Mn_Fit

A Fitting and Plotting Package Using MINUIT.

Ian C. Brock

Carnegie Mellon University
Pittsburgh, PA 15213

Electronic Mail: BROCK@VXCERN.CERN.CH or VXCERN::BROCK

June 5, 1991

CDF/DOC /CDF /PUBLIC/1481
J.D. Lewis
6 June 1991

Mn_Fit: A Fitting and Plotting Package Using Minuit

Mn_Fit is an interactive fitting and plotting package, that uses MINUIT to fit histograms or data read in from a file and displays the fit results on a screen. Mn_Fit was developed by Ian Brock of Carnegie Mellon University. Although it provides many of the same functions as PAW, the fitting package is generally more informative and versitile and it is much easier to produce publication quality plots. At present Mn_Fit is implemented on Vaxes, Decstations and Apollos. If there is demand it should also be fairly easy to get it running on an IBM or Silicon Graphics.

Mn_Fit is fully self-documenting. At any time you you can type `help` or `?` for help. The following document is taken directly from Mn_Fit's help library. Three graphics interfaces are currently availible with Mn_Fit on Fermilab Vaxes: DI3000, DECGKS, and PLTSUB (a free package developed at Cornell). Because Mn_Fit is written with HIGZ and other CERN libraries, it should be possible to implement it on an computer that supports the CERN libraries.

As with most software products at Fermilab, to use Mn_Fit, on the FNALD and CDF Vax clusters issue the command:

```
SETUP MN_FIT
```

To run say:

```
MN_FIT -qual
```

where `-qual` is a qualifier for the graphics interface: **-3** for DI3000, **-v** for DECGKS (VAX), and **-p** for PLTSUB. On the FNAL cluster, only the DI3000 executable and the help library are availible. To use Mn_Fit on those systems, issue the command:

```
@usr$root:[jdl.mn_fit]MN_FIT
```

Please report problems to me (FNALD::JDL), not to Ian Brock.

Contents

List of Figures

Chapter 1

Introduction

Mn_Fit is an interactive fitting and plotting package, that uses MINUIT to fit histograms or data read in from a file and displays the fit results on a screen. Hardcopies of pictures can easily be made with a quality suitable for publication. Mn_Fit can also be used purely as a plotting package without using the fitting.

In this manual I will first give a brief introduction to what you can do with Mn_Fit, how to run it and where to find more information. This is followed by a description of some of the concepts and a list of the examples to demonstrate various features. A brief list of all the commands is given, followed by a detailed description of each command. In the appendices you can find examples of the symbols, fonts and hatchings. There is also a series of demonstration macros ranging from simple plotting and fitting to the use of colours, fonts, overlays and inserts and simultaneous fitting of 2 plots.

This manual corresponds to Mn_Fit Version 3.01/03. By its very nature the manual is not completely up-to-date by the time you get it. Interactively you can find out what has changed by giving the command `HELP CHANGES`. It is possible for you to produce this manual yourself, or to produce updates to the manual when new versions are released. If you have the standard Mn_Fit logicals or links defined (automatically the case if you have run Mn_Fit), documentation can be found in the CMZ file `mn_fit_cmz:mn_util.cmz` (Vax) or `~/mn_fit_cmz/mn_util.cmz` (Apollo), in the deck `//mn_util/doc/manual`. The version number n.mm/ll has the following meaning: Major changes result in n being increased; addition of new commands or modification of existing commands result in mm being increased; ll is for bug fixes.

1.1 What Mn_Fit Can Do

Features of the plotting include user specification of all the plotting variables, overlaying of one or more histograms or functions and the trivial changing of symbols used in plotting. There are also secondary histogram identifiers so that you can keep track of data and Monte Carlo plots for example by giving them different secondary identifiers. If you make projections or slices of a plot, they are stored with the same primary identifier, but are given a different secondary identifier. Similarly the functions used in fitting and projections of Ntuples are given the same primary identifier and a different secondary identifier (see section 1.8 on page 20 (Secondary_identifiers) for more information). It is easy to change the parameters for one plot in a picture and then `REDRAW` the whole picture without having to re-enter all the plotting commands (see section 4.79

on page 92 (SET) for details).

It is possible to interactively specify cuts on a plot and then make projections of the plot using the cuts (see section 4.18 on page 47 (CUT) and section 4.71 on page 90 (PROJECT)). Cuts can be in standard FORTRAN syntax or using symbols ($<, >$ etc.). The cut can be either a simple expression (variable condition value), a more complicated expression (expression condition expression), or a COMIS function. It is also possible to project onto a variable or an expression and you can use a variable or expression as a weight for each point.

For suitable commands you can give a range of either primary or secondary identifiers for the command (e.g. `FET 1:300` will fetch all histograms with identifiers between 1 and 300); see section 1.7 on page 19 (Identifiers).

For details on the implementation of the MINUIT package see section 3 on page 39 (MINUIT). There are 2 versions of MINUIT available: the standard CERN version and a much older version extensively modified by C. Rippich. All of the standard MINUIT commands are available, with a few minor modifications. Several new commands have also been added to give you more control on what you fit and to permit a graphical display of the fit results.

There are many predefined functions available such as Gaussians in many variations, Breit-Wigner, Polynomials etc. (see section 4.41.9 on page 68 (FUNCTION LIST) for details). If you need a more specialized function you can either define it interactively using the CERN COMIS package, meaning you can write your own functions to fit with without relinking Mn_Fit (see section 1.13 on page 25 (Using_COMIS) and section 4.41.9 on page 72 (FUNCTION LIST COMIS) for more details), or you can relink Mn_Fit with a user function (see section 4.41.9 on page 74 (FUNCTION LIST USER) for more details).

Mn_Fit can be run either interactively, or from a file, or you can execute a series of commands in a file. To read from a file while running interactively issue the command `EXECUTE` (or `READ COMMAND`) `filename`. You can also define your own commands within Mn_Fit using the `DEFINE` command. Within macros there are DO loops and you can pass parameters to the macro as well as setting them within the macro.

All commands can be abbreviated to the point of ambiguity, and commands are read in using the TYPSCN package. You can define aliases for any commands or strings and specify whether alias translation is turned on or off (see section 4.4 on page 43 (ALIAS) for more details). Most of the time that you should give a value you can give a number, register, function parameter, histogram parameter or a variable (see section 1.6 on page 17 (Numbers) for more details).

Mn_Fit is designed so that you do not have to know the complete command syntax before starting to give a command. Just give the first word of a command and you should be prompted for everything else needed. Within a command you can often get more help by typing `?`. The prompt will indicate if this is possible. Interactive help is always available. Just give the command `HELP`. The help is standard Vax Help on Vaxes and a simulated version on all other machines.

1.2 How To Run Mn_Fit

At present Mn_Fit is implemented on Vaxes, Decstations and Apollos. If there is demand it should also be fairly easy to get it running on an IBM. If you are on a Cornell machine (or other CLEO group Vax) give the command:

```
DO MN_FIT
```

If you are on the CERN Vax cluster give the command:

```
    @DISK$L32:[BROCK]MN_FIT
```

If you are on the L3 online Vax give the command:

```
    MN_FIT                              for the current official version
    @USR2:[BROCK]MN_FIT                 for my latest "official" version
```

If you are at FNAL give the commands:

```
    SETUP MN_FIT
    MN_FIT
```

If you are on an L3 Apollo give the command:

```
    mn_fit                              for the current official version
                                        (not yet working - but should be soon)
    /user/brock/bin/mn_fit              for my latest "official" version
```

Versions for Apollo DN1000 and 3000 type cpu's are available and the correct one will be automatically selected (using the ISP environment variable).
You can also give parameters with the command to specify which version to run:

filename	**Read all the Mn_Fit commands from the given filename.**
-g	**GKSGRAL version**
-dec	**DECGKS version (same as VAXGKS)**
-v	**VAXGKS version (same as DECGKS)**
-dgks	**Dec GKS-3D version**
-di3	**DI3000 version**
-p	**PLTSUB version**
-t	**Test version**
-disp	**Display version (only implemented for L3)**
-dbg	**Run Mn_Fit in debug**

Note that if you specify a **filename** with the Mn_Fit command, the first line of the file should be blank, or should specify the screen device.

When you start up Mn_Fit it will look for the file **mn_logon.mnf** in your local directory and if that does not exist it will look for the logical name **mn$logon** on the Vax or the file **~/user_data/mn_logon.mnf** on the Apollo. The file will be executed and then control given to the terminal. A warning will be given if neither file is found.

Mn_Fit is interfaced to graphics devices using either HIGZ or PLTSUB. Within HIGZ the following GKS flavours have been implemented: GKSGRAL, DECGKS (VAXGKS), DEC GKS-3D, DI3000. Each institute has a default interface, but you can also link with any other.

1.3 Fitting With Mn_Fit

The usual way to fit is using MINUIT. However spline fitting and histogram smoothing are also available (Vax only) for one dimensional plots. There are 2 versions of MINUIT implemented. Up to October 1990 a version modified by C. Rippich was used. Since then the standard CERN version is the default as it contains all the extra options added in the C. Rippich version. The older version is still available, but has to be linked at your institute.

There is great flexibility in what you can fit and the functions used to fit with. The usual procedure is to select the function(s) you want to fit with using the `FUNCTION ADD` and `FUNCTION USE` commands and then to invoke MINUIT using the `FIT` command. You will be asked whether you want to do likelihood or χ^2 fitting.

It is possible to fit 1 or 2 dimensional histograms or a series of data points. It is not possible to fit to a true scatter plot. Use the `PROJECT` command to make a binned 2-d histogram out of the scatter plot if you want to fit it. You can, however, fit surfaces where the x,y,z,dz values are variables in an Ntuple or a file containing an Ntuple card. You use the `SET NTUPLE` command to assign the variables to the axes. You can, of course, use the same mechanism to do a 1-dimensional fit to 2 variables of an Ntuple.

You can simultaneously fit 2 or more plots by giving both their identifiers with the fit command. If you want to use different background functions for each plot, but the same signal, e.g. when you fit a mass peak from 2 different decay modes, you should give the function numbers NOT to be used for a plot after you give its identifier. If you are using the option `SET RATIO ON` (see below), you can only use different background functions for 2 plots as Mn_Fit could not constrain the sum of a set of parameters to be 1 in MINUIT. (As of Mn_Fit version 3.00 constraints are possible, so I could add this feature if anyone needs it).

There are 2 options on how the parameters are specified. In the first (`SET RATIO ON` - default) the parameters will be modified so that the first one for each function is the total area, the second one is the ratio of the area under the function in the first plot to the total, etc. This form is more linear than using the ratio of the areas in each plot as a parameter. If you use the option `SET RATIO OFF` then each plot has an independent area.

The `FIT` command then defines the MINUIT parameters or writes the necessary MINUIT datacards and checks that the histograms you want to fit actually exist. The histogram(s) are copied to a separate storage space so that you cannot accidently delete or modify them while fitting. MINUIT is invoked and after the parameters are listed in the standard format, you can give MINUIT commands for fitting.

It is possible to constrain parameters. For example you may wish to fix the relationship between 2 means in 2 Gaussians (see section 5.3 on page 120 (MINUIT CONSTRAIN) for more details). It is possible to exclude regions from a fit or include regions, dump the χ^2 or likelihood for each point, display the fit, either as an overlayed function, or background subtracted or both ways (see section 4.79.8 on page 95 (SET DISPLAY MODE)).

Note that the χ^2 or the likelihood is stored in register 111 (access it using the syntax `R111`), and the confidence level is stored in register 112, so that you can use it to put in a plot for example.

There are a large number of built-in functions (see section 4.41.9 on page 68 (FUNCTION LIST)), which are sufficient for most purposes. However you can write your own functions and either invoke COMIS or relink Mn_Fit depending on the amount of CPU time the fitting takes. The form for the COMIS and USER functions is virtually interchangeable (usually only the function name is different), so you can start with a COMIS function and then relink if you have a lot of fitting to do. You can also use a 1 or 2-dimensional histogram as a function. Use the **FUNCTION ADD HISTOGRAM id** syntax, and you will be asked if the histogram errors should be included in the χ^2 calculation or not. The only built-in functions for 2-dimensional histograms are a 2-D Gaussian and fitting to another 2-dimensional histogram. For any other form you must write your own USER or COMIS function (see section 4.41.9 on page 74 (FUNCTION LIST USER) or section 4.41.9 on page 72 (FUNCTION LIST COMIS) for more details).

If your function changes a lot across a bin you can integrate it across the bin for each data point. Use the command **SET FIT INTEGRATE ON ninterval**, where **ninterval** is the number of intervals for the integration. Note that the cpu time used is directly proportional to the number of intervals.

You can also fit with several functions and include an overall normalization factor using the command **SET NORM ON**. This turns on an overall normalization factor for the function(s) you are fitting with. The parameter will be called **NORM00** and will be the first parameter. This is useful if you want to fit to functions of the form:

```
NORM00 * (HIST1 + ALPHA*HIST2)
```

where **HIST1** and **HIST2** are 2 histograms and **ALPHA** is the relative contribution of each.

Always check that the results of the fit are reasonable - that the errors are reasonable (see section 1.3.1 on page 14 (Errors)) and that it looks as if the fit has converged properly.

WARNING: A standard procedure is to fit the background either side of a peak; then remove the peak exclusion, fix the background and fit the peak; and finally to float everything and fit again. While it is possible to do this without exiting MINUIT, in the C. Rippich version, for some unknown reason, the latter 2 fits often do not converge properly and so I recommend exiting MINUIT and restarting it after each of the 3 fits. This seems to give better results. I believe that this problem does not occur with the new CERN version of MINUIT.

1.3.1 MINUIT Errors

MINUIT gives two different errors from its fitting routines: parabolic errors and MINOS errors. For functions which have no correlations between the parameters the parabolic error and the MINOS errors should be the same. If there are correlations, or the errors are asymmetric you should always use the MINOS errors.

Occasionally MINUIT will claim to have converged and the parabolic errors are ridiculously small. This often happens if you fit with functions having correlated parameters (like Gaussians!). The usual fix to this is to give the command **HESSE**, which recalculates the covariance matrix, and then **MINIMIZE** again. A good check is that the parabolic error should lie between the MINOS errors if there are no correlations and be smaller than the MINOS errors if there are correlations. Again, this problem is much reduced with the CERN version of MINUIT.

If you want to have 90% confidence level upper limits for example, it is possible to get these directly by changing the χ^2 change used to calculate the error (command **ERROR_DEF 1.69**). Note that the likelihood is scaled by a factor of 2, so that 1 sigma errors also correspond to a likelihood change of 1.

You can get a contour plot on the terminal using the **CONTOUR** command and a graphical contour plot using the **MNCONTOUR** command. Note that **CONTOUR** just scans the 2-dimensional space of the 2 variables, whereas **MNCONTOUR** minimizes the FCN at each point on the contour.

1.3.2 Fitting With Small Numbers of Events

To start with, DO NOT fit with small numbers of events if you can avoid it! If you have to fit with small numbers of events, then DO NOT do a χ^2 fit with small numbers of entries per bin (the nominal value is less than 10). You should be doing a likelihood fit. If you cannot likelihood fit, for example if you have small numbers of events and have done a continuum subtraction, or if for some reason your errors are not the square root of the number of entries, you have a problem! In general, the Gaussian error on a low fluctuation is too small, so that a χ^2 fit will tend to give a total area under the function you are fitting with which is smaller than the area of the plot you are fitting to.

The previous policy in Mn_Fit was to ignore bins with no entries for a χ^2 fit (they are included in a likelihood fit properly). This has now been changed to give these bins an error of 1. This is obviously not strictly correct, but may be better than before.

As an experiment I created a plot with 10 bins, 2 of them with 1 entry and the others with zero:

```
With a likelihood fit to a flat spectrum the result was     0.2 +/- 0.03
With a chi**2 fit and errors of 1 on all points it was      0.2 +/- 0.3!!
With a chi**2 fit and errors of 0 on zero points it was     1.0 +/- ??
```

Clearly the likelihood fit gave the correct answer and is what should be used if you can. While the χ^2 fit also gave the right value, the error is way wrong! The third result is as expected because all the zero points were ignored.

Closely connected with the above comments is, what to do if you want to do a continuum or background subtraction, when some of the bins in either the signal or background plot are zero.

As I said for χ^2 fitting with small numbers of entries per bin, if you can possibly avoid it, DO NOT do it! The preferable way, if you know the background shape, is to fit it and then subtract the fitted form from the signal (including the error from the fit).

Mn_Fit used to assign an error of zero to bins with zero entries. The default has now been changed to assign an error of 1 when using the plot in the commands `ADD`, `SUBTRACT`, `MULTIPLY`, `DIVIDE`, `NORMALIZE`, and `SCALE`. Thus if you had 0 entries in a $\Upsilon(4S)$ bin and 1 in the corresponding continuum bin and scaled the continuum by a factor 2, you used to get an error of 2 for that bin, while now you will get an error of 2.24. Neither of these errors is strictly statistically correct, but until someone tells me a better way to do it that's the way it's going to be. To get the old error use the command `SET ERR_ZERO OFF`.

When calculating an efficiency by dividing 2 plots, the error was underestimated for small numbers of entries. I now use the same form of the error calculation as Paul Avery derived for MULFIT. This form is still not completely statistically correct and will be the subject of a later note.

Finally I will give a brief summary of some of the commands relating to how errors are treated and displayed:

`SET ERR_ZERO ON`	Sets errors on zero bins to 1 for `ADD` etc. and `FIT`.
`SET ERR_ZERO OFF`	Sets errors on zero bins to 0 for `ADD` etc. and `FIT`.
`HIST ERROR id nmode`	Recalculates the errors for a plot. There are 3 modes:
`nmode = 0`	Zero errors on all points,

`nmode = 1`	Errors are square root of the number of entries,
`nmode = 2`	Same as 1 except error on zero entries is 1.
`SET SHOW_ZERO ON`	Zero points with zero errors are shown (now default).
`SET SHOW_ZERO OFF`	Zero points with zero errors are not shown (used to be default).
`HIST DUMP id`	Dumps the contents of a plot, so you can see what the errors are.
`DUMP`	A MINUIT command that dumps each point and its χ^2 or likelihood contribution to the overall χ^2 or likelihood.

1.4 Ntuple Manipulation

When talking about Ntuples in Mn_Fit most commands apply to Ntuples, scatter plots, series of points, tables and multi-dimensional histograms (including 2-d). There are a number of commands available for manipulating them all. See section 4.66 on page 86 (NTUPLE) for details. You can define cuts, which are either simple expressions (variable condition value), more complicated expressions (expression condition expression), or a COMIS function filename. These cuts can then be applied using the `NTUPLE PROJECT` or `NTUPLE PLOT` commands and you can either project onto either a 1 or 2-dimensional histogram or a scatter plot. You can either specify what the binning should be for the resultant plot or let the program do it for you. It is possible to use a variable as a weight or error on the weight for each point. I have used this when measuring a 2-dimensional surface. The measurements were stored in a table as `x,y,z,dx,dy,dz` and read in using `DAT_FETCH` with an `NTUPLE` card.

It is also possible to plot Ntuples directly. This is most useful if you have read in a table of numbers (using the `DAT_FETCH` command for example) and want to plot one vs. the other without applying any cuts.

If you specify a COMIS function as a cut (`CUT FILE` command), and the file does not exist a skeleton will be made. If you give the identifier of an existing Ntuple as part of the command then the variable names will be used in the skeleton, making it easy for you to specify your cuts.

It is also possible to merge several Ntuples together, which have been created by separate jobs for example. Finally you can call a COMIS subroutine for each event in the Ntuple (`NTUPLE SCAN`) and can make whatever plots or calculations you wish there.

It should be trivial to adapt COMIS functions which have been used in PAW to Mn_Fit. Although KUIP vectors are not available, Mn_Fit registers and user variables are, so you can use these instead.

Ntuple commands are still being developed and so there are likely to be additions and improvements to them in the future. Suggestions for new commands or how to make existing commands easier to use or more flexible are welcome.

1.5 Command Files or Macros

Any Mn_Fit commands can be put into a file and executed instead of being typed in. Files are run using the `EXECUTE` command. You can also `DEFINE` new commands to be a list of existing Mn_Fit commands. These definitions are stored in memory and so should execute quicker than files. The same facilities exist for `DEFINE`'d commands as for macros.

There are a number of extra facilities available in macros. Parameters can be passed to the macro either on the command line or the first time a parameter is encountered in a file you will be prompted for its value. Up to 9 parameters can be passed and you refer to them in the file by preceding the number with an **@**. Direct string substitution takes place as each line is executed, therefore you can even construct histogram numbers in a macro or pass the name of another macro to be executed. It also means that you can put in alternative returns from a macro by giving a parameter as a command and waiting for the prompt before deciding if you want to continue executing the file. See section 4.36 on page 61 (EXECUTE) for examples.

It is also possible to construct `DO` loops inside a macro. The `DO` loop variable must be a single letter and it can be used inside the loop by preceding the letter by a **@**. `DO` loops are terminated by an **ENDDO** command. Please note that there has been no attempt to make the `DO` loops execute quickly; each line is interpreted in every iteration and the file is rewound for each iteration. However they are extremely useful if you do not try to do too much with them! `DO` loops should execute faster in **DEFINE**'d commands as the file does not have to be rewound and read again every iteration.

Note that the last 100 commands are logged (see section 4.81.13 on page 114 (SHOW LOG)) and can be written to a file with the **WRITE LOG** command. You can also turn on logging of all commands to the file **mn_fit.log** using the command **SET LOG ON**. This is particularly useful for recording sessions if you have problems and want to report them.

1.6 Numbers in Mn_Fit

In most places when Mn_Fit asks for a number, such as a cut, or the bin limits for a new plot, or the axis label parameters, you can give either a number or a register, parameter, etc. You can also define new variable names using the syntax **DEP name = value**. The variable names must be alphanumeric expressions of maximum length 8 characters and can also include **$** and _. Variable names which match one of the names below (e.g. R, ERR), or a FORTRAN intrinsic function are not allowed nor is the name **ALL**. The names are converted to upper case. See section 4.27 on page 56 (DEPOSIT) for more details:

```
e.g. dep pi = 3.14159
     dep r2 = sin(pi/2)
```

These extra variables are stored in registers > 300.

In general if you want to keep the current value of a number use the = sign. You can also add, subtract, multiply or divide the current value by using the syntax **=+1.0, =*R2**, etc. The following lists the possibilities for giving a number:

- A number. e.g. 4, -5.2, 3.2E+02. It is not necessary to give the decimal point for real numbers.

- A register

  Rn or IRn where n can be from 0 to 400.

  Note that you are allowed to fill registers 0-99 with whatever you wish using the **DEPOSIT** command. All register contents are stored as real numbers. The integer format **IR** is used when you want to convert the contents to an integer to put them in a text string (see section 1.11 on page 23 (TEXT)).

Registers 100 and above contain the following information:

101 The result of a `SUM` or `INTEGRATE` command

111 The χ^2 or likelihood from the fit

112 The confidence level of a fit

Registers 201-204 contain the positions of the corners of the current plot in cm:

201 x position left

202 x position right

203 y position bottom

204 y position top

Registers 205-210 contain the limits used for the drawing of each of the axes in plot co-ordinates:

205 x minimum

206 x maximum

207 y minimum

208 y maximum

209 z minimum

210 z maximum

Registers 121 to 200 are filled if you give the command `SET PLOT id [&idb] DEFAULT`:

121 The plot identifier

122 The secondary plot identifier

123 The number of entries (histograms) or points

124 The dimension of the plot (positive for histograms, negative for Ntuples and a series of points).

125 The area under the plot (i.e. sum of weights)

126 The minimum number of entries (weight)

127 The maximum number of entries (weight)

131 The number of bins on x-axis (0 for Ntuples and points)

132 The lower limit of the x-axis

133 The upper limit of the x-axis

134 The number of bins on y-axis (0 for Ntuples and points)

135 The lower limit of the y-axis

136 The upper limit of the y-axis

etc. up to 23rd dimension of an Ntuple.

Registers $>$ 300 contain extra variable names that you have defined. Up to 100 variables are allowed.

- A parameter, its parabolic error, its positive or negative error, or its limits. The syntax to use is:

  Pn(m) for the parameter

  ERRn(m) for the parabolic error on the parameter

ERNn(m) for the negative MINOS error on the parameter

ERPn(m) for the positive MINOS error on the parameter

LOLIMn(m) for the lower limit on the parameter value

HILIMn(m) for the upper limit on the parameter value

where n is the function number and m is the parameter number. If you are fitting you can also use just the MINUIT parameter number e.g. Pn.

- The centre of a bin, the bin width, the contents or the error on the contents (including asymmetric errors). The syntax to use is:

  Xn(m) for the bin centre or the x value of a point

  DXn(m) for half the bin width or the error on the x value

  Yn(m) for the bin contents or the y value of a point

  DYn(m) for the error on the y value

  DNXn(m) for half the bin width or the negative error on the x value

  DNYn(m) for the negative error on the y value

  DPXn(m) for half the bin width or the positive error on the x value

  DPYn(m) for the positive error on the y value

  where n is the plot number (including the optional secondary identifier if needed) and m is the bin number. For 2-dimensional histograms give both bin numbers. e.g. `Xn(1,m)`. However note that it is not possible to use the Y value of a bin, as Y **ALWAYS** means the bin contents for histograms. You can calculate the y value of a particular bin or the y bin width using registers 134, 135 and 136.

- The value of an Ntuple variable. The syntax to use is:

  Xn(m,nvar) or

  Xn(m,tvar) where n is the plot number, m is the event number, nvar is the variable number and tvar is the variable name.

  For cuts and expressions for projections you can just give the variable name.

1.7 Identifiers

Identifiers are numbers associated with a plot. In Mn_Fit there are two identifiers associated with each plot. For some commands, e.g. `INDEX`, `FETCH`, `PLOT`, you can specify a range of plots which the command applies to. The range is delimited by a `:`. You can specify a range of primary or secondary identifiers or both. In general if you give as primary identifier 0, this means all histograms you have read in or created; e.g. `FETCH filename 0` will fetch all the histograms in file `filename`.

It is also possible to use a register, parameter etc. as a histogram identifier or secondary identifier.

For more on secondary identifiers, see section 1.8 on page 20 (Secondary_identifiers).

1.8 Secondary_identifiers

Instead of the usual single identifier for each plot Mn_Fit has two identifiers associated with each plot. This permits significantly more flexibility in numbering your histograms and enables you to group together histograms which are associated with each other. To select a plot with a particular secondary identifier when you are asked for a histogram number use the syntax `id&idb` (e.g. `10&1` to get histogram 10 which has secondary identifier 1). If the secondary identifier is omitted, the default will be used. The default can be changed with the `SET IDB` command. Any plots read in after this command have the new secondary identifier.

It is also possible to use a register, parameter, etc. as a histogram identifier or secondary identifier.

1.8.1 Examples

1. If you fit a histogram and want to store the function as a plot, it will be given the same primary identifier as the histogram you are fitting, but a different secondary identifier so that you know with which histogram it is associated. For example if you are fitting histogram 10 issue the commands:

   ```
   DISPLAY
   FUN PLOT 0
   1 -1
   ```

   This will store all the functions you are using in the fit as a smooth curve in histogram 10&1.

2. If you wish to compare Monte Carlo and data histograms, you can read in both sets of histograms, giving them different secondary identifiers, but they will have the same primary identifiers, so that you know which ones to compare and can use the same code to make Monte Carlo and data plots. For example:

   ```
   SET IDB 10
   FET MONTE.HIS 0
   SET IDB 0
   FET DATA.HIS 0
   ```

   The Monte Carlo plots will now all have secondary identifier 10, while the data have secondary identifier 0.

1.9 Expressions

An expression can include any of the parameters which can be used for numbers (see section 1.6 on page 17 (Numbers) for more details). It is a FORTRAN-like calculation including `+,-,*,/,** (or ^), sin, cos, sqrt` etc. Parentheses are also allowed. The code has been adapted from Paul Avery's MULFIT, so all the operations listed there are available, although they have not all been fully tested. If you want to use an expression in a cut or as a variable to project onto I recommend that you enclose it in parentheses. The following operations are available:

```
+, -, *, /, ** (or ^)
SQRT, SIN, COS, TAN, ALOG, ALOG10, EXP, ASIN, ACOS, ATAN
ABS, INT, NINT
```

Note that the results of INT and NINT are converted back to floating point immediately. In addition the following operations on functions are available, although these have not been fully tested, so use with care:

```
FPOS, FNEG, DFPOS, DFNEG, DDFPOS, DDFNEG, FINT
```

where **POS** and **NEG** mean the value of the function at the given value of **X** or just below it. **DFPOS, DFNEG** are the first derivatives of the function and **DDFPOS, DDFNEG** are the second derivatives. 'FINT' is the integral of the function between 2 limits. For the derivative the step size is currently set to 1.0. In the future it will be possible to set it using the **SET FUNCTION STEP** command. The number of intervals to integrate over will be set using the command **SET FUNCTION INTEGRATE npnt** (current default is 100). The function number can be that of a single function, or **0** to mean all currently selected functions. The syntax is **FPOSn(x)** where **n** is the function number and **x** is the value at which it should be evaluated.

1.10 Symbols

For each type of plot a default symbol is defined (which you get if you give the command **SET SYMBOL 0**):

```
Symbol  1 for 1-d histograms
       -1 for a series of points without errors
      -32 for a series of points with errors
       12 for 2-d histograms
        1 for scatter plots
```

To get a picture of the available symbols issue the command **EXEC MN_FIT_HELP:SYMBOL.MNF** (Vax) or **exec ~/mn_fit_help/symbol.mnf** (Apollo). This picture is in Appendix B of the Mn_Fit manual.

In HIGZ/GKS versions hatching and patterns are available. You can specify the type using the commands **SET HATCH** and **SET PATTERN**.

See section 4.79.24 on page 99 (SET HATCH) for details on some of the hatching available and also the figures in Appendix B. Also see the HIGZ/PAW documentation for HIGZ hatchings, and the GKS device documentation. As far as I know patterns are only available with DECGKS.

The following symbols are available for histograms:

-1	Solid line joining the centres of the bins
-2	Dashed line joining the centres of the bins
-3	Dotted line joining the centres of the bins
-4	Dash-dot line joining the centres of the bins
-5	HIGZ line style 12 - a dashed line
-6	HIGZ line style 13 - a dash-dot line
-7	HIGZ line style 14 - a widely spaced dotted line

-8	HIGZ line style 15 - a dotted line
1	Solid line histogram mode
2	Dashed line histogram mode
3	Dotted line histogram mode
4	Dash-dot line histogram mode
5	HIGZ line style 12 histogram mode
6	HIGZ line style 13 histogram mode
7	HIGZ line style 14 histogram mode
8	HIGZ line style 15 histogram mode
10	Dot
11	Circle (really an octagon!)
12	Square
13	Triangle
14	Inverted triangle
15	Diamond
16	Plus (+)
17	Cross (x)
18	Asterix (*)
20-28	Show x error bars for histograms
30-38	Show y error bars
40-48	Show x and y error bars
-n	Show symbol filled

The following symbols are available for scatter plots:

-1	Joins the points with a solid line
-2	Joins the points with a dashed line
-3	Joins the points with a dotted line
-4	Joins the points with a dash-dot line
-5	HIGZ line style 12 - a dashed line
-6	HIGZ line style 13 - a dash-dot line
-7	HIGZ line style 14 - a widely spaced dotted line
-8	HIGZ line style 15 - a dotted line
1-10	One dot per point
10-18	As for histograms
20-48	As for 10-18
-n	Show symbol filled

The following symbols are available for 2-dimensional histograms:

-1	.,1,2,3,...X,Y,Z
-2	Number of entries i.e. table form
1-10	Randomized dots, where the number of dots is equal to the number of entries
10-18	Area of symbol is proportional to number of entries in the bin for scatter plots
20-48	As for 10-18
-n	Show symbol filled

1.11 Text

If you are required to give a title, comment, key, or axis label the text should be enclosed in single quotes, if you want to also put other parameters on the same line. If you omit the quotes the whole of the rest of the line will be used. This enables you to put the parameters associated with the text (e.g. comment position) on the same command line:

```
COMMENT id new 'This is a comment positioned at 10,10' 10.0 10.0
```

Text which is always a single word (parameter names, filenames etc.) should not be included in quotes. If you want to pass text as a parameter to a macro, enclose the whole string in double quotes:

```
EXEC test "'Text to pass'"
```

If you want to pass or give a null string (for example set the axis label back to a blank), give 2 single quotes:

```
Give axis label: ''
```

A <CR> always keeps the current text.

As well as giving normal text (see section 1.12 on page 24 (FONTS) for details of the fonts available), it is also possible to include the values of parameters or registers etc. in a text string. The format to use is:

```
Text {parameter[,format]} MORE TEXT
```

i.e. a section enclosed in braces { } signifies that this part should be translated into text. **Parameter** can be a number, register, parameter, bin contents etc. (see section 1.6 on page 17 (Numbers) for what is available). **Format** is a FORTRAN format statement including parentheses. If you omit the format statement the default format is **(1PG12.5)** and **(I8)** for real and integer numbers respectively. Integers are integer registers (syntax **IR**) and any user variable name that starts with **I**. **NINT** is used to convert real numbers to integers.

This form of text string applies to comments, keys, axis labels, titles etc. To include a normal { in a text string, precede it by a **@**.

1.11.1 Examples

1. Two examples of how to include the contents of registers and parameters in a text string:

```
The cat took {IR3,(I3)} weeks to eat the dog.
The parameter limits are {LOLIM2(3)} {HILIM2(3),('and',1PG11.4)}.
```

1.12 Fonts

See section 1.11 on page 23 (TEXT) for details on how to include registers, parameters or plot content values inside text strings.

Font 0 uses the IGTEXT routine (formerly HPLSOF). It is the default and is always available. A list of the characters available is given in Appendix B of the Mn_Fit write-up and in the HPLOT and HIGZ write-ups (see subtopic IGTEXT).

In GKS the fonts available depend on the workstation - see details in the GKS implementation manual. In the GKSGRAL implementation of GKS fonts -1 to -11, -101 to -111, -201 to -211, and -301 to -311 are available for normal text, plus fonts -13, -113, -213, -313 for Greek characters, and -51, -151, -251, -351 which are solid filled. These fonts are all with precision 2. See Appendix B for examples of the fonts available. In addition all the other fonts listed in the PAW manual (section 8.10 - Text Fonts) are available.

Within PLTSUB there is only one other font (the PLTSUB SYMBOL routine).

The font and precision are specified with the form `spfff` where `s` is the sign of the font, `p` is the precision and `fff` is the font (e.g. -2013 to get font -13 with precision 2). Use the **SET FONT** command to change the general font, or you can set the font used in the **COMMENT, KEY, SET SCALE, SET LABEL** and **SET TITLE** commands.

1.12.1 IGTEXT

There are a number of changes from the standard IGTEXT input in the way you give a text string to make it easier and quicker to get what you want. Upper and lower case characters should be entered as you want them to be, including Greek and special characters. To switch modes precede the string you want in a different mode by the relevant escape character given below. You will stay in the new mode until you give the termination character. The exception is `@` which is only valid for the next character.

The following escape characters are used as in IGTEXT:

`[`	`go to Greek`
`]`	`end of Greek`
`"`	`go to special symbols`
`#`	`end of special symbols`
`^`	`go to superscript`
`?`	`go to subscript`
`!`	`end of superscript or subscript`
`&`	`backspace one character`
`$`	`termination character (do not use in your strings)`
`@`	`to get any of the escape characters (if they are in the IGTEXT character set)`

The following characters can be entered directly:

`a,b...z`	`lowercase alphabetic`
`A,B...Z`	`uppercase alphabetic`
`0,1...9`	`the digits`
`,`	`comma`

.	period
;	semicolon
:	colon
+	plus sign
-	minus sign
*	star
/	slash
=	equals
_	underscore
|	vertical bar
'	quote or apostrophe
< >	less than, greater than
( )	left/right parentheses
{ }	curly brackets

The following sequences will be printed as a single character:

=<	less than or equal to
>=	greater than or equal to
->	right arrow
<-	left arrow
+/-	plus or minus

As the $<$ and $>$ signs can be typed in directly, you cannot use them for switching between lower and upper case. This only affects getting small versions of digits. You can force a switch by preceding the symbol by an **@**.

Examples

@<1234@> will get you 1234 and
[p]^+![p]^-! Invariant Mass (GeV/c^2!)@! will get you $\pi^+\pi^-$ Invariant Mass (GeV/c^2)!

1.13 Using_COMIS

COMIS is the COMpilation and Interpretation System that enables you to write and execute FORTRAN functions without having to relink your program.

COMIS is used in several places in Mn_Fit. You can use COMIS to write a function to fit to (see section 4.41.9 on page 72 (FUNCTION LIST COMIS) for more details), as a cut when projecting an Ntuple or n-dimensional histogram

(see section 4.66.4 on page 87 (NTUPLE PROJECT)), as a subroutine which is called for every event of an Ntuple (see section 4.66.5 on page 89 (NTUPLE SCAN)), or as a subroutine which you can just call (see section 4.10 on page 45 (CALL_COMIS)).

In all cases if the filename you give does not exist a skeleton file will be written and you can then edit it. You can also just invoke COMIS by giving the command **COMIS** if you have something special to do (see section 4.15 on page 46 (COMIS)), although the **CALL_COMIS** command is probably better for all the purposes I have been able to think of.

See section 4.66 on page 86 (NTUPLE) for details of the common blocks available when you are looking at Ntuples.

From any COMIS function or subroutine you can call many CERNLIB subroutines. These routines are the same as in PAW with a few additions:

```
Mn_Fit:
  XMNCLC,XMNCNT,XMNFRG,XMNHIS,XMNRD3,WMNRES,XMNC2D,XMNDFUN
HBOOK:
  HBOOK1,HBOOK2,HBOOKN,HFILL,HF1,HPRINT,HDELET,HRESET,HEXIST
  HFITGA,HFITPO,HFITEX,HPROJ1,HPROJ2,HFN,HGNPAR,HROPEN,HCDIR,HGIVEN
  HPAK,HPAKE,HUNPAK,HGIVE,HGN,HGNF,HF2,HFF1,HFF2
  HRIN,HROUT,HI,HIE,HIX,HIJ,HIDALL,HNOENT,HX,HXY
  HCOPY,HSTATI,HBPROF,HOPERA,HIDOPT,HDERIV
  HBFUN1,HBFUN2,HRNDM1,HRNDM2,HBARX,HBARY
HPLOT:
  HPLOT,HPLSYM,HPLERR,HPLEGO,HPLNT,HPLSUR,HPLSOF,HPLSET,HPLGIV,HPLOC
KUIP:
  KUGETV,KUDPAR,KUVECT,KILEXP,KUTIME
HIGZ:
  IPL,IPM,IFA,IGTEXT,IGBOX,IGAXIS,IGPIE,IGRAPH,IGHIST
  IGARC,IGLABL,IGRNG,IGMETA,IGSA,IGSET,IRQLC,IRQST
  ISELNT,ISFAIS,ISFASI,ISLN,ISMK,ISVP,ISWN,ITX,ICLRWK
PAW:
  PAOPEN,PACLOS,PAREAD,PAWRIT
ZEBRA:
  FZIN,FZOUT,FZENDI,FZENDO
  RZCDIR,RZLDIR,RZFILE,RZEND,RZIN,RZOUT,RZVIN,RZVOUT
CERNLIB:
  VZERO,UCOPY,RNDM,RANNOR,LENOCC
  SBIT0,SBIT1,SBYT,JBIT,JBYT,UCTOH,UHTOC
  DENLAN,BINSIZ
```

1.14 Examples

Appendix C contains some demonstration Mn_Fit macros, which have also been used to make the figures there. These macros are in the directory referred to by the logical name **mn_fit_help** (Vax) or by the link **~/mn_fit_help** (Apollo). The macros used to produce the figures in Appendix B are also available in the same directory. To execute these files give the command **exec mn_fit_help:filename** (Vax) or **exec ~/mn_fit_help/filename** (Apollo):

demonn.mnf	- Demonstration macro number nn (nn = 01, 02 ...)
symbol.mnf	- Shows the available symbols
font.mnf	- Shows a sample of the available fonts in GKS
hatch1.mnf	- Shows GKSGRAL hatchings
hatch2.mnf	- Shows HIGZ hatchings

If these files are not available they can be found in the CMZ file **mn_fit_cmz:mn_util.cmz** (Vax) or **~/mn_fit_cmz/mn_util/cmz** in the directory **//mn_util/demo**. To make the files startup CMZ and give the following commands:

```
* Vax
  file mn_fit_cmz:mn_util -r
  sel VAX
* Apollo
  file ~/mn_fit_cmz/mn_util -r
  sel APOLLO
*
  set *.mnf -d
  ctot -y demo
```

A series of test files are available in the directory referred to by the logical name **mn_fit_test** (Vax) or by the link **~/mn_fit_test** (Apollo). Be aware that these files reserve the right to delete any plots or functions you have already read in or defined. To execute the files give the command **exec mn_fit_test:filename** (Vax) or **exec ~/mn_fit_test/filename** (Apollo). The following files are available:

```
all_test.mnf      - Runs all of the following command files
fetch_test.mnf    - Tests all the fetch commands
store_test.mnf    - Tests the various store commands
plot_test.mnf     - Tests lots of the plotting features
slice_test.mnf    - Tests the ability to extract and plot HBOOK slices etc.
fit_test.mnf      - Tests the fitting
do_test.mnf       - Tests DO loops
define_test.mnf   - Tests the defining of new commands
comis_test.mnf    - Tests the COMIS interface
define_test.mnf   - Test defining new commands
alias_test.mnf    - Tests defining aliases
squeeze_test.mnf  - Tests recovery of unused space
cut_test.mnf      - Tests cut and project to apply cuts on plots
cut2_test.mnf     - Tests cuts involving expressions and COMIS functions
merge_test.mnf    - Tests merging Ntuples
scan_test.mnf     - Tests scanning Ntuples
call_test.mnf     - Tests calling a COMIS function or subroutine
oper_test.mnf     - Tests the plot operation commands: partition, add etc.
eff_test.mnf      - Tests the efficiency calculations
calc_test         - Tests out calculations, use of registers, parameters etc.
var_test.mnf      - Tests defining user variables
spline_test.mnf   - Test spline fitting and smoothing (not yet written)
draw_test.mnf     - Tests out drawing lines etc.
symb_test.mnf     - Tests out the available symbols
hatch_test.mnf    - Shows the available hatchings
pattern_test.mnf  - Shows the available patterns
font_test.mnf     - Tests out changing fonts
```

If you are not at Cornell or CERN and do not have the files please ask me to send them to you (BROCK@VXCERN.CERN.CH). These are also the files used to check out Mn_Fit, so if you have any problems please let me know. Note that there are some errors built in to test out error handling. Read `//mn_util/doc/testing` for details of what to expect.

1.15 Screen_devices

The screen devices that can be referred to by name depend on the graphics package used. Below I list most of the devices that are available. The letters following the name indicate which graphics package they are available in. (G = GKSGRAL, V = VAXGKS, DG = DEC GKS-3D, DE = Decstations, DI = DI3000, P = PLTSUB). You can use any other device by giving its number (see the relevant manual). You can find the complete list of devices available by asking for help when you start up Mn_Fit or giving the command `CAPTURE ?`.

When you start up Mn_Fit you will be prompted for which device you want to use. Give a ? when asked for the screen device at the beginning of Mn_Fit to find out which ones are available. If your Tektronix is another device you must give the logical name for the device. In addition you must be able to allocate the device for it to work.

For DI3000 you must define which devices you want to have available before you start Mn_Fit. You have to check your local DI3000 documentation to find out what devices you can define. Help inside Mn_Fit will then tell you what you have defined.

Device `None` is useful if you want Mn_Fit to go through the motions of making a picture without actually making one, as in `FUNCTION HIST` if you just want the histogram without looking at it:

None	All	Capture a null device.
Tektronix	G/V/DG/DI/P	Tektronix screen. You will be prompted for type and the logical name if it is another device.
T10	P	10 bit resolution Tektronix
T12	P	12 bit resolution Tektronix
Vaxstation	G/V/DG/DI/P	Opens a UIS window
VS2000	DG	Vaxstation 2000
VS3200	DG	Vaxstation 3200
VS3500	DG	Vaxstation 3500
VSII	DG	Vaxstation II monochrome
CVSII	DG	Vaxstation II/GPX colour
Decwindows	V/DG/DE/P	Decwindows input and output device
VT240	G/V/DG/P	VT240 graphics terminal
CVT240	DG	VT240 colour
VT340	G/DG	VT340 graphics terminal
Pericom	G	Pericom graphics terminal
Falco	G/P	Falco graphics terminal
Apollo	G	Apollo monochrome workstation
CApollo	G	Apollo colour workstation
Grinnell	P	Grinnell screen
nnnn	G/V/DG/DI/P	The workstation type number

1.16 Hardcopy_devices

The hardcopy devices that can be referred to by name depend on the graphics package used. Below I list most of the devices that are available. The letters following the name indicate which graphics package they are available in. (G = GKSGRAL, V = VAXGKS, DG = DEC GKS-3D, DE = Decstations, DI = DI3000, P = PLTSUB). The default filename is also given. You can use any other device by giving its number (see the relevant manual). When you give the command `HARDCOPY` you are prompted for the device you want to use. Use the `SET HARDCOPY` command to change the output filename for hardcopies if you want to (this is only implemented in GKS versions).

If you are reading commands from a file and omit the device a `METAFILE` will be assumed. The following hardcopy devices are supported:

Metafile	G/V/DG/DE	plot.meta	GKS Metafile
Postscript	G/V/DG/DE/DI/P	plot.ps	Postscript file (portrait)
EPostscript	G/DE	plot.eps	Encapsulated Postscript file
LPostscript	G/DE	plot.ps	Postscript file (landscape)
Tektronix	G	plot.tek	Tektronix 4014 file
Tektronix	P	plot.t12	12 bit for a real Tektronix
Talaris	P	plot.qms	Talaris laserprinter file
Imagen	P	plot.tek	Imagen printer
Versatec	P	plot.vrs	Versatec printer
LN03	P	plot.t12	LN03 plus printer (Tektronix)

1.17 Condition_handler

A condition handler has been installed in Mn_Fit which traps errors and `CTRL/C` interrupts. If an error is detected control is returned to the `MN_CMD>` level. This means that overflows which sometimes happen during `IMPROVE` should not cause the program to exit. It will return to the `MN_CMD>` level instead. Note that the condition handler is only installed on the Vax at present. WARNING: A `CTRL/C` during FORTRAN I/O can cause strange effects. Hence for most read and writes the condition handler is disabled. To suppress the output during an `INDEX` command for example use `CTRL/O` and then you can try `CTRL/C` with care.

Chapter 2

Menu of all the Available Commands

2.1 Menu of Top Level Commands

The following is a list of the available commands. For more information see the detailed HELP on each command. For the MINUIT commands see section 3 on page 39 (MINUIT).

EXIT	Exits from program or back to previous level.
QUIT	Exits from program.
ADD	Adds 2 histograms together.
ALIAS	Defines an alias.
UNALIAS	Undefines one or all aliases.
ATTACH	Attaches you to another process.
AVE_FETCH	Fetches an AVEHST format histogram (ASCII files only).
BOOK	Books a new plot inside Mn_Fit (alias for `HIST BOOK`).
CALCULATE	Calculates an expression, changes a register, parameter etc., or defines a user variable.
CALL_COMIS	Calls a COMIS subroutine.
CAPTURE	Captures a new output device.
CDIRECTORY	Changes the current HBOOK directory.
CLEAR	Clears the screen device display.
CLOSE	Closes the hardcopy devices output files.
COMIS	Invokes the COMpilation and Interpretation System package.
COMMENT	Adds a comment to a plot.
COPY	Copies one histogram to another one.
CUT	Defines cuts for Ntuples and multi-dimensional histograms (see section 4.18 on page 47 (CUT) for more).

NO_CUT	Removes cuts from a multi-dimensional histogram.
DAT_FETCH	Reads a series of data points from a file (alias for `READ DATA`).
DAT_STORE	Stores a series of data points in a file (alias for `WRITE DATA`).
DATABASE	Interface to plotting contents of L3 database.
DB_HISTORY	Plots the history of an element in L3 database.
DB_SNAP	Plots the contents of an L3 database directory.
DEFINE	Defines a new command.
UNDEFINE	Deletes a command you made with the `DEFINE` command.
DELETE	Deletes one or more plots.
DEPOSIT	Calculates an expression, changes a register, parameter etc., or defines a user variable.
DISPLAY	Makes a special display - only in L3 display version.
DIVIDE	Divides 2 histograms.
DO	Starts a `DO` loop (only available inside macro's).
DRAW	Draws a line, arrow, box etc. on a picture.
EDIT	Edits a file (use `SET EDIT` to select your favourite editor).
EFFICIENCY	Divides 2 histograms using binomial errors.
ENDDO	Terminates a `DO` loop.
EXAMINE	Examines the contents of a register, parameter, bin etc.
EXECUTE	Reads a list of commands from a file (alias for `READ COMMAND`).
EXTRACT	Extracts part of an HBOOK histogram (alias for `HIST EXTRACT`).
FETCH	Fetches one or more HBOOK version 4 histograms.
FILL	Fills a plot inside Mn_Fit (alias for `HIST FILL`).
FIT	Fits a plot (only valid in `MN_CMD>` level).
FUNCTION	Do things with functions (see section 4.41 on page 66 (FUNCTION) for more).
HARDCOPY	Makes a hardcopy of the last picture.
HB3_FETCH	Fetches one or more HBOOK version 3 histograms.
HB_FETCH	Fetches one or more HBOOK version 4 histograms.
HB_OPEN	Opens an HBOOK version 4 histogram file.
HB_STORE	Stores one or more HBOOK version 4 histograms in a file.
HB_MN_FIT	Converts HBOOK plots to Mn_Fit plots.
HCOPY	Copies one HBOOK histogram to another one (HCOPY).
HDELETE	Deletes an HBOOK histogram.
HMAKE	Makes an HBOOK histogram from another sort.

HINDEX	Gives the HBOOK index for histogram currently in core.
HISTOGRAM	Plots, overlays or dumps a hist (see section 4.51 on page 77 (HISTOGRAM) for more information).
HRENAME	Renames an HBOOK histogram.
HY_FETCH	Fetches one or more HYBRID histograms.
INDEX	Gives an index of the histograms read in.
INQUIRE	Gets the value for a macro parameter (only available inside macros).
INTEGRATE	Integrates a function over a range.
KEY	Gives a key for the meaning of symbols.
LDIRECTORY	Lists the contents of the current HBOOK directory.
LEGO	Makes a lego plot of a 2-dimensional histogram.
MDIRECTORY	Makes a new HBOOK directory in memory.
MESSAGE	Writes a message to the current output device.
MN_FETCH	Reads in a multi-dimensional histogram.
MN_STORE	Stores a multi-dimensional histogram.
MULTIPLY	Multiplies 2 histograms.
NORMALIZE	Normalizes the total contents of a histogram.
NTUPLE	Operations on Ntuples and multi-dimensional histograms (see section 4.66 on page 86 (NTUPLE) for more).
OVERLAY	Overlays one histogram on another (alias for `HIST OVERLAY`).
PARTITION	Cuts out part of a histogram.
PLOT	Plots a histogram (alias for `HIST PLOT`).
PRINT	Prints a histogram on the terminal or to a file.
PROJECT	Makes a projection of an Ntuple or n-dimensional histogram (alias for `NTUPLE PROJECT`).
QUIT	Exits the program.
READ	Reads in commands or data (see section 4.72 on page 91 (READ) for more).
REBIN	Rebins a histogram.
REDRAW	Redraws the last picture.
REMOVE	Deletes a user defined variable.
RENAME	Renumbers a histogram.
RETURN	Returns from a macro.
SCALE	Multiplies the contents of a histogram by a scale factor.
SCT_FETCH	Fetches an AVEHST true scatter plot.
SET	Changes a plotting parameter (see section 4.79 on page 92 (SET) for more).

SHELL	Executes a single DCL command or spawns a subprocess (Vax). Executes a shell command or starts a new shell (Apollo).
SHOW	Shows plot drawing parameters (see section 4.81 on page 113 (SHOW) for more).
SMOOTH	Smoothes a histogram.
SPAWN	Executes a single DCL command or spawns a subprocess (Vax). Executes a shell command or starts a new shell (Apollo).
SPLINE	Spline fits a histogram.
SURFACE	Makes a surface plot of a 2-dimensional histogram.
STORE	Stores one or more HBOOK histograms in a file.
SQUEEZE	Gets back unused space for Mn_Fit.
SUBTRACT	Subtracts 2 histograms.
SUM	Sums the contents of a plot.
TITLE	Gives a new title to a histogram.
WAIT	Stops Mn_Fit for a time.
WINDOW	Draws several plots on one page.
NO_WINDOW	Goes back to one plot per page.
WRITE	Writes out a list of commands or data points.

2.2 Menu of Commands Inside CUT

The following is the list of `CUT` commands. For more details see the individual topics.

NEW	Specifies a new cut.
CHANGE	Changes an existing cut.
FILE	Gives a COMIS filename to be used as a cut function.
EDIT	Edits and then compiles a COMIS cut.
COMPILE	Compiles a COMIS cut.
DELETE	Deletes one or more cuts.
LIST	Lists the currently defined cuts.
USE	Specifies the cuts to use when making a projection.
END	Ends a list of cut commands.

2.3 Menu of Commands Inside FUNCTION

The following is a list of the available commands. For more information see the detailed HELP on each command:

ADD	Adds a new function.

COMPILE	Compiles a COMIS function.
DELETE	Deletes a function from the list.
EDIT	Enables you to edit and recompile a COMIS function.
FETCH	Fetches a function that has been stored in a file.
HISTOGRAM	Stores the fit in a histogram.
HOVERLAY	Overlays a function as a histogram on a plot.
INFO	Gets information on the functions used.
LIST	Lists the predefined functions that you can add.
OVERLAY	Overlays a function on a plot.
PLOT	Plots one or more of the functions.
STORE	Stores a function in a file.
USE	Uses a function in the fit or not.

2.4 Menu of Commands Inside HISTOGRAM

The following is a list of the available commands. For more information see the detailed HELP on each command:

BOOK	Books a new plot inside Mn_Fit.
DISPLAY	Makes an L3 detector display - only in display version.
DUMP	Dumps the contents of a histogram.
ERRORS	Changes the errors of a plot.
EXTRACT	Extracts part of an HBOOK histogram e.g. BANX or SLIY or FUN.
FILL	Fills a new plot inside Mn_Fit.
LEGO	Makes a lego plot of a 2-dimensional histogram.
OVERLAY	Overlays a histogram on the present plot.
PLOT	Plots a histogram.
SURFACE	Makes a surface plot of a 2-dimensional histogram.

2.5 Menu of Commands Inside NTUPLE

The following is a list of the available commands. For more information see the detailed help on each topic.

DUMP	Dumps the contents of an Ntuple.
MERGE	Merges Ntuples from different files into a single Ntuple.
PLOT	Projects and plots the projection of an Ntuple.
PROJECT	Projects an Ntuple.
SCAN	Calls a Comis subroutine for every Ntuple event.

2.6 Menu of Commands Inside READ

The following is a list of the available commands. For more information see the detailed HELP on each command:

COMMAND	Reads a series of commands and execute them from a file.
DATA	Reads data points to be fit from a file.

2.7 Menu of Commands Inside SET

The following is a list of the available commands. For more information see the detailed HELP on each command.

EXIT	Exits from SET.
ENDSET	Exits from SET (use instead of EXIT in command files).
X	Sets a variable for the X-axis.
Y	Sets a variable for the Y-axis.
Z	Sets a variable for the Z-axis.
ALIAS	Turns on or off alias translation.
AUTOSWITCH	Switches between plotting and alphanumeric without asking for a <CR>.
AXIS	Sets axis labelling text, position, size and angle.
BACKGROUND	Specifies which functions should be considered as background.
BOX	Turns on or off drawing a box round the plot.
COLOUR	Changes the colour of all or part of a picture.
DEBUG	Turns on/off the debug flag.
DEFAULT	Sets parameters back to default values or sets the default plot.
DISPLAY	Changes the mode or defaults for displaying fit results.
DIRECTORY	Sets the current HBOOK4 directory.
DSIZE	Sets the scale factor for the dot size.
DUMP	Changes the output unit for dump commands.
ECHO	Turns on or off echoing commands when reading from a file.
EDIT	Gives the command to invoke your favourite editor.
ERR_ZERO	Sets the error on zero points to 1 or not.
EXCLUSIONS	Turns on or off showing excluded parts of functions.
FIT	Specifies options for fitting.
FONT	Sets the font for text in pictures.
FRAME	Sets where you want a frame round the plot.
GRID	Turns on or off a grid.

HATCH	Sets the hatching number (only works with GKS).
HARDCOPY	Sets the name of the file that hardcopies are written to.
HEADER	Turns on or off displaying the histogram identifiers on the picture.
HISTOGRAM	Sets the default histogram number.
IDB	Specifies the default secondary identifier.
IDSIZE	Sets size of text for histogram identifier.
IDSHOW	Turns on or off displaying the histogram identifiers on the picture.
LABEL	Sets axis labelling text, position, size and angle.
LOG	Turns on/off logging of all commands.
MANUAL	Turns on/off special command echoing used to make Mn_Fit manual.
NEXT_WINDOW	Sets the window for the next plot.
NTUPLE	Specifies the axes corresponding to Ntuple variables.
NORMALIZE	Turns on or off an overall normalization factor when fitting.
ORDER	Specifies the order of variables in a file read with `DAT_FETCH`.
ORTHOGONAL	Sets the orthogonality limits for Chebyshev polynomials etc.
PARAMETER	Sets parameters for the L3 detector displays.
PATTERN	Sets the pattern number (only works with GKS).
PLOT	Specifies which plot you want to change something for.
RATIO	Specifies using ratio of areas or not when fitting more than 1 plot.
RECL	Sets the record length for HBOOK version 4 files.
REDRAW	Redraws the last picture with the `SET` changes on it.
SCALE	Sets where to put the scale and its position and size.
SECONDARY_ID	Specifies the default secondary identifier.
SHELL	Specifies the shell to be used on the Apollo.
SHOW_ZERO	Shows zero points which have zero errors or not.
SIGNAL	Specifies which functions should be considered as signal.
SSIZE	Sets the symbol size.
STATISTICS	Specifies whether HBOOK or Mn_Fit calculation of means and sigmas should be used.
SYMBOL	Sets the symbol number.
TEXT	Turns on or off showing fit information on the screen.
THICKNESS	Changes the line thickness for all or part of a picture.
TICKS	Sets number of ticks and where to put them.
TIME	Sets the time mode for storing data and the reference time.
TITLE	Changes the title for a histogram.

TSIZE	Sets the title size.
USIZE	Sets the size for printing the fit results.
WAIT_CR	Turns on/off waiting for a <CR> before the next picture.
WINDOW	Sets up windowing or changes the window for a particular plot.
NO_WINDOW	Turns off windowing.

Parameters specific to an axis. These must be preceded by the axis you are referring to:

AXIS	Sets axis labelling text and position, size and angle.
GRID	Turns on or off a grid.
HSIZE	Sets the histogram size.
LABEL	Sets axis labelling text and position, size and angle.
LIMITS	Sets lower and upper limits for axis.
MARGIN	Sets size of lefthand or lower margin.
MODE	Sets mode to draw axis numbers, **Real**, **Integer** or Log.
NULL	Turns on or off drawing a line at x=0 or y=0.
OPT_ZERO	Turns on or off having 0 as the lower scale limit for entries.
PSIZE	Sets the overall size of the plot.
SCALE	Sets variables for scale.
SIZE	Sets the histogram size.
TICKS	Sets number of ticks.
WMARGIN	Changes the margin inside a window.
WSIZE	Changes the size of a plot inside a window.
ZERO	Turns on or off drawing a line at x=0 or y=0.

2.8 Menu of Commands Inside SHOW

The following is a list of the available commands for which there are not SET equivalents. For more information see the detailed HELP on each command:

ALL	Shows everything there is to show.
ALIAS	Lists the currently defined aliases.
COMMANDS	Lists the internally defined commands (made with DEFINE).
COMMENTS	Lists the current comments.
CONSTRAINT	Lists the current constraints on parameters being fit.
CUTS	Lists the cuts defined and set.
DEFINITION	Lists the internally defined commands (made with DEFINE).
DIRECTORY	Lists the contents of the currently selected HBOOK4 directory.

EXCLUSIONS Lists the parts of histograms excluded from the fit.

FILES Lists the filenames for hardcopy, dump and currently open.

FLAGS Lists the logical flags that control what is plotted.

KEYS Lists the current keys.

LOG Lists the last 100 commands given.

ORDER Lists the expected order of variables when doing DAT_FETCH.

PLOT Lists the plots currently in the buffer for plotting and enables you to list something for a particular plot.

REGISTER Lists the contents of one or more registers (0-400).

SEGMENTS Lists the segments currently in use (mainly for experts).

SIZES Lists the parameters relating to plot sizes, margins, windows.

VARIABLE Gives the value of one or all user defined variables.

2.9 Menu of Commands Inside WRITE

The following is a list of the available commands. For more information see the detailed HELP on each command:

DATA Writes a histogram to a file in card image format.

LOG Writes out the log of commands given at the terminal.

Chapter 3

MINUIT

3.1 Introduction

There are 2 versions of the MINUIT package interfaced to Mn_Fit. The default is now the latest CERN MINUIT (89.12j) and the previous version was based on a much older version of MINUIT and was extensively modified by Chris Rippich. To get this old version, which was the one used in all versions of Mn_Fit up to 3.00, you have to link it at your institute and run Mn_Fit with the option `-c` (short for `cgr_minuit`).

In both versions a number of new commands have been added to give you a graphical display of the fit results and to control which parts of the plots are being fit.

All MINUIT commands have the standard syntax, except `MIGRAD`, `SIMPLEX` and `MINOS` where the maximum number of calls is not part of the command, but can be set using the `MAX_CALLS` command.

The extra MINUIT commands are: `MODIFY`, `DISPLAY`, `EXCLUDE`, `INCLUDE`, `NO_EXCLUDE`, `NO_INCLUDE`, `DUMP`, `ITERATIONS`, `MAX_CALLS`, `PRECISION`, `FLOAT`, `INFO`, `FIT_INFO` and `BACK_SUB`.

In addition there are a few commands in the CERN version which are not available in the C. Rippich version: `CONSTRAIN`, `SCAN`, `STRATEGY`, `RELEASE`. In the CERN version of MINUIT you can give any standard MINUIT command with the standard syntax by prefixing the command with `MINUIT`. e.g. `MINUIT SHOW COV`.

Note that in all MINUIT commands a number can passed as a register, parameter etc. (see section 1.6 on page 17 (Numbers) for the syntax).

The default printout level when you startup MINUIT is 0. This gives quite a lot of information about what is going on. If you are doing repetitive fitting I recommend reducing the printout level to -1 (command `PRINTOUT -1`)

3.2 Menu of MINUIT Commands

The following is a list of the available commands. For more information see the detailed HELP on each command. If you set a parameter to 0 or do not supply it, the default is taken. The following are the standard MINUIT commands:

HELP

CALL_FCN Calls FCN with flag `iflag`.

MAX_CALLS	Sets the maximum number of calls for `SIMPLEX`, `MIGRAD` and `MINOS`.
CONTOUR	Draws a contour plot on the screen.
COVARIANCE	Reads in the covariance matrix.
ERROR_DEF	Sets the χ^2 or likelihood change for the errors.
EXIT	Exits MINUIT.
FIX	Fixes the value of a variable.
FLOAT	Floats a previously fixed variable.
FORCE_ISW	Changes MINUIT status flags - VERY RISKY!! (only in C. Rippich version).
GRADIENT	Indicates that FCN can calculate derivatives - DO NOT USE.
NO_GRADIENT	Turns off `GRADIENT` - DO NOT USE.
HESSE	Recalculates the covariance matrix.
IMPROVE	Tries to find a new minimum somewhere nearby.
INFO	Gives information on the current fit status.
MATOUT	Writes out the covariance matrix.
MIGRAD	Runs MIGRAD.
MINIMIZE	Runs SIMPLEX then MIGRAD.
MINOS	Calculates MINOS errors.
MINUIT	Executes a MINUIT command (CERN version of MINUIT only).
MNCONTOUR	Makes a graphical contour minimizing w.r.t. other parameters.
MODIFY	Changes the value of a parameter, its error or limits.
PAGE	Starts a new page on unit 6.
PRECISION	Sets the convergence criteria.
PRINTOUT	Changes the printout level.
PUNCH	Output goes to punch unit.
RELEASE	Floats a previously fixed variable (CERN version of MINUIT only).
RESTORE	Floats variable(s) which were fixed.
SEEK	Monte Carlo minimization.
SIMPLEX	Runs the fairly crude SIMPLEX minimizer.
STANDARD	Calls subroutine STAND.
STRATEGY	Sets the MINUIT strategy (CERN version of MINUIT only).
STOP	Stops MINUIT.
UNIT	Changes the output unit (do not use in Mn_Fit).

The following commands have been defined by Mn_Fit to give you more information and control on what to fit and the results:

BACK_SUB	Makes a background subtracted plot.
CONSTRAIN	Constrains a parameter w.r.t. other parameters.
UNCONSTRAIN	Removes a constraint.
DISPLAY	Shows the fit results on the softcopy device.
DUMP	Dumps the point by point fit values.
EXCLUDE	Excludes part of histogram for fitting.
NO_EXCLUDE	Removes exclusions.
FCN_DRAW	Draws the FCN vs a variable on the terminal.
FCN_PLOT	Plots the χ^2 or likelihood vs. a variable.
FIT_INFO	Gets information on the fit.
INCLUDE	Includes part of a histogram for fitting.
NO_INCLUDE	Removes inclusions.
ITERATIONS	Draws fit iterations.
NO_ITERATIONS	Stops drawing fit iterations.
PROB_PLOT	Plots the probability vs. a variable (replaces FCN_DRAW).

Chapter 4

Reference Manual for Mn_Fit

4.1 EXIT

```
Syntax: EXIT
```

Exits the current level. If you are in MN_CMD> then you exit the program, otherwise you get back to the last level. e.g. MINUIT> -> MN_CMD Note that the minimum abbreviation for this command is EXI to avoid accidental exits.

4.2 QUIT

```
Syntax: QUIT
```

Exits the program.

4.3 ADD

```
Syntax: ADD id1 [:id1n] [&idb1] id2 [:id2n] [&idb2] id3 [:id3n] [&idb3]
        [scale1] [scale2]                    (default scale = 1.0 1.0)
 where: id1,id2   are the input histogram identifiers
        id3       is the output histogram identifier
        idb1,idb2 are the (optional) input secondary identifiers
        idb3      is the (optional) output secondary identifier
```

Adds two histograms (id1, id2) together to make a third one. To specify the secondary identifier, precede it by a &, otherwise the default will be used. (Use the SET IDB command to change the default). The scale factors are optional. To avoid confusion, you should give a <CR> after the identifiers or make sure the scale factors are given as real numbers.

To add a range of histograms, the primary identifiers you give for the input and output histograms must be the same, but you can specify different secondary identifiers. For example, `ADD 300:400&1 300:400&2 300 :  400 & 10` will add all histograms with primary identifiers 300 to 400 and secondary identifiers 1, to those with secondary identifiers 2, putting the results into histograms with the same primary identifiers and secondary identifier 10. If you give primary identifier 0, the operation will be performed on all plots with the given secondary identifier.

4.4 ALIAS

```
Syntax:   ALIAS name string
where:    name    is the alias name
          string  is the string that the alias defines
```

Defines an alias for a string. If you do not want to continually type a long string or keep macros computer independent it is often useful to define an alias. It is not allowed to define **ALL** as an alias name. When a command line is parsed any defined aliases are searched for. To be recognized as an alias the **name** must be followed by a non-alphanumeric character. If you want to concatenate an alias with another string you can use **//**.

Alias translation can be turned on and off using the command **SET ALIAS ON|OFF**. By default it is on. If you do not want a **name** to be translated as an alias precede it by a **@**. You can undefine an alias using the command **UNALIAS name**, but see the warnings below. You can list one or more aliases using the command **SHOW ALIAS name**. If you omit the **name** all aliases will be listed.

WARNINGS: If alias translation is on and you want to undefine an alias you must use the syntax **UNALIAS @name**. Similarly if you want to redefine an alias use the syntax **ALIAS @name string**, or show the translation use **SHOW ALIAS @name**.

HINT: I find it useful to precede all aliases by a prefix (e.g. **a_** or **$**), so that you cannot accidentally include an alias in a command and get unexpected results.

4.4.1 Examples

1. This example defines an alias **junk**, uses it and then lists all aliases:

```
alias junk show
junk all                   will be replaced by 'show all'
show alias all
unalias @junk
```

2. Shows concatenating aliases and effect of turning alias translation off:

```
alias name1 'John Smith'
alias name2 son
message name1//name2  will print 'John Smithson'
alias name  'Jane Jones'
message name1's real @name is name
                      will print 'John Smith's real name is Jane Jones'
set alias off
message What is your name?
                      will print 'What is your name?'
```

4.5 UNALIAS

```
Syntax:   UNALIAS name|ALL
where:    name  is the alias name
```

Undefines the alias **name** or all aliases if the command **UNALIAS ALL** is given. See section 4.4 on page 43 (ALIAS) for a complete description of the alias mechanism.

4.6 ATTACH

```
Syntax: ATTACH [process_name]
```

Attaches you to another process. If you omit the process name you will be attached to the name you gave last. This command is only valid on the Vax.

4.7 AVE_FETCH

```
Syntax: AVE_FETCH filename id1[:id2] [id3...]
```

Fetches one or more AVEHST histograms from an ASCII file. If you want to fetch all the histograms give the command **AVE_FETCH filename 0**. To fetch a range of histograms give the command **AVE_FETCH filename id1:id2**. If you want to fetch another histogram(s) from the same file give the command **AVE_FETCH id1[:id2] [id3...]**.

4.8 BOOK

```
Syntax: BOOK[/BINNED|/UNBINNED|/ERROR|/NOERR|/ASYMMETRIC] id [&idb]
        title
        ndim
        maxpnt   OR nbinx xlo xhi [nbiny ylo yhi ...]
 where: id     is the plot identifier
        idb    is the (optional) secondary identifier
        title  is the title for the new plot
        ndim   is the number of dimensions
        maxpnt is the maximum number of points (for an unbinned plot)
        nbinx  is the number of x bins
        xlo    is the lower limit on x
        xhi    is the upper limit on x
```

Alias for **HISTOGRAM BOOK**. See section 4.51.1 on page 77 (HISTOGRAM BOOK) for more details.

4.9 CALCULATE

```
Syntax: CALCULATE [parameter =] expression
```

Alias for **DEPOSIT**. If you omit the parameter the expression will be evaluated and the answer put in **R0**, i.e. Register 0. See section 4.27 on page 56 (DEPOSIT) for more details.

4.10 CALL_COMIS

```
Syntax:   CALL_COMIS filename[(arg)] Y|N
where:    filename  is the name of the file with the COMIS subroutine
          arg       is the (optional) argument with which the subroutine
                    will be called
          Y|N       decides if the file will be edited before compilation
Defaults: arg = 0.0
```

Calls a COMIS subroutine with argument **arg**. You can use this to do anything you want. However it is probably most useful to do things with histograms you have read in with Mn_Fit and cannot do easily directly in Mn_Fit. You will be asked whether you want to edit the file or not before compiling it.

4.11 CAPTURE

```
Syntax: CAPTURE device_name [Grinell_number]
    or          device_name [Logical_name]
```

Captures a new output device for the plots. See section 1.15 on page 28 (SCREEN_DEVICES) and section 4.42 on page 75 (HARDCOPY) for details of the devices available. If you want a device other than those listed you must give its number (see the relevant manual for GKS or PLTSUB) and it will be captured as device type **unknown**.

4.12 CDIRECTORY

```
Syntax: CDIRECTORY dirname
where   dirname  is the directory name
```

This command is an alias for **SET DIR**. Sets the current HBOOK directory name to **dirname**. Note that the HCDIR command is only executed when a **FETCH**, **LDIR**, **ZDIR** or **SHOW DIR** command is executed. This enables you to **SET** the directory you want to fetch from, and the secondary identifier you want to give the plots, before fetching them. However, this means that successive **CDIR** commands without a **FETCH** etc. inbetween will only remember the last name.

Note that the directory name should not be preceded by a **/**. However **//** is allowed so you can go to the top level.

Note that the Mn_Fit HBOOK file top level directory name is **//MN_HBIN**.

4.13 CLEAR

```
Syntax: CLEAR
```

Clears the currently selected screen device.

4.14 CLOSE

```
Syntax: CLOSE
```

Closes all the hardcopy output devices that have been selected. This enables you to print out plots (use the **SPAWN** or **SHELL** command) without exiting your Mn_Fit session. Next time you give a **HARDCOPY** command a new file will be opened.

4.15 COMIS

```
Syntax: COMIS
```

Invokes COMIS, the CERN COMpilation and Interpretation System, enabling you to write FORTRAN functions and use them without relinking Mn_Fit. For more details on how COMIS is interfaced to Mn_Fit see section 1.13 on page 25 (Using_COMIS). See the COMIS manual for more details on COMIS. Normally you should use one of the Mn_Fit commands that automatically invokes COMIS: **FUN ADD COMIS**, **NTUPLE SCAN**, **PROJECT** or **PLOT**, **CALL_COMIS** etc. rather than this command. This direct interface is provided in case you want to do something special.

4.16 COMMENT

```
Syntax: COMMENT
    or  COMMENT id [&idb]
    or  COMMENT command [ncomm]
    or  COMMENT id [&idb] command [ncomm]
        text
        x, y, size, angle, option, mode, font, colour, thickness
 where:  id        is the histogram number the comment applies to
         idb       is the optional secondary identifier
         command   can be ?|NEW|CHANGE|DELETE|LIST|END
         ncomm     is the comment number to change or delete
         text      is the text you want to display
         x,y       are the position of the comment
         size      is the size of the text in cm (default = 0.4cm)
         angle     is in degrees with respect to the horizontal
         option    can be: LEFT   Comment is left adjusted
                           CENTRE Comment is centred (default)
                           RIGHT  Comment  is right adjusted.
         mode      can be: CM   = Position is in cm (default)
                           PLOT = Position in terms of plot co-ordinates
         font      is the font to use
         colour    is the colour number for the comment
         thickness is the text thickness factor
```

When the **COMMENT** command is given in a file or a defined command the last syntax must always be used and the comment number must be given if it is applicable to the command

(CHANGE or DELETE). You always need the END command when reading from a file to exit COMMENT unless you give the **command** on the same line as COMMENT.

Comments are associated with histograms you have already plotted. Therefore, if you have only plotted one histogram on the picture (using the PLOT, LEGO, SURFACE, DISPLAY or OVERLAY/DIFF commands - OVERLAYS on the same scale do not count), the syntax is the first line. If you already have a comment you will be asked if you want a new one or to move or change the old one. If you have plotted more than one histogram on the display you will be prompted for the identifier the comment applies to.

If you just give the command COMMENT or COMMENT **id[&idb]** you will remain in COMMENT until you give the END command or <CR>. However, if you give some or all of the rest of the COMMENT command on the same line you will automatically exit COMMENT after you have finished the command.

By default the text will be written using the HIGZ routine IGTEXT. For details on the other fonts available see section 1.12 on page 24 (FONTS). The default colour and thickness of the comments are the same as the title.

You can omit all of the parameters except **x** and **y** for a new comment. For **mode** CM, x=0, y=0 is the bottom left-hand corner of the picture If you change or delete a comment use the command REDRAW to see the effect of what you have done.

4.17 COPY

```
Syntax: COPY id1 [&idb1] id2 [&idb2]
 where: id1 and id2 are the input and output histogram identifiers
        idb1 and idb2 are the secondary identifiers
```

Makes a copy of a histogram. You can copy all histograms with a particular secondary identifier using the syntax COPY 0 **&idb1** 0 **&idb2**.

4.18 CUT

```
Syntax: CUT
     or CUT command
where:      command is one of ?|NEW|CHANGE|FILE|EDIT|COMPILE|CHANGE|
            DELETE|LIST|USE|END
```

Specifies what cuts to use when making a projection of a plot, or modifies, deletes or lists existing cuts. To define cuts you can either use a FORTRAN syntax (.EQ., .NE., .LT., .LE., .GT., .GE.) or symbols (=, <>, <, <=, >, >=). Cuts can either be expressions or a COMIS function. If the nearest integer to the return value of the Comis function is 1, the cut is passed.

If the command is included on the same line as CUT then you will exit CUT after the command has been executed.

If the histogram is an Ntuple or has been made with the M_BOOK package the variable names are as you gave them at booking time, otherwise the default names are x,y and **z** for the first 3 dimensions.

The sequence to use is **CUT NEW** to define the cuts you want to use and then use the **CUT USE** command to say how they should be strung together using the usual FORTRAN syntax with **.AND.** and **.OR.**. You can also use the symbols (& and |). For a COMIS function you can also give the value of the argument that it is called with. When you have finished cut give the command **END** or hit <**CR**>. Then give the **PROJECT** command to implement the cuts you have selected (see section 4.66.4 on page 87 (NTUPLE PROJECT) for more details).

For a short description of the commands see section 2.2 on page 33 (CUT Menu). See the individual topics for more information.

4.18.1 NEW

```
Syntax: CUT NEW expression1 condition expression2
where:  expression1  is a variable name or an expression
        condition    can be one of .LT., .LE., .GT., .GE., .EQ., .NE.
                     or <, <=, >, >=, =, <>
        expression1  is a variable name or an expression
```

Specifies a new cut. In its simplest form **expression1** is the name of a variable and **expression2** is a number. A number can be given in the form of a number, register, parameter etc. For more details on expressions see section 1.9 on page 20 (Expressions).

The text of the cut will be stored and then parsed when you ask for a projection. Thus if you use registers etc. in the cut their values at the time that the projection is made will be used.

4.18.2 CHANGE

```
Syntax: CUT CHANGE [ncut] expression1 condition expression2
where   ncut   is the cut number to change
        expression1  is a variable name or an expression
        condition    can be one of .LT., .LE., .GT., .GE., .EQ., .NE.
                     or <, <=, >, >=, =, <>
        expression1  is a variable name or an expression
```

Change a cut (use the = sign if you don't want to change something). Note that you must use == for the condition as = would change the condition to **.EQ.**:

```
cut new x>0.5
cut
CHANGE 1 = == 1.0
end
```

would change the above cut **X .GT. 0.5** to **X .GT. 1.0**. Registers, parameters etc. can be used for the value of the cut. Remember that they only get evaluated when the cut is used.

4.18.3 FILE

```
Syntax: CUT FILE filename [id[&idb]] Y|N
where   filename  is the name of a file containing a COMIS function
        id        is the plot number associated with the cut.
        Y|N       decides if the file is to be edited before compilation
```

Adds a new cut that is a COMIS function. The function will be called for each event in the Ntuple or each bin in the n-dimensional histogram. If the nearest integer to the return value is 1 the event will pass the cut.

If the filename does not exist a skeleton file will be written. If you give a plot number then the variable names from that plot will be used to make the skeleton file. Therefore I recommend always giving a plot number when you make a new file.

Information on the Ntuple is available in 3 common blocks: `PAWIDN, MNTPL1, MNTPL2`. For the contents of these common blocks and when and how they are filled see section 4.66 on page 86 (NTUPLE).

4.18.4 EDIT

```
Syntax: CUT EDIT filename
     or CUT EDIT ncut
where   filename  is the name of a file containing a COMIS function
        ncut      is a cut number which is a COMIS function
```

Edits the file with the default editor (set with the SET EDIT command) and then recompiles the cut.

4.18.5 COMPILE

```
Syntax: CUT COMPILE filename
     or CUT COMPILE ncut
where   filename  is the name of a file containing a COMIS function
        ncut      is a cut number which is a COMIS function
```

Recompiles the cut. You can use this command if you edit the cut from outside Mn_Fit or just with the `EDIT` command instead of `CUT EDIT`. Note that the cut is always recompiled just before it is used in `NTUPLE PROJECT` or `PLOT`.

4.18.6 DELETE

```
Syntax: CUT DELETE [ncut]
where   ncut  is the cut number to change
```

Deletes a cut (`DELETE 0` means delete all cuts).

4.18.7 LIST

```
Syntax: CUT LIST
```

Lists the cuts already specified.

4.18.8 USE

```
Syntax: CUT USE expression
where   expression  is the list of cuts to use and how to combine them
```

Gives the list and order of cuts to use e.g. `1 .and. (2 .or. 3)` including whatever parentheses are necessary. You can use `.AND.` or `.OR.` or the symbols `&` and |. If you do not include the complete cut in parentheses they will be added for you. If a cut is a COMIS function you can give the argument with which the function should be called. e.g. `1 .and. 2(0.5)` would call the COMIS function corresponding to cut 2 with the argument value 0.5.

The are 3 special values of the expression allowed:

```
 0 means no cuts are to be used
-1 means all cuts should be ANDed
-2 means all cuts should be ORed
```

4.18.9 END

```
Syntax: CUT END
```

Exits cut.

4.18.10 Examples

1. Define the cuts:

```
CUT NEW
X .GT. 0.5
CUT
NEW X< 2.0
NEW Y.LT.7.0
NEW Y.GT.10.0
file cut.for
```

Specify which cuts to use. The COMIS function will be called with argument 1:

```
CUT USE 1 .AND. 2 .AND. (3 | 4) & 5(1)
END
```

Make a projection:

```
PROJECT id X &idb
```

Change 1 of the cuts:

```
CUT CHANGE 3
= .LE. 4
END
```

Make a second projection:

```
PROJECT id X &idb2
```

4.19 NO_CUT

```
Syntax: NO_CUT id [&idb]
 where: id       is the plot identifier
        idb      is the (optional) secondary identifier
```

Deletes all the specified cuts on a particular histogram.

4.20 DATABASE

```
Syntax: DATABASE DB_HISTORY|DB_SNAP
```

Interface to plotting from the L3 database. This facility should be able to be easily extended to anyone who uses an RZ based database. The following detector databases are known: FLUM, ECAL, HCAL, TECH, MUCH, L3RC.

4.20.1 DB_HISTORY

```
Syntax: DB_HISTORY directory chan1 chan2 start stop [mode]
where:  directory  is the full pathname (e.g. //DBFL/BGO/LED/AVG)
        chan1      is the first channel number
        chan2      is the last channel number
        start      is the starting date and time (default = 890801.000000)
        stop       is the finishing date and time (991231.000000)
        mode       specifies the mode to plot (default is S)
                   N  plots the numbers as is
                   S  scales them by a reference entry (the first of the year)
                   C  scales them by the latest entry
```

Plots one or more elements of a database vs. time. You can use the `SET TIME` command to specify which mode to store the data (default is days). In the future you will be able to set the plotting mode (`DATE` or `TIME`). For now if no x label has been given the reference time will be shown (i.e. T=0 on the plot).

If you give more than 1 channel they will be averaged. All times should be given in the form `YYMMDD.HHMMSS`. The default starting time is 890801.000000 and the default finishing time is 991231.000000. If the time is omitted it is 000000 by default.

You can plot the data as is or scaled by either a reference entry (the first entry of the year) or the latest entry.

The plot is stored in identifier 98766 with the current secondary identifier (specified with the `SET IDB` command).

4.20.2 DB_SNAP

```
Syntax: DB_SNAP directory chan1 chan2 time [mode]
where:  directory  is the full pathname (e.g. //DBFL/BGO/LED/AVG)
        chan1      is the first channel number
        chan2      is the last channel number
```

```
time      is the time for which you want the data (default = latest)
mode      specifies the mode to plot (default is S)
          N  plots the numbers as is
          S  scales them by a reference entry (the first of the year)
          P  scales them by the previous entry
```

Plots part or all of the contents of a database directory.

Time should be given in the form (YYMMDD.HHMMSS). The default time gets you the last entry. You can plot the data as is or scaled by either a reference entry (the first entry of the year) or the previous entry.

The plot is stored in identifier 98767 with the current secondary identifier (specified with the SET IDB command).

4.21 DAT_FETCH

```
Syntax: DAT_FETCH filename
```

Reads a file containing a series of data points or an Ntuple which have been written in ASCII format. The file can also contain data vs. time.

The first non-comment line of the file must contain the identifier for the plot. If the identifier is omitted it will be set to 1 with the current secondary identifier.

The following cards are allowed (only the ID card is required):

```
!              denotes a comment card (must be in column 1 or 2 for the
               datacards)
ID    id [idb] to specify the identifier for the plot
NTUPLE ndim name1 name2 ...
               gives the number of dimensions and the names of the variables
               for an Ntuple
LIMIT xlo xhi  to specify the lower and upper limits for the plot
TITLE title    to specify the title for the plot
ORDER          to specify the order of the variables
TIME           to specify the time mode to store and the reference time
DATA           to flag that the cards following this contain the data
END            to flag that this is the end of the data
```

The default order for the data is: x y dx dy or if you want asymmetric error bars: x y dnx dny dpx dpy. You can change the order either by having an ORDER card or using the SET ORDER command (see section 4.79.40 on page 102 (SET ORDER) for more details). The following ways of giving the time are recognized:

DATE_TIME	YYMMDD HHMMSS
DATE	YYMMDD
TIME	HHMMSS
DATE_MIN	YYMMDD HHMM
TIME_MIN	HHMM
VAXTIME	Char*23 Vaxtime DD-MMM-YYYY HH:MM:SS.SS

VAXTIME can be abbreviated on the Vax, but must be the full CHAR*23 format on any other computer. These commands can be given on the ORDER card or with the SET ORDER command. The data are stored in the form given on the TIME card or with the SET TIME command and can be (DAY, HOUR, MINUTE or SECOND, default is DAY). A reference time (T=0) can be given and this must be in the form YYMMDD HHMMSS. If this is omitted or not specified the first point will be used.

A blank card, one with a number as the first element, or the DATA card will be taken to signal the start of data.

You can also use DAT_FETCH to read in an Ntuple which you have written to a file. Use the NTUPLE card to give the dimension and the names of the variables for the Ntuple (there must be as many names as dimensions). The LIMITS and ORDER cards will be ignored if given. Again the data must be given as 1 record per line and should be in the same order as the names given in the NTUPLE card.

Either an END card or an ID card signify the end of one dataset and the start of the next.

4.21.1 Examples

1. A simple plot:

```
ID 1
LIMIT  0.0 1.0
TITLE This is a test file
0.4 5.7 0.01 3
0.2 1   0.02 0.5
0.6,0.1 0.1  0.4
```

2. A file containing dummy entries and an Ntuple:

```
ID 2
TITLE This file is more complicated
ORDER X DX DUMMY Y DY
DATA
3.4 1.6,35.0,5 2
2.2 1   45.0,6 1
END
ID 3
TITLE A second dataset in the same file
0.4 0.1  35.0 3   1.0
0.2 0.05 45.0 0.5 0.2
0.6,0.1  55.0 0.4 0.1
ID 4
NTUPLE 4 P MASS THETA PHI
TITLE Ntuple containing particle parameters
0.35 0.498 0.6 3.6
1.20 0.480 -0.5, 1.6
END
```

4.22 DAT_STORE

```
Syntax: DAT_STORE filename id [&idb] [id2 [&idb2]...]
```

Alias for WRITE DATA. Writes one or more plots (only 1-dimensional or a true scatter plot) to a file, in card image format. You can read the plot in again using the READ DATA or DAT_FETCH commands. This format is useful if you want to change something in the histogram like deleting a point or if you at some time later want to add more points. You can edit the file with your normal editor. This is also a good trick for converting a histogram to a series of points.

You can give a list of plot numbers to store, but they must all be on 1 line and neither a range nor 0 is allowed.

4.23 DB_HISTORY

```
Syntax: DB_HISTORY directory chan1 chan2 start stop [mode]
where:  directory  is the full pathname (e.g. //DBFL/BGO/LED/AVG)
        chan1      is the first channel number
        chan2      is the last channel number
        start      is the starting date and time (default = 890801)
        stop       is the finishing date and time (991231)
        mode       specifies the mode to plot (default is S)
                   N  plots the numbers as is
                   S  scales them by a reference entry (the first of
                      the year)
                   C  scales them by the latest entry
```

Alias for DATABASE DB_HISTORY. See section 4.20.1 on page 51 (DATABASE DB_HISTORY) for more details.

4.23.1 DB_SNAP

```
Syntax: DB_SNAP directory chan1 chan2 time [mode]
where:  directory  is the full pathname (e.g. //DBFL/BGO/LED/AVG)
        chan1      is the first channel number
        chan2      is the last channel number
        time       is the time for which you want the data (default = latest)
        mode       specifies the mode to plot (default is S)
                   N  plots the numbers as is
                   S  scales them by a reference entry (the first of the year)
                   P  scales them by the previous entry
```

Alias for DATABASE DB_SNAP. See section 4.20.2 on page 51 (DATABASE DB_SNAP) for more details.

4.24 DEFINE

```
Syntax: DEFINE name
```

```
where: name  is the new command name
```

DEFINE creates interactively whole sets of commands, where **name** is any name you like except ALL. However, if there already is a command of that **name** the new definition will be ignored. Also, first standard Mn_Fit commands are searched for and then ones made with DEFINE. Therefore, if your new command is ambiguous with a Mn_Fit command, the Mn_Fit command will always be used. After giving the command line you will see the prompt DEFINE> and you give the defining commands one at a time. If you give the command ENDDEF, you will revert to the usual prompt and are ready to use the new command. DEFINE works both at the MN_CMD> and the MINUIT> level.

If you want to DEFINE a command with parameters from inside a file then use the syntax **@@1, @@2** etc. This will be translated to **@1, @2,** etc and then you can use the parameters within the DEFINE command. If you DEFINE a command interactively, use the syntax **@1, @2,** etc. For more on the use of parameters see section 4.36 on page 61 (EXECUTE).

4.24.1 Examples

1. Define the command LOOK:

```
MN_CMD> DEFINE LOOK
DEFINE> PRINTOUT -5
DEFINE> SIMPLEX @1 50 @2
DEFINE> MIGRAD @1 @
DEFINE> INFO
DEFINE> ENDDEF
```

If you then give:

```
MINUIT> LOOK 5 3
```

then you will execute:

```
PRINTOUT -5
SIMPLEX 5 50 3
MIGRAD 5 3
INFO
```

If you give:

```
MINUIT> LOOK
```

then you will execute:

```
PRINTOUT -5
You will be prompted for the values of parameters @1 and @2.
If you hit <CR> then you will execute:
SIMPLEX 50
MIGRAD
INFO
```

2. You can nest commands:

```
MN_CMD> DEFINE FULL
DEFINE> FIT @1
DEFINE> 0
DEFINE> LOOK @2 @3
DEFINE> PRINTOUT -5
DEFINE> MINOS 1000 @3
DEFINE> PRINTOUT 1
DEFINE> ENDDEF
```

FULL will invoke the just defined LOOK command. All resulting actual commands will be echoed on the screen, just so that you know what is going on. Also if you have not given a parameter when invoking the command you will be prompted for its value the first time that the parameter is found.

4.25 UNDEFINE

```
Syntax: UNDEFINE name|ALL
```

Deletes a command that you have created using the DEFINE command. If you give the command UNDEFINE ALL all defined commands will be deleted. You, therefore, are not allowed to define a command ALL.

4.26 DELETE

```
Syntax: DELETE id[:id2] [&idb [:[idb2]]
```

Deletes one or more histograms. To delete all the histograms, issue the command DELETE 0. To delete all those with a particular secondary identifier, issue the command DELETE 0 &IDB. To delete a range of histograms, use the command DELETE 1:300 & 0 : 50. This will delete all histograms with primary identifiers between 1 and 300 and secondary identifiers between 0 and 50. DELETE 0&1:2 will delete all histograms with secondary identifiers between 1 and 2.

4.27 DEPOSIT

```
Syntax: DEPOSIT parameter = expression
 where: parameter    is the parameter you want to change
        expression   is the expression for its new value
```

The expression can be any sort of mathematical calculation using numbers, registers, parameters, errors, bin contents, errors on bin contents, centre of bins etc. See section 1.9 on page 20 (Expressions) for more details. Details on the meaning of all the parameters can be found in HELP NUMBERS. Use the **EXAMINE** command to look at any of the parameters described below.

The parameter can be any of the possibilities given below or a user variable. User variables are alphanumeric strings (including _ and **$**) up to 8 characters long. They cannot have the same name as any of the parameters listed below.

There are 100 registers available for your use numbered 0 to 99. Registers >=100 are used by Mn_Fit and contain numbers you may wish to access. See section 1.6 on page 17 (Numbers) for what is in them. Registers >300 contain user variables in the order that they are defined. You can use the **SHOW REGISTER** command to list a range of registers.

To change the contents of a register:

```
DEPOSIT Rn      = expression
        where  n is the register number
```

To change the value of a parameter or its error in a function:

```
DEPOSIT Pn(m)      = expression
        ERRn(m)    = expression
        ERNn(m)    = expression
        ERPn(m)    = expression
        LOLIMn(m) = expression
        HILIMn(m) = expression
        where  n is the function number (as in FUNCTION INFO)
               m is the parameter number (as in FUNCTION INFO)
```

This command also changes the values of MINUIT parameters so can be used instead of **MODIFY**. However if you want to change a parameter and its error for example **MODIFY** is probably simpler to use. You can use the MINUIT parameter numbers when you are fitting using the form **Pn** where **n** is the MINUIT parameter number.

To change the contents or errors of a bin in a plot:

```
DEPOSIT Yn(m)    = expression
        Xn(m)    = expression
        DYn(m)  = expression
        DXn(m)  = expression
        DNXn(m) = expression
        DNYn(m) = expression
        DPXn(m) = expression
        DPYn(m) = expression
        where  n is the plot number (can include the secondary identifier)
               m is the bin number
```

Note that the x values and its error can only be changed for a series of data points.

You can also change the contents of a 2-d histogram using the form:

```
DEPOSIT Yn(l,m) = expression
        where n is the plot number (can include the secondary identifier)
              l,m are the bin numbers
```

You can change a variable in an Ntuple, provided that the Ntuple is stored in memory (<50000 words long):

```
DEPOSIT Xn(m,nvar) = expression
or      Xn(m,tvar) = expression
        where n    is the plot number (can include the secondary identifier)
              m    is the event number
              nvar is the variable number
              tvar is the variable name
```

4.27.1 Examples

1. Setting the initial values of parameters before fitting or plotting a function:

```
DEPOSIT P1(2) = 1000
DEPOSIT ERR1(2) = 500
FIT ...
```

2. Using a register to calculate an upper limit and putting that in a plot Store the upper limit in one plot and the result with asymmetric errors in another plot:

```
DEPOSIT R10 = R10 + 0.4  !Set x value for which the limit applies
DEP     R11 = P1(2) + 1.64*ERP1(2)
FILL  10 R10 R11
FILL  11 R10 P1(2) 0.2 ERN1(2) 0.2 ERP1(2)
```

3. Evaluate some expression:

```
dep pi = 3.14159
DEP R1 = R1 + 0.3*(R2^2 + SQRT(P1(4) - P2(4))) - sin(pi/4)
```

4.28 DISPLAY

```
Syntax: DISPLAY detnam id [&idb]
 where: detnam  is the detector name
        id      is the histogram identifier
        idb     is the (optional) secondary identifier
```

Makes a special display. This has been implemented for the following subdetectors in the L3 experiment: ECAL, FBGO and FWCH.

Alias for HIST DISPLAY. See section 4.51.2 on page 78 (HISTOGRAM DISPLAY) for more details.

4.29 DIVIDE

```
 Syntax: DIVIDE id1 [:id1n] [&idb1] id2 [:id2n] [&idb2] id3 [:id3n] [&idb3]
         [scale1] [scale2]                           (default scale = 1.0 1.0)
  where: id1,id2   are the input histogram identifiers
         id3       is the output histogram identifier
         idb1,idb2 are the (optional) input secondary identifiers
    and  idb3      is the (optional) output secondary identifier
```

Divides two histograms (id1, id2) to make a third one. To specify the secondary identifier, precede it by a &, otherwise the default will be used. (Use the SET IDB command to change the default). The scale factors are optional. To avoid confusion, you should give a <CR> after the identifiers or make sure the scale factors are given as real numbers.

To divide a range of histograms, the primary identifiers you give for the input and output histograms must be the same, but you can specify different secondary identifiers. For example, `DIVIDE 300:400&1 300:400&2 300 :  400 & 10` will divide all histograms with primary identifiers 300 to 400 and secondary identifiers 1, by those with secondary identifiers 2, putting the results into histograms with the same primary identifiers and secondary identifier 10. If you give primary identifier 0, the operation will be performed on all plots with the given secondary identifier.

4.30 DO

```
Syntax: DO par = low high [step]
        where  par   is the loop variable (must be a single letter)
               low   is the starting value
               high  is the finishing value
               step  is the (optional) step size (default = 1)
```

Starts a DO loop. The loop is terminated by an ENDDO. DO loops only work inside macros or DEFINEd commands. The limits can be given explicitly or they can be registers, parameters, bin sizes, contents etc. (see section 1.6 on page 17 (NUMBERS) for more details).

Please note that I have made no attempt to optimize the speed of execution of a DO loop. Every time through the loop the file is rewound. However, they are very useful for doing repetitive procedures.

The value of the loop parameter can be used with the syntax @par and a direct character substitution is performed. This is useful for constructing histogram identifiers for example.

4.30.1 Examples

1. The following example adds histogram 1&103 to 2&103 putting the result in histogram 3&3. A new histogram 4&3 is booked. Then a loop over the 100 bins of histogram 3&3 is made and the square of the bin contents is put in histogram 4&3.

   This procedure is repeated for histograms 1&104, 2&104, 3&4 4&4 etc. up to 1&108 etc:

   ```
   DEP R1 = 3
   ```

```
DEP R2 = 8
DO I= R1,R2
   ADD 1&10@I 2&10@I 3&@I 1 1
   book/bin/err 4&@I New plot
   1 100 0 100
   dep R5  = 0.0
   DO J=1,100
        dep R5 = R5 + 1
        DEP R6 = Y3&@I(@J) ** 2
        FILL 4&@I R5 R6
   ENDDO
ENDDO
```

4.31 ENDDO

Syntax: `ENDDO`

Terminates a `DO` loop. See section 4.30 on page 59 (DO) for more details.

4.32 DRAW

```
Syntax: DRAW LINE|ARROW|BOX|TRIANGLE|POLYGON|HELP|CHANGE|DELETE|LIST|END
        [number]
    or  symbol colour thickness mode hatch pattern
    or  npoint symbol colour thickness mode hatch pattern
        x1,y1
        x2,y2...
 where: number      is the item number to change
        npoint      is the number of points in the polygon
        symbol      is the symbol number
        colour      is the colour of the line
        thickness   is the line thickness (scale factor)
        mode        is the mode (CM or PLOT units)
        hatch       is the hatch symbol to fill the shape
        pattern     is the pattern symbol to fill the shape
```

Draws an item on the plot. All of the first set of parameters can take default values if you hit <CR>. For `mode CM`, x=0, y=0 is the bottom left-hand corner of the picture. If you want to change an item you have drawn, then give the command `DRAW CHANGE` and you will be prompted for the item you want to change and what you want to change.

If you just give the command `DRAW` you will remain in `DRAW` until you give the `END` command or <CR>. However, if you give some or all of the rest of the `DRAW` command on the same line you will automatically exit `DRAW` after you have defined one.

If you are fitting more than one histogram the item(s) will only be drawn on the last plot. If you select plot units (`mode PLOT`) they will be for the last plot that you have drawn. However, if you have more than one window the correct units will be used.

4.33 EDIT

```
Syntax: EDIT filename
```

Edits a file. The default EDIT command is EDIT/TPU on the Vax. On the Apollo it is the Domain editor. To change it to your favourite editor use the SET EDIT command (only possible on the Vax).

4.34 EFFICIENCY

```
Syntax: EFFICIENCY id1[:id1n] [&idb1] id2[:id2n] [&idb2] id3[:id3n] [&idb3]
 where: id1,id2   are the input histogram identifiers
        id3       is the output histogram identifier
        idb1,idb2 are the (optional) input secondary identifiers
   and  idb3      is the (optional) output secondary identifier
```

Divides two histograms (id1, id2) to make a third one using binomial errors. To specify the secondary identifier, precede it by a & otherwise the default will be used. (Use the SET IDB command to change the default).

To calculate the efficiency for a range of histograms, the primary identifiers you give for the input and output histograms must be the same, but you can specify different secondary identifiers. For example, EFF 300:400&1 300:400&2 300 : 400 & 10 will divide all histograms with primary identifiers 300 to 400 and secondary identifiers 1, by those with secondary identifiers 2, putting the results into histograms with the same primary identifiers and secondary identifier 10. If you give primary identifier 0, the operation will be performed on all plots with the given secondary identifier.

4.35 EXAMINE

```
Syntax: EXAMINE parameter
 where: parameter  is a register, function parameter, bin etc.
```

Enables you to find the contents of a register (similar to the SHOW REGISTER command), the value of a parameter or its error, the contents of a bin, their error, the bin centre or bin width, or a user variable. See section 1.6 on page 17 (Numbers) and section 4.27 on page 56 (DEPOSIT) for details on the meaning of all the parameters.

4.36 EXECUTE

```
Syntax: EXECUTE filename  [parameter_list]
 where: filename        is the filename with the commands
        parameter_list  are parameters you want to pass
```

This command is an alias for READ COMMAND. It executes a series of commands which you have written to a file. You can input parameters to the file and they are referenced with the syntax @1, @2, etc. You are allowed up to 9 parameters and the parameters can be either numbers or character strings. Parameters are separated by spaces. If you omit a parameter in the EXECUTE command, you will be prompted for its value the first time it is referenced. Blanks are used to delimit parameters. Therefore, if a parameter contains an imbedded blank it should be included in single or double quotes. e.g. 'This name' or "Who is this". A null parameter can be passed as ''. You can also use the INQUIRE command to ask for the value of a parameter. See section 4.56 on page 82 (INQUIRE) for more details.

Comment lines must begin with a !. The default filename qualifier is .mnf. You can exit a command file at any point by putting the RETURN command in it.

Within macros you can have DO loops (see section 4.30 on page 59 (DO) for more details).

If you give a PLOT command in the file and it is not the first picture this Mn_Fit session you have to give a <CR> for the next plot. If you give any other character the plot command will be skipped and if you give a q the file will be aborted.

4.36.1 Examples

1. You write 2 files TEST.MNF and TEST2.MNF with the following commands:

```
!
! This file fetches, renames and plots a histogram
!
inquire 1 'Give filename'
inquire 2 'Give histogram number(s)'
FET @1 @2
REN @2 1000&@3
PLOT 1000&@3
INDEX
EXEC TEST2 1000 @3 2000
```

The file TEST2.MNF contains the following commands:

```
COP @1&@2 @3
PLOT @3
! This is a way to look at a plot before deciding whether you want
! a hardcopy for example
inquire 4 'Give extra command (e.g. hardcopy)'
@4
HIST DUMP @3
inquire -4 'Give extra command or <CR>'
@4
RETURN
! The following command will not be executed
HELP
```

To execute the file you give the command:

```
EXEC TEST.MNF 3S_EXCL.HIS 300 10
```

and the following commands will be executed:

```
FET 3S_EXCL.HIS 300
REN 300 1000&10
PLOT 1000&10
INDEX
EXEC TEST2 1000 10 2000
```

and TEST2.MNF will execute the following commands:

```
COP 1000&10 2000
PLOT 2000
! This is a way to look at a plot before deciding whether you want
! a hardcopy for example
Give extra command: HARD P
HIST DUMP 2000
Give extra command or <CR>: <CR>
RETURN
```

4.37 EXTRACT

Syntax: HIST EXTRACT id part [npart] [&idb]

Alias for HIST EXTRACT. See section 4.51.5 on page 79 (HISTOGRAM EXTRACT).

4.38 FETCH

Syntax: FETCH filename id1 [:id2] [id3...]

Fetches HBOOK version 4 histogram(s) from a file. To fetch all histograms give the command FETCH **filename** 0. To specify a range of histograms give the command FETCH **filename id1:id2**. If you want to fetch another histogram(s) from the same file give the command FETCH **id1,id2....** If the histogram number you ask to fetch already exists it will be overwritten. All the histograms will be given the default secondary identifier.

The record length for direct access HBOOK version 4 files must be known. The default is 1024 (Vax) and 4096 (Apollo). If your files have been made with a different record length, use the SET RECL command to set the length. This length will also be used for STORE commands.

Ntuples and variable bin width plots can also be fetched using the above syntax. Ntuples of <50000 words will be read directly into memory. Longer Ntuples which have been made with the CHDIR option not blank, will not be read in. This means that the directory name must be correctly set before you try to manipulate such an Ntuple (e.g. with the PROJECT, or HIST DUMP commands) and the last fetch command you gave must have been for the file containing the Ntuple.

If your histograms are in several subdirectories, you should use SET DIRECTORY or CDIR in connection with the SET IDB command to give them different secondary identifiers. See the examples. If you do not know the directory structure of the file use the HB_OPEN command and the LDIR, ZDIR and CDIR commands to list what is in the directory, before giving the FETCH command.

Note that the top level directory name for the file is //MN_HBIN.

WARNING: If you fetch all histograms from a file, or specify a range, any existing HBOOK histograms in the current directory in memory will be deleted! However, Mn_Fit will still store the histograms internally. This means that you will lose any slices, bands, projections or functions associated with the HBOOK histogram unless you have already extracted them into Mn_Fit plots. You will also lose the contents of an Ntuple if it is not stored in memory.

4.38.1 Examples

1. Get all plots from a file:

```
FETCH filename 0
```

2. Get some plots from the same file as the last fetch:

```
FETCH 1:10,20,100:200
```

3. Get all the plots from various subdirectories:

```
SET DIRECTORY dir1 IDB 100 ENDSET  FETCH filename 0
SET DIRECTORY dir2 IDB 200 ENDSET  FETCH filename 1:100
SET DIRECTORY \/dir3 IDB 300 ENDSET  FETCH 0
```

or:

```
HB_OPEN filename
SET DIRECTORY dir1 IDB 100 ENDSET  FETCH 0
SET DIRECTORY dir2 IDB 200 ENDSET  FETCH 0
SET DIRECTORY dir3 IDB 300 ENDSET  FETCH 0
```

Note that if you give a filename then the directory name you give is relative to the top directory. If you do not give a filename then the name is relative to the current directory name. The second method avoids opening and closing the file for every fetch.

4.39 FILL

```
Syntax: FILL id [&idb] x y [dx [dy] ...]
    or  FILL id [&idb] x [y]
 where: id    is the plot identifier
        idb   is the (optional) secondary identifier
        x     is the x value of the point
        y     is the y value of the point
        dx    is the error on x
        dy    is the error on y etc.
```

Fills a plot. Alias for HIST FILL. See section 4.51.6 on page 79 (HISTOGRAM FILL) for more details.

4.40 FIT

```
Syntax: FIT id1 [&idb1] [-nfun1 [-nfun2 ...]] id2 [&idb2] [-nfun3 ...] ...
        nmode
 where: id1,id2   are the histograms to fit
        idb1,idb2 are the (optional) secondary identifiers
        nfun      are the numbers of functions not to be used in the
                  fit to the preceding id
        nmode     = 0 means do a chi**2 fit
                  = 1 means do a likelihood fit
```

Gets you into MINUIT, so that you can actually start fitting. You must have already fetched the histogram(s) you want to fit and added the function(s) you want to use. You will be asked whether you want to do a χ^2 or likelihood fit. Default is χ^2. Note that the χ^2 or the likelihood is stored in register 111 (access it using the syntax R111), and the confidence level is stored in register 112, so that you can use it to put in a plot for example.

See the section in the manual "Fitting With Mn_Fit" for more complete details on how to fit. See section 1.3.1 on page 14 (ERRORS) for more information on MINUIT errors and how to interpret them.

It is possible to fit 1 or 2 dimensional histograms or a series of data points. However, it is not possible to fit to a true scatter plot. Use the PROJECT command to make a binned 2-D histogram out of the scatter plot if you want to fit it.

You can simultaneously fit 2 or more plots by giving both their identifiers with the fit command. If you want to use different background functions for each plot, but the same signal, e.g. when you fit a mass peak from 2 different decay modes, you should give the function numbers NOT to be used for a plot after you give its identifier. However, if you are using the option SET RATIO ON which means that the areas are parametrized as ratios this only works properly for 2 plots as Mn_Fit could not constrain the sum of a set of parameters to be 1 in MINUIT.

If you give the FIT command inside a macro I recommend that you leave a blank line after the 0 or 1 for χ^2 or likelihood fit. This is because you are sometimes asked if you want to calculate new orthogonality limits.

Note that when you enter MINUIT, any limits you have set on plots will be reset to 0 0. This is so that you do not get confused, if you do a DISPLAY and only see part of the fit. You can of course set the limits to whatever you like again, if you really do not want to see the whole plot.

When inside MINUIT it is possible to include or exclude regions from the fit and also to constrain parameters relative to other parameters. For more details see section 3 on page 39 (MINUIT).

The following is a list of commands which control how you fit and what parameters are used:

```
SET NORM ON|OFF     Turns on/off an overall normalization factor.
SET RATIO ON|OFF    Controls how the areas are parametrized, when
```

```
                         fitting more than one plot.
SET FIT
  INTEGRATE ON|OFF  Turns on/off integrating the function across each bin.
```

4.40.1 Examples

1. To do a χ^2 fit to plot 300:

```
FIT 300
0
```

2. Do a likelihood fit to plots 400 and 501&10 simultaneously, using the same functions for each plot:

```
FIT 401 501&10
1
```

3. Do a χ^2 fit to plots 1 and 17 simultaneously. Function 1 will be used as the signal in all plots and function 4 will be the background for plot 1, function 3 for plot 17:

```
func use 1 2 3 4
fit 1 -2 -3 17 -2  -4
0
```

4.41 FUNCTION

Syntax: FUNCTION command

Does something with a function. See section 2.3 on page 33 (FUNCTION Menu) for a short description of each command. For a detailed Help on the functions available see section 4.41.9 on page 68 (FUNCTION LIST) and it subtopics. When you are asked to give function number(s), 0 means all the defined or selected functions.

4.41.1 ADD

```
Syntax: FUNCTION ADD nfun
 where: nfun  is either the function name or number
```

Only valid in `MN_CMD>`.

Selects a new function to be used for fitting with or plotting. You will be prompted for the initial values of the parameters.

See section 4.41.9 on page 68 (FUNCTION LIST) for a list of the available functions and the various subtopics for details of the syntax for USER and COMIS functions.

4.41.2 COMPILE

```
Syntax: FUNCTION COMPILE nfun
  where: nfun  is the function number as in FUNCTION INFO
```

Only valid in MN_CMD>.

Recompiles a COMIS function which you have already added with the FUNCTION ADD command.

4.41.3 EDIT

```
Syntax: FUNCTION EDIT nfun
  where: nfun  is the function number as in FUNCTION INFO
```

Only valid in MN_CMD>.

Invokes the editor and then recompiles the COMIS function you have already added with the FUNCTION ADD command. The default edit command is EDIT/TPU. You can change it with the SET EDIT command.

4.41.4 DELETE

```
Syntax: FUNCTION DELETE nfun
  where: nfun  is the function number as in FUNCTION INFO
```

Only valid in MN_CMD>.

Deletes a function from the list of those you have added. nfun = 0 will delete all functions that have been added.

4.41.5 FETCH

```
Syntax: FUNCTION FETCH filename
  where: filename  is the name of the file containing the functions.
```

Only valid in MN_CMD>.

Fetches functions which you have previously stored from a file.

4.41.6 HISTOGRAM

```
Syntax: FUNCTION HISTOGRAM nfun1 [nfun2...]
        [&]idb nsymb
        npoint xlo xhi
  where: nfun   is the function number(s) as in FUNCTION INFO
         idb    is the secondary identifier for the function
    and  nsymb  is the symbol you want to plot the function with
```

Plots one or more of the functions you have added as a histogram. nfun = 0 means use all functions which are currently in use. This command is the same as FUN PLOT, but the function will be stored as a histogram. If you are fitting and your last picture was a DISPLAY, the function will be given the same primary identifier as the one you are fitting. Otherwise it will be given the primary identifier 98765. To specify the secondary identifier on the same line as the function numbers precede it by a &. It will be given the secondary identifier you specify.

4.41.7 HOVERLAY

```
Syntax: FUNCTION HISTOGRAM nfun1 [nfun2...]
        [&]idb nsymb
 where: nfun   is the function number(s) as in FUNCTION INFO
        idb    is the secondary identifier for the function
   and  nsymb  is the symbol you want to plot the function with
```

Overlays one or more of the functions you have added as a histogram. `nfun` = `0` means use all functions which are currently in use. This command is the same as `FUN OVERLAY`, but the function will be stored as a histogram with the same number of points as the plot you are fitting. If you are fitting and your last picture was a `DISPLAY`, the function will be given the same primary identifier as the one you are fitting. Otherwise it will be given the primary identifier 98765. To specify the secondary identifier on the same line as the function numbers precede it by a `&`. It will be given the secondary identifier you specify.

4.41.8 INFO

```
Syntax: FUNCTION INFO
```

Gives information on the functions you have selected, whether they are in use, whether they are signal or background, and the values of their parameters.

4.41.9 LIST

```
Syntax: FUNCTION LIST
```

Lists the functions which are available:

```
 1 Polynomial
 2 Exponential
 3 Quadratic Joining Two Lines
 4 Chebyshev Polynomial
 5 Legendre Polynomial
 6 Gaussian Distribution (sigma)
 7 Gaussian Distribution (FWHM)
 8 Landau Distribution
 9 Breit-Wigner Distribution
10 (x-a)*Gaussian Distribution (sigma)
11 (x-a)*Gaussian Distribution (FWHM)
12 Lepton Spectrum Phase Space
13 Lepton Spectrum V-A for b->c
14 Lepton Spectrum V+A for b->u
15 Lepton Spectrum Spin 0->Spin 0
16 Hadron Spectrum Phase Space
17 Hadron Spectrum V-A or V+A
18 Hadron Spectrum Spin 0->Spin 0
19 Two Gaussians (sigma)
```

```
20 Two Gaussians (FWHM)
21 Bifurcated Gaussian (sigma)
22 Bifurcated Gaussian (FWHM)
23 Sum of Two Bifurcated Gaussians (sigma)
24 Sum of Two Bifurcated Gaussians (FWHM)
25 ARGUS background function - phase space going to 0 at beam energy
26 P*exp(Q)  A polynomial multiplied by the exponential of a polynomial
27 Trigonometric Functions
28 CB Line Shape
29 Dipion Inv Mass
30 Fragmentation Functions
31 Resonance Cross Section
32 Continuum Cross Section
33 User Defined Function (see subtopic USER for details)
34 Histogram used as a function
35 COMIS defined function (see subtopic COMIS for details)
36 2D Gaussian Distribution (sigma)
37 2D Gaussian Distribution (FWHM)
```

Polynomial

Polynomial of any order with an offset. The form used is:

```
NORM + P(1)*(x-OFFSET) + P(2)*(x-OFFSET)**2 + ...
```

Exponential

Falling exponential with an offset. The form used is:

```
NORM * EXP(-SLOPE * (x - OFFSET))
```

Quadratic_Joining_Two_lines

Given ordinate `CONST` and abscissa `JOIN1` of one point on a straight line of slope `SLOPE1`, it finds a quadratic curve that smoothly connects this line at that point to another straight line of slope `SLOPE2` at an abscissa `JOIN2`. In terms of the five parameters, the equation of the quadratic equation is:

```
a + b*x + c*x**2

where  c = (SLOPE2 - SLOPE1)/(2.0*(JOIN2-JOIN1)
       b = (SLOPE1*JOIN2 - SLOPE2*JOIN1)/(JOIN2 - JOIN1)
       a = CONST - b*JOIN1 - c*JOIN2**2
```

The constant term of the second straight line (`a2` + `SLOPE2*x**2`) is:

```
a2 = a +b*JOIN2 + c*JOIN2**2 - SLOPE2*JOIN2
```

Chebyshev

The Chebyshev is valid between the minimum and maximum limits which are either set to the plot limits or can be manually set with the SET ORTHOG command.

The form used is:

```
DELX = (2*X - XMIN - XMAX) / (XMAX - XMIN)
1st term = NORM
2nd term = CHEB01 * DELX
3rd term = CHEB02 * (2*DELX*DELX - 1)
4th term = CHEB03 * (2*DELX*(2*DELX*DELX - 1) - DELX)
etc.
```

and the function value is given by the sum of the terms

Bifurcated_Gaussian

A Gaussian with different slopes either side of the mean. The form used is:

```
AREA * (2 * SIGA / (SIGA + SIGB) / SQRT(2 * PI * SIGA) *
  EXP(-0.5 * (X - MEAN) / SIGA**2)                      for X <  MEAN
AREA / (2 * SIGB / (SIGA + SIGB) / SQRT(2 * PI * SIGB) *
  EXP(-0.5 * (X - MEAN) / SIGB**2)                      for X >= MEAN
```

Sum_Two_Bifurcated_Gaussians

The sum of two Gaussians with different slopes either side of the mean. The mean is the same for both Gaussians. The form used for each Gaussian is:

```
AREA * (2 * SIGA / (SIGA + SIGB) / SQRT(2 * PI * SIGA) *
  EXP(-0.5 * (X - MEAN) / SIGA**2)                      for X <  MEAN
AREA / (2 * SIGB / (SIGA + SIGB) / SQRT(2 * PI * SIGB) *
  EXP(-0.5 * (X - MEAN) / SIGB**                        for X >= MEAN
```

The parametrization uses AR2/AREA so that you can fix the ratio of the areas, and SIGB/SIGA so you can fix the ratio of the slopes. In addition the ratio of the both the slopes for the two Gaussians is same and is the parameter SIG2/SIG1.

Trigonometric

The form used is:

```
ASINE * SIN(OMEGA * x) + ACOSINE * COS(OMEGA * x)
```

Histogram

Use one histogram to fit to another. You will be asked if you want to include the errors on the histogram you are fitting with in the χ^2. Clearly the errors will not be included if you try to likelihood fit even if you ask them to be.

This is very useful if you want to fit a complicated shape you got from Monte Carlo or an efficiency corrected function to a plot.

ARGUS_Background

This function is the form ARGUS uses to parametrize their background for fitting B mass peaks. The form used is:

```
P(1) * (x + P(2)) * SQRT(1 - ((x + P(2)) / P(3))**2) *
 EXP(P(4) * (1 - ((x + P(2))) / P(3))**2)
where P(1) is the normalization
      P(2) is an offset which should always be fixed when fitting
           If you plot the mass P(2) should be 0,
           but if you plot (Mass - Ebeam), P(2) should be Ebeam
      P(3) is the beam energy
      P(4) is a scale factor for the exponential (0 to not use it)
```

CB_Line_Shape

This function uses the NaI line shape as obtained by the Crystal Ball experiment. The form used is:

```
AREA * exp( -.5*((Et-Em)/s)**2 )                            for Em > Et - a*s
AREA * (n/a)**n * exp(-.5*a**2) / (((Et-Em)/s +n/a -a )**n) for Em < Et - a*s
where
 1/A = s*( (n/a)*exp(-.5*a**2)/(n-1.) + sqrt(pi/2.)*(1.+erf(a/sqrt(2.)) ))
 Em is measured energy
 Et is true energy
 s(igma) is energy resolution
 n and a(lpha) are empirical parameters as far as I can tell.
```

Dipion_Inv_Mass

This is a set of functions containing theoretical formulae for the dipion invariant mass spectrum from J/Psi and Upsilon decays.

Fragmentation_Functions

This is a set of functions containing various theoretical predictions for the fragmentation of quarks and gluons.

Resonance_Cross_Section

Parametrizes a resonance cross section such as an Upsilon state.

Continuum_Cross_Section

Parametrizes the continuum cross-section, including new flavour thresholds and the beam energy dependence, for e+e- annihilation.

COMIS

```
  Syntax: FUN ADD COMIS filename [nfun]
          nparm
          Title
          Name(s)
   where: filename  is the file containing the function
          nfun      is the (optional) user function number
          nparm     are the number of parameters in the function
          Title     is a title for the user function (<CR> means the
                    title will be "Comis Function [name]"
          Name(s)   are the names of the parameters
```

Uses the CERN COMIS facility (COmpilation and Interpretation System) to enable you to define FORTRAN functions interactively and execute them without recompiling the code. The form of the function is identical to that of a user defined function. The function name can be whatever you like, but the arguments must be the same as the user function:

```
DOUBLE PRECISION FUNCTION XMNUSR(X,Y,NP,DPAR,NFUSER,WFERR)
where X      is the current x value for which the function is to be
             evaluated
      Y      is the current y value for which the function is to be
             evaluated
      NP     is the number of parameters
      DPAR   is a DOUBLE PRECISION array containing the parameters
      NFUSER is the user function number (set by the command
             FUN ADD USER n, where n is the user function number)
      WFERR  is the error on the function (usually 0, except when you
             are fitting with histograms) in DOUBLE PRECISION.
```

All the arguments are input, except `WFERR`. An example of a COMIS function can be found in `mn_fit_test:comis_test.for` (Vax) or `~/mn_fit_test/comis_test.for` (Apollo).

The orthogonality limits for legendres etc. and the bin widths are stored in the common block:

```
COMMON/MNUSR/XMINNM,XMAXNM,XBINNM,YMINNM,YMAXNM,YBINNM
XMINNM,XMAXNM are the orthogonality limits for the x axis
YMINNM,YMAXNM are the orthogonality limits for the y axis
XBINNM        is the bin width for the x axis
YBINNM        is the bin width for the y axis
```

If you want to have debug printout you can use the common block:

```
COMMON/MNDBG/QDEBUG,NDEBUG
QDEBUG             is a logical set with the SET DEBUG command
NDEBUG             is the printout level for DEBUG output
```

You will be prompted for the file containing the function and if the file does not exist a new one will be created with the correct function name and arguments supplied. You can then edit the file. The default edit command used is `EDIT/TPU`. Use the `SET EDIT` command to change to your favourite editor. If the file is a new one the `EDIT` command will be executed automatically. When you exit the editor, the function will be compiled and is then ready to be used. Refer to the COMIS manual as to how to edit the function if it has compilation errors or type `HELP` if you get the prompt `MED>`.

If you want your user function to be displayed as a histogram instead of a smooth curve then the title should start with `Hist`.

To change a COMIS function you have already added either edit it and use the `FUNCTION COMPILE` command or use the `FUNCTION EDIT` command.

Note that the file must be in a directory that you have write access to, and that its name must begin with a letter. These are Comis restrictions.

Note that single precision constants cannot be assigned directly to double precision variables in a COMIS routine - but you can use `DBLE(X)` etc. if you need to. Also COMIS is fairly picky about your FORTRAN. For example if `DVAL` is in `DOUBLE PRECISION`, the `IF` statement `IF(DVAL.LT.36.0)` will always be `.FALSE.`. You must use the form `IF(DVAL.LE.36.0D0)`.

Mn_Fit_functions_in_COMIS All Mn_Fit functions (except the dipion mass spectra) may be called in a COMIS routine. The function names and arguments are listed below. Note that the functions are all `DOUBLE PRECISION`. The arguments generally used are:

```
NPAR=  Number of Parameters
P=     Parameters (Double Precision)
X, Y=  X and Y
```

Other arguments are listed with the functions:

```
1) General Functions (1-28):  XMNCLC(NFUN,P,NPAR,NPAR2,X,Y)
   NFUN=  Function Number
   NPAR2= Number of secondary parameters

2) Continuum Cross-Section:  XMNCNT(P,NPAR,X)

3) Fragmentation Functions:  XMNFRG(MODEL,BIN,P,NPAR,X)
   BIN=   Unused

4) Fit to Histogram:  XMNHIS(IDHA,IDHB,MODE,X,Y,WMNHER)
   IDHA=   Primary Identifier of Histogram
   IDHB=   Secondary Identifier of Histogram
   NMODE=  0= No Errors, 1= Errors from histogram
```

```
  WMNHER= Error returned from XMNHIS - DOUBLE PRECISION

5) Resonance Cross-section:  XMNRES(P,NPAR,X)

6) 2D Functions: XMNC2D(NFUN,P,NPAR,NPAR2,X,Y)
```

User

```
Syntax: FUN ADD USER [nfun]
        nparm
        Title
        Name(s)
 where: nfun    is the (optional) user function number
        nparm   are the number of parameters in the function
        Title   is a title for the user function (<CR> means the
                title will be "User Function [nfun]"
        Name(s) are the names of the parameters
```

User functions are identical to COMIS functions except that they must have the function name XMNUSR, so see section 4.41.9 on page 72 (FUNCTION LIST COMIS) for details.

To link Mn_Fit with a user function you can use the command file **mn_fit_dir:mn_user.lnk** (Vax) or **~/mn_fit_dir/mn_user.lnk** (Apollo).

You just have to give the executable name (default is mn_user.exe), the filename for your compiled function and any extra libraries or object files that you need.

4.41.10 OVERLAY

```
Syntax: FUNCTION OVERLAY nfun1 [nfun2...]
        [&]idb nsymb
 where: nfun  is the function number(s) as in FUNCTION INFO
        idb   is the secondary identifier for the function
   and  nsymb is the symbol you want to plot the function with
```

Stores one of the functions being fitted as a histogram and overlays it on the current picture. **nfun** = 0 means use all functions which are currently in use.

If you are fitting and your last picture was a DISPLAY, the function will be given the same primary identifier as the one you are fitting. Otherwise, it will be given the primary identifier 98765. It will be given the secondary identifier you specify. To specify the secondary identifier on the same line as the function numbers precede it by a **&**. The function will be drawn with 500 points (unless at least one of them is a histogram), so it will appear as a smooth curve.

4.41.11 PLOT

```
Syntax: FUNCTION PLOT nfun1 [nfun2...]
        [&]idb nsymb
        [xlo xhi]
 where: nfun  is the function number(s) as in FUNCTION INFO
```

```
        idb   is the secondary identifier for the function
    and  nsymb is the symbol you want to plot the function with
```

Stores one of the selected functions as a histogram and plots it. **nfun** = 0 means use all functions which are currently in use. You will be prompted for the lower and upper limits for the plot. The plot will be given the primary identifier 98765. It will be given the secondary identifier you specify. To specify the secondary identifier on the same line as the function numbers precede it by a **&**. The function will be drawn with 500 points (unless at least one of them is a histogram), so it will appear as a smooth curve.

4.41.12 STORE

```
Syntax: FUNCTION STORE filename n1 [n2...]
 where: n1,n2... are the function numbers you want to store
```

Stores functions in a file. You can access them at some other time with the **FUNCTION FETCH** command.

4.41.13 USE

```
Syntax: FUNCTION USE n1 [n2...]
 where: n1,n2 are the function number(s) as in FUNCTION INFO
```

Only valid in **MN_CMD**>.

Selects the function(s) you want to use in fitting. **n** positive means use the function, negative means do not.

4.42 HARDCOPY

```
Syntax: HARDCOPY [device]
```

The first time you ask for a hardcopy you will be prompted for the hardcopy device you want. All subsequent plots will go to the same device, unless you say **HARD new_device**. If you give the **HARDCOPY** command in a macro and omit the device name the first time, device **METAFILE** will be selected. To print the plots you have already made issue the command **CLOSE**. You can then print the file from another session, or use the **SPAWN** or **SHELL** commands.

Different hardcopy devices are supported for GKS and PLTSUB. See section 1.16 on page 29 (Hardcopy_Devices) for more details. If you want to use a device other than those listed, you must give its number (see GKS or PLTSUB manuals). It will be captured as device type **unknown** and the output file will be **plot.dat** unless you have specified otherwise.

4.43 HB3_FETCH

```
Syntax: HB3_FETCH filename id1 [:id2] [id3...]
```

Fetches HBOOK version 3 histograms from a file. To fetch all histograms, give the command **HB3_FETCH filename 0**. To specify a range of histograms, give the command **HB3_FETCH filename id1:id2**. If you want to fetch other histograms from the same file give the command **HB3_FETCH id1,id2....** If the histogram number you ask to fetch already exists, it will be overwritten. All the histograms will be given the default secondary identifier.

WARNING: If you fetch all histograms from a file or specify a range any existing HBOOK histograms will be deleted! However, Mn_Fit will still store the histograms internally. This means that you will lose any slices, bands projections or functions associated with the HBOOK histogram unless you have already extracted them into Mn_Fit plots.

4.44 HB_FETCH

```
Syntax: HB_FETCH filename id1 [:id2] [id3...]
```

Alias for **FETCH**. See section 4.38 on page 63 (FETCH) for more details.

4.45 HB_MN_FIT

```
Syntax:   HB_MN_FIT
```

Converts all HBOOK histograms in the current HBOOK directory into Mn_Fit histograms. This command is automatically invoked after **NTUPLE SCAN** and **CALL_COMIS**. However if you have made HBOOK histograms in directories other than the current one when you exit the COMIS subroutine you should give the commands:

```
CDIR newdir
HB_MN_FIT
```

to get those histograms into Mn_Fit memory.

4.46 HB_OPEN

```
Syntax: HB_OPEN filename
where   filename  is the name of the file to open.
```

Opens an HBOOK4 file. This command is not usually necessary if you just want to fetch histograms, but is useful in connection with the commands **CDIR** and **LDIR** (or **SET DIRECTORY** and **SHOW DIRECTORY**). You can then find out what histograms are in a file and fetch histograms from several subdirectories and give them separate secondary identifiers.

The record length for direct access HBOOK version 4 files must be known. The default is 1024 (Vax) and 4096 (Apollo). If your files have been made with a different record length, use the **SET RECL** command to set the length. This length will also be used for **STORE** commands.

4.47 HB_STORE

```
Syntax: HB_STORE filename id1 [id2...]
```

Alias for **STORE**. See section 4.86 on page 117 (STORE) for more details.

4.48 HCOPY

```
Syntax: HCOPY id1 id2
```

Calls subroutine HCOPY to copy one HBOOK histogram to another.

4.49 HDELETE

```
Syntax: HDELETE id
```

Calls subroutine HDELET to delete an HBOOK histogram.

4.50 HINDEX

```
Syntax: HINDEX
```

Calls subroutine HINDEX to give an index of the HBOOK histograms currently in memory.

4.51 HISTOGRAM

Do something with a plot. See section 2.4 on page 34 (HISTOGRAM Menu) for a short description of each command.

4.51.1 BOOK

```
Syntax: BOOK[/BINNED|/UNBINNED|/ERROR|/NOERR|/ASYMMETRIC] id [&idb]
        title
        ndim
        maxpnt  OR nbinx xlo xhi [nbiny ylo yhi ...]
 where: id      is the plot identifier
        idb     is the (optional) secondary identifier
        title   is the title for the new plot
        ndim    is the number of dimensions
        maxpnt  is the maximum number of points (for an unbinned plot)
        nbinx   is the number of x bins
        xlo     is the lower limit on x
        xhi     is the upper limit on x etc.
```

Book a new plot inside Mn_Fit. The default is an unbinned plot with errors. This command is useful if you want to fill a plot with the results of a fit for example. See section 4.51.6 on page 79 (HISTOGRAM FILL) for more details. Note that the `title` must be included in quotes if it is on the same line as the number of dimensions.

/BINNED

Books a binned plot. You will be promped for the number of bins and the upper and lower limits for each variable.

/UNBINNED

Books an unbinned plot. When you fill the plot the data you give will be stored as the next point. Use the PROJECT command to convert an unbinned to a binned plot if you want to do so.

/NOERRORS

Plot will be booked with no space reserved to store the errors on the data. Use the HIST ERROR command to add errors at a later time if you wish to.

/ERRORS

Plot will be booked with space reserved for errors. When you fill the plot you will be prompted for the errors if it is an unbinned plot.

/ASYMMETRIC

Plot will be booked with space reserved for asymmetric errors. When you fill the plot you will be prompted for both the negative and positive errors if it is an unbinned plot.

4.51.2 DISPLAY

```
Syntax: HIST DISPLAY[/CLEAR|/NOCLEAR] detnam id [&idb]
 where: detnam  is the detector name
        id      is the histogram identifier
        idb     is the (optional) secondary identifier
```

Makes a special display. This has been implemented for the following subdetectors in the L3 experiment: ECAL, FBGO and FWCH.

/CLEAR

Default option. Specifies that the screen will be cleared before the next plot is made. Note that if you are plotting with more than 1 window (see section 4.79.66 on page 111 (SET WINDOW)) this option will be overridden, except for the plot in the top lefthand corner (WINDOW 1 1).

/NOCLEAR

Specifies that the screen not be cleared before drawing this plot. This is useful if you want to overlay displays.

4.51.3 DUMP

```
Syntax: HIST DUMP id [&idb]
 where: id   is the histogram identifier
        idb  is the (optional) secondary identifier
```

Dumps the contents of a histogram. If you omit the secondary identifier the default is used.

4.51.4 ERRORS

```
Syntax: HIST ERROR id [&idb] nmode
 where: id    is the histogram identifier
        idb   is the (optional) secondary identifier
        nmode is the mode: 0 = zero errors,
                           1 = square root of the number of entries,
                           2 = same as 1, except the error on 0 points
                               is set to 1.
```

Changes the errors for a histogram. By default all HBOOK histograms get errors assigned to the bin contents when they are read in. If errors were not stored with the histogram they are set to the square root of the bin contents. Note that the errors are included in the calculation of the minimum and maximum limits when plotting the histograms.

4.51.5 EXTRACT

```
Syntax: HIST EXTRACT id part [npart] [idb]
 where: id    is the histogram identifier
        part  is the name of the part you want to extract
              HIST, FUN, PROX, PROY, SLIX, SLIY, BANX, or BANY
        npart is the slice or band number
        idb   is the secondary identifier for the part
```

Extracts part of an HBOOK histogram, such as an associated function, projection, slice or band that you have filled. You cannot make a part that you have not already booked. To make a slice or band use the CUT and PROJECT commands.

4.51.6 FILL

```
Syntax: HIST FILL id [&idb] x y [dx [dy] ...]
    or  HIST FILL id [&idb] x [y] [weight]
 where: id      is the plot identifier
        idb     is the (optional) secondary identifier
        x       is the x value of the point
        y       is the y value of the point
        dx      is the error on x
        dy      is the error on y etc.
        weight  is the weight for that point
```

Fills a plot. The first syntax is for an unbinned plot and the second for a binned plot. The values and errors can be numbers, registers, parameters, etc. See section 1.6 on page 17 (Numbers) for more details on the things available and see section 4.27 on page 56 (DEPOSIT) on how to fill registers etc.

Examples

1. If you want to fill the plot with an upper limit, for example, use the DEPOSIT command to calculate the upper limit, storing the answer in a register and then FILL the plot with that value:

```
DEPOSIT R10 = P2(1) + 1.64*ERP2(1)
FILL 10 0.5 R10
```

4.51.7 LEGO

```
Syntax: HIST LEGO id [&idb] theta phi        (default theta=30, phi=30)
```

Make a lego plot of a 2-dimensional histogram. **Theta** and **phi** are the angles from which you want to view the lego plot in degrees. They must lie between 0 and 90.

4.51.8 OVERLAY

```
Syntax: HIST OVERLAY[/SAME|/DIFFERENT] id [&idb] symbol hatch pattern
```

Overlays a histogram on the last one you plotted. If you omit the secondary identifier, the default is used. You will be prompted for the symbol number, hatching and pattern. Only the symbol number is required. Hatching and patterns only work with the HIGZ version of Mn_Fit and are device dependent (see section 4.79.24 on page 99 (SET HATCH) and section 4.79.43 on page 104 (SET PATTERN) for more details).

By default the plot will have the same y scale as the last one (/SAME qualifier). If you want it on a different y scale, give the command OVERLAY/DIFFERENT. The new scale will be drawn on the right-hand side of the plot. Give the command REDRAW to get rid of the scale on the right from the first plot.

/SAME

Default option. Specifies that the overlayed plot will have the same y scale as the first one.

/DIFFERENT

Specifies that the overlayed plot will have its own y scale drawn on the right hand side of the plot.

4.51.9 PLOT

```
Syntax: HIST PLOT[/CLEAR|/NOCLEAR] id1[:id2] [&idb1:idb2]]
    or  HIST PLOT[/CLEAR|/NOCLEAR] id part
 where: part  is the part of an HBOOK histogram that you want to plot
              e.g. BANX, FUN etc.
```

Plot a histogram on the currently selected screen device. If you omit the secondary identifier, the default is used. You can plot a range of histograms by using the syntax **id1:id2** for either or both the primary and secondary identifiers. The default option is **/CLEAR**. If you want to add a second plot as an insert, for example, use the **/NOCLEAR** qualifier and specify the size and position of the second plot with the **SET X|Y MARGIN and SIZE** commands. If the plot is within a window you should use the **SET X|Y WMARGIN and WSIZE** commands.

/CLEAR

Default option. Specifies that the screen will be cleared before the next plot is made Note that if you are plotting with more than 1 window (see section 4.79.66 on page 111 (SET WINDOW)) this option will be overridden, except for the plot in the top lefthand corner (**WINDOW 1 1**).

/NOCLEAR

Specifies that the screen not be cleared before drawing this plot. This is useful if you want to make inserts and specify exactly where they should go and how big they should be (**SET X|Y MARGIN** and **SIZE** or **SET X|Y WMARGIN** and **WSIZE** commands).

4.51.10 SURFACE

```
Syntax: HIST SURFACE id [&idb] theta phi      (default theta=30, phi=30)
```

Make a surface plot of a 2-dimensional histogram. **Theta** and **phi** are the angles from which you want to view the plot in degrees. They must lie between 0 and 90.

4.52 HMAKE

```
Syntax: HMAKE id [&idb]
 where: id  is the histogram identifier
        idb is the (optional) secondary identifier
```

Makes an HBOOK histogram from another sort of histogram. The HBOOK histogram will have the same identifier.

4.53 HRENAME

```
Syntax: HRENAME id1 id2
```

Renames an HBOOK histogram.

4.54 HY_FETCH

```
Syntax: HY_FETCH filename id1[&idb] [:id2] [id3...]
```

Fetches HYBRID histograms from a file. To fetch all histograms give the command `HY_FETCH filename 0`. To specify a range of histograms, give the command `HY_FETCH filename id1:id2`. If you want to fetch other histograms from the same file give the command `HY_FETCH id1,id2...` To fetch 2-dimensional plots give the command `HY_FETCH filename 0&2`. To fetch 1 and 2-dimensional plots give the command `HY_FETCH filename 0&1:2`. All the 1-dimensional histograms will be given the default secondary identifier. The 2-dimensional histograms will be given the default secondary identifier +1.

4.55 INDEX

```
Syntax: INDEX [id1][:id2] [&idb][:idb2]
```

Gives an index of the histograms currently in memory. If you specify the primary identifier, you will get an index of all histograms with that identifier. To get all the histograms with a particular secondary identifier, give the command `INDEX 0 &idb`. To get all the histograms in a particular range, give the command `INDEX id1:id2 &idb1:idb2`.

4.56 INQUIRE

```
Syntax:   INQUIRE par [prompt]
where:    par     is the parameter number
          prompt  is the optional prompt
Defaults: prompt = 'Give parameter n:'
```

Only valid inside macros. If a parameter has not been defined you are prompted to give the value of the parameter. You can force a new value for the parameter by giving a negative parameter number. A : will automatically be added at the end of the prompt:

```
INQUIRE 1 'Give label'
inq -3
```

4.57 INTEGRATE

```
Syntax: INTEGRATE nfun1 [nfun2...]
        xlo xhi
 where: nfun1,nfun2 are the function number(s) to integrate
        xlo         is the lower limit of the integration
        xhi         is the upper limit of the integration
```

Integrates one or more functions over the given range. The default number of points to sum over is 100. If you want to integrate all selected functions, give the function number 0. The result is printed and is also stored in register 101, which you can use with the syntax `R101`.

4.58 KEY

```
 Syntax: KEY
     or  KEY id [&idb]
     or  KEY command [nkey]
     or  KEY id [&idb] command [nkey]
         nsymb
         text
         x, y, size, angle, option, mode, font, colour, thickness
  where: id        is the histogram number the key applies to
         idb       is the optional secondary identifier
         command   can be ?|NEW|CHANGE|DELETE|LIST|END
         nkey      is the key number to change or delete
         nsymb     is the symbol number to describe
                   Use the form +/-(1000 + abs(nsymb)) for hatching
                                +/-(2000 + abs(nsymb)) for patterns
                   where the sign is the sign of the hatch or
                   pattern number
         text      is the text you want to display
         x,y       are the position of the key
         size      is the size of the text in cm (default = 0.4cm)
         angle     is in degrees with respect to the horizontal
         option    can be: LEFT   Key is left adjusted
                           CENTRE Key is centred (default)
                           RIGHT  Key  is right adjusted.
         mode      can be: CM   = Position is in cm (default)
                           PLOT = Position in terms of plot co-ordinates
         font      is the font to use
         colour    is the colour number for the key
         thickness is the text thickness factor
```

When the KEY command is given in a file or a defined command, the last syntax must always be used and the key number must be given if it is applicable to the command (CHANGE or DELETE). You always need the END command when reading from a file to exit KEY unless you give the command on the same line as KEY

Keys are associated with histograms you have already plotted. Therefore, if you have only plotted one histogram on the picture (using the PLOT, LEGO, SURFACE, DISPLAY or OVERLAY/DIFF commands - OVERLAYS on the same scale do not count), the syntax is the first line. If you already have a key, you will be asked if you want a new one or to move or change the old one. If you have plotted more than one histogram on the display, you will be prompted for the identifier the key applies to.

If you just give the command KEY or KEY id[&idb] you will remain in KEY until you give the END command or <CR>. However, if you give some or all of the rest of the KEY command on the same line you will automatically exit KEY after you have finished the command.

The position you give is where the symbol will be drawn. The text will be started twice the text size to the right of the symbol. The size of the symbol will be the same as the text size.

By default the text will be written using the HIGZ routine IGTEXT. For details on the other fonts available, see section 1.12 on page 24 (FONTS). For `mode CM`, x=`0`, y=`0` is the bottom left-hand corner of the picture You can omit all of the parameters except x and y for a new key. The option for the key will always be set to `LEFT`, whether you try to change it or not. If you change or delete a key, use the command `REDRAW` to see the effect of what you have done.

4.59 LDIRECTORY

```
Syntax: LDIRECTORY
```

Alias for `SHOW DIRECTORY`.
List of the contents of the currently selected HBOOK directory in an HBOOK4 file.

4.60 LEGO

```
Syntax: LEGO id [&idb] theta phi           (default theta=30, phi=30)
```

Alias for `HIST LEGO`. Makes a lego plot of a 2-dimensional histogram. `Theta` and `phi` are the angles from which you want to view the lego plot in degrees. They must lie between 0 and 90.

4.61 MESSAGE

```
Syntax: MESSAGE text
 where: text  is the text of the message
```

Writes a message to the current output device (selected by `SET DUMP`. The default is to the terminal). This command is most useful in macros to let the user know what is going on. The text can include registers, parameters etc., using the form {`Parameter,(Format)`}. See section 1.11 on page 23 (Text) for more details.

4.62 MN_FETCH

```
Syntax: MN_FETCH filename id1[:id2] [&idb1[:idb2]] [id3...]
 where: filename     is the file the plots are stored in
        id1,id2,id3  are the identifiers you want to fetch
```

Fetches plots made with the M_BOOK package or ones stored with the `MN_STORE` command. To fetch all the plots, use the command `MN_FETCH filename 0`. To fetch a range of plots, use the command `MN_FETCH filename id1:id2`, or `MN_FETCH filename 0&idb1:idb2` to get all the plots with secondary identifiers between `idb1` and `idb2`.

4.63 MN_STORE

```
Syntax: MN_STORE filename id1[:id2] [&idb1[:idb2]] [id3...]
 where: filename is the file to store the plots
        id1,id2  are the identifiers you want to store
```

Stores plots in a format for reading with the MN_FETCH command. To store all the plots, use the command MN_STORE filename 0. To store a range of plots, use the command MN_STORE filename id1:id2, or MN_STORE filename 0&idb1:idb2 to store all the plots with secondary identifiers between idb1 and idb2. If you omit the secondary identifier the default will be used, and only plots with the default secondary identifier will be stored (unless you say MN_STORE filename 0 which stores all the plots you have made or fetched).

4.64 MULTIPLY

```
Syntax: MULTIPLY id1 [:id1n] [&idb1] id2 [:id2n] [&idb2] id3 [:id3n] [&idb3]
        [scale1] [scale2]                          (default scale = 1.0 1.0)
 where: id1,id2   are the input histogram identifiers
        id3       is the output histogram identifier
        idb1,idb2 are the (optional) input secondary identifiers
   and  idb3      is the (optional) output secondary identifier
```

Multiplies two histograms (id1, id2) together to make a third one. To specify the secondary identifier, precede it by a &, otherwise the default will be used. (Use the SET IDB command to change the default). The scale factors are optional. To avoid confusion, you should give a <CR> after the identifiers or make sure the scale factors are given as real numbers.

To multiply a range of histograms, the primary identifiers you give for the input and output histograms must be the same, but you can specify different secondary identifiers. For example, MULT 300:400&1 300:400&2 300 : 400 & 10 will multiply all histograms with primary identifiers 300 to 400 and secondary identifiers 1, by those with secondary identifiers 2, putting the results into histograms with the same primary identifiers and secondary identifier 10. If you give primary identifier 0, the operation will be performed on all plots with the given secondary identifier.

4.65 NORMALIZE

```
Syntax: NORMALIZE id1 [:id1n] [&idb1] id2 [:id2n] [&idb2]
        [normalization]                    (default normalization = 1.0)
    or  HIST id3 [&idb3]
 where: id1  is the input histogram identifiers
        id2  is the output histogram identifier
        idb1 is the (optional) input secondary identifier
   and  idb2 is the (optional) output secondary identifier
```

Normalizes a histogram (id1) to the given area or to the area of another histogram. To specify the secondary identifier, precede it by a &, otherwise the default will be used. (Use the SET IDB command to change the default). The scale factors are optional. To avoid confusion you should give a <CR> after the identifiers or make sure the scale factors are given as real numbers.

To normalize a range of histograms, the primary identifiers you give for the input and output histograms must be the same, but you can specify different secondary identifiers. For example, NORM 300:400&1 300 : 400 & 10 10.0 will normalize all histograms with primary identifiers

300 to 400 and secondary identifiers 1, to an area of 10, putting the results into histograms with the same primary identifiers and secondary identifier 10. If you give primary identifier 0, the operation will be performed on all plots with the given secondary identifier.

4.66 NTUPLE

```
Syntax: NTUPLE DUMP|MERGE|PLOT|PROJECT|SCAN
```

Does an operation on an Ntuple. If you want to apply cuts you should first define them using `CUT NEW` or `CUT FILE`, and `CUT USE` to specify which cuts should be used. Cuts will be applied when using the `PLOT` or `PROJECT` commands.

Ntuple commands are still under development. It is possible that syntaxes will change and very likely that new commands and options will be added.

When you use a Comis function 5 common blocks are available to you. (At present (version 3.01) `MNTPL2` is not available as it is not possible to pass a common block starting with a character variable to COMIS):

```
COMMON/PAWIDN/IDNEVT,VIDN1,VIDN2,VIDN3
 , X1, X2, ...
COMMON/MNTPL1/ID,NVAR,ALO(NVAR),AHI(NVAR)
COMMON/MNTPL2/TITLE,TAGS(NVAR)
COMMON/MNREGI/REGIS(0:300)
 , VAR1, VAR2, ...
COMMON/MNDBG/QDEBUG,NDEBUG
```

```
IDNEVT is the current Ntuple event number
VIDN1,VIDN2,VIDN3 are the values of the variables you are projecting onto
X1,X2...          are the Ntuple variables for the current event
ID                is the Ntuple identifier
NVAR              is the number of variables
ALO               are the lower limits for each variable
AHI               are the upper limits for each variable
TITLE             is the Ntuple title
TAGS              are the variable names
REGIS             are Mn_Fit registers
VAR1,VAR2         are user defined Mn_Fit variables
QDEBUG            is a logical set with the SET DEBUG command
NDEBUG            is the printout level for DEBUG output
```

Note that the `PAWIDN` common block is compatible with PAW so that you should be able to use the same functions in Mn_Fit and PAW. When you specify a Comis function as a cut it will be a real function. For a cut and `SCAN` the name of the function or subroutine will be the name of the file without the header and type. When you give the cut to use you can give an argument for the function.

User variables are stored in registers >300. Therefore you can either use the variable names or `REGIS(301)` etc. in the subroutine or function. These function as a substitute for KUIP vectors which are not available.

4.66.1 DUMP

```
Syntax: NTUPLE DUMP id [&idb]
where   id   is the ntuple identifier
and     idb  is the (optional) secondary identifier
```

Dumps the contents of an Ntuple.

4.66.2 MERGE

```
Syntax: NTUPLE MERGE id fileout filein1 [filein2 ...]
where   id       is the ntuple identifier
        fileout  is the name of the output file for the Ntuple
        filein1  is the first filename etc.
```

Merges several Ntuples with the same identifier into a single Ntuple. A <CR> signifies that the merging has finished. The merged Ntuple will then be read in and is available for further manipulation.

4.66.3 PLOT

```
Syntax: NTUPLE PLOT id[&idb] var1 [var2 ...] &idb2 nbinx [xlo xhi]
        [nbiny ylo yhi]
where   id     is the ntuple identifier
        var1   is the first variable or expression to project onto
        var2   is the second etc.
        idb2   is the secondary identifier of the projection
        nbinx  are the number of bins for the x-axis of the projection
               if the input is an Ntuple.
               -1 means automatic binning
                0 means keep as an Ntuple
        xlo    is the lower limit for the x-axis
        xhi    is the upper limit for the x-axis
        nbiny  are the number of bins for the y-axis
        ylo    is the lower limit for the y-axis
        yhi    is the upper limit for the y-axis
```

Projects and plots the projection of an Ntuple or an n-dimensional histogram. For more details see section 4.66.4 on page 87 (NTUPLE PROJECT).

4.66.4 PROJECT

```
Syntax: NTUPLE PROJECT id[&idb] var1 [var2 ...] &idb2 nbinx [xlo xhi]
        [nbiny ylo yhi]
where   id     is the ntuple identifier
        var1   is the first variable or expression to project onto
        var2   is the second etc.
        idb2   is the secondary identifier of the projection
```

```
nbinx  are the number of bins for the x-axis of the projection
       if the input is an Ntuple.
       -1 means automatic binning
        0 means keep as an Ntuple
xlo    is the lower limit for the x-axis
xhi    is the upper limit for the x-axis
nbiny  are the number of bins for the y-axis
ylo    is the lower limit for the y-axis
yhi    is the upper limit for the y-axis
```

The project command has several uses. For example if you have specified cuts on a plot, use the PROJECT command to make a second plot which shows the results of the cuts; or if you have a scatter plot and want to make a binned histogram out of it.

The end of the list of variables is signified by the end of line or the & of the secondary identifier. As for cuts you can give a variable name or an expression for the projection. You can also give the variable number instead. An expression is recognized if it starts with a (or if it contains any arithmetic operator +-*/^(). Thus if, for some strange reason you want to use a register for an axis you would have to enclose the register in parentheses (R1) so that it is recognized as an expression. When an expression is parsed, first Ntuple variable names are searched for, then user defined variables and then the standard registers, parameters etc. At present you can only project onto a maximum of 2 variables.

You only need to specify the binning if the input is an Ntuple. If you are projecting a scatter plot, specify 0 bins to keep it as a scatter plot, or give the binning if you want the projection to be a histogram. If you want to plot the result directly you can use the NTUPLE PLOT command.

Examples

1. Project an Ntuple onto a 2-D histogram using automatic binning:

```
FET 10 filename
PROJ 10 VAR1 VAR2 &1 -1
```

2. Project the sum of 2 variables and specify the binning:

```
PROJ 10 (VAR1+VAR2) &2 100 0 50
```

3. Make a scatter plot of an expression and a variable using a COMIS function as a cut plus a cut on another variable:

```
CUT DEL 0
CUT
NEW VAR4.GT.5
FILE CUT_NTUPLE.FOR
USE 1 .OR. 2
END
NTUPLE PLOT 10 var3 (var1/var2) &3 0
```

4. Project a 2-D histogram onto 1 of its axes:

```
NTUPLE PLOT 10 1 &1
```

5. Project a 2-D histogram onto the sum and difference of the axes. If you project onto an expression the lower and upper limits on each variable will be used and the expression evaluated for them. These will then be the default limits for the projection:

```
NTUPLE PROJ 1 (X+Y) X-Y &4  -1
```

4.66.5 SCAN

```
Syntax: NTUPLE SCAN id[&idb] filename[(arg)] Y|N
where   id        is the ntuple identifier
        idb       is the (optional) secondary identifier
        filename  is the name of a file containing the Comis subroutine
                  that should be called
        arg       is an optional argument for the subroutine
```

Loops over all the events in an Ntuple and calls a subroutine which is contained in `filename`. If the file does not exist a skeleton file with the common blocks containing information on the Ntuple will be created. You will be asked if you want to edit the file or not. The subroutine will be called with the optional argument **arg**, which can be a register, parameter etc. If it is not given it will be set to `0.0`.

After the scan all HBOOK histograms in the current directory in memory will be read in as Mn_Fit histograms. If you also want to get histograms from another directory you can use the command `HB_MN_FIT` (see section 4.45 on page 76 (HB_MN_FIT) for more details).

4.67 OVERLAY

```
Syntax: OVERLAY[/SAME|/DIFFERENT] id [&idb] symbol [hatch pattern]
```

Alias for `HIST OVERLAY`. See section 4.51.8 on page 80 (HISTOGRAM OVERLAY) for more details.

4.68 PARTITION

```
Syntax: PARTITION id [&idb] xlo xhi [ylo yhi]
 where: id  is the histogram identifier
        idb is the (optional) secondary identifier
        xlo is the new lower limit for the x axis
        xhi is the new upper limit for the x axis
        ylo is the new lower limit for the y axis (only for 2-d plots)
        yhi is the new upper limit for the y axis (only for 2-d plots)
```

Cuts out part of a plot between the limits `xlo` and `xhi` (and between `ylo` and `yhi` if it is a 2-dimensional plot). The new plot will be stored with the same identifiers as the old one. To make one with different identifiers use the `CUT` and `PROJECT` commands, or `COPY` then `PARTITION`.

4.69 PLOT

```
Syntax: PLOT[/CLEAR|/NOCLEAR] id1[:id2][&idb1[:idb2]]
    or  PLOT id part
```

Alias for HIST PLOT. See section 4.51.9 on page 80 (HISTOGRAM PLOT) for more details.

4.70 PRINT

```
Syntax: PRINT id [&idb]
 where: id  is the histogram identifier
        idb is the (optional) secondary identifier
```

Prints a histogram on the terminal or in a file. The output is set using the `SET DUMP` command. If you want to make a line printer copy of the histogram, use the commands:

```
SET DUMP LPT
PRINT id [&idb]
SET DUMP CLOSE
```

Then either `SPAWN` a print command or go to another session and print the file `mn_dump.dat`. You can change the filename with the `SET DUMP FILE` command.

4.71 PROJECT

```
Syntax: PROJECT id[&idb] var1 [var2 ...] &idb2 nbinx [xlo xhi]
        [nbiny ylo yhi]
where   id      is the ntuple identifier
        var1    is the first variable or expression to project onto
        var2    is the second etc.
        idb2    is the secondary identifier of the projection
        nbinx   are the number of bins for the x-axis of the projection
                if the input is an Ntuple.
                -1 means automatic binning
                 0 means keep as an Ntuple
        xlo     is the lower limit for the x-axis
        xhi     is the upper limit for the x-axis
        nbiny   are the number of bins for the y-axis
        ylo     is the lower limit for the y-axis
        yhi     is the upper limit for the y-axis
```

Projects a plot (either a scatter plot or an n-dimensional histogram) onto 1 or more axes. Alias for `NTUPLE PROJECT`. See section 4.66.4 on page 87 (NTUPLE PROJECT) for more details.

4.72 READ

```
Syntax: READ COMMAND|DATA
```

Reads in a series of commands from a file or a histogram stored in card image format. See section 2.6 on page 35 (READ Menu) for a short description of the commands.

4.72.1 COMMAND

```
Syntax: READ COMMAND filename [parameter_list]
```

Alias for EXEC. See section 4.36 on page 61 (EXECUTE) more more details.

4.72.2 DATA

```
Syntax: READ DATA filename
```

Alias for DAT_FETCH. See section 4.21 on page 52 (DAT_FETCH) for more details.

4.73 REBIN

```
Syntax: REBIN id [&idb]
        nbinxlo nbinxhi nbinx
        [nbinylo nbinyhi nbiny]
 where: id      is the  histogram identifier
        idb     is the (optional) secondary identifier
        nbinxlo is the first bin
        nbinxhi is the last bin
        nbinx   is the number of new bins
```

Rebins a plot into nbin new bins from bin nbinlo to bin nbinhi. If (nbinhi - nbinlo) is not divisible by nbin, nbinhi is rounded down until it is. Rebin works on 1 and 2-dimensional histograms.

4.74 REDRAW

```
Syntax: REDRAW
```

Redraw the last picture you made with any changes you made via the SET, COMMENT, KEY or DRAW commands implemented.

4.75 REMOVE

```
Syntax:   REMOVE var|ALL
where:    var is the user variable name
```

Removes the user variable name var. If the command REMOVE ALL is given all user variables will be deleted. See section 1.6 on page 17 (Numbers) and section 4.27 on page 56 (DEPOSIT) for details on defining user variables.

4.76 RENAME

```
Syntax: RENAME id1 [&idb1] id2 [&idb2]
 where: id1   is the input identifier
        idb1  is the (optional) input secondary identifier
        id2   is the output identifier
        idb2  is the (optional) output secondary identifier
```

Renames a plot. If you omit a secondary identifier, the default is used. Use the **SET IDB** command to change the default. If you give **id1** and **id2** = **0** all the plots with secondary identifier **idb1** will be renamed to have secondary identifier **idb2**.

4.77 SCALE

```
Syntax: SCALE id1 [:id1n] [&idb1] id2 [:id2n] [&idb2]
        [scale]                         (default scale = 1.0)
 where: id1  is the input histogram identifiers
        idb1 is the (optional) input secondary identifiers
        id2  is the output histogram identifier
  and   idb2 is the (optional) output secondary identifier
```

Scales a histogram (**id1**) by a given factor. To specify the secondary identifier, precede it by a **&**, otherwise the default will be used. (Use the **SET IDB** command to change the default). The scale factors are optional. To avoid confusion you should give a <CR> after the identifiers or make sure the scale factors are given as real numbers.

To scale a range of histograms, the primary identifiers you give for the input and output histograms must be the same, but you can specify different secondary identifiers. For example, **SCALE 300:400&1 300 : 400 & 10 10.0** will scale all histograms with primary identifiers 300 to 400 and secondary identifiers 1, by a factor of 10, putting the results into histograms with the same primary identifiers and secondary identifier 10. If you give primary identifier 0, the operation will be performed on all plots with the given secondary identifier.

4.78 SCT_FETCH

```
Syntax: SCT_FETCH filename id1[:id2] [id3...]
 where: filename      is the file containing the plots
        id1,id2,id3   are the plot number you want to fetch
```

Fetch scatter plots made with the AVEHST package. To fetch all the plots, give the command **SCT_FETCH filename 0**. To fetch a range of plots, give the command **SCT_FETCH filename id1:id2**.

4.79 SET

```
Syntax: SET
        parameter value(s)
```

```
or  X|Y|Z parameter value(s)
or  SET PLOT id [&idb] parameter value
or  SET PLOT id [&idb] X|Y|Z parameter value
```

In general if you SET something, it will affect the next plot you make. To change something on a picture you have already made, use the format SET PLOT id [&idb]. This will not change the value of that parameter for any other plots. If you want to change something for all plots, use the format SET PLOT 0 ... This will also change the value of the parameter for all subsequent plots. You can usually omit the word PLOT in the above syntax.

If you do not want to change the value of a parameter, you can use = in the position where the value would go, in the same way as in the MODIFY command. See section 1.6 on page 17 (Numbers) for more details. Set the parameters to 0 to get back to their default values or to turn off their effect. For example. SET Y LIMIT 0 0 returns you to automatic limit setting.

If you give the **parameter** on the same line as SET or SET PLOT id [&idb], you will exit automatically at the end of your command. Otherwise you must give the ENDSET command to exit SET if you are reading commands from a file or a DEFINE command. If you are giving commands interactively you can also hit <CR>.

To change a parameter in the DISPLAY of the fit results, use the SET DISPLAY **parameter** form. See section 4.79.44 on page 104 (SET PLOT) and section 4.79.8 on page 95 (SET DISPLAY) for more details.

If a parameter has to be preceded by the axis it applies to, you no longer have to leave a space between the axis label and the parameter name, i.e. SET X LIMIT and SET XLIMIT are both valid.

If you turn off the frame for a plot then no ticks, grid or scale will be drawn. If you turn off the ticks then no scale will be drawn either. Possible positions for the frame, scale, ticks and label are the BOTTOM, TOP, LEFT or RIGHT of the picture. In addition for lego and surface plots you can set parameters for the VERTICAL axis. For the grid and the line at x or y = 0, you specify whether it is for the X, Y, or Z axes.

4.79.1 ALIAS

```
Syntax: SET ALAIS ON|OFF                (default = ON)
```

Turns on or off alias translation on the command line. See section 4.4 on page 43 (ALIAS) for full details on aliases.

4.79.2 AUTOSWITCH

```
Syntax: SET AUTOSWITCH ON|OFF           (default = ON)
```

Enables automatic switching between Tektronix plotting and alphanumeric mode. If your screen clears when you switch modes, you should set this option off. Also if you are running Versaterm with 2 windows on top of one another, this option should probably be OFF.

4.79.3 AXIS

```
Syntax: SET X|Y AXIS
        text
```

```
          x  y  size  angle  option font
      or  SET AXIS X|Y text...
      or  SET AXIS ALL|BOTTOM|TOP|LEFT|RIGHT|VERTICAL ON|OFF|PLOT|PAGE
  where:  text    is the axis label
          x,y     are its position relative to the centre of the axis
          size    is the size of the text
          angle   is the angle with respect to the horizontal
          option  can be LEFT   = left adjusted to x,y
                         CENTRE = centred (default)
                         RIGHT  = right adjusted
          font    is the font to use for the text
```

Alias for SET LABEL. See section 4.79.31 on page 100 (SET LABEL) for more details.

4.79.4 BACKGROUND

```
Syntax: SET BACKGROUND nfun1 [nfun2...]
```

Specifies which of the functions you have defined should be considered as background when you do a background subtraction. nfun1 = 0 means that they are all background. When you add a new function, it is defined to be signal by default.

4.79.5 BOX

```
Syntax:  SET BOX ON|OFF
Default: OFF
```

Turns on or off drawing a box around the plot.

4.79.6 COLOUR

```
Syntax: SET COLOUR [item] colour
    or: SET COLOUR REPRESENTATION index red green blue
where:  item    is the name of what you want to change
        colour  is a number corresponding to the colour you want
```

Changes the colour of the lines on a picture (only works in HIGZ version). To change the colour a particular plot, precede the command with SET PLOT id [&idb].

The colour map is that of HIGZ:

```
0 = Background (White)        4 = Dark blue
1 = Foreground (Black)        5 = Yellow
2 = Red                       6 = Magenta
3 = Green                     7 = Cyan
```

If you give omit the item, the colour of all items will be changed. The following items are valid:

```
FRAME                TICK                 SCALE
LABEL                HEADER               TITLE
SYMBOL               ZERO_LINE            COMMENT
REPRESENTATION
```

The command `SET COLOUR REPRESENTATION` **`index red green blue`** changes the colour representation for that index. This enables you to customize your colour indices if you wish. **`index`** is colour index, **`red, green, blue`** are the fractions of red, green and blue for that index. They must be between 0 and 1.

4.79.7 DEBUG

```
Syntax:   SET DEBUG ON|OFF level
where:    level  is a number for the debug level
Defaults: OFF, level=0
```

Turns on or off a logical `QDEBUG` and sets the debug level. These variables are available in COMIS functions in the common block:

```
LOGICAL QDEBUG
COMMON/MNDBG/QDEBUG,NDEBUG
```

This is very useful for debugging COMIS functions, meaning you can interactively change the amount of printout you get. It can equally well be used in user functions.

Print levels >100 are reserved for future Mn_Fit debug output. Although, not used at the moment I can imagine that it might be useful in the future.

4.79.8 DISPLAY

```
Syntax: SET DISPLAY MODE|X|Y|Z [parameter]
```

You can give this command to change parameters in the `DISPLAY` of the fit results or to change what is displayed. If you give a `X Y` or `Z` or hit `<CR>` after the `SET DISPLAY` command then any subsequent parameters you change will apply to the `DISPLAY` only.

The command `SET DISPLAY MODE` changes the display form for the fit results.

MODE

```
Syntax: SET DISPLAY MODE nmode
 where: nmode = 1  means show the fit and the function
              = 2  means show the background subtracted fit
              = 3  means show both
              -ve  means divide by the background also before
                   making the background subtracted plot
```

You can show the standard plot, background subtracted, or both. For the background subtracted plot, you can do a simple subtraction or divide by the background also. You must specify which functions are the background using the `SET BACKGROUND` command.

4.79.9 DEFAULT

```
Syntax: SET DEFAULT
    or  SET PLOT id [&idb] DEFAULT
```

If you do not precede the command with PLOT, all plot parameters are set back to their default values. If you precede command with PLOT, it is equivalent to SET HIST (see section 4.79.26 on page 100 (SET HISTOGRAM) for more details), and fills registers 121-199 with information on the histogram and defines the default plot for the DEPOSIT command.

4.79.10 DIRECTORY

```
Syntax: SET DIRECTORY dirname
where   dirname  is the directory name
```

Sets the current HBOOK directory name to **dirname**. Note that the HCDIR command is only executed when a FETCH, LDIR, ZDIR or 'SHOW DIR' command is executed. This enables you to SET the directory you want to fetch from, and the secondary identifier you want to give the plots, before fetching them. However, this means that successive CDIR commands without a FETCH etc. in between will only remember the last name.

Note that the directory name should not be preceded by a /. However // is allowed so you can go to the top level.

Note that the Mn_Fit HBOOK top level file directory name is //MN_HBIN.

4.79.11 DSIZE

```
Syntax: SET DSIZE size                      (default = 1.0)
```

Control the scale factor for the dot size. This command only works in the HIGZ version of Mn_Fit.

4.79.12 DUMP

```
Syntax: SET DUMP TTY|LPT|CLOSE
    or  SET DUMP FILE filename
```

Change the output unit for the DISPLAY, DUMP, HIST DUMP, INDEX and MESSAGE commands. If you specify LPT, the output will be written to the file **mn_dump.dat**. To close the file, specify CLOSE.

To change the filename from **mn_dump.dat**, use the command SET DUMP FILE **filename**. You then still have to give the command SET DUMP LPT to change the output routing to that file.

4.79.13 ECHO

```
Syntax: SET ECHO ON|OFF                (default = ON)
```

Turns on or off echoing of commands read from a file.

4.79.14 EDIT

```
Syntax: SET EDIT command                (default = EDIT/TPU)
```

Changes the command used to invoke the editor. This command only applies to the Vax.

4.79.15 ENDSET

```
Syntax: ENDSET
```

Gets you back to the `MN_CMD>` or `MINUIT>` level. You must use `ENDSET` in command files to exit `SET` if you did not give the parameter on the same line as the `SET` command.

4.79.16 ERR_ZERO

```
Syntax: SET ERR_ZERO ON|OFF               (default = ON)
```

Sets the error on zero points to 1 when a plot is fit or one of the commands `ADD, SUBTRACT, MULTIPLY, DIVIDE, SCALE, NORMALIZE` is applied to the plot.

4.79.17 EXCLUSIONS

```
Syntax: SET EXCLUSIONS ON|OFF             (default = ON)
```

Turns on or off showing the excluded parts of plots when drawing in a function.

4.79.18 EXIT

```
Syntax: EXIT
```

Gets you back to the `MN_CMD>` or `MINUIT>` level. Use `ENDSET` rather than `EXIT` to avoid confusion and accidental exiting from Mn_Fit.

4.79.19 FIT

```
Syntax: SET FIT INTEGRATE
```

Set options for fitting. At the moment the only option available is whether to integrate over the bins or not.

INTEGRATE

```
Syntax: SET FIT INTEGRATE ON|OFF [ninterval]
where   ninterval  is the number of intervals in the integration.
```

Specifies whether the function is integrated over each bin of the plot you are fitting. The integration uses Simpson's approximation and the number of intervals must be even.

4.79.20 FONT

```
Syntax:   SET FONT [item] nfont
where:    item    is the name of what you want to change
          nfont  is the font number in the form spfff
                 s    is the sign of the font (+|-)
                 p    is the precision
                 fff  is the font number
Defaults: nfont = 0
```

Sets the font used for text on a picture. To change the font of an item in a particular plot, precede the command with `SET PLOT id [&idb]`. If you give omit the item, the font of all items will be changed. The following items are valid:

```
HEADER              TITLE               COMMENT
SCALE               LABEL
```

Font 0 uses the IGTEXT routine (see section 1.12 on page 24 (FONTS) for details). The other fonts available depend on the workstation type. HIGZ also has a number of device independent fonts. See the HIGZ manual for more details. In addition all the other fonts listed in the PAW manual (section 8.10 - text fonts) are available. e.g. `SET FONT -2013` will choose font -13 with precision 2.

Note that if you change the font for the `SCALE` or `LABEL` it will be changed for all axes. To change it for a particular axis use the `SET X|Y|Z SCALE` or `SET X|Y|Z LABEL` commands. If the `SET SCALE` or `SET LABEL` commands are not for a particular plot then the font will be set to that specified in the `SET FONT` command.

4.79.21 FRAME

```
Syntax:   SET FRAME ALL|BOTTOM|TOP|LEFT|RIGHT|VERTICAL ON|OFF
Defaults: ALL ON
```

Sets up where you want a frame around the plot. Default is to draw a frame all around it. If you want open axes (i.e. only the x and y axes drawn), give the commands `SET FRAME TOP OFF RIGHT OFF`. `VERTICAL` is the z-axis in `LEGO` and `SURFACE` plots.

To set or change the frame for a particular plot, precede the command with `SET PLOT id [&idb]`.

4.79.22 GRID

```
Syntax:   SET X|Y GRID ON|OFF [option nsymb]
 where:   option  use which ticks to draw the grid on (BIG or ALL)
          nsymb   is the symbol number for the grid
Defaults: OFF BIG 3     except for z-axis: ON BIG 3
```

Turns on or off drawing a grid on the plot and specifies which ticks to put the grid on (ALL or BIG) and the symbol to use for it. The default is that the grid will be drawn at the big tick positions. The grid in each direction can be turned on or off independently. Turning off the frame means that no grid will be drawn.

To change the grid for a particular plot, precede the command with `SET PLOT id [&idb]`.

4.79.23 HARDCOPY

```
Syntax:   SET HARDCOPY filename
where     filename  is the name of the file you want hardcopy output in
Defaults: PLOT.PS, PLOT.META etc.
```

Changes the name of the file written to by the HARDCOPY command or if you CAPTURE a hardcopy device.

4.79.24 HATCH

```
Syntax: SET HATCH nhatch
 where: nhatch  is the hatch number
```

Sets the hatch number to use for plots.

WARNING: This command only works with the HIGZ version of Mn_Fit, and the hatchings available are device dependent. For the Vaxstation and Postscript, numbers -1 to -11 are available; the Vaxstation has more, but they will be plotted as -1 if you make a hardcopy. GKSGRAL has a number of device independent hatchings numbers -101 to -124. See the HIGZ or CERN graphics manuals for more details. For all versions using HIGZ, the HIGZ portable hatches are available. These use an index coded using 3 digits ijk:

```
i: Distance between each hatch in mm;
j: Angle between 90 and 180 degrees;
k: Angle between 0 and 90 degrees.
```

The numbers are coded according to the table below:

i		j		k	
		0	180 degrees	0	0 degrees
1	0.75mm	1	170 degrees	1	10 degrees
2	1.50mm	2	160 degrees	2	20 degrees
3	2.25mm	3	150 degrees	3	30 degrees
4	3.00mm	4	135 degrees	4	45 degrees
5	3.75mm	5	Not drawn	5	Not drawn
6	4.50mm	6	120 degrees	6	60 degrees
7	5.25mm	7	110 degrees	7	70 degrees
8	6.00mm	8	100 degrees	8	80 degrees
9	6.75mm	9	90 degrees	9	90 degrees

Note that the space between hatch lines is visible. If you overlay a plot with a hatch, then the one below it can be seen. The area under a pattern is not visible.

4.79.25 HEADER

```
Syntax: SET HEADER ON|OFF|FULL|BRIEF       (default = FULL)
```

Controls whether to print the histogram numbers being plotted on the picture. The OFF mode turns off the text, BRIEF shows the identifiers and the symbols, FULL also shows the area, mean and sigma for the plot. Alias for SET IDSHOW.

4.79.26 HISTOGRAM

```
Syntax: SET HISTOGRAM id [&idb]
 where: id  is the histogram identifier
        idb is the (optional) secondary identifier
```

Defines a default histogram for the DEPOSIT command, and fills registers 121 - 199 with information on the histogram. See section 1.6 on page 17 (Numbers) for a list of the variables stored.

4.79.27 HSIZE

```
Syntax: SET X|Y HSIZE size                    (default = 15.0/15.0 cm)
```

Alias for SET SIZE. See section 4.79.54 on page 106 (SET SIZE) for more details.

4.79.28 IDB

```
Syntax: SET IDB n          (default = 0)
```

Changes the default secondary identifier. All histograms subsequently fetched will have this secondary identifier. In addition if you omit the secondary identifier anytime you are asked for a plot number, the default will be used. Alias for SET SECONDARY_ID.

4.79.29 IDSHOW

```
Syntax: SET IDSHOW ON|OFF|FULL|BRIEF        (default = FULL)
```

Controls whether to print the histogram numbers being plotted on the picture. The OFF mode turns off the text, BRIEF shows the identifiers and the symbols, FULL also shows the area, mean and sigma for the plot. Alias for SET HEADER.

4.79.30 IDSIZE

```
Syntax: SET IDSIZE size                       (default = 0.3cm)
```

Controls the size of the histogram numbers if the IDSHOW option is turned on.

4.79.31 LABEL

```
Syntax: SET X|Y LABEL
        text
        x  y  size  angle  option font
    or  SET LABEL X|Y text...
    or  SET LABEL ALL|BOTTOM|TOP|LEFT|RIGHT|VERTICAL ON|OFF|PLOT|PAGE
 where: text    is the axis label
        x,y     are its position relative to the centre of the axis
        size    is the size of the text
        angle   is the angle with respect to the horizontal
```

```
option  can be LEFT   = left adjusted to x,y
               CENTRE = centred (default)
               RIGHT  = right adjusted
font    is the font to use for the text
```

Changes or adds a label to an axis. You will be prompted for the text and the position of the label. If you want to specify on which axes the labels should be written, use the `SET LABEL ALL|BOTTOM|TOP` etc. command. `VERTICAL` is the z-axis in `LEGO` and `SURFACE` plots. If you do not give a font, it will be that chosen with the `SET FONT` command (default = 0). To set or change the label for a particular plot, precede the command with `SET PLOT id [&idb]`.

The mode `PLOT` will put a label on each plot, whereas `PAGE` will put one label on each page, centering it with respect to the overall plot size.

4.79.32 LIMITS

```
Syntax: SET X|Y|Z LIMIT low high                (default =  0 0)
 where: low  is the lower limit on the axis scale
   and  high is the upper limit on the axis scale
```

Specifies the lower and upper limits for the scale. To reset the limits back to default, use the command `SET X|Y|Z LIMIT 0 0`. To change the limits for a particular plot, precede the command with `SET PLOT id [&idb]`.

4.79.33 LOG

```
Syntax:   SET LOG ON|OFF
Default:  OFF
```

Turns on or off writing all commands typed in at the terminal to a log file. The file is called `mn_fit.log`.

4.79.34 MARGIN

```
Syntax: SET X|Y MARGIN size                     (default = 3.0/2.0 cm)
```

Sets the offset of the plot from the bottom left-hand corner of the picture.

4.79.35 MODE

```
Syntax: SET X|Y MODE REAL|INTEGER|LOG           (default = REAL)
```

Changes the way the scale is drawn on the x or y axis. The following modes are available:

```
REAL    = real numbers
INTEGER = integer numbers
LOG     = log scale. You will be prompted for the lower limit and the
          number of decades you want to plot.
```

To change the mode for a particular plot, precede the command with `SET PLOT id [&idb]`.

4.79.36 NEXT_WINDOW

```
Syntax: SET NEXT_WINDOW nx ny
 where: nx is the window number in the x direction
        ny is the window number in the y direction
```

Changes the window number for the next plot you make. If you select 1 1 the screen will be cleared before the plot is drawn. This can be overridden by using the PLOT/NOCLEAR command.

4.79.37 NORMALIZE

```
Syntax: SET NORMALIZE ON|OFF
```

Only valid in MN_CMD>.

Turns on or off an overall normalization factor for the function(s) you are fitting with. The parameter will be called NORM00 and will be the first parameter. This is useful if you want to fit to functions of the form:

```
NORM00 * (HIST1 + ALPHA*HIST2)
```

where HIST1 and HIST2 are 2 histograms and ALPHA is the relative contribution of each.

4.79.38 NULL

```
Syntax: SET X|Y NULL ON|OFF nsymb            (default = ON)
 where: nsymb  is the symbol number for the line (default = -1)
```

Alias for SET ZERO. See section 4.79.70 on page 112 (SET ZERO) for more details.

4.79.39 OPT_ZERO

```
Syntax: SET X|Y|Z OPT_ZERO ON|OFF           (default = ON)
```

Turns on or off having 0 as the lower limit for plotting the number of entries in a plot. For 1-dimensional histograms use SET Y OPT_ZERO and SET Z OPT_ZERO for lego and surface plots of 2-dimensional histograms.

4.79.40 ORDER

```
Syntax: SET ORDER X|Y|DX|DY|DNX|DNY|DPX|DPY|DUMMY|DATE...
```

Specifies the order of the variables in a file to be read in with the DAT_FETCH or READ DATA commands. The name DUMMY can be used if a variable is not to be read in. The default order is X Y DNX DNY DPX DPY. N signifies the negative error and P the positive error if you have asymmetric error bars.

If a variable is time, the following names and formats are supported:

DATE_TIME	YYMMDD HHMMSS
DATE	YYMMDD
TIME	HHMMSS
DATE_MIN	YYMMDD HHMM
TIME_MIN	HHMM
VAXTIME	Char*23 Vaxtime DD-MMM-YYYY HH:MM:SS.SS

The old variable names of DXN, DYN, DXP and DYP are still recognized, but are not recommended.

Examples

1. If I have a series of numbers:

   ```
   1.0 3.0 0.8
   ```

   which are X,Y,DY the command is:

   ```
   SET ORDER X,Y,DY
   ```

2. If I have a series of numbers:

   ```
   1.0 5.0 8.0 10.0 0.3 0.5 1.0 1.2
   ```

   where the first 2 are x and y and their asymmetric errors are in variables 5,6,7,8 the command is:

   ```
   SET ORDER X Y DUMMY DUMMY DXN DXN DYP DYP
   ```

4.79.41 ORTHOGONAL

```
Syntax: SET ORTHOGONAL xlo xhi
 where: xlo  is the lower orthogonality limit
        xhi  is the upper limit
```

Sets the orthogonality limits for Chebyshevs and Legendres. If the limits are not set, they will be taken as the lower and upper limits on the plot(s) you are fitting. To restore the automatic setting of limits, SET ORTHOGONAL 0 0.

4.79.42 PARAMETER

```
Syntax: SET PARAMETER detnam values
 where: detnam  is the detector name (ECAL, FBGO or FWCH)
        values  are the new values (see subtopics for their meaning)
```

Sets parameters for the L3 detector displays. Use the syntax SET PARAMETER detnam followed by <CR> to see the current values.

As usual to keep the current value of a parameter give an =.

4.79.43 PATTERN

```
Syntax: SET PATTERN npattern
 where: npattern  is the pattern number
```

Set the pattern number to use for plots.

Warning: This command only works with the HIGZ version of Mn_Fit, and the patterns available are device dependent. The pattern is drawn every 250 points of the area you are filling. This means that if you try to make a pattern under a function, you will get a vertical line every 250 points. This seems to be built into at least the GKSGRAL version of GKS, and I do not know how to avoid it.

Pattern 100 means solid fill and pattern 200 means hollow fill. For the Vaxstation and Postscript, numbers 1 to 99 are also available. If you try to set any other value, it will be reset to 1, as DECGKS blows up for an invalid pattern number. Note that the current version of DECGKS gives different patterns for the same pattern number on the Vaxstation and in a Postscript file.

Note that the area under a pattern is not visible. The space between hatch lines is visible, so if you overlay a plot with a hatch, then the plot below can be seen.

4.79.44 PLOT

```
Syntax: SET PLOT id [&idb]
 where: id  is the plot identifier
        idb is the (optional) secondary identifier
```

Specify which plot the following `SET` command applies to. If you give `id=0`, the command applies to all plots currently being displayed and also to subsequent plots; otherwise, it only applies to the plot you specified. If you want to change something on a plot you have already made, you must use this form and then issue the `REDRAW` command after you have made your changes to see the effect of them.

4.79.45 PSIZE

```
Syntax: SET X|Y PSIZE size                (default = 19.5/20.5 cm)
```

Sets the overall size of the picture, i.e. histogram size plus margins each side of the histogram.

4.79.46 RATIO

```
Syntax: SET RATIO ON|OFF                  (default = ON)
```

Only valid in `MN_CMD>`.

Turns on or off using the ratio of the areas under each function when fitting more than 1 plot. With the option `OFF`, the area of the function under each plot is used as a fitting parameter. With the option `ON` the first parameter is the total area, the second is the fraction of the total area under the first plot, the third is the fraction of the total area under the second plot etc. For the last plot being fit the area used is what is left over from the other plots.

4.79.47 RECL

```
Syntax: SET RECL n
```

Sets the record length for HBOOK version 4 direct access files. The default is 1024 (Vax) and 4096 (Apollo). Changing this affects both the FETCH and STORE commands.

4.79.48 REDRAW

```
Syntax:  REDRAW
```

Alias for REDRAW so that you can REDRAW without exiting SET. See section 4.74 on page 91 (REDRAW) for more details.

4.79.49 SCALE

```
Syntax:   SET X|Y SCALE
          x, y, size, angle, ndigit, ndecimal, factor, font
    or    SET SCALE X|Y x, y, ...
    or    SET SCALE ALL|BOTTOM|TOP|LEFT|RIGHT|VERTICAL ON|OFF
 where:   x,y      are the position of the scale relative to the
                   point where the tick crosses the axis
          size     is the size of the scale numbers
          angle    is the angle with respect to the horizontal
          ndigit   is the maximum number of digits before switching
                   to exponential mode (e.g. 1.0*10-4)
          ndecimal is the number of decimal places to display
                   -1 means the program will set ndecimal
          factor   is a multiplication factor the scale
          font     is the font to use for the scale
Defaults: x axis  0.0, -0.6, 0.4, 0.0, 5, -1, 1.0, 0
          y axis -0.2,  0.0, 0.4, 0.0, 6, -1, 1.0, 0
          z axis -0.2,  0.0, 0.4, 0.0, 6, -1, 1.0, 0
```

Changes the position, size or way in which the scale is shown, or sets up where you want the scale put. VERTICAL is the z-axis in LEGO and SURFACE plots. The default is just to put it on the bottom, left and vertical axes. For an overlayed plot with a different scale, the scale will be put on the right axis. To change the scale for a particular plot, precede the command with SET PLOT id [&idb].

4.79.50 SECONDARY_ID

```
Syntax: SET SECONDARY_ID n           (default = 0)
```

Change the default secondary identifier. All histograms subsequently fetched will have this secondary identifier. In addition if you omit the secondary identifier anytime you are asked for a plot number, the default will be used. Alias for SET IDB.

4.79.51 SHOW_ZERO

```
Syntax: SET SHOW_ZERO ON|OFF        (default = ON)
```

Controls showing zero points which also have zero errors.

4.79.52 SHELL

```
Syntax:   SET SHELL command
where:    command  is the shell command
Defaults: command = /bin/csh
```

This option only applies to the Apollo. Possible values for **command** are **/com/sh, /bin/sh, /bin/ksh, /bin/csh**. The shell given by the command will be invoked when **SHELL** or **SPAWN** is given.

Note that **SHELL** or **SPAWN** without a command seems to work fine for all shells **/com/sh, /bin/sh, /bin/ksh and /bin/csh**. However only the Aegis shell **/com/sh** works with a command at present. For the Unix shells you must give CTRL/D to exit and not **return**.

4.79.53 SIGNAL

```
Syntax: SET SIGNAL nfun1 [nfun2...]
```

Specifies which of the functions you have defined should be considered as signal when you do a background subtraction. **nfun1** = 0 means that they are all signal. When you add a new function, it is defined to be signal by default. If you have not given the **SET BACKGROUND** command or you specify that all the functions are signal and then give a **BACK_SUB** command or set the **DISPLAY** mode to 2, 3, -2 or -3, you will be prompted for which of the functions are background.

4.79.54 SIZE

```
Syntax: SET X|Y SIZE size                (default = 15.0/15.0 cm)
```

Alias for **SET HSIZE**. Sets the size of the plot in the x or y direction. To change the size for a particular plot, precede the command with **SET PLOT id [&idb]**.

4.79.55 SSIZE

```
Syntax: SET SSIZE size                   (default = 0.3cm)
```

Sets the size of the symbols in the picture. To change the size for a particular plot, precede the command with **SET PLOT id [&idb]**.

4.79.56 STATISTICS

```
Syntax:   SET STATISTICS MN_FIT|HBOOK
Default:  MN_FIT
```

Specifies whether Mn_Fit or HBOOK statistics will be used to calculate the means and sigmas of histograms. The option only applies to 1-dim HBOOK histograms. This option is useful if you calculate the HBOOK mean and sigma from the HFILL calls using the option `CALL HIDOPT(ID,`'STAT")'.

4.79.57 SYMBOL

```
Syntax:   SET SYMBOL n
where:    n  is the symbol number
Default:  n = 0
```

For each type of plot a default symbol is defined (which you get if you give the command `SET SYMBOL 0`):

```
Symbol  1 for 1-d histograms
       -1 for a series of points without errors
      -32 for a series of points with errors
       12 for 2-d histograms
        1 for scatter plots
```

To get a picture of the available symbols issue the command `EXEC MN_FIT_HELP:SYMBOL.MNF` (Vax) or `exec ~/mn_fit_help/symbol.mnf` (Apollo). This picture is in Appendix B of the Mn_Fit manual.

In HIGZ/GKS versions hatching and patterns are available. You can specify the type using the commands `SET HATCH` and `SET PATTERN`. See section 4.79.24 on page 99 (SET HATCH) for details on some of the hatching available and also the figures in Appendix B. Also see the HIGZ/PAW documentation for HIGZ hatchings, and the GKS device documentation. As far as I know patterns are only available with DECGKS.

The following symbols are available for histograms:

-1	Solid line joining the centres of the bins
-2	Dashed line joining the centres of the bins
-3	Dotted line joining the centres of the bins
-4	Dash-dot line joining the centres of the bins
-5	HIGZ line style 12 - a dashed line
-6	HIGZ line style 13 - a dash-dot line
-7	HIGZ line style 14 - a widely spaced dotted line
-8	HIGZ line style 15 - a dotted line
1	Solid line histogram mode
2	Dashed line histogram mode
3	Dotted line histogram mode
4	Dash-dot line histogram mode
5	HIGZ line style 12 histogram mode

6	HIGZ line style 13 histogram mode
7	HIGZ line style 14 histogram mode
8	HIGZ line style 15 histogram mode
10	Dot
11	Circle (really an octagon!)
12	Square
13	Triangle
14	Inverted triangle
15	Diamond
16	Plus (+)
17	Cross (x)
18	Asterix (*)
20-28	Show x error bars for histograms
30-38	Show y error bars
40-48	Show x and y error bars
-n	Show symbol filled

The following symbols are available for scatter plots:

-1	Joins the points with a solid line
-2	Joins the points with a dashed line
-3	Joins the points with a dotted line
-4	Joins the points with a dash-dot line
-5	HIGZ line style 12 - a dashed line
-6	HIGZ line style 13 - a dash-dot line
-7	HIGZ line style 14 - a widely spaced dotted line
-8	HIGZ line style 15 - a dotted line
1-10	One dot per point
10-18	As for histograms
20-48	As for 10-18
-n	Show symbol filled

The following symbols are available for 2-dimensional histograms:

-1	.,1,2,3,...X,Y,Z
-2	Number of entries i.e. table form
1-10	Randomized dots, where the number of dots is equal to the number of entries
10-18	Area of symbol is proportional to number of entries in the bin for scatter plots
20-48	As for 10-18
-n	Show symbol filled

To change the symbol for a particular plot, precede the command with `SET PLOT id [&idb]`.

4.79.58 TEXT

```
Syntax: SET TEXT ON|OFF                (default = ON)
```

Controls whether the header text with the fit results is shown on your softcopy device. Use the command SET HEADER if you want to change it for hardcopies also.

4.79.59 THICKNESS

```
Syntax: SET THICKNESS [item] thick                    (default = 1.0)
 where: thick  is the scale factor to increase the line width by
```

Changes the thickness of the lines on a picture (only works in GKS version) To change the pen width for a particular plot, precede the command with SET PLOT id [&idb].

If you give omit the item, all line thicknesses will be changed. The following items are valid:

```
FRAME               TICK                SCALE
LABEL               HEADER              TITLE
SYMBOL              ZERO_LINE           COMMENT
```

4.79.60 TICKS

```
Syntax: SET X|Y TICK
        ntick, nbtick, notick, size, bigsize, xtlo, xthi
    or  SET TICK X|Y ntick, nbtick, ...
    or  SET TICK ALL|BOTTOM|TOP|LEFT|RIGHT|VERTICAL [INSIDE|OUTSIDE] ON|OFF
 where: ntick   is the number of ticks
        nbtick  is the number of ticks per big tick
        notick  is the offset of the first big tick from the
                first tick
        size    is the tick size (default=0.25cm)
        bigsize is the big tick size (default=0.5cm)
        xtlo    is the lower limit for drawing ticks
        xthi    is the upper limit for drawing ticks
```

Specifies exactly how many, what size and where the ticks should start or stop, or on which axes and which side of the axes you want the ticks. To set back to automatic tick calculation and limits, give the command SET TICK 0. Unless you specify xtlo and xthi, they will be calculated automatically.

To specify where you want the ticks, use the SET ALL|BOTTOM|TOP form. If you omit the INSIDE|OUTSIDE option, the specification will apply to both sides. VERTICAL is the z-axis in LEGO and SURFACE plots. The default is to put them on the inside of all 4 sides of the plot. For the vertical axis, they are put on the outside. To set or change the ticks for a particular plot, precede the command with SET PLOT id [&idb].

Examples

1. This series of commands will give you ticks on the lower and left axes, with the ticks going through the axis and ticks on the top axis:

```
SET TICK ALL OFF
SET TICK BOTTOM ON
SET TICK LEFT   ON
SET TICK TOP INSIDE ON
```

4.79.61 TIME

```
Syntax: SET TIME mode [reference]
 where: mode       is the units in which to store the data
        reference  is the reference time (T=0)
```

Sets the mode for storing data vs. time and the reference time to be used. The mode can be DAY (default), HOUR, MINUTE, or SECOND. The reference time must be given in the form YYMMDD.HHMMSS.

4.79.62 TITLE

```
Syntax: SET TITLE USER title
    or  SET TITLE DEFAULT|ON|OFF
    or  SET TITLE POSITION [xoff yoff size angle option font]
 where: title   is an overall title
        xoff    is the x position of the title
        yoff    is the y position of the title
        size    is the character size for the title
        angle   is the angle of the title
        option  can be LEFT, CENTRE, or RIGHT adjusted
        font    is the text font to use
default is 0.0 0.4 0.4 0.0 CENTRE 0
```

Controls whether to display the plot title or not, or whether you want to give a user title to all plots. A user title is put at the top of the picture by default and if it is shown the individual plot titles are not. SET TITLE POSITION controls the position of title, which is relative to the top centre of the plot, and the font used for it. To change whether the title is drawn or its position for a particular plot, precede the command with SET PLOT id [&idb].

4.79.63 TSIZE

```
Syntax: SET TSIZE value                    (default = 0.4cm)
```

Changes the size of the title. To change the title size for a particular plot, precede the command with SET PLOT id [&idb]. You can also use the SET TITLE POSITION command to do this.

4.79.64 USIZE

```
Syntax: SET USIZE value                    (default = 0.3cm)
```

Controls the size of the smaller text in the fit display. To change the size for a particular plot, precede the command with SET PLOT id [&idb].

4.79.65 WAIT_CR

```
Syntax: SET WAIT_CR ON|OFF                    (default =  ON)
```

Turns on/off the request for a <CR> before making the next picture. This command is most useful if you send the output to a file (Postscript) directly without putting it on the screen.

4.79.66 WINDOW

```
Syntax: SET WINDOW nwindx nwindy
        wx, wy
    or  SET PLOT id [&idb] WINDOW nx ny
 where: nwindx  are the number of windows in the x direction
        nwindy  are the number of windows in the y direction
        wx      is the separation of the windows in the x direction
        wy      is the separation of the windows in the y direction
        nx      is the window number in the x direction
        ny      is the window number in the y direction
```

This command enables you to put more than one plot on each page. The plot (overall size specified with **SET X|Y HSIZE**) is divided up into **nwindx** by **nwindy** windows with separations **wx, wy** between them. All the plots will be drawn with the same size on the page. To change the size or position of a plot in a window, use the **SET PLOT id [&idb] WMARGIN** or **WSIZE** commands.

If the tick, title and scale sizes are default then they will be automatically rescaled if you use more than one window.

If you give 0 separation, the scale for plots other than the leftmost or bottom one will be suppressed. You can turn it back on again after you have made a picture using the syntax **SET PLOT id SCALE LEFT ON** etc. In addition, if the last big tick corresponds to the right or top of the plot (and it is not the rightmost or top plot), the number for the scale will not be drawn.

If you want to change the window number of a plot you have already made, use the **SET PLOT id [&idb] WINDOW** command. To set the window number for a plot you are about to make, use the **SET NEXT_WINDOW** command.

Examples

1. This example shows how to manipulate windows and move plots around within a window:

```
SET WINDOW  2 2 2.0 0     !Split the plot into 4 windows with no
                          !separation in the y direction
PLOT 1
PLOT 2
SET WINDOW  1 2 = =       !Change to 2 windows so the bottom plot
                          !will go across the full page
PLOT 3
SET PLOT 2 X WMARG 1.0    !Move plot 2 1cm over
SET PLOT 2 Y WSIZE 5.0    !Make plot 2 5cm wide in x.
```

```
                                   !Its original size was
                                   !(15(HSIZE) - 2.0(WX)) / 2(NWINDX) = 6.5cm
         REDRAW                    !Look at the new picture
```

4.79.67 NO_WINDOW

```
Syntax: SET NO_WINDOW
```

Turns off any windowing.

4.79.68 WMARGIN

```
Syntax: SET X|Y WMARGIN size          (default = 0.0/0.0 cm)
```

Sets the offset of the histogram from the bottom left-hand corner of the window. To change the size for a particular plot, precede the command with `SET PLOT id [&idb]`. Normally the margin is set to 0 inside the window. Issue this command after the window command.

4.79.69 WSIZE

```
Syntax: SET X|Y WSIZE size
```

Changes the size of the plot inside a window in the x or y direction. To change the size for a particular plot, precede the command with `SET PLOT id [&idb]`. Normally the size is calculated automatically from the number of windows. Issue this command after the window command.

4.79.70 ZERO

```
Syntax: SET X|Y ZERO ON|OFF nsymb       (default = ON)
 where: nsymb  is the symbol number for the line
```

Turns on or off drawing a line at x=0 or y=0 and specifies the symbol for the line. To change the line for a particular plot, precede the command with `SET PLOT id [&idb]`.

4.80 SHELL

```
Syntax: SHELL [command]
```

Alias for `SPAWN`. Spawns a DCL (Vax) or Shell (Apollo) command. You can specify the shell on the Apollo using the `SET SHELL` command. If no command is given a new process is created. Use the command `Return` to return to Mn_Fit when you have finished in the subprocess. On the Vax you can also `ATTACH` to the main process and then use `ATTACH` in Mn_Fit to re-enter the subprocess if you wish.

Note that `SHELL` or `SPAWN` without a command seems to work fine for all shells `/com/sh, /bin/sh, /bin/ksh and /bin/csh`. However only the Aegis shell `/com/sh` works with a command at present. For the Unix shells you must give CTRL/D to exit and not `return`.

4.81 SHOW

```
Syntax:  SHOW name|ALL
    or   SHOW PLOT id [&idb] name|ALL
```

The SHOW PLOT command enables you to get detailed information on the parameters being used in making the plots currently in the buffer. You can change these parameters with the SET and SET PLOT id [&idb] commands.

SHOW is being expanded, so that you can give a SHOW command for every SET command. Therefore only those SHOW commands that are in addition to the SET commands are listed. If the SHOW command does not give you what you want, you can usually get a list of the current values of parameters for a particular SET command by using the syntax SET command <CR>.

4.81.1 ALL

```
Syntax: SHOW ALL
```

Lists everything there is to show.

4.81.2 ALIAS

```
Syntax: SHOW ALIAS [name]
where:  name  is the alias name
```

Lists the definition of alias name or all defined aliases if you omit name. See section 4.4 on page 43 (ALIAS) for more information on use of aliases.

4.81.3 COMMANDS

```
Syntax: SHOW COMMANDS
```

Lists the commands defined using the DEFINE command. Alias for SHOW DEFINITION.

4.81.4 COMMENT

```
Syntax: SHOW COMMENT
```

Lists the current comments associated with plots.

4.81.5 CONSTRAINT

```
Syntax: SHOW CONSTRAINT
```

Lists the current constraints on parameters that are being fit. See section 5.3 on page 120 (MINUIT CONSTRAIN) for more details on how to constrain parameters.

4.81.6 CUTS

```
Syntax: SHOW CUT
```

Lists the cuts set for a plot.

4.81.7 DEFINITION

Syntax: `SHOW DEFINITION`

Lists the commands defined using the `DEFINE` command. Alias for `SHOW COMMANDS`.

4.81.8 DIRECTORY

Syntax: `SHOW DIRECTORY.`

Alias for `LDIRECTORY`. Lists the contents of the currently selected HBOOK directory in an HBOOK4 file.

4.81.9 EXCLUSIONS

Syntax: `SHOW EXCLUSIONS`

Lists the parts of histograms excluded from the fit.

4.81.10 FILES

Syntax: `SHOW FILES`

Lists the current filenames for `HARDCOPY` and `DUMP`. Lists the units and the filenames associated with them for units currently open.

4.81.11 FLAGS

Syntax: `SHOW FLAGS`

Lists the logical flags that control what is plotted.

4.81.12 KEYS

Syntax: `SHOW KEYS`

Lists the current keys associated with plots.

4.81.13 LOG

Syntax: `SHOW LOG [n]`

Lists the last `n` commands given. If `n` is not given the last 100 commands are shown.

4.81.14 ORDER

Syntax: `SHOW ORDER`

Lists the expected order of variables when executing `DAT_FETCH`.

4.81.15 PLOT

```
Syntax: SHOW PLOT
        [id [&idb]]
```

Lists which plots are currently in the buffer for plotting and also enables you to get more information on the parameters being used for each plot by preceding what you want to show with the plot identifier (e.g. SHOW PLOT id SCALE).

4.81.16 REGISTER

```
Syntax: SHOW REGISTER r1:[r2]|ALL
where:  r1  is a register number
        r2  is the upper limit of a range of registers to show
```

Lists the contents of one or more registers. If you give the command SHOW REGISTER ALL all user registers will be listed (0-99). You can also list a range of registers e.g. SHOW REGISTER 301:320. The DEPOSIT and CALCULATE commands can be used to set the contents of registers 0-99. For example, DEP R1 = 2.0*P2(1) will set the contents of register 1 to twice the value of parameter 2 in function 1. You can also use these commands to define user variables which are stored in registers >300 (see section 4.27 on page 56 (DEPOSIT) for more details).

4.81.17 SEGMENTS

```
Syntax: SHOW SEGMENTS
```

Lists the current segments in use. Only applies to the GKS version and is mainly for expert use.

4.81.18 SIZES

```
Syntax: SHOW SIZES
```

Lists the parameters relating to plot sizes, margins, windows, etc.

4.81.19 VARIABLE

```
Syntax: SHOW VARIABLE name|ALL
where   name  is a user variable name
```

Lists the value of a user variable or all user variables.

4.82 SMOOTH

```
Syntax: SMOOTH id [&idb]
        parameter          (default = number of points in plot)
        [&]idb2            (default = idb + 1)
        mode               (default = 1)
```

Smooths a histogram using the IMSL library routine ICSSCU. See the IMSL manual for complete details. The smoothing parameter is the χ^2 for the histogram to be smoothed and the smoothed histogram. It is recommended to lie between `N-SQRT(2N)` and `N+SQRT(2N)`. Smoothing **parameter** = 0 produces the natural cubic spline which interpolates the histogram points. The result of the smoothing will be put in a histogram with the same primary identifier as the input histogram and the secondary identifier you specify. The two modes are to store it as a smooth curve (errors on each point will be 0), or as a histogram with the same number of points as the input histogram (errors on each point will be the same as errors in the input histogram).

4.83 SPAWN

```
Syntax: SPAWN [command]
```

Alias for **SHELL**. Spawns a DCL (Vax) or Shell (Apollo) command. You can specify the shell on the Apollo using the **SET SHELL** command. If no command is given a new process is created. Use the command **Return** to return to Mn_Fit when you have finished in the subprocess. On the Vax you can also **ATTACH** to the main process and then use **ATTACH** in Mn_Fit to re-enter the subprocess if you wish.

Note that **SHELL** or **SPAWN** without a command seems to work fine for all shells **/com/sh, /bin/sh, /bin/ksh and /bin/csh**. However only the Aegis shell **/com/sh** works with a command at present. For the Unix shells you must give CTRL/D to exit and not **return**.

4.84 SPLINE

```
Syntax: SPLINE id [&idb]
        nknot
        knot1,knot2,...
        [&]idb2                      (default = idb + 1)
        mode                         (default = 0)
```

Spline fits a histogram using the IMSL library routine ICSVKU. See the IMSL manual for complete details. The number of knots must be at least 2 and the knots must be monotonically increasing. The first knot must be less than or equal to the first point in the histogram and the last knot must be greater than or equal to the last point in the histogram. The result of the smoothing will be put in a histogram with the same primary identifier as the input histogram and the secondary identifier you specify. The 2 modes are to store it as a smooth curve (errors on each point will be 0), or as a histogram with the same number of points as the input histogram (errors on each point will be the same as errors in the input histogram).

4.85 SQUEEZE

```
Syntax: SQUEEZE
```

Gets back unused space in the array where the histograms are stored.

4.86 STORE

```
Syntax:   STORE[/NEW/UPDATE] filename id1[&idb1] [id2[&idb2]...]
where:    filename       is the filename to store the plots in
          id1,id2...     are the plot identifiers
          idb1,idb2...   are the (optional) secondary identifiers
Defaults: /NEW
```

Stores histograms in a file. They can be read back in with the FETCH command. To store all histograms specify id1 = 0. You can also specify a range of histograms using the syntax id1:id2. The secondary identifier will not be used in the HBOOK identifier. You can use the MDIR and CDIR commands to specify which directory the histogram should be stored in. The file is closed after each STORE command. To add more histograms to the file use STORE/UPDATE.

4.86.1 /NEW

```
Syntax: STORE/NEW ...
```

Stores the histograms in a new file. On computers which do not have version numbers for files any old file will get overwritten.

4.86.2 /UPDATE

```
Syntax: STORE/UPDATE ...
```

Stores the histograms in an already existing HBOOK direct access file.

4.87 SUBTRACT

```
Syntax: SUBTRACT id1 [:id1n] [&idb1] id2 [:id2n] [&idb2] id3 [:id3n] [&idb3]
        scale1 scale2                             (default scale = 1.0 1.0)
 where: id1,id2   are the input histogram identifiers
        id3       is the output histogram identifier
        idb1,idb2 are the (optional) input secondary identifiers
   and  idb3      is the (optional) output secondary identifier
```

Subtracts two histograms (id1, id2) to make a third one. To specify the secondary identifier, precede it by a &, otherwise the default will be used. (Use the SET IDB command to change the default). The scale factors are optional. To avoid confusion, you should give a <CR> after the identifiers or make sure the scale factors are given as real numbers.

To subtract a range of histograms, the primary identifiers you give for the input and output histograms must be the same, but you can specify different secondary identifiers. For example, SUBTRACT 300:400&1 300:400&2 300 : 400 & 10 will subtract all histograms with primary identifiers 300 to 400 and secondary identifiers 1, from those with secondary identifiers 2, putting the results into histograms with the same primary identifiers and secondary identifier 10. If you give primary identifier 0, the operation will be performed on all plots with the given secondary identifier.

4.88 SUM

```
Syntax: SUM id [&idb] xlo xhi
 where: id   is the plot identifier
        idb  is the (optional) secondary identifier
        xlo  is the lower limit of the range to sum
        xhi  is the upper limit of the range to sum
```

Sums the contents of a plot over the range given. The result is printed and is also stored in register 101, which you can use with the syntax R101. The number of points summed over is in register 102.

4.89 SURFACE

```
Syntax: SURFACE id [&idb] theta phi         (default theta=30, phi=30)
```

Alias for HIST SURFACE. Makes a surface plot of a 2-dimensional histogram. Theta and phi are the angles from which you want to view the plot in degrees. They must lie between 0 and 90.

4.90 TITLE

```
Syntax: TITLE id [&idb] new_title
```

Changes the title on a histogram.

4.91 WAIT

```
Syntax: WAIT sec
 where  sec  is the time to wait in seconds
```

Tells Mn_Fit to wait for a time – implemented on the Vax and Apollo.

4.92 WINDOW

```
Syntax: WINDOW nx ny
        sepx sepy
 where: nx   is the number of windows in the x direction
        ny   is the number of windows in the y direction
        sepx is the separation between the plots in the x direction
        sepy is the separation between the plots in the y direction
```

Alias for SET WINDOW. See section 4.79.66 on page 111 (SET WINDOW) for more details.

4.93 NO_WINDOW

Syntax: NO_WINDOW

Alias for `SET NO_WINDOW`. Turns off windowing.

4.94 WRITE

Syntax: WRITE DATA|LOG

Writes out either a histogram to a card image format file or a log of the commands you gave at the terminal.

4.94.1 DATA

Syntax: WRITE DATA filename id [&idb]

Alias for `DAT_STORE`. See section 4.22 on page 54 (DAT_STORE) for more details.

4.94.2 LOG

Syntax: WRITE LOG filename ncommand

Writes the last **ncommand** commands you have given from the terminal to a file, which you can then edit and execute as a command file. Use the `SHOW LOG` command to look at the commands you have given and the `EDIT` command to edit the file you write out. Use the `EXECUTE` or `READ COMMAND` commands to execute the commands in the file. You can use the `SET LOG ON` command to turn on automatic logging of commands and then all commands will get written to the file `mn_fit.log`.

Chapter 5

Reference Manual for MINUIT

5.1 BACK_SUB

```
Syntax: BACK_SUB [&]idb [[&]idb2 ...] nmode
 where: idb     is the secondary identifier for the background
                subtracted plot
        nmode   = 1 means just subtract the background
                = 2 means divide by it also
```

Makes a background subtracted histogram from the last fit. You must use the **SET BACKGROUND** command beforehand to specify which functions are the background. You will be prompted for which mode you want to use, and what secondary identifier you wish to give the result. If you are fitting more than 1 histogram simultaneously, you should give a secondary identifier for each of them.

5.2 CALLS

Command has been renamed to MAX_CALLS.

5.3 CONSTRAIN

```
Syntax:   CONSTRAIN par expression
where:    par         is a MINUIT parameter number
          expression  is an arithmetic expression
```

Constrains the value of a parameter that is being fit. The parameter is fixed and its value is calculated every iteration from the constraint. The constraint can be any arithmetic expression (see section 1.9 on page 20 (Expressions) for details). The most common use is to fix 2 parameters relative to each other. You can remove a constraint using the **UNCONSTRAIN** command. You can list the current constraints using the **SHOW CONSTRAINT** command.

WARNINGS: The constraint is parsed at the time it is defined and is calculated before the loop over the points being fit is done. Thus if you use registers in the constraint their contents at the time of definition will be used.

This command is only available in the CERN version of MINUIT.

5.4 UNCONSTRAIN

```
Syntax:   UNCONSTRAIN npar
where:    npar  is the MINUIT parameter number
```

Removes the constraint on a parameter. See section 5.3 on page 120 (MINUIT CONSTRAIN) for details on the use of constraints.

This command is only available in the CERN version of MINUIT.

5.5 CONTOUR

```
Syntax: CONTOUR i1 i2 [devs] [ngrid]
    or  CONTOUR i1 i2 xlo xhi ylo yhi
Defaults: devs=2.5, ngrid=25
```

In the CERN version of MINUIT only the first syntax is available. The ngrid parameter is not in the C. Rippich version of MINUIT.

Makes a 2-dim plot of FCN in the 2 variables, ranging in:

```
VAR1 = CURRENT-VAR1 +- DEVS*CURRENT-ERROR
VAR2 = CURRENT-VAR2 +- DEVS*CURRENT-ERROR
```

(or uses explicit range xlo...xhi, ylo...yhi if given). Plotted are characters 1, 2, ... A, ... Z, *, ... That means that 1 stands for AMIN+1*UP, 2 for AMIN+4*UP, 3 for AMIN+9*UP, etc.

5.6 COVARIANCE

```
Syntax: COVARIANCE
```

Reads in the covariance matrix from the current input unit. Do not use inside Mn_Fit.

5.7 DISPLAY

```
Syntax: DISPLAY [&idb]
```

Displays the fit results on the current graphics device. In all that follows id are the primary identifier(s) of the plot(s) you are fitting. You can change what is displayed using the SET DISPLAY MODE command (options include showing the fit, background subtracted fit or both).

If you are fitting a 1-dimensional plot, the part of the function which is included in the fit is stored in plot id&981, and that which is excluded is stored in id&982.

Parameters that are fixed are prefixed by a * and those that are constrained are prefixed by a #.

If you are using display mode -2, 2, -3 or 3 you must give the secondary identifiers of the background subtracted plots. You must also have given the SET BACKGROUND command beforehand to specify which function(s) are the background.

If you are fitting a 2-dimensional plot, the part of the function which which is included in the fit is stored in plot **id&981**, while **id&982** contains the signed χ^2 contribution of each point. A positive value means that the data was higher then the function for that point. Positive values are displayed as filled squares, while negative values are displayed as open squares (unless you change the symbol). Plot **id&982** is the one that is drawn in the display for 2-dimensional plots, so if, for example, you want to change the limits, you must use the syntax **SET PLOT id&982 X LIMIT** etc.

5.8 DUMP

```
Syntax: DUMP
```

Gives the data and fit values for each point and their contribution to the χ^2 or likelihood.

5.9 END

```
Syntax: END [par]
```

This command has been eliminated in the Mn_Fit version of MINUIT.

Restarts with a new set of datacards. Giving a non-zero parameter means do not enforce a **CALL_FCN 3** if it has not been done yet.

5.10 EXCLUDE

```
Syntax: EXCLUDE xlo xhi
    or  EXCLUDE axis xlo xhi
 where: axis  is the axis name (defaults are x and y)
        xlo   is the lower limit of the exclusion
        xhi   is the upper limit of the exclusion
```

Excludes a region of the histogram from the fit. If you are fitting a 2-dimensional histogram, you must specify which axis the exclusion applies to. If you omit the axis name the default is the x axis. If you are fitting more than one histogram simultaneously, you will be prompted for which histogram the exclusion applies to. If you just give the command **EXCLUDE**, you can give several regions to exclude, then hit **<CR>** to exit **EXCLUDE**.

5.11 NO_EXCLUDE

```
Syntax: NO_EXCLUDE
```

Removes any exclusions.

5.12 EXIT

```
Syntax: EXIT [par]
```

Returns to program that called MINUIT. Giving a non-zero parameter means do not enforce a **CALL_FCN 3** if it has not been done yet.

5.13 FCN_DRAW

```
Syntax: FCN_DRAW k xlo xhi ylo yhi
Defaults: xlo = ALIM,   xhi = BLIM,  ylo = AMIN-20*UP, yhi=AMIN+20*UP.
          (If x has no limits, defaults are x(current) +- parabolic error)
```

Draws FCN as a function of 1 parameter. This command is superceded by FCN_PLOT and PROB_PLOT if you have a graphics device.

Scans xlo to xhi in 50 steps and bins the FCN value from ylo to yhi in 30 steps (so as to fit on 1 page).

NO check is done on exceeding limits on variable x (if any), so be careful!

5.14 FCN_PLOT

```
Syntax: FCN_PLOT name xlo xhi ylo yhi
 where: name  is the name of the variable you want to plot
        xlo   is the minimum value
        xhi   is the maximum value
        ylo   is the minimum value of the chi**2 or likelihood
        yhi   is the maximum value of the chi**2 or likelihood
```

Draws the χ^2 or likelihood as a function of 1 parameter. NO check is done on exceeding limits on variable x (if any), so be careful!

5.15 FIT_INFO

```
Syntax: FIT_INFO
```

Gives information on the value of the parameters, etc. Parameters that are fixed are prefixed by a * and those that are constrained are prefixed by a #. Both the MINUIT and the Mn_Fit parameter numbers are given.

5.16 FIX

```
Syntax: FIX var1 [var2 ...]
```

Fixes the list of variables you give. You can specify up to 7.

5.17 FLOAT

```
Syntax: FLOAT var1 [var2 ...]
```

Restores up to 7 parameters (reverse of FIX).

5.18 GRADIENT

```
Syntax: GRADIENT
```

This command should not be used inside Mn_Fit. Indicates that your FCN routine is capable of calculating first derivatives. (Means IFLAG=2 will also be used in addition to the standard IFLAG=4). They will then be used by MIGRAD (instead of by-hand finite difference quotients which may be inaccurate). Mind you, it really does not pay off. Probably you will boil off a lot of c.p.u. time without getting your money's worth back. Difference quotients are usually quite good (except maybe when calculating the error matrix, but you'll run MINOS anyway, won't you???)

NO_GRADIENT turns it off again.

5.19 NO_GRADIENT

```
Syntax: NO_GRADIENT
```

Turns off GRADIENT.

5.20 HESSE

```
Syntax: HESSE
```

Recalculates the covariance matrix. This is very useful if the errors get too small for some reason.

Note that when the matrix gets printed the off-diagonal elements are the correlation coefficients, not the elements of the matrix:

```
C(i,j) = V(i,j) / sqrt(V(i,i)*V(j,j)).
```

The diagonal elements are the parabolic errors.

5.21 IMPROVE

```
Syntax: IMPROVE nloop
 where: nloop  is the number of tries to make
               (default = number of variables)
```

Tries to improve the current fit. Be careful that your parameters do not go wild using this routine. It has a nasty habit of giving floating overflows, so you should protect your function against going negative, by setting limits on parameters if necessary.

5.22 INFO

```
Syntax: INFO
```

Provides full info of current fit status. The value of FCN is the current value of the χ^2 or likelihood.

5.23 ITERATIONS

```
Syntax: ITERATIONS [niter]        (default = 1)
```

Shows the current fit every **niter** calls to FCN. WARNING: This command has not been tested for a long time – use with care!

5.24 NO_ITERATIONS

```
Syntax: NO_ITERATIONS
```

Turns off showing of fit iterations.

5.25 MATOUT

```
Syntax: MATOUT
```

Writes the covariance matrix to the current output unit.

5.26 MAX_CALLS

```
Syntax: MAX_CALLS ncall
 where: ncall  is the maximum number of calls
```

Sets the maximum number of calls for SIMPLEX, MIGRAD and MINOS.

5.27 MIGRAD

```
Syntax: MIGRAD [vtest]
 where: vtest  is the convergence criterion (default = 0.04)
```

Invokes the MIGRAD minimizer which is the best one to use for almost all cases. The maximum number of calls can be set with the MAX_CALLS command.

5.28 MINIMIZE

```
Syntax: MINIMIZE
```

This command does SIMPLEX followed by a standard MIGRAD (VTEST=0.04)

5.29 MINOS

```
Syntax: MINOS [pars]
 where: pars  is a list of parameters
```

Calculates the MINOS errors for the list of parameters given. If you omit the list of parameters, the errors are calculated for all floating parameters. The maximum number of calls can be set with the MAX_CALLS command.

Note that when the external covariance matrix gets printed the off-diagonal elements are the correlation coefficients, not the elements of the matrix:

```
C(i,j) = V(i,j) / sqrt(V(i,i)*V(j,j)).
```

The diagonal elements are the parabolic errors.

5.30 MNCONTOUR

```
Syntax:   MNCONTOUR par1 par2 [npnt]
where:    par1  is the first MINUIT parameter number
          par2  is the second MINUIT parameter number
          npnt  is the number of points
Defaults: npnt = 40
```

Makes a graphical contour plot of 2 parameters. For each point the function is minimized with respect to all the other parameters. You can set the contour level using the ERROR_DEF command, which determines the change in χ^2 or likelihood that the contour represents. If the print level is >-1 the contour will also be printed on the terminal.

Note that if you change the error definition to get a 2 sigma contour for example you should re-minimize before giving the MNCONTOUR again. The contour is stored in a scatter plot 98765&999 which you can COPY or RENAME before giving another contour command if you want to keep it.

This command is only available in the CERN version of MINUIT.

5.31 MODIFY

```
Syntax: MODIFY K UK WK AK BK
 where: K    is the MINUIT parameter number you want to modify.
        UK   is its value.
        WK   is its starting error.
        AK   is its lower limit (if any).
        BK   is its upper limit (if any).
```

Modify, fixes and/or restores MINUIT parameters. In addition, a parameter with(out) limits can be changed to have not (have) limits. Each of last 4 parameters can be specified as a (floating) number or a register, parameter etc. or as the character = which means keep the present value.

MODIFY will reset ISW(2) to zero, meaning that all fits start from rock bottom (i.e. SIMPLEX). The covariance matrix is also killed.

In the CERN version MODIFY completely redefines the parameter. Thus if you set the error to 0, the parameter is constant and can only be made variable again by using the MODIFY command - FLOAT does not work.

In the C. Rippich version a parameter that was fixed in the input dialogue stays fixed for all eternity!

5.32 PAGE

```
Syntax: PAGE
```

Starts a new page on unit 6 [ISYSWR].

5.33 PRECISION

```
Syntax: PRECISION edm          (default = 0.1)
 where: edm  is the convergence criterium for the minimizers
```

Sets the convergence criterium for SIMPLEX and MIGRAD.

5.34 PRINTOUT

```
Syntax: PRINTOUT level
 where: level  is the printout level     (default = 0)
```

Changes the level of printout. Suggest `PRINTOUT -4` if you want to suppress all printout. In that case, `MINOS` still gives a full analysis, and `INFO` still works, but all that other junk is cut off. Error/warning messages still get through!

5.35 PROB_PLOT

```
Syntax: PROB_PLOT name xlo xhi ylo yhi
 where: name  is the name of the variable you want to plot
        xlo   is the minimum value
        xhi   is the maximum value
        ylo   is the minimum value of the probability to plot
        yhi   is the maximum value of the probability to plot
```

Draws the fit probability as a function of 1 parameter. Note that the probability is relative to that at the current minimum, i.e. the probability should always be 1 at the present parameter values. If a new minimum is found, an error message will be printed and the probability set to 1 there also. NO check is done on exceeding limits on variable `x` (if any), so be careful!

5.36 RESTORE

```
Syntax: RESTORE n
 where: n = 0 means restore all previously fixed variables
          = 1 means restore the last variable fixed
```

Restores a parameter. In Mn_Fit use the **FLOAT** command. See section 5.17 on page 123 (MINUIT FLOAT).

5.37 SEEK

```
Syntax:   SEEK nloop [devs]
where:    nloop  is the number of calls made in the search
          devs   is the number of deviations each side of the parameter
Defaults: devs = 3
```

Make a random search in all variable parameters for the function minimum. **SEEK** does a uniform search within **devs** times the current step size for each parameter. Each new minimum found shifts the parameters and the centre of the search.

5.38 SIMPLEX

```
Syntax: SIMPLEX
```

Invokes the fairly crude SIMPLEX minimizer.

5.39 SCAN

```
Syntax:   SCAN [npar] [numpts] [from] [to]
where:    npar    is the MINUIT parameter number
          numpts  is the number of points in the scan
          from    is the lower limit of the scan
          to      is the upper limit of the scan
Defaults: npar = 0
```

Scans one or all parameters leaving the others fixed.
This command is only available in the CERN version of MINUIT.

5.40 STANDARD

```
Syntax: STANDARD
```

Calls routine **STAND** (without argument list) whose default version sits on MINUIT.OLB as a simple **RETURN**+**END**. To get your own, include it in the LINK command.

5.41 STOP

```
Syntax: STOP
```

Executes a STOP right now.

5.42 STRATEGY

```
Syntax:    STRATEGY num
where:     num  is the strategy number
Defaults:  num = 1
```

Specifies the strategy for fitting. Higher values mean more FCN calls and more reliable minimization, but more CPU time.

Command only valid in CERN MINUIT.

5.43 UNIT

```
Syntax: UNIT lun
 where: lun  is a new input unit
```

WARNING: Do not use this command in Mn_Fit. Use the EXEC command to read in commands from a file.

Redefines the input unit 5 [ISYSRD]. You get a prompt MINUIT> or OPER> as long as ISYSRD=5. When you switch to ISYSRD.NE.5 (for which you have to make an ASSIGN statement before running your job). You will no longer get a prompt. Unit 5 is understood to be a terminal (also in batch mode).

Appendix A

Acknowledgements

Many of the improvements and bug fixes have been prompted by my CLEO colleagues at Cornell. In particular, Jon Lewis has tested the new versions of Mn_Fit I sent to him and has implemented a number of useful features. He has also read through the manual several times and made a number of suggestions.

While Mn_Fit can probably be seen as competition for PAW, I have made use of a number of features built into PAW, particularly COMIS and the lego plotting code.

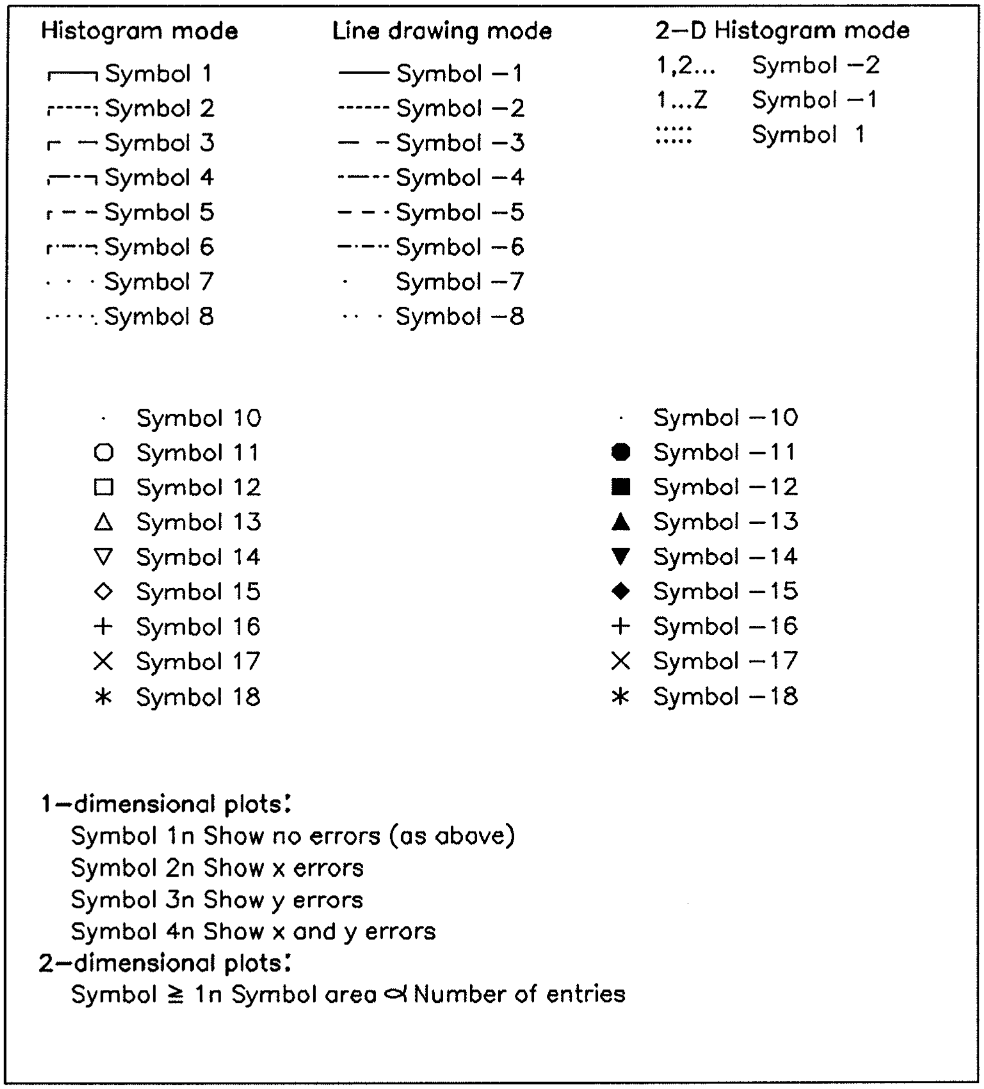
Jon Lewis has implemented the DI3000 interface using HIGZ/DI3000.

Appendix B

Examples of Symbols, Fonts, and Hatching

The same fonts are available in Mn_Fit as in PAW. For some examples see Fig. B.3. Note that the font and precision are specified with the form **spfff**, where **s** is the sign of the font, **p** is the precision and **fff** is the font, e.g. -2013 to get font -13 with precision 2. The fonts available depend on the version of the graphics package used. The ones shown in these figures are for GKSGRAL and Postscript. Note that you only see the Postscript fonts when you actually produce a Postscript file.

GKSGRAL hatchings or DI3000 patterns are shown in Fig. B.4. If HIGZ is used it has device independent hatchings and these are shown in Fig. B.5. Patterns are only rarely available (DECGKS) and are often different for screen and hardcopy devices, so are not shown here.



Histogram mode	Line drawing mode	2–D Histogram mode
┌──┐ Symbol 1	—— Symbol −1	1,2... Symbol −2
┌-----┐ Symbol 2	------ Symbol −2	1...Z Symbol −1
┌ — Symbol 3	— – Symbol −3	::::: Symbol 1
┌—-┐ Symbol 4	-—-- Symbol −4	
┌ – – Symbol 5	– – · Symbol −5	
┌·—·┐ Symbol 6	—·—·· Symbol −6	
· · · Symbol 7	· Symbol −7	
·····. Symbol 8	·· · Symbol −8	

· Symbol 10	· Symbol −10
○ Symbol 11	● Symbol −11
□ Symbol 12	■ Symbol −12
△ Symbol 13	▲ Symbol −13
▽ Symbol 14	▼ Symbol −14
◇ Symbol 15	◆ Symbol −15
+ Symbol 16	+ Symbol −16
× Symbol 17	× Symbol −17
∗ Symbol 18	∗ Symbol −18

1–dimensional plots:

Symbol 1n Show no errors (as above)
Symbol 2n Show x errors
Symbol 3n Show y errors
Symbol 4n Show x and y errors

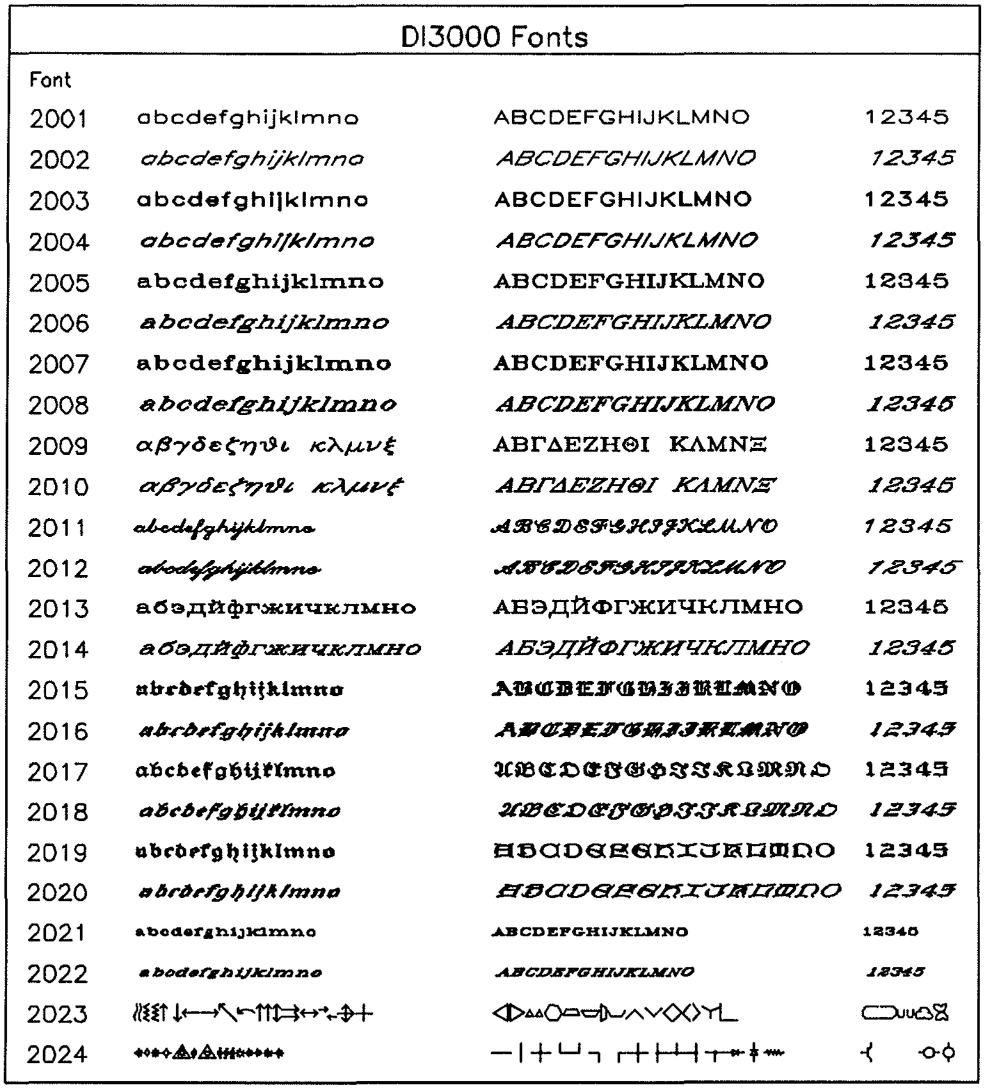
2–dimensional plots:

Symbol ≧ 1n Symbol area ∝ Number of entries

Figure B.1: Mn_Fit Symbols

Upper Roman	Lower Roman	Upper Greek	Lower Greek	Upper Special	Lower Special
A	a	A	α	±	±
B	b	B	β	|	|
C	c	H	η	✤	✡
D	d	Δ	δ	$	$
E	e	E	ε	!	!
F	f	Φ	φ	#	#
G	g	Γ	γ	>	>
H	h	X	χ	?	?
I	i	I	ι	∫	∫
J	j	I	ι	:	:
K	k	K	κ	;	;
L	l	Λ	λ	<	<
M	m	M	μ	[	[
N	n	N	ν	]	]
O	o	O	ο	≥	≤
P	p	Π	π	{	{
Q	q	Θ	ϑ	}	}
R	r	P	ρ	√	√
S	s	Σ	σ	♤	🔔
T	t	T	τ	♡	☨
U	u	ϒ	υ	◇	♣
V	v	X	χ	♧	❀
W	w	Ω	ω	&	&
X	x	Ξ	ξ	×	×
Y	y	Ψ	ψ	%	∝
Z	z	Z	ζ	∞	∞
0	0	0	0	●	○
1	1	1	1	■	□
2	2	2	2	▲	△
3	3	3	3	▼	◊
4	4	4	4	★	☆
5	5	5	5	→	→
6	6	6	6	↑	↑
7	7	7	7	←	←
8	8	8	8	↓	↓
9	9	9	9	,	,
.	.	.	.	◇	□
,	,	,	,	,	,
+	+	+	+	+	+
-	-	-	-	¯	¯
*	*	*	*	∗	∗
/	/	/	/	/	/
=	=	=	=	=	=
(	(	(	(	(	(
)	)	)	)	)	)

Figure B.2: HIGZ Portable Software Characters



DI3000 Fonts

Font			
2001	abcdefghijklmno	ABCDEFGHIJKLMNO	12345
2002	*abcdefghijklmno*	*ABCDEFGHIJKLMNO*	*12345*
2003	**abcdefghijklmno**	**ABCDEFGHIJKLMNO**	**12345**
2004	***abcdefghijklmno***	***ABCDEFGHIJKLMNO***	***12345***
2005	abcdefghijklmno	ABCDEFGHIJKLMNO	12345
2006	*abcdefghijklmno*	*ABCDEFGHIJKLMNO*	*12345*
2007	**abcdefghijklmno**	**ABCDEFGHIJKLMNO**	**12345**
2008	***abcdefghijklmno***	***ABCDEFGHIJKLMNO***	***12345***
2009	αβγδεζηϑι κλμνξ	ΑΒΓΔΕΖΗΘΙ ΚΛΜΝΞ	12345
2010	*αβγδεζηϑι κλμνξ*	*ΑΒΓΔΕΖΗΘΙ ΚΛΜΝΞ*	*12345*
2011	*abcdefghijklmno*	*ABCDEFGHIJKLMNO*	*12345*
2012	***abcdefghijklmno***	***ABCDEFGHIJKLMNO***	***12345***
2013	абэдйфгжичклмно	АБЭДЙФГЖИЧКЛМНО	12345
2014	*абэдйфгжичклмно*	*АБЭДЙФГЖИЧКЛМНО*	*12345*
2015	**abcdefghijklmno**	**ABCDEFGHIJKLMNO**	**12345**
2016	***abcdefghijklmno***	***ABCDEFGHIJKLMNO***	***12345***
2017	abcdefghijklmno	ABCDEFGHIJKLMNO	12345
2018	*abcdefghijklmno*	*ABCDEFGHIJKLMNO*	*12345*
2019	abcdefghijklmno	ABCDEFGHIJKLMNO	12345
2020	*abcdefghijklmno*	*ABCDEFGHIJKLMNO*	*12345*
2021	abcdefghijklmno	ABCDEFGHIJKLMNO	12345
2022	*abcdefghijklmno*	*ABCDEFGHIJKLMNO*	*12345*
2023	⫽↓⟵↖↑↑⇉↔	◁▷▵▵○▭◇∧∨◇◇⊤⊥	⊐⊔⊔◇⊠
2024	◆◆◆▲◆▲◆◆◆	—│+└┐ ┌+├┤┬	┤ ⊸

Figure B.3: Examples of Fonts

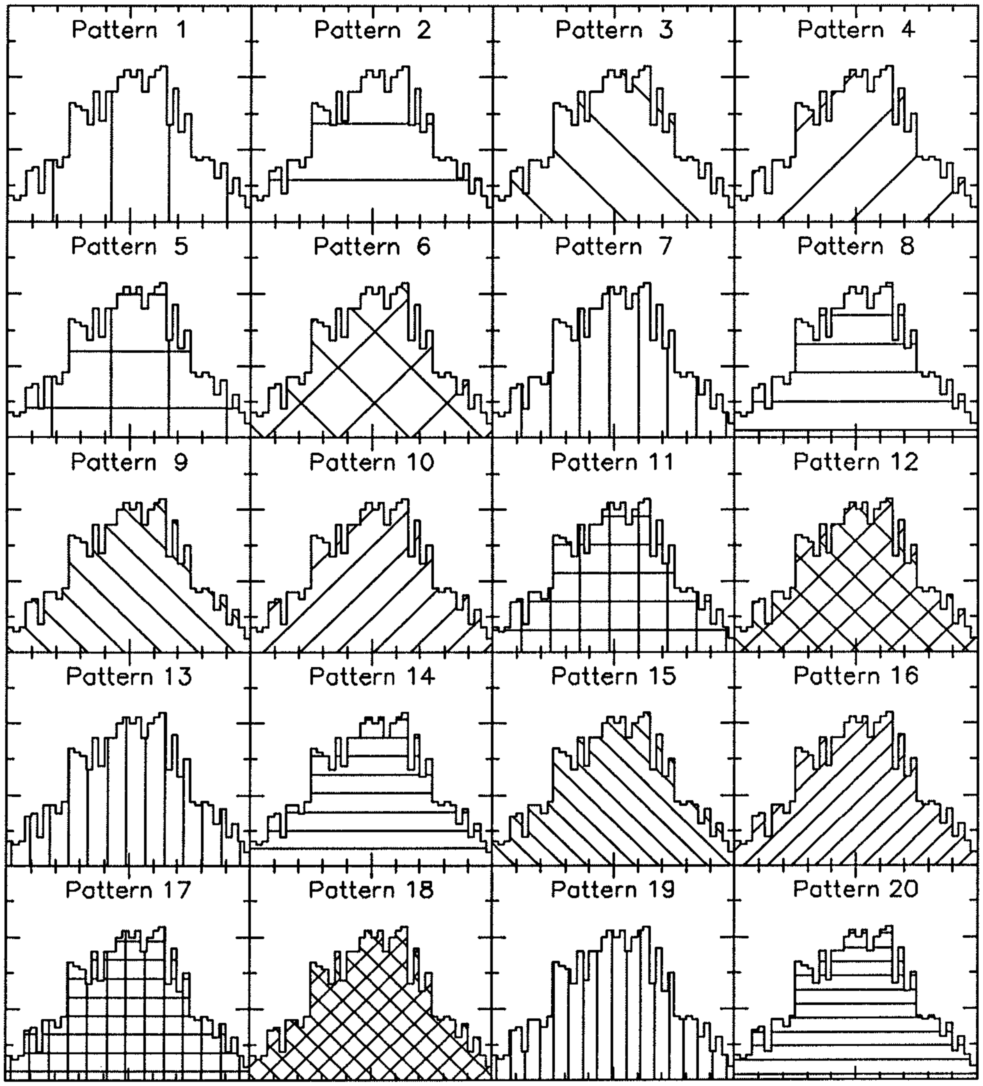


Figure B.4: Examples of DI3000 Patterns

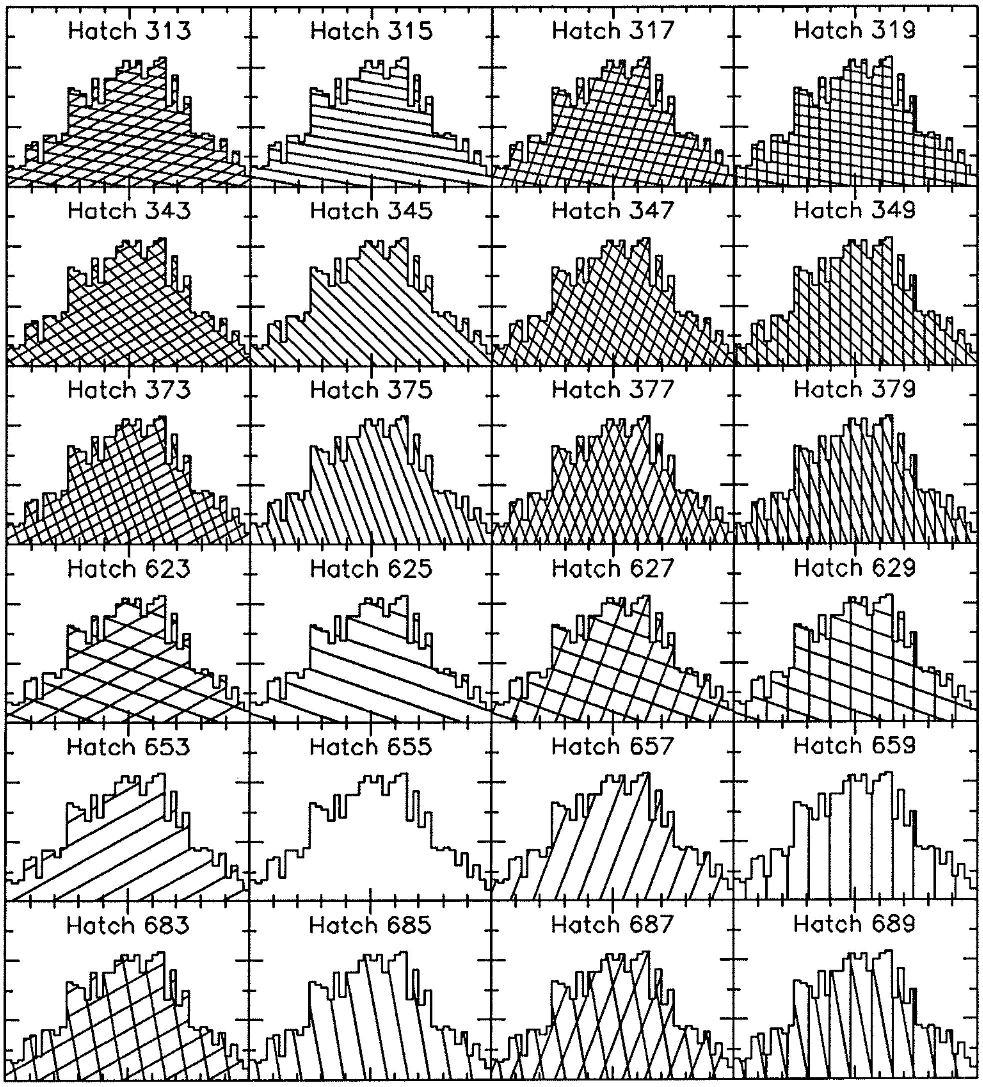


Figure B.5: Examples of HIGZ Hatching

Appendix C

Mn_Fit Examples

This chapter contains a series of examples of Mn_Fit command files. These files were used to produce the figures included here. They start from simple plotting and fitting examples and then add more complicated possibilities. These demonstration files can be found in the CMZ file `mn_fit_cmz:mn_util.cmz` (Vax) or `~/mn_fit_cmz/mn_util.cmz` (Apollo) in the directory `//mn_util/demo`. They are called `demo01, demo02, ...` In the same directory you can find the command files used to produce the figures in Appendix B. These command files can also be found in the directory `mn_fit_dir:` (Vax) or `~/mn_fit_dir` (Apollo). The demonstration files were run on the Apollo. On the Vax you have to replace any `~/mn_fit_test/` by `mn_fit_test:`. You can make these files from the CMZ file by starting up CMZ and giving the commands:

```
* Vax
  file mn_fit_cmz:mn_util -r
  sel VAX
* Apollo
  file ~/mn_fit_cmz/mn_util -r
  sel APOLLO
*
  set *.mnf -d
  ctot -y demo
```

The following demonstration files exist:

`demo01` Simple fetch and plot of an HBOOK 4 histogram.

`demo02` Gaussian fit to the same histogram and display of the result.

`demo03` Fancy plotting of 1-dimensional histograms using colour and hatching. Use of `COMMENT` and `KEY` and control of axes.

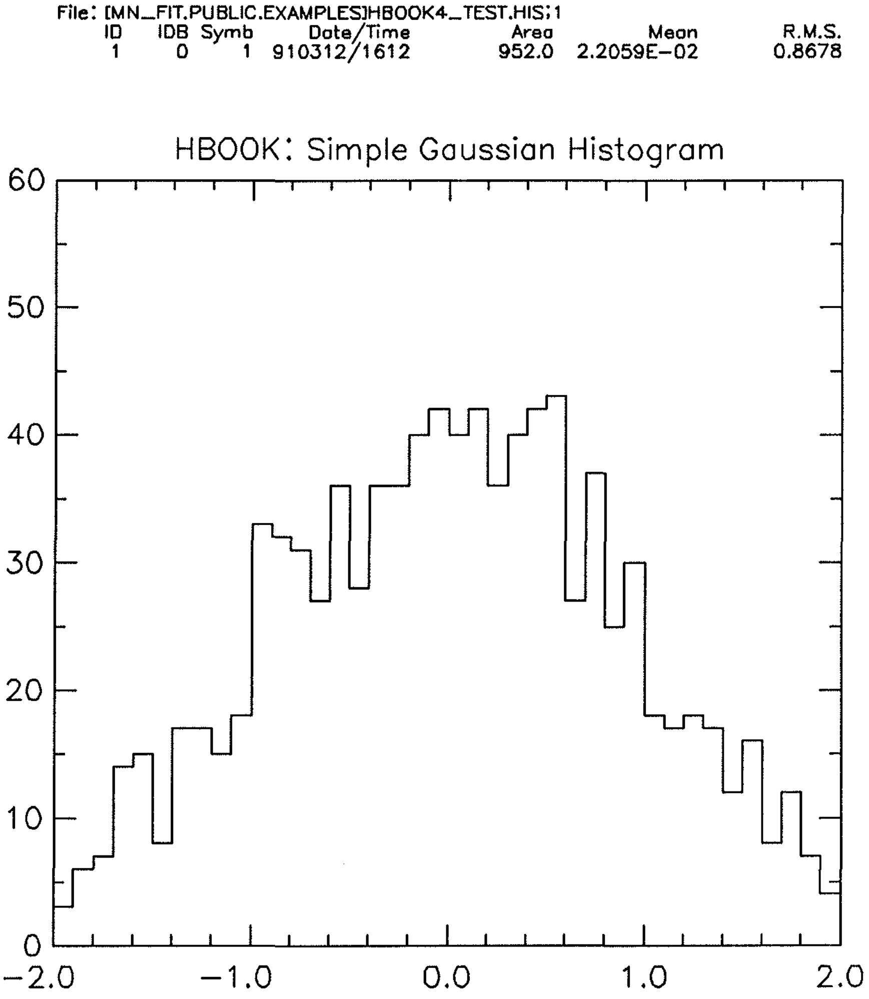
`demo04` 2-dimensional histogram plotting and direct plotting of Ntuples.

`demo05` Simultaneous fit to 2 histograms and overlay of functions.

In the following examples the prompt is shown in **boldface**, user typin is underlined, and comments are shown in *italics*.

C.1 Demonstration File 1: Simple Fetch and Plot

```
MN_CMD> exec demo01
Reading commands from unit  10
File: CDF$ROOT5:[MN_FIT.PUBLIC.DOC]DEMO01.MNF;1
MN_CMD> !                         ----------------------------------------
MN_CMD> !                         Mn_Fit Demonstration file number 1.
MN_CMD> !                         Simple fetch and plot.
MN_CMD> !                         ----------------------------------------
MN_CMD> !
MN_CMD> fet mn_fit_test:hbook4_test.his 1
Reading histograms from unit   1
File: CDF$ROOT5:[MN_FIT.PUBLIC.EXAMPLES]HBOOK4_TEST.HIS;1
The current HBOOK directory is: //MN_HBIN
MN_HBF: A total of     1 plots have been read in
MN_CMD> set default !             Set everything back to default
MN_CMD> set y mode integer !      Show y scale as an integer
MN_CMD> !
MN_CMD> plot 1
MN_CMD> !
MN_CMD> exec demohard 01 !        Make a hardcopy file
Reading commands from unit  11
File: CDF$ROOT5:[MN_FIT.PUBLIC.DOC]DEMOHARD.MNF;1
MN_CMD> !
MN_CMD> !                         Macro to make a QMS hardcopy
MN_CMD> !
MN_CMD> inquire 1 'Give demonstration number'
MN_CMD> set hard demo01.qms !     Set the hardcopy filename
MN_CMD> hardcopy TLP !            Make a Talaris hardcopy
TVCAP: Hardcopy to unit 12, File: demo01.qms
TLP selected
TVRNG: Device TLP        Scaling factor 0.001 will be used
None selected
MN_CMD> close !                   Close the hardcopy file
End of file reached, unit=  11. Exiting
End of file reached, unit=  10. Exiting
```



```
File: [MN_FIT.PUBLIC.EXAMPLES]HBOOK4_TEST.HIS;1
      ID    IDB  Symb        Date/Time              Area          Mean            R.M.S.
       1      0     1   910312/1612            952.0   2.2059E-02          0.8678
```

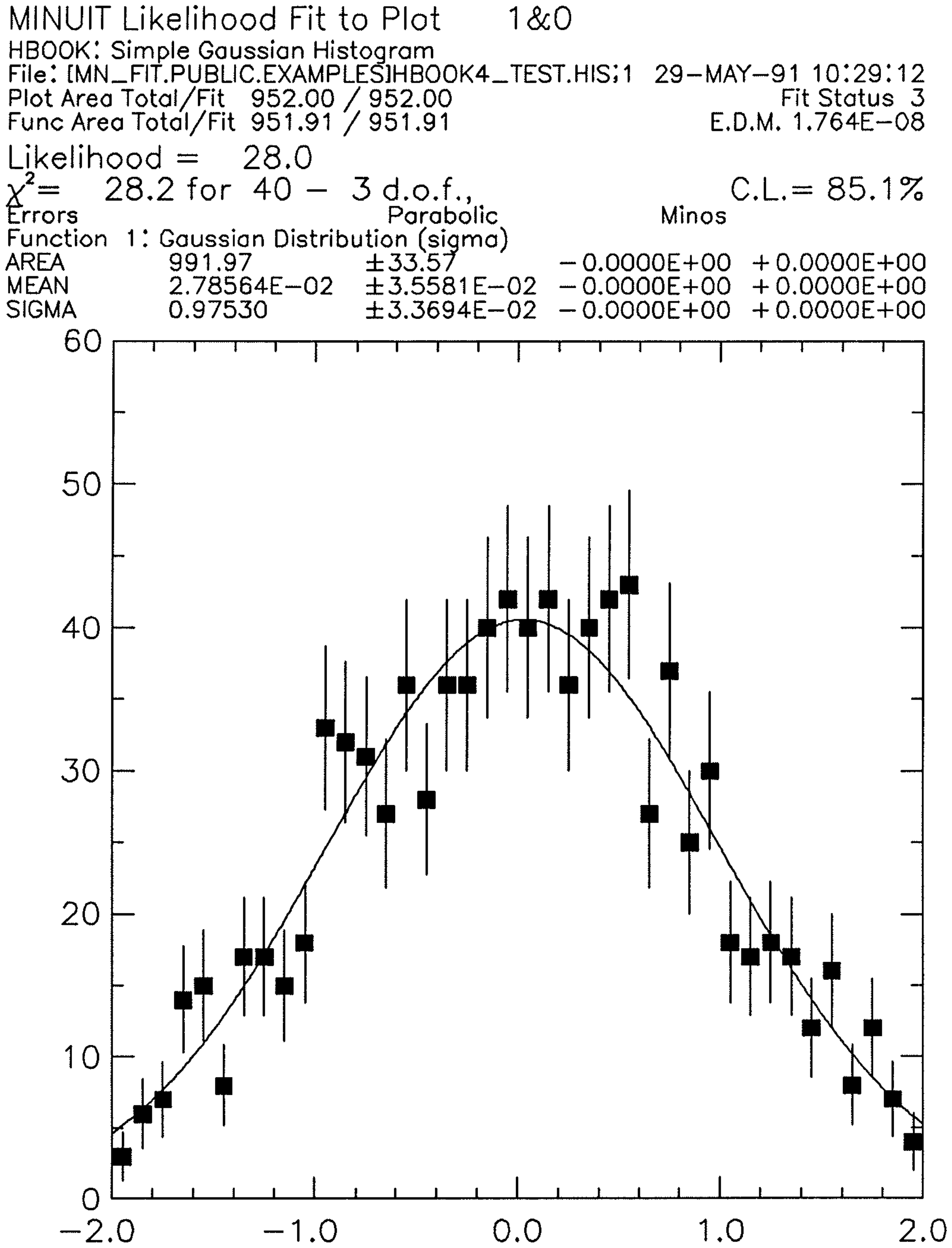
Figure C.1: Demonstration File 1 — Simple Fetch and Plot.

C.2 Demonstration File 2: Simple Gaussian Fit

```
MN_CMD> exec demo02
Reading commands from unit  10
File: CDF$ROOT5:[MN_FIT.PUBLIC.DOC]DEMO02.MNF;1
MN_CMD> !                       ----------------------------------------
MN_CMD> !                       Mn_Fit Demonstration file number 2.
MN_CMD> !                       Gaussian Fit to Histogram 1
MN_CMD> !                       ----------------------------------------
MN_CMD> !
MN_CMD> fet mn_fit_test:hbook4_test.his 1
Reading histograms from unit   1
File: CDF$ROOT5:[MN_FIT.PUBLIC.EXAMPLES]HBOOK4_TEST.HIS;1
The current HBOOK directory is: //MN_HBIN
MN_HBF: A total of     1 plots have been read in
MN_CMD> set default !           Set everything back to default
MN_CMD> set y mode integer !    Show y scale as an integer
MN_CMD> !
MN_CMD> !                       Add a Gaussian in terms of sigma and give the initial values
MN_CMD> fun del 0
MN_CMD> fun add gaus sigma
Function: Gaussian Distribution (sigma)
Give Initial Values for Parameters (Value, Step Size, Lower Limit, Upper Limit)
1 AREA : 1000 100
2 MEAN : 1 1
3 SIGMA : 1 1
MN_CMD> !                       Do a likelihood fit to histogram 1 - fit type 1
MN_CMD> fit 1
Chi**2 (0) or likelihood (1) fit (<CR>=0)? 1
Will do a likelihood fit to histogram:       1    0;
==============================================================================
MINUIT RELEASE 90.10  INITIALIZED.   DIMENSIONS 100/ 50  EPSMAC= 0.56E-16
==============================================================================
**********
**    1 **SET PRINTOUT          0.0000E+00
**********
PARAMETER DEFINITIONS:
NO.   NAME          VALUE      STEP SIZE      LIMITS
1 'AREA       '    1000.0      100.00         no limits
2 'MEAN       '    1.0000      1.0000         no limits
3 'SIGMA      '    1.0000      1.0000         no limits
**********
**    2 **CALL FCN               1.000
**********
FCN=   710.1020     FROM CAll fcn  STATUS=RESET         1 CALLS        1 TOTAL
EDM= unknown      STRATEGY= 1      NO ERROR MATRIX
```

```
EXT PARAMETER              CURRENT GUESS      PHYSICAL LIMITS
NO.   NAME        VALUE        ERROR       NEGATIVE      POSITIVE
1    AREA       1000.0       100.00
2    MEAN       1.0000       1.0000
3    SIGMA      1.0000       1.0000
MINUIT> minimize !                Do the fit
**********
**    3 **MINIMIZE               2000.      0.1000
**********
MIGRAD MINIMIZATION HAS CONVERGED.
FCN=   27.97319     FROM MIGRAD    STATUS=CONVERGED     72 CALLS       73 TOTAL
EDM=  0.18E-07  STRATEGY=1  ERROR MATRIX UNCERTAINTY=  2.3%
EXT PARAMETER                                    STEP          FIRST
NO.   NAME        VALUE          ERROR          SIZE       DERIVATIVE
1    AREA        991.97         33.565       0.88253E-01  -0.67388E-05
2    MEAN      0.27856E-01   0.35581E-01  -0.34004E-03  -0.16390E-02
3    SIGMA      0.97530       0.33694E-01  -0.31218E-04  -0.13471E-02
MINUIT> display !                 Show the results
Results of Fit to Plot(s):        1   0;
Likelihood =    28.0
Chi**2 =      28.2  for  40 -   3 d.o.f.                C.L. = 85.1 %
Plot Area Total/Fit  952.000     /  952.000             Fit Status  3
Func Area Total/Fit  951.911     /  951.911             E.D.M. 1.764E-08
Name         Value                    Errors
Parabolic                Minos
Function  1: Gaussian Distribution (sigma)
1( 1)  1 AREA         991.97      +/-  33.565     - 0.00000E+00 + 0.00000E+00
1( 2)  2 MEAN       2.78564E-02 +/- 3.55811E-02 - 0.00000E+00 + 0.00000E+00
1( 3)  3 SIGMA      0.97530     +/- 3.36939E-02 - 0.00000E+00 + 0.00000E+00
MINUIT> !
MINUIT> exec demohard 02 !        Make a hardcopy file
Reading commands from unit  11
File: CDF$ROOT5:[MN_FIT.PUBLIC.DOC]DEMOHARD.MNF;1
MINUIT> !
MINUIT> !                         Macro to make a QMS hardcopy
MINUIT> !
MINUIT> inquire 1 'Give demonstration number'
MINUIT> set hard demo02.qms !     Set the hardcopy filename
MINUIT> hardcopy TLP !            Make a Talaris hardcopy
TVCAP: Hardcopy to unit 12, File: demo02.qms
TLP selected
TVRNG: Device TLP         Scaling factor 0.001 will be used
None selected
MINUIT> close !                   Close the hardcopy file
End of file reached, unit=  11. Exiting
MINUIT> exit !                    Return from MINUIT
```

```
**********
**    4 **CLEAR
**********
End of file reached, unit=  10. Exiting
```



```
MINUIT Likelihood Fit to Plot      1&0
HBOOK: Simple Gaussian Histogram
File: [MN_FIT.PUBLIC.EXAMPLES]HBOOK4_TEST.HIS;1  29-MAY-91 10:29:12
Plot Area Total/Fit   952.00 / 952.00                    Fit Status  3
Func Area Total/Fit  951.91 / 951.91                  E.D.M. 1.764E-08
Likelihood =    28.0
χ²=    28.2 for  40 -  3 d.o.f.,                     C.L.= 85.1%
Errors                          Parabolic              Minos
Function  1: Gaussian Distribution (sigma)
AREA          991.97          ±33.57          -0.0000E+00  +0.0000E+00
MEAN          2.78564E-02     ±3.5581E-02     -0.0000E+00  +0.0000E+00
SIGMA         0.97530         ±3.3694E-02     -0.0000E+00  +0.0000E+00
```

Figure C.2: Demonstration File 2 — Simple Gaussian Fit.

C.3 Demonstration File 3: Fancy plotting with overlays, inserts, colours and fonts

```
MN_CMD> exec demo03
Reading commands from unit  10
File: CDF$ROOT5:[MN_FIT.PUBLIC.DOC]DEMO03.MNF;1
MN_CMD> !                    ------------------------------------
MN_CMD> !                    Mn_Fit Demonstration file number 3.
MN_CMD> !                    Demonstration of Windowing, with COMMENTS, KEYS
MN_CMD> !                    and use of options for axes. If you have a colour printer you can
MN_CMD> !                    make this picture in colour also.
MN_CMD> !                    ------------------------------------
MN_CMD> !
MN_CMD> !                    Set errors on 0 bins to 0, and do not display these points
MN_CMD> set err_zero off
MN_CMD> set show_zero off
MN_CMD> fetch mn_fit_test:hbook4_test.his 0
Reading histograms from unit   1
File: CDF$ROOT5:[MN_FIT.PUBLIC.EXAMPLES]HBOOK4_TEST.HIS;1
The current HBOOK directory is: //MN_HBIN
MN_HBF: A total of    11 plots have been read in
MN_CMD> COP 7 7&1
MN_CMD> PART 7&1 0 4
MN_CMD> copy 5 5&1
MN_CMD> part 5&1 0 4
MN_CMD> !
MN_CMD> set
SET> Give variable name or ?: default
SET> Give variable name or ?: header off
SET> Give variable name or ?: x psize 20
SET> Give variable name or ?: y psize 24
SET> Give variable name or ?: x marg 2.5
SET> Give variable name or ?: thick 1.5
SET> Give variable name or ?: thick frame 2
SET> Give variable name or ?: x mode int
SET> Give variable name or ?: y mode int
SET> Give variable name or ?: sym 32
SET> Give variable name or ?: !   Make 3 plots with 0 y separation and only d
SET> Give variable name or ?: !   and left. 1 x label per page.
SET> Give variable name or ?: WIND 1 3 0 0
SET> Give variable name or ?: FRAME TOP OFF RIGHT OFF
SET> Give variable name or ?: LABEL BOTTOM PAGE
SET> Give variable name or ?: X LABEL 'Arbitrary Units' = = = = = -2104
SET> Give variable name or ?: X SCALE = = 0.3 = = -2003
SET> Give variable name or ?: Y SCALE = = 0.3 = = -2003
SET> Give variable name or ?: TSIZE 0.3
SET> Give variable name or ?: endset
MN_CMD> !
MN_CMD> !                      Set up the colours. Use a user variable to store the default symbol
MN_CMD> !
MN_CMD> dep csymbol = 2
csymbol =   2.00000
MN_CMD> set
SET> Give variable name or ?: colour frame 1
SET> Give variable name or ?: colour header 1
SET> Give variable name or ?: colour label 4
SET> Give variable name or ?: colour scale 3
SET> Give variable name or ?: colour symbol 2
SET> Give variable name or ?: endset
MN_CMD> !
MN_CMD> !                      Set an overall user title for all the plots.
MN_CMD> !                      Make it a bit bigger and use a nice font.
MN_CMD> !
MN_CMD> SET TITLE USER 'Gaussian Signal + Exponential Background'
MN_CMD> set title position = = 0.4 = = -2003
MN_CMD> !
MN_CMD> !                      First make the bare plots. Then edit them to get them
MN_CMD> !                      exactly as I want and finally REDRAW.
MN_CMD> !
MN_CMD> plot 7
MN_CMD> !
MN_CMD> !                      Closed axes again
MN_CMD> !
MN_CMD> SET FRAM TOP ON
MN_CMD> SET FRAM RIGHT ON
MN_CMD> !
MN_CMD> !                      Add an insert to this plot. Window margins are measured
MN_CMD> !                      from the position of the x and y margins.
MN_CMD> !
MN_CMD> set
SET> Give variable name or ?: x wsize 4
SET> Give variable name or ?: y wsize 3
SET> Give variable name or ?: x wmarg 10
SET> Give variable name or ?: y wmarg 12
SET> Give variable name or ?: sym 1
SET> Give variable name or ?: hatch 343
SET> Give variable name or ?: colour symbol 6
SET> Give variable name or ?: endset
MN_CMD> plot/noclear 5&1
MN_CMD> !
MN_CMD> !                           Set margins back to default
MN_CMD> !
MN_CMD> set
SET> Give variable name or ?: x wsize 0
SET> Give variable name or ?: y wsize 0
SET> Give variable name or ?: x wmarg 0
SET> Give variable name or ?: y wmarg 0
SET> Give variable name or ?: sym -14
SET> Give variable name or ?: hatch 0
SET> Give variable name or ?: colour symbol 2
SET> Give variable name or ?: Y LABEL 'Number of Events' -1.5 = = = = -2104
SET> Give variable name or ?: endset
MN_CMD> !
MN_CMD> PL 5
MN_CMD> set colour symbol 7
MN_CMD> OVE/Diff 6 1 653 !          Overlay plot 6 on a different scale
MN_CMD> set sym 0
MN_CMD> set hatch 343 !             Cross-hatching
MN_CMD> set colour symbol 7
MN_CMD> PL 7&1
MN_CMD> set colour symbol 1
MN_CMD> ove 7&1 43 0
MN_CMD> !
MN_CMD> !                           Plot 5:
MN_CMD> !                           I want ticks just on the bottom and left on the outside
MN_CMD> !                           and on the inside on the left as well
MN_CMD> !                           Plot 7&1:
MN_CMD> !                           Separate it from the other plots.
MN_CMD> !                           Change its y label
MN_CMD> !                           Plot 6:
MN_CMD> !                           Add an axis label.
MN_CMD> !                           Plot 5&1
MN_CMD> !                           Add the scale to the bottom of the plot.
MN_CMD> !                           Reduce the tick sizes
MN_CMD> !
MN_CMD> set plot 5
SET> Give variable name or ?: TICK BOTTOM OUT ON
SET> Give variable name or ?: TICK ALL INS OFF LEFT INSIDE ON
SET> Give variable name or ?: TICK LEFT OUT ON
SET> Give variable name or ?: Y SCALE -0.5
SET> Give variable name or ?: SCALE BOTTOM ON
SET> Give variable name or ?: SCALE X = -0.7
SET> Give variable name or ?: XTICK = = = 0.15 0.3
SET> Give variable name or ?: Y LIM 0 500
SET> Give variable name or ?: ENDSET
MN_CMD> SET
```

```
SET> Give variable name or ?: PL 7 & 1
SET> Give variable name or ?: !   Make the plot smaller and centre it
SET> Give variable name or ?: X WSIZE 10
SET> Give variable name or ?: Y WSIZE 3.5
SET> Give variable name or ?: X WMARG 2.5
SET> Give variable name or ?: y label 'y label' =
SET> Give variable name or ?: PLOT 6
SET> Give variable name or ?: y lab 'More Events' -1.6 =
SET> Give variable name or ?: plot 5&1
SET> Give variable name or ?: scale bot on
SET> Give variable name or ?: x scale = -0.4
SET> Give variable name or ?: x tick = = = 0.15 0.3
SET> Give variable name or ?: y tick = = = 0.15 0.3
SET> Give variable name or ?: ENDSET
MN_CMD> REDRAW
MN_CMD> !
MN_CMD> !                    Add some keys and comments.
MN_CMD> !                    Keys and Comments are associated with plots, so you have to start
MN_CMD> !                    a new set for each plot. Use cm for some of them and plot
MN_CMD> !                    co-ordinates for others. Note that if you include the position on
MN_CMD> !                    the same line as the text, the text must be in single quotes
MN_CMD> !
MN_CMD> KEY 7 NEW
KEY> Give symbol number: 32 'Signal + Background' 3.5 16.5 0.3 = = = -6
MN_CMD> END
MN_CMD> dep r1 = 400
r1 =    400.000
MN_CMD> dep r2 = r1 - 80
r2 =    320.000
MN_CMD> KEY
KEY> Give histogram ID: 5 NEW -34 'Signal' 3.3 r1 0.3 = = pl -6 2
KEY> Give command or ?: NEW
KEY> Give symbol number: 1 Background
Give new values or ?: 3.3 r2 0.3 = = pl = 7
KEY> Give command or ?: END
MN_CMD> !
MN_CMD> !                    Add a comment and an arrow pointing to the signal
MN_CMD> !
MN_CMD> COMM 7&1 NEW
Give text: Signal
Give new values or ?: 0.9 50 0.3 = r pl -8 2
MN_CMD> END
MN_CMD> !
MN_CMD> DRAW ARROW
Give new values or ?: -1 2 2 PL
```

```
DRAW> Point 1 Give x,y: 1,90
DRAW> Point 2 Give x,y: 1.8 160
DRAW> Give item to draw, command or ?: END
MN_CMD> !
MN_CMD> exec demohard 03 !        Make a hardcopy file
Reading commands from unit  11
File: CDF$ROOT5:[MN_FIT.PUBLIC.DOC]DEMOHARD.MNF;1
MN_CMD> !
MN_CMD> !                         Macro to make a QMS hardcopy
MN_CMD> !
MN_CMD> inquire 1 'Give demonstration number'
MN_CMD> set hard demo03.qms !    Set the hardcopy filename
MN_CMD> hardcopy TLP !            Make a Talaris hardcopy
TVCAP: Hardcopy to unit 12, File: demo03.qms
TLP selected
None selected
MN_CMD> close !                   Close the hardcopy file
End of file reached, unit=  11. Exiting
End of file reached, unit=  10. Exiting
MN_CMD> !
MN_CMD> !                         Make a colour version of this plot.
MN_CMD> !
MN_CMD> set hard demo03.510
MN_CMD> hard 510
TVCAP: Hardcopy to unit 12, File: demo03.510
510 selected
TVRNG: Device 510          Scaling factor 0.001 will be used
None selected
MN_CMD> close
```

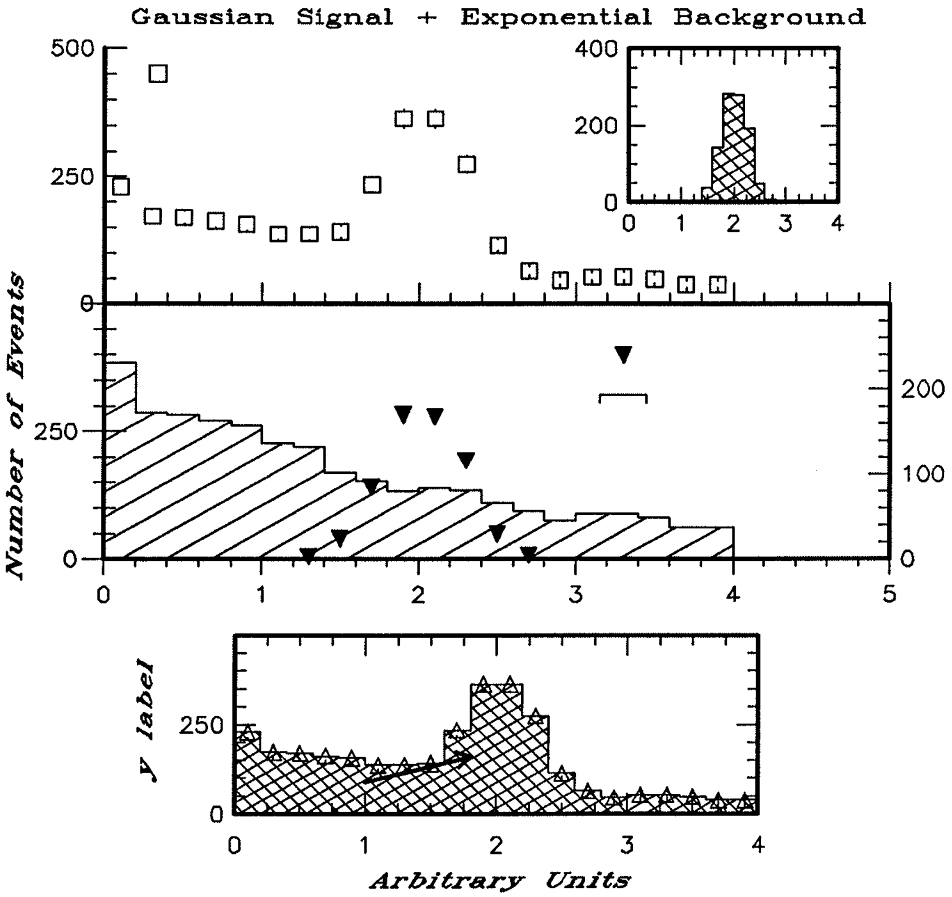
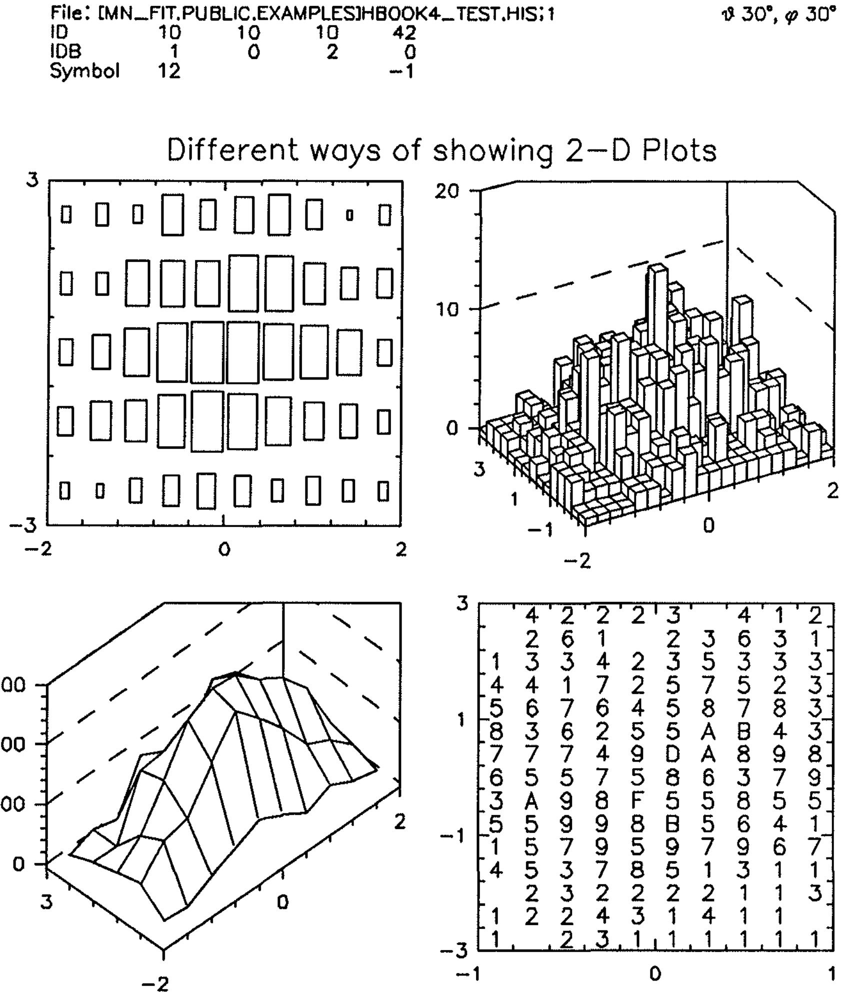


Figure C.3: Demonstration File 3 — Fancy plotting with overlays, inserts, colours and fonts.

C.4 Demonstration File 4: Different ways of displaying 2-d histograms

```
MN_CMD> exec demo04
Reading commands from unit  10
File: CDF$ROOT5:[MN_FIT.PUBLIC.DOC]DEMO04.MNF;1
MN_CMD> !                            ----------------------------------------
MN_CMD> !                            Mn_Fit Demonstration file number 4.
MN_CMD> !                            Lego and surface plots + direct plotting of an Ntuple
MN_CMD> !                            ----------------------------------------
MN_CMD> !
MN_CMD> fetch mn_fit_test:hbook4_test.his 10
Reading histograms from unit    1
File: CDF$ROOT5:[MN_FIT.PUBLIC.EXAMPLES]HBOOK4_TEST.HIS;1
The current HBOOK directory is: //MN_HBIN
MN_HBF: A total of      1 plots have been read in
MN_CMD> cdir ntuple
MN_CMD> fetch mn_fit_test:hbook4_test.his 42
Reading histograms from unit    1
File: CDF$ROOT5:[MN_FIT.PUBLIC.EXAMPLES]HBOOK4_TEST.HIS;1
The current HBOOK directory is: //MN_HBIN/NTUPLE
The current HBOOK directory is: //PAWC/NTUPLE
MN_HBF: A total of      1 plots have been read in
MN_CMD> set
SET> Give variable name or ?: default
SET> Give variable name or ?: x mode int
SET> Give variable name or ?: y mode int
SET> Give variable name or ?: z mode int
SET> Give variable name or ?: Only give plot id and symbol
SET> Give variable name or ?: header brief !
SET> Give variable name or ?: title user 'Different ways of showing 2-D Plots
SET> Give variable name or ?: Windows 2x2 with 1.5cm spaci
SET> Give variable name or ?: wind 2 2 1.5 1.5 !
SET> Give variable name or ?: endset
MN_CMD> cop 10 10&1
MN_CMD> scale 10&0 10&2 5.0
MN_CMD> rebin 10&1 1 20 10 1 15 5
X        axis will have   10 bins, using bins    1 ->    20
Y        axis will have    5 bins, using bins    1 ->    15
MN_CMD> rebin 10&2 1 20 10 1 15 5
X        axis will have   10 bins, using bins    1 ->    20
Y        axis will have    5 bins, using bins    1 ->    15
MN_CMD> !
MN_CMD> set x zero off !          Turn off the lines at x=0 and y=0
MN_CMD> set y zero off
MN_CMD> PLOT 10&1 !               Plot as a table
MN_CMD> LEGO 10 30 30 !          Lego plot
MN_CMD> surf 10&2 45 45 !        Rebinned version as a surface plot
MN_CMD> !
MN_CMD> !                        Plot the Ntuple directly as a table
MN_CMD> !
MN_CMD> set ntuple 42 x var1 y var2 z weight dz dweight
MN_CMD> set sym -1
MN_CMD> set x lim -1 1
MN_CMD> set y lim -3 3
MN_CMD> pl/ntuple 42
MN_CMD> !
MN_CMD> exec demohard 04 !       Make a hardcopy file
Reading commands from unit  11
File: CDF$ROOT5:[MN_FIT.PUBLIC.DOC]DEMOHARD.MNF;1
MN_CMD> !
MN_CMD> !                        Macro to make a QMS hardcopy
MN_CMD> !
MN_CMD> inquire 1 'Give demonstration number'
MN_CMD> set hard demo04.qms !    Set the hardcopy filename
MN_CMD> hardcopy TLP !           Make a Talaris hardcopy
TVCAP: Hardcopy to unit 12, File: demo04.qms
TLP selected
TVRNG: Device TLP         Scaling factor 0.001 will be used
None selected
MN_CMD> close !                  Close the hardcopy file
End of file reached, unit=  11. Exiting
End of file reached, unit=  10. Exiting
```



```
File: [MN_FIT.PUBLIC.EXAMPLES]HBOOK4_TEST.HIS;1                    ϑ 30°, φ 30°
ID          10        10        10        42
IDB          1         0         2         0
Symbol      12                            -1
```

```
   4  2  2  2  3     4  1  2
   2  6  1     2  3  6  3  1
1  3  3  4  2  3  5  3  3  3
4  4  1  7  2  5  7  5  2  3
5  6  7  6  4  5  8  7  8  3
8  3  6  2  5  5  A  B  4  3
7  7  7  4  9  D  A  8  9  8
6  5  5  7  5  8  6  3  7  9
3  A  9  8  F  5  5  8  5  5
5  5  9  9  8  B  5  6  4  1
1  5  7  9  5  9  7  9  6  7
4  5  3  7  8  5  1  3  1  1
   2  3  2  2  2  2  1  1  3
1  2  2  4  3  1  4  1  1
1     2  3  1  1  1  1  1  1
```

Figure C.4: Demonstration File 4 — Different ways of displaying 2-d histograms.

C.5 Demonstration File 5: Simultaneous fit of 2 plots with function overlays

```
MN_CMD> exec demo05
Reading commands from unit  10
File: CDF$ROOT5:[MN_FIT.PUBLIC.DOC]DEMO05.MNF;1
MN_CMD> !                    ————————————————————————————————
MN_CMD> !                    Mn_Fit Demonstration file number 5.
MN_CMD> !                    Fit 2 plots simultaneously and do some function overlaying
MN_CMD> !                    ————————————————————————————————
MN_CMD> fetch mn_fit_test:hbook4_test.his 0
Reading histograms from unit   1
File: CDF$ROOT5:[MN_FIT.PUBLIC.EXAMPLES]HBOOK4_TEST.HIS;1
The current HBOOK directory is: //MN_HBIN
MN_HBF: A total of    11 plots have been read in
MN_CMD> !
MN_CMD> !                    Add the functions needed for fitting
MN_CMD> !
MN_CMD> fun del 0
MN_CMD> !
MN_CMD> !                    Add a Gaussian in terms of sigma
MN_CMD> !
MN_CMD> fun add gauss sig
Function: Gaussian Distribution (sigma)
Give Initial Values for Parameters (Value, Step Size, Lower Limit, Upper Limit)
1 AREA : 2000 500 !          Area
2 MEAN : 1.9 0.1 !           Mean
3 SIGMA : 0.3 0.1 !          Sigma
MN_CMD> !
MN_CMD> !                    Add a Chebyshev polynomial of second order
MN_CMD> !
MN_CMD> fun add cheb 2
Function: Chebyshev Polynomial of Order 2
Give Initial Values for Parameters (Value, Step Size, Lower Limit, Upper Limit)
1 NORM : 1000 200 !          Normalization
2 CHEB01 : 0.0 100 !         1st order coefficient
3 CHEB02 : 0.0 100 !         2nd order coefficient
MN_CMD> !
MN_CMD> !                    Add a 2nd Chebyshev of first order
MN_CMD> !
MN_CMD> fun add cheb 1
Function: Chebyshev Polynomial of Order 1
Give Initial Values for Parameters (Value, Step Size, Lower Limit, Upper Limit)
1 NORM : 100 50 !            Normalization
2 CHEB01 : 0.0 10 !          1st order coefficient
MN_CMD> !
MN_CMD> !                       Fiddle stuff so area of second plot is twice that of first
MN_CMD> !
MN_CMD> cop 8 8&1
MN_CMD> add 8&1 5 8&1 1 1
MN_CMD> part 8&1 1 5
MN_CMD> cop 7 7&1
MN_CMD> add 7&1 6 7&1 1 1
MN_CMD> part 7&1 0 4
MN_CMD> scale 7&1 7&1 2
MN_CMD> !
MN_CMD> set
SET> Give variable name or ?: default
SET> Give variable name or ?: thick 1.5
SET> Give variable name or ?: thick frame 2
SET> Give variable name or ?: x mode int
SET> Give variable name or ?: y mode int
SET> Give variable name or ?: sym -34
SET> Give variable name or ?: endset
MN_CMD> !
MN_CMD> !                       Fit plot 7&1 not using function 3 and plot 8&1 not using function 2
MN_CMD> !                       First fit background, then signal, then everything.
MN_CMD> !
MN_CMD> fit 7&1 -3, 8&1 -2
Chi**2 (0) or likelihood (1) fit (<CR>=0)? 0
Will do a chi**2 fit to histograms:       7   1;      8   1;
==============================================================================
MINUIT RELEASE 90.10  INITIALIZED.   DIMENSIONS 100/ 50  EPSMAC= 0.56E-16
==============================================================================
**********
**    1 **SET PRINTOUT          0.0000E+00
**********
PARAMETER DEFINITIONS:
NO.   NAME           VALUE       STEP SIZE      LIMITS
1 'AREA       '    2000.0       500.00         no limits
2 'AR1/AREA   '   0.50000      0.10000        0.00000E+00    1.0000
3 'MEAN       '    1.9000      0.10000         no limits
4 'SIGMA      '   0.30000      0.10000         no limits
5 'NORM       '    1000.0       200.00         no limits
6 'NRM1/NORM  '    1.0000      constant
7 'CHEBO1     '   0.00000E+00   100.00         no limits
8 'CHEBO2     '   0.00000E+00   100.00         no limits
9 'NORM       '    100.00      50.000          no limits
10 'NRM1/NORM '    0.00000E+00  constant
11 'CHEBO1    '    0.00000E+00   10.000         no limits
**********
**    2 **CALL FCN               1.000
```

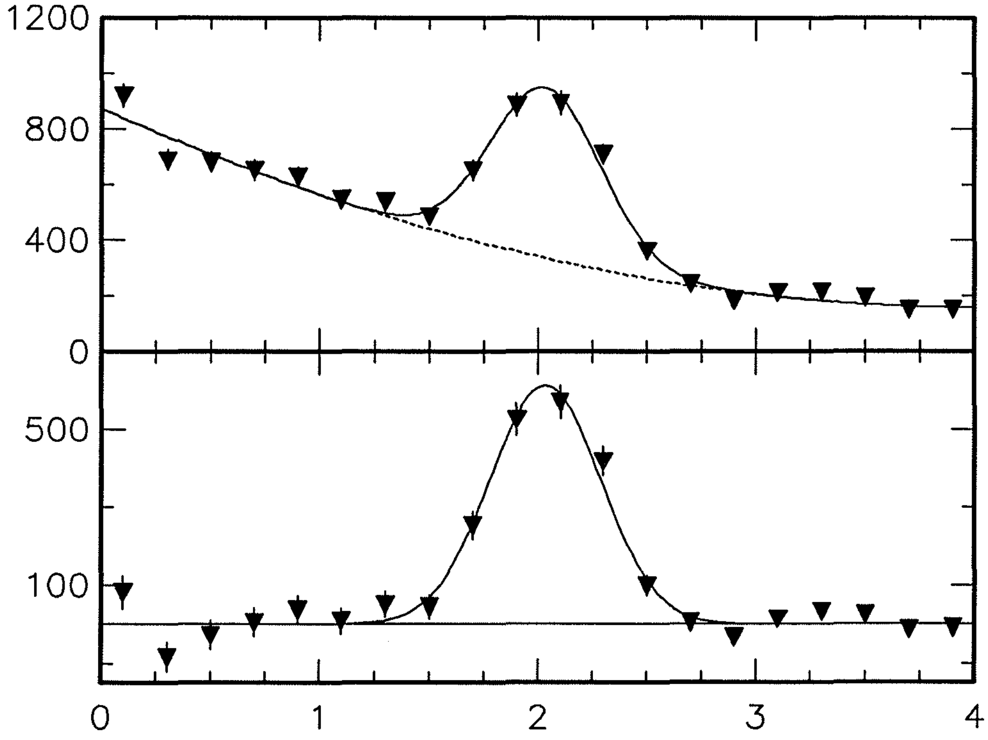
```
**********
FCN=   17684.28     FROM CALl fcn  STATUS=RESET         1 CALLS         1 TOTAL
EDM= unknown      STRATEGY= 1      NO ERROR MATRIX
EXT PARAMETER                 CURRENT GUESS     PHYSICAL LIMITS
NO.   NAME         VALUE          ERROR        NEGATIVE      POSITIVE
1  AREA          2000.0         500.00
2  AR1/AREA     0.50000        0.10000        0.00000E+00    1.0000
3  MEAN          1.9000        0.10000
4  SIGMA        0.30000        0.10000
5  NORM          1000.0         200.00
6  NRM1/NORM     1.0000        constant
7  CHEB01       0.00000E+00     100.00
8  CHEB02       0.00000E+00     100.00
9  NORM          100.00         50.000
10  NRM1/NORM    0.00000E+00    constant
11  CHEB01       0.00000E+00     10.000
MINUIT>
MINUIT> printout -5
MINUIT> modify 2 0.8 !            Modify the ratio of normalizations for signal
MINUIT> !                         Modify the background with the DEPOSIT command
MINUIT> !                         DEPOSIT does not like inline comments
MINUIT> dep p2(1) = 200
p2(1)   =   200.000
MINUIT> dep err2(1) = 100
err2(1) =   100.000
MINUIT> !
MINUIT> fix 1 2 3 4 !             Get the background close first
MINUIT> printout -2
MINUIT> excl 1.5 2.5 0 !          When fitting more than 1 plot give id for exclusions
+++ MINUIT will start from scratch
MINUIT> seek
MINUIT> minimize
MINUIT> set text off !            Turn off the display text for screen device - saves t
MINUIT> display
Results of Fit to Plot(s):       7   1;       8   1;
Chi**2 =      31.7  for  28 -   5 d.o.f.                 C.L. = 10.7 %
Individual chi**2 =    18.5,    13.2,
Hist        7   1
Plot Area Total/Fit  10000.0    /  6010.00               Fit Status  3
Func Area Total/Fit  9559.32    /  6425.49               E.D.M. 1.484E-22
Hist        8   1
Plot Area Total/Fit  2305.00    /  749.000
Func Area Total/Fit  1691.75    /  855.020
Name        Value                      Errors
Parabolic              Minos
Function  1: Gaussian Distribution (sigma)
```

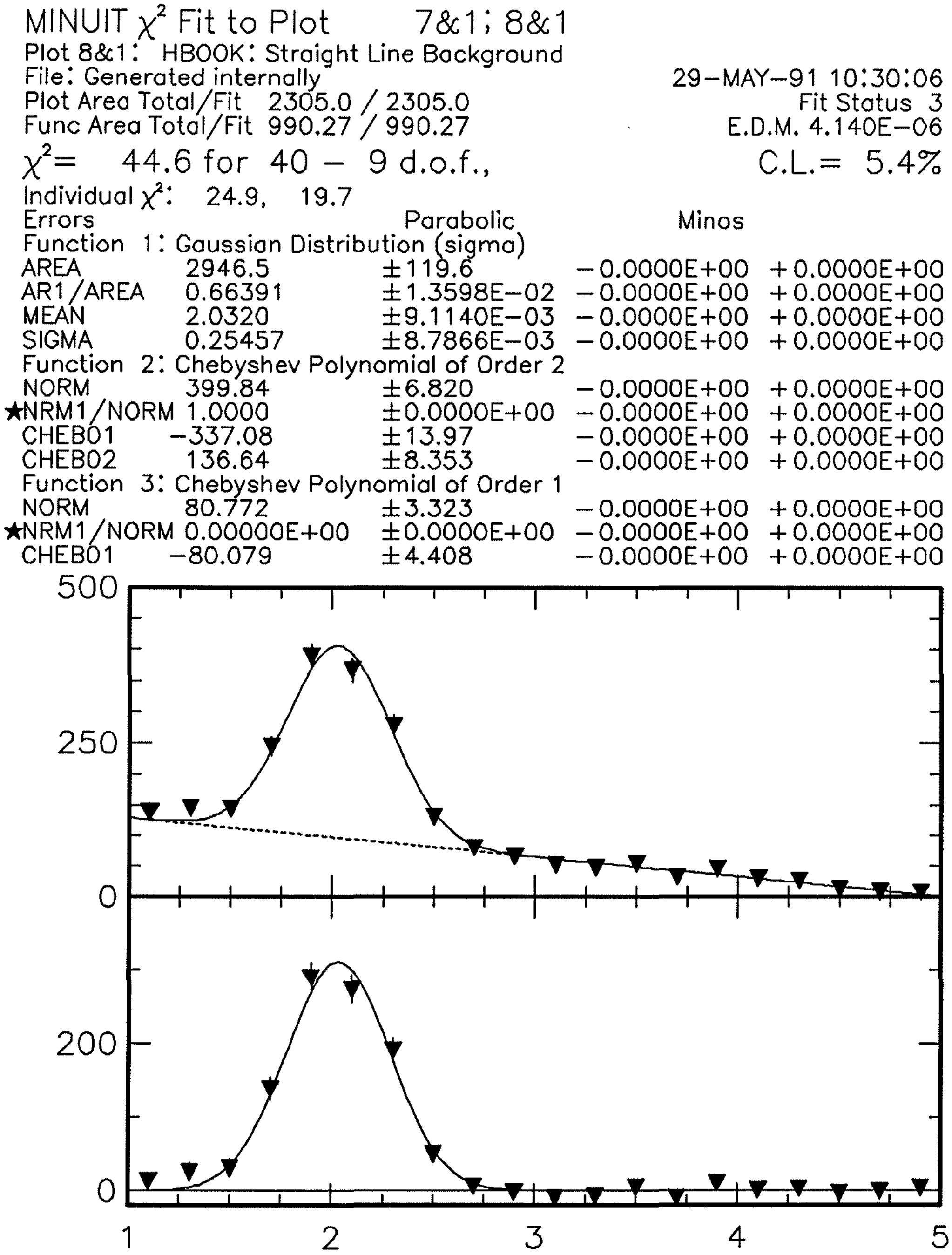
```
* 1( 1)  1 AREA       2000.0     +/- 0.00000E+00 - 0.00000E+00 + 0.00000E+00
* 1( 2)  2 AR1/AREA   0.80000    +/- 0.00000E+00 - 0.00000E+00 + 0.00000E+00
* 1( 3)  3 MEAN        1.9000    +/- 0.00000E+00 - 0.00000E+00 + 0.00000E+00
* 1( 4)  4 SIGMA      0.30000    +/- 0.00000E+00 - 0.00000E+00 + 0.00000E+00
Function  2: Chebyshev Polynomial of Order 2
2( 1)  5 NORM       398.85     +/-  11.726     - 0.00000E+00 + 0.00000E+00
* 2( 2)  6 NRM1/NORM  1.0000     +/- 0.00000E+00 - 0.00000E+00 + 0.00000E+00
2( 3)  7 CHEB01     -333.22    +/-  17.986     - 0.00000E+00 + 0.00000E+00
2( 4)  8 CHEB02     136.88     +/-  19.852     - 0.00000E+00 + 0.00000E+00
Function  3: Chebyshev Polynomial of Order 1
3( 1)  9 NORM       80.537     +/-  3.2921     - 0.00000E+00 + 0.00000E+00
* 3( 2) 10 NRM1/NORM  0.00000E+00 +/- 0.00000E+00 - 0.00000E+00 + 0.00000E+00
3( 3) 11 CHEB01     -79.610    +/-  4.3916     - 0.00000E+00 + 0.00000E+00
MINUIT> float 1 2 3 4
PARAMETER   1, AREA        RESTORED TO VARIABLE.
PARAMETER   2, AR1/AREA    RESTORED TO VARIABLE.
PARAMETER   3, MEAN        RESTORED TO VARIABLE.
PARAMETER   4, SIGMA       RESTORED TO VARIABLE.
MINUIT> !
MINUIT> no_excl 0 !               Remove all exclusions
Exclusions for plot     7   1 removed
Exclusions for plot     8   1 removed
+++ MINUIT will start from scratch
MINUIT> fix 5 7 8 9 11 !          Now get the signal close as well
MINUIT> minimize
MINUIT> display
Results of Fit to Plot(s):       7   1;       8   1;
Chi**2 =     44.7  for  40 -   4 d.o.f.                   C.L. = 15.2 %
Individual chi**2 =     25.1,    19.6,
Hist        7   1
Plot Area Total/Fit  10000.0     /  10000.0               Fit Status  3
Func Area Total/Fit  9927.16     /  9927.16               E.D.M. 3.746E-07
Hist        8   1
Plot Area Total/Fit  2305.00     /  2305.00
Func Area Total/Fit  2285.23     /  2285.23
Name        Value                   Errors
Parabolic              Minos
Function  1: Gaussian Distribution (sigma)
1( 1)  1 AREA       2960.8     +/-  99.390     - 0.00000E+00 + 0.00000E+00
1( 2)  2 AR1/AREA   0.66463    +/- 1.30703E-02 - 0.00000E+00 + 0.00000E+00
1( 3)  3 MEAN       2.0317     +/- 9.47151E-03 - 0.00000E+00 + 0.00000E+00
1( 4)  4 SIGMA      0.25529    +/- 8.05873E-03 - 0.00000E+00 + 0.00000E+00
Function  2: Chebyshev Polynomial of Order 2
* 2( 1)  5 NORM       398.85     +/- 0.00000E+00 - 0.00000E+00 + 0.00000E+00
* 2( 2)  6 NRM1/NORM  1.0000     +/- 0.00000E+00 - 0.00000E+00 + 0.00000E+00
* 2( 3)  7 CHEB01     -333.22    +/- 0.00000E+00 - 0.00000E+00 + 0.00000E+00
* 2( 4)  8 CHEB02      136.88     +/- 0.00000E+00 - 0.00000E+00 + 0.00000E+00
Function  3: Chebyshev Polynomial of Order 1
* 3( 1)  9 NORM        80.537     +/- 0.00000E+00 - 0.00000E+00 + 0.00000E+00
* 3( 2) 10 NRM1/NORM  0.00000E+00 +/- 0.00000E+00 - 0.00000E+00 + 0.00000E+00
* 3( 3) 11 CHEB01     -79.610     +/- 0.00000E+00 - 0.00000E+00 + 0.00000E+00
MINUIT> float 5 7 8 9 11
PARAMETER   5, NORM       RESTORED TO VARIABLE.
PARAMETER   7, CHEB01     RESTORED TO VARIABLE.
PARAMETER   8, CHEB02     RESTORED TO VARIABLE.
PARAMETER   9, NORM       RESTORED TO VARIABLE.
PARAMETER  11, CHEB01     RESTORED TO VARIABLE.
MINUIT> !
MINUIT> !                           Now fit everything and show the fit and background subtracted fit
MINUIT> !
MINUIT> minimize
MINUIT> dump
Histogram        7    1  Number of points    20
Pnt     X    +/-   DX          Y   +/-   DY         Yfit       Chi**2 Total
1 0.1000    0.100         920.0     42.9         838.7        3.596 3.596
2 0.3000    0.100         688.0     37.1         771.5       -5.063 8.659
3 0.5000    0.100         680.0     36.9         707.8       -0.567 9.226
4 0.7000    0.100         652.0     36.1         647.6        0.015 9.241
5 0.9000    0.100         628.0     35.4         590.9        1.096 10.336
6  1.100    0.100         548.0     33.1         538.4        0.084 10.420
7  1.300    0.100         538.0     32.8         497.8        1.503 11.923
8  1.500    0.100         486.0     31.2         510.8       -0.633 12.556
9  1.700    0.100         652.0     36.1         660.9       -0.061 12.617
10  1.900    0.100         886.0     42.1         895.8       -0.054 12.671
11  2.100    0.100         894.0     42.3         915.8       -0.265 12.937
12  2.300    0.100         710.0     37.7         644.3        3.043 15.980
13  2.500    0.100         362.0     26.9         376.4       -0.286 16.266
14  2.700    0.100         244.0     22.1         257.6       -0.379 16.645
15  2.900    0.100         184.0     19.2         218.1       -3.159 19.804
16  3.100    0.100         212.0     20.6         198.1        0.453 20.258
17  3.300    0.100         216.0     20.8         183.3        2.472 22.730
18  3.500    0.100         196.0     19.8         172.1        1.458 24.188
19  3.700    0.100         152.0     17.4         164.4       -0.503 24.691
20  3.900    0.100         152.0     17.4         160.1       -0.218 24.909
Histogram        8    1  Number of points    20
Pnt     X    +/-   DX          Y   +/-   DY         Yfit       Chi**2 Total
1  1.100    0.100         139.0     11.8         126.0        1.208 26.116
2  1.300    0.100         145.0     12.0         124.2        2.990 29.106
3  1.500    0.100         144.0     12.0         147.8       -0.098 29.204
4  1.700    0.100         245.0     15.7         239.0        0.148 29.352
5  1.900    0.100         390.0     19.7         371.3        0.896 30.248
6  2.100    0.100         368.0     19.2         393.1       -1.712 31.960
```

```
7  2.300    0.100          278.0     16.7         265.5         0.558 32.518
8  2.500    0.100          131.0     11.4         138.1        -0.382 32.900
9  2.700    0.100          80.00     8.94         84.30        -0.231 33.131
10  2.900    0.100          66.00     8.19         68.89        -0.124 33.255
11  3.100    0.100          52.00     7.28         61.60        -1.739 34.994
12  3.300    0.100          48.00     7.00         55.15        -1.043 36.037
13  3.500    0.100          54.00     7.42         48.74         0.503 36.540
14  3.700    0.100          33.00     5.83         42.33        -2.563 39.102
15  3.900    0.100          46.00     6.86         35.93         2.158 41.261
16  4.100    0.100          30.00     5.57         29.52         0.007 41.268
17  4.300    0.100          26.00     5.20         23.12         0.308 41.576
18  4.500    0.100          13.00     3.74         16.71        -0.983 42.559
19  4.700    0.100          9.000     3.16         10.30        -0.170 42.729
20  4.900    0.100          8.000     3.00         3.897         1.871 44.600
MINUIT> set text on !            Turn text back on for final result
MINUIT> set back 2 3
MINUIT> set display mode 3
MINUIT> display &11 &11 !        Must give secondary id's for background subtracted p
Results of Fit to Plot(s):       7   1;        8   1;
Chi**2 =     44.6  for  40 -   9 d.o.f.                  C.L. = 5.41 %
Individual chi**2 =    24.9,    19.7,
Hist        7   1
Plot Area Total/Fit  10000.0     /  10000.0              Fit Status  3
Func Area Total/Fit  9953.19     /  9953.19              E.D.M. 4.140E-06
Hist        8   1
Plot Area Total/Fit  2305.00     /  2305.00
Func Area Total/Fit  2285.40     /  2285.40
Name        Value                   Errors
Parabolic              Minos
Function  1: Gaussian Distribution (sigma)
1( 1)  1 AREA        2946.5      +/-  119.57     - 0.00000E+00 + 0.00000E+00
1( 2)  2 AR1/AREA   0.66391      +/- 1.35983E-02 - 0.00000E+00 + 0.00000E+00
1( 3)  3 MEAN        2.0320      +/- 9.11404E-03 - 0.00000E+00 + 0.00000E+00
1( 4)  4 SIGMA      0.25457      +/- 8.78662E-03 - 0.00000E+00 + 0.00000E+00
Function  2: Chebyshev Polynomial of Order 2
2( 1)  5 NORM        399.84      +/-  6.8202     - 0.00000E+00 + 0.00000E+00
* 2( 2)  6 NRM1/NORM   1.0000      +/- 0.00000E+00 - 0.00000E+00 + 0.00000E+00
2( 3)  7 CHEB01     -337.08      +/-  13.965     - 0.00000E+00 + 0.00000E+00
2( 4)  8 CHEB02      136.64      +/-  8.3531     - 0.00000E+00 + 0.00000E+00
Function  3: Chebyshev Polynomial of Order 1
3( 1)  9 NORM        80.772      +/-  3.3229     - 0.00000E+00 + 0.00000E+00
* 3( 2) 10 NRM1/NORM  0.00000E+00 +/- 0.00000E+00 - 0.00000E+00 + 0.00000E+00
3( 3) 11 CHEB01     -80.079      +/-  4.4077     - 0.00000E+00 + 0.00000E+00
Plot       7   1:
Background subtracted plot will be stored as plot      7  11
Background function will be stored as plot             7 983
```

```
Plot       8   1:
Background subtracted plot will be stored as plot      8  11
Background function will be stored as plot             8 984
MINUIT> !
MINUIT> !                        Overlay the background
MINUIT> !
MINUIT> fun over 2 &12 -2
Plot       7   1:  Function will be stored as plot      7  12
Plot       8   1:  Function will be stored as plot      8  12
MINUIT> fun over 3 &13 -2
Plot       7   1:  Function will be stored as plot      7  13
Plot       8   1:  Function will be stored as plot      8  13
MINUIT> !
MINUIT> !                        Note that are 2 pages to this picture.
MINUIT> !
MINUIT> exec demohard 05 !       Make a hardcopy file
Reading commands from unit  11
File: CDF$ROOT5:[MN_FIT.PUBLIC.DOC]DEMOHARD.MNF;1
MINUIT> !
MINUIT> !                        Macro to make a QMS hardcopy
MINUIT> !
MINUIT> inquire 1 'Give demonstration number'
MINUIT> set hard demo05.qms !    Set the hardcopy filename
MINUIT> hardcopy TLP !           Make a Talaris hardcopy
TVCAP: Hardcopy to unit 12, File: demo05.qms
TLP selected
None selected
MINUIT> close !                  Close the hardcopy file
End of file reached, unit=  11. Exiting
MINUIT> !
MINUIT> exit !                   Exit from MINUIT
WARNING. Value of parameter AR1/AREA    Function   1 will be lost
WARNING. Value of parameter NRM1/NORM   Function   2 will be lost
WARNING. Value of parameter NRM1/NORM   Function   3 will be lost
End of file reached, unit=  10. Exiting
```

```
MINUIT χ² Fit to Plot        7&1; 8&1
Plot 7&1:  HBOOK: Gaussian Signal + Exponential Background
File: Generated internally                         29-MAY-91 10:29:55
Plot Area Total/Fit  10000. / 10000.                     Fit Status  3
Func Area Total/Fit  1956.2 / 1956.2                  E.D.M. 4.140E-06
χ²=    44.6 for  40 -  9 d.o.f.,                      C.L.=  5.4%
Individual χ²:  24.9,   19.7
Errors                       Parabolic                 Minos
Function  1: Gaussian Distribution (sigma)
AREA         2946.5         ±119.6       -0.0000E+00  +0.0000E+00
AR1/AREA     0.66391        ±1.3598E-02  -0.0000E+00  +0.0000E+00
MEAN         2.0320         ±9.1140E-03  -0.0000E+00  +0.0000E+00
SIGMA        0.25457        ±8.7866E-03  -0.0000E+00  +0.0000E+00
Function  2: Chebyshev Polynomial of Order 2
NORM         399.84         ±6.820       -0.0000E+00  +0.0000E+00
★NRM1/NORM   1.0000         ±0.0000E+00  -0.0000E+00  +0.0000E+00
CHEB01      -337.08         ±13.97       -0.0000E+00  +0.0000E+00
CHEB02       136.64         ±8.353       -0.0000E+00  +0.0000E+00
Function  3: Chebyshev Polynomial of Order 1
NORM         80.772         ±3.323       -0.0000E+00  +0.0000E+00
★NRM1/NORM   0.00000E+00    ±0.0000E+00  -0.0000E+00  +0.0000E+00
CHEB01      -80.079         ±4.408       -0.0000E+00  +0.0000E+00
```





```
MINUIT χ² Fit to Plot       7&1; 8&1
Plot 8&1:  HBOOK: Straight Line Background
File: Generated internally                                  29-MAY-91 10:30:06
Plot Area Total/Fit  2305.0 / 2305.0                            Fit Status 3
Func Area Total/Fit  990.27 / 990.27                        E.D.M. 4.140E-06
χ²=   44.6 for 40 -  9 d.o.f.,                              C.L.=  5.4%
Individual χ²:   24.9,   19.7
Errors                        Parabolic                Minos
Function  1: Gaussian Distribution (sigma)
AREA        2946.5            ±119.6         -0.0000E+00  +0.0000E+00
AR1/AREA    0.66391           ±1.3598E-02    -0.0000E+00  +0.0000E+00
MEAN        2.0320            ±9.1140E-03    -0.0000E+00  +0.0000E+00
SIGMA       0.25457           ±8.7866E-03    -0.0000E+00  +0.0000E+00
Function  2: Chebyshev Polynomial of Order 2
NORM        399.84            ±6.820         -0.0000E+00  +0.0000E+00
★NRM1/NORM  1.0000            ±0.0000E+00    -0.0000E+00  +0.0000E+00
CHEB01     -337.08            ±13.97         -0.0000E+00  +0.0000E+00
CHEB02      136.64            ±8.353         -0.0000E+00  +0.0000E+00
Function  3: Chebyshev Polynomial of Order 1
NORM        80.772            ±3.323         -0.0000E+00  +0.0000E+00
★NRM1/NORM  0.00000E+00       ±0.0000E+00    -0.0000E+00  +0.0000E+00
CHEB01     -80.079            ±4.408         -0.0000E+00  +0.0000E+00
```

Figure C.5: Demonstration File 5 — Simultaneous fit of 2 plots with function overlays.

Appendix D

List of Changes

The following subtopics list the changes made for the version indicated. The top name Vn_mm_ll corresponds to the CMZ version number Vn.mm/ll which is given when you start up Mn_Fit. Note that all changes before 01/12/89 are under topic V2_00_00.

D.1 Version 3.01/03 25/03/91

The following version numbers apply to this version of Mn_Fit:

```
Mn_Fit  3.01/03
Mn_Util 1.02/05
MINUIT  1.00/09 (C. Rippich),  90.10 or 89.12j (CERN)
TYPSCN  1.00/12
L3_DSP  1.00/06
```

This section lists the changes made since Mn_Fit version 2.05/13.

Known Bugs or Problems:

- You should not try to rename histograms which have the same primary identifier as an Ntuple which is stored with the disk option.
- The user function number sometimes does not get passed properly to COMIS functions. Be careful using it.
- If you get an error when adding a COMIS function, you get several questions asking you whether you want to abort this command.
- If you get an error in some `SET` commands, you again get several questions asking you whether you want to abort this command.
- Not all `SET` commands have a `SHOW` equivalent. Therefore sometimes if you give `SHOW` **`parameter`** you will get into `SET`. Just hit <CR> to exit.
- SET X|Y LIMITS do not work for lego/surface plots - use PARTITION.
- Sometimes you only get one big tick on an axis.

New Features:

- The latest CERN version of MINUIT has been added (89.12j). This seems to be significantly better than the old version. The old version is still available, but must be compiled at your site. NOTE: You should check that you get the same or better results from fitting compared with the previous MINUIT version. This is the first release using the CERN MINUIT.

- New possibilities in fitting include constraints on parameters, contour plots and graphical contour plots. It is also possible to integrate functions across each bin to improve precision, as well as including and excluding regions from the fit (see section 4.40 on page 65 (FIT) and section 4.79.19 on page 97 (SET FIT) for details). The DISPLAY and FIT_INFO output has been modified to give the MINUIT and Mn_Fit parameter numbers and also to indicate which parameters are constrained (a # sign).

- There has been a lot of development and new commands for Ntuple manipulation. In addition the `CUT` mechanism has been substantially improved and cuts can now include arithmetic expressions and COMIS functions. You can use >,< etc. instead of `.GT.`,`.LT.` etc. in cuts and `&` and | instead of `.AND.` and `.OR.` in `CUT USE`. It is also possible to project into arithmetic expressions. Ntuples can be merged. It is possible to plot projections of Ntuples directly and to use automatic binning for the projections. You can also specify if an Ntuple variable should be used as the weight or error on the weight. See section 4.66 on page 86 (NTUPLE) for full details.

- If you store tables in Ntuples, you can plot them directly using the PLOT/NTUPLE command.

- It is possible to define user variables and use them in arithmetic expressions.

- It is possible to define aliases for character strings.

- The COMIS interface has been expanded to include cuts, scanning of Ntuples and calling a COMIS subroutine. Information on histograms and Ntuples is now available inside the functions or subroutine and skeleton subroutines get made automatically if the filename does not exist. All the Cernlib routines that are available inside PAW are now also available, plus Mn_Fit functions and some extra useful Cernlib routines (see section 1.13 on page 25 (USING_COMIS) for full details).

- You can now access the overall normalization, Ntuple contents and 2-D histogram contents when you are allowed to use registers, parameters etc. You can also `DEPOSIT` into the same places. Deposit on parameters when fitting now updates the MINUIT parameter values also, so this is an alternative to the `MODIFY` command (see section 1.6 on page 17 (Numbers) and section 4.27 on page 56 (DEPOSIT) for details). You can also use the MINUIT parameter number rather than the Mn_Fit ones when you are fitting.

- Default symbols now depend on the type of plot being made and new symbols for scatter plots are available.

- This is the first release that almost works fully on the Apollo. However `SMOOTH` and `SPLINE` do not work on the Apollo at present.

- New command `SET FONT` which makes changing fonts much easier.
- New command MDIRECTORY to make a new HBOOK directory in memory. This is most useful in connection with STORE.
- You can now store histograms in an existing HBOOK RZ file, `STORE/UPDATE`. In addition you can store histograms in directories other than the top level using the CDIR and MDIR commands.
- Automatic logging of all commands typed is available (`SET LOG`).
- A debug flag is available in common for COMIS and user functions and can be set using the command `SET DEBUG`.
- Device `None` has been added. This enables Mn_Fit to go through the motions of making a picture without actually doing so. Useful to do `FUNCTION HIST` for example if you do not want to look at the plot.
- Landscape and Encapsulated Postscript devices are available for most graphics packages.
- New line modes 5,6,7,8 corresponding to HIGZ line modes 12,13,14,15 added.
- When you get asked to hit <CR> for the next plot you can type `q` to quit the macro altogether. Typing any other character will skip that plot command.
- You can now use `?` instead of `HELP` to get help. Indeed when you want help in the middle of a command (CUT for example), you must use `?` instead of `Help`. This is indicated in the prompt.

New Commands:

?	Replacement or alternative for HELP.
ALIAS	Defines an alias for a string.
UNALIAS	Undefines one or all aliases.
NTUPLE MERGE	Merges several Ntuples into one.
NTUPLE PLOT	Projects and plot an Ntuple or n-dimensional histogram.
NTUPLE PROJECT	Projects an Ntuple or n-dimensional histogram.
NTUPLE SCAN	Calls a subroutine for every Ntuple event.
CUT FILE	Defines a COMIS function as a cut.
CUT COMPILE	Compiles a COMIS function which is defined as a cut.
CUT EDIT	Edits and compiles a COMIS cut function.
STORE/NEW	Stores plots in a new HBOOK RZ file.
STORE/UPDATE	Stores plots in an existing HBOOK RZ file.
HB_MN_FIT	Converts HBOOK to Mn_Fit histograms.
REMOVE	Deletes a user variable.

CALL_COMIS	Calls a COMIS subroutine.
MDIRECTORY	Makes a new HBOOK directory in memory.
SET ALIAS	Turns on or off alias translation.
SET DEBUG	Turns on or off the debug flag for COMIS and user functions.
SET FIT	Sets parameters for fitting.
SET FONT	Changes the fonts for various parts of the picture.
SET LOG	Turns on or off automatic logging of commands.
SET NTUPLE	Specifies the Ntuple variables to plot with PLOT/NTUPLE.
SET SHELL	Specifies the shell command on the Apollo.
SET STATISTICS	Specifies whether to use HBOOK or Mn_Fit means and sigmas.
SHOW ALIAS	Shows the definitions of one or all aliases.
SHOW FIT	Shows flags and parameters used when fitting.
SHOW CONSTRAIN	Shows the current constraints on fit parameters.
SHOW VARIABLE	Shows the values of one or all user variables.
INQUIRE	Gets the value of a macro parameter.
CONSTRAIN	Constrains a parameter being fitted.
UNCONSTRAIN	Removes a constraint on a parameter.
INCLUDE	Include a region in a fit.
SCAN	Scan the fitting function with one or all parameters.
MNCONTOUR	Make a graphical contour.
MAX_CALLS	Replaces the CALLS command.
HELP EXPRESSION	New help on arithmetic expressions.
HELP USING_COMIS	New help on COMIS interface to Mn_Fit.

Modified Commands:

DEPOSIT	Can create user variables and deposit into Ntuples and 2-d histograms.
UNDEFINE	Can now undefine all defined commands.
PROJECT	Alias for NTUPLE PROJECT - many new possibilities.
CUT NEW	Only 1 cut per line is now allowed.
CUT CHANGE	Keeping current condition now needs ==.
CUT USE	Allow & and | and COMIS function cuts.
SET COLOUR REPRESENT	Changes the colour representation.
SET FONT	Can now be used to set fonts for all parts of a picture. Precision can also be given - this means font numbers are not necessarily backwards compatible.
SET HATCH	Portable HIGZ hatches are available.

SET SYMBOL Symbol 0 is plot type dependent. New symbols for scatter plots.

SHOW REGISTER Can now show a range or all user registers.

CALLS Renamed to `MAX_CALLS`.

FIT_INFO Shows the Mn_Fit and MINUIT parameter numbers. Indicates which parameters are constrained.

DISPLAY Same changes as in FIT_INFO.

HARDCOPY TEKFILE Replaces HARDCOPY TEKTRONIX.

HELP MINUIT Substantially updated.

Improvements and Bug Fixes:

- `CUT NEW` can only define one cut per line. `CUT CHANGE` needs == on the condition instead of = to keep the current condition. It is not possible to use the form =`*5` to change the value of a cut. Cuts are now parsed when they are used, not when they are defined. This means that if a cut includes a register for example, the value of the register at the time the cut is used will be taken.

- `SET SYMBOL 0` now defines a default symbol which depends on the plot type. Symbols -1 -> -4 exist for scatter plots and join the dots using lines 1 -> 4.

- HBOOK statistics rather than Mn_Fit ones are now available on request (`SET STATISTICS HBOOK|MN_FIT`).

- `CUT, COMMENT, SET` and `DRAW` exit after the completion of the first command if you include something on the line with the command. If you do not you must give an `END` (or <`CR`> when not in a macro or defined command). This behaviour should now be consistent. It was not always before.

- On the VAX the edit commands `EDIT/TPU` and `EDF` now use CALLABLE_TPU instead of spawning a subprocess.

- Background subtraction works properly for variable sized binning plots.

- 2-D functions were not declared `DOUBLE PRECISION`.

- If you are plotting with error bars and the point, but not the point +/- error is outside the plot, the error bar will now be drawn.

- The command sequence `FETCH filename id; CDIR dirname; FETCH id` did not work. That is now fixed.

- The Vax condition handler has been improved. There is no condition handler on the Apollo at present.

- `SHOW ORDER` and `SHOW ORTHOGONAL` have been fixed.

- `SET Z OPT_ZERO` now works.

- The y axis label is now at the correct angle for lego/surface plots. The offset of lego/surface plots from the bottom left corner of a plot has been reduced to 0, so the scales are now in the margins as they are for all other sorts of plots.

- PARTITION Only worked properly for monotonically increasing x values. It now works for monotonically increasing or decreasing values.

- Tektronix devices now work for GKSGRAL if they are the same device. They also now work for DECGKS and DEC GKS3D.

- Registers, parameters etc. can now be used for overlay symbols hatches and patterns and also for lego/surface plot angles. They can also be used for secondary identifiers in PROJECT etc.

- If the prompt indicates you can get extra help when giving a command (terminates in or ?:), this should now always work.

- The direct access file length is checked using RZFILE for HBOOK histogram files. This should avoid Mn_Fit bombing if the record length is wrong.

- Occasionally _ was no longer recognized in commands. This has been fixed.

- You have to type at least EXI to exit otherwise the command gets passed onto the standard command list without further checking.

- All code and files needed for linking, running and testing Mn_Fit as well as producing the manual is now in 4 CMZ files (5 for L3). This makes keeping up-to-date much easier and will enable me to produce updates to the manual. The HELP Changes has been modified somewhat and is now keyed on version number rather than date. All institutions should now be able to produce a working version of Mn_Fit directly from these files without introducing any local fixes except to define the necessary logical names or links. See the deck //mn_util/docs/code for details.

- The number of entries and underflows and overflows were not being fetched properly for M_BOOK histograms.

- There were a couple of bugs if you switched many times between hardcopy and screen devices. These have hopefully been fixed.

- The SET X|Y|Z OPT_ZERO command also changed whether the fit function was shown on excluded regions of a display.

- Ntuples were not read in properly in DAT_FETCH. There was a format error in one of the error messages in DAT_FETCH.

- Bug fixed in recovery from failure to open a hardcopy file.

- Empty command lines are now echoed properly.

- Number of plots fetched when secondary identifiers were used in DAT_FETCH was wrong.

- The news system now reads the file **`mn_fit_dir:mn_news.fil`** which is made from the deck //mn_util/news/news every time a final version of Mn_Fit is linked.
- The help library has been renamed to mn_fit.hlb.
- The demonstration and manual files (including symbol.mnf) have been moved to the mn_fit_help: directory.
- The flag **$GTSGRAL** has been renamed to **$GKSGRAL** in all the code.