



1 SUMMARY

To estimate first or second derivatives given a table of function values and finite differences. The estimates are obtained using finite difference formulae and the user specifies the highest order of differences which can be used and the required accuracy.

The subroutine will return a condition flag indicating whether the number of differences supplied was sufficient to achieve the accuracy. The subroutine does not use special formulae at the end of the range and the user is expected to extend the table so that there are a sufficient number of differences for the end values.

ATTRIBUTES — **Version:** 1.0.0. **Types:** TD01A; TD01AD. **Original date:** 1965. **Origin:** P.Hallowell, Atlas Lab., Chilton, Oxon.

2 HOW TO USE THE PACKAGE

2.1 The differences and formulae

The user is expected to provide a table of the differences of the function $y(x)$, i.e.

$$\begin{array}{ccccccc}
 y_l & & & & & & \\
 y_{l+1} & \delta y_{l+1} & \delta^2 y_l & & & & \\
 y_{l+2} & \delta y_{l+2} & \delta^2 y_{l+1} & \delta^3 y_{l+1} & \delta^4 y_{l+2} & - & \\
 y_{l+3} & \delta y_{l+3} & \delta^2 y_{l+2} & \delta^3 y_{l+2} & - & - & \\
 y_{l+4} & \delta y_{l+4} & - & - & - & - & \\
 - & - & - & \delta^3 y_{n-1} & \delta^4 y_{n-2} & - & \\
 - & - & \delta^2 y_{n-1} & \delta^3 y_{n-1} & & & \\
 y_n & \delta y_{n-1} & & & & &
 \end{array}$$

where the odd terms are defined by

$$\delta^{2j+1} y_{i+1} = \delta^{2j} y_{i+1} - \delta^{2j} y_i \quad i=r, r+1, \dots, m$$

and the even terms are defined by

$$\delta^{2j} y_i = \delta^{2j-1} y_{i+1} - \delta^{2j-1} y_{i-1}.$$

There are several possible storage arrangements for the differences depending on the choice of arguments IOPT and NS. The library subroutine TA03A/AD can be used to generate a table of differences in any of the required arrangements. The table must be extended sufficiently at both ends to generate the full range of high order differences needed for the calculation. When using TA03A/AD this means that the range specified by its arguments, NL and NU, must extend beyond the range specified by ML and MU in TD01A/AD.

Estimates of the first derivatives y'_i $i=r, r+1, \dots, m$ will be calculated using the formula

$$h y'_0 = (\mu \delta_0 - \frac{1}{6} \mu \delta_0^3 + \frac{1}{30} \mu \delta_0^5 - \dots) y_0$$

or alternatively, estimates of the second derivatives y''_i $i=r, r+1, \dots, m$ are calculated using the formula

$$h^2 y''_0 = (\delta_0^2 - \frac{1}{12} \delta_0^4 + \frac{1}{90} \delta_0^6 - \dots) y_0$$

The user must specify the accuracy required and no special formulae are used at the ends of the range.

2.2 Argument list and calling sequence

The single precision version

```
CALL TD01A(H, Y, IY, NL, NS, NU, NDIFF, IOPT,
* L0, D, EPS, E, LC, LA, NDERIV)
```

The double precision version

```
CALL TD01AD(H, Y, IY, NL, NS, NU, NDIFF, IOPT,
* L0, D, EPS, E, LC, LA, NDERIV)
```

- H is a REAL (DOUBLE PRECISION in the D version) **variable** which must be set by the user to the tabulation interval, $h = x_i - x_{i-1}$, where x is the independent variable.
- Y is a two-dimensional REAL (DOUBLE PRECISION in the D version) **array** of first dimension IY in which the user must put: the tabulated values of the function y and the table of differences. The values of $y_0, y_r, y_{r+1}, \dots, y_m$ must be stored in the first column of Y and in the positions $Y(\text{ML}, 1), Y(\text{ML}+\text{NS}, 1), Y(\text{ML}+2*\text{NS}, 1), \dots, Y(\text{MU}, 1)$. There are several possible storage arrangements for the differences depending on the choice of arguments IOPT and NS; this is described more fully in §2.2.
- IY is an INTEGER **variable** which must be set by the user to the first dimension of the two-dimensional array Y.
- ML is an INTEGER **variable** which must be set by the user to specify where the beginning of the table is in the array Y, i.e. y_r is stored in $Y(\text{ML}, 1)$.
- NS is an INTEGER **variable** which must be set by the user to specify the spacing of the members of the table in the array Y, i.e. y_{r+1} is stored in $Y(\text{ML}+\text{NS}, 1)$, y_{r+2} in $Y(\text{ML}+2*\text{NS}, 1)$, and so on. If both odd and even differences are present NS must be set to two but NS can be set to one when only odd differences are present (for first derivative estimates), or when only even differences are present (for second derivative estimates), see §2.2.
- MU is an INTEGER **variable** which must be set by the user to specify where the end of the table comes in the array Y, i.e. y_m is stored in $Y(\text{MU}, 1)$.
- NDIFF is an INTEGER **variable** which must be set by the user to the order of the highest order differences present in the table which may be used in the finite difference approximations to the derivatives. The subroutine will not necessarily use differences up to this order if it can achieve the accuracy with lower order differences.
- IOPT is an INTEGER **variable** which must be set by the user to specify one of three possible storage arrangements for the differences in the array Y. The storage arrangements and the choices for IOPT and NS are described in §2.2.
- D is a REAL (DOUBLE PRECISION in the D version) **array** of length at least MU in which the subroutine will return its estimates of the derivatives. These will be returned in a manner similar to that of the function values in the array Y, i.e. the derivative values $d_i, i=r, r+1, \dots, m$ are returned in $D(\text{ML}), D(\text{ML}+\text{NS}), \dots, D(\text{MU})$.
- EPS is a REAL (DOUBLE PRECISION in the D version) **variable** which must be set by the user to an accuracy requirement ϵ . If $e_i^k, i=r, r+1, \dots, m$ are the k th order terms in the finite difference approximations given in §4, an approximation of the k th order is accepted if

$$\max_{r \leq i \leq m} |e_i^k| \leq \epsilon.$$

The user is told whether this requirement is satisfied and if so whether k is less than the order specified in NDIFF. The user will note that this accuracy criterion assumes a certain degree of smoothness in the function $y(x)$ and that under the circumstances the accuracy cannot always be guaranteed.

- E is a REAL (DOUBLE PRECISION in the D version) **array** of length at least ML in which the subroutine returns the highest order terms used in the finite difference approximations in calculating the derivative estimates returned in D. These are the values $e_i^k, i=r, r+1, \dots, m$ and they are stored in $E(\text{ML}), E(\text{ML}+\text{NS}), \dots, E(\text{MU})$.
- LC is an INTEGER **variable** which is set by the subroutine to indicate success or failure. The possible values are

- (a) when the accuracy has been achieved.
- 1 highest order term used was less than the order specified in NDIFF, and the order actually used is given in LA.
 - 2 the highest order term used was exactly NDIFF.
- (b) when the accuracy is not achieved.
- 3 higher orders than NDIFF were required.
 - 4 the differences of order NDIFF are diverging.
 - 5 differences of order LA are diverging, where $LA < NDIFF$.
 - 7 argument errors; a diagnostic is printed.

LA is an INTEGER **variable** which is set by the subroutine to the highest order of differences used to obtain the estimates returned in the array D, $LA \leq NDIFF$.

NDERIV is an INTEGER **variable** which must be set by the user to select either first or second derivatives. The two possible values are

- 1 first derivatives
- 2 second derivatives

2.2 The storage arrangements for the differences

The user is given the option of providing the differences in the array Y in a variety of arrangements including a simple arrangement, more compacted arrangements and arrangements where either the odd or the even differences are omitted to save on storage. The choice of arrangement is controlled by the argument IOPT and the spacing argument NS.

The simplest arrangement, IOPT=0 and NS=2, is illustrated by the following diagram.

y_{i-1}		$\delta^2 y_{i-1}$	
	δy_{i-1}		$\delta^3 y_{i-1}$
y_i		$\delta^2 y_i$	
	δy_{i+1}		$\delta^3 y_{i+1}$
y_{i+1}		$\delta^2 y_{i+1}$	

A more compact scheme, IOPT=1 and NS=2, which closes up the differences and hence requires less storage is the following

y_{i-1}	$\delta^2 y_{i-1}$	$\delta^4 y_{i-1}$	$\delta^6 y_{i-1}$
	δy_{i-1}	$\delta^3 y_{i-1}$	$\delta^5 y_{i-1}$
y_i	$\delta^2 y_i$	$\delta^4 y_i$	$\delta^6 y_i$
	δy_{i+1}	$\delta^3 y_{i+1}$	$\delta^5 y_{i+1}$
y_{i+1}	$\delta^2 y_{i+1}$	$\delta^4 y_{i+1}$	$\delta^6 y_{i+1}$

The third arrangement, IOPT=2 and NS=2, also compacts the differences but differs slightly from the previous arrangement.

y_{i-1}		$\delta^2 y_{i-1}$	$\delta^4 y_{i-1}$
	$\delta y_{i-\frac{1}{2}}$	$\delta^3 y_{i-\frac{1}{2}}$	$\delta^5 y_{i-\frac{1}{2}}$
y_i		$\delta^2 y_i$	$\delta^4 y_i$
	$\delta y_{i+\frac{1}{2}}$	$\delta^3 y_{i+\frac{1}{2}}$	$\delta^5 y_{i+\frac{1}{2}}$
y_{i+1}		$\delta^2 y_{i+1}$	$\delta^4 y_{i+1}$

The three schemes illustrated are all for the case when $NS=2$. Another possibility is to set $NS=1$ to close up the rows and omitting either all odd order differences or all even differences (except order zero). Thus with $NS=1$ the three values of IOPT will give

0 the i th row will contain $y_i, \delta y_{i+\frac{1}{2}}, \delta^2 y_i, \delta^3 y_{i+\frac{1}{2}}$, etc.

1 even differences will overwrite odd differences so that the i th row will contain $y_i, \delta^2 y_i, \delta^4 y_i$, etc.

2 the odd differences will overwrite the even differences so that the i th row will contain $y_i, \delta y_{i+\frac{1}{2}}, \delta^3 y_{i+\frac{1}{2}}$, etc.

The second dimension of the array Y , call it MY say, must be large enough to accommodate the table of differences and the minimum possible value for MY varies with the setting of IOPT as follows

0 $MY \geq NDIFF+1$.

1 $MY \geq (NDIFF+3)/2$.

2 $MY \geq (NDIFF+4)/2$

Note that NS must be greater than or equal to one.

3 GENERAL INFORMATION

Use of common: none.

Workspace: none.

Other subroutines: uses a private subroutine TD01B/BD.

Input/Output: diagnostic printing when errors occur.

Restrictions:

$NS \geq 1$.

IOPT must be 0, 1 or 2.

4 METHOD

Two forms of finite difference formulae are, see §2.1. The estimates $d_i = y'_i$, or $d_i = y''_i$, are accumulated by adding into each successively the next term in the formula. At each stage a test is made to see if the maximum term added in is less than the accuracy specified or if the differences involved in that term are diverging.