1 SUMMARY

To integrate a system of **first order ordinary differential equations**

$$y_i' = f_i(y_1, y_2, ..., y_n, x) \qquad i=1,2,...,n$$

where $n \leq 20$ and given initial values $y_i(x_0)=y_i^{(0)}$. This subroutine calls DC03 forfeiting some of its facilities to provide a simplified calling sequence.

The user provides a subroutine to compute the functions f_i, but printed output is produced by the package's standard output subroutine. The range of integration x_0 to x_{end} and number of output points must be specified; these are spaced at equal intervals.

The method used is designed to work well on stiff problems but is also effective on non-stiff problems (for further information see the specification of DC03).

ATTRIBUTES — Version: 1.0.0. **Types:** DC04A; DC04AD. **Calls:** DC03. **Original date:** February 1984. **Origin:** W. P. Sweetenham, Harwell.

2 HOW TO USE THE PACKAGE

Important: in this description of DC04 references are made to the specification for subroutine DC03. The user may find it useful to refer to a copy of the DC03 documentation.

2.1 The Argument List and Calling Sequence

The single precision version

```
CALL DC04A(DERIV,NEQ,X,XEND,Y,TOL,NPR,IP,INIT)
```

The double precision version

```
CALL DC04AD(DERIV,NEQ,X,XEND,Y,TOL,NPR,IP,INIT)
```

DERIV is the name of a user-supplied subroutine which calculates the derivative functions $f_i(y_1, y_2, ..., y_n, x)$, $i=1,2,...,n$. The calling sequence is described in §2.2.

NEQ is an INTEGER variable which must be set by the user to n the number of equations. **Restriction:** $n \leq 20$.

X is a REAL (DOUBLE PRECISION in the D version) variable which must be set by the user to the initial value x_0 of x, and is used thereafter for the current value.

XEND is a REAL (DOUBLE PRECISION in the D version) variable which must be set by the user to the value of x_{end} where $[\, x_0, x_{end} \,]$ is the range of integration. Note that x_{end} must be greater than x_0.

Y is a REAL (DOUBLE PRECISION in the D version) array of length at least n which must be set by the user to the initial values $y_i^{(0)}$ of y_i $i=1,2,...,n$, and it is thereafter to hold the current values.

TOL is a REAL (DOUBLE PRECISION in the D version) variable which must be set by the user to specify the accuracy required. TOL is a relative error and should be a number between 10^{-2} and the number TOLMIN (about 10^{-10} on most computers, see DC03 section 10). The formula for the error in y_i at a particular value of x is roughly:

$$|\text{local error}| \leq \texttt{TOL} \times [\max_{t<x} |y_i(t)| \times 10^{-10} + |y_i(x)|]$$

The method has built-in refinements to deal with cases where y_i has an initial value of zero, or passes through zero during integration; this is why the inequality above is only approximate.

NPR is an INTEGER variable which must be set by the user to the number of equally spaced intervals into which the

range x_0, x_{end} is to be divided for output purposes (see §2.3).

IP is an INTEGER variable which must be set by the user to the Fortran stream number to be used for printed output. Printing can be suppressed by setting IP to zero (see §2.3).

INIT is an INTEGER variable which must be set by the user to 0 or 1. INIT=0 means that DC04 is starting on a new problem. After a successful run DC04 sets INIT=1 so that the subroutine can be called again to continue solving the same equations without doing a restart (see §2.5). If the problem is changed then the user must set INIT=0. If an error has occurred, INIT is set to a negative number (see §2.4). After each call to DC04 INIT must be tested to see whether an error has occurred.

2.2 The Subroutine to Compute the Derivatives

The user must supply a subroutine to compute values of the derivative functions $f_i(y_1, y_2, ..., y_n, x)$, i=1,2,...,n given values of y_i i=1,2,...,n and x. The user may choose the name of the subroutine and this must be coded as the first argument of DC04 (see DERIV in §2.1). The name must be declared external by including it in an EXTERNAL statement in the calling program. The subroutine must have the following calling sequence.

```
CALL name(X,Y,F,RPAR,IPAR)
```

X is a REAL (DOUBLE PRECISION in the D version) variable which will contain the current value of x. Its value must not be altered.

Y is a REAL (DOUBLE PRECISION in the D version) array and the first n elements will contain the current values of y_i i=1,2,...,n. These must not be altered.

F is a REAL (DOUBLE PRECISION in the D version) array and the user must set the first n elements to the values of the derivative functions $f_i(y_1, y_2, ..., y_n, x)$ i=1,2,...,n before returning. In many cases the derivative subroutine will be called at the end of the integration range with values of x beyond x_{end}.

RPAR and IPAR are parameter arrays which are included solely for compatibility with the calling sequence of DERIV in DC03A/AD. They are not used or altered by DC04A/AD, and should be declared as arrays of dimension 1, of type REAL (DOUBLE PRECISION in the D version) and INTEGER respectively.

2.3 Printed Output

Printed output is called at the points

$$x = x_0 + \frac{k(x_{end} - x_0)}{m}, \qquad k=0,1,2,...,m$$

where m the number of intervals is specified by the user in the DC04 argument NPR. The variables x and y_i, f_i, i=1,2,...,n are printed at each call, on the Fortran stream number specified by the user in the argument IP. For $k>0$ the values of f_i are derived from interpolation by DC03, not by calling DERIV directly. Printing is suppressed by setting IP=0.

2.4 Error Diagnosis

If an error is diagnosed by DC03, DC04 causes an error message to be printed (unless IP=0), sets INIT to a negative number, and then returns to the calling program The value to which INIT is set for errors diagnosed by DC03 is INIT=IDID, where IDID is the error code defined in the DC03 specification (see DC03 section 5). Please note INIT is set to zero if the value of NEQ exceeded the limit on n.

2.5 Calling DC04A/AD Again

After DC04 has ended and execution has returned to the calling program it can be called again for the same set of equations with DERIV, NEQ, X, Y and INIT (=1) unchanged by the user. DC04 will then call DC03 to continue integrating the equations from the current point returned in X without treating them as a new problem (see DC03 section 5). Thus it is possible to integrate with interval h_1 between output points from x=p_0 up to x=p_1 and then to

continue integrating from $x=p_1$ up to $x=p_2$ with interval h_2 without doing a restart. To do this first set DERIV, NEQ, X, XEND, Y, TOL, NPR, IP and INIT with

$$\texttt{X}=p_0,\ \texttt{XEND}=p_1,\ \texttt{NPR}=\frac{p_1-p_0}{h_1},\ \texttt{INIT=0}$$

and call DC04. On return DC04 sets INIT=1. Then reset

$$\texttt{XEND}=p_2,\ \texttt{NPR}=\frac{p_2-p_1}{h_2}$$

and call DC04 again. This can be repeated as often as is desired.

Important: after each call to DC04 INIT must be tested to ensure that it has not been set to a negative number by DC04 (see §2.4)

The user may want DC04 to integrate a set of equations and return the values of the y_i to the calling program at certain values of x without doing any printing. To do this set IP=0 and each time another set of values y_i is required the calling program should set XEND to the corresponding value of x, with NPR=1, and call DC04 again. Since, after the first call, INIT will have been set to 1 by DC04, the equations will be integrated successfully up to the point x=XEND so long as DERIV, X and Y have not been altered by the user's program.

2.6 Integrating Non-smooth Equations

Sometimes the user may want to integrate non-smooth equations, i.e. ones in which either

$$f_i(y_1,y_2,...,y_n,x)$$

or

$$\frac{d^r f_i}{dx^r}$$

for some i, r is discontinuous at points $x=x_1,x_2,...,x_s$. The best way to solve these equations is to integrate between each pair of points x_j,x_{j+1} using a smooth set of equations. So when X = x_j set XEND = x_{j+1}, DERIV=DERIVJ and INIT=0, where DERIVJ is the name of a subroutine of the form described in §2.2 which calculates a smooth set of equations valid for $x_j \le x \le x_{j+1}$ and INIT=0 forces DC04 to do a restart with the current values of X, Y. Subroutine DC04 returns to the calling program with X = x_{j+1} and the corresponding value for Y, and is ready to be called again with a new subroutine DERIV. Note that these subroutines must all be declared EXTERNAL in the calling program and must be smooth outside the range for which they are valid, since DC04 may take a step outside this range and then interpolate to get values for the y_i at $x=x_{j+1}$.

3 GENERAL INFORMATION

Workspace: none.

Use of common: uses common areas DC03Y/YD, DC03Z/ZD, MA30E/ED and MA30F/FD; see section 10 of DC03 specification sheets.

Other routines called directly: calls DC03A/AD.

Input/output: provides an option for printing results (see §2.3) and will produce error messages in the event of errors (see §2.4). The Fortran stream number is specified by the user and printing may suppressed (see IP in §2.1).

Portability: At least 8-byte arithmetic is recommended.

Restrictions: $n \le 20$.

4 METHOD

The sole purpose of DC04 is to provide a simple interface to the subroutine DC03 which uses Gear's method.

5 EXAMPLE OF USE

The Fortran program on the following page solves the system of equations

$$\frac{dy_1}{dx} = dy_2, \quad \frac{dy_2}{dx} = -dy_1,$$

with $y_1(0)=0$ and $y_2(0)=1$, from $x=0$ to $x=10$, with output at each interval of 0.1.

```
      DOUBLE PRECISION TOL,X,XEND,Y(2)
      EXTERNAL DERIV
C DERIV IS THE NAME OF THE SUBROUTINE CALCULATING THE DERIVATIVES
      NEQ=2
      TOL=1.E-3
C AN ERROR TOLERENCE OF 1 IN 1000
      X=0.0D0
      XEND=10.0D0
      Y(1)=0.0D0
      Y(2)=1.0D0
      NPR=100
      IP=6
C PRINT THE RESULTS ON FORTRAN UNIT 6
      INIT=0
      CALL DC04AD(DERIV,NEQ,X,XEND,Y,TOL,NPR,IP,INIT)
      IF (INIT.LT.0) WRITE(6,1) INIT
C CHECK FOR THE OCCURANCE OF AN ERROR
    1 FORMAT(' ERROR DETECTED: INIT EQUALS',I4)
      STOP
      END
      SUBROUTINE DERIV(X,Y,F,RPAR,IPAR)
      DOUBLE PRECISION X,Y(2),F(2),RPAR(1)
      INTEGER IPAR(1)
      F(1)=Y(2)
C DY(1)/DX=Y(2)
      F(2)=-Y(1)
C DY(2)/DX=-Y(1)
      RETURN
      END
```