1 SUMMARY

Given an interval (a_0, b_0), $a_0 < b_0$, this subroutine automatically **determines axis limits, label points** and **label formats** suitable for drawing a labelled axis on the graph plotter.

A new interval (a, b) and a label interval h is calculated such that a and b are multiples of h and are round figure numbers which satisfy $a \le a_0 < b_0 \le b$. The choice of limits and h is based on a table, of intervals and h values, located in a common block which the user has the option to change.

For the purpose of labelling the axis the subroutine will return recommendations for the format type, field width and number of decimal places appropriate for all label values. The positions of the most significant, and least significant, digits in the label values are made available to the user in the common block.

Also returned is a recommendation for the number of minor subdivisions of h. On an axis these would normally be unlabelled and drawn at a lower intensity.

No graph plotting is performed and the subroutine is not dependent on any particular graph plotting package. It could be used equally well for determining the formats needed to print out a table of values.

ATTRIBUTES — Version: 1.0.0. **Types:** `FP01A`. **Original date:** May 1974. **Origin:** M.J.Hopper, Harwell.

2 HOW TO USE THE PACKAGE

2.1 The argument lists and calling sequence

There are three entries to the subroutine: `FP01A` calculates the limits and formats, `FP01B` just calculates limits, and `FP01C` just calculates the formats (with the user suppling the limits).

Note that the subroutine does not return the value of the label interval h but this can always be calculated from $h=(b-a)/m$.

(i) To obtain round-figure limits, the label intervals and formats

```
CALL FP01A(A,B,M,NFMT,NFW,NDEC,NSUBD)
```

`A,B` are `REAL` variables which must be set by the user to the original limit values a_0 and b_0 which require adjustment. **Restriction:** $a_0 < b_0$. On return, the subroutine will have reset them to the final adjusted values a and b.

`M` is an `INTEGER` variable which is set by the subroutine to the number of label intervals (number of label points = `M+1`). If the subroutine finds that $a_0 \ge b_0$ it returns `M` set to zero.

`NFMT` is an `INTEGER` variable which is set by the subroutine to the format type recommended for printing the labels. The possible values are

0 error: `A` ≥ `B`.

1 integer (I format)

2 fixed decimal (F format)

3 floating point exponent form (E format)

The form of the E format is that used by Fortran output and is ±0.*dddE*±*nn*. So that the field width is always equal to `NDEC+7`. If a more compact form is desired the `LSD` and `MSD` fields in common can be used to determine the sign and number of digits in the exponent.

`NFW` is an `INTEGER` variable which is set by the subroutine to the minimum field width required to print the labels.

NDEC is an INTEGER variable which is set by the subroutine and its precise meaning depends on the format type as follows

I format (NFMT=1) NDEC = 0

F format (NFMT=2) NDEC = number of decimal places for the format.

E format (NFMT=3) NDEC = number of significant figures for the format.

NSUBD is an INTEGER variable which is set by the subroutine to the recommended number of minor sub-intervals of *h*.

(ii) To obtain adjusted limits and label interval but no format information.

```
CALL FP01B(A,B,M,NSUBD)
```

All four arguments are exactly as described for FP01A above.

(iii) To obtain the label formats given the limits and number of label intervals

```
CALL FP01C(A,B,M,NFMT,NFW,NDEC)
```

A,B are REAL variables which must be set by the user to the axis limits $a<b$, and it is assumed that they are reasonable round figure numbers of the sort that FP01A would have chosen. They are not adjusted and A and B are not altered by the subroutine.

In calculating the least significant digit position of the labels it is assumed that the least significant digit present in the labels is also the least significant digit of *h* the label interval. If *a* and *b* are multiples of *h* this will be so.

M is an INTEGER variable which must be set by the user to *m* the number of label intervals, so that the label interval is given by $h=(b-a)/m$. **Restriction:** $m>0$. This argument is not altered by the subroutine.

NFMT, NFW and NDEC are all as described for FP01A except that an additional possible error for a return of NFMT=0 is that the value of M passed to the subroutine was less than or equal to zero.

2.2 The common area

The subroutine uses a common area called FP01B which may be referenced by the calling program.

```
COMMON/FP01B/ MAXFW,EPS,MSD,LSD,N,W(3,5)
```

MAXFW is an INTEGER variable and specifies an upper limit to the label field-widths – the default is 13. It will act as the limiting factor in the cases when the least significant digit position cannot be located correctly because the limits are not round figure numbers. If it becomes effective digits may be lost from the least significant end in the case of F or E formats. The default value of 13 allows E formats to have 6 significant digits, which should be adequate for single precision work. If MAXFW ≥ 10 no loss of digits can occur from integer values but, for values of MAXFW below that, the loss would be from the most significant end.

EPS is a REAL variable which specifies a tolerance to be used by the subroutine when estimating the rounding errors in the limit values – the default for EPS is 2×10^{-6}. If altered it should be set to a slight overestimate of the relative rounding errors in the limits, if its set too small it will prevent the subroutine from distinguishing rounding errors from significant digits.

MSD and LSD are INTEGER variables which are set by the subroutine to the most significant digit (m.s.d.) position and the least significant digit (l.s.d.) position in the label set. The positions are numbered

$$\ldots(3)(2)(1)(0).(-1)(-2)\ldots$$

about the decimal point, e.g the value 463.500 would give MSD=2 and LSD=-1.

N is an INTEGER variable which specifies the number of intervals present in W below – the default is N=3.

`W` is a `REAL` two-dimensional array of first dimension 3 and second dimension at least `N` (default 5). It holds a set of intervals, label-interval sizes, and minor subdivision recommendations the significance of which is described in §4.

`W(1,I) I=1,N` hold the values w_j j=1,2,...,n and the default values are 10, 20 and 50.

`W(2,I) I=1,N` hold the label intervals h_j j=1,2,...,n and the default values are 1, 2 and 5.

`W(3,I) I=1,N` hold the minor subdivision numbers s_j j=1,2,...,n and the default values are 10, 2 and 5.

These values are used by the subroutine to choose suitable limit values and label points. They may be respecified by the user but if this causes `N` to be greater than five the whole common area should be re-initialized through a `BLOCK DATA` subprogram.

3 GENERAL INFORMATION

Use of common: the subroutine uses a common block called `FP01B`, see §2.2.

Workspace: none.

Other subroutines: none.

Input/Output: none.

Restrictions:

$a_0 < b_0$,

$a < b$,

$m > 0$ (`FP01C`).

4 METHOD

The method can be conveniently divided into two parts: (1) adjusting the limits and choosing the label interval size; and (2) determining the ideal label format.

4.1 Adjusting the limits and choosing the label interval

Essential to the method are the intervals and label interval sizes stored in the array `W` in common.

Let the intervals be (w_{j-1}, w_j) j=1,2,...,n, and they must satisfy the conditions $w_0 < w_1 < ... < w_n$ where w_0 is defined to be $w_0 = \frac{w_n}{10}$, (w_0 is not actually stored in `W`). It will also be usual, although not strictly necessary for, $10 \le w_n < 100$.

The subroutine is given the limits $a_0 < b_0$ which are immediately adjusted to $\hat{a}_0 = a_0 + \tau$ and $\tilde{b}_0 = b_0 + \tau$, where τ is an estimate of the rounding error in a_0 and b_0, i.e. $\tau = \max|a_0|, |b_0| \times \varepsilon$ where ε is the relative rounding error tolerance given in the common variable `EPS`.

A reduced range r is generated such that $(\tilde{b}_0 - \hat{a}_0) = r \times 10^k$, with k chosen so that $w_0 \le r < w_n$ and then the table of intervals in `W` is searched to find the interval (w_{i-1}, w_i) containing r, i.e. so that $w_{i-1} \le r < w_i$.

Corresponding to the set of intervals (w_{j-1}, w_j) j=1,2,...,n are a set of label interval sizes h_j j=1,2,...,n and a set of subdivision numbers s_j j=1,2,...,n. For an axis range satisfying $w_{i-1} \le r < w_i$, the label interval value will be $h = h_i \times 10^k$ and the number of minor subdivisions will be recommended to be s_i.

The limits (a_0, b_0) are then adjusted to (a, b) where a is set to the maximum multiple of h which is less than or equal to $\hat{a}_0$, and b is set to the minimum multiple of h which is greater than equal to $\tilde{b}_0$. Further it is insured that $a \le a_0 < b_0 \le b$.

A new value for the axis range r is computed such that $(b–a)=r\times10^k$ and this is tested to see if it still lies in (w_{i-1}, w_i). If it does not, the new limits are fed back into the algorithm to be readjusted. This insures that the method is consistent.

The number of label intervals is simply calculated from

$$m=\text{int.part}\{(b-a+\tfrac{1}{2}h)/h\}.$$

4.2 The label formats

First the subroutine obtains the most significant digit position (m.s.d.) by reducing $\max|a|,|b|=\sigma\times10^k$ such that $1\le\sigma<10$, then the m.s.d. equals k.

The least significant digit (l.s.d.) is found by estimating the rounding error in h and, after adding it into h, discarding the digits at and below the rounding error level and then stripping off any trailing zeros. The rounding error estimate is

$$e=\frac{2h\tau}{b–a},$$

where τ is defined above.

Having found the l.s.d. and m.s.d. the formats can be obtained quite easily. They are chosen with a preference for integer first, F type second and if all else fails an E type is used. In the case of F and E types an attempt is made to minimise the field width and this can sometimes result in an E type being chosen in preference to an F type.