255920D-01   8.927D-04     8.9
    1.0D-05   3.144100384D+00   3.488103977D-01   2.364D-04    23.6
    1.0D-06   3.144305803D+00   3.488140901D-01   3.099D-05    31.0
    1.0D-07   3.144333374D+00   3.488185406D-01   3.416D-06    34.2
    1.0D-08   3.144336480D+00   3.488191083D-01   3.106D-07    31.1
    1.0D-09   3.144336768D+00   3.488191599D-01   2.187D-08    21.9
```

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 straight forward, 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 <u>and</u> 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.D0/
      EXTERNAL FCN2
C
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.D0 ;   IND  = 2   ;   TOL = 1.D-6
      NW = 1 ;   Y(1) = 1.D0 ;   C(1) = 1.D0
C
C     OUTPUT HEADING
      WRITE(6,1)
    1 FORMAT(1H1, 18HOUTPUT FROM TEST3A
     +      / 1H0, 20X, 22HABSOLUTE ERROR CONTROL
     +      / 1H0, 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     LOOP
   10    XEND = X + 5.D0
         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 =, 1PD13.6)
C     ......EXIT LOOP
            GO TO 30
   20    CONTINUE
         RELERR = Y(1)/DEXP(X) - 1.D0
         WRITE(6,3) X, Y(1), RELERR, C(24)
    3    FORMAT(1H , 5X, F3.0, 1P2D15.6, 0PF12.0)
         IF (XEND .LT. 50.D0) GO TO 10
C     END LOOP
   30 CONTINUE
C
      RETURN
      END
```

```
OUTPUT FROM TEST3A

                     ABSOLUTE ERROR CONTROL

     X       Y(1)            RELATIVE ERR      NO FCN EVALS

     5.    1.484132D+02   -2.457570D-08         184.
    10.    2.202647D+04   -2.471737D-08         616.
    15.    3.269017D+06   -2.471816D-08        1632.

ERROR RETURN WITH IND = -3 AT X = 16.212, Y = 1.098433D+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).)

```
      SUBROUTINE TEST3B
      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
C     CALCULATE THE SOLUTION TO THE PROBLEM Y' = Y, Y = 1 AT X = 0
C     USING THE RELATIVE ERROR CONTROL OPTION
      N  = 1 ;    X    = 0.D0 ;    IND  = 2    ;    TOL = 1.D-6
      NW = 1 ;    Y(1) = 1.D0 ;    C(1) = 2.D0
C
C     OUTPUT HEADING
      WRITE(6,1)
    1 FORMAT(1H1, 18HOUTPUT FROM TEST3B
     +      / 1H0, 20X, 22HRELATIVE ERROR CONTROL
     +      / 1H0, 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
      DO 10 K = 5, 50, 5
         XEND = DFLOAT(K)
         CALL DVERK(N, FCN2, X, Y, XEND, TOL, IND, C, NW, W)
C        CHECK IND .EQ. 3 - OTHERWISE TAKE APPROPRIATE ACTION
         RELERR = Y(1)/DEXP(X) - 1.D0
         WRITE(6,2) X, Y(1), RELERR, C(24)
    2    FORMAT(1H , 5X, F3.0, 1P2D15.6, 0PF12.0)
   10 CONTINUE
C
      RETURN
      END
```

```
OUTPUT FROM TEST3B

                    RELATIVE ERROR CONTROL

     X        Y(1)           RELATIVE ERR      NO FCN EVALS

     5.    1.484131D+02   -9.052112D-08         128.
    10.    2.202646D+04   -1.863018D-07         256.
    15.    3.269016D+06   -2.818623D-07         384.
    20.    4.851650D+08   -3.774255D-07         512.
    25.    7.200487D+10   -4.729887D-07         640.
    30.    1.068647D+13   -5.685518D-07         768.
    35.    1.586012D+15   -6.641149D-07         896.
    40.    2.353851D+17   -7.596780D-07        1024.
    45.    3.493424D+19   -8.552412D-07        1152.
    50.    5.184701D+21   -9.508043D-07        1280.
```

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.D0/
      EXTERNAL FCN1
C
C     CALCULATE AN ACCURATE SOLUTION TO THE PREDATOR-PREY PROBLEM
C     USING THE RELATIVE ERROR CONTROL OPTION
      N   = 2 ;   X    = 0.D0    ;   YTRUE(1) = 1.D0
      NW  = 2 ;   XEND = 20.D0   ;   YTRUE(2) = 7.D0
      IND = 2 ;   TOL  = 1.D-12  ;   C(1)     = 2.D0
C
      CALL DVERK(N, FCN1, X, YTRUE, XEND, TOL, IND, C, NW, W)
C     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
      X   = 0.D0  ;   Y(1) = 1.D0 ;   IND = 1
      TOL = 1.D-3 ;   Y(2) = 7.D0
C
      CALL DVERK(N, FCN1, X, Y, XEND, TOL, IND, C, NW, W)
C     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(6,1) Y(1), ERR(1), Y(2), ERR(2)
    1 FORMAT(1H1, 18HOUTPUT FROM TEST4A
     +      / 1H0, 7X, 4HY(1), 11X, 6HERR(1), 8X, 4HY(2), 11X, 6HERR(2)
     +      / 1H0, 2(1PD17.6, 1PD12.3) )
C
      RETURN
      END
```

```
OUTPUT FROM TEST4A

       Y(1)            ERR(1)         Y(2)           ERR(2)

    9.078123D-03  -8.909D-03    5.582552D+00  -3.786D+00
```

Figure 4A The predator-prey problem is solved once more, but with different initial conditions: $Y_1(0)=1$, $Y_2(0)=7$. The value of Y_1 is very small over significant intervals of the integration. A "true" solution, 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.D0/
      EXTERNAL FCN1
C
C     CALCULATE AN ACCURATE SOLUTION TO THE PREDATOR-PREY PROBLEM
C     USING THE RELATIVE ERROR CONTROL OPTION
      N   = 2 ;   X    = 0.D0   ;   YTRUE(1) = 1.D0
      NW  = 2 ;   XEND = 20.D0  ;   YTRUE(2) = 7.D0
      IND = 2 ;   TOL  = 1.D-12 ;   C(1)     = 2.D0
C
      CALL DVERK(N, FCN1, X, YTRUE, XEND, TOL, IND, C, NW, W)
C     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.D-4 AND 1.D0
      X   = 0.D0 ;   Y(1) = 1.D0 ;   IND  = 2    ;   C(31) = 1.D-4
      TOL = 1.D-3 ;  Y(2) = 7.D0 ;   C(1) = 4.D0 ;   C(32) = 1.D0
C
      CALL DVERK(N, FCN1, X, Y, XEND, TOL, IND, C, NW, W)
C     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(6,1) Y(1), ERR(1), Y(2), ERR(2)
    1 FORMAT(1H1, 18HOUTPUT FROM TEST4B
     +     / 1H0, 7X, 4HY(1), 11X, 6HERR(1), 8X, 4HY(2), 11X, 6HERR(2)
     +     / 1H0, 2(1PD17.6, 1PD12.3) )
C
      RETURN
      END
```

```
OUTPUT FROM TEST4B

       Y(1)           ERR(1)         Y(2)           ERR(2)

   1.693960D-04   -5.467D-07    1.853165D+00   -5.645D-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, .5D0, 1.D0, 2.D0, 4.D0 / ,
     +      TOL   / 1.D-5, 1.D-7, 1.D-9 / ,
     +      C(1), C(2), C(3), C(4), C(5), C(6), C(7), C(8), C(9) /9*0.D0/
      EXTERNAL FCN2
C
C     CALCULATE THE SOLUTION TO THE PROBLEM Y' = Y, Y = 1 AT X = 0
C     USING SEVERAL VALUES OF SCALE AND TOL
      N  = 1 ;    XEND  = 10.D0      :    C(4) = 2.5D0
      NW = 1 ;    YTRUE = DEXP(XEND)
C
C     OUTPUT HEADING
      WRITE(6,1) (TOL(I), I = 1, 3)
    1 FORMAT(1H1, 70HOUTPUT FROM TESTS - RELATIVE ERROR FOR SEVERAL VALU
     +ES OF TOL AND SCALE
     +      / 1H0, 5X, 5HSCALE, 3(5X, 5HTOL =, 1PD8.1) / )
C
C     FOR EACH VALUE OF SCALE AND TOL CALCULATE THE SOLUTION TO THE
C     PROBLEM AND OUTPUT ITS RELATIVE ERROR
      DO 20 K = 1, 6
         C(5) = SCALE(K)
         DO 10 I = 1, 3
            X = 0.D0 ;    Y(1) = 1.D0 ;    IND = 2
            CALL DVERK(N, FCN2, X, Y, XEND, TOL(I), IND, C, NW, W)
C           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, 1P3D18.6)
   20 CONTINUE
C
      RETURN
      END
```

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

SCALE	TOL = 1.0D-05	TOL = 1.0D-07	TOL = 1.0D-09
0.125	-4.922995D-03	-4.909518D-03	-4.909404D-03
0.250	-4.915586D-03	-4.909459D-03	-4.909404D-03
0.500	-4.912483D-03	-4.909431D-03	-4.909404D-03
1.000	-2.160714D-06	-1.836273D-08	-1.675572D-10
2.000	-1.016184D-06	-9.002836D-09	-8.237690D-11
4.000	-5.190714D-07	-4.451122D-09	-4.117786D-11

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.D0/
      EXTERNAL FCN1
C
C     CALCULATE THE SOLUTION TO THE PREDATOR-PREY PROBLEM BUT LIMIT
C     THE NUMBER OF FUNCTION EVALUATIONS TO 100
      N   = 2 ;    X    = 0.D0 ;    XEND = 10.D0
      NW  = 2 ;    Y(1) = 1.D0 ;    TOL  = 1.D-6
      IND = 2 ;    Y(2) = 3.D0 ;    C(7) = 100.D0
C
      CALL DVERK(N, FCN1, X, Y, XEND, TOL, IND, C, NW, W)
C
C     OUTPUT HEADING AND RESULTS
      WRITE(6,1) IND, X, Y(1), Y(2), C(24)
    1 FORMAT(1H1, 17HOUTPUT FROM TEST6
     +      / 1H0, 5X, 37HTHE INTEGRATION TERMINATED WITH IND =, I3,
     +        7H AT X =, F8.5
     +      / 1H , 5X, 10HAND Y(1) =, 1PD12.5, 8H   Y(2) =, 1PD12.5
     +      / 1H0, 5X, 43HTNE NUMBER OF FUNCTION EVALUATIONS USED WAS,
     +        0PF5.0)
C
C     RE-CALL DVERK TO CAUSE AN ABORT
      CALL DVERK(N, FCN1, 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) = 1.89519D-01   Y(2) = 3.12799D-01

      TNE NUMBER OF FUNCTION EVALUATIONS USED WAS 103.



COMPUTATION STOPPED IN DVERK WITH THE FOLLOWING VALUES -

IND =  -1      TOL  = 1.0000000D-06      X           = 2.702816859719279D+00
N   =   2      HMIN = 9.7958790D-10      XEND        = 1.000000000000000D+01
NW  =   2      HMAX = 2.0000000D+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

   1.895187185023556D-01   3.127986715040333D-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, HMAG, of the trial step-size. However, the final value of HMAG 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 HMAG, it will not attempt to go more than half-way to XEND. The corresponding value of HTRIAL may also differ from HMAG 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 TEST7
      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.D0/
      EXTERNAL FCN2
C
      N   = 1 ;   X    = 0.D0 ;   TOL  = 1.D-6 ;   EPS = 5.D-4
      NW  = 1 ;   XEND = 1.D2 ;   C(1) = 2.D0
      IND = 2 ;   Y(1) = 1.D0 ;   C(9) = 1.D0
C
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.D0)) GO TO 20
         GO TO 10
   20 CONTINUE
C
      XLOW = X ;   XHIGH = C(17) ;   IND = 6
C
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.D0
         CALL DVERK(N, FCN2, X, Y, XEND, TOL, IND, C, NW, W)
         IF ((IND.NE.5) .OR. (W(1,9).LT.100.D0)) GO TO 40
C           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.D0)) GO TO 50
C           YTRIAL ACCEPTABLE AND LT 100 - RESET XLOW AND ACCEPT STEP
            XLOW = C(17)
   50    CONTINUE
         GO TO 30
   60 CONTINUE
C
C     OUTPUT THE APPROXIMATE SOLUTION TO THE EQUATION EXP(X) = 100
      ROOT = (XHIGH + XLOW) / 2.D0
      WRITE(6,1) ROOT
    1 FORMAT(1H1, 17HOUTPUT FROM TEST7
     +     / 1H0, 5X, 45HTHE SOLUTION OF EXP(X) = 100 IS APPROXIMATELY,
     +       F6.3 )
C
      RETURN
      END
```

```
OUTPUT FROM TEST7

     THE SOLUTION OF EXP(X) = 100 IS APPROXIMATELY 4.605
```

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:

```
      CALL TEST1
      CALL TEST2
      CALL TEST3A
      CALL TEST3B
      CALL TEST4A
      CALL TEST4B
      CALL TEST5
      CALL TEST7
      CALL TEST6
C
      STOP
      END
```

```
SUBROUTINE FCN1(N, X, Y, YP)
INTEGER N
DOUBLE PRECISION X, Y(N), YP(N)
YP(1) = 2.D0 * Y(1) * (1.D0 - Y(2))
YP(2) = Y(2) * (Y(1) - 1.D0)
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 Equations, SIAM J. Numer. Anal. (Dec., 1976), to appear.

(2) W. H. Enright, T. E. Hull and B. Lindberg, Comparing Numerical Methods for Stiff Systems of O.D.E:s, BIT 15, 1 (1975), 10-48.

(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***********************************************************************
C                                                                      *
C     PURPOSE - THIS IS A RUNGE-KUTTA  SUBROUTINE  BASED  ON  VERNER'S *
C FIFTH AND SIXTH ORDER PAIR OF FORMULAS FOR FINDING APPROXIMATIONS TO *
C THE SOLUTION OF  A  SYSTEM  OF  FIRST  ORDER  ORDINARY  DIFFERENTIAL *
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.                                               *
C                                                                      *
C                                    HULL-ENRIGHT-JACKSON   1/10/76    *
C                                                                      *
C***********************************************************************
C                                                                      *
C     USE - THE USER MUST SPECIFY EACH OF THE FOLLOWING                *
C                                                                      *
C     N  NUMBER OF EQUATIONS                                           *
C                                                                      *
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) *
C             IS PROPORTIONAL TO TOL.  (THE NORM  IS  A  MAX  NORM  WITH *
C             WEIGHTS  THAT  DEPEND ON THE ERROR CONTROL STRATEGY CHOSEN *
C             BY THE USER.  THE DEFAULT WEIGHT FOR THE K-TH COMPONENT IS *
C             1/MAX(1,ABS(Y(K))),  WHICH THEREFORE PROVIDES A MIXTURE OF *
C             ABSOLUTE AND RELATIVE ERROR CONTROL.)                      *
C                                                                        *
C     IND   INDICATOR - ON INITIAL ENTRY IND MUST BE SET EQUAL TO  EITHER *
C             1  OR  2. IF THE USER DOES NOT WISH TO USE ANY OPTIONS, HE *
C             SHOULD SET IND TO 1 - ALL THAT REMAINS FOR THE USER TO  DO *
C             THEN  IS  TO  DECLARE C AND W, AND TO SPECIFY NW. THE USER *
C             MAY ALSO  SELECT  VARIOUS  OPTIONS  ON  INITIAL  ENTRY  BY *
C             SETTING IND = 2 AND INITIALIZING THE FIRST 9 COMPONENTS OF *
C             C AS DESCRIBED IN THE NEXT SECTION.  HE MAY ALSO  RE-ENTER *
C             THE  SUBROUTINE  WITH IND = 3 AS MENTIONED AGAIN BELOW. IN *
C             ANY EVENT, THE SUBROUTINE RETURNS WITH IND EQUAL TO        *
C                3 AFTER A NORMAL RETURN                                 *
C                4, 5, OR 6 AFTER AN INTERRUPT (SEE OPTIONS C(8), C(9))  *
C                -1, -2, OR -3 AFTER AN ERROR CONDITION (SEE BELOW)      *
C                                                                        *
C       C   COMMUNICATIONS VECTOR - THE DIMENSION MUST BE GREATER THAN OR *
C             EQUAL TO 24, UNLESS OPTION C(1) = 4 OR 5 IS USED, IN WHICH *
C             CASE THE DIMENSION MUST BE GREATER THAN OR EQUAL TO N+30   *
C                                                                        *
C      NW   FIRST DIMENSION OF WORKSPACE W -  MUST  BE  GREATER  THAN  OR *
C             EQUAL TO N                                                 *
C                                                                        *
C       W   WORKSPACE MATRIX - FIRST DIMENSION MUST BE NW AND SECOND MUST *
C             BE GREATER THAN OR EQUAL TO 9                              *
C                                                                        *
C        THE SUBROUTINE  WILL  NORMALLY  RETURN  WITH  IND  =  3,  HAVING *
C     REPLACED THE INITIAL VALUES OF X AND Y WITH, RESPECTIVELY, THE VALUE *
C     OF XEND AND AN APPROXIMATION TO Y AT XEND.  THE  SUBROUTINE  CAN  BE *
C     CALLED  REPEATEDLY  WITH NEW VALUES OF XEND WITHOUT HAVING TO CHANGE *
C     ANY OTHER ARGUMENT.  HOWEVER, CHANGES IN TOL, OR ANY OF THE  OPTIONS *
C     DESCRIBED BELOW, MAY ALSO BE MADE ON SUCH A RE-ENTRY IF DESIRED.   *
C                                                                        *
C        THREE ERROR RETURNS ARE ALSO POSSIBLE, IN WHICH  CASE  X  AND  Y *
C     WILL BE THE MOST RECENTLY ACCEPTED VALUES -                        *
C        WITH IND = -3 THE SUBROUTINE WAS UNABLE  TO  SATISFY  THE  ERROR *
C           REQUIREMENT  WITH A PARTICULAR STEP-SIZE THAT IS LESS THAN OR *
C           EQUAL TO HMIN, WHICH MAY MEAN THAT TOL IS TOO SMALL          *
C        WITH IND = -2 THE VALUE OF HMIN  IS  GREATER  THAN  HMAX,  WHICH *
C           PROBABLY  MEANS  THAT THE REQUESTED TOL (WHICH IS USED IN THE *
C           CALCULATION OF HMIN) IS TOO SMALL                            *
C        WITH IND = -1 THE ALLOWED MAXIMUM NUMBER OF FCN EVALUATIONS  HAS *
C           BEEN  EXCEEDED,  BUT  THIS  CAN ONLY OCCUR IF OPTION C(7), AS *
C           DESCRIBED IN THE NEXT SECTION, HAS BEEN USED                 *
C                                                                        *
C        THERE ARE SEVERAL CIRCUMSTANCES THAT WILL CAUSE THE CALCULATIONS *
C     TO  BE  TERMINATED,  ALONG WITH OUTPUT OF INFORMATION THAT WILL HELP *
C     THE USER DETERMINE THE CAUSE OF  THE  TROUBLE.  THESE  CIRCUMSTANCES *
C     INVOLVE  ENTRY WITH ILLEGAL OR INCONSISTENT VALUES OF THE ARGUMENTS, *
C     SUCH AS ATTEMPTING A NORMAL  RE-ENTRY  WITHOUT  FIRST  CHANGING  THE *
C     VALUE OF XEND, OR ATTEMPTING TO RE-ENTER WITH IND LESS THAN ZERO.  *
C                                                                        *
C*************************************************************************
C
```

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

```
C           IF C(6).NE.0 AND C(5).NE.0, HMAX IS TAKEN TO BE              *
C              MIN(ABS(C(6)),2/ABS(C(5)))                                *
C           IF C(6).NE.0 AND C(5).EQ.0, HMAX IS TAKEN TO BE  ABS(C(6))   *
C           IF C(6).EQ.0 AND C(5).NE.0, HMAX IS TAKEN TO BE              *
C              2/ABS(C(5))                                               *
C           IF C(6).EQ.0 AND C(5).EQ.0, HMAX IS GIVEN A DEFAULT  VALUE   *
C              OF 2                                                      *
C                                                                        *
C  C(7)  MAXIMUM NUMBER OF FUNCTION EVALUATIONS  -  IF  NOT  ZERO,  AN   *
C           ERROR  RETURN WITH IND = -1 WILL BE CAUSED WHEN THE NUMBER   *
C           OF FUNCTION EVALUATIONS EXCEEDS ABS(C(7))                    *
C                                                                        *
C  C(8)  INTERRUPT NUMBER  1  -  IF  NOT  ZERO,  THE  SUBROUTINE  WILL   *
C           INTERRUPT   THE  CALCULATIONS  AFTER  IT  HAS  CHOSEN  ITS   *
C           PRELIMINARY VALUE OF HMAG, AND JUST BEFORE CHOOSING HTRIAL   *
C           AND  XTRIAL  IN  PREPARATION FOR TAKING A STEP (HTRIAL MAY   *
C           DIFFER FROM HMAG IN SIGN, AND MAY  REQUIRE  ADJUSTMENT  IF   *
C           XEND  IS  NEAR) - THE SUBROUTINE RETURNS WITH IND = 4, AND   *
C           WILL RESUME CALCULATION AT THE POINT  OF  INTERRUPTION  IF   *
C           RE-ENTERED WITH IND = 4                                      *
C                                                                        *
C  C(9)  INTERRUPT NUMBER  2  -  IF  NOT  ZERO,  THE  SUBROUTINE  WILL   *
C           INTERRUPT   THE  CALCULATIONS  IMMEDIATELY  AFTER  IT  HAS   *
C           DECIDED WHETHER OR NOT TO ACCEPT THE RESULT  OF  THE  MOST   *
C           RECENT  TRIAL STEP, WITH IND = 5 IF IT PLANS TO ACCEPT, OR   *
C           IND = 6 IF IT PLANS TO REJECT -  Y(*)  IS  THE  PREVIOUSLY   *
C           ACCEPTED  RESULT, WHILE W(*,9) IS THE NEWLY COMPUTED TRIAL   *
C           VALUE, AND W(*,2) IS THE UNWEIGHTED ERROR ESTIMATE VECTOR.   *
C           THE  SUBROUTINE  WILL  RESUME CALCULATIONS AT THE POINT OF   *
C           INTERRUPTION ON RE-ENTRY WITH IND = 5 OR 6. (THE USER  MAY   *
C           CHANGE IND IN THIS CASE IF HE WISHES, FOR EXAMPLE TO FORCE   *
C           ACCEPTANCE OF A STEP THAT WOULD OTHERWISE BE REJECTED,  OR   *
C           VICE VERSA. HE CAN ALSO RESTART WITH IND = 1 OR 2.)          *
C                                                                        *
C*************************************************************************
C                                                                        *
C  SUMMARY OF THE COMPONENTS OF THE COMMUNICATIONS VECTOR                *
C                                                                        *
C     PRESCRIBED AT THE OPTION       DETERMINED BY THE PROGRAM           *
C            OF THE USER                                                 *
C                                                                        *
C                                    C(10) RREB(REL ROUNDOFF ERR BND)    *
C     C(1) ERROR CONTROL INDICATOR   C(11) DWARF (VERY SMALL MACH NO)    *
C     C(2) FLOOR VALUE               C(12) WEIGHTED NORM Y               *
C     C(3) HMIN SPECIFICATION        C(13) HMIN                          *
C     C(4) HSTART SPECIFICATION      C(14) HMAG                          *
C     C(5) SCALE SPECIFICATION       C(15) SCALE                         *
C     C(6) HMAX SPECIFICATION        C(16) HMAX                          *
C     C(7) MAX NO OF FCN EVALS       C(17) XTRIAL                        *
C     C(8) INTERRUPT NO 1            C(18) HTRIAL                        *
C     C(9) INTERRUPT NO 2            C(19) EST                           *
C                                    C(20) PREVIOUS XEND                 *
C                                    C(21) FLAG FOR XEND                 *
C                                    C(22) NO OF SUCCESSFUL STEPS        *
C                                    C(23) NO OF SUCCESSIVE FAILURES     *
C                                    C(24) NO OF FCN EVALS               *
C                                                                        *
C  IF C(1) = 4 OR 5, C(31), C(32), ... C(N+30) ARE FLOOR VALUES          *
```

```
C                                                                          *
C***************************************************************************
C                                                                          *
C  AN OVERVIEW OF THE PROGRAM                                              *
C                                                                          *
C     BEGIN INITIALIZATION, PARAMETER CHECKING, INTERRUPT RE-ENTRIES       *
C  ......ABORT IF IND OUT OF RANGE 1 TO 6                                  *
C  .     CASES - INITIAL ENTRY, NORMAL RE-ENTRY, INTERRUPT RE-ENTRIES      *
C  .     CASE 1 - INITIAL ENTRY (IND .EQ. 1 OR 2)                          *
C  V........ABORT IF N.GT.NW OR TOL.LE.0                                   *
C  .        IF INITIAL ENTRY WITHOUT OPTIONS (IND .EQ. 1)                  *
C  .           SET C(1) TO C(9) EQUAL TO ZERO                              *
C  .        ELSE INITIAL ENTRY WITH OPTIONS (IND .EQ. 2)                   *
C  .           MAKE C(1) TO C(9) NON-NEGATIVE                              *
C  .           MAKE FLOOR VALUES NON-NEGATIVE IF THEY ARE TO BE USED       *
C  .        END IF                                                         *
C  .        INITIALIZE RREB, DWARF, PREV XEND, FLAG, COUNTS                *
C  .     CASE 2 - NORMAL RE-ENTRY (IND .EQ. 3)                             *
C  .........ABORT IF XEND REACHED, AND EITHER X CHANGED OR XEND NOT        *
C  .        RE-INITIALIZE FLAG                                             *
C  .     CASE 3 - RE-ENTRY FOLLOWING AN INTERRUPT (IND .EQ. 4 TO 6)        *
C  V        TRANSFER CONTROL TO THE APPROPRIATE RE-ENTRY POINT.......      *
C  .     END CASES                                                 .       *
C  .  END INITIALIZATION, ETC.                                     .       *
C  .                                                               V       *
C  .  LOOP THROUGH THE FOLLOWING 4 STAGES, ONCE FOR EACH TRIAL STEP .       *
C  .     STAGE 1 - PREPARE                                         .       *
C***********ERROR RETURN (WITH IND=-1) IF NO OF FCN EVALS TOO GREAT .       *
C  .        CALC SLOPE (ADDING 1 TO NO OF FCN EVALS) IF IND .NE. 6 .       *
C  .        CALC HMIN, SCALE, HMAX                                 .       *
C***********ERROR RETURN (WITH IND=-2) IF HMIN .GT. HMAX           .       *
C  .        CALC PRELIMINARY HMAG                                  .       *
C***********INTERRUPT NO 1 (WITH IND=4) IF REQUESTED.......RE-ENTRY.V      *
C  .        CALC HMAG, XTRIAL AND HTRIAL                           .       *
C  .     END STAGE 1                                               .       *
C  V     STAGE 2 - CALC YTRIAL (ADDING 7 TO NO OF FCN EVALS)       .       *
C  .     STAGE 3 - CALC THE ERROR ESTIMATE                         .       *
C  .     STAGE 4 - MAKE DECISIONS                                  .       *
C  .        SET IND=5 IF STEP ACCEPTABLE, ELSE SET IND=6           .       *
C***********INTERRUPT NO 2 IF REQUESTED....................RE-ENTRY.V      *
C  .        IF STEP ACCEPTED (IND .EQ. 5)                                  *
C  .           UPDATE X, Y FROM XTRIAL, YTRIAL                             *
C  .           ADD 1 TO NO OF SUCCESSFUL STEPS                             *
C  .           SET NO OF SUCCESSIVE FAILURES TO ZERO                       *
C**************RETURN(WITH IND=3, XEND SAVED, FLAG SET) IF X .EQ. XEND     *
C  .        ELSE STEP NOT ACCEPTED (IND .EQ. 6)                            *
C  .           ADD 1 TO NO OF SUCCESSIVE FAILURES                          *
C**************ERROR RETURN (WITH IND=-3) IF HMAG .LE. HMIN                *
C  .        END IF                                                         *
C  .     END STAGE 4                                                       *
C  .  END LOOP                                                             *
C  .                                                                       *
C  BEGIN ABORT ACTION                                                      *
C     OUTPUT APPROPRIATE  MESSAGE  ABOUT  STOPPING  THE  CALCULATIONS,     *
C        ALONG WITH VALUES OF IND, N, NW, TOL, HMIN,  HMAX,  X,  XEND,     *
C        PREVIOUS XEND,  NO OF  SUCCESSFUL  STEPS,  NO  OF  SUCCESSIVE     *
C        FAILURES, NO OF FCN EVALS, AND THE COMPONENTS OF Y                *
C     STOP                                                                 *
C  END ABORT ACTION                                                        *
```

```
C                                                                          *
C**************************************************************************
C
C     *********************************************************************
C     * BEGIN INITIALIZATION, PARAMETER CHECKING, INTERRUPT RE-ENTRIES *
C     *********************************************************************
C
C  ......ABORT IF IND OUT OF RANGE 1 TO 6
         IF (IND.LT.1 .OR. IND.GT.6) GO TO 500
C
C        CASES - INITIAL ENTRY, NORMAL RE-ENTRY, INTERRUPT RE-ENTRIES
         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
    5       IF (N.GT.NW .OR. TOL.LE.0.D0) GO TO 500
            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
               DO 10 K = 1, 9
                  C(K) = 0.D0
   10          CONTINUE
               GO TO 35
   15       CONTINUE
C              INITIAL ENTRY WITH OPTIONS (IND .EQ. 2)
C              MAKE C(1) TO C(9) NON-NEGATIVE
               DO 20 K = 1, 9
                  C(K) = DABS(C(K))
   20          CONTINUE
C              MAKE FLOOR VALUES NON-NEGATIVE IF THEY ARE TO BE USED
               IF (C(1).NE.4.D0 .AND. C(1).NE.5.D0) GO TO 30
                  DO 25 K = 1, N
                     C(K+30) = DABS(C(K+30))
   25             CONTINUE
   30          CONTINUE
   35       CONTINUE
C           INITIALIZE RREB, DWARF, PREV XEND, FLAG, COUNTS
            C(10) = 16.D0**(-13)
            C(11) = 1.D-50
C           SET PREVIOUS XEND INITIALLY TO INITIAL VALUE OF X
            C(20) = X
            DO 40 K = 21, 24
               C(K) = 0.D0
   40       CONTINUE
            GO TO 50
C        CASE 2 - NORMAL RE-ENTRY (IND .EQ. 3)
C  .........ABORT IF XEND REACHED, AND EITHER X CHANGED OR XEND NOT
   45       IF (C(21).NE.0.D0 .AND.
     +                      (X.NE.C(20) .OR. XEND.EQ.C(20))) GO TO 500
C           RE-INITIALIZE FLAG
            C(21) = 0.D0
            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                                                     V
   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       ***************************************************************
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***********ERROR RETURN (WITH IND=-1) IF NO OF FCN EVALS TOO GREAT
            IF (C(7).EQ.0.D0 .OR. C(24).LT.C(7)) GO TO 100
               IND = -1
               RETURN
  100       CONTINUE
C
C           CALCULATE SLOPE (ADDING 1 TO NO OF FCN EVALS) IF IND .NE. 6
            IF (IND .EQ. 6) GO TO 105
               CALL FCN(N, X, Y, W(1,1))
               C(24) = C(24) + 1.D0
  105       CONTINUE
C
C           CALCULATE HMIN - USE DEFAULT UNLESS VALUE PRESCRIBED
            C(13) = C(3)
            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)
               TEMP = 0.D0
               IF (C(1) .NE. 1.D0) GO TO 115
C                 ABSOLUTE ERROR CONTROL - WEIGHTS ARE 1
                  DO 110 K = 1, N
                     TEMP = DMAX1(TEMP, DABS(Y(K)))
  110             CONTINUE
                  C(12) = TEMP
                  GO TO 160
  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(12) = 1.D0
                  GO TO 160
  120          IF (C(1) .NE. 3.D0) GO TO 130
C                 WEIGHTS ARE 1/MAX(C(2),ABS(Y(K))).
                  DO 125 K = 1, N
                     TEMP = DMAX1(TEMP, DABS(Y(K))/C(2))
  125             CONTINUE
                  C(12) = DMIN1(TEMP, 1.D0)
```

```
                  GO TO 160
  130          IF (C(1) .NE. 4.D0) GO TO 140
C                 WEIGHTS ARE 1/MAX(C(K+30),ABS(Y(K)))
                  DO 135 K = 1, N
                     TEMP = DMAX1(TEMP, DABS(Y(K))/C(K+30))
  135             CONTINUE
                  C(12) = DMIN1(TEMP, 1.D0)
                  GO TO 160
  140          IF (C(1) .NE. 5.D0) GO TO 150
C                 WEIGHTS ARE 1/C(K+30)
                  DO 145 K = 1, N
                     TEMP = DMAX1(TEMP, DABS(Y(K))/C(K+30))
  145             CONTINUE
                  C(12) = TEMP
                  GO TO 160
  150          CONTINUE
C                 DEFAULT CASE - WEIGHTS ARE 1/MAX(1,ABS(Y(K)))
                  DO 155 K = 1, N
                     TEMP = DMAX1(TEMP, DABS(Y(K)))
  155             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
C           CALCULATE SCALE - USE DEFAULT UNLESS VALUE PRESCRIBED
            C(15) = C(5)
            IF (C(5) .EQ. 0.D0) C(15) = 1.D0
C
C           CALCULATE HMAX - CONSIDER 4 CASES
C           CASE 1 BOTH HMAX AND SCALE PRESCRIBED
               IF (C(6).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(6).NE.0.D0 .AND. C(5).EQ.0.D0) C(16) = C(6)
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
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
C              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.D0/.9D0)**6*C(19))
     +                            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(14) = .5D0*C(14)
  185     CONTINUE
C
C         CHECK AGAINST HMAX
          C(14) = DMIN1(C(14), C(16))
C
C         CHECK AGAINST HMIN
          C(14) = DMAX1(C(14), C(13))
C
C***********INTERRUPT NO 1 (WITH IND=4) IF REQUESTED
          IF (C(8) .EQ. 0.D0) GO TO 1111
            IND = 4
            RETURN
C         RESUME HERE ON RE-ENTRY WITH IND .EQ. 4   ........RE-ENTRY..
 1111     CONTINUE
C
C         CALCULATE HMAG, XTRIAL - DEPENDING ON PRELIMINARY HMAG, XEND
          IF (C(14) .GE. DABS(XEND - X)) GO TO 190
C           DO NOT STEP MORE THAN HALF WAY TO XEND
            C(14) = DMIN1(C(14), .5D0*DABS(XEND - X))
            C(17) = X + DSIGN(C(14), XEND - X)
            GO TO 195
  190     CONTINUE
C           HIT XEND EXACTLY
            C(14) = DABS(XEND - X)
            C(17) = XEND
  195     CONTINUE
C
C         CALCULATE HTRIAL
          C(18) = C(17) - X
C
C       END STAGE 1
C
C       ***************************************************************
C       * STAGE 2 - CALCULATE YTRIAL (ADDING 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)/1398169080000.D0
C
          DO 200 K = 1, N
            W(K,9) = Y(K) + TEMP*W(K,1)*233028180000.D0
  200     CONTINUE
          CALL FCN(N, X + C(18)/6.D0, W(1,9), W(1,2))
C
          DO 205 K = 1, N
            W(K,9) = Y(K) + TEMP*(   W(K,1)*74569017600.D0
     +                             + W(K,2)*298276070400.D0  )
  205     CONTINUE
          CALL FCN(N, X + C(18)*(4.D0/15.D0), W(1,9), W(1,3))
```

```
C
            DO 210 K = 1, N
               W(K,9) = Y(K) + TEMP*(   W(K,1)*116514090000.D0
     +                                - W(K,2)*372845088000.D0
     +                                + W(K,3)*349542270000.D0 )
  210       CONTINUE
            CALL FCN(N, X + C(18)*(2.D0/3.D0), W(1,9), W(1,4))
C
            DO 215 K = 1, N
               W(K,9) = Y(K) + TEMP*( - W(K,1)*360465465937 5.D0
     +                                + W(K,2)*1281654990000.D0
     +                                - W(K,3)*928471654687 5.D0
     +                                + W(K,4)*123796220625 0.D0 )
  215       CONTINUE
            CALL FCN(N, X + C(18)*(5.D0/6.D0), W(1,9), W(1,5))
C
            DO 220 K = 1, N
               W(K,9) = Y(K) + TEMP*(   W(K,1)*335560579200 0.D0
     +                                - W(K,2)*1118535264000 0.D0
     +                                + W(K,3)*917262885000 0.D0
     +                                - W(K,4)*427218330000.D0
     +                                + W(K,5)*482505408000.D0  )
  220       CONTINUE
            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)*770204740536.D0
     +                                + W(K,2)*2311639545600.D0
     +                                - W(K,3)*1322092233000.D0
     +                                - W(K,4)*453006781920.D0
     +                                + W(K,5)*326875481856.D0  )
  225       CONTINUE
            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)*2845924389000.D0
     +                                - W(K,2)*9754668000000.D0
     +                                + W(K,3)*7897110375000.D0
     +                                - W(K,4)*192082660000.D0
     +                                + W(K,5)*400298976000.D0
     +                                + W(K,7)*201586000000.D0  )
  230       CONTINUE
            CALL FCN(N, X + C(18), W(1,9), W(1,8))
C
C           CALCULATE YTRIAL, THE EXTRAPOLATED APPROXIMATION AND STORE
C              IN W(*,9)
            DO 235 K = 1, N
               W(K,9) = Y(K) + TEMP*(   W(K,1)*104862681000.D0
     +                                + W(K,3)*545186250000.D0
     +                                + W(K,4)*446637345000.D0
     +                                + W(K,5)*188806464000.D0
     +                                + W(K,7)*15076875000.D0
     +                                + W(K,8)*97599465000.D0    )
  235       CONTINUE
C
C           ADD 7 TO THE NO OF FCN EVALS
            C(24) = C(24) + 7.D0
C
C        END STAGE 2
```

```
C
C        ****************************************************************
C        * STAGE 3 - CALCULATE THE ERROR ESTIMATE EST. FIRST CALCULATE *
C        * THE  UNWEIGHTED  ABSOLUTE  ERROR  ESTIMATE VECTOR (PER UNIT *
C        * STEP) FOR THE UNEXTRAPOLATED APPROXIMATION AND STORE IT  IN *
C        * W(*,2).  THEN  CALCULATE THE WEIGHTED MAX NORM OF W(*,2) AS *
C        * SPECIFIED BY THE ERROR  CONTROL  INDICATOR  C(1).  FINALLY, *
C        * MODIFY  THIS RESULT TO PRODUCE EST, THE ERROR ESTIMATE (PER *
C        * UNIT STEP) FOR THE EXTRAPOLATED APPROXIMATION YTRIAL.      *
C        ****************************************************************
C
C           CALCULATE THE UNWEIGHTED ABSOLUTE ERROR ESTIMATE VECTOR
            DO 300 K = 1, N
               W(K,2) = (    W(K,1)*8738556750.D0
     +                     + W(K,3)*9735468750.D0
     +                     - W(K,4)*9709507500.D0
     +                     + W(K,5)*8582112000.D0
     +                     + W(K,6)*95329710000.D0
     +                     - W(K,7)*15076875000.D0
     +                     - W(K,8)*97599465000.D0)/1398169080000.D0
  300       CONTINUE
C
C           CALCULATE THE WEIGHTED MAX NORM OF W(*,2) AS SPECIFIED BY
C           THE ERROR CONTROL INDICATOR C(1)
            TEMP = 0.D0
            IF (C(1) .NE. 1.D0) GO TO 310
C              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
C              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
C              WEIGHTS ARE 1/MAX(C(2),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
C              WEIGHTS ARE 1/MAX(C(K+30),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
C              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/MAX(1.ABS(Y(K)))
               DO 355 K = 1, N
                  TEMP = DMAX1(TEMP, DABS(W(K,2))
     +                                / DMAX1(1.D0, DABS(Y(K)))  )
  355          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        ******************************************************************
C        * STAGE 4 - MAKE DECISIONS.                                      *
C        ******************************************************************
C
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
            IF (C(9) .EQ. 0.D0) GO TO 2222
               RETURN
C           RESUME HERE ON RE-ENTRY WITH IND .EQ. 5 OR 6   ...RE-ENTRY..
 2222       CONTINUE
C
            IF (IND .EQ. 6) GO TO 410
C              STEP ACCEPTED (IND .EQ. 5), SO UPDATE X, Y FROM XTRIAL,
C                 YTRIAL, ADD 1 TO THE NO OF SUCCESSFUL STEPS, AND SET
C                 THE NO OF SUCCESSIVE FAILURES TO ZERO
               X = C(17)
               DO 400 K = 1, N
                  Y(K) = W(K,9)
  400          CONTINUE
               C(22) = C(22) + 1.D0
               C(23) = 0.D0
C**************RETURN(WITH IND=3, XEND SAVED, FLAG SET) IF X .EQ. XEND
               IF (X .NE. XEND) GO TO 405
                  IND = 3
                  C(20) = XEND
                  C(21) = 1.D0
                  RETURN
  405          CONTINUE
               GO TO 420
  410       CONTINUE
C              STEP NOT ACCEPTED (IND .EQ. 6), SO ADD 1 TO THE NO OF
C                 SUCCESSIVE FAILURES
               C(23) = C(23) + 1.D0
C**************ERROR RETURN (WITH IND=-3) IF HMAG .LE. HMIN
               IF (C(14) .GT. C(13)) GO TO 415
                  IND = -3
                  RETURN
  415          CONTINUE
  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, 58HCOMPUTATION STOPPED IN DVERK WITH THE FOLLOWIN
     +G VALUES -
     +   / 1H0, 5HIND =, I4, 5X, 6HTOL  =, 1PD13.6, 5X, 11HX          =,
     +          1PD22.15
     +   / 1H , 5HN   =, I4, 5X, 6HHMIN =, 1PD13.6, 5X, 11HXEND       =,
     +          1PD22.15
     +   / 1H , 5HNW  =, I4, 5X, 6HHMAX =, 1PD13.6, 5X, 11HPREV XEND =,
     +          1PD22.15
     +   / 1H0, 14X, 27HNO OF SUCCESSFUL STEPS     =, 0PF8.0
     +   / 1H , 14X, 27HNO OF SUCCESSIVE FAILURES =, 0PF8.0
     +   / 1H , 14X, 27HNO OF FUNCTION EVALS       =, 0PF8.0
     +   / 1H0, 23HTHE COMPONENTS OF Y ARE
     +   // (1H , 1P5D24.15)                                           )
C
      STOP
C
C  END ABORT ACTION
C
      END
```