Chapter 1

KUIP

Command Processor commands.

1.0.1 HELP [item option]

ITEM	C	"Command or menu path"	D='␣'
OPTION	C	"View mode"	D='N'

Possible OPTION values are:

EDIT	The help text is written to a file and the editor is invoked,
E	Same as 'EDIT'.
NOEDIT	The help text is output on the terminal output.
N	Same as 'NOEDIT'

Give the help of a command. If ITEM is a command its full explanation is given: syntax (as given by the command USAGE), functionality, list of parameters with their attributes (prompt, type, default, range, etc.). If ITEM='/' the help for all commands is given.
If HELP is entered without parameters or ITEM is a submenu, the dialogue style is switched to 'AN', guiding the user in traversing the tree command structure.
'HELP -EDIT' (or just 'HELP -E') switches to edit mode: instead of writing the help text to the terminal output, it is written into a temporary file and the pager or editor defined by the command HOST_PAGER is invoked. (On Unix workstations the pager can be defined to display the help text asynchrously in a separated window.) 'HELP -NOEDIT' (or just 'HELP -N') switches back to standard mode. The startup value is system dependent.

1.0.2 USAGE item

ITEM	C	"Command name"

Give the syntax of a command. If ITEM='/' the syntax of all commands is given.

1.0.3 MANUAL item [output option]

ITEM	C	"Command or menu path"	
OUTPUT	C	"Output file name"	D='␣'
OPTION	C	"Text formatting system"	D='␣'

Possible OPTION values are:

'␣'	plain text : plain text format
LATEX	LaTeX format (encapsulated)
TEX	LaTeX format (without header)

Write on a file the text formatted help of a command. If ITEM is a menu path the help for all commands linked to that menu is written. If ITEM='/' the help for the complete command tree is written. If OUTPUT=' ' the text is written to the terminal.
The output file produced with option LATEX can be processed directly by LaTeX, i.e. it contains a standard header defining the meta commands used for formatting the document body. With option TEX only the document body is written into the output file which can be included by a driver file containing customized definitions of the standard meta commands. Example:

```
MANUAL / MAN.TEX LATEX
```

will produce the file MAN.TEX containg the documentation of all available commands in LaTeX format.

1.0.4 EDIT fname

FNAME	C	"File name"

Invoke the editor on the file. The command HOST_EDITOR can be used to define the editor.
If FNAME does not contain an extension the default filetype '.KUMAC' is supplied. The search path defined by the command DEFAULTS is used to find an already existing file. If the file does not exist it is created with the given name.

1.0.5 PRINT fname

`FNAME` C "File name"
Send a file to the printer. The command HOST_PRINT can be used to define the host command for printing the file depending on it file extension.

1.0.6 PSVIEW fname

`FNAME` C "File name"
Invoke the PostScript viewer on the file. The command HOST_PSVIEWER can be used to define the PostScript viewer.
If FNAME does not contain an extension the default filetype '.PS' is supplied.

1.0.7 LAST [n fname]

`N` I "N last commands to be saved" `D=-99 R=-99:`
`FNAME` C "File name" `D='␣'`
Perform various operations with the history file.
If FNAME is not specified, the current history file is assumed by default (the startup history file name is LAST.KUMAC). To change the history file the command LAST 0 NEW-FNAME must be entered.
If N.EQ.-99 (default case) the default host editor is called to edit the current history file, containing all the commands of the session.
If N.LT.0 the last -N commands are printed on the screen. On MVS this allows to edit and resubmit commands. On workstations this allows to resubmit blocks of commands by mouse-driven cut-and-paste operations.
If N.EQ.0 the history file FNAME is rewound and set as the current one (the command LAST 0 FNAME itself is not recorded).
If N.GT.0 the last N commands of the session are saved in the current history file.
See also the command RECORDING.

1.0.8 MESSAGE [string]

`STRING` C "Message string" `D='␣' Separate`
Write a message string on the terminal. A useful command inside a macro. Several message strings can be given in the same command line, each of them separated by one or more spaces (the usual parameter separator); therefore multiple blanks will be dropped and only one will be kept. If multiple blanks should not be dropped, the string must be surrounded by single quotes.

1.0.9 SHELL [cmd]

`CMD` C "Shell command string" `D='␣'`
Execute a command of the host operating system. The command string is passed to the command processor defined by HOST_SHELL. If CMD=' ' the shell is spawned as interactive subprocess. To return from the shell enter 'RETURN' (the full word, not just ⟨CR⟩) or 'exit' (depending on the operation system).

1.0.10 WAIT [string sec]

`STRING` C "Message string" `D='␣'`
`SEC` R "Number of seconds" `D=0 R=0:`
Make a pause (e.g. inside a macro). Wait a given number of seconds (if SEC.GT.0) or just until ⟨CR⟩ is entered (if SEC.EQ.0). A message string is also written on the terminal before waiting.

1.0.11 IDLE sec [string]

`SEC` I "Number of seconds" `R=0:`
`STRING` C "Command string" `D='␣'`
Execute a command if program is idle. The command string is executed if there was no keyboard activity during SEC seconds.

1.0.12 UNITS

List all Input/Output logical units currently open. The files attached to them are also shown.

1.0.13 EXIT

End of the interactive session.

1.0.14 QUIT

End of the interactive session.

1.0.15 FUNCTIONS

```
*** KUIP System Functions ***
```

The function name (and arguments) is literally replaced, at run-time, by its current value. At present, the following functions are available:

```
$DATE ......................  Current date in format DD/MM/YY
$TIME ......................  Current time in format HH.MM.SS
$CPTIME ....................  CP time elapsed since last call (in sec)
$RTIME .....................  Real time elapsed since last call (in sec)
$VDIM(VNAME,IDIM) ..........  Physical length of vector VNAME
                              on dimension IDIM (1..3)
$VLEN(VNAME,IDIM) ..........  As above, but for the logical length
                              (i.e. stripping trailing zeroes)
$NUMVEC ....................  Current number of vectors
$VEXIST(VNAME) .............  Index of vector VNAME
                              (1..$NUMVEC or 0 if VNAME does not exist)
$SUBSTRING(STRING,IX,NCH) ... STRING(IX:IX+NCH-1)
$UPPER(STRING) .............  STRING changed to upper case
$LOWER(STRING) .............  STRING changed to lower case
$LEN(STRING) ...............  Length of STRING
$INDEX(STR1,STR2) ..........  Position of first occurrence of STR2 in
STR1
$WORDS(STRING,SEP) .........  Number of words separated by SEP
$WORD(STRING,K,N,SEP) ......  Extract N words starting at word K
$QUOTE(STRING) .............  Add quotes around STRING
$UNQUOTE(STRING) ...........  Remove quotes around STRING
$EXEC('macro args') ........  EXITM value of EXEC call
$DEFINED('var_name') .......  List of defined macro variables
$EVAL(Expression) ..........  Result of the Expression computed by KUIP
$SIGMA(Expression) .........  Result of the Expression computed by SIGMA
$RSIGMA(Expression) ........  As above but a decimal point is added to
                              integer results
$FORMAT(number,format) ......  Format a number according to a Fortran
                              format string, e.g.
                              $FORMAT(1.5,F5.2) ==> ' 1.50'
                              $FORMAT(123,I5.5) ==> '00123'
$ARGS ......................  Command line at program invocation
$KEYNUM ....................  Address of latest clicked key in style GP
$KEYVAL ....................  Value of latest clicked key in style GP
$LAST ......................  Latest command line executed
$ANUM ......................  Number of aliases
$ANAM(I) ...................  Name of I-th alias
```

`$AVAL(I)`	Value of I-th alias
`$STYLE`	Current style as defined by SET/STYLE
`$OS`	Operating system name, e.g. UNIX or VMS
`$MACHINE`	Hardware or Unix brand, e.g. VAX or HPUX
`$PID`	Process ID
`$IQUEST(I)`	Value of IQUEST(I) status vector
`$ENV(var)`	Value of environment variable
`$FEXIST(file)`	1 if file exists or 0 otherwise
`$SHELL(cmd,N)`	N'th line of shell command output (Unix only)
`$SHELL(cmd,sep)`	Shell output with newlines replaced by sep
`$SHELL(cmd)`	Same as $SHELL(cmd,' ')

`$CALL('fun(args)')`	Call a Fortran REAL FUNCTION
`$ICALL('ifun(args)')`	Call an INTEGER FUNCTION
`$LCALL('lfun(args)')`	Call a LOGICAL FUNCTION and return 0 or 1
`$DCALL('dfun(args)')`	Call a DOUBLE PRECISION FUNCTION
`$HCDIR()`	Current Hbook working directory
`$HEXIST(id)`	1 if histogram ID exists or 0 otherwise
`$HINFO(id,'ENTRIES')`	Number of entries
`$HINFO(id,'MEAN')`	Mean value
`$HINFO(id,'RMS')`	Standard deviation
`$HINFO(id,'EVENTS')`	Number of equivalent events
`$HINFO(id,'OVERFLOW')`	Content of overflow channel
`$HINFO(id,'UNDERFLOW')`	Content of underflow channel
`$HINFO(id,'MIN')`	Minimum bin content
`$HINFO(id,'MAX')`	Maximum bin content
`$HINFO(id,'SUM')`	Total histogram content
`$HINFO(id,'NSLIX')`	Number of X slices
`$HINFO(id,'NSLIY')`	Number of Y slices
`$HINFO(id,'NBANX')`	Number of X bandes
`$HINFO(id,'NBANY')`	Number of Y bandes
`$HINFO(id,'NPROX')`	Projection X (0 or 1)
`$HINFO(id,'NPROY')`	Projection Y (0 or 1)
`$HINFO(id,'XBINS')`	Number of bins in X direction
`$HINFO(id,'XMIN')`	Lower histogram limit in X direction
`$HINFO(id,'XMAX')`	Upper histogram limit in X direction
`$HINFO(id,'YBINS')`	Number of bins in Y direction
`$HINFO(id,'YMIN')`	Lower histogram limit in Y direction
`$HINFO(id,'YMAX')`	Upper histogram limit in Y direction
`$HTITLE(id)`	Histogram title
`$GRAFINFO('XZONES')`	Number of zones in X direction
`$GRAFINFO('YZONES')`	Number of zones in Y direction
`$GRAFINFO('NT')`	Current Normalization Transformation number
`$GRAFINFO('WNXMIN')`	Lower X limit of window in current NT
`$GRAFINFO('WNXMAX')`	Upper X limit of window in current NT
`$GRAFINFO('WNYMIN')`	Lower Y limit of window in current NT
`$GRAFINFO('WNYMAX')`	Upper Y limit of window in current NT
`$GRAFINFO('VPXMIN')`	Lower X limit of viewport in current NT
`$GRAFINFO('VPXMAX')`	Upper X limit of viewport in current NT
`$GRAFINFO('VPYMIN')`	Lower Y limit of viewport in current NT
`$GRAFINFO('VPYMAX')`	Upper Y limit of viewport in current NT
`$GRAFINFO('TXALIH')`	Horizontal text alignment
`$GRAFINFO('TXALIV')`	Vertical text alignment

`$GRAFINFO('TXFONT')`	Text font
`$GRAFINFO('TXPREC')`	Text precision
`$GRAFINFO('?attr')`	HPLOT/HIGZ attribute (see HELP SET for valid names)
`$RGBINFO(icol,'R')`	Weight of Red in color table
`$RGBINFO(icol,'G')`	Weight of Green in color table
`$RGBINFO(icol,'B')`	Weight of Blue in color table
`$CUT(n)`	Cut expression $n
`$CUTEXPAND(string)`	Replace $n in the (quoted) string by $CUT(n)

1.0.16 BUGREPORT [chopt]

`CHOPT` C "Options" `D='B'`

Possible `CHOPT` values are:

- `B` Send a bug report
- `C` Send a comment, suggestion, etc.

Email a bug report or comment to the PAW team. The local editor is invoked with a template to be filled out. After the template has been edited, version information about PAW and the operating system is appended. The user is asked for a confirmation before the report is send.
In Paw++ this command can be accessed via the "Help" menu of the "Executive Window" or the "Main Browser" (menu item "Mail Paw++ Developers").
This command is implemented only on UNIX, VMS and VM systems.

1.0.17 VERSION

Print the version string for PAW and the underlying packages.

Chapter 2

KUIP/ALIAS

Operations with aliases. Aliases are defined to provide shortcut abbreviations for the input line or some part of it. When encountered on an input line an alias is replaced by its string value which can contain further aliases. (Be careful not to define recursive aliases.)

To juxtaposition aliases, a double slash can be used as concatenation sign. Inside quoted strings and for the ALIAS commands themselves the alias substitution is inhibited. Otherwise

```
ALIAS/CREATE ALPHA BETA
ALIAS/CREATE ALPHA BETA
```

whould create an recursive alias BETA and

```
ALIAS/CREATE ALPHA BETA
ALIAS/CREATE BETA GAMMA
ALIAS/DELETE ALPHA
```

would delete the alias name BETA instead of ALPHA itself.

2.0.18 CREATE name value [chopt]

`NAME` C "Alias name"
`VALUE` C "Alias value"
`CHOPT` C "Option" `D='A'`
Possible `CHOPT` values are:

- `A` create an Argument alias
- `C` create a Command alias
- `N` No alias expansion of value

Create an alias NAME which should be substituted by VALUE. An alias name is a sequence of letters and digits starting with a letter. The underscores ('_'), the at-sign ('@') and the dollar-sign ('$') count as letters.
There are two types of aliases: Command aliases are recognized only if they occur in the command position, i.e. as the first token on the line. Argument aliases are recognized anywhere on the command line (except inside quoted strings) if they are surrounded by one of the following separators:

```
blank  /  ,  =  :  .  %  '  (  )
```

Also switch ON the alias translation, i.e. ALIAS/TRANSLATION ON. If CHOPT='C' then the alias is a command alias, i.e. an alias that will only be translated when it is the first token on a command line. Example:

```
Alias/Create GG Graph/Struct/Scratch
Alias/Create FF File1/Name1/Name2
GG FF/ID
```

is equivalent to

```
Graph/Struct/Scratch File1/Name1/Name2/ID
```

```
Alias/Create LS DIR C
```

is equivalent to

```
DIR
```

only when LS is the first token on a command line. In the following case LS will not be translated

```
SHELL LS
```

Aliases occuring inside an value are expanded indepedent whether the value is enclosed by quotes. The option -N allows to suppress this implicit alias expansion.

2.0.19 LIST [name]

`NAME` C "Alias name wildcard" `D='*'`
List all aliases matching the wildcard (names and values).

2.0.20 DELETE name

`NAME` C "Alias name wildcard" `Loop`
Delete the definition of aliases matching the wildcard. NAME='*' deletes all aliases.

2.0.21 TRANSLATION [option]

`OPTION` C "Option" `D='ON'`

Possible `OPTION` values are:

- `?` show current setting
- `ON` switch alias translation ON
- `OFF` switch alias translation OFF

Switch ON/OFF the alias translation. If OFF, alias definitions are not used in parsing the command lines. It is automatically switched ON when an alias is created. If OPTION='?' the current value is shown. The startup value is OFF.

Chapter 3

KUIP/SET_SHOW

Set or show various KUIP parameters and options.

3.0.22 STYLE [option sgylen sgsize sgyspa sgbord wktype]

OPTION	C	"Option" D='?'
SGYLEN	R	"max Y LENgth of each menu item box" D=0.025 R=0.005:0.25
SGSIZE	R	"space available for the application" D=0.8 R=0:0.90
SGYSPA	R	"max Y length of space between menus" D=0.02 R=-0.5:0.50
SGBORD	R	"X or Y border for menus" D=0.015 R=0:0.25
WKTYPE	I	"Graphics workstation type" D=0

Possible OPTION values are:

- `?` show current style
- `C` Command line : select Command line input
- `AN` Menu with Numbers : select general Alpha menu (with Numbers)
- `AL` Menu with Letters : select general Alpha menu (with Letters)
- `G` Graphics menu hardware : select Graphics menu (with hardware character fonts)
- `GW` Graphics menu shadowed : select Graphics menu (with shadowed Width effect)
- `GS` Graphics menu Software : select Graphics menu (with Software character fonts)
- `GP` Panel keys : select Graphics menu (with Panel keys only, i.e. no command tree menu)
- `XM` Motif/X11 : select Motif/X11 interface

Select the user dialogue style (or working mode). The startup value is 'C' (command mode). The current value is returned by the system function $STYLE.
The G-styles are only available if the application program is calling KUWHAG instead of KUWHAT. When one of these options is choosen the remaining parameters control the geometrical layout of the menus on the screen and the graphics workstation type (in case HIGZ was not initialized).
Style 'XM' is only available if the program is calling KUWHAM. In that case switching to other styles is not possible.

3.0.23 PANEL line [gkey]

LINE	R	"Line number" D=0
GKEY	C	"Graphics key value(s)" D='␣'

Set up a panel of graphics keys. The "panel interface" is available in "STYLE GP" and in KUIP/Motif (not in the basic command mode). N.B. in "STYLE GP" only one panel of commands can be set up, whereas in KUIP/Motif there is no limitation.
Examples:

```
PANEL  0                        | reset the panel (in memory)
PANEL  2 A/L QUIT V/L           | initialize line 2 with 3 graphics keys,
                                  respectively A/L, QUIT, V/L
PANEL  2 A/L ' '  V/L ' ' ' '   | initialize line 2 with 5 graphics keys,
                                  and fill 1st and 3rd keys
PANEL  2.04 MESSAGE             | initialize 4th key of 2nd line to MESSAGE
PANEL  2.04                     | clear 4th key of 2nd line
PANEL -2.08                     | initialize line 2 with 8 graphics keys
PANEL -6.16                     | initialize line 6 with 16 graphics keys
```

Note that the key number on the right of the decimal point must always be defined with two digits.
Keys ending with a minus sign make an additional request of keyboard input; the complete command line will be the key text, with a blank at the place of the minus, concatenated with the additional keyboard input. Example:

```
PANEL 1.03 'VEC/PRI-'        | entering VAB will execute VEC/PRI VAB.
```

Keys ending with a double minus sign behave as above but no blank is put at the place of the double minus. Example:

```
PANEL 1.03 'VEC/PRI V--'     | entering AB will execute VEC/PRI VAB
```

The dollar sign inside a key is replaced by additional keyboard input. Example:

```
PANEL 1.03 'VEC/PRI V($)'    | entering 11:20 will execute VEC/PRI V(11:20)
```

In KUIP/Motif there are 2 addditinal commands in order to display or to close one panel:

```
PANEL  0 D [title] [geometry]
PANEL  0 C [title]
```

Examples:

```
- PANEL  0 D 'This is my first panel' 500x300+500+600
```

displays the panel which has been set in memory by the key definition, and sets the title to "This is my first panel", the window size to "500x300" (WxH) and the window position to "500 600" in x and y. If no title and/or no geometry is specified one is given by default.

```
- PANEL  0 C 'This is my first panel'
```

closes (destroys and erases from the screen) the panel whith title "This is my first panel". If no title is specified the last created panel is closed by default.

3.0.24 NEWPANEL line col title width height xpos ypos

`LINE`	I	"Number of lines" `D=5 R=1:30`
`COL`	I	"Number of columns" `D=5 R=1:30`
`TITLE`	C	"Panel Title" `D='NewPanel'`
`WIDTH`	I	"Panel width (in pixels)" `D=300 R=10:`
`HEIGHT`	I	"Panel height (in pixels)" `D=300 R=10:`
`XPOS`	I	"X Position (in pixels)" `D=0 R=0:`
`YPOS`	I	"Y Position (in pixels)" `D=0 R=0:`

Set up a new panel with empty keys (to be filled interactively).

3.0.25 COMMAND [chpath]

`CHPATH` C "Path name for command line" `D='␣'`

Set a filter for the parsing of command lines. If it has been called, it means that whenever a command line is entered, if and only if it is not an existing command (not just ambiguous), it is inserted into the CHPATH string, with $n (n=1..9) being replaced by the n-th token of the command (tokens are separated by spaces), or $* being replaced by the whole command line. Examples:

```
COMMAND 'V/CR $*(10)'
AA                      =>   V/CR AA(10)
BB                      =>   V/CR BB(10)
V/LIST                  =>   V/LIST

COMMAND 'VECTOR/PLOT $1 555 $2'
AA E                    =>   VECTOR/PLOT AA 555 E
BB                      =>   VECTOR/PLOT BB 555

COMMAND                 =>   shows its current value
COMMAND *               =>   reset (equivalent to COMMAND $*)
```

Note that COMMAND and subsequent command lines can be used inside macros, excepted when producing macro statements (like EXEC, IF, GOTO, etc.). For example, the above examples would work also inside macros, while COMMAND 'EXEC $*' or COMMAND 'GOTO $1' will not.

3.0.26 APPLICATION path [cmdex]

`PATH` C "Application name" `D='␣'`
`CMDEX` C "Exit command" `D='EXIT'`

Set the application name. This means that all input lines will be concatenated to the string PATH (until the command specified by the parameter CMDEX is executed, which resets the application to the null string). The value of CMDEX may be specified if the default value EXIT has to be changed (i.e. because already used by the application). APPLICATION can also be inserted in a macro: in this case at least 4 characters must be specified (i.e. APPL).

3.0.27 ROOT [path]

`PATH` C "Root directory" `D='/'`

Set the root for searching commands. If PATH='?' the current root is shown. This allows to access commands regardless of possible ambiguities with different menus. Commands are first searched starting from the current root: if a command is found it is executed. Only if a command is not found a second pass of search is done, starting now from the top root of the command tree (i.e. '/').

3.0.28 TIMING [option]

`OPTION` C "Option" `D='ON'`

Possible `OPTION` values are:

`ON`

`OFF`

`ALL`

Set ON/OFF/ALL the timing of commands. If ON, the real time and the CPU time for the latest executed command (or macro) are presented. If ALL, the time is shown for each command being executed within a macro. The startup value is OFF.

3.0.29 PROMPT prompt

`PROMPT` C "Prompt string" `D='␣'`

Set the prompt string for the command mode dialogue. If PROMPT is blank the current prompt is left unchanged. If PROMPT contains the character sequence '[]' the current command number is inserted between the square brackets.

3.0.30 BREAK [option]

`OPTION` C "Option" `D='ON'`

Possible `OPTION` values are:

`ON`

`OFF`

`TB`

`?`

Set ON/OFF the break handling. If OPTION='?' the current value is shown. The startup value is ON. Hitting the keyboard interrupt (CTRL/C on VMS or CTRL/Q on the Apollo) under break ON condition, the current command or macro execution will be interrupted and the user will get again the application prompt.
BREAK TB switch ON the traceback of the routines called, with their line numbers, when an error occurs. This allows the detection of the routines which provoked the error.

3.0.31 COLUMNS [ncol]

`NCOL` I "Number of columns for terminal output" `D=80 R=-1:`

Set the maximum number of columns for terminal output. If NCOL=0 the current number of columns is shown. If NCOL=-1 the current number of columns is taken from the environment variable COLUMNS. If COLUMNS is undefined the startup value is 80.

3.0.32 RECORDING [nrec]

`NREC` I "Rate for recording on history file" `D=25 R=0:`

Set the recording rate for the history file. Every NREC commands of the session the current history file is updated. If NREC=0 the history is not kept at all (i.e. the file is not written). See also the command LAST.

3.0.33 HOST_EDITOR [editor top left width height dxpad dypad npads]

EDITOR	C	"Host editor command"	D='?'
TOP	I	"Top position of the edit window"	D=20 R=0:
LEFT	I	"Left position of the edit window"	D=20 R=0:
WIDTH	I	"Width of the edit window"	D=0 R=0:
HEIGHT	I	"Height of the edit window"	D=0 R=0:
DXPAD	I	"X offset for help PAD windows"	D=30 R=0:
DYPAD	I	"Y offset for help PAD windows"	D=20 R=0:
NPADS	I	"Maximum number of shifted pads"	D=4 R=1:

Set the host command to invoke the editor. The EDIT command will invoke this editor. If EDITOR='?' the current host editor command is shown.
On Apollo the special value EDITOR='DM' invoke Display Manager pads. The special values EDITOR='WINDOW' and 'PAD' can be used to specify the window positions (in pixel units). 'WINDOW' defines the parameters for edit pads, while 'PAD' defines the parameters for read-only pads (e.g. used by 'HELP -EDIT').
On VMS the special values EDITOR='EDT' and 'TPU' invoke the callable editors. The startup time is considerably lower compared to spawning the editor as a subprocess. The callable EDT has one disadvantage though: after an error, e.g. trying to edit a file in a non-existing directory, subsequent calls will always fail. The TPU call can be augmented by command line options, e.g.

```
HOST_EDITOR TPU/DISP=DECW    | DECwindow interface to EVE
```

On Unix a variety of editors are available, e.g.

```
HOST_EDITOR vi
HOST_EDITOR 'emacs -geometry 80x48'
```

On Unix workstations it is possible to do asynchronous editing via the KUIP edit server, i.e. to start an editor in a separate window while the application can continue to receive commands. In order to do that the following conditions must be fulfilled:

```
- The KUIP edit server 'kuesvr' must be found in the search path.
- The editor command set by HOST_EDITOR must end with an ampersand ('&').
- The environment variable 'DISPLAY' must be set.
```

The ampersand flags your intention to use the edit server if possible. If the edit server cannot be used the ampersand will be ignored, i.e. even with

```
HOST_EDITOR 'vi &'
```

the KUIP/EDIT command will block until the editor terminates if either the 'kuesvr' is not available or 'DISPLAY' is undefined. When using the edit server the editor command is expected to create its own window. 'vi' being a frequent choice, the above command is automatically interpreted as

```
HOST_EDITOR 'xterm -e vi &'
```

The startup value can be defined by the environment variable 'EDITOR'. Otherwise it is set to a system dependent default: 'DM' (Apollo), 'EDT' (VMS), 'XEDIT' (VM/CMS), 'vi' (Unix).

3.0.34 HOST_PAGER [pager]

PAGER C "Host pager command" D='?'

Set the host command to view a file in read-only mode. If OPTION='?' the current host pager command is shown. The 'HELP -EDIT' command will invoke this pager, e.g.

```
HOST_PAGER more
```

On Unix workstations the pager can be asynchronous by creating a separate window, e.g.

```
HOST_PAGER 'xterm -e view &'
HOST_PAGER 'ved &'
```

On Apollo the special value PAGER='DM' defines the use of Display Manager read-only pads. The pad positions can be adjusted by the HOST_EDITOR command.
The startup value can be defined by the environment variables 'KUIPPAGER' or 'PAGER'. If neither of them is defined the value set by the HOST_EDITOR command is used. On VAX/VMS the startup value is 'TYPE/PAGE'.

3.0.35 HOST_PRINTER [command filetype]

COMMAND	C	"Host printer command"	D='?'
FILETYPE	C	"File extension"	D='␣'

Set the host commands for printing files with KUIP/PRINT. The KUIP/PRINT command will use the host command matching the file extension or use the default command defined for FILETYPE=' '.
If COMMAND='?' the currently set commands are shown. If COMMAND=' ' the currently defined command is delete. The command string can contain '$*' and '$-' to indicate the position where the file name with/without file extension should be inserted. For example,

```
MANUAL / refman.tex latex
HOST_PRINTER 'latex $* ; dvips $-' .tex
KUIP/PRINT refman.tex
```

invokes the shell command 'latex refman.tex ; dvips refman'. The predefined defaults are not guaranteed to work since the actual print commands are very much installation dependent.

3.0.36 HOST_PSVIEWER [psviewer]

PSVIEWER C "Host PostScript Viewer command" D='?'

Set the host command to invoke the PostScript Viewer. The PSVIEW command will invoke this PostScript Viewer. If PSVIEWER='?' then the current viewer command is shown.
The startup value can be defined by the environment variables 'KUIPPSVIEWER' or 'PSVIEWER'.
On Unix workstations it is by default set to 'ghostview'. On VAX/VMS the default commands is 'VIEW/FORM=PS/INTERFACE=DECWINDOWS'.

3.0.37 HOST_SHELL [shell]

SHELL C "Host shell command" D='?'

Set the default host shell invoked by the KUIP/SHELL command. If OPTION='?' the current host shell is shown. The startup value is taken from the 'SHELL' environment variable.

3.0.38 RECALL_STYLE [option]

OPTION C "Command recall and editing style" D='?'

Possible OPTION values are:

`?`	show current setting
`KSH`	Korn shell : Emacs like command line editing
`KSHO`	Korn shell + Overwrite : like 'KSH' but overwrite instead of insert mode
`DCL`	VAX/VMS DCL : DCL command line editing
`DCLO`	VAX/VMS DCL + Overwrite : like 'DCL' but overwrite instead of insert mode
`NONE`	disable command line editing

Set the command recall and editing style. If OPTION='?' the current style is shown. The startup value is 'DCL' on VAX/VMS, 'NONE' on Cray and Apollo DM pads, and 'KSH' on other systems.

If the terminal emulator returns ANSI escape sequences (hpterm doesn't!) the up/down arrow keys can be used to recall items from the command history list and the left/right arrow keys to move the cursor.

'KSH' style provides the following control keys for editing:

```
^A/^E   : Move cursor to beginning/end of the line.
^F/^B   : Move cursor forward/backward one character.
^D      : Delete the character under the cursor.
^H, DEL : Delete the character to the left of the cursor.
^K      : Kill from the cursor to the end of line.
^L      : Redraw current line.
^O      : Toggle overwrite/insert mode. Text added in overwrite mode
          (including yanks) overwrites existing text, while insert mode
          does not overwrite.
^P/^N   : Move to previous/next item on history list.
^R/^S   : Perform incremental reverse/forward search for string on
          the history list.  Typing normal characters adds to the
          current search string and searches for a match.  Typing
          ^R/^S marks the start of a new search, and moves on to
          the next match.  Typing ^H or DEL deletes the last
          character from the search string, and searches from the
          starting location of the last search.
          Therefore, repeated DELs appear to unwind to the match
          nearest the point at which the last ^R or ^S was typed.
          If DEL is repeated until the search string is empty the
          search location begins from the start of the history
          list. Typing ESC or any other editing character accepts
          the current match and loads it into the buffer,
          terminating the search.
^T      : Toggle the characters under and to the left of the cursor.
^U      : Kill from the prompt to the end of line.
^Y      : Yank previously killed text back at current location.
          Note that this will overwrite or insert, depending on
          the current mode.
TAB     : By default adds spaces to buffer to get to next TAB stop
          (just after every 8th column).
LF, CR  : Returns current buffer to the program.
```

'DCL' style provides the following control keys for editing:

```
BS/^E   : Move cursor to beginning/end of the line.
^F/^D   : Move cursor forward/backward one character.
```



```
DEL     : Delete the character to the left of the cursor.
^A      : Toggle overwrite/insert mode.
^B      : Move to previous item on history list.
^U      : Delete from the beginning of the line to the cursor.
TAB     : Move to next TAB stop.
LF, CR  : Returns current buffer to the program.
```

3.0.39 VISIBILITY cmd [chopt]

`CMD`	C	"Command name" `D='␣'`
`CHOPT`	C	"?, OFF, ON" `D='?'`

Possible `CHOPT` values are:

`?`

`OFF`

`ON`

Set or show the visibility attributes of a command.
If CHOPT='OFF':

```
- the command it is not executable anymore
- STYLE G draws a shadowed box on the command
- HELP may be still requested on the command
```

The startup value is ON.

3.0.40 DOLLAR [option]

`OPTION`	C	"Substitution of environment variables" `D='?'`

Possible `OPTION` values are:

`?`	show current setting
`ON`	enable substitution
`OFF`	disable substitution

Set or show the status of environment variable substitution.
This command allows to enable/disable the interpretation of environment variables in command lines.
The startup value is 'ON', i.e. "$var" is substituted by the variable value.
Note that the system function "$ENV(var)" allows using environment variables even for 'DOLLAR OFF'
.

3.0.41 FILECASE [option]

`OPTION`	C	"Case conversion for filenames" `D='?'`

Possible `OPTION` values are:

`?`	show current setting
`KEEP`	filenames are kept as entered on the command line
`CONVERT`	filenames are case converted
`RESTORE`	restore previous FILECASE setting

Set or show the case conversion for filenames.
This command has only an effect on Unix systems to select whether filenames are kept as entered on the command line. The startup value is 'CONVERT', i.e. filenames are converted to lowercase.
On other systems filenames are always converted to uppercase.
The 'RESTORE' option set the conversion mode to the value effective before the last FILECASE KEEP/CONVERT command. E.g. the sequence

```
FILECASE KEEP; EDIT Read.Me; FILECASE RESTORE
```

forces case sensitivity for the EDIT command and restores the previous mode afterwards.

3.0.42 LCDIR [directory]

`DIR*ECTORY` C "Directory name" `D='␣'`
Set or show the local working directory.
The current working directory is set to the given path name or the current directory is shown.
To show the current directory used LCDIR without argument. 'LCDIR ' switches to the home directory.
'LCDIR .' switches back to the working directory at the time the program was started.

Chapter 4

MACRO

Macro Processor commands.

4.0.43 EXEC mname [margs]

`MNAME` C "Macro name"
`MARGS` C "Macro arguments" `D='␣' Separate`

Execute the command lines contained in the macro MNAME. As a file can contain several macros, the character '#' is used to select a particular macro inside a file as explained below.
If MNAME does not contain the character '#', the file MNAME.KUMAC is searched and the first macro is executed (it may be an unnamed macro if a MACRO statement is not found as first command line in the file).
If MNAME is of the form FILE#MACRO, the file named FILE.KUMAC is searched and the macro named MACRO is executed.
Examples:

```
EXEC ABC   to exec first (or unnamed) macro of file ABC.KUMAC
EXEC ABC#M to exec macro M of file ABC.KUMAC
```

The command MACRO/DEFAULTS can be used to define a directory search path for macro files.

4.0.44 LIST [mname]

`MNAME` C "Macro name pattern" `D='␣'`

List all macros in the search path defined by MACRO/DEFAULTS. Macros are files with the extension KUMAC. MNAME may be specified to restrict the list to the macros containing such a string in the first part of their name. For example,

```
MACRO/LIST ABC
```

will list only macros starting with ABC.

4.0.45 TRACE [option level]

`OPTION` C "Option" `D='ON'`
`LEVEL` C "Level" `D='␣'`

Possible `OPTION` values are:

`ON`

`OFF`

Possible `LEVEL` values are:

`'␣'`

`TEST`

`WAIT`

`FULL`

`DEBUG`

Set ON/OFF the trace of commands during macro execution. If TRACE='ON' the next command is written on the terminal before being executed. If LEVEL='TEST' the command is only echoed but not executed. If LEVEL='WAIT' the command WAIT is automatically inserted after the execution of each command. The startup values are OPTION='OFF' and LEVEL=' '.

4.0.46 DEFAULTS [path option]

`PATH` C "Search path for macro files" `D='?'`
`OPTION` C "Automatic EXEC" `D='?'`

Possible `OPTION` values are:

`?`	show current setting
`Command`	search for commands only
`C`	same as 'Command'
`Auto`	search for commands before macros
`A`	same as 'Auto'
`AutoReverse`	search for macros before commands
`AR`	same as 'AutoReverse'

Set or show MACRO search attributes.
On Unix and VMS systems PATH defines a comma separated list of directories in which the commands KUIP/EDIT, MACRO/EXEC, and MACRO/LIST search for macro files. For example,

```
MACRO/DEFAULT '.,macro,~/macro'          | Unix
MACRO/DEFAULT '[],[.macro],[macro]'      | VMS
```

defines to search files first in the current directory, then in the subdirectory 'macro' of the current directory, and last the subdirectory 'macro' of the home directory.
On VM/CMS system PATH defines a comma separated list of filemodes. E.g.

```
MACRO/DEFAULT '*'       | search all disks
MACRO/DEFAULT 'A,C'     | search only disks A and C
```

If PATH='?' the currently defined search path is shown. If PATH='.' the search path is undefined, i.e. files are search for in the current directory (A-disk on VM/CMS) only. The startup value is PATH='.'.
The search path is not applied if the file specification already contains an explicit directory path or if it starts with a '-' character (which is stripped off).
OPTION allows to define whether macros can be invoked by their name only without prepending the KUIP/EXEC command:

```
DEFAULT -Command
CMD                     | CMD must be a command
DEFAULT -Auto
CMD                     | if CMD is not a command try EXEC CMD
DEFAULT -AutoReverse
CMD                     | try EXEC CMD first; if not found try command CMD
```

The startup value is 'Command' (also reset by PATH='.').
Important note:
Inside macros the DEFAULT -A (or -AR) logic is disabled, i.e. DEFAULT -C is always assumed.

4.0.47 DATA

Application command to store immediate data into a file. Example:

```
Application DATA vec.dat
1  2  3
4  5  6
7  8  9
vec.dat
vec/read x,y,z vec.dat
```

Chapter 5

MACRO/GLOBAL

Operations on global variables.

5.0.48 CREATE name [value text]

`NAME`	C	"Variable name"	`Loop`
`VALUE`	C	"Initial value"	`D='␣'`
`TEXT`	C	"Comment text"	`D='␣'`

Create a global variable.
If used inside a macro the variable [name] is declared as global.

5.0.49 IMPORT name

`NAME`	C	"Variable name"	`Loop`

Import global variables.
If used inside a macro the variables listed are declared as global. The name may contain '*' as a wildcard matching any sequence of characters.

5.0.50 DELETE name

`NAME`	C	"Variable name"	`Loop`

Delete global variables.
The global variables listed are deleted. The name may contain '*' as a wildcard matching any sequence of characters.

5.0.51 LIST [name file]

`NAME`	C	"Variable name"	`D='*'`
`FILE`	C	"Output file"	`D='␣'`

List global variables.
If a file name is specified the output is the list of GLOBAL/CREATE commands to define the selected global variables. The default file extension is .kumac.

Chapter 6

MACRO/SYNTAX

Explanation of KUIP macro syntax.

A macro is a set of command lines stored in a file, which can be created and modified with any text editor.

In addition to all available KUIP commands the special "macro statements" listed below are valid only inside macros. Note that the statement keywords are fixed. Aliasing such as "ALIAS/CREATE jump GOTO" is not allowed.

Chapter 7

MACRO/SYNTAX/Expressions

Explanation of KUIP expression syntax.

KUIP has a built-in parser for different kinds of expressions: arithmetic expressions, boolean expressions, string expressions, and "garbage expressions".

7.0.52 Arithmetic

Explanation of arithmetic expression syntax.
The syntactic elements for building arithmetic expressions are:

```
expr ::=  number
        | vector-name                (for scalar vectors)
        | vector-name(expr)
        | vector-name(expr,expr)
        | vector-name(expr,expr,expr)
        | [variable-name]            (if value is numeric or
                                      the name of a scalar vector)
        | [variable-name](expr...)   (if value is a vector name)
        | alias-name                 (if value is numeric constant)
        | $system-function(...)
        | - expr
        | expr + expr
        | expr - expr
        | expr * expr
        | expr / expr
        | (expr)
        | ABS(expr)
        | INT(expr)
        | MOD(expr,expr)
```

They can be used in the macro statements DO, FOR, and EXITM, in macro variable assignments, as system function arguments where a numeric value is expected, or as the argument to the $EVAL function. Note that all arithmetic operations are done in floating point, i.e., "5/2" becomes "2.5". If a floating point result appears in a place where an integer is expected, for example as an index, the value is truncated.

7.0.53 Boolean

Explanation of Boolean expression syntax.
Boolean expressions can only be used in the macro statements IF, WHILE, and REPEAT. The possible syntactic elements are shown below.

```
 bool  ::= expr rel-op expr
         | string eq-op string
         | expr eq-op string
         | .NOT. bool
         | bool .AND. bool
         | bool .OR. bool
         | ( bool )

rel-op ::= .LT. | .LE. | .GT. | .GE.
         |  <   |  <=  |  >   |  >=
         | eq-op

eq-op  ::= .EQ. | .NE.
         |  =   | <>
```


7.0.54 String

Explanation of string expression syntax.
String expressions can be used in the macro statements CASE, FOR, and EXITM, in macro variable assignments, as system function arguments where a string value is expected, or as the argument to the $EVAL function. They may be constructed from the syntactic elements shown below.

```
string ::= quoted-string
         | unquoted-string
         | string // string        (concatenation)
         | expr // string          (expr represented as string)
         | [variable-name]
         | alias-name
         | $system-function(...)
```

7.0.55 Garbage

Explanation of "garbage" expression syntax.
Expressions which do not satisfy any of the other syntax rules we want to call "garbage" expressions. For example,

```
s = $OS$MACHINE
```

is not a proper string expression. Unless they appear in a macro statement where specifically only an arithmetic or a boolean expression is allowed, KUIP does not complain about these syntax errors. Instead the following transformations are applied:

```
o  alias substitution

o  macro variable replacement; values containing a
   blank character are implicitly quoted

o  system function calls are replaced one by one with
   their value provided that the argument is a syntactically
   correct expression

o  string concatenation
```

Chapter 8

MACRO/SYNTAX/Variables

Explanation of KUIP macro variables.

Macro variables do not have to be declared. They become defined by an assignment statement,

```
name = expression
```

The right-hand side of the assignment can be an arithmetic expression, a string expression, or a garbage expression (see MACRO/SYNTAX/Expressions). The expression is evaluated and the result is stored as a string (even for arithmetic expressions).

A variable value can be used in other expressions or in command lines by enclosing the name in square brackets, [name]. If the name enclosed in brackets is not a macro variable then no substitution takes place.

8.0.56 Numbered

Accessing macro arguments.
The EXEC command can pass arguments to a macro. The arguments are assigned to the numbered variables [1], [2], etc., in the order given in the EXEC command. The name of the macro, including the file specification, is assigned to [0].
A numbered variable cannot be redefined, i.e., an assignment such as "1 = foo" is illegal. See MACRO/SYNTAX/SHIFT.

8.0.57 Special

Predefined special macro variables.
For each macro the following special variables are always defined:

```
[0]    Fully qualified name of the macro.
[#]    Number of macro arguments
[*]    List of all macro arguments, separated by blanks
[@]    EXITM return code of the last macro called by
       the current one.  The value is "0" if the last
       macro did not supply a return code or no macro
       has been called yet.
```

As for numbered variables these names cannot be used on the left-hand side of an assignment. The values or [#] and [*] are updated by the SHIFT statement.

8.0.58 Indirection

Referencing a macro variable indirectly.
Macro variables can be referenced indirectly. If the variable [name] contains the name of another variable the construct

```
[%name]
```

is substituted by that other variable's value. For example, this is another way to traverse the list of macro arguments:

```
DO i=1,[#]
  arg = [%i]
  ...
ENDDO
```

There is only one level of indirection, i.e., the name contained in "name" may not start with another "%".

8.0.59 Global

Declaring a global variable.

```
EXTERN name ...
```

The variable names listed in the EXTERN statement are declared as global variables. If a name has not been defined with the GLOBAL/CREATE command, it is created implicitly and initialized to the empty string. The name list may contain wildcards, for example

```
EXTERN *
```

makes all defined global variables visible.

8.0.60 READ

Reading a variable value from the keyboard.

```
READ name  [ prompt ]
```

Variable values can be queried from the user during macro execution. The READ statement prompts for the variable value. If name is already defined the present value will be proposed as default.

8.0.61 SHIFT

Manipulation numbered variables.
The only possible manipulation of numbered variables is provided by the SHIFT statement which copies [2] into [1], [3] into [2], etc., and discards the value of the last defined numbered variable. For example, the construct

```
WHILE [1] <> ' ' DO
  arg = [1]
  ...
  SHIFT
ENDDO
```

allows to traverse the list of macro arguments.

Chapter 9

MACRO/SYNTAX/Definitions

Statements for defining macros.

9.0.62 MACRO

Defining a macro.
A .kumac file may contain several macros. An individual macro has the form

```
MACRO macro-name [ parameter-list ]
   statements
RETURN
```

Each statement is either a command line or one of the macro constructs described in this section (MACRO/SYNTAX). For the first macro in the file the MACRO header can be omitted. For the last macro in the file the RETURN trailer may be omitted. Therefore a .kumac file containing only commands (like the LAST.KUMAC) already constitutes a valid macro.

9.0.63 RETURN

Ending a macro definition

```
RETURN [ value ]
```

The RETURN statement flags the end of the macro definition and not the end of macro execution, i.e., the construct

```
IF ... THEN
  RETURN         | error!
ENDIF
```

is illegal. See MACRO/SYNTAX/EXITM.
The value is stored into the variable [@] in the calling macro. If no value is given it defaults to zero.

9.0.64 EXITM

Terminate macro execution and return to calling macro.

```
EXITM [ value ]
```

In order to return from a macro prematurely the EXITM statement must be used. The value is stored into the variable [@] in the calling macro. If no value is given it defaults to zero.

9.0.65 STOPM

Terminate macro execution and return to command line prompt.

```
STOPM
```

The STOPM statement unwinds nested macro calls and returns to the command line prompt.

9.0.66 ENDKUMAC

Ignore rest of KUMAC file.
A logical "end of file" marker. The KUIP parser will not read any part of a .kumac file which appears after the "ENDKUMAC" command.

Chapter 10

MACRO/SYNTAX/Branching

Macro statements for general flow control.

10.0.67 CASE

Select one of many branches.

```
CASE expression IN
(label)  statement  [ statements ]
...
(label)  statement  [ statements ]
ENDCASE
```

The CASE switch evaluates the string expression and compares it one by one against the label lists until the first match is found. If a match is found the statements up to the next label are executed before skipping to the statement following the ENDCASE. None of the statements are executed if there is no match with any label.
Each label is a string constant and the comparison witht the selection expression is case-sensitive. If the same statement sequence should be executed for distinct values a comma-separated list of values can be used.
The "*" character in a label item acts as wildcard matching any string of zero or more characters, i.e., "(*)" constitutes the default label.

10.0.68 GOTO_and_IF_GOTO

Unconditional and conditional branching.

```
GOTO label
```

The simplest form of flow control is provided by the GOTO statement which continues execution at the statement following the target "label:". If the jump leads into the scope of a block statement, for example a DO-loop, the result is undefined.
The target may be given by a variable containing the actual label name.

```
IF expression GOTO label
```

This old-fashioned construct is equivalent to

```
IF expression THEN
   GOTO label
ENDIF
```

10.0.69 IF_THEN

Conditional execution of statement blocks.

```
IF expression THEN
   statements
ELSEIF expression THEN
   statements
...
ELSEIF expression THEN
   statements
ELSE
   statements
ENDIF
```

The general IF construct executes the statements following the first IF/ELSEIF clause for with the boolean expression is true and then continues at the statement following the ENDIF. The ELSEIF clause can be repeated any number of times or can be omitted altogether. If none of the expressions is true, the statements following the optional ELSE clause are executed.

10.0.70 ON_ERROR

Installing an error handler.
Each command returns a status code which should be zero if the operation was successful or non-zero if any kind of error condition occurred. The status code can be tested by $IQUEST(1) system function.

```
ON ERROR GOTO label
```

installs an error handler which tests the status code after each command and branches to the given label when a non-zero value is found. The error handler is local to each macro.

```
ON ERROR EXITM  [ expression ]
```

and

```
ON ERROR STOPM
```

are short-hand notations for a corresponding EXITM or STOPM statement at the targat label.

```
ON ERROR CONTINUE
```

continues execution with the next command independent of the status code. This is the initial setting when entering a macro.

```
OFF ERROR
```

An error handler can be deactivated by this statement.

```
ON ERROR
```

An error handler can be reactivated by this statement.

Chapter 11

MACRO/SYNTAX/Looping

Macro statements for construction loops.

11.0.71 DO

Loop incrementing a loop counter.

```
DO loop = start_expr, finish_expr  [, step_expr ]
   statements
ENDDO
```

The step size (setp_expr) defaults to "1". The arithmetic expressions involved can be floating point values but care must be taken of rounding errors.
Note that "DO i=1,0" results in zero iterations and that the expressions are evaluated only once.

11.0.72 FOR

Loop over items in an expression list.

```
FOR name IN expr_1 [ expr_2 ... expr_n ]
   statements
ENDFOR
```

In a FOR-loop the number of iterations is determined by the number of items in the blank-separated expression list. The expression list must not be empty. One by one each expression evaluated and assigned to the variable name before the statements are executed.
The expressions can be of any type: arithmetic, string, or garbage expressions, and they do not need to be all of the same type. In general each expression is a single list item even if the result contains blanks. The variable [*] is treated as a special case being equivalent to the expression list "[1] [2] ... [n]" which allows yet another construct to traverse the macro arguments:

```
FOR arg IN [*]
   ...
ENDFOR
```

11.0.73 REPEAT

Loop until condition becomes true.

```
REPEAT
   statements
UNTIL expression
```

The body of a REPEAT-loop is executed at least once and iterated until the boolean expression evaluates to true.

11.0.74 WHILE

Loop while condition is true.

```
WHILE expression DO
   statements
ENDWHILE
```

The WHILE-loop is iterated while the boolean expression evaluates to true. The loop body is not executed at all if the boolean expression is false already in the beginning.

11.0.75 BREAKL

Terminate a loop.

```
BREAKL [ level ]
```

Allows to terminate a loop prematurely. The BREAKL continues executing after the end clause of a DO, FOR, WHILE, or REPEAT block, where "level" indicates how many nested constructs to terminate. The default value level=1 terminates the innermost loop construct.

11.0.76 NEXTL

Continue with next loop iteration.

```
NEXTL [ level ]
```

Allows to continue with the next loop iteration without executing the rest of the loop body. Execution continues just before the end clause of a DO, FOR, WHILE, or REPEAT block, where "level" indicates how many nested blocks to skip. The default value level=1 skips to the end of the innermost loop construct.

Chapter 12

VECTOR

Vector Processor commands. Vectors are equivalent to FORTRAN 77 arrays and they use the same notation except when omitting indexes (see last line below). Up to 3 dimensions are supported. Examples:

```
Vec(20) (mono-dimensional with 20 elements)
```

may be addressed by:

```
Vec         for all elements
Vec(13)     for element 13-th
Vec(12:)    for elements 12-th to last
Vec(:10)    for elements first to 10-th
Vec(5:8)    for elements 5-th to 8-th
```

```
Vec(3,100) (2-dimensional with 3 columns by 100 rows):
```

may be addressed by:

```
Vec(2,5:8)   for elements 5-th to 8-th in 2-nd column
Vec(2:3,5:8) for elements 5-th to 8-th in 2-nd to 3-rd columns
Vec(2,5)     for element 5-th in 2-nd column
Vec(:,3)     for all elements in 3-rd row
Vec(2)       for all elements in 2-nd column (SPECIAL CASE)
```

The latest line shows the special (and non-standard with FORTRAN 77) notation such that missing indexes are substituted to the right.

An 'invisible' vector called '?', mono-dimensional and of length 100, is always present. Is is used for communicating between user arrays and KUIP vectors, being equivalenced with the real array VECTOR(100) in the labeled common block /KCWORK/.

12.0.77 CREATE vname [type values]

`VNAME` C "Vector name(length)"
`TYPE` C "Vector type" `D='R'`
`VALUES` C "Value list" `D='␣' Separate Vararg`
Possible `TYPE` values are:
`R`

`I`

Create a vector named VNAME (elements are set to zero). The dimensions are taken from the name, for example VEC(20), VEC(3,100), VEC(2,2,10). Up to 3 dimensions are supported. Dimensions which are not specified are taken to 1, for example VEC(10) —⟩ VEC(10,1,1) and VEC —⟩ VEC(1,1,1). The vector may be of type Real or Integer. A vector is filled at the same time if parameters are given after the TYPE:

```
VEC/CREATE V(10) R 1 2 3 4 5 66 77 88 99 111
VEC/CREATE W(20) R 1 2 3
```

In the last example only the first three elements are filled. Vector elements may be changed later with the command VECTOR/INPUT.
If many equal values have to be entered consecutively, one can specify just one value and precede it by a repetition factor and an asterisk. Example:

```
VEC/CREATE Z(20) R 5*1 2 4*3   --->   VEC/CREATE Z(20) R 1 1 1 1 1 2 3 3 3
3
```

Enter HELP VECTOR for more information on vector addressing.

12.0.78 LIST

List all vectors (name, dimensions, type).

12.0.79 DELETE vlist

`VLIST` C "Vector list" `D='␣' Loop`
Delete from memory all vectors in the list VLIST. The vectors are separated in the list by a comma and embedded blanks are not allowed. An asterisk at the end of VLIST acts as wild-card:

```
VEC/DEL AB*          --->  deletes all vectors starting by AB
VEC/DEL *            --->  deletes all vectors
```

12.0.80 COPY vnam1 vnam2

`VNAM1` C "Source vector name"
`VNAM2` C "Destination vector name"
Copy a vector into another one. Mixed vector type copy is supported (e.g. Integer —⟩ Real and viceversa). If VNAM2 does not exist it is created with the required dimensions, not necessarily the same as the source vector if a sub-range was specified. For example, if A is a 3 x 100 vector and B does not exist, COPY A(2,11:60) B will create B as a 50 elements mono-dimensional vector; a special (and non-standard with FORTRAN 77) notation is used such that, still using the above vectors, COPY A(2,1:100) B and COPY A(2) B have the same effect.
Note that VECTOR/COPY does not allow a range for the destination vector not specifying consecutive elements (i.e. along the first dimension):

```
VEC/COPY V(5)      W(3,4)     | O.K.
VEC/COPY V1(2:3,5) V2(4:5,9)  | O.K.
VEC/COPY V1(5,2:3) V2(4:5,9)  | O.K.
VEC/COPY V1(3,3:4) V2(4,4:5)  | NOT allowed
VEC/COPY V1(2:3,5) V2(2,4:5)  | NOT allowed
```

Enter HELP VECTOR for more information on vector addressing.

12.0.81 INPUT vname [values]

`VNAME` C "Vector name"
`VALUES` C "Value list" `D='␣' Separate Vararg`
Enter values into a vector from the terminal. Example:

```
VEC/INPUT V(6:10) 1.1 2.22 3.333 4.4444 5.55555
```

If many equal values have to be entered consecutively, one can specify just one value and precede it by a repetition factor and an asterisk. Example:

```
VEC/INPUT V 5*1 2 4*3   --->   VEC/INPUT V 1 1 1 1 1 2 3 3 3 3
```

Enter HELP VECTOR for more information on vector addressing.

12.0.82 PRINT vname [dense]

`VNAME` C "Vector name"
`DENSE` I "Output density" `D=1 R=0,1,2`
Write to the terminal the content of a vector. Enter HELP VECTOR for more information on vector addressing.
If DENSE.EQ.0 the output is one vector element per line. If DENSE.EQ.1 the output for a sequence of identical vector elements is compressed to two lines stating the start and end indices. If DENSE.EQ.2 the output for a sequence of identical vector elements is compressed to a single line.

12.0.83 READ vlist fname [format opt match]

`VLIST`	C	"Vector list"
`FNAME`	C	"File name" `D='␣'`
`FORMAT`	C	"Format" `D='␣'`
`OPT`	C	"Options" `D='OC'`
`MATCH`	C	"Matching pattern" `D='␣'`

Possible `OPT` values are:

`OC`

`O`

`'␣'`

`C`

Enter values into vector(s) from a file. A format can be specified, e.g. FORMAT='F10.5,2X,F10.5', or the free format is used if FORMAT is not supplied.
If vector(s) are not existing they will be created of the size as read from the file.
Vectors in the list VLIST are separated by a comma and embedded blanks are not allowed. If subscripts are present in vector names, the smallest one is taken.
OPT is used to select between the following options:

```
'OC'   file is Opened, read and then Closed (default case)
'O'    file is Opened and then read (left open for further reading)
' '    file is read (already open, left so for further reading)
'C'    file is read and then Closed (already open)
```

If the character 'Z' is present in OPT, the vector elements equal to zero after reading are set to the latest non-zero element value (for example reading 1 2 3 0 0 4 0 5 will give 1 2 3 3 3 4 4 5).
MATCH is used to specify a pattern string, restricting the vector filling only to the records in the file which verify the pattern. Example of patterns:

```
 /string/      match a string (starting in column 1)
-/string/      do not match a string (starting in column 1)
 /string/(n)   match a string, starting in column n
 /string/(*)   match a string, starting at any column
```

Enter HELP VECTOR for more information on vector addressing.

12.0.84 WRITE vlist [fname format chopt]

`VLIST`	C	"Vector list"
`FNAME`	C	"File name" `D='␣'`
`FORMAT`	C	"Format" `D='5(1X,G13.7)'`
`CHOPT`	C	"Options" `D='OC'`

Possible `CHOPT` values are:

`OC`

`O`

`'␣'`

`C`

Write to a file the content of vector(s). If FNAME=' ' the content is written to the terminal. A format can be specified, e.g. FORMAT='F10.5,2X,F10.5', or the default one is used if FORMAT is not supplied.
Vectors in the list VLIST are separated by a comma and embedded blanks are not allowed. If subscripts are present in vector names, the smallest one is taken.
CHOPT is used to select between the following options:

```
'OC'   file is Opened, written and then Closed (default case)
'O'    file is Opened and then written (left open for further writing)
' '    file is written (already open, left so for further writing)
'C'    file is written and then Closed (already open)
```

Enter HELP VECTOR for more information on vector addressing.

12.0.85 DRAW vname [id chopt]

`VNAME`	C	"Vector name"
`ID`	C	"Histogram Identifier" `D='12345'`
`CHOPT`	C	"Options" `D='␣'`

Possible `CHOPT` values are:

`'␣'`	Draw an histogram.
`C`	Draw a smooth curve.
`S`	Superimpose plot on top of existing picture.
`+`	Add contents of ID to last plotted histogram.
`B`	Select Bar chart format.
`L`	Connect channels contents by a line.
`P`	Draw the current polymarker at each channel.
`*`	Draw a * at each channel.

Draw vector VNAME interpreting it as a histogram. Optionally save the contents in histogram ID.

12.0.86 HFILL vname id

`VNAME`	C	"Vector name"
`ID`	C	"Histogram Identifier"

Fill the existing histogram ID with vector VNAME. Note that the command VECTOR/PLOT can automatically book, fill and plot the contents of a vector.

12.0.87 PLOT vname [id chopt]

VNAME	C	"Vector name"
ID	C	"Histogram Identifier" D='12345'
CHOPT	C	"Options" D='␣'

Possible CHOPT values are:

- '␣' Draw an histogram.
- C Draw a smooth curve.
- S Superimpose plot on top of existing picture.
- + Add contents of ID to last plotted histogram.
- B Select Bar chart format.
- L Connect channels contents by a line.
- P Draw the current polymarker at each channel.
- * Draw a * at each channel.

Each element of VNAME is used to fill an histogram which is automatically booked with 100 channels and then plotted. If VNAME has the form VNAME1%VNAME2 then a scatter-plot of vector VNAME1 versus VNAME2 is plotted. If ID is given different of 12345, then a 2-Dim histogram is created with 40 bins by 40 bins and filled. One can use the command VECTOR/HFILL to fill an already existing histogram.

12.0.88 FIT x y ey func [chopt np par step pmin pmax errpar]

X	C	"Vector of X coordinates"
Y	C	"Vector of Y coordinates"
EY	C	"Vector of errors on Y" D='?'
FUNC	C	"Function name"
CHOPT	C	"Character options" D='␣'
NP	I	"Number of parameters" D=0 R=0:20
PAR	C	"Vector of parameters"
STEP	C	"Vector of steps size"
PMIN	C	"Vector of lower bounds"
PMAX	C	"Vector of upper bounds"
ERRPAR	C	"Vector of errors on parameters"

Possible CHOPT values are:

- '␣' Do the fit, plot the result and print the parameters.
- 0 Do not plot the result of the fit. By default the fitted function is drawn unless the option 'N' below is specified.
- N Do not store the result of the fit bin by bin with the histogram. By default the function is calculated at the middle of each bin and the fit results stored with the histogram data structure.
- Q Quiet mode. No print
- V Verbose mode. Results after each iteration are printed By default only final results are printed.
- B Some or all parameters are bounded. The vectors STEP,PMIN,PMAX must be specified. Default is: All parameters vary freely.
- L Use Log Likelihood. Default is chisquare method.
- D The user is assumed to compute derivatives analytically using the routine HDERIV. By default, derivatives are computed numerically.
- W Sets weights equal to 1. Default weights taken from the square root of the contents or from HPAKE/HBARX (PUT/ERRORS).
- M The interactive Minuit is invoked.
- E Performs a better Error evaluation (MIGRAD + HESSE + MINOS).
- Z FUNC is the user fitting model

Fit a user defined function to the points defined by the two vectors X and Y and the vector of associated errors EY. See command Histo/Fit for explanation of parameters. Note that if option 'W' is specified or EY='?' (default), the array EY is ignored. Option 'L' is not available.

```
When option 'Z' is given, FUNC is the user fitting model.
FUNC is a subroutine with the calling sequence:
   Subroutine FUNC(N,X,Y,EY,NPAR,IFLAG,NPFITS)
where
   - X(N),Y(N),EY(N) are the input vectors,
   - NPAR the number of parameters
   - NPFITS is an output parameter = Number of points used in the fit
The user must declare the common/HCFITD/FITPAD(24),FITFUN in FUNC
```

Chapter 13

VECTOR/OPERATIONS

Simple arithmetic operations between vectors. In all the operations only the minimum vector length is considered, i.e. an operation between a vector A of dimension 10 and a vector B of dimension 5 will involve the first 5 elements in both vectors. If the destination vector does not exist, it is created with the same length as the source vector.

13.0.89 VBIAS vnam1 bias vnam2

`VNAM1`	C	"Source vector name"
`BIAS`	R	"Bias value"
`VNAM2`	C	"Destination vector name"

VNAM2(I) = BIAS + VNAM1(I)

13.0.90 VSCALE vnam1 scale vnam2

`VNAM1`	C	"Source vector name"
`SCALE`	R	"Scale factor"
`VNAM2`	C	"Destination vector name"

VNAM2(I) = SCALE * VNAM1(I)

13.0.91 VADD vnam1 vnam2 vnam3

`VNAM1`	C	"First source vector name"
`VNAM2`	C	"Second source vector name"
`VNAM3`	C	"Destination vector name"

VNAM3(I) = VNAM1(I) + VNAM2(I)

13.0.92 VMULTIPLY vnam1 vnam2 vnam3

`VNAM1`	C	"First source vector name"
`VNAM2`	C	"Second source vector name"
`VNAM3`	C	"Destination vector name"

VNAM3(I) = VNAM1(I) * VNAM2(I)

13.0.93 VSUBTRACT vnam1 vnam2 vnam3

`VNAM1`	C	"First source vector name"
`VNAM2`	C	"Second source vector name"
`VNAM3`	C	"Destination vector name"

VNAM3(I) = VNAM1(I) - VNAM2(I)

13.0.94 VDIVIDE vnam1 vnam2 vnam3

`VNAM1`	C	"First source vector name"
`VNAM2`	C	"Second source vector name"
`VNAM3`	C	"Destination vector name"

VNAM3(I) = VNAM1(I) / VNAM2(I) (or 0 if VNAM2(I)=0)

Chapter 14

HISTOGRAM

Manipulation of histograms, Ntuples. Interface to the HBOOK package.

14.0.95 FILE lun fname [lrecl chopt]

`LUN`	I	"Logical unit number" `R=0:128`
`FNAME`	C	"File name"
`LRECL`	I	"Record length in words" `D=1024`
`CHOPT`	C	"Options" `D='␣'`

Possible `CHOPT` values are:

- `'␣'` Existing file is opened (read mode only).
- `N` A new file is opened.
- `U` Existing file is opened to be modified.
- `D` Reset lock.

Open an HBOOK direct access file. If LUN is 0 the next free logical unit will be used. If LRECL is 0 the system will determine the correct record length of an existing file.

14.0.96 LIST [chopt]

`CHOPT`	C	"Options" `D='␣'`

Possible `CHOPT` values are:

- `'␣'` List histograms and Ntuples in the current directory.
- `I` A verbose format is used (HINDEX), (only for //PAWC).
- `S` List with histograms sorted by increasing IDs.

List histograms and Ntuples in the current directory.

14.0.97 DELETE id

`ID`	C	"Histogram Identifier" `Loop`

Delete histogram/Ntuple ID in Current Directory (memory). If ID=0 delete all histograms and Ntuples. To delete histograms in disk files use command HIO/HSCRATCH.

14.0.98 PLOT [id chopt]

`ID`	C	"Histogram Identifier" `Loop Minus`
`CHOPT`	C	"Options" `D='␣' Minus`

Possible `CHOPT` values are:

`'␣'`	Draw the histogram.
`C`	Draw a smooth curve.
`S`	Superimpose plot on top of existing picture.
`+`	Add contents of ID to last plotted histogram.
`-`	Substract contents of ID to last plotted histogram.
`+-`	Draw the delta with the last plotted histogram.
`B`	Select Bar chart format.
`L`	Connect channels contents by a line.
`P`	Draw the current polymarker at each channel or cell.
`*`	Draw a * at each channel.
`K`	Must be given if option 'U' is given later.
`U`	Update channels modified since last call.
`E`	Draw error bars and current marker.
`E0`	Draw error bars without symbols clipping.
`E1`	Draw small lines at the end of the error bars.
`E2`	Draw error rectangles.
`E3`	Draw a filled area through the end points of the vertical error bars.
`E4`	Draw a smoothed filled area through the end points of the vertical error bars.
`A`	Axis labels and tick marks are not drawn.
`BOX`	Draw 2-Dim with proportional boxes.
`COL`	Draw 2-Dim with a color table.
`Z`	Used with COL or SURF, it draws the color map.
`SURF`	Draw as a surface plot (angles are set via the command angle).
`SURF1`	Draw as a surface with color levels
`SURF2`	Same as SURF1 but without cell lines.
`SURF3`	Same as SURF but with the contour plot (in color) on top.
`SURF4`	Draw as a surface with Gouraud shading.
`LEGO`	Draw as a lego plot (angles are set via the command angle).
`LEGO1`	Draw lego plot with light simulation.
`LEGO2`	Draw lego plot with color levels.
`BB`	Suppress the Back Box on 3D plots.
`FB`	Suppress the Front Box on 3D plots.
`CONT`	Draw 2-Dim as a contour plot (15 levels).
`TEXT`	Draw 2-Dim as a table.
`CHAR`	Draw 2-Dim with characters (a la HBOOK).
`HIST`	Draw only histogram (no errors or associated function).
`FUNC`	Draw only the associated function (not the histogram).
`CYL`	Cylindrical coordinates for 3D plots.
`POL`	Polar coordinates for 3D plots.
`SPH`	Spherical coordinates for 3D plots.
`PSD`	Pseudo-rapidity/phi coordinates for 3D plots.

Plot a single histogram or a 2-Dim projection. If ID=0 or ID=* all the histograms in the current directory

14.0.99 ZOOM [id chopt icmin icmax]

`ID`	C	"Histogram Identifier" `Loop Minus`
`CHOPT`	C	"Options" `D='␣'`
`ICMIN`	I	"First channel" `D=1`
`ICMAX`	I	"Last channel" `D=9999`

Possible `CHOPT` values are:

`'␣'`	Plot the zoomed histogram.
`C`	Draw a smooth curve.
`S`	Superimpose plot on top of existing picture.
`+`	Add contents of ID to last plotted histogram.
`B`	Select Bar chart format.
`L`	Connect channels contents by a line.
`P`	Draw the current polymarker at each channel.
`*`	Draw a * at each channel.

Plot a single histogram between channels ICMIN and ICMAX. Each plotted histogram will start either a new picture or a new zone in the current picture. If no parameters are given to the command, then the system waits for two points using the graphics cursor. To quit ZOOM, click the right button of the mouse or CRTL/E.

14.0.100 MANY_PLOTS idlist

`IDLIST`	C	"List of histogram Identifiers" `Vararg`

Plot one or several histograms into the same plot. Plotted histograms are superimposed on the same zone of the picture.

14.0.101 PROJECT id

`ID`	C	"Histogram Identifier" `Loop`

Fill all booked projections of a 2-Dim histogram. Filling is done using the 2-D contents of ID.

14.0.102 COPY id1 id2 [title]

`ID1`	C	"First histogram Identifier"
`ID2`	C	"Second histogram Identifier" `Loop`
`TITLE`	C	"New title" `D='␣'`

Copy a histogram (not Ntuple) onto another one. Bin definition, contents, errors, etc. are preserved. If TITLE is not given, ID2 has the same title as ID1.

14.0.103 FIT id func [chopt np par step pmin pmax errpar]

`ID`	C	"Histogram Identifier"
`FUNC`	C	"Function name" `D='␣'`
`CHOPT`	C	"Options" `D='␣'`
`NP`	I	"Number of parameters" `D=0 R=0:34`
`PAR`	C	"Vector of parameters"
`STEP`	C	"Vector of steps size"
`PMIN`	C	"Vector of lower bounds"
`PMAX`	C	"Vector of upper bounds"
`ERRPAR`	C	"Vector of errors on parameters"

Possible `CHOPT` values are:

'␣' Do the fit, plot the result and print the parameters.

- `O` Do not plot the result of the fit. By default the fitted function is drawn unless the option 'N' below is specified.
- `N` Do not store the result of the fit bin by bin with the histogram. By default the function is calculated at the middle of each bin and the fit results stored with the histogram data structure.
- `Q` Quiet mode. No print
- `V` Verbose mode. Results after each iteration are printed By default only final results are printed.
- `B` Some or all parameters are bounded. The vectors STEP,PMIN,PMAX must be specified. Default is: All parameters vary freely.
- `L` Use Log Likelihood. Default is chisquare method.
- `D` The user is assumed to compute derivatives analytically using the routine HDERIV. By default, derivatives are computed numerically.
- `W` Sets weights equal to 1. Default weights taken from the square root of the contents or from HPAKE/HBARX (PUT/ERRORS). If the L option is given (Log Likelihood), bins with errors=0 are excluded of the fit.
- `M` The interactive Minuit is invoked. (see Application HMINUIT below).
- `E` Performs a better Error evaluation (MIGRAD + HESSE + MINOS).
- `U` User function value is taken from /HCFITD/FITPAD(24),FITFUN.
- `K` Keep the settings of Application HMINUIT for a subsequent command.

Fit a user defined (and parameter dependent) function to a histogram ID (1-Dim or 2-Dim) in the specified range. FUNC may be:

```
A- The name of a file which contains the user defined
   function to be minimized. Function name and file name
   must be the same. For example file FUNC.FOR is:
     FUNCTION FUNC(X)   or FUNC(X,Y) for a 2-Dim histogram
     COMMON/PAWPAR/PAR(2)
     FUNC=PAR(1)*X +PAR(2)*EXP(-X)
     END
    Ex: His/fit 10 func.for ! 5 par

   When the option U is given, the file FUNC.FOR should look like:
     FUNCTION FUNC(X)   or FUNC(X,Y) for a 2-Dim histogram
     DOUBLE PRECISION FITPAD(24),FITFUN
     COMMON/HCFITD/FITPAD,FITFUN
     FITFUN=FITPAD(1)*X +FITPAD(2)*EXP(-X)
     FUNC=FITFUN
     END

B- One of the following keywords (1-Dim only):
   G : to fit Func=par(1)*exp(-0.5*((x-par(2))/par(3))**2)
   E : to fit Func=exp(par(1)+par(2)*x)
   Pn: to fit Func=par(1)+par(2)*x+par(3)*x**2......+par(n+1)*x**n
    Ex: His/fit 10 g

C- A combination of the keywords in B with the 2 operators + or *.
```



```
Ex: His/Fit 10 p4+g ! 8 par
    His/Fit 10 p2*g+g ! 9 par
  Note that in this case, the order of parameters in PAR must
  correspond to the order of the basic functions.
  For example, in the first case above, par(1:5) apply to
  the polynomial of degree 4 and par(6:8) to the gaussian while
  in the second case par(1:3) apply to the polynomial of degree 2,
  par(4:6) to the first gaussian and par(7:9) to the second gaussian.
  Blanks are not allowed in the expression.
```

For cases A and C, before the execution of this command, the vector PAR must be filled (via Vector/Input) with the initial values. For case B, if NP is set to 0, then the initial values of PAR will be calculated automatically. After the fit, the vector PAR contains the new values of parameters. If the vector ERRPAR is given, it will contain the errors on the fitted parameters. A bin range may be specified with ID.

```
Ex. Histo/Fit 10(25:56).
```

When the Histo/it command is used in a macro, it might be convenient to specify MINUIT directives in the macro itself via the Application HMINUIT as described in this example:

```
Macro fit
Application HMINUIT exit
name 1 par_name1
name 2 par_name2
migrad
improve
exit
Histo/fit id fitfun.f M
Return
```

Chapter 15

HISTOGRAM/2D_PLOT

Plotting of 2-Dim histograms in various formats.

15.0.104 LEGO [id theta phi chopt]

ID	C	"Histogram Identifier"	Loop
THETA	R	"Angle THETA in degrees"	D=30.
PHI	R	"Angle PHI in degrees"	D=30.
CHOPT	C	"Options"	D='␣'

Possible CHOPT values are:

- '␣' Hidden line algorithm is used.
- 1 Hidden surface algorithm is used. The colour of the lego is given by SET HCOL CI where CI is a colour index. For the top and the sides of the lego the same hue is used but with a different light.
- 2 Hidden surface algorithm is used. The colour of each bar changes according to the value of Z. It is possible to change the set of colours used with SET HCOL c.L where L define a palette of colours given by the command ATT/PALETTE.

Draw a lego plot from 2-Dim or 1-Dim histograms. It is also possible to produce stacked lego plots. A stacked lego plot consists of a superimposition of several histograms, whose identifiers are given in the command LEGO separated by the character "+".

```
PAW > LEGO ID1+ID2+ID3    | Maximum number of ID's is 10. The colours of
                          | each IDn is given by the command ATT/PALETTE
```

Examples:

```
PAW > SET HCOL 2          | The colour the histogram is 2 (red)
PAW > LEGO 20             | Display a lego with lines
PAW > LEGO 20 ! ! 1       | Display a lego with different lights
PAW > LEGO 20 ! ! 2       | Display a lego with colours
PAW > PALETTE  1 3 2 3 4  | Create the palette number 1 with 3
                          | elements: 2,3
PAW > SET HCOL 0.1        | The subsequent stack lego plots will use list 1
PAW > LEGO 10+20+30       | Plot a stack of lego plots with lines
PAW > LEGO 10+20+30 ! ! 1 | Plot a stack of lego plots with light
```

Notes: - The commands OPTION BAR, SET BARW and SET BARO act on lego plots

```
- The options 1 and 2 must be used only on selective erase
  devices.
```

15.0.105 SURFACE [id theta phi chopt]

ID	C	"Histogram Identifier" `Loop`
THETA	R	"Angle THETA in degrees" `D=30.`
PHI	R	"Angle PHI in degrees" `D=30.`
CHOPT	C	"Options" `D='␣'`

Possible CHOPT values are:

- '␣' Hidden line algorithm is used.
- 1 Hidden surface algorithm is used and each cell is filled with a colour corresponding to the Z value (or grey scale with PostScript). It is possible to change the set of colours used with SET HCOL ic.L where L define a palette of colours given by the command ATT/PALETTE.
- 2 Similar to option '1' except that the cell lines are not drawn. This is very useful to draw contour plots with colours if THETA=90 and PHI=0.
- 3 Surface is drawn with a contour plot in color on top. The contour plot is drawn with the colors defined with the command PALETTE.
- 4 Surface is drawn with Gouraud shading.

Draw a surface plot from 2-Dim or 1-Dim histograms. With this command it is possible to draw color contour plots:

```
PAW > ATT/PAL 1 3 2 3 4   | Define the palette 1 with 3 elements
PAW > SET HCOL 0.1        | Set the list 1 as colours for histograms
PAW > SET NDVZ 4          | Set the number of Z divisions to 4
PAW > SURF id 90 0 2      | Draw the contour
```

Note: - The options 1 to 4 must be used only on selective erase devices.

15.0.106 CONTOUR [id nlevel chopt param]

ID	C	"Histogram Identifier" `Loop`
NLEVEL	I	"Number of contour lines" `D=10`
CHOPT	C	"Options" `D='1'`
PARAM	C	"Vector of contour levels"

Possible CHOPT values are:

- 0 Use colour to distinguish contours.
- 1 Use line style to distinguish contours.
- 2 Line style and colour are the same for all contours.
- 3 The contour is drawn with filled colour levels. The levels are equidistant. The color indices are taken in the current palette (defined with the command PALETTE). If the number of levels (NLEVEL) is greater than the number of entries in the current palette, the palette is explore again from the beginning in order to reach NLEVEL.
- S Superimpose plot on top of existing picture.

Draw a contour plot from a 2-Dim histogram. If PARAM is not given, contour levels are equidistant. If given, the vector PARAM may contain up to 50 values.

Chapter 16

HISTOGRAM/CREATE

Creation ("booking") of HBOOK objects in memory.

16.0.107 1DHISTO id title ncx xmin xmax [valmax]

`ID`	C	"Histogram Identifier" `Loop`
`TITLE`	C	"Histogram title" `D='␣'`
`NCX`	I	"Number of channels" `D=100`
`XMIN`	R	"Low edge" `D=0.`
`XMAX`	R	"Upper edge" `D=100.`
`VALMAX`	R	"Maximum bin content" `D=0.`

Create a one dimensional histogram. The contents are set to zero. If VALMAX=0, then a full word is allocated per channel, else VALMAX is used as the maximum bin content allowing several channels to be stored into the same machine word.

16.0.108 PROFILE id title ncx xmin xmax ymin ymax [chopt]

`ID`	C	"Histogram Identifier"
`TITLE`	C	"Histogram title" `D='␣'`
`NCX`	I	"Number of channels" `D=100`
`XMIN`	R	"Low edge in X" `D=0.`
`XMAX`	R	"Upper edge in X" `D=100.`
`YMIN`	R	"Low edge in Y" `D=-1.E30`
`YMAX`	R	"Upper edge in Y" `D=1.E30`
`CHOPT`	C	"Options" `D='␣'`

Possible `CHOPT` values are:

`'␣'` Error on mean

`S` Spread option

Create a profile histogram. Profile histograms accumulate statistical quantities of a variable y in bins of a variable x. The contents are set to zero.

16.0.109 BINS id title ncx xbins [valmax]

`ID`	C	"Histogram Identifier"
`TITLE`	C	"Histogram title" `D='␣'`
`NCX`	I	"Number of channels" `D=100`
`XBINS`	C	"Vector of NCX+1 low-edges"
`VALMAX`	R	"Maximum bin content" `D=0.`

Create a histogram with variable size bins. The low-edge of each bin is given in vector XBINS (NCX+1) values. The contents are set to zero. See 1DHISTO for VALMAX.

16.0.110 2DHISTO id title ncx xmin xmax ncy ymin ymax [valmax]

`ID`	C	"Histogram Identifier" `Loop`
`TITLE`	C	"Histogram title" `D='␣'`
`NCX`	I	"Number of channels in X" `D=40`
`XMIN`	R	"Low edge in X" `D=0.`
`XMAX`	R	"Upper edge in X" `D=40.`
`NCY`	I	"Number of channels in Y" `D=40`
`YMIN`	R	"Low edge in Y" `D=0.`
`YMAX`	R	"Upper edge in Y" `D=40.`
`VALMAX`	R	"Maximum bin content" `D=0.`

Create a two dimensional histogram. The contents are set to zero. See 1DHISTO for VALMAX.

16.0.111 PROX id

`ID` C "Histogram (2-Dim) Identifier" `Loop`

Create the projection onto the x axis. The projection is not filled until the Histo/Project command is executed.

```
To plot projection X of ID type:
  PAW > HI/PLOT ID.PROX
```


16.0.112 PROY id

`ID` C "Histogram (2-Dim) Identifier" `Loop`

Create the projection onto the y axis. The projection may be filled with Histo/Project.

```
To plot projection Y of ID type:
  PAW > HI/PLOT ID.PROY
```

16.0.113 SLIX id nslices

`ID`	C	"Histogram (2-Dim) Identifier" `Loop`
`NSLICES`	I	"Number of slices"

Create projections onto the x axis, in y-slices. The projection may be filled with Histo/Project.

```
To plot slice 3 in X of ID type:
  PAW > HI/PLOT ID.SLIX.3
```

16.0.114 SLIY id nslices

`ID`	C	"Histogram (2-Dim) Identifier" `Loop`
`NSLICES`	I	"Number of slices"

Create projections onto the y axis, in x-slices. The projection may be filled with Histo/Project.

```
To plot slice 2 in Y of ID type:
  PAW > HI/PLOT ID.SLIY.2
```

16.0.115 BANX id ymin ymax

`ID`	C	"Histogram (2-Dim) Identifier" `Loop`
`YMIN`	R	"Low edge in Y"
`YMAX`	R	"Upper edge in Y"

Create a projection onto the x axis, in a band of y. Several bands can be defined on the one histogram. The projection may be filled with Histo/Project.

```
To plot band 1 in X of ID type:
  PAW > HI/PLOT ID.BANX.1
```

16.0.116 BANY id xmin xmax

`ID`	C	"Histogram (2-Dim) Identifier" `Loop`
`XMIN`	R	"Low edge in X"
`XMAX`	R	"Upper edge in X"

Create a projection onto the y axis, in a band of x. Several bands can be defined on the one histogram. The projection may be filled with Histo/Project.

```
To plot band 1 in Y of ID type:
  PAW > HI/PLOT ID.BANY.1
```

16.0.117 TITLE_GLOBAL [chtitl chopt]

`CHTITL`	C	"Global title" `D='␣'`
`CHOPT`	C	"Options" `D='␣'`

Possible `CHOPT` values are:

`'␣'` The global title is plotted at the top of each picture.

`U` If the option 'UTIT' is on, a user title is plotted at the bottom of each histogram.

Set the global title. The size and the Y position of the global title may be changed by the commands SET GSIZ and SET YGTI respectively. The size and the Y position of the user title may be changed by the commands SET TSIZ and SET YHTI respectively.

Chapter 17

HISTOGRAM/HIO

Input/Output operations of histograms.

17.0.118 HRIN id [icycle iofset]

`ID`	C	"Histogram Identifier"	`Loop`
`ICYCLE`	I	"Cycle number"	`D=999`
`IOFSET`	I	"Offset"	`D=0`

Read histogram/Ntuple ID from the current directory on direct access file to memory. An identical histogram is created but with an ID equal to that of the original histogram plus the offset IOFSET. Identifier may be '0' or '*' (for all histograms). If ICYCLE ⟩ 1000 and ID=0 read all histograms in all subdirectories as well. If IOFSET = 99999 then the contents of histogram ID on the disk file are added to the current histogram in memory if it exists. For example to add all histograms from FILE1 and FILE2 in memory, the sequence of commands can be:

```
PAW > Histo/File 1 FILE1
PAW > Hrin 0
PAW > Histo/File 2 FILE2
PAW > Hrin 0 ! 99999
```

17.0.119 HROUT id [chopt]

`ID`	C	"Histogram Identifier"	`Loop`
`CHOPT`	C	"Options"	`D='␣'`

Possible `CHOPT` values are:

`'␣'`	Write histo/Ntuple ID from memory to current directory.
`T`	Writes all histograms in subdirectories as well.

Write histo/Ntuple ID from memory to current directory. Identifier may be '0' or '*' (for all histograms).

17.0.120 HSCRATCH id

`ID`	C	"Histogram Identifier"	`Loop`

Delete histogram ID in Current Directory on disk. If ID='0' or '*' delete all histograms. To delete histograms in memory use command HISTO/DELETE.

17.0.121 HFETCH id fname

`ID`	C	"Histogram Identifier"
`FNAME`	C	"File name"

Fetch histogram ID from file FNAME. FNAME has been created by the old version of HBOOK3 (Unformatted).

17.0.122 HREAD id fname

`ID`	C	"Histogram Identifier"
`FNAME`	C	"File name"

Read histogram ID from file FNAME. FNAME has been created by the old version of HBOOK3 (Formatted).

17.0.123 PRINT id [chopt]

`ID`	C	"Histogram Identifier"	`Loop`
`CHOPT`	C	"Options"	`D='␣'`

Possible `CHOPT` values are:

`'␣'`	Print histograms.
`S`	Only statistics (Number of entries, mean, RMS, underflow, overflow) are printed.

Print histograms (line-printer format) on screen. The command OUTPUT_LP may be used to change the output file.

17.0.124 DUMP id

`ID`	C	"Histogram Identifier"	`Loop`

Dump the histogram ZEBRA data structure on the terminal.

17.0.125 OUTPUT_LP [lun fname]

`LUN` I "Logical unit number" `D=6`
`FNAME` C "File name" `D='␣'`
Change the HBOOK "line printer" file name. If FNAME=' ' then OUTPUT is appended to an already opened file on unit LUN. If LUN is negative, the file is closed and subsequent output is directed to unit 6.

17.0.126 GLOBAL_SECT gname

`GNAME` C "Global section name" `D='␣'`
Map the global section GNAME (VAX only). The current directory is changed to //GNAME.

17.0.127 GRESET id

`ID` C "Histogram Identifier"
Reset histogram ID in the global section.

Chapter 18

HISTOGRAM/OPERATIONS

Histogram operations and comparisons.

18.0.128 ADD id1 id2 id3 [c1 c2 option]

ID1	C	"First histogram Identifier"
ID2	C	"Second histogram Identifier"
ID3	C	"Result histogram Identifier"
C1	R	"Scale factor for ID1" D=1.
C2	R	"Scale factor for ID2" D=1.
OPTION	C	"Option" D='␣'

Possible OPTION values are:

'␣'

E

Add histograms: ID3 = C1*ID1 + C2*ID2. Applicable to 1-Dim and 2-Dim histograms. See command HRIN to add histograms with same IDS from different files. If option 'E' is set, error bars are calculated for ID3.

18.0.129 SUBTRACT id1 id2 id3 [c1 c2 option]

ID1	C	"First histogram Identifier"
ID2	C	"Second histogram Identifier"
ID3	C	"Result histogram Identifier"
C1	R	"Scale factor for ID1" D=1.
C2	R	"Scale factor for ID2" D=1.
OPTION	C	"Option" D='␣'

Possible OPTION values are:

'␣'

E

Subtract histograms: ID3 = C1*ID1 - C2*ID2. Applicable to 1-Dim and 2-Dim histograms. If option 'E' is set, error bars are calculated for ID3.

18.0.130 MULTIPLY id1 id2 id3 [c1 c2 option]

ID1	C	"First histogram Identifier"
ID2	C	"Second histogram Identifier"
ID3	C	"Result histogram Identifier"
C1	R	"Scale factor for ID1" D=1.
C2	R	"Scale factor for ID2" D=1.
OPTION	C	"Option" D='␣'

Possible OPTION values are:

'␣'

E

Multiply histogram contents: ID3 = C1*ID1 * C2*ID2. Applicable to 1-Dim and 2-Dim histograms. If option 'E' is set, error bars are calculated for ID3.

18.0.131 DIVIDE id1 id2 id3 [c1 c2 option]

ID1	C	"First histogram Identifier"
ID2	C	"Second histogram Identifier"
ID3	C	"Result histogram Identifier"
C1	R	"Scale factor for ID1" D=1.
C2	R	"Scale factor for ID2" D=1.
OPTION	C	"Option" D='␣'

Possible OPTION values are:

'␣'

E

Divide histograms: ID3 = C1*ID1 / C2*ID2. Applicable to 1-Dim and 2-Dim histograms. If option 'E' is set, error bars are calculated for ID3.

18.0.132 RESET id [title]

ID	C	"Histogram Identifier" Loop
TITLE	C	"New title" D='␣'

Reset contents and errors of an histogram. Bin definition is not modified.

18.0.133 DIFF id1 id2 [chopt]

ID1	C	"First Histogram Identifier"
ID2	C	"Second Histogram Identifier"
CHOPT	C	"Options" D='D'

Possible CHOPT values are:

- '␣' The comparison is done only on the shape of the two histograms.
- N Include also comparison of the relative normalization of the two histograms, in addition to comparing the shapes. PROB is then a combined confidence level taking account of absolute contents.
- D Debug printout, produces a blank line and two lines of information at each call, including the ID numbers, the number of events in each histogram, the PROB value, and the maximum Kolmogorov distance between the two histograms. For 2-Dim histograms, there are two Kolmogorov distances (see below). If 'N' is specified, there is a third line of output giving the PROB for shape alone, and for normalization.
- O Overflow, requests that overflow bins be taken into account.
- U Underflow, requests that underflow bins be taken into account.
- L Left: include x-underflows
- R Right: include x-overflows
- T Top: include y-overflows
- B Bottom: include y-underflows
- F1 Histogram 1 has no error (is a function)
- F2 Histogram 2 has no error (is a function)

Test of compatibility for two 1-Dim histograms ID1 and ID2. A probability PROB is calculated as a number between zero and one, where PROB near one indicates very similar histograms, and PROB near zero means that it is very unlikely that the two arose from the same parent distribution. For two histograms sampled randomly from the same distribution, PROB will be (approximately) uniformly distributed between 0 and 1. See discussion in HBOOK manual under "HDIFF- Statistical Considerations". By default (if no options are selected with CHOPT) the comparison is done only on the shape of the two histograms, without consideration of the difference in numbers of events, and ignoring all underflow and overflow bins.

18.0.134 SORT id [chopt]

ID	C	"Histogram Identifier" Loop
CHOPT	C	"Options" D='XA'

Possible CHOPT values are:

- X X-axis is being treated.
- Y Y-axis is being treated.
- Z Z-axis is being treated.
- A Alphabetically.
- E Reverse alphabetical order.
- D By increasing channel contents.
- V By decreasing channel contents.

Sort the alphanumeric labels of the histogram ID according to the value of CHOPT.

18.0.135 SMOOTH id [option sensit smooth]

`ID`	C	"Histogram or Ntuple Identifier"	`Minus`
`OPTION`	C	"Options"	`D='2M'`
`SENSIT`	R	"Sensitivity parameter"	`D=1. R=0.3:3.`
`SMOOTH`	R	"Smoothness parameter"	`D=1. R=0.3:3.`

Possible `OPTION` values are:

- `0` Replace original histogram by smoothed.
- `1` Replace original histogram by smoothed.
- `2` Store values of smoothed function and its parameters without replacing the original histogram (but see note below) - the smoothed function can be displayed at editing time - see HISTOGRAM/PLOT.
- `M` Invoke multiquadric smoothing (see HBOOK routine HQUAD).
- `Q` Invoke the 353QH algorithm (see HBOOK routine HSMOOF).
- `S` Invoke spline smoothing.
- `V` Verbose (default for all except 1-D histogram).
- `N` Do not plot the result of the fit.
- `F` Write Fortran77 function to HQUADF.DAT (multiquadric only)

Smooth a histogram or "simple" ntuple. ("simple" = 1, 2, or 3 variables.)
For multiquadric smoothing, SENSIT controls the sensitivity to statistical fluctuations. SMOOTH controls the (radius of) curvature of the multiquadric basis functions.
Notes:
1) The multiquadric basis functions are SQRT(R**2+D**2), where R is the distance from the "centre", and D is a scale parameter and also the curvature at the "centre". "Centres" are located at points where the 2nd differential or Laplacian of event density is statistically significant.
2) The data must be statistically independent, i.e. events (weighted or unweighted) drawn randomly from a parent probability distribution or differential cross-section.
For spline smoothing, SENSIT and SMOOTH control the no. of knots (= 10 * SENSIT) and degree of splines (= SMOOTH + 2) (thus if SENSIT and SMOOTH are at their default values a 10-knot cubic spline is used).
Notes:
1) The spline option ALWAYS replaces the contents of a 2-D histogram. (Also chi-squared is unavailable in this case.)
2) Use the SPLINE command for more flexibility.

18.0.136 SPLINE id [isel knotx kx]

`ID`	C	"Histogram Identifier"
`ISEL`	I	"Option flag" `D=2`
`KNOTX`	I	"Number of knots" `D=10`
`KX`	I	"Degree of the spline" `D=3`

Smooth 1-Dim or 2-Dim histogram ID using B-splines. If ID is a 1-Dim histogram then:

```
ISEL = 0,1 replace original histogram by smoothed.
     = 2   superimpose as a function when editing.
```

If ID is a 2-Dim histogram then original contents are replaced.

18.0.137 PARAM id [isel r2min maxpow]

`ID`	C	"Histogram Identifier"
`ISEL`	I	"Control word" `D=11`
`R2MIN`	R	"Min correlation coefficient" `D=1.`
`MAXPOW`	I	"Max degree of polynomials" `D=5 R=1:20`

Perform a regression on contents of the 1-Dim histogram ID. Find the best parameterization in terms of elementary functions (regressors). See HBOOK guide HPARAM. Control word ISEL=1000*T +100*W +10*S +P

```
S = 1 resulting parametric fit superimposed on histogram
    0 no superposition
P = 0 minimal output: the residual sum of squares is printed
    1 normal output: in addition, the problem characteristics and
      options are printed; also the standard deviations and
      confidence intervals of the coefficients.
    2 extensive output: the results of each iteration are printed
      with the normal output.
W = 0 weights on histogram contents are already defined via HBARX
      or HPAKE. If not they are taken to be equal to the
      square-root of the contents.
    1 weights are equal to 1.
T = 0 monomials will be selected as the elementary functions
    1 Chebyshev polynomials with a definition region: [-1,1]
    2 Legendre polynomials with a definition region: [-1,1]
    3 shifted Chebyshev polynomials with a definition region: [0,1]
    4 Laguerre polynomials with a definition region: [0,+infinite]
    5 Hermite polynomials with a definition region: [-inf,+inf]
```

The FORTRAN code of the parameterization is written onto the file FPARAM.DAT.

18.0.138 HSETPR param value

`PARAM`	C	"Parameter name" `D='FEPS'`
`VALUE`	R	"Parameter value" `D=0.001`

Set various parameters for command PARAM.

Chapter 19

HISTOGRAM/GET_VECT

Fill a vector from values stored in HBOOK objects.

19.0.139 CONTENTS id vname

ID	C	"Histogram Identifier"
VNAME	C	"Vector name"

Get contents of histogram ID into vector VNAME.

19.0.140 ERRORS id vname

ID	C	"Histogram Identifier"
VNAME	C	"Vector name"

Get errors of histogram ID into vector VNAME.

19.0.141 FUNCTION id vname

ID	C	"Histogram Identifier"
VNAME	C	"Vector name"

Get function associated to histogram ID into vector VNAME.

19.0.142 ABSCISSA id vname

ID	C	"Histogram Identifier"
VNAME	C	"Vector name"

Get values of center of bins abscissa into vector VNAME.

19.0.143 REBIN id x y ex ey [n ifirst ilast chopt]

ID	C	"Histogram Identifier"
X	C	"Name of vector X"
Y	C	"Name of vector Y"
EX	C	"Name of vector EX"
EY	C	"Name of vector EY"
N	I	"Number of elements to fill" D=100
IFIRST	I	"First bin" D=1
ILAST	I	"Last bin" D=100
CHOPT	C	"Option" D='␣'

Possible CHOPT values are:

N Do not normalize values in Y

The specified channels of the 1-Dim histogram ID are cumulated (rebinned) into new bins. The final contents of the new bin is the average of the original bins by default. If the option N is given, the final contents of the new bin is the sum of the original bins. Get contents and errors into vectors, grouping bins. Bin width and centers are also extracted. Allow to combine 2, 3 or more bins into one.

```
E.g.: REBIN 110 X Y EX EY 25 11 85
       will group by 3 channels 11 to 85  and return
       new abscissa, contents and errors.
       Errors in X are equal to 1.5*BINWIDTH.
N.B.:
      REBIN ID X Y EX EY  is a convenient way to return in
      one call abscissa, contents and errors for 1-Dim histogram.
      In this case the errors in X are equal to 0.5*BINWIDTH.
```

Chapter 20

HISTOGRAM/PUT_VECT

Replace histogram contents with values in a vector.

20.0.144 CONTENTS id vname

ID	C	"Histogram Identifier"
VNAME	C	"Vector name"

Replace contents of histogram with values of vector VNAME.

20.0.145 ERRORS id vname

ID	C	"Histogram Identifier"
VNAME	C	"Vector name"

Replace errors of histogram with values of vector VNAME.

Chapter 21

HISTOGRAM/SET

Set histogram attributes.

21.0.146 MAXIMUM id vmax

`ID`	C	"Histogram Identifier"	`Loop`
`VMAX`	R	"Maximum value"	

Set the maximum value on the Y axis. To select again an automatic scale, just set VMAX less then the minimum.

21.0.147 MINIMUM id vmin

`ID`	C	"Histogram Identifier"	`Loop`
`VMIN`	R	"Minimum value"	

Set the minimum value on the Y axis. To select again an automatic scale, just set VMIN greater then the maximum.

21.0.148 NORMALIZE_FACTOR id [xnorm]

`ID`	C	"Histogram Identifier"	
`XNORM`	R	"Normalization factor"	`D=1`

Set the contents/errors normalization factor. Only valid for histograms (1-Dim). (does not change contents, only presentation).

21.0.149 SCALE_FACTOR_2D id [xscale]

`ID`	C	"Histogram Identifier"	
`XSCALE`	R	"Scale factor"	`D=0`

Set the scale factor for histograms (2-Dim).

21.0.150 IDOPT id option

ID	C	"Histogram Identifier"
OPTION	C	"Options"

Possible OPTION values are:

SETD*	Set all options to the default values
SHOW	Print all the options currently set
BLAC	1 Dim histogram printed with X characters
CONT*	1 Dim histogram is printed with the contour option
STAR	1 Dim histogram is printed with a * at the Y value
SCAT*	Print a 2 Dim histogram as a scatter-plot
TABL	Print a 2 Dim histogram as a table
PROE*	Plot errors as the error on mean of bin in Y for profile histograms
PROS	Plot errors as the Spread of each bin in Y for profile histograms
STAT	Mean value and RMS computed at filling time
NSTA*	Mean value and RMS computed from bin contents only
ERRO	Errors bars printed as SQRT(contents)
NERR*	Do not print print error bars
INTE	Print the values of integrated contents bin by bin
NINT*	Do not print integrated contents
LOGY	1 Dim histogram is printed in Log scale in Y
LINY*	1 Dim histogram is printed in linear scale in Y
PCHA*	Print channel numbers
NPCH	Do not print channel numbers
PCON*	Print bin contents
NPCO	Do not print bin contents
PLOW*	Print values of low edge of the bins
NPLO	Do not print the low edge
PERR	Print the values of the errors for each bin
NPER*	Do not print the values of the errors
PFUN	Print the values of the associated function bin by bin
NPFU*	Do not print the values of the associated function
PHIS*	Print the histogram profile
NPHI	Do not print the histogram profile
PSTA*	Print the values of statistics (entries,mean,RMS,etc.)
NPST	Do not print values of statistics
ROTA	Print histogram rotated by 90 degrees
NROT*	Print histogram vertically
1EVL	Force an integer value for the steps in the Y axis
AEVL*	Steps for the Y axis are automatically computed
2PAG	Histogram is printed over two pages
1PAG*	Histogram is printed in one single page
AUTO*	Automatic scaling

Set options for histogram ID. (* means default).

Chapter 22

FUNCTION

Operations with Functions. Creation and plotting.

22.0.151 FUN1 id ufunc ncx xmin xmax [chopt]

ID	C	"Histogram Identifier"
UFUNC	C	"Name of the function"
NCX	I	"Number of channels" D=100 R=1:
XMIN	R	"Low edge" D=0.
XMAX	R	"Upper edge" D=100.
CHOPT	C	"Options" D='P'

Possible CHOPT values are:

P The function is drawn.

Create a one dimensional histogram and fill the bins with the values of a (single-valued) function. The function UFUNC may be given in two ways:
-An expression of the variable x in case of a simple function.

```
Ex: FUN1  10 sin(x)/x  100 0 10
```

-UFUNC is the name of a COMIS function in a text file with the name UFUNC.FTN or UFUNC.FOR or UFUNC FORTRAN (Apollo, VAX, IBM).

22.0.152 FUN2 id ufunc ncx xmin xmax ncy ymin ymax [chopt]

ID	C	"Histogram (2-Dim) Identifier"
UFUNC	C	"Name of the function"
NCX	I	"Number of channels in X" D=40 R=1:
XMIN	R	"Low edge in X" D=0.
XMAX	R	"Upper edge in X" D=40.
NCY	I	"Number of channels in Y" D=40 R=1:
YMIN	R	"Low edge in Y" D=0.
YMAX	R	"Upper edge in Y" D=40.
CHOPT	C	"Options" D='S'

Possible CHOPT values are:

'␣' Create the histogram.

S The function is drawn as a surface.

L The function is drawn as a lego plot.

C The function is drawn as a contour plot.

Create a two dimensional histogram and fill the bins with the values of a (two-valued) function. The function UFUNC may be given in two ways:
-An expression of the variables x and y in case of a simple function.

```
Ex: FUN2  10 abs(sin(x**2+y**2)) 40 -2 2 40 -2 2 C
```

-UFUNC is the name of a COMIS function in a text file with the name UFUNC.FTN or UFUNC.FOR or UFUNC FORTRAN (Apollo, VAX, IBM).

22.0.153 DRAW ufunc [chopt]

UFUNC	C	"Name of function"
CHOPT	C	"Options" D='␣'

Draw the function UFUNC in the current ranges specified by the command: RANGE XLOW XUP YLOW YUP ZLOW ZUP and with THETHA and PHI angles specified by the command ANGLE THETA PHI. The number of points to evaluate the function between XLOW, XUP YLOW, YUP, and ZLOW, ZUP can be changed by the command POINTS NPX NPY NPZ.
The function UFUNC may be given in two ways: - As an expression of the variables X, Y, Z in the case of a

```
simple function.
Ex:
PAW >  FUN/DRAW X*Y*Z         | equivalent to :
PAW >  FUN/DRAW X*Y*Z=0
PAW >  FUN/DRAW X**2+Y**2+Z**2=1
PAW >  FUN/DRAW X**2+Y**2=1-Z**2
```

- As a COMIS function in a text file with the name UFUNC.FTN or

```
UFUNC.FOR or UFUNC FORTRAN (Apollo, VAX, IBM).
Ex:
The file FTEST.FOR contains:

FUNCTION FTEST(X,Y,Z)
IF(X.LE.0..AND.Y.LE.0.)THEN
  FTEST=(X+0.5)**2+(Y+0.5)**2+(Z+0.5)**2-0.2
ELSE
  FTEST=(X-0.5)**2+(Y-0.5)**2+(Z-0.5)**2-0.1
ENDIF
END

  PAW > RANGE -1 1 -1 1 -1 1  | Define the range as a cube between -1 1 in
the 3
                                directions
  PAW > POINTS 20 20 20       | FUN/DRAW will use 20 points in the 3
directions
  PAW > FUN/DRAW FTEST.FOR    | Draw 2 spheres centered on (-0.5,-0.5,-0.5)
                                and (0.5,0.5,0.5) with the radius SQRT(0.2)
                                and SQRT(0.1)
```

22.0.154 PLOT ufunc xlow xup [chopt]

UFUNC	C	"Name of function"
XLOW	R	"Lower limit"
XUP	R	"Upper limit"
CHOPT	C	"Options" D='C'

Possible CHOPT values are:

- C Draw a smooth curve.
- S Superimpose plot on top of existing picture.
- + Add contents of ID to last plotted histogram.
- L Connect channel contents by a line.
- P Draw the current polymarker at each channel.
- * Draw a * at each channel.

Plot single-valued function UFUNC between XLOW and XUP. The function UFUNC may be given in two ways:
-An expression of the variable x in case of a simple function.

```
Ex: FUN/PLOT   sin(x)/x  0 10
```

-UFUNC is the name of a COMIS function in a text file with the name UFUNC.FTN or UFUNC.FOR or UFUNC FORTRAN (Apollo, VAX, IBM). For example, if the file FTEST.FOR contains:

```
      FUNCTION FTEST(X)
      FTEST=SIN(X)*EXP(-0.1*X)
      END
```

Then, FUN/PLOT FTEST.FOR 0 10, will interpret the Fortran code in the file FTEST.FOR and draw the function for x between 0 and 10.
The number of points to evaluate the function between XLOW and XUP can be changed by the command /FUN/POINTS. Only 1-Dim functions are supported. For 2-Dim use FUN2.

22.0.155 POINTS [npx npy npz]

NPX	I	"Number of points on X axis" D=20 R=2:1000
NPY	I	"Number of points on Y axis" D=20 R=2:1000
NPZ	I	"Number of points on Z axis" D=20 R=2:1000

Change the number of points to be used by FUN/DRAW and FUN/PLOT. Note that the default for NPX is 20 for 3-Dim plots (FUN/DRAW) but it is 100 for 1-Dim plots (FUN/PLOT).

22.0.156 RANGE [xlow xup ylow yup zlow zup]

XLOW	R	"X Lower limit" D=-1.
XUP	R	"X Upper limit" D=1.
YLOW	R	"Y Lower limit" D=-1.
YUP	R	"Y Upper limit" D=1
ZLOW	R	"Z Lower limit" D=-1.
ZUP	R	"Z Upper limit" D=1.

Change the range used by FUN/DRAW.

22.0.157 ANGLE [theta phi]

THETA	R	"Angle THETA in degrees" D=30.
PHI	R	"Angle PHI in degrees" D=30.

Change the angle used by FUN/DRAW and HISTO/PLOT.

Chapter 23

NTUPLE

Ntuple creation and related operations.

23.0.158 CREATE idn title nvar chrzpa nprime varlist

IDN	C	"Ntuple Identifier"
TITLE	C	"Ntuple title" D='␣'
NVAR	I	"Number of variables" D=1 R=1:512
CHRZPA	C	"RZ path" D='␣'
NPRIME	I	"Primary allocation" D=1000
VARLIST	C	"Names of the NVAR variables" Vararg

Create a Row_Wise_Ntuple. (See below how to create a Column_Wise_Ntuple). The Ntuple may be created either purely in memory or possibly using an automatic overflow to an RZ file. Memory allocation works in the following way. If CHRZPA = ' ', then a bank of NPRIME words is created. When the space in this bank is exhausted at filling time, a new linear structure of length NPRIME is created and this process will be repeated should the structure become exhausted. If CHRZPA contains the top directory name of an already existing RZ file (as declared with HISTO/FILE), then a bank of length NPRIME is also created, but at filling time, this bank is moved to the RZ file when full, and then it is overwritten by any new entries. The Ntuple can be filled by calling HFN from an interactively defined subroutine called by the command NTUPLE/LOOP or by NTUPLE/READ. The number of variables per data point is given in the parameter NVAR.
To create a Column_Wise_Ntuple, create a file, eg. newnt.f with:

```
      Subroutine Newnt
      character*8 mother,in1,in2
      common/ntupc/mother,in1,in2
      common/ntupr/xover
      lin=41
      lout=42
      id=1
      open(unit=lin,file='datafile.dat',status='old')
      call hropen(lout,'NTUPLE','New_Ntuple.hbook','N',1024,istat)
      call hbnt(id,'New Ntuple',' ')
      call hbname(id,'ntupr',xover,'XOVER')
      call hbnamc(id,'ntupc',mother,'MOTHER:c*8,in1:c*8,in2:c*8')
   10 read(lin,1000,end=20,err=20)xover,mother,in1,in2
```

1000 format(e15.7,2x,a,7x,a,7x,a)

```
      call hfnt(1)
      go to 10
   20 call hrout(id,icycle,' ')
      call hrend('NTUPLE')
      close (lin)
      close (lout)
      end
```

and then call this routine via the CALL command:

```
PAW > call newnt.f
```

23.0.159 LIST

List all Ntuples in the Current Directory. Note that the command HISTO/LIST lists all histograms and Ntuples in the Current Directory.

23.0.160 PRINT idn

IDN	C	"Ntuple Identifier"

Print a summary about Ntuple IDN. Number of entries, variables names and limits are listed.

23.0.161 HMERGE outfile infiles

OUTFILE	C	"Output file name" D='␣'
INFILES	C	"Input file names" D='␣' Vararg

Merge HBOOK files containing histograms and/or ntuples. Ntuples are merged and histograms with the same ID are added. The INFILES are merged into a new file OUTFILE. If OUTFILE already exists, it is overwritten.
Note that if there is a histogram in PAW memory that has the same identifier as a histogram in one of the files to be merged, then the contents of the histogram in memory are added to those of the histogram in the file. This can be avoided by deleting the memory histogram (using H/DEL) before issuing the HMERGE command.

23.0.162 DUPLICATE id1 id2 [newbuf title option]

ID1	C	"Source Ntuple"
ID2	I	"New Ntuple"
NEWBUF	I	"Buffer size" D=-1
TITLE	C	"Title of ID2" D='␣'
OPTION	C	"Options" D='A'

Possible OPTION values are:

'␣'

A Set the Addresses of variables in common /PAWCR4,etc/.

M Create ID2 as a Memory resident Ntuple.

'␣' Copy ID1 structure in ID2. Reset addresses of variables.

The structure of Ntuple ID1 is duplicated in a new ntuple ID2. This command is useful when one wants to create an ntuple with the same variables but only a subset of the events. NEWBUF is the buffer size for ID2. If NEWBUF⟨0 the buffer size of ID1 is taken. If NEWBUF=0 the current buffer size is taken (10000 words for RWNs). NEWBUF⟩0 will be the new buffer size. If TITLE=' ' ID2 has the same title as ID1. In case of a disk-resident ntuple (default), ID2 is created into the current working directory which must be open in WRITE mode.

Example with a Row Wise Ntuple:

```
Macro DUPRWN
Close 0
Hi/File 1 source.hbook
Hi/File 2 new.hbook ! N
Nt/Dup //lun1/30 2
*
Application Comis Quit
      Real Function Dup(dum)
      Include ?
      If (X.gt.0..Or.Y.gt.0.) call hfn(2,X)
      dup=1.
      end
Quit
*
nt/loop //lun1/30 dup
hrout 2
```

Note that the statement "include ?" allows to create automatically the inlude file (comis.inc) corresponding to the ntuple structure. The command UWFUNC is not required in this case.

Example with a Column Wise Ntuple:

```
Macro DUPCWN
*
 Close 0                    | Close all the currently opened file
 H/file 1 source.hbook
 Uwfunc //lun1/1 source.inc | generate source.inc
 H/file 2 new.hbook ! N     | Create a new hbook file
 Nt/Dup //lun1/1 2          | Duplicate the ntuple 1 in the ntuple 2
*
* Comis routine which Loop on all events of Id1 and select some events
* to be written in the new ntuple Id2.
*
Application COMIS quit
      Subroutine ntdup(Id1,Id2)
      Include 'source.inc'
      Call Hnoent(Id1,Noent)
      Do Ievent=1,Noent
         Call Hgnt(Id1,Ievent,Ierr)
         If (Ierr.ne.0) Goto 20
         If (X.Gt.0..Or.Y.Gt.0.) Then
            Call Hfnt(Id2)
         Endif
      Enddo
  20  Continue
*
      End
Quit
*
 Call Ntdup(1,2)             | Execute the routine Ntdup
 Hrout 2                     | Write Id2 on disk
```

23.0.163 RECOVER idn

`IDN` I "Ntuple Identifier"

To recover Ntuple ID. If the job producing the Ntuple crashed or the header was not stored correctly in the file with HROUT, RECOVER will scan the Ntuple to rebuild the header table and recompute the number of entries. The file on which the Ntuple resides must be open in Update mode.

23.0.164 SCAN idn [uwfunc nevent ifirst option varlis]

`IDN` C "Ntuple Identifier"
`UWFUNC` C "User cut function" `D='0'`
`NEVENT` I "Number of events" `D=99999999`
`IFIRST` I "First event" `D=1`
`OPTION` C "Options" `D='␣'`
`VARLIS` C "Names of the NVARS variables to scan" `D='␣' Vararg`

Possible `OPTION` values are:
`'␣'`

- `S` Graphical scan (spider plot).
- `'␣'` Alphanumeric output of the Ntuple.
- `A` Used with "S" it displays the average spider.

Scan the entries of an Ntuple subject to user cuts. Scan the variables for NEVENT events starting at IFIRST, requiring that the events satisfy cut UWFUNC. In the case of Alphanumeric output Up to 8 variables may be scanned, the default is to scan the first 8 variables.
When the option S (Spider plot) is specified, each event is presented in a graphical form (R versus PHI plot) to give a multi dimensional view of the event. Each variable is represented on a separate axis with a scale ranging from the minimum to the maximum value of the variable. A line joins all the current points on every axis where each point corresponds to the current value of the variable. When the HCOL parameter is specified (eg SET HCOL 1002) a fill area is drawn.
VARLIS may contain a list of the original variables, expressions of the original variables or/and ranges of variables. A range can be given in the following form:

```
:          means all variables (default).
var1:var2  means from variable var1 to variable var2 included.
var1:      means from variable var1 to the last.
:var2      means from variable 1 to variable var2
```

For example, if IDN=30 has the 3 variables X,Y,Z,U,V,W one can do:

```
PAW > scan 30
PAW > scan 30 option=s
      each event is drawn as a spider plot.
PAW > scan 30 option=sa
      each event is drawn as a spider plot and the average spider
      plot is also drawn.
PAW > scan 30 option=s  X:Z W
PAW > scan 30 z>10
PAW > scan 30 z>10 ! ! ! z abs(x) y+z x func.for
      where func.for is a COMIS function returning an expression
      of the original variables. This function func.for may be
      generated automatically by the PAW command:
PAW > uwfunc 30 func.for
```

23.0.165 LOOP idn uwfunc [nevent ifirst]

`IDN` C "Identifier of Ntuple"
`UWFUNC` C "Selection function or cut identifier" `D='␣'`
`NEVENT` I "Number of events" `D=99999999`
`IFIRST` I "First event" `D=1`

Invoke the selection function UWFUNC for each event starting at event IFIRST. In UWFUNC, the user can fill one or several histograms previously booked. The loop will be terminated if UWFUNC returns a negative value. For more information about UWFUNC, see command NTUPLE/PLOT.

23.0.166 MERGE idn1 idn2 [uwfunc nevent ifirst]

IDN1	C	"Identifier of first Ntuple"
IDN2	C	"Identifier of second Ntuple"
UWFUNC	C	"Selection function or cut identifier" D='␣'
NEVENT	I	"Number of events" D=99999999
IFIRST	I	"First event" D=1

Merge two Disk-Resident Row-Wise-Ntuples. Invoke the selection function UWFUNC for each of the NEVENT events starting at event IFIRST of Ntuple IDN1. Suppose you have 4 files containing Ntuple ID=10 and you want to merge the 4 files into the file 4, the sequence is:

```
PAW >Histo/file 1 file1
PAW >Histo/file 2 file2
PAW >Histo/file 3 file3
PAW >Histo/file 4 file4 1024 U
PAW >Ntuple/Merge //lun1/10 //lun4/10
PAW >Ntuple/Merge //lun2/10 //lun4/10
PAW >Ntuple/Merge //lun3/10 //lun4/10
PAW >Ntuple/plot 10.x .........
```

Only the events with UWFUNC⟩0 are appended to IDN2. IDN2 may be empty. Note that the Ntuple variables may be redefined inside UWFUNC. For more information about UWFUNC, see command NTUPLE/PLOT. Note that this command cannot be used for memory resident ntuples or CWNs. Use instead the command HMERGE.

23.0.167 PROJECT idh idn [uwfunc nevent ifirst]

IDH	C	"Identifier of histogram to fill"
IDN	C	"Identifier of Ntuple"
UWFUNC	C	"Selection function or cut identifier" D='␣'
NEVENT	I	"Number of events" D=99999999
IFIRST	I	"First event" D=1

Project an Ntuple onto a 1-Dim or 2-Dim histogram, possibly using a selection function or predefined cuts. IDN may be given as IDN or IDN.X , IDN.Y%X , IDN.1, IDN.2%1. Y%X means variable Y of Ntuple IDN versus variable X. For more information about UWFUNC, see command NTUPLE/PLOT. The histogram IDH is not reset before filling. This allows several PROJECTs from different Ntuples.

23.0.168 READ idn fname [format chopt nevent]

IDN	C	"Ntuple Identifier"
FNAME	C	"File name"
FORMAT	C	"Format" D='*'
CHOPT	C	"Options" D='␣'
NEVENT	I	"Number of events" D=1000000

Read Ntuple values from the alphanumeric file FNAME with the format specifications in FORMAT. Before executing this command, the Ntuple IDN must have been created with the command Ntuple/Create.

23.0.169 PLOT idn [uwfunc nevent ifirst nupd option idh]

IDN	C	"Ntuple Identifier"
UWFUNC	C	"Selection function" D='0'
NEVENT	I	"Number of events" D=99999999
IFIRST	I	"First event" D=1
NUPD	I	"Frequency to update histogram" D=100000000
OPTION	C	"Options" D='␣'
IDH	I	"Identifier of histogram to fill" D=1000000

Possible OPTION values are:

'␣'	
C	Draw a smooth curve.
S	Superimpose plot on top of existing picture.
+	Add contents of IDN to last plotted ntuple.
B	Bar chart format.
L	Connect channels contents by a line.
P	Draw the current polymarker at each channel or cell.
*	Draw a * at each channel.
U	Update channels modified since last call.
E	Compute (HBARX) and draw error bars with current marker.
A	Axis labels and tick marks are not drawn.
'␣'	Draw the ntuple as an histogram.
PROF	Fill a Profile histogram (mean option).
PROFS	Fill a Profile histogram (spread option).
PROFI	Fill a Profile histogram (integer spread option).

Project and plot an Ntuple as a (1-Dim or 2-Dim) histogram with automatic binning (ID=1000000), possibly using a selection algorithm. See parameter CHOPT in command HISTO/PLOT to have more details on the possible OPTION.

```
IDN may be given as IDN
                    IDN.X
                    IDN.Y%X
                    IDN.1
                    IDN.2%1
                    IDN.expression1
                    IDN.expression1%expression2
```

Y%X means a scatter-plot Y(I) versus X(I) where I is the event number. 2%1 means a scatter-plot variable 2 versus variable 1. In this example, X and Y are the names of the variables 1 and 2 respectively. Expression 1 is any numerical expression of the Ntuple variables. It may include a call to a COMIS function.

```
UWFUNC may have the following forms:
1- UWFUNC='0' or missing (only IDN given). No selection is applied.
2- UWFUNC is a CUT or combination of valid CUTS created by the
   command NTUPLE/CUTS. Ex:
          UWFUNC=$1            means use cut $1
          UWFUNC=$1.AND.$2
          UWFUNC=.NOT.($1.AND.$2)
          UWFUNC=($1.OR.$2).AND.$3
3- UWFUNC is a FORTRAN expression
   Ex:    X>3.14.AND.(Y<Z+3.15)
4- UWFUNC is a variable name or an arithmetic expression
   Ex:   NT/PLOT 30.X Y  weight of each event is variable Y
         NT/PLOT 30.X X**2+Y**2
5- UWFUNC is the name of a selection function in a text file with
   the name UWFUNC.FTN, UWFUNC.FOR, UWFUNC FORTRAN (Apollo, VAX, IBM).
```

The command UWFUNC may be used to generate automatically this function. For example if IDN=30 is an Ntuple with 3 variables per event and 10000 events, then

```
NTUPLE/PLOT 30.X SELECT.FOR
```

will process the 10000 events of the Ntuple IDN=30. For each event, the function SELECT is called. It returns the weight of the event. Example:

```
FUNCTION SELECT(X)
DIMENSION X(3)
IF(X(1)**1+X(2)**2.LT.1.5)THEN
   SELECT=0.
ELSE
   SELECT=1.
ENDIF
END
```

The file SELECT.FOR (VAX), SELECT.FTN (Apollo) or SELECT FORTRAN (IBM) can be edited from PAW using the command EDIT. Note that if the suffix (.FTN, .FORTRAN or .FOR) is omitted, then COMIS will start from the precompiled version in memory and not from the file. Results of a selection can be saved in a MASK (See NTUPLE/MASK).

```
Ex: NT/PLOT 30.X Z<0.4>>MNAME(4)
    means mark bit 4 in mask MNAME for all events satisfying
    the condition Z<0.4
```

A MASK may also be given as input to a selection expression.

```
Ex:  NT/PLOT 30.X MNAME(4).and.Z<0.4
     means all events satisfying bit 4 of MNAME AND Z<0.4
```

It is possible to plot expressions of the original variables.

```
Ex 1:  NT/PLOT 30.SIN(X)%SQRT(Y**2+Z**2)  Z<0.4
    plots a scatter-plot of variable U versus V for all events
    satisfying the condition Z<0.4. U and V are defined as being
    U=SIN(X) and V=SQRT(X**2+Y**2)
Ex 2:  NT/PLOT 30.FUNC.FTN(X)%(SIN(Y)+3.)  Z<0.2.and.TEST.FTN>6
    plots a scatter-plot of variable U versus V for all events
    satisfying the condition (Z<0.2 and the result of the COMIS
    function TEST.FTN >6). U and V are defined as being
    U=Result of the COMIS function FUNC.FTN,  V=SIN(Y)+3.
```

The default identifier of the histogram being filled is IDH=1000000. At the next invocation of this command, it will be overwritten. If either NEVENT or IFIRST or NUPD are negative, then the identifier of the histogram being filled will be taken as IDH=-NEVENT or IDH=-IFIRST or IDH=-NUPD. IDH may have been created with H/CREATE. Before filling IDH, the contents of IDH are reset if IDH already exists. Use NTUPLE/PROJECT to cumulate several passes into IDH. Note that IDH not equal to 1000000 is a convenient way to force user binning. Every NUPD events, the current status of the histogram is displayed.

23.0.170 CHAIN [cname entry]

`CNAME` C "Chain Name" `D='␣'`
`ENTRY` C "Chain Member(s) | -P Path" `D='␣' Vararg`

Using the chain command one can build logical Ntuples of unlimited size. The chain command creates an Ntuple chain CNAME and add member(s) ENTRY. If the chain already exists the member is simply added. More than one member may be specified at a time. A chain can contain three different type of members: files, logical units and other chains. The member type is deduced from the format of the member. Entries containing the characters . / : ; $ are considered to be files, entries like //LUN4 are assumed to be logical units and all other type of entries are chains. Chain names must be unique. After a chain has been defined it can be traversed, by all Ntuple commands (NT/PLOT, NT/PROJ, NT/LOOP), by changing the current working directory to the chain: CD //CNAME. A member may be deleted from a chain by preceding it by a - sign. A complete chain can be deleted by preceding the chain name by a -. All chains can be deleted by giving a - as chain name. Not specifying any parameters results in the listing of all defined chains. A chain tree will be printed by appending a ⟩ character to the chain name. The path of all chain members, from chain CNAME downwards, can be changed by specifying a chain path. This is done by giving a chain name followed by the -P option and a path specification. The chain path will be pre-pended to the member names. Chains down the tree can override a path specified higher up in the tree.

```
Examples of chain (Ntuple tree) definition:
CHAIN   Year93  Jan Feb March April May ...
CHAIN   Jan     Week1 Week2 Week3 Week4
CHAIN   Week1   file1.hbook file2.hbook ...
CHAIN   Week2   file3.hbook file4.hbook ...
CD //Jan
NT/PLOT 10.e     ; loop over all files in chains Week1, Week2, Week3, ...
CD //Year93      ; loop over all files in chains Jan, Feb, March, ...
CHAIN Year93 -P /user/delphi   ; all files from chain Year93 downward will
                                 be changed to /user/delphi/file1.hbook,
...
CHAIN Year93>    ; print the chain tree Year93
CHAIN -Feb       ; delete chain Feb
CHAIN Jan -file3.hbook   ; delete file3.hbook from chain Jan
```

23.0.171 DRAW idn [value option]

`IDN` C "Ntuple Identifier"
`VALUE` C "Isosurface value (for 3-D)" `D='0'`
`OPTION` C "Options" `D='␣'`

Draw a simple ntuple (1, 2 or 3 variables). For simple ntuples, with 1, 2 or 3 variables per event, this command will draw a histogram with HPLOT options. If the ntuple has an associated functional representation, as the result, e.g., of using SMOOTH, it will also draw the function. No selections are allowed.
For 3-variable ntuples which have been SMOOTHed, give a VALUE for the isosurface of event density. If VALUE=0, an isosurface value half way between the minimum and maximum fitted smoothing function values will be used.

23.0.172 WAVE idn [lun]

`IDN` C "Ntuple Identifier"
`LUN` I "Logical unit no." `D=-1`

Produce a formatted file suitable for Wavefront's Data Visualiser. Only for simple 3-variable ntuples which have been SMOOTHed. A file with logical unit no. LUN must previously have been opened with the FORTRAN/FILE command.

23.0.173 CUTS cutid [option fname wkid]

`CUTID`	C	"Cut identifier"
`OPTION`	C	"Options" `D='P' Minus`
`FNAME`	C	"File name" `D='␣'`
`WKID`	I	"Workstation identifier" `D=1`

Possible `OPTION` values are:

- `G` Define a new cut CUTID using graphics input on the latest 1-Dim or 2-Dim projection of the Ntuple. For a 1-Dim projection, give 2 points cutmin,cutmax. For a 2-Dim projection, give up to 20 points to delimit the selected area. The polygon will automatically be closed by PAW.
- `X` Same as G but with a tracking cross cursor.
- `P` Print definition of cut CUTID.
- `-` Reset cut CUTID.
- `R` Read definition of cut CUTID from file FNAME.
- `W` Write definition of cut CUTID on file FNAME (text file).
- `D` Draw cut contour.

Define the CUTID with the format $nn. nn is an integer between 1 and 99. This cut can then be used in subsequent commands NTUPLE/PLOT, PROJECT.

```
OPTION='expression'  allows to define the cut CUTID. For example
the command:
 PAW > CUTS $1 X<0.8.and.Y<SQRT(X)
defines the cut $1.
```

Note that CUTID=$0 means all cuts except for 'G' option. When option G is selected, graphical cuts are only operational for plots of the original Ntuple variables, not for expressions of these variables. WKID allows to define in which window the locator is performed (option 'G' or 'X' only).

23.0.174 CSELECT [chopt csize]

`CHOPT`	C	"Options" `D='N'`
`CSIZE`	R	"Comment size" `D=0.28`

Possible `CHOPT` values are:

- `'␣'` Comment is left adjusted to the current zone
- `R` Comment is right adjusted to the current zone
- `C` Comment is centered to the current zone
- `B` Comment is drawn below the top zone line
- `N` All subsequent NTUPLE/PLOT commands will print the selection mechanism with the options specified in CHOPT.

To write selection mechanism as a comment on the picture. By default, the comment is drawn left justified above the top zone line. Example:

```
CSEL          All coming NT/PLOT commands will draw a comment
              of size CSIZE=0.28cm Left justified.
CSEL NRB 0.4  All coming NT/PLOT commands will draw a comment
              of size 0.4 cm Right justified Below the top line.
CSEL  CB      Draw previous selection mechanism Centered Below
              the top zone line.
```

The Global title font (SET GFON) with precision 1 is used to draw the text.

23.0.175 MASK mname [chopt number]

`MNAME`	C	"Mask name"
`CHOPT`	C	"Options" `D='␣'`
`NUMBER`	I	"Bit number" `D=0`

Possible `CHOPT` values are:

- `'␣'` Existing mask on file MNAME.MASK is attached for READ only.
- `U` Existing mask on file MNAME.MASK is attached for UPDATE.
- `N` A new mask on file MNAME.MASK is created for NUMBER events.
- `P` The comments for all active bits is printed.
- `C` Mask is closed.
- `R` Reset bit number NUMBER.If NUMBER=99, resets all bits.

Perform Operations with masks. A mask is a direct-access file with the name MNAME.MASK. It must contain as many 32 bit words as there are events in the associated Ntuple. Masks are interesting when only a few events of a Ntuple are selected with a time consuming selection algorithm. For example if the command:

```
NT/PLOT 30.X  Z<0.4.AND.SELECT.FTN>>MNAME(6)
```

then for all events in Ntuple 30 satisfying the condition above, the bit 6 in the corresponding mask words will be set. One can then use the mask as selection mechanism. Example:

```
NT/PLOT 30.X MNAME(6)
```

will produce the same results than the NT/PLOT command above, but will be much faster if only a small fraction of all the events is selected. MASKS are automatically saved across PAW sessions on files. Example:

```
MASK  TEST N 10000
   creates a new mask on file TEST.MASK with enough words to
   process a Ntuple with 10000 events
MASK  TEST UP
   opens an existing mask for update and
   prints the active selection bits with explanation
```

23.0.176 UWFUNC idn fname [chopt]

IDN	C	"Ntuple Identifier"
FNAME	C	"File name"
CHOPT	C	"Options" D='␣'

Possible CHOPT values are:

- '␣' Generate the FORTRAN skeleton of a selection function.
- E Present the selection function in the local editor.
- P Code to print events is generated (not valid for new Ntuples).
- T Names of the Ntuple variables are generated in DATA statements (not valid for new Ntuples).

To generate the FORTRAN skeleton of a selection function or the INCLUDE file with the columns declaration.
A FORTRAN function is generated if the FNAME is of the form, xxx.f, xxx.for, xxx.fortran. Otherwise an INCLUDE file is generated. Example: If Ntuple ID=30 has variable names [X,Y,Z,ETOT,EMISS,etc] then:
NTUPLE/UWFUNC 30 SELECT.FOR will generate the file SELECT.FOR with:

```
      FUNCTION SELECT(XDUMMY)
      COMMON/PAWIDN/IDNEVT,VIDN1,VIDN2,VIDN3,X,Y,Z,ETOT,EMISS,etc
      SELECT=1.
      END
```

Then using the command EDIT one can modify this file which could then look something like (IDNEVT is the event number):

```
      FUNCTION SELECT(XDUMMY)
      COMMON/PAWIDN/IDNEVT,VIDN1,VIDN2,VIDN3,X,Y,Z,ETOT,EMISS,etc
      IF(X**2+Y**2.GT.Z**2.OR.ETOT.GT.20.)THEN
         SELECT=1.
      ELSE
         SELECT=0.
      ENDIF
      END
```

If in a subsequent command NTUPLE/PLOT, the selection function SELECT is used, then:

```
   If NTUPLE/PLOT 30.ETOT SELECT.FOR
      VIDN1=ETOT
   If NTUPLE/PLOT 30.SQRT(X**2+Y**2)%(ETOT-EMISS)
      VIDN1=ETOT-EMISS
      VIDN2=SQRT(X**2+Y**2)
```

NTUPLE/UWFUNC 30 SELECT.INC will generate an include file. This include file may be referenced in a selection function in the following way:

```
      FUNCTION SELECT(XDUMMY)
      include 'select.inc'
      SELECT=1.
      IF(X.LE.Y)SELECT=0.
      END
```

Note that the command UWFUNC is not required if the SELECT function has the following form:

```
      FUNCTION SELECT(XDUMMY)
      include ?
      SELECT=1.
      IF(X.LE.Y)SELECT=0.
      END
```


In this case (thanks to the statement "include ?") the include file will be generated automatically with the name "comis.inc".

23.0.177 LINTRA idn [chopt nevent ifirst nvars varlis]

IDN	C	"Ntuple Identifier"
CHOPT	C	"Options" D='␣'
NEVENT	I	"Number of events" D=99999999
IFIRST	I	"First event" D=1
NVARS	I	"Number of the most significant variables " D=20 R=0:20
VARLIS	C	"Names of the NVARS most significant variables "

Possible CHOPT values are:

- N The variables are normalized. This option is useful in the case the ranges of variables are very different
- P Print more results about the analysis

Data reduction on Ntuple. The method used is the PRINCIPAL COMPONENTS ANALYSIS. The Principal Components Analysis method consists in applying a linear transformation to the original variables of a ntuple. This transformation is described by an orthogonal matrix and is equivalent to a rotation of the original space to a new set of coordinates vectors, which hopefully provide easier identification and dimensionality reduction. This matrix is real positive definite and symmetric and has all its eigenvalues greater than zero. Among the family of all complete orthonormal bases, the basis formed by the eigenvectors of the covariance matrix and belonging to the largest eigenvalues corresponds to the most significant features for the description of the original ntuple. Reduction of the variables for NEVENT events starting at IFIRST The default is to take all the 20 first variables. This command creates a file : -) XTOXSI.FORTRAN or xtoxsi.for,xtoxsi.ftn. This file contains a Fortran function which computes the new variables. These new variables can be visualized in PAW with for example:

```
PAW > Ntuple/plot id.xtoxsi.ftn(1)
PAW > Ntuple/plot id.xtoxsi.ftn(1)%xtoxsi.ftn(3)
```

23.0.178 VMEM [mxsize]

MXSIZE	I	"Maximum size of dynamic memory buffer in MBytes" D=-1 R=-2:128

Change or show the size of the dynamic memory buffer used to store Ntuple columns during Ntuple analysis. The default is 10 MB. Giving a value of 0 turns the buffer facility off. The upper limit is 128 MB, but be sure you have enough swap space and realize that when the buffer is swapped to disk you loose part of the benefit of the buffer facility (which is to reduce the number of disk accesses). Omitting the argument or specifying -1 will show you the current upper limit and used and free space. Giving -2 shows which columns are currently stored in memory.

Chapter 24

GRAPHICS

Interface to the graphics packages HPLOT and HIGZ.

24.0.179 SET [chatt value]

`CHATT` C "Attribute name" D='SHOW'
`VALUE` R "Attribute value" D=0

Set a specific HPLOT attribute. If CHATT='SHOW', print defaults and current values for all attributes. If CHATT='*', restore default values for all attributes. If VALUE=0, the attribute is set to its default value.

HPLSET : Current values in use			
Parameter	Current value	Default value	Explanation
XSIZ	20.00	20.00	Size along X
YSIZ	20.00	20.00	Size along Y
XMGL	2.00	2.00	X MarGin Left
XMGR	2.00	2.00	X MarGin Right
XLAB	1.40	1.40	distance y axis to LABel
XVAL	.40	.40	distance y axis to axis VALues
XTIC	.30	.30	X axis TICk marks length
YMGL	2.00	2.00	Y MarGin Low
YMGU	2.00	2.00	Y MarGin Up
YLAB	.80	.80	distance x axis to LABel
YVAL	.20	.20	distance x axis to axis VALues
YTIC	.30	.30	Y axis TICk marks length
YNPG	.60	.60	Y position for Number of PaGe
YGTI	1.50	1.50	Y position of Global TItle
YHTI	1.20	1.20	Y position of Histogram TItle
SMGR	.00	.00	Stat MarGin Right (%)
SMGU	.00	.00	Stat MarGin Up (%)
KSIZ	.28	.28	Hershey charact. (HPLKEY) SIZe
GSIZ	.28	.28	Global title SIZe
TSIZ	.28	.28	histogram Title SIZe
ASIZ	.28	.28	Axis label SIZe
CSIZ	.28	.28	Comment and stat SIZe
PSIZ	.28	.28	Page number SIZe
VSIZ	.28	.28	axis Values SIZe
SSIZ	.28	.28	aSterisk SIZe (for functions)
2SIZ	.28	.28	scatter-plot & table char. SIZe
XWIN	2.00	2.00	X space between WINdows
YWIN	2.00	2.00	Y space between WINdows
HMAX	.90	.90	Histogram MAXimum for scale
PASS	1.00	1.00	number of PASS for characters
CSHI	.03	.03	Character SHIft between 2 pass
BARO	.25	.25	BAR histogram Offset (%)
BARW	.50	.50	BAR histogram Width (%)
DASH	.15	.15	length of basic DASHed segment
DMOD	1	1	Dash MODe (or type) for lines
GRID	3	3	GRID line type
DATE	2	2	DATE position
FILE	1	1	FILE name position
STAT	1111	1111	STAT values to be plotted
FIT	101	101	FIT values to be plotted
HTYP	0	0	Histogram fill area TYPe
BTYP	0	0	Box fill area TYPe
PTYP	0	0	Picture fill area TYPe
FTYP	0	0	Function fill area TYPe
HCOL	.00	1.00	Histogram fill area COLor
BCOL	1.00	1.00	Box fill area and shading COLor
PCOL	1	1	Picture fill area COLor
FCOL	1	1	Function fill area COLor
XCOL	1	1	X axis COLor

24.0.180 OPTION [choptn]

`CHOPTN` C "Option name" `D='SHOW'`

Set general plotting options for HPLOT. If CHOPTN='SHOW' print all current and default options. If CHOPTN='*', restore all default options.

HPLOPT : Option values

Current	Default	Alternative	Explanation
VERT	VERT	HORI	VERTical or HORIzontal orientation of paper
NEAH	NEAH	EAH	Error bars And Histogram are plotted (if both are present)
NCHA	NCHA	CHA	scatter plots drawn with dots (NCHA) or 1 char./bin (CHA)
NAST	NAST	AST	functions drawn with (AST) or without (NAST) asterisks
SOFT	SOFT	HARD	SOFTware or HARDware characters are used
NSQR	NSQR	SQR	size is set to the largest square (SQR)
HTIT	HTIT	UTIT	HBOOK TITle (HTIT) or User TITle (UTIT) is printed
TAB	TAB	NTAB	table printed as TABles (TAB) or scatter plots (NTAB)
BOX	BOX	NBOX	a box is (BOX) or is not (NBOX) drawn around picture
NTIC	NTIC	TIC	cross-wires are drawned (TIC) or not (NTIC) on each plot
NSTA	NSTA	STA	STAtistics are printed (STA) or not (NSTA) on each plot
NFIT	NFIT	FIT	FIT parameters are printed or not (NFIT) on each plot
NZFL	NZFL	ZFL	picture is (ZFL) or is not (NZFL) put in Z data base
NPTO	NPTO	PTO	PTO (Please Turn Over) (NPTO)
NBAR	NBAR	BAR	BAR charts for histogram (NBAR)
DVXR	DVXR	DVXI	Integer (DVXI) or Real (DVXR) divisions for X axis
DVYR	DVYR	DVYI	Integer (DVYI) or Real (DVYR) divisions for Y axis
NGRI	NGRI	GRID	GRID or not grid (NGRI) on X and Y axis
NDAT	NDAT	DATE	DATE is printed (DATE) or not (NDAT) on each plot
NFIL	NFIL	FILE	FILE name is printed (FILE) or not (NFIL) on each plot
A4	A4	A0/6	page format for the plotter (A0,A1,A2,A3,A4,A5,A6)
NOPG	NOPG	P	page number is (P) or is not (NOPG) printed
LINY	LINY	LOGY	LINear or LOGarithmic scale in Y
LINX	LINX	LOGX	LINear or LOGarithmic scale in X
LINZ	LINZ	LOGZ	LINear or LOGarithmic scale in Z (Lego or Surface)
NHST	HSTA	HNST	Filling statistics (HSTA)

24.0.181 METAFILE [lun metafl chmeta]

`LUN` I "Logical unit number" `D=0`
`METAFL` I "Metafile ID" `D=0`
`CHMETA` C "Metafile name" `D='␣'`

Set the metafile logical unit and metafile type. This command controls the destination of the subsequent graphics output. Example:

```
LUN =-10 output only on metafile opened on unit 10;
LUN =  0 output only on screen;
LUN = 10 output on both screen and metafile opened on unit 10;
```

Use the command FORTRAN/FILE to open a new file, FORTRAN/CLOSE to close it. Note that PAW opens the file PAW.METAFILE on the unit 10 at initialization time.

```
METAFL=   4 Appendix E GKS.
METAFL=-111 HIGZ/PostScript (Portrait).
METAFL=-112 HIGZ/PostScript (Landscape).
METAFL=-113 HIGZ/Encapsulated PostScript.
METAFL=-114 HIGZ/PostScript Color (Portrait).
METAFL=-115 HIGZ/PostScript Color (Landscape).
METAFL=-777 HIGZ/LaTex Encapsulated.
METAFL=-778 HIGZ/LaTex.
```

The PostScript metafile types have the following format:

```
                 -[Format][Nx][Ny][Type]
Where:
```

[Format] Is an integer between 0 and 99 which defines the format of the

```
paper. For example if Format=3 the paper is in the standard
A3 format. Format=4 and Format=0 are the same and
define an A4 page.
The A0 format is selected by Format=99.
The US format Letter is selected by Format=100.
The US format Legal is selected by Format=200.
The US format Ledger is selected by Format=300.
```

[Nx, Ny] Specify respectively the number of zones on the x and y axis.

```
Nx and Ny are integers between 1 and 9.
```

[Type] Can be equal to:

```
1: Portrait mode with a small margin at the bottom of the page.
2: Landscape mode with a small margin at the bottom of the page.
4: Portrait mode with a large margin at the bottom of the page.
5: Landscape mode with a large margin at the bottom of the page.
   The large margin is useful for some PostScript printers (very
   often for the colour printers) as they need more space to grip
   the paper for mechanical reasons. Note that some PostScript
```

```
            colour printers can also use the so called "special A4" format
            permitting the full usage of the A4 area; in this case larger
            margins are not necessary and {\tt Type}=1 or 2 can be used.
         3: Encapsulated PostScript. This Type permits the generation of
            files which can be included in other documents, for example
            in LaTeX files. Note that with this Type, Nx and Ny must always
            be equal to 1, and Format has no meaning. The size of the
picture
            must be specified by the user via the SIZE command. Therefore
            the workstation type for Encapsulated PostScript is -113. For
            example if the name of an encapsulated PostScript file is
            example.eps, the inclusion of this file into a LaTeX file will
            be possible via (in the LaTeX file):
                    \begin{figure}
                     \epsffile{example.eps}
                     \caption{Example of Encapsulated PostScript in LaTeX.}
                     \label{EXAMPLE}
                    \end{figure}
```

With Type=1,2,4 and 5 the pictures are centered on the page, and the usable area on paper is proportional to the dimensions of A4 format. Examples: -111 or -4111 defines an A4 page not divided. -6322 define an A6 landscape page divided in 3 columns and 2 rows.

```
+-------+-------+-------+
|   1   |   2   |   3   |
+-------+-------+-------+
|   4   |   5   |   6   |
+-------+-------+-------+
```

The first picture will be drawn in the area 1. After each clear the screen, the graphics output will appear in the next area in the order defined above. If a page is filled, a new page is used with the same grid. Note that empty pages are not printed in order to save paper. Ignoring formats smaller than A12, the total number of possible different PostScript workstation types is: 4x9x9x13+1 = 4213 !

24.0.182 WORKSTATION iwkid [chopt iwtyp]

IWKID	I	"Workstation ID"	D=1 Loop
CHOPT	C	"Options"	D='OA'
IWTYP	I	"Workstation type"	D=1

Possible CHOPT values are:

- O Open a new workstation
- C Close a workstation
- A Activate a workstation
- D Deactivate a workstation
- L Give the list of open workstations

To create/delete workstations or change status.

```
IWKID > 0  Do the action specified by CHOPT on the
           workstation identified by IWKID.
IWKID = 0  Do the action specified by CHOPT on all
           workstations.
IWKID < 0  Do the action specified by CHOPT on the
           workstation identified by -IWKID and the
           complementary action on all the others.
```


24.0.183 SLIDE

Invoke the SLIDE package.

Chapter 25

GRAPHICS/MISC

Miscellaneous HPLOT functions.

25.0.184 NEXT

Clear the screen. Initialize a new HIGZ picture if option ZFL or ZFL1 has been selected. Select the Normalization Transformation number 1 (cm).

25.0.185 CLR

Clear the screen.

25.0.186 LOCATE [ntpri chopt wkid]

`NTPRI`	C	"Transformation with highest priority" `D='-1'`
`CHOPT`	C	"Options" `D='R'`
`WKID`	I	"Workstation identifier" `D=1`

Possible `CHOPT` values are:

`R`	Request mode is used to locate the points (default)
`S`	Sample mode is used to locate the points
`I`	Integrate an histogram between 2 bins
`+`	Use the tracking cross (default is cross-hair)
`T`	The output is done on the terminal.

Locate points on the screen using the graphics cursor and output coordinates on terminal. Control is returned when the BREAK (right) mouse button is clicked (or CRTL/E) or when 20 points are located. The optional parameter NTPRI may be specified to locate a point in the specific transformation number NTPRI. NTPRI=-1 (default) means that all the histogram transformation numbers (10, 20, etc.) have priority on transformation number 1. WKID allows to define in which window the locator is performed.

`Note: With the Motif version of PAW the locator is automatically`

invoke when the mouse cursor enter the window.

25.0.187 VLOCATE vecx vecy [chopt ntpri wkid]

`VECX`	C	"Vector for coordinates X"
`VECY`	C	"Vector for coordinates Y"
`CHOPT`	C	"Options" `D='␣'` `Minus`
`NTPRI`	I	"Transformation with highest priority" `D=-1`
`WKID`	I	"Workstation identifier" `D=1`

Possible `CHOPT` values are:

`'␣'`	Use the cross-hair
`+`	Use the tracking cross
`-`	Use the rubber line
`L`	Connect points by a polyline
`P`	Draw the current polymarker at each point
`*`	Draw a * at each point
`S`	Sample mode is used. Allows to see the coordinates of point before clicking

Locate a set of points using the graphics cursor. Return corresponding coordinates in vectors X and Y. If vectors X or Y do not exist, they are automatically created. Control is returned when the point is outside picture limits or when the BREAK (right) mouse button is clicked (or CRTL/E). The optional parameter NTPRI may be specified to locate a point in the specific transformation number NTPRI (see LOCATE). WKID allows to define in which window the locator is performed.

25.0.188 HMOVE

Change the contents of a histogram channel using the cursor. Position the cursor to the channel to be changed, trigger graphics input, position the cursor to the new channel value (a rubber band box is used to visualize the change), trigger graphics input to fix the new value.

Chapter 26

GRAPHICS/VIEWING

To define Normalization transformations. Either automatically (ZONE and SIZE) or 'by hand' (SVP, SWN and SELNT).

26.0.189 ZONE [nx ny ifirst chopt]

NX	I	"Number of divisions along X" D=1
NY	I	"Number of divisions along Y" D=1
IFIRST	I	"First division number" D=1
CHOPT	C	"Option" D='␣'

Possible CHOPT values are:
'␣'

S Redefine zones on current picture

'␣' Define the zones for all subsequent pictures.

Subdivide the picture into NX by NY zones, starting at zone IFIRST (count along X first).

26.0.190 SIZE [xsize ysize]

XSIZE	R	"Size along X" D=20.
YSIZE	R	"Size along Y" D=20.

Set the size of the picture. On the terminal, the pictures will have the ratio YSIZE/XSIZE, and, if a metafile is produced, pictures will be YSIZE by XSIZE cm. This command sets the parameters for the normalization transformation number 1 to [0-XSIZE], [0-YSIZE].

26.0.191 SVP nt x1 x2 y1 y2

NT	I	"Normalization transformation number"
X1	R	"Low X of viewport in NDC" D=0 R=0:1
X2	R	"High X of viewport in NDC" D=1 R=0:1
Y1	R	"Low Y of viewport in NDC" D=0 R=0:1
Y2	R	"High Y of viewport in NDC" D=1 R=0:1

Set the viewport of the normalization transformation NT in the Normalized Device Coordinates (NDC). Note that the command SELNT should be invoke in order to validate the viewport parameters.

26.0.192 SWN nt x1 x2 y1 y2

NT	I	"Normalize transformation number"
X1	R	"Low X of window in WC" D=0
X2	R	"High X of window in WC" D=20
Y1	R	"Low Y of window in WC" D=0
Y2	R	"High Y of window in WC" D=20

Set the window of the normalization transformation NT in World Coordinates (WC). Note that the command SELNT should be invoke in order to validate the window parameters. For example:

```
PAW > Nul 0 1 -1 1       | Draw an empty frame (0,1)x(-1,1)
PAW > Line 0 0 1 1       | Draw a line in (0,1)x(-1,1)
PAW > Swn 10 0 10 0 10   | Change the coordinates to (0,10)x(0,10)
PAW > Selnt 10           | Activate the coordinates (0,10)x(0,10)
PAW > Line 0 0 1 1       | Draw a line in (0,10)x(0,10)
```

26.0.193 SELNT nt

`NT` I "Normalization transformation number"
Select a normalization transformation number.

```
   If ZONE 2 2 is active , then:       If ZONE 1 1 is active, then:
  +------------------------------+    +------------------------------+
  |                              |    |                              |
  |  +----------+  +----------+  |    |  +------------------------+  |
  |  |          |  |          |  |    |  |                        |  |
  |  |  NT=10   |  |  NT=20   |  |    |  |                        |  |
  |  |          |  |          |  |    |  |                        |  |
  |  +----------+  +----------+  |    |  |                        |  |
  |                              |    |  |         NT=10          |  |
  |  +----------+  +----------+  |    |  |                        |  |
  |  |          |  |          |  |    |  |                        |  |
  |  |  NT=30   |  |  NT=40   |  |    |  |                        |  |
  |  |          |  |          |  |    |  |                        |  |
  |  +----------+  +----------+  |    |  |                        |  |
  |                              |    |  +------------------------+  |
  |            NT=1              |    |            NT=1              |
  +------------------------------+    +------------------------------+
```


Chapter 27

GRAPHICS/PRIMITIVES

Call HIGZ drawing primitives

27.0.194 PLINE n x y

- `N` I "Number of points"
- `X` C "Vector name for X coordinates"
- `Y` C "Vector name for Y coordinates"

Draw a polyline of N points X,Y in the current Normalization transformation. The PLINE attributes can be changed with the command SET.
Example:

```
   SET * ; OPT *          | Reset the defaults
   NUL -1 1 0 1           | Draw a frame (cf HELP NULL)
* Create vector X and Y (cf HELP SIGMA)
   SIGMA X=ARRAY(100,-1#1)
   SIGMA Y=X*X
   SET PLCI 4             | The line color is blue
   SET LWID 6             | The line width is 6
   SET LTYP 2             | The line type is dashed
   PLINE 100 X Y          | Draw a 100 points line
```

27.0.195 LINE x1 y1 x2 y2

- `X1` R "X first coordinate"
- `Y1` R "Y first coordinate"
- `X2` R "X second coordinate"
- `Y2` R "Y second coordinate"

Draw a line connecting points (X1,Y1) and (X2,Y2) in the current Normalization transformation. This command is kept for backward compatibility. It has a reverse calling sequence compare to BOX or ARROW and it doesn't take LOG scales into account. It is recommended to use DLINE instead. The LINE attributes can be changed with the command SET.
Example:

```
   SET * ; OPT *          | Reset the defaults
   NUL 0 5 0 5            | Draw a frame (cf HELP NULL)
   SET PLCI 2             | The line color is red
   SET LWID 6             | The line width is 6
   SET LTYP 3             | The line type is dotted
   LINE 0 0 5 5           | Draw a line
```

27.0.196 DLINE x1 x2 y1 y2

- `X1` R "X first coordinate"
- `X2` R "X second coordinate"
- `Y1` R "Y first coordinate"
- `Y2` R "Y second coordinate"

Draw a line connecting points (X1,Y1) and (X2,Y2) in the current Normalization transformation taking care of logarithmic scales. The DLINE attributes can be changed with the command SET.
Example:

```
   SET * ; OPT *          | Reset the defaults
   OPTION LOGY            | Log scale on the Y axis.
   NUL 0 5 1 100          | Draw a frame (cf HELP NULL)
   SET PLCI 2             | The line color is red
   SET LWID 6             | The line width is 6
   SET LTYP 1             | The line type is solid
   DLINE 0 5 1 10         | Draw a line
```


27.0.197 FAREA n x y

- `N` I "Number of points"
- `X` C "Vector name for X coordinates"
- `Y` C "Vector name for Y coordinates"

Fill the area defined by the N points X,Y in the current Normalization transformation. The FAREA attributes can be changed with the command SET.
Example:

```
   SET * ; OPT *          | Reset the defaults
   NUL -1.1 1.1 -1.1 1.1  | Draw a frame (cf HELP NULL)
* Create vector X and Y (cf HELP SIGMA)
   SIGMA X=ARRAY(100,-3.14#3.14)
   SIGMA Y=SIN(X)*COS(X)
   SIGMA X=COS(X)
   SET FACI 2             | The fill area color is red
   SET FAIS 1             | The fill area interior style is solid
   FAREA 100 X Y          | Draw a 100 points line
   SET FACI 1             | The fill area color is black
   SET FAIS 0             | The fill area interior style is hollow
   FAREA 100 X Y          | Draw a 100 points line
   SET FAIS 3             | The fill area interior style is hatched
   SET FASI 245           | Defines the type of hatches
   FAREA 100 X Y          | Draw a 100 points line
```

27.0.198 PMARKER n x y

- `N` I "Number of points"
- `X` C "Vector name for X coordinates"
- `Y` C "Vector name for Y coordinates"

Draw polymarkers at the N points X,Y in the current Normalization transformation. The PMARKER attributes can be changed with the command SET.
Example:

```
   SET * ; OPT *          | Reset the defaults
   NUL -3.2 3.2 -1 1      | Draw a frame (cf HELP NULL)
* Create vector X and Y (cf HELP SIGMA)
   SIGMA X=ARRAY(100,-3.14#3.14)
   SIGMA Y=SIN(X)*COS(X)
   SET PMCI 6             | The marker color is magenta
   SET MTYP 3             | The marker type is *
   SET MSCF 2             | The marker size is 2
   PMARKER 100 X Y        | Draw a 100 points polymarker
```

27.0.199 BOX x1 x2 y1 y2

X1	R	"X coordinate of first corner"
X2	R	"X coordinate of second corner"
Y1	R	"Y coordinate of first corner"
Y2	R	"Y coordinate of second corner"

Draw and fill a box with the current fill area and line attributes. Use the current Normalization transformation. The BOX attributes can be changed with the command SET.
Example:

```
SET * ; OPT *          | Reset the defaults
NULL 0 10 0 10         | Draw a frame (cf HELP NULL)
SET FAIS 0             | Fill area interior style hollow
BOX 1 3 1 3            | Draw a box
SET FAIS 1             | Fill area interior style solid
BOX 1 3 3 5            | Draw a box
SET FAIS 3             | Fill area interior style hatched
SET FASI 245           | Changes the type of hatches
BOX 1 3 5 7            | Draw a box
SET FASI 3             | Changes the type of hatches
BOX 3 5 5 7            | Draw a box
SET BORD 1             | The border is requested
SET PLCI 2             | Line color is red
SET FASI 4             | Changes the type of hatches
BOX 5 7 5 7            | Draw a box
```

27.0.200 FBOX x1 x2 y1 y2 x3 x4 y3 y4

X1	R	"X coord of first corner of ext box"
X2	R	"X coord of second corner of ext box"
Y1	R	"Y coord of first corner of ext box"
Y2	R	"Y coord of second corner of ext box"
X3	R	"X coord of first corner of int box"
X4	R	"X coord of second corner of int box"
Y3	R	"Y coord of first corner of int box"
Y4	R	"Y coord of second corner of int box"

Draw and fill a frame (2 nested boxes) with the current fill area and line attributes. Use the current Normalization transformation. The FBOX attributes can be changed with the command SET.
Example:

```
SET * ; OPT *          | Reset the defaults
NULL 0 10 0 10         | Draw a frame (cf HELP NULL)
SET FAIS 3             | Fill area interior style hatched
SET FASI 3             | Changes the type of hatches
SET FACI 2             | Fill are color is red
SET PLCI 4             | Line color is blue
SET LWID 8             | The line width is 8
SET BORD 1             | The border is requested
FBOX 1 9 1 9 3 7 3 7   | Draw a frame box
```


27.0.201 ARROW x1 x2 y1 y2 [size]

X1	R	"X coordinate of start point"
X2	R	"X coordinate of end point"
Y1	R	"Y coordinate of start point"
Y2	R	"Y coordinate of end point"
SIZE	R	"Arrow size" D=0.4

Draw an arrow Use the current Normalization transformation. The ARROW attributes can be changed with the command SET. ARROW and LINE attributes are the same.

```
(X1,Y1) ----> (X2,Y2) if SIZE>0.
(X1,Y1) <---> (X2,Y2) if SIZE<0.
```

Example:

```
SET * ; OPT *          | Reset the defaults
NULL 0 10 0 7          | Draw a frame (cf HELP NULL)
ARROW 1 9 1 1 .2       | Draw a simple arrow (left to right)
ARROW 9 1 2 2 .4       | Draw a simple arrow (right to left)
ARROW 1 9 3 3 -.8      | Draw a double arrow
SET PLCI 2             | Arrow color is red
ARROW 1 9 4 4 -.8      | Draw a double arrow
SET LWID 8             | Arrow line width is 8
ARROW 1 9 5 5 -.8      | Draw a double arrow
SET LTYP 3             | Arrow line type is dotted
ARROW 1 9 6 6 -.8      | Draw a double arrow
```

27.0.202 HELIX [x1 y1 x2 y2 r wi phi]

X1	R	"X coordinate of the begin of helix" D=0.
Y1	R	"Y coordinate of the begin of helix" D=0.
X2	R	"X coordinate of the end of helix" D=10.
Y2	R	"Y coordinate of the end of helix" D=10.
R	R	"Radius of helix" D=.3
WI	R	"Number of turns " D=1.
PHI	R	"Projection angle " D=15.

Draw an helix with the current line attributes. Use the current Normalization transformation. Feynman graph: gluon phi = 30, photon phi = 0.
Example:

```
SET * ; OPT *            | Reset the defaults
NUL 0 10 0 10 'AB'       | Draw a frame (cf HELP NULL)
HELIX 1 1 3 3 ! 10 !     | Draw an helix
SET LWID 8               | Helix line width is 8
HELIX 3 3 7 7 1  5 !     | Draw an helix
SET PLCI 2               | Arrow color is red
SET LTYP 2               | Helix line type is dashed
HELIX 7 7 10 10 .2 5 10  | Draw an helix
```

27.0.203 ARCHELIX [x1 y1 x2 y2 r wi phi rl]

X1	R	"X coordinate of the begin of helix" D=0.
Y1	R	"Y coordinate of the begin of helix" D=0.
X2	R	"X coordinate of the end of helix" D=10.
Y2	R	"Y coordinate of the end of helix" D=10.
R	R	"Radius of helix" D=.3
WI	R	"Number of turns " D=1.
PHI	R	"Projection angle " D=30.
RL	R	"Radius of loop " D=15.

Draw an archelix with the current line attributes. Use the current Normalization transformation. Feynman graph: gluon phi = 30, photon phi = 0.
Example:

```
SET * ; OPT *                | Reset the defaults
NUL 0 10 0 10 'AB'           | Draw a frame (cf HELP NULL)
ARCHELIX 1 1 3 3 ! 9 ! 1     | Draw an helix
SET LWID 8                   | Helix line width is 8
ARCHELIX 3 3 7 7 ! 9 ! 1     | Draw an helix
SET PLCI 2                   | Arrow color is red
SET LTYP 2                   | Helix line type is dashed
ARCHELIX 7 7 10 10 ! 9 ! 3   | Draw an helix
```

27.0.204 ARLINE [x1 y1 x2 y2 h]

X1	R	"X coordinate of the begin" D=0.
Y1	R	"Y coordinate of the begin" D=0.
X2	R	"X coordinate of the end" D=10.
Y2	R	"Y coordinate of the end" D=10.
H	R	"arrow size" D=.5

Draw a line with arrow in middle (fermion line) with the current line and fill area attributes. Use the current Normalization transformation.
Example:

```
SET * ; OPT *          | Reset the defaults
NULL 0 10 0 6          | Draw a frame (cf HELP NULL)
ARLINE 1 1 9 1 .2      | Draw a arrow line (left to right)
ARLINE 9 2 1 2 .4      | Draw a arrow line (right to left)
SET PLCI 2             | Arrow color is red
SET FAIS 1             | Fill area interior style solid
ARLINE 9 3 1 3 .4      | Draw a arrow line (right to left)
SET LWID 8             | Arrow line width is 8
SET FACI 4             | The fill area color is blue
ARLINE 9 4 1 4 .4      | Draw a arrow line (right to left)
SET LTYP 3             | Arrow line type is dotted
ARLINE 9 5 1 5 .4      | Draw a arrow line (right to left)
```


27.0.205 FPOINT [x y r]

X	R	"X " D=0.
Y	R	"Y " D=0.
R	R	"Radius " D=.5

Draw a filled point (vertex) with the current fill area attributes. Use the current Normalization transformation.
Example:

```
SET * ; OPT *         | Reset the defaults
NULL 0 10 0 10        | Draw a frame (cf HELP NULL)
SET FAIS 1            | Fill area interior style solid
FPOINT 5 1 .1         | Draw a filled point
FPOINT 5 3 .2         | Draw a filled point
FPOINT 5 5 .3         | Draw a filled point
SET FACI 4            | The fill area color is blue
FPOINT 5 7 .4         | Draw a filled point
FPOINT 5 9 .5         | Draw a filled point
```

27.0.206 AXIS x0 x1 y0 y1 wmin wmax ndiv [chopt]

X0	R	"X axis origin in WC"
X1	R	"X end axis in WC"
Y0	R	"Y axis origin in WC"
Y1	R	"Y end axis in WC"
WMIN	R	"Lowest value for labels"
WMAX	R	"Highest value for labels"
NDIV	I	"Number of divisions" D=510
CHOPT	C	"Options" D='␣' Minus

Possible CHOPT values are:

'␣'	Draw an axis with default values.
G	Logarithmic scale, default is linear.
B	Blank axis. Useful to superpose axis.
U	Unlabeled axis, default is labeled.
+	Tick marks are drawn on Positive side. (default)
-	Tick marks are drawn on the negative side.
=	Tick marks are drawn on Equal side
P	Labels are drawn Parallel to the axis
O	Labels are drawn Orthogonal to the axis (Top to Down).
0	Labels are drawn Orthogonal to the axis (Down to Top).
R	labels are Right adjusted on tick mark.
L	labels are Left adjusted on tick mark.
C	labels are Centered on tick mark.
M	In the Middle of the divisions.
Y	Direction of labels DOWN . Default is RIGHT
.	Dot obligatory
T	Alphanumeric labels .
S	Tick marks Size
H	Labels Height
D	Distance labels-axis
N	No bining optimization
I	Integer labeling

Draw an axis in the current Normalization transformation.

```
      NDIV=N1 + 100*N2 + 10000*N3
      N1, N2, N3 = Number of 1st, 2nd, 3rd divisions respectively, eg:.
          NDIV=0 --> no tick marks.
          NDIV=2 --> 2 divisions, one tick mark in the middle
                   of the axis.
Orientation of tick marks on axis: Tick marks are normally drawn
on the positive side of the axis.However, if X0=X1, then Negative .
      CHOPT='+': tick marks are drawn on Positive side. (default)
      CHOPT='-': tick marks are drawn on the negative side.
      i.e: '+-' --> tick marks are drawn on both sides of the axis.
Position of labels on axis: Labels are normally drawn on side
opposite to tick marks.However:
      CHOPT= '='   on Equal side
Orientation of labels on axis: Labels are normally drawn
parallel to the axis. However if X0=X1, then Orthogonal
                              if Y0=Y1, then Parallel
      CHOPT= 'P' : Parallel to the axis
      CHOPT= 'O' : Orthogonal to the axis (Top to Down).
      CHOPT= '0' : Orthogonal to the axis (Down to Top).
Position of labels on tick marks: Labels are centered on
tick marks. However , if X0=X1, then they are right adjusted.
      CHOPT='R': labels are Right adjusted on tick mark.
```



```
                 (default is centered)
      CHOPT='L': labels are Left adjusted on tick mark.
      CHOPT='C': labels are Centered on tick mark.
      CHOPT='M': In the Middle of the divisions.
Direction of labels: Default is RIGHT
      CHOPT='Y':   Down
Format of labels: Blank characters are stripped, and then the
label is correctly aligned. The dot,if last character of the
string, is also stripped, unless
      CHOPT='.'     Dot obligatory
In the following, we have some parameters, like
tick marks length and characters height (in percentage
of the length of the axis).The default values are as follows:
   Primary tick marks: 3.0 %
   Secondary tick marks: 1.5 %
   Third order tick marks: .75 %
   Characters height for labels: 2%
   Characters spacing (related to height): 40%
   Labels offset: 4.0 %
Type of labels: Labels are normally numeric . However, alphanumeric
labels can be drawn (see command LABEL).
      CHOPT='T':   Alphanumeric labels .
Intrinsic parameters: These values can be changed with the command
SET. The default value is used unless the corresponding option is
selected by CHOPT:
      CHOPT='D' The distance between the labels and the axis
                (the offset) is given by the preceding command
                SET with the parameter LAOF.
      CHOPT='H' The size (height) of the labels is given by the
                preceding command SET with the parameter LASI.
      CHOPT='S' The size of the tick marks is given by the preceding
                command SET with the parameter TMSI.
Axis bining optimization: By default the axis bining is optimized .
      CHOPT='N': No bining optimization
      CHOPT='I': Integer labeling
```

Example:

```
SET * ; OPT *                            | Reset the defaults
NUL 0 12 0 12 'A'                        | Draw a frame (cf HELP NULL)
AXIS 1  11  1  1  0  100  510 'A'        | Axis with arrow
AXIS 1  11  3  3  1  10000  510 'G'      | LOG axis
LABEL 1 11 a b c d e f g h i j k         | define alphanumeric labels
AXIS 1  11  5  5  0  12  11 'NATY'       | alphanumeric labeling
AXIS 1  11  6  6  -100  0  510 'A'
AXIS 11  1  7  7  -100  0  810 'A+-'     | Double side tick marks
AXIS 1  11  8  11  0  1234567  615 'A'   | exponent is required
```

27.0.207 ARC x1 y1 r1 [r2 phimin phimax]

X1	R	"X coordinate of centre"
Y1	R	"Y coordinate of centre"
R1	R	"Inner radius"
R2	R	"Outer radius" D=-1.
PHIMIN	R	"Minimum angle" D=0.
PHIMAX	R	"Maximum angle" D=360.

Draw an arc of circle with the current fill area and line attributes. Use the current Normalization transformation. If R1 is not equal to R2 the area between the two arcs of radius R1 and R2 is filled according to the current fill area attributes. The border is never drawn unless the interior style is hollow or the command SET BORD 1 has been called. If R1 is equal to R2 a polyline is drawn.
Example:

```
SET * ; OPT *           | Reset the defaults
NULL 0 20 0 20 'AB'     | Draw a frame (cf HELP NULL)
SET PLCI 2              | Line color is red
SET LWID 6              | Line width is 6
ARC  5  5 4 4 ! !       | Draw an circle
ARC  5 15 4 4 30 260    | Draw an arc of circle
SET FAIS 3              | Fill area with hatches
SET FASI 3              | Type of hatches
ARC 15 15 1 4 ! !       | Draw an arc
SET BORD 1              | Border is requested
ARC 15  5 1 4 30 !      | Draw an arc
```


27.0.208 PIE x0 y0 radius n values [chopt iao ias iac]

X0	R	"X coordinate of centre of the pie"
Y0	R	"Y coordinate of centre of the pie"
RADIUS	R	"Radius of the pie chart"
N	I	"Number of values"
VALUES	C	"Vector name for N values"
CHOPT	C	"Options" D='␣'
IAO	C	"Name of vector with offsets" D='␣'
IAS	C	"Name of vector with styles" D='␣'
IAC	C	"Name of vector with colors" D='␣'

Possible CHOPT values are:

'␣'	Draw a Pie Chart with default values.
C	Colours array is present.
L	Alphanumeric labels are required.
O	Offset array is present.
N	The label of each slice will be the corresponding numeric value in array VALUES.
P	The label of each slice will be in expressed in percentage.
S	Style array is present.
H	Force the labels size to be the current character height. Without this option the labels size is computed automatically.
R	Draw the labels aligned on the radius of each slice.

Draw a pie chart in the current Normalization transformation.
Example:

```
   SET * ; OPT *                                 | Reset the defaults
   NULL 0 20 0 20 'AB'                           | Draw a frame
   LABEL 1 5 'Lab1' 'Lab2' 'Lab3' 'Lab4' 'Lab5'  | define labels
* Initialize vectors
   V/CRE VWS(5) R 28.3 18.6 16.9 13.5 22.7
   V/CRE OFFSET(5)  R 2*0. 2*20. 0.
   V/CRE COLOUR(5)  R 2 3 4 5 6
   SET FAIS 1                                    | Fill solid
   SET BORD 1                                    | Draw the border
   PIE 10. 10. 7.  5 VWS 'L' OFFSET ! COLOUR     | Draw the pie chart
```

27.0.209 TEXT x y text size [angle chopt]

X	R	"X coordinate"
Y	R	"Y coordinate"
TEXT	C	"Text to be drawn"
SIZE	R	"Text size" D=0.3
ANGLE	R	"Comment angle" D=0
CHOPT	C	"Justification option" D='L'

Possible CHOPT values are:

- L Text is Left justified.
- C Text is Centered.
- R Text is Right justified.

Draw text at position X,Y in the current normalization transformation using the software font IGTEXT. SIZE is always given in centimeters (as defined by the command SIZE). A boldface effect can be obtained using the parameters PASS and CSHI of the command SET. The text color can be changed by SET TXCI.
Example:

```
SET * ; OPT *                                    | Reset the defaults
NULL 0 10 0 10                                   | Draw a frame
TEXT 5 1 'Left justified' .5 0. L
TEXT 5 2 'Centered' .5 0. C
TEXT 5 3 'Right justified' .5 0. R
TEXT 5 4 '-- 30 degrees' .5 30. L
TEXT 5 4 '-- 60 degrees' .5 60. L
TEXT 5 4 '-- 90 degrees' .5 90. L
TEXT 5 4 '-- 120 degrees' .5 120. L
TEXT 5 4 '-- 150 degrees' .5 150. L
TEXT 5 8 'Some Greek ... [a, b, c, d]' .5 0. C
Set PASS 7                                       | Number of passes
TEXT 5 9 'Bold TEXT' .5 0. C
```


27.0.210 ITX x y text

X	R	"X coordinate"
Y	R	"Y coordinate"
TEXT	C	"Text to be drawn"

Draw text at position X,Y in the current Normalization transformation, using the current font parameters. The font and the precision can be changed by SET TXFP. The character size can be changed by SET CHHE. The text color can be changed by SET TXCI. The text orientation can be changed with SET TXAL. The text angle can be changed by SET TANG.
Example:

```
SET * ; OPT *                        | Reset the defaults
NULL 0 10 0 6                        | Draw a frame
SET TXFP -20                         | Times bold
SET CHHE .5                          | Text size 0.5 cm
SET TXAL 10                          | Horizontal align. Left
ITX 5 1 'Left justified'
SET TXAL 20                          | Horizontal align. Center
ITX 5 2 'Centered'
SET TXAL 30                          | Horizontal align. Right
ITX 5 3 'Right justified'
SET TXAL 12                          | Vertical align. Top
ITX .2 4 'Top justified'
SET TXAL 13                          | Vertical align. Middle
ITX .2 5 'Middle justified'
SET TXAL  0                          | Default align.
SET TANG 30                          | Angle 30 degrees
ITX 5 4 '-- 30 degrees --'
```

27.0.211 LABELS labnum nlabs chlabs

LABNUM	I	"Label identifier" D=1 R=1:9
NLABS	I	"Number of labels" D=0 R=0:50
CHLABS	C	"List of labels" D='␣' Vararg

Define a list of alphanumeric labels to be used by subsequent commands such as PIE and AXIS. The position of the labels on the axis may be changed with SET NDVX (NDVY).
Example:

```
SET * ; OPT *                  | Reset the defaults
ZONE 1 3
LABEL 1 3 AAAAA BBBBB CCCCC    | Define labels
SET NDVX 3.15                  | 3 div, lab id 1, 5=center on bin
NULL 0 10 0 1                  | Draw a frame
SET NDVX 3.11                  | 3 div, lab id 1, 1=center on tick
NULL 0 10 0 1                  | Draw a frame
SET NDVX 3.18                  | 3 div, lab id 1, 8=bottom -> up
NULL 0 10 0 1                  | Draw a frame
```

A full description of the possible alignments is given in the PAW manual (see NDVX in the index).

27.0.212 PAVE x1 x2 y1 y2 [dz isbox isfram chopt]

X1	R	"X bottom left corner of box"
X2	R	"X top right corner of box"
Y1	R	"Y bottom left corner of box"
Y2	R	"Y top right corner of box"
DZ	R	"Box width" D=0.4
ISBOX	I	"Box style" D=0
ISFRAM	I	"Frame style" D=5
CHOPT	C	"Option" D='TR'

Possible CHOPT values are:

- TR Top and Right frame are drawn
- TL Top and Left frame
- BR Bottom and Right frame
- BL Bottom and Left frame
- L Left frame only
- R Right frame only
- T- Top frame only pointing left
- B- Bottom frame only pointing left
- S Shadow mode
- K Key mode

Draw a paving-block (box with 3D effect). ISBOX (ISFRAM) may be 1000+ICOLOR where ICOLOR is the color index of the box (frame), otherwise the style index. If ISBOX (ISFRAM) = 0, only the box contour is drawn with the current polyline attributes.
Example:

```
SET * ; OPT *                         | Reset the defaults
NULL 0 10 0 10                        | Draw a frame
PAVE 1 4 1 4 ! ! 1001 CHOPT=TRS
PAVE 5 9 1 4 ! ! 1001 CHOPT=BLS
PAVE 1 4 5 9 ! !    3 CHOPT=TR
PAVE 5 9 5 9 ! !    3 CHOPT=BL
```


27.0.213 HIST n x y [chopt]

N	I	"Number of values"
X	C	"Vector name for X coordinates"
Y	C	"Vector name for Y coordinates"
CHOPT	C	"Options" D='AHW'

Possible CHOPT values are:

- A X and Y axes are drawn (default).
- H An histogram is drawn as a contour (default).
- W The Window/Viewport parameters are automatically computed from the X and Y values (default).
- R The histogram is Rotated, i.e. the values in X are used for the ordinate and the values in Y for the abscissa (default is the contrary). If option R is selected (and option 'N' is not selected), the user must give: 2 values for Y (Y(1)=YMIN and Y(2)=YMAX) N values for X, one for each bin. Otherwise the user must give: N values for Y, one for each bin. 2 values for X (X(1)=XMIN and X(2)=XMAX) If option 'N' is selected see below.
- N Non equidistant bins (default is equidistant). The arrays X and Y must be dimensioned as follows: If option R is not selected (default) then give: (N+1) values for X (limits of bins). N values for Y, one for each bin. Otherwise give: (N+1) values for Y (limits of bins). N values for X, one for each bin.
- F The area delimited by the histogram is filled according to the fill area interior style and the fill area style index or colour index. Contour is not drawn unless CHOPT='H' is also selected.
- C A Smooth curve is drawn across points at the centre of each bin of the histogram.
- L A straight Line is drawn across points at the centre of each bin of the histogram.
- * A star is plotted at the center of each bin of the histogram.
- P Idem as '*' but with the current marker.
- B A Bar chart with equidistant bins is drawn as fill areas. (Contours are drawn). The bar origin and the bar width can be controlled by the routine SET using the options BARO and BARW respectively.

Draw an histogram defined by arrays X and Y. The number of components needed in vectors X and/or in Y may be dependent upon the value of CHOPT (see options 'R' and 'N'). To set Log scales in X and/or Y, use OPT LOGX/LOGY. Note that when an option is specified, it is also necessary to specify the options 'W' or 'HW' in order to start a new zone or/and draw the axes.
Example

```
   SET * ; OPT *                          | Reset the defaults
   Zone 1 2
* This command needs vectors
   V/CREATE Y(10) r 1 2 3 4 5 5 4 3 2 1
   V/CREATE X(11) r 1 2 4 6 8 10 15 16 20 21 30
   HIST 10 X Y 'WH'                       | Equidistant bins
   HIST 10 X Y 'HWN'                      | Non Equidistant bins
```

27.0.214 GRAPH n x y [chopt]

N	I	"Number of values"
X	C	"Vector name for X coordinates"
Y	C	"Vector name for Y coordinates"
CHOPT	C	"Options" D='ALW'

Possible CHOPT values are:

- **A** X and Y axes are drawn (default).
- **L** Every point is connected with a straight line. (default)
- **W** The Window/Viewport parameters are automatically computed from the X and Y values (default).
- **C** The values in Y are plotted in the form of a smooth curve. A Spline approximation algorithm is used.
- **F** A fill area is drawn. If the option 'CF' is used the contour of the fill area is smooth. The border of the fill area is drawn if the command SET BORD 1 has been typed. The fill area type may be changed via the SET parameters FASI and FASI
- **R** The graph is Rotated, i.e. the values in X are used for the ordinate and the values in Y for the abscissa (default is the contrary).
- **B** A Bar chart with equidistant bins is drawn as fill areas. (Contours are drawn). The bar origin and the bar width can be controlled by the routine SET using the options BARO and BARW respectively.
- ***** A star is plotted at every point.
- **P** A marker is plotted at every point, according to current marker type and polymarker colour index.

Draw a curve through a set of points. To set Log scales in X and/or Y, use OPT LOGX/LOGY. Note that when an option is specified, it is also necessary to specify the options 'AW' or 'ALW' in order to start a new zone or/and draw the axes.
Example

```
   SET * ; OPT *                           | Reset the defaults
   ZONE 1 2
 * This command needs vectors
   V/CREATE Y(10) r 1 2 3 4 5 5 4 3 2 1
   V/CREATE X(11) r 1 2 4 6 8 10 15 16 20 21
   GRAPH 10 X Y 'WC*L'                     | Draw an "open" graph
   SET FAIS 3                              | Interior style: hatched
   SET FASI 245                            | Define hatches type
   SET BORD 1                              | Border requested
   NULL 0 22 0 6                           | define new scales
   GRAPH 10 X Y 'CF*'                      | Draw an "closed" graph
```


Chapter 28

GRAPHICS/ATTRIBUTES

Change HIGZ attributes.

28.0.215 COLOR_TABLE icol [red green blue]

ICOL	I	"Color Index" D=1
RED	R	"Weight of red" D=0. R=0.:1.
GREEN	R	"Weight of green" D=0. R=0.:1.
BLUE	R	"Weight of blue" D=0. R=0.:1.

Define the color ICOL.

28.0.216 PALETTE palnb [nel list]

PALNB	I	"Palette number" D=0 R=0:9
NEL	I	"Number of elements in the palette" D=0 R=0:50
LIST	I	"List of the palette elements" D=0

Define a palette of attributes. The palette number is used in the command SET. The command SET HCOL 0.1 defines the palette number 1 as colour indices used by the command LEGO in case of stacked lego plots and plotting of SURFACE with options 1 or 2, LEGO with option 2 and CONTOUR with option 3.
By default the palettes are initialized with 6 elements: 2,3,4,5,6,7.
If the number of elements (NEL) is equal to 0 (default), the palette is filled automatically according to the number of colours defined with the command SET NCOL:

```
a) If NCOL is smaller or equal to 8, the palette is filled with a
   subset of the 8 basic colours.
   Examples:
   PAW > SET NCOL 8         | Define the number of colours
   PAW > PALETTE 1          | The palette 1 is filled with
                            | 8 elements: 0,5,7,3,6,2,4,1
   PAW > SET NCOL 4         | Define the number of colours
   PAW > PALETTE 1          | The palette 1 is filled with
                            | 4 elements: 0,5,7,3

b) If NCOL is greater than 8, the palette is filled
   with colours varying continuously from blue to red. This is
   called a "geographical palette".
   Examples:
   PAW > SET NCOL 16      | Define the number of colours
   PAW > PALETTE 1          | Fill palette 1 with 8 elements
                            | (8,9,10,11,12,13,14,15) varying
                            | continuously from blue to red
   Note that after the command SET NCOL, the color indices from
   8 to NCOL are set with gray levels. The command PALETTE 1
   reset the same indices with a "geographical palette" varying
   continuously from blue to red.
```


Chapter 29

GRAPHICS/HPLOT

Draw various HPLOT objects (symbols, errors, key, etc.).

29.0.217 SYMBOLS x y n [isymb ssize]

`X`	C	"Vector of X coordinates"
`Y`	C	"Vector of Y coordinates"
`N`	I	"Number of points" `D=1`
`ISYMB`	I	"Symbol number" `D=24`
`SSIZE`	R	"Symbol size" `D=0.28`

Draw the same symbol at several points x,y in the current normalization transformation.

29.0.218 ERRORS x y ex ey n [isymb ssize chopt]

`X`	C	"Vector of X coordinates"
`Y`	C	"Vector of Y coordinates"
`EX`	C	"Vector of X error bars"
`EY`	C	"Vector of Y error bars"
`N`	I	"Number of points" `D=1`
`ISYMB`	I	"Symbol number" `D=24`
`SSIZE`	R	"Symbol size" `D=0.28`
`CHOPT`	C	"Options" `D='␣'`

Possible `CHOPT` values are:

- `'␣'` Coordinates are expressed in histogram coordinates (of the last drawn histogram). Error bars are drawn.
- `C` Coordinates are expressed in centimeters.
- `W` A new window is defined and axis are drawn.
- `1` Draw small lines at the end of the error bars.
- `2` Draw error rectangles.
- `3` Draw a filled area through the end points of the vertical error bars.
- `4` Draw a smoothed filled area through the end points of the vertical error bars.
- `0` Turn off the symbols clipping.

Draw (according to the CHOPT value) a series of points using a symbol and error bars in horizontal and vertical direction in the current normalization transformation.
By default, the symbols are not drawn if they are on the edges of the plot: the option '0' allows to turn off this symbols clipping.
With Option 1, the size of the tick marks at the end of the error bars is the minimum of the error on X and the marker size.
If ISYMB = 0 or SSIZE = 0. no symbol is drawn.
Note that the options can be cumulated.

29.0.219 AERRORS x y exl exu eyl eyu n [isymb ssize chopt]

`X`	C	"Vector of X coordinates"
`Y`	C	"Vector of Y coordinates"
`EXL`	C	"Vector of X error bars (Low)"
`EXU`	C	"Vector of X error bars (Up)"
`EYL`	C	"Vector of Y error bars (Low)"
`EYU`	C	"Vector of Y error bars (Up)"
`N`	I	"Number of points" `D=1`
`ISYMB`	I	"Symbol number" `D=24`
`SSIZE`	R	"Symbol size" `D=0.28`
`CHOPT`	C	"Options" `D='␣'`

Possible `CHOPT` values are:

- `'␣'` Coordinates are expressed in histogram coordinates (of the last drawn histogram). Error bars are drawn.
- `C` Coordinates are expressed in centimeters.
- `W` A new window is defined and axis are drawn.
- `1` Draw small lines at the end of the error bars.
- `2` Draw error rectangles.
- `3` Draw a filled area through the end points of the vertical error bars.
- `4` Draw a smoothed filled area through the end points of the vertical error bars.
- `0` Turn off the symbols clipping.

Draw (according to the CHOPT value) a series of points using a symbol and asymmetric error bars in horizontal and vertical direction in the current normalization transformation.
By default, the symbols are not drawn if they are on the edges of the plot: the option '0' allows to turn off this symbols clipping.
With Option 1, the size of the tick marks at the end of the error bars is the minimum of the error on X and the marker size.
If ISYMB = 0 or SSIZE = 0. no symbol is drawn.
Note that the options can be cumulated.

29.0.220 KEY x y [isymb text]

`X`	R	"X coordinate of comment"
`Y`	R	"Y coordinate of comment"
`ISYMB`	I	"Symbol number" `D=24`
`TEXT`	C	"Legend" `D='␣'`

Draw one symbol and its explanation (legend) at a point x,y in the current normalization transformation.

29.0.221 TICKS [chopt xval yval]

CHOPT	C	"Options"	D='␣'
XVAL	R	"X position"	D=1.E30
YVAL	R	"Y position"	D=1.E30

Possible CHOPT values are:

- '␣' Tick marks are drawn on the edges of the picture
- X Cross-wire drawn perpendicular to the X-axis
- Y Cross-wire drawn perpendicular to the Y-axis
- A Value drawn Above cross-wire
- B Value drawn Below cross-wire
- L Value drawn Left of cross-wire
- R Value drawn Right of cross-wire

Draw 'cross-wires' on a picture, optionally with tick marks and values. Cross-wires are lines perpendicular to the X and/or Y axis.

```
XVAL intersection on the X-axis
YVAL intersection on the Y-axis
```

The values of XVAL are always histogram coordinates. The tick marks will be drawn on both side of the cross wire, unless the cross-wires are requested on the boundary of the box surrounding the histogram (i.e. at the extreme limits of the drawn histogram). In this case tick marks will only be drawn inside the box. The options 'A' and 'B' (for Above and Below) refer only to the cross-wire perpendicular to the Y axis. In each case only one cross-wire will be drawn. Similarly 'L' and 'R' (Left and Right) refer only to the cross-wires perpendicular to the X-axis. It is possible to redefine the length of tick marks on the X or Y axis with SET XTIC or SET YTIC. The position of the axis values may be changed with SET XVAL or SET YVAL.

29.0.222 ATITLE [xtit ytit ztit]

XTIT	C	"X Axis title"	D='␣'
YTIT	C	"Y Axis title"	D='␣'
ZTIT	C	"Z Axis title"	D='␣'

Draw axis titles on the axes of the present plot zone.

29.0.223 GRID

Draw a grid in cm.

29.0.224 NULL [xmin xmax ymin ymax chopt]

XMIN	R	"Low range in X"	D=0.
XMAX	R	"High range in X"	D=1.
YMIN	R	"Low range in Y"	D=0.
YMAX	R	"High range in Y"	D=1.
CHOPT	C	"Options"	D='␣'

Possible CHOPT values are:

- '␣' Draw a frame box only.
- S Redefine the scale for the current zone.
- A Axis labels and tick marks are not drawn.
- B The box is not drawn.

Draw a frame box. If XMIN, XMAX, etc. are given, draw a frame box with the window coordinates set to XMIN, XMAX, YMIN, YMAX. Axis labels and tick marks are drawn by default.

Chapter 30

PICTURE

Creation and manipulation of HIGZ pictures.

30.0.225 FILE lun fname [lrecl chopt]

LUN	I	"Logical unit number" R=1:128
FNAME	C	"File name"
LRECL	I	"Record length in words" D=1024
CHOPT	C	"Options" D='␣'

Possible CHOPT values are:

- '␣' Existing file is opened.
- N A new file is opened.
- U Existing file is modified.
- A Automatic saving.

Open a HIGZ direct access picture file. If CHOPT='AU' or 'AN', pictures will be automatically saved on the direct access file. This automatic saving facility can be switched off using SET AURZ 0.

30.0.226 LIST

List all the HIGZ pictures currently stored in memory.

30.0.227 CREATE pname

PNAME	C	"Picture name" Loop

Create a new picture, named PNAME, in memory. Note that all commands which start a new picture (clear workstation) automatically create pictures named PICT1, PICT2, etc. if the command OPTION ZFL or OPTION ZFL1 has been executed.

30.0.228 DELETE pname

PNAME	C	"Picture name" D='␣' Loop

Delete the picture PNAME from memory. PNAME='*' means all pictures.

30.0.229 SCRATCH pname [icycle]

PNAME	C	"Picture name" D='␣' Loop
ICYCLE	I	"Cycle number " D=9999

Delete the picture PNAME from current directory on disk.

30.0.230 PLOT [pname]

PNAME	C	"Picture name" D='␣' Loop

Plot the picture PNAME. PNAME=' ' means the current picture. PNAME='*' means all pictures.

30.0.231 MODIFY [pname chopt]

PNAME	C	"Picture name" D='␣'
CHOPT	C	"Options" D='␣'

Possible CHOPT values are:

- S Software characters are used for the text in menus.
- A The option shadow is used.

Edit the picture PNAME. PNAME=' ' means the current picture. This command is only available on workstations.

30.0.232 MERGE pname [x y scale chopt]

PNAME	C	"Picture name"
X	R	"X coordinates (NDC) where to draw PNAME" D=0
Y	R	"Y coordinates (NDC) where to draw PNAME" D=0
SCALE	R	"Scale factor" D=1.
CHOPT	C	"Options" D='␣'

Possible CHOPT values are:

- '␣' Merge the picture PNAME with the current picture.
- D Picture PNAME is displayed during merging.

Add the picture PNAME to the current picture.

30.0.233 COPY pname1 pname2

PNAME1	C	"Picture name"
PNAME2	C	"New picture name" Loop

Copy a picture.

30.0.234 RENAME pname1 pname2

PNAME1	C	"Old picture name"
PNAME2	C	"New picture name"

Rename a picture.

30.0.235 PRINT [file]

FILE	C	"File name" D='␣'

Print the current picture. The current picture is transformed into a printable file. The file type is defined according to the extension of the file name i.e.

```
FILE = filename.ps    A PostScript file is generated (-111)
FILE = filename.eps   A Encapsulated PostScript file
                      is generated (-113)
FILE = filename.tex   A LaTex file is generated (-778)
```

Do HELP META for details about the metafile types. Note that a new picture is automatically created for each new plot if the OPTION ZFL1 is on.
If FILE=HIGZPRINTER or FILE=' ' the PostScript file paw.ps (-111) is generated and the operating system command defined by the environment variable HIGZPRINTER is executed.
The environment variable HIGZPRINTER should be defined as follow:

```
On UNIX sytems:
       setenv HIGZPRINTER 'lp -dprinter_name paw.ps'
  or
       export HIGZPRINTER='lp -dprinter_name paw.ps'
On VAX/VMS sytems:
       HIGZPRINTER == "XPRINT paw.ps /PRINTER=printer_name"
On CERNVM:
       setenv HIGZPRINTER 'XPRINT PAW PS (PR printer_name'
```

Note that if the environment variable HIGZPRINTER is not defined the file paw.ps is created but not printed.

30.0.236 IZOUT [pname]

PNAME	C	"Picture name" D='␣' Loop

Write the picture PNAME to a direct access picture file (see command PICTURE/FILE). PNAME=' ' means the current picture. PNAME='*' means all pictures.

30.0.237 IZIN pname [icycle]

`PNAME` C "Picture name" `Loop`
`ICYCLE` I "Cycle number " `D=9999`

Read picture into memory from a direct access picture file. (see command PICTURE/FILE). PNAME='*' means all pictures.

30.0.238 IZPICT pname [chopt]

`PNAME` C "Picture name"
`CHOPT` C "Options" `D='M'`

Possible `CHOPT` values are:

- `M` Make a new picture in memory with name PNAME. An empty structure is created in memory and becomes the current picture. If PNAME = ' ', the picture is automatically named as PICTnnn, where the starting value of nnn is either 0 (default), or the value assigned by SET to the parameter PICT.
- `D` Display the picture PNAME in memory.
- `S` Scratch the picture PNAME from memory. If PNAME = ' ' the current picture is scratched.
- `N` The picture following the current picture in memory becomes the current picture. If the current picture is the last one in memory, the first picture in memory becomes the current picture.
- `L` Give the list of the pictures in memory, following the sequence of their storage in memory.
- `F` The First picture in memory becomes the current picture.
- `P` Print the picture data structure. Useful to debug programs.
- `C` Set Current picture. All calls to HIGZ graphic functions are stored in the current structure according to the option selected be IGZSET.

Perform various operations on a picture. PNAME=' ' means the current picture. PNAME='*' means all pictures.

30.0.239 SWITCH [chopt]

`CHOPT` C "Options" `D='G'`

Possible `CHOPT` values are:

- `G` graphics output only.
- `Z` Graphics primitives stored in ZEBRA memory only.

Set the graphics switch to control plotting output to terminal (G) and/or picture in memory (Z).

30.0.240 IGSET [chatt value]

`CHATT` C "Attribute name" `D='SHOW'`
`VALUE` R "Attribute value" `D=0.`

Set a HIGZ attribute. If CHATT='SHOW' print default and current values for all attributes. If CHATT='*' restore default values for all attributes. If VALUE=0, the attribute is set to its default value.

IGSET : Current values in use

Parameter	Current value	Default value	Explanation
FAIS	0	0	Fill area interior style
FASI	1	1	Fill area style index
LTYP	1	1	Line type
BASL	.150	.010	Basic segment length (NDC)
LWID	1.000	1.000	Line width
MTYP	1	1	Marker type
MSCF	1.000	1.000	Marker scale factor
PLCI	1	1	Polyline color index
PMCI	1	1	Polymarker color index
FACI	1	1	Fill area color index
TXCI	1	1	Text color index
TXAL	0 0	0 0	Text alignment
CHHE	.280	.010	Character height
TANG	.000	.000	Text angle
TXFP	0 2	0 2	Text font and precision
PICT	1	1	Current automatic number
BORD	0	0	Border flag
PASS	1	1	Number of pass in IGTEXT
CSHI	.030	.020	IGTEXT shift
LASI	.018	.018	Label axis size
LAOF	.013	.013	Label axis offset
TMSI	.019	.019	Tick marks size
AWLN	.000	.000	Axis wire lenght
BARO	.250	.250	Offset of IGHIST (IGRAPH) bars
BARW	.500	.500	Width of IGHIST (IGRAPH) bars
NCOL	8	8	Number of COLors
CLIP	1	1	Clipping mode
NLIN	40	40	Number of line for 3D shapes
AURZ	0	0	Automatic saving flag
DIME	2	2	Dimension used (2D or 3D)

Chapter 31

ZEBRA

Interfaces to the ZEBRA RZ, FZ and DZ packages.

Chapter 32

ZEBRA/RZ

ZEBRA/RZ package: direct access Input/Output.

32.0.241 FILE lun fname [lrecl chopt]

LUN	I	"Logical unit number" R=1:128
FNAME	C	"File name"
LRECL	I	"Record length in WORDS" D=1024
CHOPT	C	"Options" D='␣'

Possible CHOPT values are:

'␣' Read only mode.

U Update mode.

Open an existing direct access file.

32.0.242 MAKE lun fname [lrecl nrec nwkey chform chtags]

LUN	I	"Logical unit number" R=1:128
FNAME	C	"File name"
LRECL	I	"Record length in WORDS" D=1024
NREC	I	"Number of records" D=1000
NWKEY	I	"Number of words per Key" D=1
CHFORM	C	"Key format" D='I'
CHTAGS	C	"List of Tags" D='HBOOK-ID'

Possible CHFORM values are:

I

B

A

H

Open a new direct access file.

32.0.243 MDIR chdir [nwkey chform chtags]

CHDIR	C	"Directory name"
NWKEY	I	"Number of words per Key" D=1
CHFORM	C	"CHFORM" D='I'
CHTAGS	C	"List of Tags" D='HBOOK-ID'

Create a new RZ directory below the current directory.

32.0.244 DDIR chdir

CHDIR	C	"Directory name"

Delete the directory CHDIR from the current directory.

32.0.245 LDIR [chpath chopt]

CHPATH	C	"Path name" D='␣'
CHOPT	C	"Options" D='␣'

Possible CHOPT values are:

'␣' List contents of a directory.

A List all the Ntuple extensions.

T List a directory Tree.

List contents of a directory (memory or disk). To list all RZ files currently opened, type 'LD //'. Note that if the Current Directory is //PAWC, this command uses the same format as HISTO/LIST.

32.0.246 CDIR [chpath chopt]

CHPATH	C	"Path name" D='␣'
CHOPT	C	"Options" D='␣'

Change the current working directory (CWD). IF CHPATH is given make it the new CWD. Otherwise, print the pathname of the CWD.

```
Ex.  CD dir1          ; make DIR1 the new CWD
     CD //file1/dir2 ; make //FILE1/DIR2 the new CWD
     CD               ; print the name of the CWD
```

32.0.247 PURGE [keep]

KEEP	I	"Number of cycles to be kept" D=1

Purge an RZ directory.

32.0.248 LOCK [chlock]

CHLOCK	C	"Lock identifier" D='RZFILE'

Lock an RZ directory.

32.0.249 FREE [chlock]

CHLOCK	C	"Lock identifier" D='RZFILE'

Free an RZ directory.

32.0.250 STAT chpath

CHPATH	C	"Name of top directory"

Print space statistics for an RZ file.

Chapter 33

ZEBRA/FZ

ZEBRA/FZ package: sequential access Input/Output.

33.0.251 FILE lun fname [lrecl chopt]

`LUN`	I	"Logical unit number" `R=1:128`
`FNAME`	C	"File name"
`LRECL`	I	"Record length in words" `D=900`
`CHOPT`	C	"Options" `D='IX'`

Possible `CHOPT` values are:

- `I` Input file.
- `O` Output file.
- `X` Binary exchange mode.
- `A` Alphanumeric exchange mode.

Open an FZ sequential formatted or unformatted file.

33.0.252 TOFZ lun [chopt]

`LUN`	I	"Logical unit number of FZ file" `R=1:128`
`CHOPT`	C	"Options" `D='␣'`

Copy the current directory tree onto an FZ file.

33.0.253 FRFZ lun [chopt]

`LUN`	I	"Logical unit number of FZ file" `R=1:128`
`CHOPT`	C	"Options" `D='␣'`

Copy the FZ file into the current directory tree.

33.0.254 TOALPHA fname

`FNAME`	C	"Name of the FZ text file"

Copy the current directory tree onto a FZ file. An alphanumeric format is used. The file FNAME can be exchanged between different machines.

33.0.255 FRALPHA fname

`FNAME`	C	"Name of the FZ text file"

Copy the FZ alphanumeric file into the current directory.

Chapter 34

ZEBRA/DZ

ZEBRA/DZ package: debugging.

34.0.256 SHOW name [number chopt]

`NAME` C "Bank name"
`NUMBER` I "Bank number" `D=1`
`CHOPT` C "Options" `D='BSV'`

Possible `CHOPT` values are:

- `B` Print the bank.
- `S` Print the bank contents from left to right Sideways with up to ten elements per line.
- `V` Print the vertical (down) structure.
- `D` Print the bank contents from top to bottom Downwards with five elements per line.
- `L` Print the linear structure.
- `Z` Print the data part of each bank in hexadecimal format

Display the contents of a bank or a data structure identified by its NAME and NUMBER. The output format of the data part is controlled by the internal or external I/O characteristic.

34.0.257 SURV name [number]

`NAME` C "Bank name"
`NUMBER` I "Bank number" `D=1`

Print a survey of the structure identified by NAME, NUMBER.

34.0.258 SNAP [idiv chopt]

`IDIV` I "Division number " `D=2 R=0:24`
`CHOPT` C "Options" `D='M'`

Possible `CHOPT` values are:

- `M` Print Map entry for each bank
- `E` Extend map entry to dump all links of each bank (otherwise only as many links as will fit on a line)
- `F` Full. Dump all active banks, links and data
- `K` Kill. Dropped banks to be treated as active (dropped banks are not normally dumped under D or F option)
- `L` Dump all Link areas associated with the store
- `W` Dump the Working space, links and data
- `Z` Dump the information in hexadecimal.

Snap of one or more divisions. Provides a snapshot of one or more divisions in a ZEBRA store. The kind of information provided is controlled by CHOPT.

34.0.259 VERIFY [idiv chopt]

`IDIV` I "Division number " `D=0 R=0:24`
`CHOPT` C "Options" `D='CLSU'`

Possible `CHOPT` values are:

- `C` Check chaining of banks only
- `L` Check validity of the structural links (implies 'C')
- `S` Check the store parameters
- `U` Check the validity of the up and origin (implies 'C')
- `F` Errors are considered fatal and generate a call to ZFATAL

Check the structure of one or more ZEBRA divisions. The verification detail depends on the settings in CHOPT.

34.0.260 STORE [ixstor]

`IXSTOR` I "Store number" `D=0 R=0:24`

Display the structure of the ZEBRA store IXSTOR. Output the parameters characterizing the store, followed by a list of all divisions and all link areas associated with the store in question.

Chapter 35

FORTRAN

Interface to MINUIT, COMIS, SIGMA and FORTRAN Input/Output.

35.0.261 HMINUIT

To input commands for Interactive MINUIT in a macro. Example:

```
Application HMINUIT EXIT
SET EPS 1.E-14
MIGRAD
SET PRIN 2
MINOS
EXIT
Histo/fit 10 g m
```

35.0.262 COMIS

Invoke the COMIS FORTRAN interpreter. COMIS allows to execute FORTRAN routines without recompiling and relinking. It communicates with PAW commands through vectors and functions. COMIS has its PAW-independent command structure. Example in command mode:

```
PAW > Comis
CS >    do 10 i=1,10
MND>       x=sqrt(i)*10.
MND>       print *,i,x
MND> 10 continue
MND>    END
CS > quit
PAW >
```

COMIS code may be inserted into a macro. Example:

```
Vector/Create Y(10) r 1 2 3 4 5 6 7 8 9 10
*
*  In the following COMIS code, the statement "Vector Y" declares
*  to COMIS an existing KUIP vector. KUIP dimension is assumed.
*  The statement "Vector X(10)" creates a new KUIP vector.
*  (Note that SUBROUTINEs must be declared before the MAIN program)
*  (KUIP vectors cannot be created into the MAIN program)
*
APPLIcation COMIS QUIT
      SUBROUTINE DEMO
      Vector Y
      Vector X(10)
      do 10 i=1,10
         XX=i
         X(i)=Y(i)*sqrt(XX)*10.
  10  CONTINUE
      END
      CALL DEMO
      END
QUIT
Vector/print X       | Print KUIP vector created by COMIS
```


35.0.263 CALL urout

UROUT C "User routine"
Execute the routine UROUT. UROUT may be a routine compiled and linked with PAW. For example : CALL HPRINT(10).
UROUT may also be the name of a file which can be edited interactively with the command EDIT. For example if file UROUT.FOR contains:

```
      SUBROUTINE UROUT(N)
      SUM=0.
      DO 10 I=1,N
        SUM=SUM+I
   10 CONTINUE
      PRINT *,SUM
      END
```

Then one can type CALL UROUT.FOR(10). The routine UROUT may also contain references to the library routines mentioned below.
The functions $CALL, $ICALL, and $DCALL allow to call REAL, INTEGER, and DOUBLE PRECISION functions, respectively. The function call must be enclosed in quotes, for example:

```
      $CALL('fun.f(1.5)')
```

with file fun.f containing

```
      FUNCTION FUN(X)
      FUN=X**2
      END
```

The following routines from the CERN Program Library can be called:

```
 From HBOOK:
   HBOOK1,HBOOK2,HBOOKN,HFILL,HF1,HPRINT,HDELET,HRESET
   HFITGA,HFITPO,HFITEX,HPROJ1,HPROJ2,HFN,HGFIT,HRENID
   HROPEN,PAOPEN,PACLOS,PAREAD,PAWRIT,HCDIR,HGIVEN,HKIND
   HTITLE,HBFUN1,HBFUN2,HRNDM1,HRNDM2,HBARX,HBARY,HDIFFB
   HPAK,HPAKE,HUNPAK,HGIVE,HGN,HGNF,HGNPAR,HF2,HFF1,HFF2
   HRIN,HROUT,HI,HIE,HIX,HIJ,HIF,HIDALL,HNOENT,HX,HXY
   HTITLE,HCOPY,HSTATI,HBPROF,HOPERA,HIDOPT,HDERIV,HBAR2
   HMAXIM,HMINIM,HMAX,HMIN,HSUM,HNORMA,HMCINI,HMCMLL
   HEXIST,HREND,HRGET,HRPUT,HSCR,HFIND,HCX,HCXY,HLABEL
   HBPROX,HBPROY,HBANDX,HBANDY,HBSLIX,HBSLIY,HPROF2
   HBOOKB,HBSTAT,HDIFF,HUNPKE,HREBIN,HERROR,HGNTB,HSTAF
   HOUTPU,HERMES,HISTDO,HFUNC,HXI,HIJXY,HXYIJ,HLPOS,HFC1
   HSPLI1,HSPLI2,HMDIR,HLDIR,HLOCAT,HFITH,HFITV,HFINAM
   HBNT,HBNAME,HBNAMC,HFNT,HFNTB,HGNT,HGNTF,HGNTV,HBSET
   HRENAME,HNTDUP,HFITHN
 From HPLOT:
   HPLOT,HPLSYM,HPLERR,HPLEGO,HPLNT,HPLSUR,HPLSOF,HPLFRA
   HPLABL,HPLSET,HPLGIV,HPLOC,HPLTOC,HPLNEW,HPLOPT
 From ZEBRA:
   MZSTOR,MZDIV,MZLINK,MZWORK,MZBOOK,MZDROP,MZPUSH
   MZWIPE,MZGARB,MZFORM,LZFIND,LZFID,DZSHOW,DZVERI
   FZIN,FZOUT,FZFILE,FZENDI,FZENDO
   RZCDIR,RZLDIR,RZFILE,RZEND,RZIN,RZOUT,RZVIN,RZVOUT
   RZOPEN,RZIODO,RZCLOS,RZQUOT
 From KUIP:
   KUGETV,KUDPAR,KUVECT,KILEXP,KUTIME,KUEXEL,KUPROS
   KUNWG,KUCMD,KUGUID,KUNDPV,KUPAR,KUPVAL,KUACT
 From HIGZ:
   IPL,IPM,IFA,IGTEXT,IGBOX,IGAXIS,IGPIE,IGRAPH,IGHIST
   IGARC,IGLBL,IGRNG,IGMETA,IGSA,IGSET,IRQLC,IRQST,ISCR
   ISELNT,ISFAIS,ISFASI,ISLN,ISMK,ISVP,ISWN,ITX,ICLRWK
   IGPAVE,IGTERM,ISFACI,IGHTOR
```

35.0.264 LOOP ntimes urout

`NTIMES` I "Number of calls" `D=1`
`UROUT` C "User routine"
The routine UROUT is called NTIMES times. See command CALL for explanation of UROUT.

35.0.265 FILE lun fname [status]

`LUN` I "Logical unit number"
`FNAME` C "File name"
`STATUS` C "File status" `D='DONTKNOW'`
Possible `STATUS` values are:

`OLD` Open existing file for reading.

`APPEND` Open existing file and position at EOF.

`NEW` Create new file; error if already existing.

`UNKNOWN` Open existing or create new file.

`DONTKNOW` Like UNKNOWN except on VMS opens highest cycle.

Open a FORTRAN formatted text file. UNKNOWN opens a file for write access without flagging an error if the file already exists. On VMS a new cycle is created. DONTKNOW is the same as UNKNOWN except on VMS where the highest cycle is opened. This option should be used if it is not yet known whether the file will be read or written.

35.0.266 CLOSE lun

`LUN` I "Logical unit number" `R=0:128`
Close the file on unit LUN. If the file has been opened with HISTO/FILE, PICTURE/FILE, etc, then before closing the unit, PAW will close correctly the file with CALL HREND or FZENDI(O), ICLWK, etc. Giving 0 as unit will close all open files.

35.0.267 REWIND lun

`LUN` I "Logical unit number" `R=1:128`
Rewind the file on unit LUN.

35.0.268 SIGMA [expr]

`EXPR` C "Expression" `D='␣'`
Invoke the SIGMA package. SIGMA is an array manipulation package using its own vector-oriented language, outside the PAW command conventions. SIGMA may be invoked in one of the three following ways:

```
1- Using the KUIP $SIGMA function. Example:
   PAW > Vector/Create x(10) r 1 2 3 4 5 6 7 8 9 10
   PAW > Graph 10 x $sigma(sqrt(x))

2- Using the SIGMA command. Example:
   PAW > sigma x=array(10,1#10)
   PAW > sigma y=sqrt(x)
   PAW > Graph 10 x y

3- Using the APPLication command. Example:
   PAW > APPLication SIGMA
   SIGMA > x=array(10,1#10)
   SIGMA > y=sqrt(x)
   SIGMA > exit
   PAW > Graph 10 x y
```


Chapter 36

NETWORK

To access files on remote computers. To send messages to a remote process (ZEBRA server)

36.0.269 RLOGIN host

`HOST` C "Host name" `D='␣'`
Start a communication with a remote machine HOST. Current Directory will be changed to //HOST.

36.0.270 RSHELL message

`MESSAGE` C "Message to remote host" `D='␣'`
Send MESSAGE to current remote host. Note that the Current Directory must be //HOST (see RLOGIN). Some PAW commands (Histo/Plot, Histo/List) can communicate directly with HOST.

Chapter 37

NETWORK/PIAF

To establish and control the connection to the Piaf server. The Parallel Interactive Analysis Facility (Piaf) is a cluster of 5 high-performance HP workstations.

A locally running PAW session (client) connected to the Piaf server can access Hbook RZ files stored on the server side in a transparent way. Commands with high CPU or I/O requirements, e.g. NT/PLOT and NT/PROJECT are processed by the server and only the resulting histograms etc. are sent back to the client.

In order to use the Piaf server the PAW client must have been compiled with the communications option CZ using TCP/IP as transport protocol.

37.0.271 CONNECT [server node]

`SERVER` C "Server name" `D='piaf'`
`NODE` C "Front-end node" `D='128.141.201.28'`
Establish a connection to the Piaf server. Subsequent HISTO/FILE commands can refer to files on the server using path names '//piaf/file.hbook'.

37.0.272 STAGE source [target option]

`SOURCE` C "Source file identifier"
`TARGET` C "Target file name" `D='␣'`
`OPTION` C "Options" `D='␣'`
Possible `OPTION` values are:
`N` NoWait. Submit the request to the staging system and return immediately.

Stage an Ntuple file on the Piaf server. The source file identifier can be the name of a local file on the client system, a Fatmen path, or a tape identifier. If the target file name is not specified it is constructed from the source identifier.
Unless the option N is used the STAGE command waits until the staging is completed and the file is ready to be used.

37.0.273 GET remote [local format recl]

`REMOTE` C "Remote file name"
`LOCAL` C "Local file name" `D='␣'`
`FORMAT` C "Text or binary" `D='RZ'`
`RECL` I "Record length in bytes" `D=0 R=0:`
Possible `FORMAT` values are:
`T` Text file.

`RZ` Zebra RZ file in exchange format.

`BIN` Binary file with record length given by RECL.

Copy a file from the Piaf server to the client system. If not specified the local file name will be same as the remote file name. RECL needs to be specified only for BIN format. For IBM only: A text file with RECL=0 is written in V-format. Otherwise it is written in F-format with the given LRECL.

37.0.274 PUT local [remote format]

`LOCAL` C "Local file name"
`REMOTE` C "Remote file name" `D='␣'`
`FORMAT` C "Text or binary" `D='RZ'`
Possible `FORMAT` values are:
`T` Text file.

`RZ` Zebra RZ file in exchange format.

`BIN` Binary file.

Copy a file from the client system to the Piaf server. If not specified the remote file name will be same as the local file name. Note for VMS: Avoid text files with variable record length. Use Stream_LF format instead.

37.0.275 LS [files]

`FILES` C "File pattern" `D='␣'`
List files stored on the Piaf server.

37.0.276 CAT file

`FILE` C "File name"
Print a Piaf file on the terminal.

37.0.277 RM file

`FILE` C "File name"
Delete a Piaf file.

37.0.278 MV from to

`FROM` C "Old file name"
`TO` C "New file name"
Rename a Piaf file.

37.0.279 CP from to

`FROM` C "Old file name"
`TO` C "New file name"
Copy a Piaf file to a new file.

37.0.280 PWD

Show current Piaf working directory.

37.0.281 MKDIR dir

`DIR` C "Directory name"
Create a new directory on Piaf.

37.0.282 RMDIR dir

`DIR` C "Directory name"
Delete a directory on Piaf.

37.0.283 MESSAGE mess

`MESS` C "Message"
Send a message to Piaf.

37.0.284 STATUS

Inquire the status of the Piaf server.

37.0.285 MODE [option]

`OPTION` C "Processing mode" `D='?'`
Possible `OPTION` values are:
`?` Inquire the current mode.

`SEQ` Set sequential processing mode.

`PAR` Set parallel processing mode.

Inquire or change the processing mode of the Piaf server. In parallel mode the Piaf server uses slave servers to process Ntuple requests on all available machines in parallel.
With certain types of COMIS selection functions, e.g. when reading from an external file for each event, parallel processing is not possible. The Piaf server should be switched to sequential mode, i.e. the master server alone processes the Ntuple request.

37.0.286 LOGLEVEL level

`LEVEL` I "Log level" `D=0`
Set the level of diagnostic output from the Piaf server.

37.0.287 DISCONNECT

Close the connection to the Piaf server.