4.30 Summary

The preceding examples were included to give you an idea of the range of possibilities for producing graphics with **VP**. It is important to remember the following facts:

- There is a general sequence of commands to most `.exg` files:
  - Initialize the **VP** *global* parameters and set the destination for graphics primitives.

    ```
    restart default        ; reset globals
    save &%0.              ; \ugh\ file has same name as
                             \exg\ file
    ```

  - Input the numeric data to be plotted either by directly including the data in the current `.exg` file or by reading a separate file.
  - Specify AXIS placement and numeric limits (or accept the defaults based on the current data) and draw the AXIS.
  - Add AXIS text labels and caption.
  - Plot the data; specify symbols, line types, color, and scaling.
  - Perform any fits or analysis on the data.
  - Add other text annotation, such as results of fits.
- There are almost always multiple ways to generate a specific graphical output. For example, the following are all equivalent:

```
xdata column 1 ^ ydata column 2
xydata columns 1 2
xydata { c1 c2 }
```

5 VP Command Reference

This chapter provides an alphabetical reference listing of the **VP** commands. The following conventions are observed:

- Required parameters are shown without brackets.
- Optional parameters or classes of parameters are enclosed in square brackets [. . .].
- Numeric parameters are shown in `typewriter` font. Most numeric parameters may be replaced by numeric calculations delimited by [. . .] or {. . .} brackets or by named INTEGER or REAL parameters.
- Upper or lower case may be used interchangeably for any of the commands and their options.
- SYNONYMS and parameter-value settings may also be included on each command line. These are processed before command execution.
- **Examples** are given at the end of each command. In these examples all text after a semicolon is a comment.

5.1 ARRAYDAT

ARRAYDAT name [filespec] [options]
`[possible numeric data lines]`

The following lines contain an array of tabular data that will be read into the named array. Values will be added until a nonnumeric field is reached. The data may be in free format, but the same amount of numbers must be given on each line (unless COLUMN is given as an option). Up to 50 named arrays may be defined.

Once saved, the array and its characteristics may be used in calculations like TABDATA:

```
{ name.min }   min value in array name
{ name.max }   max value in array name
{ name.nmbr }  number of points in array name
{ name.ncol }  number of columns in array name
{ name.nrow }  number of rows in array name (nmbr/ncol)
{ name.Cn }    column n from array name
{ name.Rn }    row n from array name
{ name }       the complete array name data as a 1D array
```

The allowed options are as follows:

ArrayName Required name of the array (maximum of 11 characters).

filespec If ArrayName is followed by a file specification, the data will be read from the specified file. Comment lines may be included within the data. After the ARRAY has been read, any additional lines in the file will be treated as part of a normal `.exg` file. If the integer parameter SKIP > 0 at the start of the file read, then the first SKIP records will be skipped.

CONTINUE The data will be added as additional rows to the existing ARRAY. The old NCOL will be preserved and a new NROW will be calculated independent of the number of data values on each line. This allows arrays with large NCOL to be read one row at a time.

ASIS Changes the characteristics of the current array when used with COLUMN. Accepts no new data.

ROW Independent of the number of data values on each line, a new value of NCOL will be calculated from the existing NROW after the data is read.

COLUMN `ncol`
Explicitly set number of columns.

ERASE Erase the existing ARRAY name, read no data.

INTEGRATE After calculation or reading of the data array, all elements after the first are replaced by their running sum. Thus,

```
XDATA 1 2 3 4 INTEGRATE,
```

results in XDATA of 1 3 6 10.

DIFF After calculation or reading of the data array, all elements after the first are replaced by their difference relative to the previous. Thus,

```
XDATA 1 3 6 10 DIFF,
```

results in XDATA of `1 2 3 4`. INTEGRATE and DIFF are inverse operations.

{ normal calculations } or [HSILOP calculations]
Use normal {...} or HSILOP (reverse POLISH) [...] calculations to define the numeric values (see Section 3.2). NROW and NCOL should be set explicitly AFTER the calculations.

Other options store the data as different binary types:

DOUBLE Saves 8-byte REALs in default hardware format.

FLOAT Saves 4-byte REALs in default hardware format.

LONG Saves 4-byte integers (possible byte reversal).

SHORT Saves 2-byte integers (possible byte reversal).

BYTE Saves 1-byte integers (0-255).

The PRINT ARRAY command will show the characteristics of each data array (including TABDATA, XDATA, YDATA, ZDATA). Each array has an associated name, MIN value, MAX value, number of points, and number of columns.

Once defined, the array `name.ncol` columns may be changed with the command `ARRAY name ASIS COLUMN newcol`.

Examples

```
array mass sample.dat     ; reads sample.dat as array
ydata { mass*mass }       ; use as YDATA

array R loop 200 {i/20}  ; calculate R with 200 elements
array R ASIS column 2     ; reformat as 2 columns
print R                   ; print the values
```

5.2 AVERAGE

AVERAGE `nbins [amin amax]` [options]

Divides the x-axis into `nbins` bins and averages all points that fall into the same bin. If numeric `amin amax` are also given, the x-axis between `amin` and `amax` is divided into `nbins` bins. The result is a 3 column TABDATA array (destroying any previous TABDATA) of non-zero bins with column 1=X, 2=Y, 3=count. The averaged data also replace the XYDATA. Only points that fall within x-axis limits are used. Only bins containing points are kept.

The numeric parameters are:

`nbins`	the required number of bins
`amin`	optional minimum axis coordinate for the bins
`amax`	optional maximum axis coordinate for the bins

The allowed options are:

X	Default.
Y	Same as X, but the `nbins` bins and all averaging are based on Y rather than X coordinates.
INDEX	The `nbins` bins and all averaging are based on the index of the X,Y point (index=1,`nmbr`). This gives a moving average over `nmbr/nbins` points.
ASIS	The current TABDATA is preserved; only XYDATA is replaced.
ALL	Empty bins are NOT removed from XYDATA and TABDATA.
CENTER	Bin center coordinate (instead of average in bin) used for averaging axis.
LOWER	Bin lower coordinate (instead of average in bin) used.
UPPER	Bin lower coordinate (instead of average in bin) used.

LOWER and UPPER may be used together with ALL to produce two points/bin and generate a histogram as shown in `demohist.exg`.

The above averages in general reduce the number of points. To perform a true arbitrary moving average, use

```
AVERAGE MOVE [options] kernal0 kernal1 kernal2 ...
```

where the `kernalN` numeric values specify relative averaging weights for a symmetric moving X,Y average as

$$w = kernal0 + 2\sum_{K=1}^{N} kernalK$$

$$X[i] = X[i]\frac{kernal0}{w} + \sum_{K=1}^{N}(X[i-K] + X[i+K])\frac{kernalN}{w}$$

$$Y[i] = Y[i]\frac{kernal0}{w} + \sum_{K=1}^{N}(Y[i-K] + Y[i+K])\frac{kernalN}{w}$$

AVERAGE MOVE is performed in place, has no effect on TABDATA, by default, drops the N starting and ending points, and uses special options:

X	Average performed only on XDATA.
Y	Average performed only on YDATA.
ZDATA	Take the `kernalN` values from ZDATA array. This allows arbitrary calculations of the averaging kernal.
ALL	The N endpoints are not removed, but averages about endpoints are not symmetric.

Examples

```
average X 100          ; divides x-axis into 100 bins
                       ; and averages based on X

average 25 Y 10 20     ; y-axis from 10-20 -> 100 bins
                       ; and averages based on Y

average 30 all lower upper  ; forms histogram of bin counts
  ydata column 3            ; see example in demohist.exg
  draw line red fill 0

average move Y ALL 1.0 0.5  ; 3-point moving average of YDATA
```

In the last case $Y[i] = (0.5Y[i-1] + Y[i] + 0.5 * Y[i+1])/2$. See `demomavg.exg` for other examples.

5.3 AXIS

AXIS [options]

This command draws the AXIS box, sets scaling transformation for later CURVEs and provides tic marks and numeric labels according to the current parameter settings. If XMIN/XMAX and YMIN/YMAX have not been explicitly set the target axis limits are taken from the current X and Y data. The AXIS (or BOXAXIS) command must precede any CURVE or LABEL command. The page size and AXIS position are determined by the LEFT, XSIZE, RIGHT, BOTTOM, YSIZE, TOP parameters in inches. If LEFT<0 the page width is forced to 8 inches. More exact control of AXIS is possible with the BOXLIMIT and TICMARKS commands.

GRID	Uses solid grid lines instead of tic marks (default).
GRIDDOT	Uses dotted grid lines instead of tic marks (default).
GRIDDASH	Uses dashed grid lines instead of tic marks (default).
LOGX	Logarithmic x-axis with base 10 major tics.
LOGY	Logarithmic y-axis with base 10 major tics.
LOGLOG	Synonym for LOGX LOGY.
INVISIBLE	Sets up all scaling, but produces no graphics.

itemRIGHT, LEFT, TOP, BOTTOM add axis box side and places tics and numbers only on the specified sides. These options work in pairs: LEFT-RIGHT and TOP-BOTTOM. If neither option from a pair is given, BOTH sides will be drawn, BOTH sides will have tics, but ONLY THE DEFAULT side (LEFT or BOTTOM) will have numeric labels. Thus, AXIS LEFT will draw default top and bottom sides, and the left side, but the right side will be completely omitted. AXIS LEFT ALL will include the right box side without tics and labels.

If ISYM=6 or 16 (ARROW or OARROW synonym), the individual lines will be extended and arrow heads will be added. Try

```
AXIS LEFT BOT ARROW SMULT 4
```

To eliminate unwanted arrows (due, for example, to a previous DRAW/LINE ARROW command), use

```
AXIS isym -1  [other_options]
```

ALL	When used with LEFT/RIGHT/TOP/BOTTOM, includes ALL axis box sides even though ONLY the specified sides will have tics and labels.
NOXLABEL	Places tics but no numeric labels on the x-axis.
NOYLABEL	Places tics but no numeric labels on the y-axis.
CENTER	Tic marks will be centered on axis box lines.
NOXTICS	Eliminates BOTH tics and labels from the x-axis.
NOYTICS	Eliminates BOTH tics and labels from the y-axis.

BOX `xmin ymin xmax ymax`
Draws a new axis box with corners at the specified graph coordinates.

BOX `xmin ymin xmax ymax` **INCHES**

BOX `xmin ymin xmax ymax` **CMS**
Draws a new axis box with corners at the specified coordinates in inches/centimeters relative to the current axis box.

BOX `xmin ymin xmax ymax` **INCHES PAGE**

BOX `xmin ymin xmax ymax` **CMS PAGE**
If PAGE is specified as well as INCHES/CMS, then the coordinates are in inches/cm's relative to page lower left corner.

CURSOR Uses the graphic cursor rubber box to place the new axis box relative to any existing graphics.

FIXXMIN, FIXXMAX, FIXYMIN, FIXYMAX
The corresponding limit will not be adjusted for axis scaling and may not have a corresponding tic.

ASIS Allows nonlinear AXIS based on the current XMULT XSHIFT XPOWER YMULT YSHIFT YPOWER parameters. With this, it may be best to explicitly specify the ticmarks. Although the AXIS may be nonlinear, the choice of tics is as usual. Thus, for example, to produce an x-axis as 1/T, use:

```
AXIS XMIN 500 XMAX 100 XPOWER -1 ASIS
```

The following parameters (default values given in parentheses in inches) determine the axis placement on the page:

TOP(0.5)	inches to top of page from top of axis box.
YSIZE(4.0)	axis box height in inches.
BOTTOM(1.5)	inches to bottom of page from bottom of axis box. TOP + YSIZE + BOTTOM determines the page height.
LEFT(-1.0)	when > 0, inches to left of axis box.
XSIZE(6.0)	axis box width in inches.
RIGHT(1.0)	inches to right of axis box. LEFT + XSIZE + RIGHT determines the page width. If LEFT< 0, the page width is 8.0 inches.

The following parameters (default values given in parentheses) vary the appearance of the axis box and numeric labels:

NFONT(12180)	font specification for numeric labels.
ANCOLOR(7)	axis numbers color.
ABCOLOR(7)	axis box color, negative to omit.
WAXIS(12)	axis box line width.

Numerous parameters control the axis limits and scaling:

XDECP(-1), YDECP(-1)	if ≥ 0, number of decimal places in X,Y axis numbers.
NINTX(5), NINTY(4)	target number of axis intervals. The AXIS command will try to adjust limits and tic marks to give NINTX/Y major intervals. This may be overridden with XINT, YINT.
XMIN, XMAX, YMIN, YMAX	override any data extrema to determine the desired axis limits.
AMAJCOL(7)	axis major tics color, negative to omit tics.
WMAJOR(8)	axis major tic line width.
XINT(0.0), YINT(0.0)	when > 0, give exact number of intervals along X,Y axis.

XIND(0.0), YIND(0.0)	indentation fraction, shrinks axis limits along box. 0.05 is a good choice to try.
TICSIZE(0.03)	tic size as fraction of axis box. TICSIZE< 0 extends tics outside of axis box.
LXROUND(1), LYROUND(1)	specifies rounding type on log axis. If L?ROUND=2 or L?ROUND=1 and limit range < 10^3, then the limits are rounded to nearest integer in decade. If L?ROUND=10 or L?ROUND=1 and limit range ≥ 10^3, then the limits are rounded to the nearest decade.
AMINCOL(7)	axis minor tics color, negative to omit
WMINOR(5)	minor tic line width
XTICS(0), YTICS(0)	the number of minor axis tics between major intervals on LINEAR axis and on LOG axis when X/YINTS> 0. For LOG axis with X/YINTS=0, specifies minor tics on integers within each decade:

```
0     integer tics and single digit labels
       when range < 10^3, otherwise nothing
1     integer tics, label only on 3
2     integer tics, label only 2 and 5
3-8   integer tics, all single digit labels
>8    integer tics, no labels
-1    integer tic and label only on 3
-2    integer tic and label only on 2 and 5
<-2   omit all minor tics and labels
```

5.4 BOXLIMIT

BOXLIMIT `xmin ymin xmax ymax` [options]

BOXLIMIT allows explicit control of the AXIS box. Draws an AXIS box without numeric labels or tic marks. The placement is determined by the TOP, YSIZE, BOTTOM, LEFT, XSIZE, and RIGHT parameters. The lower left corner will correspond to the coordinates `xmin,ymin`. The upper right corresponds to `xmax,ymax`. BOXLIMIT should be used with the TICMARKS command.

The allowed options are as follows:

LOGLOG	Log scaling for both X and Y axis.
LOGX	Log scaling for x-axis.

LOGY	Log scaling for y-axis.
INVISIBLE	Scaling is defined, but no elements are drawn.
TOP	Draw TOP of box and only specified other sides.
BOTTOM	Draw BOTTOM of box and only specified other sides.
LEFT	Draw LEFT side of box and only specified other sides.
RIGHT	Draw RIGHT side of box and only specified other sides.
ALL	Draw ALL box sides (default).

If ISYM=6 or 16 (ARROW or OARROW synonyms), the individual lines will be extended and arrow heads will be added. Try

```
BOXLIMIT LEFT BOT ARROW SMULT 4.
```

To eliminate unwanted arrows (due, for example, to a previous DRAW ARROW or LINE ARROW command), use

```
BOXLIMIT isym -1 other_options.
```

The following parameters (default values given in parentheses in inches) determine the box placement on the page:

TOP(0.5)	inches to top of page from top of axis box.
YSIZE(4.0)	axis box height in inches.
BOTTOM(1.5)	inches to bottom of page from bottom of axis box. TOP + YSIZE + BOTTOM determines the page height.
LEFT(-1.0)	when > 0, inches to left of axis box.
XSIZE(6.0)	axis box width in inches.
RIGHT(1.0)	inches to right of axis box. LEFT + XSIZE + RIGHT determines the page width. If LEFT< 0, the page width is 8.0 inches.

The following parameters (default values given in parentheses) vary the appearance of the axis box and numeric labels:

ANCOLOR(7)	axis numbers color.
ABCOLOR(7)	axis Box color, negative to omit.
WAXIS(12)	axis box line width.

Examples

```
boxlimit 0 0 1 1           ; axis box from 0,0 to 1,1

boxlimit 0 0 1 1 left bot  ; same as above but draw
                           ; only left and bottom sides.
```

5.5 BINREAD

BINREAD name filespec [Nmax] [Boff] [options]

BINREAD performs a binary read of at most Nmax data elements from **filespec** starting at Boff bytes from the start of the file. Both Nmax and Boff are optional. If not given, Boff=0, which is the start of the file. If Nmax is not specified, the read will continue until end of the file is reached. Unless otherwise specified, BINREAD reads 8-byte REAL values in the default floating point format for the hardware platform.
The required **name** field may take one of the following forms:

X or XDATA	Reads into XDATA.
Y or YDATA	Reads into YDATA.
Z or ZDATA	Reads into ZDATA.
XYDATA	Reads x,y pairs into XYDATA.
TABDATA	Reads into TABDATA. If existing NCOL> 0, calculate a new NROW based on number_of_values/NCOL. If NCOL=0 and NROW> 0, calculates new NCOL as number/NROW.
REALS	Reads 8-byte REALS to replace current REAL parameters.
INTEGERS	Places current INTEGER parameters with binary read (size and byte-reversal dependent on platform).
ArrayName	Other name for ARRAYDATA.

The filespec specifies the disk file from which data is read. A filespec of * continues with a binary read from the current .exg file.
The allowed options are

CONTINUE	Adds new data to any existing data.
FLOAT	Reads 4-byte REALs in default hardware format.

LONG	Reads 4-byte integers (possible byte reversal).
SHORT	Reads 2-byte integers (possible byte reversal).
BYTE	Reads 1-byte integers (0-255).
REVERSE	Reverse byte-order with SHORT (2-byte) or LONG (4-byte) integers from hardware default.

Examples

```
xydata loop 100 { i sqrt(i) } ; generate dummy XYDATA

binsave dummy.xy xydata       ; save all XY in binary
                              ; as dummy.xy with 1600 bytes

binread xydata dummy.xy 4 48  ; read back only 4 points
                              ; skipping the first 3 pairs
draw dot color 4 smult 1.5    ; plot these

binread xydata dummy.xy 100 112  ; read remainder
                              ; skipping first 7 pairs
draw odot                     ; plot these


; read first 300 binary atom position x,y into a1xy
; read remainder in 2nd array a2xy
; take note of different OS integer storage

if {&OS.=AIX} then
  binread a1xy demoarra.atmbin short 300
  binread a2xy demoarra.atmbin short 1000 600
else
  binread a1xy demoarra.atmbin short reverse 300
  binread a2xy demoarra.atmbin short reverse 1000 600
endif
array a1xy asis column 2      ; adjust to 2 columns
array a2xy asis column 2      ; adjust to 2 columns
```

5.6 BINSAVE

BINSAVE filespec [options]

BINSAVE performs a binary save of current data as 8-byte REALS (at current precision) onto the file **filespec** and overwrites any current data in the file. If no options are given, the XYDATA is saved.

The allowed options are

CONTINUE	Appends data to filespec instead of replacement (default).
X or XDATA	Saves XDATA.
Y or YDATA	Saves YDATA.
Z or ZDATA	Saves ZDATA.
XYDATA	Saves XYDATA as x,y pairs.
TABDATA	Saves TABDATA.
ArrayName	Saves existing array.
REALS	Saves current REAL parameters.
INTEGERS	Saves current INTEGER parameters (size and byte-reversal dependent on platform).
CURSOR	Displays graphic cursor; saves X,Y position in AXIS units and character entered (17 bytes).
RUBLINES	Displays graphic rubber line; saves endpoints and characters entered (34 bytes).
BOX	Displays graphic rubber box; saves endpoints and characters entered (34 bytes).
Pname	Saves a single INTEGER or REAL parameter value.

Examples

```
binsave dummy.xy              ; default save of XYDATA

binsave dummy.xy zdata cont   ; add ZDATA to same file
```

5.7 COMMENT

COMMENT [x y] [rotate] [options]
    text

COMMENT allows arbitrary placement of text on the graph in font CFONT with color TCOLOR. All text may contain subscripts, superscripts, Greek characters, and plotting symbols/lines. The actual text is contained on the following line. If the text line ends in + or @, an additional text line is read and will be placed beneath the previous. When CENTER, LEFT, or RIGHT is specified, all lines will have the same alignment.

The numeric parameters are

`x y`	the position of the comment in AXIS units (default). Required unless the CURSOR option is used.
`rotate`	optional counterclockwise rotation of the text in degrees. When specified, all other placement options will be ignored.

The allowed options are

INCHES **CMS**	The comment will be `x,y` inches/centimeters from the lower left corner of the axis box.
INCH PAGE **CMS PAGE**	The comment will be `x,y` inches/centimeters from the lower left corner of the page.
CURSOR	The graphic cursor will be used to position the comment. With some UGH drivers the `"V"` character will allow a temporary view of the label position. The comment position will be saved if a RECORD command has been issued.
CURSOR `rotate`	Same as CURSOR but the text will be rotated by `rotate` (degrees) counterclockwise about `x,y`.
CURSOR BOX	Uses a mobile BOX of the final comment size to position the comment. `"V"` may be used for a temporary view.
VERTICAL	The text will rotated 90 degrees, independent of the sign of CFONT.
CENTER	The text will be centered at the `x,y` position.
LEFT	The left edge of the label will be at the x position. This is the default placement.
RIGHT	The right edge of the label will be at the x position.
LOWER	The lower edge of the label will be at the y position. This is the default placement.
UPPER	The upper edge of the label will be at the y position. These last options may be used together, such as `COMMENT CENTER RIGHT` or `COMMENT UPPER RIGHT`. If CENTER is given, it should precede other options.
ASIS	No **VP** special characters will be interpreted (`_underscore_`, `|box|`, `?symbols?`). Begin text with `~-13~` for no UGH special characters.

Examples

```
comment 3 4
Simple text with lower left corner at 3,4.

comment 6 4 upper right inch
Multiple lines of text+
with upper right corner+
6,4 inches from lower-left corner of axis.

comment cursor 45
Placed with graphic cursor, text at 45 degrees.

comment { xsize ysize+0.2} inch right  ; place time stamp
&date. &time.
```

5.8 CURVE

```
CURVE [indexmin indexmax] [options]
 DRAW [options] ;synonym for CURVE ISYM -1 ILINE
-1
```

CURVE plots the current X and Y data on the current axis according to the ISYM, ILINE, PRUNE, WIDTH, SMULT, and COLOR parameters. The data are scaled before plotting as

$$X_{plot} = (XMULT \times X + XSHIFT)^{XPOWER}$$

$$Y_{plot} = (YMULT \times Y + YSHIFT)^{YPOWER} X_{plot}^{XYPOWER}$$

It is recommended that both ISYM and ILINE be specified (usually with synonyms) on the same line as the CURVE command. The DRAW synonym resets both ISYM and ILINE to blank before the CURVE command.

If the synonym SYMBOL has been defined, the text in SYMBOL will be centered at each x,y point in the current CFONT, and ISYM will not be used. See `demomac.exg` for sample use of the SYMBOL synonym.

Calculations may be used instead of the actual XYDATA for plotting (See Section 3.2 for a detailed discussion). The calculations may be specified as:

{ normal calculations } or [HSILOP calculations]
: Use normal {...} or HSILOP (reverse POLISH) [...] calculations to define the X and Y values to be plotted. Performs an implied loop of size LOOP (or NMBR or NCOL or NROW). It is recommended that LOOP be specified on the command line. If two values are placed on the calculation stack, these are taken as the X and Y values. If three values are calculated, the first is taken as a logical operator (true if INT(value) is nonzero, false if INT(value)=0) to decide if the remaining two values are to be taken as valid X and Y data points.

`indexmin indexmax`
: Only plots points X[i],Y[i] where `indexmin` ≤ `i` ≤ `indexmax`. Cannot be used with calculations. If used with FILL, `indexmin indexmax` must follow the FILL `xfill` (or `yfill`) number.

Fill options (see `demofill.exg`) fill in an area between the curve and a line:

LINE FILL Y `yfill`
: Draws the current data, then fills in between the curve and the line Y=`yfill`. If `yfill` is omitted, fills to bottom of AXIS box.

LINE FILL X `xfill`
: Draws the current data, then fills in between the curve and the line X=`xfill`. If `xfill` is omitted, fills to left AXIS box edge.

The spline options interpolate between data points (see `demospl.exg`):

SPLINE `nsp`
: Uses a SPLINE fit to draw a cubic spline through the XYDATA points with at most `nsp` (default 10) calculations between original data.

LOCALSPLINE `nsp seglen`
: Uses a local cubic spline to smoothly connect data points with at most `nsp` (default 10) calculations between original data. The 2 points on either side of each segment are used to determine the tangents at the ends. `seglen` (default 1.0) gives the relative length of the connection between the endpoints. Try

```
DRAW LINE LOCAL 10 2} or 10 0.5
```

to see the effect of changing `seglen`.

BETASPLINE `nsp beta1 beta2`
: Uses a Beta-SPLINE fit to draw a smooth curve with the XYDATA as "control" points. This will not, in general, pass through the original data. At most `nsp` new points will be calculated between each pair. `Beta1` and `Beta2` govern the closeness of the control. Use `1 0` for minimum control, `1 5` for intermediate, `1 20` for close approach.

CLOSE
: When used with one of the spline options, connects the last point to the first to form a closed curve.

SAVE
: When used with one of the spline options, saves the calculated data points in TABDATA.

END
: When used with one of the spline options, omits all symbols except for the last point. This gives a useful way to add a curving arrow.

Numerous parameters (defaults given in parentheses) alter the graphics produced by CURVE or DRAW:

ISYM(1) symbol number,

```
<10 solid, >10 open, -1=omit, 0=invisible,
 1=dot,       2=diamond,    3=triangle,    4=square,
 5=del,       6=arrow,      7=plus,        8=X
 11=circle, 12=odiamond, 13=otriangle, 14=osquare,
 15=odel,    16=oarrow
```

ILINE(0) line type

```
-1=omit, 0=invisible, 1-4 include symbols,
1=lines connecting symbols, 11=lines, no symbols
2=pattern 1 line + symbols, 12=pattern 1 only,
3=pattern 2 line + symbols, 13=pattern 2 only,
4=pattern 3 line + symbols, 14=pattern 3 only,
  Default pattern 1 gives "dotted" line.
  Default pattern 2 gives "dashed" line.
  Default pattern 3 gives "dot-dashed" line.
Use the PATTERN command to define a new patterns.
```

PRUNE(1)	0 to allow plotting outside axis box limits; a value > 0 truncates lines at the box edges and omits outside symbols.
COLOR(1)	current plotting color for lines and symbols.
SMULT(2.0)	symbol size multiplier.
XMULT(1.0)	X scale factor.
YMULT(1.0)	Y scale factor.
XSHIFT(0.0)	X scale add shift value.
YSHIFT(0.0)	Y scale add shift value.
XPOWER(1.0)	X scale power exponent.
YPOWER(1.0)	Y scale power exponent.
XYPOWER(0.0)	XY scale power exponent to scale Y values by X.

Examples

```
draw dot red         ; simple plot using synonyms DOT and RED

curve line fill y 0 blue    ; fill to y=0
curve line yellow           ; line on top

draw dot blue { y>0 x y }   ; plot points with y>0

draw line cyan { x*x y*y }  ; plot y*y vs x*x

draw line green spline      ; different spline options
draw line yellow beta       ; on same x,y data
draw line white local close
```

5.9 DOFILE

DOFILE filespec(*/stdin) `max-lines` options >pattern

DOFILE processes each line of the file named in filespec according to pattern as a **VP** command. Each line of filespec is parsed into at most 9 fields (separated by spaces) `#1 #2` ... `#9` (`#9` contains the remainder of the line after 8th field). These are substituted for each occurrence of `#1 #2` ... `#9` in the pattern, and the result is interpreted as a **VP** command.

Any lines of the file that begin with * are treated as comments and are not processed.

During execution the real parameter FLINE is set to the file line number currently being processed.

The allowed options are

filespec	The name of the file to open and start processing, or
*	to continue with previously opened file, or
stdin	to read the lines from standard input (keyboard).
`max-lines`	Maximum number of lines to process (default 32768) before a blank line or end of file.
CLOSE	Close the file at end. Cannot be followed by DOFILE *.
SKIP `n`	Skip the next `n` lines of the file.
INCLUDE	Includes all lines of file (up to blank or EOF) after pattern (without `#n` substitution) and executes as an internal file. Useful to read comments/labels from file.

Examples If the file `test.pat` contains

```
1 2 comment at 1,2
5 6 comment at 5,6
  (blank line)
X axis label
Y axis label
A simple caption
```

and the macro XYCOM is defined as

```
macro xycom
comment &%1. &%2.
&%3.
return
```

then the command

```
DOFILE test.pat >xycom #1 #2 '#3 #4 #5 #6 #7 #8 #9'
```

will read test.pat until the blank line and place the two comments using the macro XYCOM as

```
xycom 1 2 'comment at 1,2'
xycom 5 6 'comment at 5,6'
```

The command

```
DOFILE * include >label
```

will then read the remainder of test.pat as input data for the LABEL command as:

```
label
X axis label
Y axis label
A simple caption
```

The command

```
DOFILE test.pat 1 skip 1 close >xycom #1 #2 '#3 #4 #5 #6 #7
                                          #8 #9'
```

will only process the 2nd line in test.pat, then close the file.

See `barchart.exg` for the use of DOFILE to generate simple bar charts from a type of data file. Try `barchart demobar`.

See `piechart.exg` for the use of DOFILE to generate simple pie charts from a type of data file. Try `piechart demopie`.

5.10 ECHO

```
ECHO message
```

ECHO substitutes for all `&` . synonyms in message and output to CONSOLE. Useful for tracking progress of MACROs and REPEAT-END loops and checking synonym substitution. If RECORDing is ON, `"message"` with substitutions will be placed in the record file.

Examples

```
echo the XDATA limits are xdmin &xdmin. xdmax &xdmax.

echo the parameters are %1 &%1. %2 &%2. %3 &%3. %4 &%4.

echo EOF reached
```

5.11 FIT

```
FIT [order x0 y0] [options]
```

FIT generates a least-squares fit of all x,y data pairs within the current axis limits to a polynomial or specified order (default `order=1` for a linear fit). Up to 19 terms may be specified. For Log coordinates the fit uses log_{10} of the position. Thus, with a fit from LOGLOG, the slope

gives the power-law exponent. The polynomial is calculated relative to an optional origin (default `x0,y0 0,0`) as

$$(y - y0) = \sum_{n=0}^{\mathtt{order}} C[n](x - x0)^n.$$

The coefficients and reduced Chi of the fit are typed and their values are assigned to REAL parameters Y0, SLOPE, Cn for n>1 and CHI. For linear fits (`order=1`) the Y0 and SLOPE errors are calculated and defined as EY0 and ESLOPE. The "correlation coefficient" RAB and its error ECORCO are also defined. The FIT is shown on the graph in the current color and current ILINE type (e.g., FIT DOTTED will draw the fit in a dotted line).

The allowed options are

ALL Uses ALL x,y points including those outside axis limits.

CURSOR `order`
Uses the graphic cursor rubberbox to indicate the limits of the points to be fit. If "0" is entered to select the last fit point, the cursor is then used to indicate the origin `x0,y0` (press "Q" for default origin).

BOX `xmin ymin xmax ymax order x0 y0`
Only fits the points inside the specified limits.

MOMENTS Only calculates the first `order` moments of the N XDATA and YDATA points (relative to `x0,y0`) that fall within the CURSOR or BOX limits. These are printed and defined as real synonyms MXn, MYn:

$$MXn = N^{-1}\sum(x - x0)^n$$

$$MYn = N^{-1}\sum(y - y0)^n.$$

Thus, MX1 is the average X; MY1 is the average Y.

Examples

```
draw dot red              ; plot XYDATA as red dots
fit 2 yellow              ; quadratic fit as yellow line
fit blue dotten           ; linear fit as dotted blue line
fit green box 1 4 10 10 3  ; cubic fit of part of data
                           ; drawn as a green line

fit moments 2 all          ; average X as MX1, Y as MY1
                           ; X*2 as MX2, Y*Y as MY2

restart default            ; start a self-contained example
xydata loop 10 { i i*i }   ; generate some exact data
axis loglog                ; log-log axis
draw dot red               ; plot as red dots
fit yellow                 ; linear fit gives exponent
echo The exponent is &slope..
```

See `demofit.exg` for other examples.

5.12 FILTER

FILTER fromdata todata prog_name [options]

The FILTER command allows an external program to process **VP** internal data arrays (XDATA YDATA ZDATA XYDATA TABDATA) and return results. The FILTER command is not available in DOS, but the SYSTEM command can be used for many of the same functions.

fromdata	is one of X Y Z XYDATA TABDATA NULL and specifies which **VP** internal data array (or NULL for none) is piped to stdin of prog_name for processing.
todata	is one of X Y Z XYDATA TABDATA NULL and specifies which **VP** internal data array (or NULL for none) is filled from stdout of prog_name after processing.
prog_name	is the name of an executable program that can process 8 byte reals as stdin to 8 byte reals as stdout. Any messages generated by the program must be written to stderr.
options	the remaining text of the filter command (after resolution of **VP** synonyms) is passed to prog_name as command line arguments.

Several sample filters for the IBM RISC 6000 are included. The source code of the filters is included in vp6_acc.tar.

vp_urand	Generate uniform random numbers.
vp_dydx	Estimate dY/dX for a series of X,Y data points.
vp_integ	Trapezoid integration of Y(x).
vp_fft	Forward and reverse FFT transforms or spectral density.
vp_acor	Sequence autocorrelation.
vp_csort	Sort an array on a particular column.

For the above programs that take options,

```
prog_name
```

with no options prints a usage/option summary.

Examples

```
filter null y vp_urand 100 77777  ; generate 100 random YDATA

filter xyd xyd vp_dydx      ; XYDATA becomes dY/dX estimates

filter xyd xyd vp_integ     ; XYDATA by trapezoid integration

filter y tabd vp_fft 512 1 Ncol 2 ; forward FFT of 512 YDATA
                            ; to 257 sin cos pairs in TABDATA

filter y z vp_fft 512 0     ; spectral density of 512 YDATA
                            ; saved as 257 points in ZDATA

filter tabd y vp_fft 512 -1 ; reverse FFT of 257 sin cos
                            ; TABDATA to 512 YDATA values

filter y z vp_acor 1        ; autocorrelation of YDATA (with
                            ; finite-size correction) to ZDATA

filter tab tab vp_csort &ncol. 2 1 ; sort TABDATA on
                                   ; increasing column 2
```

Synonyms of the form &ARRAYNAME. are replaced by the starting address of the data. Thus, under DOS,

```
xdata loop 100 { 0 }           ; make space for 100 values
system vp_rand &xdata. 100 7777 ; generate values
```

will use the DOS program VP_RAND.DOS to place 100 pseudorandom numbers generated from the C library call RAND() with seed 777 into XDATA. The program VP_RAND.EXE and its source code VP_RAND.C are included in the package for DOS.

5.13 GEDIT

GEDIT DIVIDE `xmin ymin xmax ymax [MinCol MaxCol NewCol]` [options]

GEDIT DIVIDE divides the elements of the current UGH file into two new files, inside.ugh(bin) and outside.ugh(bin), depending on whether they fall inside or outside the specified limits. The elements in the current UGH file remain unchanged. Lines are cut at the edges. All other elements are treated as an indivisible entity. If any part of an element (other than a line) is outside, it is placed in outside.ugh. Nonpositional elements such as font changes are placed in both new files. The bound limits `xmin ymin xmax ymax` are in current axis units. If INCHES PAGE is specified, the bound limits are in inches from the lower left corner of the PAGE (not the axis). If the numbers `MinCol` and `MaxCol` are specified, only elements with `MinCol`≤color≤`MaxCol` are considered INSIDE. If `NewCol` is specified in addition to `MinCol` and `MaxCol`, then the INSIDE elements will also have their color changed to `NewCol`.

The allowed options are

CURSOR `MinCol MaxCol NewCol`	Uses the graphic cursor to define the INSIDE/OUTSIDE box.
INCHES **CMS**	The limits are in inches/centimeters from the lower left corner of the axis box.
INCH PAGE **CMS PAGE**	The limits are in inches/centimeters from the lower left corner of the page.

GEDIT INCLUDE filespec `[xmin ymin xmax ymax]` [options]

GEDIT INCLUDE includes the element of the specified file in the current UGH file and scales the elements to fit in the specified bounding box. In general this will give a change of aspect ratio. The include filename MUST be the third field in the command. If no extension is given, .ugh(bin) will be appended.

The allowed options are

CURSOR	Uses the graphic cursor to define the bounding box.
INCHES **CMS**	The bounding box coordinates are in inches/centimeters from the lower left corner of the axis box.
INCH PAGE **CMS PAGE**	The bounding box coordinates are in inches/centimeters from the lower left corner of the page.
ASIS	The included elements are scaled by the current XMULT and YMULT parameters and the origin (0,0) of the included file is mapped to the lower left corner of the specified bounding box.
ASIS FILL	The included elements are scaled by the current XMULT and YMULT parameters and the lower left corner (as opposed to 0,0) is mapped to the lower left corner of the specified bounding box.
CENTER	The included elements are centered in and scaled to fill the bounding box with no change in aspect ratio.
IMAGE	The included filespec contains *only* UGH image data (.uim(bin) files). This option does add an UGH command 37 (image include) to the current UGH file but does not include the image data itself.

The two GEDIT functions provide a relatively convenient way of "editing" UGH files. Portions of a file may be deleted, selected for inclusion in another file with rescaled sizes, or moved elsewhere. Since the UGH elements are stored in integer coordinates, rescaling by large amounts may result in significant rounding errors.

Examples

```
gedit divide -.4 -.4 2.70 1.9 inch    ; remove a portion

gedit divide 4 -.4 8 3.0 inch 0 100 6        ; remove and
gedit include inside 0 0 1 1 inch page asis  ; change color

gedit include watson image center 1 5 9 9    ; image include
```

5.14 IF

```
IF {...} STATEMENT
  IF ... STATEMENT
```

The IF command executes STATEMENT as a **VP** command when the result of the normal {...} or HSILOP [...] calculation is TRUE (i.e., INT(value)≠0). Multiple **VP** commands may be included in STATEMENT with the logical end-of-line character ; The IF statement may also be used in the compound form:

```
IF {...} THEN           ; or   IF [...] THEN
  STATEMENT T1          ; T1 - Tn are executed if {...} true
  ...
  STATEMENT Tn
ELSE
  STATEMENT F1          ; F1 - Fm are executed if {...} false
  ...
  STATEMENT Fm
ENDIF
```

In the compound form THEN, ELSE, and ENDIF must be spelled completely.

NOTE: Only one level of IF statements is supported. Nested IF statements will yield unpredictable results.

Examples

```
if { nmbr>10} fit 3              ; conditional fit

color { color+1 }                ; increment color number
if { color>7 } color 2           ; but reset to 2 if > 7

read wait Hit Q to quit, otherwise continue
if { &%1.Q == QQ } read end     ; conditional file exit

if { &%0. == CONSOLE } comm cursor ^Theory ; multiple lines

if { &%0. == CONSOLE } then     ; IF THEN ELSE example
  comment cursor
  Theory
else
  comment 4 4
  Theory
endif
```

5.15 LABELS

```
LABELS [options]
  x label text (one line)
  y label text (one line)
  caption line 1
  ...
  caption line n
  (blank line )
```

This command labels the x- and y-axis and adds a caption below the axis box. All text may contain subscripts, superscripts, and Greek characters as described in Section 3.4. The font specifications XFONT, YFONT, and CFONT are used for the different elements. The Y label is typically in a rotated font. The caption text will be concatenated to form a single block below the axis. The caption breaks lines only between words. The REAL parameter CCSIZE gives the approximate number of characters a line is allowed to extend beyond the right margin before being split. The spacing between lines may be changed with the LSPACE parameter.

Note: The default x-axis label (BOTTOM) is placed below any current axis numeric labels or prior labels or captions. Successive LABEL X or LABEL CAPTION commands will be placed below the previous. A new AXIS command resets the LABEL positions.

The allowed options are

X	Include x-axis label.
Y	Include y-axis label.
CAPTION	Include caption.

The above options are used to include only the specified elements. Thus LABEL X specifies that only the X label follows, while LABEL Y CAPTION indicates that both the Y label and the caption follow. The text of the elements must always be in the order X, Y, CAPTION. Other options include

CAPTION `[L_margin R_margin] [top_pos] [inches/cms] [page]`
Provides explicit control over CAPTION placement. If only 2 numbers are given, top_pos is taken as usual (below X label). LABEL CAPTION also allows LEFT, RIGHT, or CENTER as options to force placement of small (or user-specified by +) lines. **Note:** `LABEL RIGHT` alters the Y label but not CAPTION. `LABEL CAPTION RIGHT` alters the caption placement. `LABEL Y CAPTION RIGHT` alters both.

To avoid confusion, specify separate LABEL Y and LABEL CAPTION commands.

If INCHES or CMS are given the units of caption placement will be inches/centimeters from the lower left corner of the axis box. The default is INCHES.

If PAGE is given the units of caption placement will be inches/centimeters from the lower left corner of the page.

TOP	Puts X label on top of axis box (default is BOTTOM).
RIGHT	Puts Y label at right of axis box (default is LEFT).
ASIS	No **VP** special characters will be interpreted (_underscore_, |box|, ?symbols?). Begin text with ~-13~ for no UGH special characters.

Examples

```
labels                   ; the default includes all
simple X label
simple Y label
a one-line caption
                         ; blank line ends LABEL command

label y tcolor 4
y-axis label only (and in red)

label x top
x-axis label on top of the axis box

label caption { 0 xsize+2 -1}
This caption starts 1 inch below the axis box
and extends 2 inches to the right.

label caption center
This caption+
consists of 4 lines+
each ended by a plus sign+
that are centered under the axis box
```

5.16 LINE

```
LINE x1a y1a x2a y2a [options]
  x1b y1b x2b y2b
  x1c y1c x2c y2c
  (additional lines)
```

This is the basic command for drawing lines. Line endpoints in current axis units follow on successive lines (4 points: `x1,y1 x2,y2` per line) that will be plotted using the current ILINE. If ILINE initially ≤ 0, ILINE 11 will be used for plotting. LINE temporarily omits symbols by setting ISYM=-1 (except for arrows) and restores later. New parameter values such as COLOR, ILINE, and ISYM (to include symbols) may be changed for each line by including the parameter name and value on the same line as the endpoints. Lines will be truncated at the axis boundary. To draw outside the current axis, set PRUNE 0. Use the PATTERN command to define new types of dotted or dashed lines.

The allowed options are

filespec	Indicates that the LINE data are to be taken from filespec. When used, filespec must be the 2nd field on the command line. Other options may follow.
BOX	The 4 numbers on each line give opposite corners of a box that will be drawn in the current color. If you want the box to be solid, set WIDTH<0 or use the FILL option in addition to BOX. This is a simple way of "erasing" parts of a graph. A COLOR of 0 (the background) with WIDTH<0 will effectively erase everything the box covers on monitors and most hardcopy output. Unfortunately, there is no erase for plotter output.
HATCHBOX `dx dy`	The 4 numbers on each line give opposite corners of a box that will be hatched with lines in vector direction `dx,dy` (in UGH units) and separated by `SQRT(dx**2+dy**2)`.
ELLIPSE	The 4 numbers on each line give opposite corners of a limiting box for an ellipse in the current color. For a solid ellipse, use the FILL option in addition to ELLIPSE or set WIDTH<0.
FILL	May be used with BOX or ELLIPSE to provide filled elements.
CURSOR	Indicates that cursor-controlled rubber lines will be used to plot successive lines with the current ILINE.

Input is ended with a "Q." Remember that the endpoints will be saved if the RECORD command has been issued. When used with BOX, HATCHBOX, or ELLIPSE, CURSOR uses a rubber box instead of rubber line.

{ normal calculations } or [HSILOP calculations]
Use normal {...} or HSILOP (reverse POLISH) [...] calculations to define the numeric values (see Section 3.2). Any calculations should produce 4 numbers.

Examples

```
line 1 2 6 7 red        ; simple line in red (color 4)

line dotted arrow blue  ; dotted line ended by arrow
  1 2 6 7               ; from 1,2 to 6,7

line 1 2 6 7 red        ; multiple lines
 3 2  6 5  green        ; in different colors
 8 6  2 1  blue

                        ; calculations from XYDATA
line red box fill { x-0.05 y x+0.05 0 }

line box fill color 0  ; erase part of graph
6 2 9 4
line hatch 50 50 white ; replace same area by hatching
6 2 9 4
line ellip fill yellow ; filled ellipse in same area
6 2 9 4
line ellip wid 10 red  ; with a red outline
6 2 9 4
```

5.17 MACRO

```
MACRO Mname
  vp commands
  ...
  RETURN
```

MACRO starts and names a **VP** internal MACRO definition. All following lines up to RETURN are stored as internal MACRO Mname. Mname may then be used just like any other **VP** command. The effect is the same as READing an external .exg file, but the processing is faster. MACROs are best for short sequences of commands that are used frequently with slight variations. As with EXG file input, the synonyms %1 to %8 are defined from the fields following Mname. RESTART DEFAULT does not remove MACRO definitions. To remove all MACRO definitions, use RESTART MACRO.

Examples

```
macro pfit                  ; PFIT Xcolumn Ycolumn color
  xydata column &%1. &%2. ; get XYDATA from arguments
  draw dot color &%3.      ; plot data in specified color
  fit                       ; show linear fit in same color
return

macro labdot                    ; plot XYDATA as labeled dots
  smult 4.5                     ; large symbols
  cfont 12075 fontw 0.4         ; small font
  define symbol ''              ; reset symbols
  draw odot ofcolor 0           ; blank dot centers
  draw dot define symbol &%1.  ; 1st argument as symbols
  return
```

may be used as

```
pfit 1 2 6                  ; X=c1, Y=c2 shown in yellow
pfit 1 3 4                  ; X=c1, Y=c3 shown in red

labdot A                    ; plot circles with A in center
labdot !a!                  ; plot circles with Greek alpha
```

5.18 PATTERN

```
PATTERN on1 off1 on2 off2 patnumber
```

PATTERN defines a new onoff line pattern for CURVEs drawn with ILINE 2 3 4 (which include symbols) or 12 13 14 (which omit symbols). The on1 off1 on2 off2 are in units relative to the line WIDTH. At least 4 numbers must be specified. If patnumber, is not given it is assumed to be 1. Three patterns may be active at one time. Patnumber of 1 or 12 sets the first pattern for ILINE = 2 or 12, 2 or 13 sets for ILINE = 3 or 13, while 3 or 14 sets for ILINE = 4 or 14.

The default patterns

```
PATTERN 2 6 2 6 12
PATTERN 8 6 8 6 13
PATTERN 2 6 8 6 14
```

give dotted lines for ILINE 2 or 12, dashed lines (longer on) for ILINE 3 or 13, and a dot-dash line for ILINE 4 or 14.

Examples

```
pattern 12 4 12 4 13    ; use LONG dashes
pattern 12 4 12 4 2     ; same as above, 2nd pattern

pattern 2 2 2 2 12      ; close spaced dots
pattern 2 2 2 2         ; same as above, default patnum
```

5.19 POLYGON

```
POLYGON dx dy [x0 y0]
```

POLYGON draws a hatched polygon from the current XYDATA with hatch lines in direction `dx,dy` separated by `SQRT(dx*dx+dy*dy)` similar to LINE HATCHBOX. The "phase" of the hatching is determined by the condition that one hatch line goes through the point `x0 y0` (default `0 0`). `dx dy x0 y0` are in UGH units (default 800 UGH units = 1 inch). If `dx dy` are omitted, the default value of 4 times WIDTH is used.

The parameter OFCOLOR controls how the outline is drawn.

If OFCOLOR < 0, then hatch lines will be draw in the current color and the outline will not be drawn.

If OFCOLOR ≥ 0, then hatch lines will be draw in OFCOLOR and an outline will be drawn in the current COLOR.

Examples

```
xydata column 1 2               ; get TABDATA into XYDATA
poly -50 50 magenta ofcol -1    ; hatch upper left to
                                ; lower right
poly cyan ofcol 4               ; lower left to upper right
                                ; outline drawn afterwards
```

5.20 PRIMITIVE

```
PRIMITIVE {normal calculations} [HSILOP
calculations]
  number color width x1 y1 ... xn yn
  (additional lines)
```

Lines of numeric data following the PRIMITIVE command specify UGH graphic elements that will be converted from axis units and added to the current UGH file. One primitive is specified on each line where number is the UGH primitive number: 20=triangle, 21=trapezoid, 22=box, 23=polygon, 30=ellipse, 32=line, 35=vector; color is the element color

number, width is the element width in screen units (a width < 0 gives a filled-in element), and `xi,yi` are the coordinate pairs in current axis units. The elements will not be pruned to the axis boundaries.

VP implements partial arcs with a variation of the UGH 30 command as

```
30  color width Rx Ry x0 y0 theta1 theta2 edge\_col
```

where `Rx,Ry` are the radii, `x0,y0` is the center, and the arc is drawn from `theta1` counterclockwise to `theta2` in radians. If `edge_col` is given, the central rays are drawn in this color. Use `edge_col=0` to erase them. The pie chart example piechart.exg uses this arc primitive.

{ normal calculations } or [HSILOP calculations]
Use normal {...} or HSILOP (reverse POLISH) [...] calculations to define the numeric values instead of taking them from the following lines (see Section 3.2).

Special PRIMITIVE cursor commands can be used for graphics input in **VP** axis units (UGH cursor commands return UGH coordinates). The values are returned in the REAL parameters CX1, CY1, CX2, CY2 and the characters entered in the SYNONYMS C1 and C2.

PRIMITIVE 17	CURSOR, sets CX1, CY1 and returns C1 as character.
PRIMITIVE 18	RUBBER LINE, sets CX1, CY1, CX2, CY2 and C1, C2.
PRIMITIVE 19	RUBBER BOX, sets CX1, CY1, CX2, CY2 and C1, C2.

The PRIMITIVE cursor coordinates are not automatically saved in a RECORD file. If you want to save them issue an ECHO command as `ECHO &cx1. &cx2.` after the PRIMITIVE command.

The demoprim.exg file demonstrates a number of PRIMITIVE examples.

Examples

```
prim
20 12 -1 1 1 3 5 9 2.5                 ; filled triangle
21  4 40 1 1 3 5 8 6.5  7.7 0.3        ; open trapezoid
22  1 -1 2 3 5 5                        ; filled rectangle
30  7 50 3 2 5 5                        ; open ellipse
32  6 100 0.5 9 8 1.3                   ; line
30  6 -1 1 1.3 8.5 8.5 0.28 6.0         ; happy face as arc
30  2 -1 1 1.3 8.8 8.5 6.0 6.56
```

```
primitive 17                          ; read cursor coordinates
if { (&C1.==Q) or (&C1.==q) } read EOF   ; quit if Q entered

primitive { 30 4 -1 c3 c3 c1 c2 }  ; draw circles of varying R
                                   ; c3 = R, c1,c2 = center

macro SLICE  ;SLICE r Acenter Asize label ; macro for slices
  define a2 { &%3./2 }
  prim {30 color width &%1. &%1. 0 0 &%2.-a2 &%2.+a2}
  comm { 0.5*&%1.*Cos(&%2.) 0.5*&%1.*Sin(&%2.) } center
  &%4.
return

axis xmin -5 xmax 5 ymin -5 ymax 5 invis ysize { xsize }
define pi {4*atan(1)}
slice 5 0 {pi} "half of pie" width -1 blue
slice 5 {0.75*pi pi/2} 1/4 width 10 red
```

5.21 PRINT

PRINT [names] [options]

With no options, PRINT will print a listing of all parameters, their current values, and all nonblank synonyms on the terminal. Options specify subsets for printing:

INTEGERS	Prints all integer parameters and their values.
REALS	Prints all real parameters and their values. The number of decimal places is determined by the parameter NDECPL. If NDECPL < 0, exponential notation is used with -NDECPL places.
SYNONYMS	Prints the definitions of all non-blank synonyms.
MACRO	Print the names of all defined MACROs.
ARRAY	Print the names and characteristics of data arrays.
Pname	Print parameter beginning with pname. Thus, `PRINT XDM` can be used to print the current X data limits, XDMIN and XDMAX.
REVERSE	Reverse rows and columns with TABDATA and arrays.
ArrayName `inc places field`	Print a listing of the current data values incremented by `inc` (default 1) with `places` decimal places (default NDECPL); neg `places` uses E notation in fields of size `field` (default 12). ArrayName may be XDATA, YDATA, ZDATA, XYDATA, TABDATA, or a name defined with the ARRAYDATA command. When RECORDing, the values will be appended to the RECORD file instead of typed at the console.

Examples

```
print int              ; print the integers

ndec -4                ; set E-notation for reals
print real             ; print real values

record sampdata        ; start RECORDing on sampdata.exg
print xydata 1 3       ; xydata, 3 places, to sampdata.exg
record                 ; stop recording
```

5.22 QUIT

QUIT

This is last command evaluated by **VP**. For the executable program **VP**, all open files will be closed, internal storage will be released, and the program will return to the operating system. When the **VP** subroutines are linked with other executable code, **VP** will return to the caller without releasing internal storage and will close open files, but save the position. If the subroutines are reentered, **VP** will restore the state from the previous QUIT.

Example

```
quit                   ; final command
```

5.23 READ

READ filespec field1 field2 ... [options]

READ begins reading **VP** commands from filespec (which may contain a complete path specification). If filespec does not contain a "." the suffix `.exg` is appended. Reading continues from filespec until EOF (end of file) or a new read is encountered. On EOF, control returns to the calling file. A maximum of 10 files may be open at one time. The parameters %1 to %n are defined as field1 to fieldn before the read starts. The synonym

%0 will be defined as filespec (minus any path specification), and the synonym PATHEXG will contain the path information.

If the file is not found in the current directory, subdirectories given in the EXGPATH environment variable will be searched followed by those in the environmental variables VPUGHPATH and PATH.

If the integer parameter SKIP > 0 at the start of a READ command, then the first SKIP records will be skipped.

The allowed options are

RETURN	Interrupts current `.exg` processing. RETURNs control to calling file or console.
CONTINUE	Resumes reading after previous READ RETURN. READ RETURN and READ CONTINUE allow interruptions in the processing of a `.exg` file that are useful for debugging.
EOF	Stops all processing of `.exg` file. Remainder of the file is ignored.
END	Stops all file reads, closes all open files, and returns to console input.
WAIT message	If WAIT immediately follows the READ command, then message will be printed on the console and **VP** will WAIT for a keyboard response (ended by the return key). The %1 to %8 synonyms are redefined from any fields entered in the response.
WAIT `msec`	If WAIT comes after filespec and any fields, then **VP** enters a loop that, after EOF, waits `msec` milliseconds (default 100) and rereads the file whenever it is modified. This can conveniently be used with any editor in a multitasking environment to make changes interactively to an .exg file. Each time the file is saved by the editor, it is automatically reprocessed by **VP**. Use Ctrl-C in the **VP** text window to stop looping.

Examples

```
read demofit                      ; read the demofit.exg sample

read wait Continue/Stop/Pause?    ; wait for user input
if {&%1.==S} read end             ; stop all reads
if {&%1.==S} read return          ; temporary return to console

read continue                     ; continue from READ RETURN
```

```
read newfig wait 200    ; reads newfig.exg, waits 200 msec
                        ; loops until newfig.exg changes

; the following example accepts arguments to read and plot
; different data files on an existing axis.

;    plotdata.exg  contains
xydata &%1. skip 2      ; read xydata from 1st argument
                        ; skips 1st 2 lines as a label
draw line color &%2.    ; line in color of 2nd argument

; this may be used as follows

axis xmax 100 ymax 100     ; appropriate axis limits
read plotdata file1.dat 4  ; 1st data color 4 = red
read plotdata file2.dat 6  ; 2nd data color 6 = yellow

; since the READ command is assumed if the initial field is
; not recognized, this example could also be processed as

axis xmax 100 ymax 100     ; appropriate axis limits
plotdata file1.dat 4       ; 1st data color 4 = red
plotdata file2.dat 6       ; 2nd data color 6 = yellow
```

5.24 RECORD

RECORD filespec [CONTINUE]

Begins saving all keyboard, file, and graphical input on the file filespec. If filespec does not contain a "." the suffix `.exg` is added. If filespec is omitted, any current RECORDing is stopped. If filespec exits, it will be erased and a new file started unless the CONTINUE option is given. If filespec does not exist, it will be created.

The only allowed option is

CONTINUE	If filespec exits, the new information will be appended to the end.

Examples

```
record save          ; start recording on file save.exg
xydata cursor        ; graphical input of data points
                     ; coordinates saved on save.exg
record               ; stop recording
```

```
ydata { sqrt(y) }    ; scale YDATA by sqrt
record save cont     ; continue recording on file save.exg
print ydata          ; put the new YDATA files in save.exg
record               ; stop recording
```

5.25 REPEAT

```
REPEAT Ntimes
  VP commands
  ...
END
```

REPEAT and END define a block of commands that will be executed `Ntimes` times. During execution the integer parameter COUNT is incremented starting with 1 on the first pass. Outside of the REPEAT-END loop COUNT has value 0. The REPEAT-END loop uses internal MACRO storage. Nesting of REPEAT-END loops is not allowed.

Examples

```
repeat 5                    ; repeat the following commands 5 times
                            ; to process a series of data files
                            ; count indexes files and changes color

  echo plotting data from file file&count..dat
  color &count.         ; set a new color
  plotdata file&count..dat

end                         ; end loop
```

5.26 RESTART

RESTART [options]

RESTART, with no options, erases the current stored UGH data file.
The allowed options are

SAVE	Erases stored UGH data and saves all current parameter values and synonym definitions for a later RESTART DEFAULT. RESTART DEFAULT is issued at the end of `profile.exg` to save the customized values.
DEFAULT	Erases stored UGH data and resets all parameters and synonym definitions to their value at the last RESTART SAVE, then removes any existing XDATA, YDATA, ZDATA, TABDATA and ARRAYs.
ASIS	Like DEFAULT, but XDATA, YDATA, ZDATA, TABDATA, and ARRAYs are preserved.
MACRO	Erases all internal MACRO definitions.

Examples

```
restart default       ; recommended 1st line in all .exg files
                      ; to set a consistent state

restart save          ; save current parameters/synonyms

restart asis macro    ; like RESTART DEFAULT, preserves arrays
                      ; and erases all MACROs
```

5.27 SAVE

SAVE [filespec] [options]

SAVE specifies the destination of the binary UGH graphic elements generated by **VP**. With no options, SAVE halts all UGH data saving.
The allowed options are

filespec	Begins saving UGH graphic data on filespec. If filespec does not contain ".", `.ugh(bin)` is appended. If the synonym UGHPATH has been defined, it is used as a prefix. If filespec already exists, and the option CONTINUE is given after filespec, the UGH data will be appended at the end. If CONTINUE is not given, any old data will be erased. If filespec ends in `.exg`, then `.exg` will be replaced by `.ugh(bin)` to prevent overwriting an existing `.exg` file.
CONTINUE	Appends new UGH data to existing data in filespec.
SCREEN	Sends UGH data directly to the graphics monitor.

The command

```
SAVE filespec SCREEN
```

saves the UGH data on filespec and displays it on the graphics UGH window as it is generated.

Examples

```
save &%0.        ; recommended command to save the binary UGH
                 ; graphics on a {\tt .ugh(bin)} file with the
                 ; same name as the {\tt .exg} file.
axis             ; default axis
view erase fill  ; view current graphics scaled to screen
                 ; view additional graphics as generated
save &%0. continue screen

save dummy       ; save binary UGH on dummy.ugh(bin)
```

5.28 SYSTEM

SYSTEM command line

This will substitute for all &xxx. synonyms in command line and attempt to execute command line as an operating system (DOS, AIX, ...) command. Nonzero return codes will be printed.

Examples

```
system edit demo.exg    ; edit the demo.exg file

system dir *.exg        ; list the .exg files in DOS

system ls *.exg         ; list the .exg files in UNIX

xdata loop 100 { 0 }    ; make space for 100 values
                        ; use the DOS program vp_rand
                        ; to generate random numbers
                        ; &xdata. replaced by DOS address
system vp_rand &xdata. 100 7777
```

5.29 TABDATA

TABDATA [filespec] [options]
    [possible numeric data lines]

The following lines contain an array of tabular data that will be read into the TABDATA array. Values will be added until a nonnumeric field is reached. The data may be in free format but the same amount of numbers must be given on each line.

The allowed options are:

filespec	If TABDATA is followed by a file specification, the data will be read from the specified file. Comment lines may be included within the data. After the TABDATA has been read, any additional lines in the file will be treated as part of a normal `.exg` file. If the integer parameter SKIP > 0 at the start of the file read, then the first SKIP records will be skipped.
CONTINUE	The data will be added as additional rows to the existing TABDATA. The old NCOL will be preserved and a new NROW will be calculated independent of the number of data values on each line. This allows arrays with large NCOL to be read one row at a time.
ROW	A new value of NCOL will be calculated from the existing NROW after the data is read, independent of the number of data values on each line.

{ normal calculations } or [HSILOP calculations]
Use normal {...} or HSILOP (reverse POLISH) [...] calculations to define the numeric values (see Section 3.2). NROW and NCOL should be set explicitly *after* the calculations.

Other options store the data as different binary types:

DOUBLE	Saves 8-byte REALs in default hardware format.
FLOAT	Saves 4-byte REALs in default hardware format.
LONG	Saves 4-byte integers (possible byte reversal).
SHORT	Saves 2-byte integers (possible byte reversal).
BYTE	Saves 1-byte integers (0-255).

Examples

```
tabdata                  ; read a simple 3 column 2 row table
  1 2 3                  ; in free format
  8.3 -3.2e-2 1.1e12

tabdata continue         ; add another row
11  33 65.1

tabdata sample.tab float   ; read as 4-byte floats
```

5.30 TABS

TABS [tab-char] [positions] [options]

This defines a character *tab-char* (such as $) that will be interpreted as a tab in text input and places up to 10 tab stops. Each occurrence of *tab-char* aligns all following text with the next tab position.

If no `positions` are specified, the tab positions will be taken from the next occurrence of *tab-char* in text. This is conveniently used with the TEXTBOX command.

The allowed options specify the units for the tab `positions`:

INCHES **CMS**	The units of `positions` will be inches/centimeters from the lower left corner of the axis box.
INCH PAGE **CMS PAGE**	The units of `positions` will be inches/centimeters from the lower left corner of the page.
CURSOR	The graphic cursor will be used to set the `positions`. A "Q" will end input of the tab `positions`.

See `demofont.exg`, `demosymb.exg`, and `demotext.exg` for examples.

Examples

```
tabs $ 1 2 3 4 5 inch    ; $ as tabs spaced every inch
comment 0 5 inch         ; multiple-line comment with tabs
$stop 1 $stop2 $stop3+
$ $ $ $stop4

tabs $                   ; $ as tabs, from next occurrence
comment 0 5 inch         ; multiple-line comment with tabs
this comment has a $tab+
 $this will come under tab
```

5.31 TEXTBOX

```
TEXTBOX [xmin ymin xmax ymax] [boxcolor] [options]
  text
  ...
  (blank line )
```

TEXTBOX allows simple placement of extensive text information on a graph. This is particularly useful for generating presentation graphics.

Successive lines (up to a blank line) will be read as text with simple control words. The text will be scaled to fit within the specified limits. Each text input line produces one output line. There is no line concatenation. If the parameters XIND and/or YIND are nonzero, the text will be indented by the specified fraction. All text may contain subscripts, superscripts, and Greek characters as described in Section 3.4. The numeric parameters are

`xmin ymin xmax ymax`	Are the required opposite corners of the text bounding box in axis units. The data scaling parameters (XSHIFT, XMULT, ...) are *not* used to scale the position.
`boxcolor`	If > 0, draws the box outline in specified color. If not given, the outline is not drawn.

The allowed options are

CURSOR	The graphics cursor is used to set bounding box.
ASIS	The text is not scaled to fit the bounding box. However, .ce and .ri control words will align the text with the margins. When used with ASIS the ymin specification is not used.
FILL	If `boxcolor` > 0, the background box will be drawn in a solid color before the text is added.
INCHES **CMS**	The bounding box units are in inches/cm from the lower left corner of the axis box.
INCH PAGE **CMS PAGE**	The bounding box units are in inches/cm from the lower left corner of the page.
NULL	No **VP** special characters will be interpreted (`_underscore_`, `|box|`, `?symbols?`). Begin text with `~-13~` for no UGH special characters.

Lines beginning with period (.) are taken as local text placement or formatting specifications according to the initial word:

.right *text*	The *text* will be aligned with right edge [`xmax`]. If *text* not given, all following text will be right justified.
.center *text*	The *text* will be centered between [`xmin`] and [`xmax`] If *text* not given, all following text will be centered.

.left *text*	The *text* will be aligned with left edge [xmin] (default). If *text* not given, all following text will be left justified.
.hr color	Draws a horizontal rule in specified color from [xmin] to [xmax].
.imbed *filespec*	Reads remaining text from filespec.
.indent ns	Indent following lines ns spaces from both right and left edges.
.skip ns	Skip a vertical distance ns*LSPACE.
.set p1 value1 p2 value2	Set **VP** parameters to specified new values.

Only the first 2 letters of each control word are needed.

Examples

```
                                ; sample from demotext.exg
top 0 ysize 6 bot 0 right 0 xsize 8
BOXLIMIT 0 0 8 6 abcol 1  ; axis coordinates = page inches
xind .02 yind .08 ulcolor -4 lspace 1.1  ; set parameters
textbox .5 .3 7.5 1.8 3
\TEXTBOX\ adjusts the following text to just fit
inside the box margins
.ri with optional right
.ce | ~0~center |
or left justification
                                ; ended by blank line

text 0.3 0 7.7 5.7 asis     ; another example from demogeom.exg
.ce                         ; center following
\GEOMETRY\
mathematical language
to describe, relate, and manipulate shapes
                            ; end this textbox, start another
width 20 cfont 12160 c2font 24200 fontwid .42 lspace 1.2
text 0.3 0.1 4.0 4.5 4 asis
.sk 0.8                     ; skip some space
.ce
\Euclidean\
.hr
.sk 0.5
traditional
>2000 years
.sk 0.5
.hr
objects have characteristic
size or scale
.hr
appropriate for man-made objects
nature complicated
.hr
.sk 1
described by algebraic formula
.sk 1
                                    ; end this textbox, start another
text 4.0 0.1 7.7 4.5 4 asis
.sk 0.8
.ce
\Fractal\
.hr
modern
~6~fractal~7~ coined in 1975 by Mandelbrot
monsters from early 1900''s
.hr
dilation symmetry
self-similarity or self affinity
.hr
nature simplified
quantified with fractal dimension
.hr
described by constructive procedure
(recursive) algorithm
ideal for computers
                                    ; end this textbox
```

5.32 TICMARKS

```
TICMARKS [exponent placement] [options]
   tic positions (numeric)
   tic labels (text)
```

TICMARKS allows exact placement and format of AXIS tics and labels. The 1st line following TICMARKS contains a series of numeric values in axis units that determine where the tics will be placed. The 2nd line following TICMARKS contains a series of alphanumeric fields corresponding to each of the numeric values that will be used as the corresponding tic label.

`exponent placement`	If a numeric `exponent` and `placement` are provided as the 1st 2 fields, an axis scale factor of the form $\times 10^{\texttt{exponent}}$ will be shown at the specified `placement` along the axis.

The allowed options are

X	Tics and labels refer to the X axis.
Y	Tics and labels refer to the Y axis.
MAJOR	Tics and labels will be drawn full size from NFONT.
MINOR	Tics and labels will be drawn half size from NFONT.
GRID	Tics will be drawn as grid lines.
TOP	x-axis tics and labels added only to top.
BOTTOM	x-axis tics and labels added only to bottom.
LEFT	y-axis tics and labels added only to left.
RIGHT	y-axis tics and labels added only to right.
INVISIBLE	The labels are drawn, but the tics themselves are not.

See Section 5.3 for additional information about tics and labels.

Example

```
xmin 1 xmax 12 xind 0.05  ; setup for months
axis nolabels             ; draw axis without labeling
ticmarks x major          ; use months instead of numbers
1    2    3    4   5   6   7   8   9   10  11  12
Jan  Feb  Mar  Apr May Jun Jul Aug Sep Oct Nov Dec
```

5.33 UGH

UGH `numeric-data` [options] ..text

UGH immediately executes the indicated UGH command, without saving the binary data in the current `.ugh(bin)` file. The UGH command

provides increased control over the graphics screen. See Section 7.1 for details of the UGH commands.

The allowed options are

SAVE	The UGH command will also be included in the current `.ugh(bin)` file.
RESULT	After execution, the contents of the 6 UGH return registers (ax bx cx dx di si) are typed on the console.
WAIT	Waits for a key before returning to **VP**.
SYNONYMS	Defines the synonyms AX, BX, CX, DX, DI, SI as the UGH return register values for later use in other commands.
..text	Any ascii text for the UGH command (begins with "..").

Examples

```
ugh 38 -2 save     ; adds extra space between the characters

ugh 10 3           ; fill the screen with color 3
                   ; VIEW ERASE always uses color 0

ugh 9 result       ; return UGH window number as AX

                   ; change X-windows graphics window title
ugh 41 6 ..new title

ugh 4 synonym      ; get UGH driver characteristics
define nscr &si.   ; find # of graphics screens
if {&nscr.==1} then
  view erase fill wait
else
  view erase fill
endif
```

5.34 VIEW

```
VIEW [filespec] [options]
```

VIEW begins a replay of stored graphic data from the current .ugh(bin) file as set by the SAVE command. If the parameter PBCOLOR > 0, the page boundary limits will be drawn in color PBCOLOR.

The allowed options are as follows:

filespec	If a filespec immediately follows VIEW, this filespec will be displayed instead of the current `.ugh(bin)` file. If necessary, `.ugh(bin)` will be appended.
ERASE	Erases the screen before the VIEW starts.
PAGE	Rests display limits to the current page size before VIEW.
FILL	Resets display limits to approximately fill the screen with the .ugh(bin) file graphics.
BLOWUP	Uses a cursor controlled rubberbox to indicate new display limits within current VIEW.
ROTATE	Rotates the image 90 degrees counterclockwise on screen.
BEST	Equivalent to FILL with optional ROTATE to best fill the screen with the graphics.
`window-num`	If a number is specified along with any other options, the current UGH graphics window is redefined to `window-num`.
SWAP	With single-screen DOS implementations, switch to graphics mode and WAIT for key after the VIEW before returning to text mode.
SWAP `msec`	With single screen DOS implementations, switch to graphics mode and wait `msec` after the VIEW before returning to text mode.
WAIT	Waits for a key stroke (or mouse button) after display. This is performed automatically with single-screen setups when VIEW is called from console. VIEWs from `.exg` files do not generate an automatic WAIT on single-screen setups.
FRACTION `f`	Uses a fraction `f` of the screen with FILL or BEST.
MAGNIFY `f`	Magnifies by a factor `f` centered at graphics cursor.

KEEP	Enters an interactive VIEW mode with the current `.ugh(bin)` file. Single letter commands (B = box magnify, M = magnify, U = unmagnify, F = rescale to full screen, Q=quit KEEP mode), allow detailed examination of the graphics.
NEW	Adjusts FILL/PAGE window size based on graphics X size.
FAST	Set MaxLineWidth=1 before drawing. This draws all lines at 1 pixel width for fastest execution.
SLOW	Set MaxLineWidth=32000 before drawing. This draws varying line widths but is slower.
DEFAULTCOLORS	Restores default colors before display.
MIN `n`	Skip first `n-1` UGH graphics elements.
MAX `n`	Stop after displaying `n` UGH graphics elements.
COMLIM `min max`	Only output UGH commands `minc≤com≤maxc`.

Several UGH commands modify the view with X-Windows implementations:

UGH 41 3	Forces immediate draws to the graphics window. This increases speed, but the graphics cannot be recovered if the window is obscured or resized.
UGH 41 3	Forces graphics into a saved pixmap. This is slower, but allows redisplay after the window is obscured or resized.
UGH 31 `maxlw`	sets new MaxLineWidth for lines.

Examples

```
view erase fill       ; standard command for viewing graphics
                      ; scales to fit everything on screen
view er best pbcol -1 ; allow for rotation, no page boundary

view er fill wait     ; used with DOS single screens

view erase page       ; show graphics relative to page limits

view previous erase fill  ; view previous.ugh(bin)
```

5.35 xyzDATA

```
XDATA [filespec] [options]
        [possible numeric data lines]
YDATA [filespec] [options]
        [possible numeric data lines]
ZDATA [filespec] [options]
        [possible numeric data lines]
```

This indicates that the following lines contain numeric data that will be read into the *DATA array (* = X Y Z). Values will be added until a nonnumeric field is reached. The data may be in free format with different counts of numbers on each line. The data limits *DMIN, *DMAX will be updated to reflect the new values.
The allowed options are

filespec If *DATA is followed by a file specification, the *DATA will be read from the specified file. Comment lines may be included within the data. After the *DATA has been read, any additional lines in the file will be treated as part of a normal .exg file. If the integer parameter SKIP > 0 at the start of the file read, then the first SKIP records will be skipped.

CONTINUE Appends the data to the end of the existing *DATA.

COLUMN n Takes the data from column n of the current TABDATA. This is equivalent to the calculation { Cn }.

ROW n Takes the data from row n of the current TABDATA. This is equivalent to the calculation { Rn }.

INTEGRATE After calculation or reading of the data array, all elements after the first are replaced by their running sum. Thus,
`XDATA 1 2 3 4 INTEGRATE,`
results in XDATA of 1 3 6 10.

DIFF After calculation or reading of the data array, all elements after the first are replaced by their difference relative to the previous. Thus,
`XDATA 1 3 6 10 DIFF,`
results in XDATA of 1 2 3 4. INTEGRATE and DIFF are inverse operations.

{ normal calculations } or [HSILOP calculations]
Uses normal {...} or HSILOP (reverse POLISH) [...] calculations to define the numeric values (see Section 3.2).

Other options store the data as different binary types:

DOUBLE	Saves 8-byte REALs in default hardware format.
FLOAT	Saves 4-byte REALs in default hardware format.
LONG	Saves 4-byte integers (possible byte reversal).
SHORT	Saves 2-byte integers (possible byte reversal).
BYTE	Saves 1-byte integers (0-255).

Examples

```
xdata                   ; simple data from following lines
  1 2 3 4 7 4           ; in free format with
  8  72 26 2.3          ; different numbers of values
  -3.2e4                ; on each line

ydata column 3          ; take YDATA from existing TABDATA
xdata row 1             ; take ZDATA from existing TABDATA

ydata sample.dat        ; read YDATA from a file
xdata index             ; XDATA as corresponding index 1-NMBR

xdata loop 10 { i*i }  ; 1*1 2*2 ... 10*10
```

5.36 XYDATA

```
XYDATA [filespec] [options]
        [possible numeric data lines]
```

XYDATA indicates that the following lines contain X Y data pairs that will be read into XDATA and YDATA. Values will be added until a nonnumeric field is reached. The data may be in free format with different counts of numbers on each line. The data limits XDMIN,XDMAX and YDMIN,YDMAX will be updated.
The allowed options are

filespec If XYDATA is followed by a file specification, the XYDATA will be read from the specified file. Comment lines may be included within the data. After the XYDATA has been read, any additional lines in the file will be treated as part of a normal .exg file. If the integer parameter SKIP > 0 at the start of the file read, then the first SKIP records will be skipped.

CONTINUE	Appends the data to the end of the existing XYDATA.
CURSOR	The graphic cursor specifies the data until a "Q". If RECORDing, the data values will be saved in the RECORD file.
ASIS	Plots the data with the current ISYM and ILINE as they are input. They will not be saved in the XDATA and YDATA arrays.
COLUMN n m	Takes the XDATA from column n of the current TABDATA and YDATA from column m. This is equivalent to the calculation { Cn Cm }.
ROW n m	Takes the XDATA from row n of the current TABDATA and YDATA from row m. This is equivalent to the calculation { Rn Rm }.

{ normal calculations } or [HSILOP calculations]
Uses normal {...} or HSILOP (reverse POLISH) [...] calculations to define the numeric values (see Section 3.2).

Other options store the data as different binary types:

DOUBLE	Saves 8-byte REALs in default hardware format.
FLOAT	Saves 4-byte REALs in default hardware format.
LONG	Saves 4-byte integers (possible byte reversal).
SHORT	Saves 2-byte integers (possible byte reversal).
BYTE	Saves 1-byte integers (0-255).

Examples

```
xydata                   ; simple data from following lines
  1 2  3 4  7 4          ; in free format with
  8  72   26 2.3         ; different numbers of values
  -3.2e4  2.44e5         ; on each line

xydata column 1 2        ; take XYDATA from existing TABDATA

xydata sample.xy         ; read XYDATA from a file

xydata loop 10 { i sqrt(i) } ; calculations
```

5.37 XYERRDATA

```
XYERRDAT [options]
    x y xmin xmax ymin ymax
    (additional lines)
```

Successive lines contain 6 numeric values
`x y xmin xmax ymin ymax`
that define the points to be plotted (x,y) with the current ISYM and ILINE, and the specific error bar limits xmin,xmax, ymin,ymax for each individual point. Error bars will be added around each symbol only if they extend past the symbol size. Each error bar is terminated in a perpendicular line of size SMULT. Identical error bars may be added to all symbols using the XERRMIN, XERRMAX, YERRMIN, and YERRMAX REAL parameters. The X Y points from XYERRDATA become the new XYDATA.

The allowed options are

filespec If XYERRDATA is followed by a file specification, the data will be read from the specified file. 6 numeric values must given on each line. Comment lines may be included within the data. After the XYERRDATA has been read, any additional lines in the file will be treated as part of a normal .exg file. If the integer parameter SKIP > 0 at the start of the file read, then the first SKIP records will be skipped.

{ normal calculations } or [HSILOP calculations]
Uses normal {...} or HSILOP (reverse POLISH) [...] calculations to define the numeric values (see Section 3.2). Each calculation must give 6 numeric values.

The following examples are from demoerrb.exg.

Examples

```
                         ;  error bars, no symbols (just leave space)
xyerrdata color 5 isym 0 connect
   2.0e14 8    1.5e14 2.5e14     7    9
   5.0e14 4    4.1e14 5.0e14     3.9  6.3
   8.0e14 2    7.0e14 9.0e14     1.5  2.5
curve dot color 6     ; add the symbols in a different color

;  calculated error bars based on TABDATA
xyerr iline 0 dot red {c1 c2 c1 c1 c2-(0.12*c1) c2+(0.19*c1)}
```