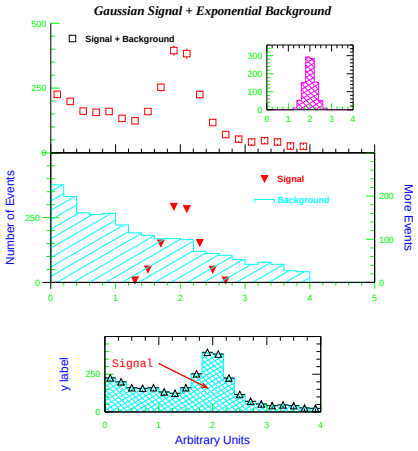
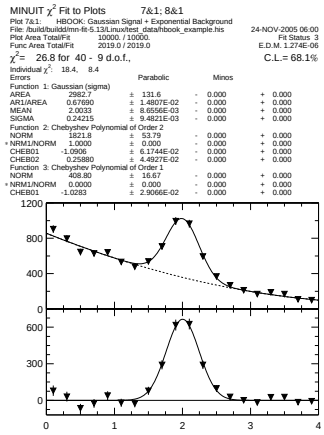




Mn_Fit

A Fitting and Plotting Package Using MINUIT

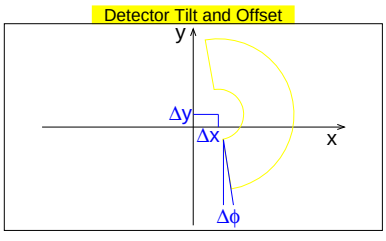
Version 5_13
November 24, 2005

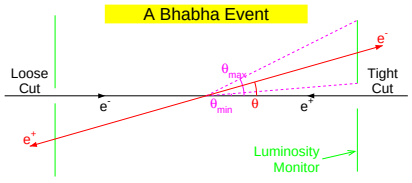
BONN-MS-99-02

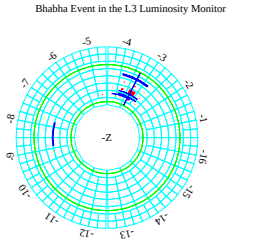
Ian C. Brock

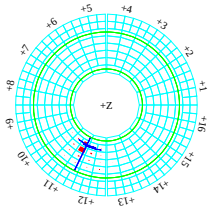
Physikalisches Institut der Universität Bonn
Nußallee 12, D-53115 Bonn

EMail: Ian.Brock@cern.ch or brock@physik.uni-bonn.de
Homepage: http://www-zeus.physik.uni-bonn.de/~brock/mn_fit.html









Preliminary Remarks

In this manual `commands` that you type are in a typewriter font, optional elements of a command are enclosed in [ ]. In the index the main reference to a command is indicated in **boldface**.

The figures on the cover page are made using the macros given in Appendix C.
Mn_Fit can be obtained via my homepage in Bonn:

`http://www-zeus.physik.uni-bonn.de/\til{}brock/mn\_fit.html`

The CERN page that used to be the default, now switches you to the Bonn page.

For several types of Unix machine, you should be able to download the Mn_Fit executable and the other files that are necessary to run Mn_Fit directly. Copy the tar file to a temporary directory and unpack it. You can then install the files by changing your directory to the top-level Mn_Fit directory and giving the command `make install`. Adjust the variables `libdir` and `bindir` to suit your system.

If you cannot find a ready-made executable or have problems with it, you can make a Mn_Fit executable and all other necessary files by downloading the source tar file and unpacking it into a Mn_Fit directory tree. In the top-level Mn_Fit directory check that the `Makefile` is OK for your system and then give the commands:

```
make all
make install [libdir=$HOME/mn_fit] [bindir=$HOME/bin]
```

If you want to install to the default directories you will need root privileges. The file `INSTALL` as well as the more specific files `INSTALL.unix` and `INSTALL.vms` contain information that may be useful for trying to install Mn_Fit on different machines. `INSTALL` also has information on how to include the ROOT interface (`make all_root`).If you still have problems or need to modify the installation scripts contact the author to see if the modifications can be included in the next release.

The manual is in the form of a Postscript and PDF files, which you should be able to print on most Postscript printers and/or look at using ghostview/gv or acroread. There are two versions of the manual, `mn_fit.ps/pdf` which is 1 side per page and `mn_fit_2ppp.ps/pdf` which is 2 sides per page, saving some trees, but requiring somewhat better eyesight! The file `mn_fit_figs.ps/pdf` contains the figures from Appendix B, which show the Mn_Fit symbols, hatches, patterns and examples of fonts. `mn_fit_cover.ps/pdf` contains the Mn_Fit cover page in colour! Some printers do not work with Postscript fonts -14 onwards. To test this print `mn_fit_font.ps` first – 2 pages should be printed. If this works OK, then you can print the normal files. If there are problems, I can create a version without the extra fonts. The manual should fit on both A4 paper and American 8.5x11 in. Please let me know if it does not. An HTML version of the manual is also available via the Mn_Fit homepage.

Contents

1 Introduction **1**

1.1 What Mn_Fit Can Do . . . 1
1.2 How To Run Mn_Fit . . . 2
1.3 Getting Started . . . 4
1.4 Giving Commands . . . 6
1.5 Fitting With Mn_Fit . . . 6
1.5.1 MINUIT Errors . . . 8
1.5.2 Fitting With Small Numbers of Events . . . 9
1.5.3 Including Monte Carlo Statistical Errors . . . 10
1.6 Command Files or Macros . . . 10
1.7 Numbers in Mn_Fit . . . 11
1.8 Expressions . . . 14
1.8.1 Examples . . . 15
1.9 Identifiers . . . 16
1.10 Secondary Identifiers . . . 16
1.10.1 Examples . . . 16
1.11 Symbols . . . 17
1.12 Text . . . 18
1.12.1 Examples . . . 20
1.12.2 Fonts . . . 21
1.12.3 IGTEXT . . . 21
1.13 Mouse . . . 23
1.14 Using Ntuples . . . 23
1.15 ColumnWise Ntuples . . . 24
1.16 Using COMIS . . . 24
1.17 Time . . . 26
1.17.1 Examples . . . 27
1.18 Y2K . . . 27
1.19 Root . . . 28
1.20 Vectors . . . 28
1.20.1 Examples . . . 28
1.21 Examples . . . 29
1.22 Screen Devices . . . 30
1.23 Hardcopy Devices . . . 31
1.24 Condition Handler . . . 32

2 Menu of all the Available Commands **33**
2.1 Menu of Top Level Commands . . . 33
2.2 Menu of Commands inside CUT . . . 37
2.3 Menu of Commands inside DRAW . . . 37
2.4 Menu of Commands inside FUNCTION . . . 38
2.5 Menu of Commands inside HISTOGRAM . . . 38
2.6 Menu of Commands inside NTUPLE . . . 39
2.7 Menu of Commands inside READ . . . 39
2.8 Menu of Commands inside SET . . . 39
2.9 Menu of Commands inside SHOW . . . 42
2.10 Menu of Commands inside WRITE . . . 43

3 MINUIT **44**
3.1 Introduction . . . 44
3.2 Menu of MINUIT Commands . . . 44

4 Reference Manual for Mn_Fit **47**
4.1 EXIT . . . 47
4.2 QUIT . . . 47
4.3 2DIM . . . 47
4.3.1 ARROW . . . 48
4.3.2 BOX . . . 48
4.3.3 CHAR . . . 49
4.3.4 COLOUR . . . 49
4.3.5 CONTOUR . . . 49
4.3.6 LEGO . . . 50
4.3.7 SCATTER . . . 50
4.3.8 SURFACE . . . 50
4.3.9 TEXT . . . 51
4.3.10 Examples . . . 51
4.4 ADD . . . 53
4.5 ALIAS . . . 54
4.5.1 Examples . . . 54
4.6 ATTACH . . . 55
4.7 AVE_FETCH . . . 55
4.8 AVERAGE . . . 55
4.9 BOOK . . . 55
4.10 CALCULATE . . . 56
4.11 CALL_COMIS . . . 56
4.12 CAPTURE . . . 56
4.13 CDIRECTORY . . . 56
4.14 CLEAR . . . 57
4.15 CLOSE . . . 57
4.16 COMIS . . . 57
4.17 COMMENT . . . 57
4.18 COPY . . . 58
4.19 CUT . . . 59
4.19.1 NEW . . . 59
4.19.2 NAME . . . 60

4.19.3 CHANGE . . . 60
4.19.4 FILE . . . 61
4.19.5 EDIT . . . 61
4.19.6 COMPILE . . . 61
4.19.7 DELETE . . . 61
4.19.8 LIST . . . 61
4.19.9 USE . . . 62
4.19.10 END . . . 62
4.19.11 Examples . . . 62
4.20 CWN . . . 63
4.21 DAT_FETCH . . . 63
4.21.1 Examples . . . 64
4.22 DAT_STORE . . . 65
4.23 DATABASE . . . 65
4.23.1 DB_HISTORY . . . 65
4.23.2 DB_SNAP . . . 66
4.24 DB_HISTORY . . . 67
4.25 DB_SNAP . . . 67
4.26 DEFINE . . . 68
4.26.1 Examples . . . 68
4.27 DELETE . . . 69
4.28 DEPOSIT . . . 69
4.28.1 Examples . . . 71
4.29 DIRECTORY . . . 72
4.30 DISPLAY . . . 72
4.31 DIVIDE . . . 72
4.32 DO . . . 72
4.32.1 Examples . . . 73
4.33 DRAW . . . 73
4.33.1 ARC . . . 74
4.33.2 ARROW . . . 75
4.33.3 BOX . . . 75
4.33.4 CIRCLE . . . 75
4.33.5 ELLIPSE . . . 76
4.33.6 GLUON . . . 76
4.33.7 LINE . . . 77
4.33.8 POLYGON . . . 77
4.33.9 POLYLINE . . . 77
4.33.10 SEGMENT . . . 78
4.33.11 SINE . . . 78
4.33.12 SYMBOL . . . 78
4.33.13 TRIANGLE . . . 78
4.33.14 CHANGE . . . 79
4.33.15 DELETE . . . 79
4.33.16 FETCH . . . 79
4.33.17 LIST . . . 79
4.33.18 STORE . . . 79
4.33.19 END . . . 79
4.33.20 Examples . . . 80

4.34 DUMP . . . 80
4.35 EDIT . . . 81
4.36 EFFICIENCY . . . 81
4.37 ELIF . . . 81
4.38 ELSE . . . 82
4.39 ENDDO . . . 82
4.40 ENDIF . . . 82
4.41 EXAMINE . . . 82
4.42 EXECUTE . . . 82
4.42.1 Examples . . . 83
4.43 EXTRACT . . . 84
4.44 FETCH . . . 84
4.44.1 Examples . . . 85
4.45 FI . . . 86
4.46 FILL . . . 86
4.47 FIT . . . 87
4.47.1 Likelihood . . . 88
4.47.2 Examples . . . 89
4.48 FUNCTION . . . 89
4.48.1 ADD . . . 89
4.48.2 COMPILE . . . 90
4.48.3 EDIT . . . 90
4.48.4 DELETE . . . 90
4.48.5 FETCH . . . 90
4.48.6 HISTOGRAM . . . 91
4.48.7 HOVERLAY . . . 91
4.48.8 INFO . . . 91
4.48.9 LIST . . . 91
4.48.10 OVERLAY . . . 107
4.48.11 PLOT . . . 108
4.48.12 STORE . . . 108
4.48.13 USE . . . 108
4.49 HARDCOPY . . . 108
4.50 HB3_FETCH . . . 109
4.51 HB_FETCH . . . 109
4.52 HB_MN_FIT . . . 109
4.53 HB_OPEN . . . 110
4.54 HB_STORE . . . 110
4.55 HCLOSE . . . 110
4.56 HCOPY . . . 110
4.57 HDELETE . . . 110
4.58 HINDEX . . . 110
4.59 HISTOGRAM . . . 110
4.59.1 BOOK . . . 111
4.59.2 DISPLAY . . . 112
4.59.3 DUMP . . . 113
4.59.4 ERRORS . . . 114
4.59.5 EXTRACT . . . 114
4.59.6 FILL . . . 114

4.59.7 IGTABLE . . . 115
4.59.8 LEGO . . . 115
4.59.9 OVERLAY . . . 115
4.59.10 PLOT . . . 116
4.59.11 SURFACE . . . 118
4.59.12 2DIM . . . 119
4.60 HISTORY . . . 119
4.61 HMAKE . . . 119
4.62 HMERGE . . . 119
4.63 HRECOVER . . . 120
4.64 HRENAME . . . 120
4.65 HY_FETCH . . . 120
4.66 IF . . . 120
4.66.1 expression . . . 121
4.66.2 Examples . . . 121
4.67 IGTABLE . . . 122
4.68 INDEX . . . 122
4.69 INQUIRE . . . 122
4.70 INTEGRATE . . . 123
4.71 KEY . . . 123
4.72 LDIRECTORY . . . 124
4.73 LEGO . . . 125
4.74 LS . . . 125
4.75 MADD . . . 125
4.76 MDIRECTORY . . . 125
4.76.1 Examples . . . 125
4.77 MERGE . . . 126
4.78 MESSAGE . . . 126
4.79 MN_FETCH . . . 126
4.80 MN_STORE . . . 126
4.81 MSUBTRACT . . . 127
4.82 MULTIPLY . . . 127
4.83 NO_CUT . . . 127
4.84 NO_WINDOW . . . 128
4.85 NORMALIZE . . . 128
4.86 NTUPLE . . . 128
4.86.1 DUMP . . . 129
4.86.2 EPROFILE . . . 129
4.86.3 FILTER . . . 130
4.86.4 MERGE . . . 130
4.86.5 PLOT . . . 130
4.86.6 PROJECT . . . 131
4.86.7 SCAN . . . 134
4.86.8 SPROFILE . . . 134
4.87 OPEN . . . 134
4.88 OVERLAY . . . 135
4.88.1 /SAME . . . 135
4.88.2 /DIFFERENT . . . 135
4.88.3 /NEXT . . . 136

4.88.4 /NTUPLE . . . 136
4.88.5 /SMOOTH . . . 136
4.89 PARSE . . . 136
4.89.1 Examples . . . 136
4.90 PARTITION . . . 137
4.91 PLOT . . . 137
4.91.1 /CLEAR . . . 138
4.91.2 /NOCLEAR . . . 138
4.91.3 /EMPTY . . . 139
4.91.4 /NEXT . . . 139
4.91.5 /NTUPLE . . . 139
4.91.6 /SMOOTH . . . 139
4.91.7 Examples . . . 139
4.92 PRINT . . . 140
4.93 PROJECT . . . 140
4.94 PWD . . . 140
4.95 READ . . . 141
4.95.1 COMMAND . . . 141
4.95.2 DATA . . . 141
4.96 REBIN . . . 141
4.97 REDRAW . . . 141
4.98 REMOVE . . . 142
4.99 RENAME . . . 142
4.100 RETURN . . . 142
4.101 ROOT_FETCH . . . 142
4.101.1 Examples . . . 143
4.102 ROOT_OPEN . . . 144
4.103 SCALE . . . 144
4.104 SCAN . . . 144
4.105 SCT_FETCH . . . 144
4.106 SET . . . 145
4.106.1 ABORT . . . 145
4.106.2 ALIAS . . . 146
4.106.3 AUTOFETCH . . . 146
4.106.4 AUTOSCALE . . . 146
4.106.5 AUTOSWITCH . . . 146
4.106.6 AUTOTRIM . . . 146
4.106.7 AXIS . . . 146
4.106.8 BACKGROUND . . . 147
4.106.9 BIN . . . 147
4.106.10 BOX . . . 147
4.106.11 BREAK . . . 147
4.106.12 CHARACTER . . . 147
4.106.13 COLOR . . . 148
4.106.14 COLOUR . . . 148
4.106.15 DBASE . . . 149
4.106.16 DEBUG . . . 149
4.106.17 DEFAULT . . . 149
4.106.18 DIRECTORY . . . 150

4.106.19 DISPLAY . . . 150
4.106.20 DSIZE . . . 150
4.106.21 DUMP . . . 151
4.106.22 ECHO . . . 151
4.106.23 EDIT . . . 151
4.106.24 ENDSET . . . 151
4.106.25 ERR_ZERO . . . 151
4.106.26 EXCLUSIONS . . . 151
4.106.27 EXIT . . . 151
4.106.28 FIT . . . 152
4.106.29 FONT . . . 153
4.106.30 FOOTER . . . 153
4.106.31 FRAME . . . 154
4.106.32 FSIZE . . . 154
4.106.33 FUNCTION . . . 154
4.106.34 GRID . . . 155
4.106.35 GSIZE . . . 156
4.106.36 HARDCOPY . . . 156
4.106.37 HATCH . . . 156
4.106.38 HEADER . . . 157
4.106.39 HIGZ . . . 157
4.106.40 HISTOGRAM . . . 157
4.106.41 IDB . . . 157
4.106.42 IDR . . . 158
4.106.43 IDSHOW . . . 158
4.106.44 IDSIZE . . . 158
4.106.45 IGARC . . . 158
4.106.46 IGTABLE . . . 158
4.106.47 LABEL . . . 159
4.106.48 LIMITS . . . 159
4.106.49 LOG . . . 159
4.106.50 LSIZE . . . 160
4.106.51 MANUAL . . . 160
4.106.52 MARGIN . . . 160
4.106.53 MODE . . . 160
4.106.54 MOUSE . . . 161
4.106.55 NEXT_WINDOW . . . 161
4.106.56 NORMALIZE . . . 161
4.106.57 NTUPLE . . . 161
4.106.58 NULL . . . 162
4.106.59 OPT_ZERO . . . 162
4.106.60 ORDER . . . 162
4.106.61 ORTHOGONAL . . . 163
4.106.62 PAGER . . . 163
4.106.63 PAPER . . . 163
4.106.64 PARAMETER . . . 164
4.106.65 PATH . . . 170
4.106.66 PATTERN . . . 171
4.106.67 PI . . . 171

4.106.68 PLOT . . . 171
4.106.69 PSIZE . . . 171
4.106.70 RATIO . . . 172
4.106.71 RECL . . . 172
4.106.72 REDRAW . . . 172
4.106.73 ROOT_ID . . . 172
4.106.74 ROTATION . . . 172
4.106.75 SCALE . . . 172
4.106.76 SECONDARY_ID . . . 173
4.106.77 SHOW_ZERO . . . 173
4.106.78 SHELL . . . 173
4.106.79 SIGNAL . . . 173
4.106.80 SIZE . . . 174
4.106.81 SSIZE . . . 174
4.106.82 STATISTICS . . . 174
4.106.83 SYMBOL . . . 174
4.106.84 TEXT . . . 176
4.106.85 THICKNESS . . . 176
4.106.86 TICKS . . . 176
4.106.87 TIME . . . 177
4.106.88 TITLE . . . 177
4.106.89 TKTCL . . . 178
4.106.90 TSIZE . . . 178
4.106.91 USIZE . . . 178
4.106.92 WAIT_CR . . . 178
4.106.93 WINDOW . . . 178
4.106.94 NO_WINDOW . . . 179
4.106.95 WMARGIN . . . 179
4.106.96 WORKING_DIR . . . 180
4.106.97 WSIZE . . . 180
4.106.98 ZERO . . . 180
4.107 SHELL . . . 180
4.108 SHOW . . . 180
4.108.1 ALL . . . 181
4.108.2 ALIAS . . . 181
4.108.3 CHAR . . . 181
4.108.4 COMMANDS . . . 181
4.108.5 COMMENT . . . 181
4.108.6 CONSTRAINT . . . 181
4.108.7 CUTS . . . 181
4.108.8 DEFINITION . . . 182
4.108.9 DIRECTORY . . . 182
4.108.10 EXCLUSIONS . . . 182
4.108.11 FILES . . . 182
4.108.12 FLAGS . . . 182
4.108.13 FRAME . . . 182
4.108.14 INCLUSIONS . . . 182
4.108.15 KEYS . . . 182
4.108.16 LABEL . . . 182

4.108.17 LIMIT . . . 183
4.108.18 LOG . . . 183
4.108.19 MODE . . . 183
4.108.20 ORDER . . . 183
4.108.21 PLOT . . . 183
4.108.22 REGISTER . . . 183
4.108.23 SCALE . . . 183
4.108.24 SEGMENTS . . . 184
4.108.25 SIZES . . . 184
4.108.26 TICK . . . 184
4.108.27 UNITS . . . 184
4.108.28 VARIABLE . . . 184
4.109 SMOOTH . . . 184
4.109.1 /HBOOK . . . 185
4.109.2 /IMSL . . . 185
4.109.3 /NAGLIB . . . 185
4.109.4 /TOPDRAW . . . 185
4.110 SPAWN . . . 185
4.111 SPLINE . . . 186
4.111.1 /HBOOK . . . 186
4.111.2 /IMSL . . . 186
4.111.3 /NAGLIB . . . 186
4.112 SQUEEZE . . . 187
4.113 STAT . . . 187
4.114 STORE . . . 187
4.114.1 /NEW . . . 187
4.114.2 /UPDATE . . . 187
4.115 SUBTRACT . . . 188
4.116 SUM . . . 188
4.117 SURFACE . . . 188
4.118 TITLE . . . 188
4.119 UNALIAS . . . 189
4.120 UNDEFINE . . . 189
4.121 WAIT . . . 189
4.122 WDIRECTORY . . . 189
4.123 WINDOW . . . 189
4.124 WRITE . . . 189
4.124.1 DATA . . . 190
4.124.2 LOG . . . 190
4.125 XSCALE . . . 190
4.126 XSHIFT . . . 190
4.127 YSCALE . . . 190
4.128 YSHIFT . . . 191
4.129 ZDIRECTORY . . . 191
4.130 ZSCALE . . . 191
4.131 ZSHIFT . . . 191

5 Reference Manual for MINUIT **192**
5.1 MINUIT . . . 192
5.2 BACK_SUB . . . 192
5.3 CALLS . . . 192
5.4 CHI_PLOT . . . 192
5.5 CONSTRAIN . . . 193
5.5.1 Examples . . . 193
5.6 CONTOUR . . . 194
5.7 COVARIANCE . . . 194
5.8 DISPLAY . . . 194
5.9 DUMP . . . 194
5.10 END . . . 195
5.11 ERROR_DEF . . . 195
5.12 EXCLUDE . . . 195
5.13 EXIT . . . 195
5.14 FCN_DRAW . . . 195
5.15 FCN_PLOT . . . 196
5.16 FIT_INFO . . . 196
5.17 FIX . . . 196
5.18 FLOAT . . . 196
5.19 GRADIENT . . . 196
5.20 HESSE . . . 197
5.21 IMPROVE . . . 197
5.22 INCLUDE . . . 197
5.23 INFO . . . 197
5.24 ITERATIONS . . . 197
5.25 MATOUT . . . 198
5.26 MAX_CALLS . . . 198
5.27 MIGRAD . . . 198
5.28 MINIMIZE . . . 198
5.29 MINOS . . . 198
5.30 MNCONTOUR . . . 199
5.31 MODIFY . . . 199
5.32 NO_EXCLUDE . . . 199
5.33 NO_GRADIENT . . . 199
5.34 NO_INCLUDE . . . 200
5.35 NO_ITERATIONS . . . 200
5.36 PAGE . . . 200
5.37 PRECISION . . . 200
5.38 PRINTOUT . . . 200
5.39 PROB_PLOT . . . 200
5.40 PUNCH . . . 201
5.41 RELEASE . . . 201
5.42 RESTORE . . . 201
5.43 SAVE . . . 201
5.44 SCAN . . . 201
5.45 SEEK . . . 201
5.46 SIMPLEX . . . 202
5.47 STANDARD . . . 202

5.48 STOP . . . 202
5.49 STRATEGY . . . 202
5.50 UNCONSTRAIN . . . 202
5.51 UNIT . . . 202

A Acknowledgements **203**

B Examples of Symbols, Fonts, Hatching and Keys **204**

C Mn_Fit Examples **223**
C.1 Demo 1: Simple Fetch and Plot . . . 224
C.2 Demo 2: Simple Gaussian Fit . . . 226
C.3 Demo 3: Plotting with overlays, inserts, colours and fonts . . . 230
C.4 Demo 4: Different ways of displaying 2-D histograms . . . 236
C.5 Demo 5: Simultaneous fit of 2 plots with function overlays . . . 239
C.6 Demo 6: Drawing Possibilities . . . 248
C.7 Demo 7: Display of an L3 Luminosity Monitor Event . . . 253

D List of Changes **256**
D.1 Version 5.13 07/11/2005 . . . 256
D.2 Version 5.06 16/05/2005 . . . 258
D.3 Version 4.07 02/12/2002 . . . 260
D.4 Version 4.06 06/04/2000 . . . 263
D.5 Version 4.05 16/06/99 . . . 264
D.6 Version 4.04 12/06/96 . . . 270
D.7 Version 4.03 15/09/95 . . . 273
D.8 Version 4.02 25/07/94 . . . 277
D.9 Version 4.01 17/12/93 . . . 280

List of Figures

B.1 Mn_Fit Symbols 206
B.2 HIGZ Portable Software Characters 208
B.3 Postscript Version of HIGZ Portable Software Characters 209
B.4 Examples of HIGZ Postscript Fonts 210
B.5 Examples of GKSGRAL Fonts 211
B.6 Examples of HIGZ Hatching 212
B.7 Examples of GKSGRAL Hatching 213
B.8 Examples of Patterns 214
B.9 Examples of Keys 215
B.10 Picture sizes and the commands to set them 216
B.11 Postscript characters - Codes 0 to 75 217
B.12 Postscript characters - Codes 75 to 149 218
B.13 Postscript characters - Codes 150 to 224 219
B.14 Postscript characters - Codes 225 to 299 220
B.15 Postscript characters - Codes 300 to 374 221
B.16 Postscript characters - Codes 375 to 449 222

C.1 Demo 1 — Simple Fetch and Plot. 225
C.2 Demo 2 — Simple Gaussian Fit. 229
C.3 Demo 3 — Plotting with overlays, inserts, colours and fonts. 235
C.4 Demo 4 — Different ways of displaying 2-D histograms. 238
C.5 Demo 5 — Simultaneous fit of 2 plots with function overlays. 247
C.6 Demo 6 — Drawing Possibilities. 252
C.7 Demo 7 — Display of an L3 Luminosity Monitor Event. 255

Chapter 1

Introduction

Mn_Fit is an interactive fitting and plotting package, that uses MINUIT [1] 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. A selection of the most commonly used commands sorted into categories is given, to help you get started if you are a new Mn_Fit user. 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.

By its very nature a 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`. The version number N_MM 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. If bug fixes are necessary the version will be of the form N_MM_LL.

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 1.10 on page 16 (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 4.106 on page 145 (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 4.19 on page 59 (CUT) and section 4.86.6 on page 131 (NTUPLE 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 1.9 on page 16 (Identifiers).

For details on the implementation of the MINUIT package see section 3 on page 44 (MINUIT). 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 4.48.9 on page 91 (FUNCTION LIST) for details). If you need a more specialized function you can either define it interactively using the CERN COMIS package [2], meaning you can write your own functions without relinking Mn_Fit (see section 1.16 on page 24 (Using COMIS) and section 4.48.9 on page 104 (FUNCTION LIST COMIS) for more details), or you can relink Mn_Fit with a user function (see section 4.48.9 on page 106 (FUNCTION LIST User) for more details). A user function is useful on 64 bit machines when COMIS sometimes has problems or if you have a complicated user function that uses a lot of CPU time.

It is also possible to apply constaints between the parameters of the functions you are fitting (see section 5.5 on page 193 (MINUIT CONSTRAIN)), and you can convolute a Gaussian resolution function with your function to include the effect of experimental resolution (see section 4.106.28 on page 152 (SET FIT CONVOLUTE)). Likelihood fitting is a built-in option and it is also possible to fit using a histogram as a function taking into account the statistical errors on each bin (see section 4.47 on page 87 (FIT)).

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 `IF` blocks and you can pass parameters to the macro as well as setting them within the macro. You can give a list of directories that Mn_Fit will search (see section 4.106.65 on page 170 (SET PATH)) when it is looking for a file. The working directory for output files can also be set (`WDIRECTORY` or `SET WORKING_DIR`). On Unix machines filename completion and command line editing are available using the GNU readline package.

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 4.5 on page 54 (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 1.7 on page 11 (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 VMS Help on VMS machines and a simulated version on all other machines.

1.2 How To Run Mn_Fit

Mn_Fit has implemented on Vax, Alpha, Decstation, Apollo, HP, SunOS, SGI, Linux and IBM RS6000 computers. However, the latest version has only been compiled and tested on Linux, Alpha/OSF1 and SunOS. Getting it running on other Unix machines should be very easy. The

VMS implementation is frozen, as most experiments have or are going to turn off their VMS machines in the near future.

If you are on a Unix machine and Mn_Fit is installed in a standard area (e.g. `/usr/local/bin`), then you can just give the command:

```
mn_fit [arguments]
```

If you are on a DESY Unix machine with access to AFS give the command:

```
/afs/desy.de/user/b/brock/bin/mn_fit [arguments]
```

If you are on a CERN Unix machine with access to AFS give the command:

```
/afs/cern.ch/user/b/brock/bin/mn_fit [arguments]
```

Not all flavours of Mn_Fit are available at both CERN and DESY. Let me know if you cannot find a version for your system.

On Unix machines where Mn_Fit is not installed in a standard area you should set the `MN_FIT` environment variable to the top level directory of the Mn_Fit tree. and then you can say:

```
$MN_FIT/bin/mn_fit [arguments]
```

or include `$MN_FIT/bin` in your `PATH`. Preferably the system manager should edit the file `$MN_FIT/bin/mn_fit` so that the `MN_FIT` environment variable is defined correctly. He/she should also put the script in a "normal" directory such as `/usr/local/bin`.
You can give arguments with the command to specify which version to run:

```
filename  Read all the Mn_Fit commands from the given filename.
-r        Mn_Fit with ROOT interface (mn_root_x.exe, if available)
-u        User version (mn_user_x.exe, if available)
-disp     Display version (only implemented for L3 and ZEUS)
-test     Test version (mn_test_x.exe, if available)
-dev      Development version (mn_dev_x.exe, if available)
-old      Old version (mn_old_x.exe, if available)
-x        X Windows version (default)
-n        Host name for the X windows version
-t        Transport mode for X windows version (VMS only)
-s        Server number for X windows version (VMS only)
-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.

For the X windows version you can define the DISPLAY variable before you start Mn_Fit or in the command line (using `-n hostname`). Line 10 of `higz_windows.dat` will be used to set the display size. If you start the X windows version and the environment variable `DISPLAY` (Unix) or the logical `decw$display` (VMS) is not defined, then the display will be set to `hostname‘ (Unix) or‘sys$node` (VMS), if the `hostname` is not given. Under VMS, if the DISPLAY is not set use the command line `-n hostname -t transport`. The transport is usuallly either tcpip (could also be decnet). The host name is either the tcp/ip name (or denet name). If you specify the display in one of these ways just hit <`CR`> when prompted for 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 file `~/.mn_fitrc` on Unix machines. or the logical name `mn$logon` on VMS. A warning will be given if neither file is found.

Mn_Fit is interfaced to graphics devices using HIGZ [3] (PLTSUB is still in the code for illustration only). Within HIGZ only the X Windows interface is now supported.

1.3 Getting Started

The number of commands available in Mn_Fit is probably a bit overwhelming for a newcomer. The following lists some of the most commonly used commands sorted into different categories. For more details on these commands look in the relevant section in the reference manual or use the online help. Look at the Table of Contents or the Menu of Commands (chapter 2 on page 33) to get an overview of all the commands.

General

shell shell_command	Get back to the command level.
help	Help for Mn_Fit.
exec	Execute a series of Mn_Fit commands in a file.
define	Define a new command that is a collection of other Mn_Fit commands.

Histogramming

fetch filename hist_number	Fetch some histograms from a file.
open filename	Open an RZ file with histograms.
index	List the histograms fetched.
index hist_number	
plot hist_number	Plot a histogram.
overlay hist_number	Overlay a histogram on one that has already been plotted.
2dim option hist_number	Extra options for plotting 2-D histograms, e.g. 2DIM LEGO
add hist1 hist2 hist3	Add histograms together.
subtract, multiply, divide, efficiency	Alternatives for add.
dump hist_number	Dump a histogram's content
rebin hist_number low_bin high_bin new_number_of_bins	
store/new filename hist_number	Save some histograms in a new file.
store/update filename hist_number	Save some histograms in an existing file.

Hist_number can be 0 for all, or `number1:number2` for a range of histograms. In Mn_Fit a histogram is identified by two numbers, `id` and `idb`. `id` is the number the user booked with HBOOK, for example, in the analysis fortran code, idb is a secondary identifier. The following example shows how to use it. Use data ON the Υ(4S) and OFF the peak to obtain the net $B\bar{B}$ contribution for many histograms with identifiers between 100 to 400:

```
set idb 1
fetch ON_data.hst 100:400
set idb 2
fet OFF_dat.hst 100:400
sub 0&1 0&2 0&3 1.0 2.0
store/new BBbar.hst 0&3
```

Here, it is assumed that files `ON_data.hst` and `OFF_data.hst` are obtained by runing the same program on two datasets so that the histogram identifers are the same for both datasets. It is also assumed that twice as much luminosity was taken on the peak as in the continuum, hence the scale factors of 1.0 and 2.0. Note that commands can always be abbreviated to the point of ambiguity.

Booking and filling new histograms

book hist_number ...	Book a new histogram.
fill hist_number ...	Fill the new histogram.

Ntuples

fetch filename hist_number	Here hist_number is an ntuple.
ntuple/project hist_number	Project an ntuple with cuts.
ntuple/plot hist_number	Project an ntuple with cuts and plot the projection.
cut new a + b > c + d	Cut expression – fortran style.
cut list	
cut use 1 & 2 | 3 & 4	Logical combination of cuts to use.
cut delete cut_number	Remove a cut.
cut name cut_name ...	Give a cut a name to be used in `cut use` command.

Fitting

function list	List more than 40 predefined functions.
function add	Add a function for fitting.
function info	List the added functions.
function use	Function(s) to use when fitting
fit hist_number	Command to get into MINUIT, where and one can do the usual MINUIT stuff such as `migrad`, `minimize`, constrained fit, simultaneous fits, ... Use the `exit` command to get back to the normal command level.
display	Show the fit results.

Cosmetic

set font item font_number	Change the font.
set colour item colour	Change the colour of parts of a plot.
comment	Add some text to a picture.
key	Add a legend for a figure.
hardcopy	Write the current picture to a file (by default Postscript).
set window ...	Make multiple windows (zones) in one picture.
set ...	Sets lots of things - see section 4.106 on page 145 (SET).

1.4 Giving Commands

On most Unix machines Mn_Fit is linked with the GNU readline package that allows command recall, command line editing and filename completion. The command line editor follows emacs conventions. You can use the left and right arrow keys to edit the command line and also control keys. Useful control keys are:

CTRL/A Go to the beginning of the line.

CTRL/D Delete the character that the cursor is on.

CTRL/E Go to the end of the line.

CTRL/H Delete the character before the cursor.

CTRL/K Delete from the cursor to the end of the line.

CTRL/R Search for the last command containing the string you give.

CTRL/T Toggle the position of the character the cursor is on and the character before.

CTRL/U Delete from the beginning of the line to the cursor.

The history mechanism is very similar to that of the c-shell. You can either use the up and down arrows to recall commands or `!`. You can use filename completion by typing part of the filename and then hitting `<TAB>`. The filename will complete until it finds more than 1 option. If you hit `<TAB>` again it will give a list of the possible completions. Filename completion does not work when you include environment variables in the filename, but they do get interpreted properly.

On VMS machines you can also edit the command line in the usual way. Command recall is available using the up and down arrow keys. Useful control keys for command line editing are:

CTRL/A Toggle between insert and overstrike mode.

CTRL/E Go to the end of the line.

CTRL/H Go to the beginning of the line.

CTRL/U Delete from the beginning of the line to the cursor.

The list of control characters for both machines is not complete, but these are the ones I most commonly use. For more details see the GNU readline manual (or man readline if the man pages are installed on your machine) on Unix machines and HELP Line_editing on VMS machines.

1.5 Fitting With Mn_Fit

The usual way to fit is using MINUIT. However spline fitting and histogram smoothing are also available for one dimensional plots.

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 can specify whether you want to do likelihood or χ^2 fitting, and whether the Monte Carlo statistics should be taken into account when calculating the likelihood.

If you just want to do a simple fit of a Gaussian, flat line or a line with a slope you can use the `FIT/qual` syntax. This form will add the correct function(s), disable all others and start fitting. A default list of commands is executed, that can also be changed by the user.

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 PLOT` 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 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 5.5 on page 193 (MINUIT CONSTRAIN) for more details). You can convolute the fitting functions with a Gaussian resolution in order to include detector effects (see section 4.106.28 on page 152 (SET FIT) CONVOLUTE for more details). The resolution has to be given with the 'SET FIT CONVOLUTE' command at present. Note that the Gaussian is convoluted with ALL the functions you are fitting. I intend to allow the user to specify which functions should be convoluted with the gaussian and allow the resolution to be a free parameter in a future release.

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 4.106.19 on page 150 (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. The fit status is put in register 113 (3 means it converged properly), while the EDM (Estimated Distance to Minimum) is in registr 114.

There are a large number of built-in functions (see section 4.48.9 on page 91 (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 (see section 4.48.9 on page 106 (FUNCTION LIST USER)) for instructions on how to relink Mn_Fit with a user function.

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. If you want to be able to shift or scale the histogram that is used as a function use the `SMOOTHED HISTOGRAM` function. When fitting with histograms you often want to know the fractional contribution of each histogram. The `FIT/FRACTION` command sets an overall normalization and the fraction of each function as free parameters. 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 4.48.9 on page 106 (FUNCTION LIST User) or section 4.48.9 on page 104 (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` or `FIT/NORM`. 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. To get the fraction and its error directly use the `FIT/FRACTION` command.

Always check that the results of the fit are reasonable - that the errors are reasonable (see section 1.5.1 on page 8 (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, you should make sure that things do indeed converge properly each time. If for some reason, the latter 2 fits often do not converge properly try exiting MINUIT and restarting it after each of the 3 fits.

1.5.1 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 or the variations are linear, the parabolic error and the MINOS errors should be almost the same. However if there are correlations, or the errors are asymmetric you should always use the MINOS errors to be sure.

There is a nice discussion of the meaning of the MINUIT errors and what you should be careful about in the MINUIT manual. You should read this for more information.

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 or run `MINOS`. A good check is that the parabolic error should lie between the MINOS errors if the problem is linear and be smaller than the MINOS errors if there are correlations. Again, this problem is much reduced with the CERN version of MINUIT.

Note that in the CERN version of MINUIT the parabolic error includes the correlations between the parameters.

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.

If you make a plot of your fit parameter with MINOS errors and then you want to store the histogram, you should not use the `STORE` command, as HBOOK does not know about asymmetric errors in its normal histograms. Instead use the `DAT_STORE` or `MN_STORE` commands that write the plots out in an ascii file or in Mn_Fit format.

1.5.2 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, EFFICIENCY`, and `AVERAGE`. I do not assign an error of 1 for `SCALE` and `NORMALIZE` as this can give strange effects when plotting with empty bins and large scale factors. Thus if you had 0 entries in an Υ(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` before fetching the histograms. If you want to `SCALE` or `NORMALIZE` a histogram that has bins with 0 entries you should give the command `HIST ERR id 2` to set the errors on these bins to 1.

When calculating an efficiency by dividing 2 plots, the error was underestimated for small numbers of entries. I 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. The form is:

$$\sigma_\alpha = \sqrt{\frac{(n+1)(N-n+1)}{(N+3)(N+2)^2}}$$

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 5 modes:
`nmode = -1`	Removes errors on all points,
`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,
`nmode = 3`	Set the error on all points to the same value.
`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.

1.5.3 Including Monte Carlo Statistical Errors

With the large data samples available at many experiments, Monte Carlo statistics are often a significant error. For measurements such as τ polarisation at LEP one usually fits a distribution to 2 components which are both represented by histograms with limited statistics.

There are a number of approaches on how to include this error – the one that is included in Mn_Fit version 4.02 is from Ian Scott [4], which assumes Poisson errors for both Monte Carlo and data and calculates a joint probability, P, to observe m Monte Carlo events and n data events, given a factor r more Monte Carlo events than data events. Using $\Gamma(p+1) = p!$ he obtains:

$$P = \frac{\Gamma(n+m+1)}{\Gamma(n+1)\Gamma(m+1)} \frac{r^{m+1}}{(1+r)^{n+m+1}}$$

There is a more recent paper by Barlow and Beeston [5] that allows reweighting of events also that I will look into implementing.

These approaches can probably also be used for the problem above of subtracting a continuum background with limited statistics. Instead it may be better to just fit the data to the sum of the signal and continuum background.

1.6 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 `@`. The `INQUIRE` command is also available so that you can prompt for the value of parameters. With this command you can also give the default value for a parameter, if you want to avoid beng prompted for it. 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 4.42 on page 82 (EXECUTE) for examples.

It is 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.

`IF` statements are available in macros and defined commands. Simple statements with up to four conditions are possible at present (see section 4.66 on page 120 (IF)).

If you need to put a number into a character string as part of a command you can use the `PARSE` command. This is particularly useful if you want to pass a histogram identifier that is in a register as a text parameter to another macro, or if you have a series of histogram identifiers in registers and you want to loop over them, constructing commands that use the identifiers.

Note that the last 100 commands are logged (see section 4.108.18 on page 183 (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.

WARNING: In Mn_Fit versions before 4.07/32 you should not include comment lines inside `DEFINE` commands, if there is a `DO` loop inside the same file. Inline comments are OK. Mn_Fit keeps track of which line is being read for future use in `IF` blocks and `DO` loops. If you have comment lines inside the `DEFINE` command, the counting of the line number gets out of sync. As a result any `DO` loops that are executed after the definitions (within the same macro) will not work properly. After the first run through the loop Mn_Fit will rewind the file and then jump to the wrong line for the second and future runs through the loop.

1.7 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.

Certain character variables can also be given - an array of character strings (`CHAR`) histogram titles (`CHTITLE`) and Ntuple variable names (`CHNAME`) and/or the values (`CHVAL`) of character variables - see section 1.12 on page 18 (Text) for more details. These are mostly useful for messages and for manipulating histogram titles when making projections of Ntuples. They could also be useful to translate from ROOT string histogram identifiers to Mn_Fit numerical identifiers.

In addition you can define new variable names using the syntax `DEPOSIT 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. If you want to be able to treat the variables as an integer they should start with a letter between I and N. The new variables can be deleted using the REMOVE command. See section 4.28 on page 69 (DEPOSIT) for more details:

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

These extra variables are stored in registers > 300. You can look at the values of registers, parameters, bin contents etc. using the 'EXAMINE' command or the 'SHOW REGISTER' command for registers only. Note that numbers are always floating, with the exception of 'IR'.

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 500.

  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 1.12 on page 18 (Text)).

  Registers 100 and above contain the following information:

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

  102 The number of points summed over in 'SUM'

  111 The chi**2 or likelihood from the fit

  112 The confidence level of a fit

  113 The fit status (3 means converged properly)

  114 The EDM (estimated distance to minimum) of the 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 and 231 to 267 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)

128 The creation date of the histogram (yymmdd)

129 The creation time of the histogram (hhmm)

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 mean value for the x-axis

135 The sigma for the x-axis

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

137 The lower limit of the y-axis

138 The upper limit of the y-axis

139 The mean value for the y-axis

140 The sigma for the y-axis

etc. up to 14th dimension of an Ntuple.

Registers 231-257 contain total contents, as well as underflows and overflows of the default histogram:

231 Underflows x-axis

232 Contents x-axis

233 Overflows x-axis

For 2-dimensional histograms 9 registers are filled contents in register 235), while for 3-dimensional histograms 27 registers are filled (contents in register 244).

Registers $>$ 300 contain extra variable names that you have defined. Up to 200 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(l,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. For plots with asymmetric errors `DX` and `DY` are interpreted as the negative errors if you want to get the value and as both errors if you are using the `DEPOSIT` command.

- 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. Within the 'DEPOSIT' command you can access CWN variables, but not arrays.

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

  If the plot identifier is stored in a register, use the `PARSE` command to convert it into a number.

1.8 Expressions

An expression can include any of the parameters which can be used for numbers (see HELP 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. 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, MIN, MAX, MOD, SIGN
DATE, TIME, DATE_TIM, DATE_MIN, TIME_MIN
```

Note the result of an expression is always returned as a floating point number, i.e. also the results of `INT` and `NINT` are converted back to floating point immediately. `MIN` and `MAX` take 2 arguments.

The `DATE` function expects the date in the form `YYMMDD`, `TIME` is in the form `HHMMSS`, `DATE_TIM` and `DATE_MIN` take 2 arguments `YYMMDD,HHMMSS`. `DATE_MIN` and `TIME_MIN` expect the time in the form `HHMM`. The date functions return the number of days,hours,minutes or seconds (depending

on the time mode that has been set with the SET TIME command) since the reference time (or 1-Jan-1980 if not reference time has been set). The time functions return the same things, but always starting from midnight.

In addition the following operations on functions are available:

```
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, which you give as arguments.

The syntax is FPOSn(x) where n is the function number and x is the value at which it should be evaluated. For FINT the syntax is FINTn(xlo,xhi). x, xlo and xhi can also be registers, parameters, or even expressions.

For the derivative the step size is currently set to 1.0. It is possible to set it using the SET FUNCTION STEP command. The number of intervals to integrate over can be set using the command SET FUNCTION INTEGRATE nint (default is 100). The function number can be that of a single function, or 0 to mean all currently selected functions.

For evaluating expressions with functions, if the bin width is set (i.e. you have done a fit, or used the FUNCTION HISTOGRAM command), then it is used. You can turn off the use of the bin width with the SET FUNCTION BIN_WIDTH command.

1.8.1 Examples

1. Calculate the integral of a Gaussian using either INTEGRATE or FINT:

```
fun add gaus sigma
1000
0
1
integrate 1
0 3
message 'Integral is {r101,(f6.2)}'
dep r1 = 0
dep r2 = 1.5
dep r3=fint1(r1,2*r2)
message 'Integral is {r3,(f6.2)}'
```

2. Illustating the effect of the bin width:

```
fun add gaus sigma
1000
0
1
dep r1 = fpos1(0)
fun hist 1 &1 0 100 -2 2
set function bin_width on
dep r2 = fpos1(0)
set function bin_width off
dep r3 = fpos1(0)
```

1.9 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 1.10 on page 16 (Secondary Identifiers).

1.10 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.

You can use the `MN_STORE` and `MN_FETCH` commands to store and fetch histograms in Mn_Fit format, that will retain the secondary identifiers.

1.10.1 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
   PLOT 1&0
   OVER 1&10 2/BLUE
   ```

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

1.11 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
```

Note that histograms always get converted to a series of points when fitting, so the default symbol for `DISPLAY` is always -32.

To get a picture of the available symbols issue the command `exec $MN_FIT/help/symbol.mnf` (Unix) or `EXEC MN_FIT_HELP:SYMBOL.MNF` (VMS). 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 4.106.37 on page 156 (SET HATCH) for details on some of the hatching available and 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
 12        Square
 13        Triangle
 14        Inverted triangle
 15        Diamond
 16        Plus  (+)
 17        Cross (x)
 18        Asterix (*)
 19        Octogon (used to be 11)
20-28      Show x error bars for histograms
```

```
30-38     Show y error bars
40-48     Show x and y error bars
60-69     Show x error bars with line at end (symbol size, SET SSIZE)
70-79     Show y error bars with line at end (symbol size, SET SSIZE)
80-89     Show x and y error bars with line at end (symbol size, SET SSIZE)
-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
10-19     As for histograms
20-49     As for 10-19
-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
10-19     Area of symbol is proportional to number of entries in
          the bin
20-49     As for 10-19
-n        Show symbol filled
```

If the minimum weight is negative and the maximum weight is positive for 2-D histograms and you use a symbol number greater than 10 or less than -10 you will get the symbol for positive weights and its inverse for negative weights. This mode is also used for displaying the result of a 2-D fit. This mode is only used if either the lower or upper plotting limit is 0.0. Use the `SET Z LIMIT` command to set the limits. Otherwise only those entries within the specified range are shown.

2-D histograms can also be plotted using the `LEGO` or `SURFACE` commands or preferably using the interface to the HIGZ IGTABL routines, `2DIM` or `IGTABLE`.

I use the HIGZ circle routine to draw circles. It is also possible to use large dots (symbol 10 or -10) and set the dot scale size using the `SET DSIZE` command. A scale factor of 10 or 20 is usually good. Note that you only see the big dots when you print the picture.

1.12 Text

You should normally just type in the text you want to see. For details on how to get special characters, Greek letters, German special character etc. see the subtopics fonts and IGTEXT.

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'"
```

With this form the parameter `@1` would contain '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, 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 1.7 on page 11 (Numbers) for what is available). It can also be a text element, i.e. a title, an Ntuple variable name or the value of an Ntuple variable fo the type character (see below for more details). `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 a letter between `I` and `N`. `NINT` is used to convert real numbers to integers. Any variables that start with `CH` are assumed to be character variables.

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 `@`. You do not have top precede } by a `@`. If you also want to translate numbers into strings in other commands you can use the `PARSE` command (see section 4.89 on page 136 (PARSE) and the examples).

Superscripts and subscripts are terminated by a `!`. `!` is also used for command line history in the Unix version (4.01 onwards), therefore you have to quote `!` with a backslash to get an exclamation mark if you give the command interactively. In addition if `!` is defined as the comment character you must also enclose the expression in single quotes (version 4.03 onwards):

```
set x label '(GeV/c^2\!)'
```

You can use certain character variables to set histogram titles and print messages etc. The character array is useful in loops and also for correlating ROOT character identifiers with Mn_Fit numerical identifiers. The items available are:

```
CHAR(nn)[[n1]:[n2]]
CHTITLE(ida[&idb])[[n1]:[n2]]
CHNAME(ida[&idb],nvar)[[n1]:[n2]]
CHNAME(ida[&idb],tvar)[[n1]:[n2]]
CHVAL(ida[&idb],nvar,npnt[,nelem])[[n1]:[n2]]
CHVAL(ida[&idb],tvar,npnt[,nelem])[[n1]:[n2]]
```

where nn is the character array element; ida is the primary histogram identifier and idb is the optional secondary identifier; nvar is the Ntuple variable number and tvar is the Ntuple variable name; npnt is the Ntuple event number; nelem is the (optional) element number in the character array. A character range can be given in almost the usual FORTRAN way [n1:n2] for characters n1 to n2, [n1:] for character n1 to the end of the string or [:n2] for character 1 to n2:

```
parse root_fetch {char(4)};{ir4,(i4.4)}
message 'Title of histogram 34 is {chtitle(34)}'
message 'Value of event 3, variable 4, {chname(34,4)} is {chval(34,4,3)}'
message 'Event 5, Element 2 of cval2 is {chval(34,cval2,5,2)}'
```

In order to be sure that things get parsed correctly, use the PARSE command or enclose the string with the character variables in quotes.

You can set the character variables CHAR, CHTITLE, CHNAME using the DEPOSIT or CALCULATE command:

```
DEPOSIT char(4) = 'ABC'
deposit char(21) = {chtitle(10&4)[:20]}
```

1.12.1 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)}.
   ```

2. You can use the even use PARSE to set the name of a macro that you want to execute:

   ```
   do i=1,10
     parse exec demo{@i,(i2.2)}
   enddo
   ```

   will execute demo01 to demo10.

3. You can use the use PARSE together with registers and the character array to convert ROOT identifiers to Mn_Fit numerical identifiers:

   ```
   dep char(1) = habc
   dep char(2) = hdef
   dep char(3) = hghi
   do i=1,3
     dep r@i = 100 + @i
     parse root_fetch {char(@i)};{ir@i,(i4.4)}
   enddo
   ```

   will read in histograms habc,hdef,hghi and give them identifiers 101,102,103.

1.12.2 Fonts

If you use a negative font number, e.g. -13 and precision 1 then IGTEXT (i.e. font 2000) will be used on the screen and Postscript fonts will be used for hardcopies. This seems to be a good way to get the best of both worlds (special, Greek characters etc. on your screen and Postscript fonts in a file, which save a lot of space and are nicer than font 2000). Font -1004 (Helvetica) is therefore now the default font on all machines. The only minor drawbacks are that underscores appear as `z` on the screen and the text width is a bit different for the 2 fonts.

Font 2000 uses the IGTEXT routine (formerly HPLSOF). It is the default on the screen and is always available. Font -1004 (Helvetica) is nicer for plots and is the default when you print a picture. Another commonly used font is -1013 (Times-Roman). 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).

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. -1013 to get font -13 with precision 1). 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.

Postscript fonts are available for hardcopies if you have a Postscript printer. Fonts -1 to -24 are availble if you use the HIGZ Postcript interface, which is the default, while fonts -1 to -13 are available with GKSGRAL.

HIGZ Postscript fonts can be obtained using precision 0,1, or 2. They are identical when you make a hardcopy. The differences are how they appear on the screen. If you use Postscript fonts and precision 2 you get the hardware default font on the screen. If you use precision 1 then the standard IGTEXT font (font 2000) is used on the screen and the Postscript font is used when printing. If you use precision 0, you get the Postscript fonts on the screen, but not superscripts, subscripts, Greek characters etc.

Postscript characters that are not part of the normal keyboard can be obtained with the syntax `\nnn` where `nnn` is the Postscript character number. A set of tables with the complete set of Postscript characters can be obtained by running the macro `exec $MN_FIT/help/ps_char` and printing out the file `ps_char.ps` on a Postscript printer. If you have a complete Mn_Fit installation this file should be available as `$MN_FIT/help/ps_char.ps`. Some Postscript characters such as German letters with umlauts are available with a simpler syntax, `\"a` for example. See the subtopic IGTEXT for more details.

Note that there is an offset between the HIGZ/Postscript font numbers and those in GKSGRAL. Times-Roman is -1 in GKSGRAL and -13 in HIGZ/Postscript, Times-Italic is -2 and -1 respectively.

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).

1.12.3 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
~  go to Zapf Dingbats font (HIGZ Postscript only)
!  end of superscript or subscript or Zapf Dingbats
&  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
```

German letters with umlauts as well as the sharp s (also called sz) can be obtained with the syntax \"a etc. The sharp s is \"S.

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 @.

Superscripts and subscripts are terminated by a !. ! is also used for command line history in Unix version (4.01 onwards), therefore you have to quote ! with a backslash to get an exclamation mark if you give the command interactively. In addition if ! is defined as the comment character you must also enclose the expression in single quotes (version 4.03 onwards):

```
set x label '(GeV/c^2\!)'
```

If you put the command in a file, then do not include the backslash, but you must still include the quotes:

```
set x label '(GeV/c^2!)'
```

There are some characters that are often requested but are not readily available. I list a few of the more common ones here:

```
Prime          [\242]  Postscript character 242 in Greek
Per mille      \275
\vartheta      [\112]  The commonly used TeX character
\varphi        [\152]  The commonly used TeX character
Re             [\302]  Real part of a number
Im             [\301]  Imaginary part of a number
```

Unfortunately the commonly used ℓ TeX character is not available, as I far as I can tell. I recommend that you browse through `$MN_FIT/help/ps_char.ps` if you are searching for some other characters. These tables are also included in Appendix B of the Mn_Fit manual.

Examples

`@<1234@>` will get you 1234 and
`'[p]^+![p]^-! Invariant Mass (GeV/c^2!)@!'` will get you $\pi^+\pi^-$ Invariant Mass (GeV/c^2)!
`Die Bl\"atter sind gro\"S'` will get you Die Blätter sind groß when printed on a Postscript printer.

1.13 Mouse

If you are running Mn_Fit at a workstation that has a mouse, you can use it to position `COMMENT`s, `KEY`s and to `DRAW` things like lines, arrows, boxes and polygons. Use the `SET MOUSE on|off` command to turn on or off usage of the mouse. When asked to give the position move the cursor to the correct position and click on the left mouse button. If you are changing an item and are happy with its current position, just put the cursor anywhere on the picture and click on the right mouse button. Note that posiitoning things with the mouse only works in `CM` mode.

For experts: I use the HIGZ routine IRQLC to get the mouse position.

1.14 Using Ntuples

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 4.86 on page 128 (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 project onto either a 1 or 2-dimensional histogram, a profile plot, a scatter plot or another Ntuple. 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. You can call a COMIS subroutine for each event in the Ntuple (NTUPLE SCAN) and can make whatever plots or calculations you wish there. You can select the events that you want to use for further analysis using the NTUPLE FILTER command.

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.

Columnwise Ntuples (CWNs) are also supported. For details on how to book this sort of Ntuple see the HBOOK manual. The main advantage of them is the variable number of columns per event and that only the columns that are needed are fetched. If you SCAN a CWN, all columns will be fetched. NTUPLE FILTER does not yet work with CWNs. If you PROJECT then only those variables that are projected onto will be fetched. However I do not know which variables you want to access if you are using a COMIS function as a cut, so you have to give the command SET NTUPLE VARIABLE to list which variables are needed.

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.

1.15 ColumnWise Ntuples

ColumnWise Ntuples (CWN) are now (version 4.03) supported in Mn_Fit. In addition to treating them like single variable ntuples you can also specify a range of elements or an array element that should be accessed.

The number of columns you give for a CWN variable should always be the same as its definition. Use : to give a range of variables. If you want to give a range of more than 1 variable, it is up to you to make sure that the ranges are consistent:

```
ntuple project 34 rval2(2:nval2)*rval3(1,2:nval2) rval3(4,2:nval2) -
 &11 75 -5 10 75 -5 10
```

Note that the limits of each variable of a CWN are not known. Therefore you always have to give the limits for plotting yourself.

CWN variables that you want to access in a COMIS function as part of an NTUPLE PROJECT command have to be specified using the SET NTUPLE VARIABLE command. However, for the NTUPLE SCAN command all variables will be fetched.

WARNING: If you try to loop over a CUT variable this will not be checked, but the loop will be ignored. If you need a cut variable to change with the element number of a CWN, then you will have to use a COMIS function for the cut.

For more details see section 4.86.6 on page 131 (NTUPLE PROJECT).

1.16 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 4.48.9 on page 104 (FUNCTION LIST COMIS) for more details), as a cut when projecting an Ntuple or n-dimensional histogram (see section 4.86.6 on page 131 (NTUPLE PROJECT)), as a function which is called for every event of an Ntuple (see section 4.86.7 on page 134 (NTUPLE SCAN)), or as a subroutine which you can just call (see section 4.11 on page 56 (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 4.16 on page 57 (COMIS)), although the `CALL_COMIS` command is probably better for all the purposes I have been able to think of.

See section 4.86 on page 128 (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. You can find documentation on most of the CERNLIB routines via the CERNLIB web pages:

```
'http://wwwinfo.cern.ch/asd/cernlib/libraries.html' and
'http://wwwinfo.cern.ch/asdoc/cernlib.html'.
```

The following routines are known and therefore can be used:

```
Mn_Fit:
  XMNCLC, XMNCNT, XMNFRG, XMNHIS, XMNRD3, XMNRES, XMNC2D, XMNDFUN
  RMNCLC, RMNCNT, RMNFRG, RMNHIS, RMNRD3, RMNRES, RMNC2D, RMNDFUN
  SMCTRL, SMQHAN, SMRMED, SMRMEN, SMSORT
MINUIT:
  MNEMAT,MNERRS,MNSTAT
HBOOK:
  HBOOK1,HBOOK2,HBOOKN,HFILL,HF1,HPRINT,HDELET,HRESET
  HFITGA,HFITPO,HFITEX,HPROJ1,HPROJ2,HFN,HGNPAR
  HROPEN,PAOPEN,PACLOS,PAREAD,PAWRIT,HCDIR,HGIVEN
  HPAK,HPAKE,HUNPAK,HGIVE,HGN,HGNF,HF2,HFF1,HFF2
  HMAXIM,HMINIM,HMAX,HMIN,HSUM,HNORMA,HREND
  HI,HIE,HIX,HIJ,HIF,HIDALL,HNOENT,HX,HXY
  HRIN,HROUT,HCOPY,HBPROF,HOPERA,HIDOPT,HDERIV,HGFIT
  HEXIST,HRGET,HRPUT,HSCR,HFIND,HCX,HCXY,HLABEL
  HBPROX,HBPROY,HBANDX,HBANDY,HBSLIX,HBSLIY
  HBOOKB,HBSTAT,HDIFF,HUNPKE,HREBIN,HERROR,HPROF2
  HOUTPU,HERMES,HISTDO,HFUNC,HXI,HIJXY,HXYIJ,HFINAM
  HSTATI,HLPOS,HFC1
  HSPLI1,HSPLI2,HMDIR,HLDIR,HRDIR,HLOCAT,HFITH,HFITV
  HTITLE,HBFUN1,HBFUN2,HRNDM1,HRNDM2,HBARX,HBARY
  HBNT,HBNAME,HBNAMC,HFNT,HFNTB,HGNT,HGNTF,HGNTV,HBSET
  HGNTB,HNBENT,HVXIST
HPLOT:
  HPLOT,HPLSYM,HPLERR,HPLEGO,HPLNT,HPLSUR,HPLSOF,HPLFRA
  HPLABL,HPLSET,HPLGIV,HPLOC,HPLTOC,HPLNEW,HPLOPT
KUIP:
  KUGETV,KUDPAR,KUVECT,KILEXP,KUTIME,KUEXEL,KUPROS
  KUNWG,KUCMD,KUGUID,KUNDPV,KUPAR,KUPVAL,KUACT
HIGZ:
  IPL,IPM,IFA,IGTEXT,IGBOX,IGAXIS,IGPIE,IGRAPH,IGHIST
  IGARC,IGLBL,IGRNG,IGMETA,IGSA,IGSET,IRQLC,IRQST,ISCR
  ISELNT,ISFAIS,ISFASI,ISLN,ISMK,ISVP,ISWN,ITX,ICLRWK
```

```
    IGPAVE,IGTERM
  ZEBRA:
    FZIN,FZOUT,FZFILE,FZENDI,FZENDO,MZLOGL
    RZCDIR,RZLDIR,RZFILE,RZEND,RZIN,RZOUT,RZVIN,RZVOUT
    RZOPEN,RZIODO,RZCLOS,RZQUOT
  CERNLIB:
    VZERO,UCOPY,RNDM,RANNOR,LENOCC,CLTOU,CUTOL
    SBIT0,SBIT1,SBYT,JBIT,JBYT,UCTOH,UHTOC,TIMED
    ERF,ERFC,FREQ,PROB,RANLAN
    DENLAN,DSTLAN,DIFLAN,XM1LAN,XM2LAN
    BINSIZ,DERF,DERFC,DFREQ,GAMMA,DGAMMA
    RANMAR,RMARIN,RMARUT,RMMAR,RMMAQ,RANECU,RANECQ
    RANLUX,RLUXGO,RLUXAT,RLUXIN,RLUXUT,RNORML,RNORMX
    FUNLXP,FUNLUX,RADAPT,RGS56P
```

1.17 Time

Use the `SET X MODE DATE` or `SET X MODE TIME` commands to plot things as a function of date or time. For the date the day will be printed as the scale and the month will be shown below the scale.

Note that the command `SET X MODE DATE|TIME` has more effect than the usual `INTEGER`, `REAL` or `LOG` modes. If `DATE|TIME` mode is selected plot limits must also be given as a date or time. If you project an Ntuple, then the x-axis is also assumed to be in date or time mode.

See section 1.18 on page 27 (Y2K) for details on how I cope with the 21st century in Mn_Fit. In general years less than 80 are assumed to be in the 21st century. Mn_Fit can only cope with 2 digit years, i.e. 99 or 02. If you have a date in an Ntuple stored as YYYYMMDD single precision is not sufficient to reliably get the month and day correct.

It is possible to make plots as a function of date and/or time. The easiest way to read in and store something as a function of time is to use the `DAT_FETCH` command or to store the data in an Ntuple and then use one of the date/time functions. The time can be given in a number of different forms:

```
DATE_TIM     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 VMS, but must be the full CHAR*23 format on any other computer. 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`.

Internally the plots are then stored in terms of the numbers of days (hours, minutes or seconds) since the reference time or since the 1st point if you have not specified a reference time.

The reference date/time is only used if you have set the axis mode to the same as the time mode:

```
SET X MODE DATE       and SET TIME DAY  yymmdd
SET X MODE TIME       and SET TIME HOUR yymmdd.hhmmss
```

Otherwise the reference time is set to 0 and the axis mode is used to specify how the plot should be stored.

If you store the date/time in an Ntuple and want to make a plot as a function of date or time use one of the date/time expressions (`DATE, TIME, DATE_TIM, DATE_MIN, TIME_MIN`). For more details on these functions see section 1.8 on page 14 (Expressions). Note that you should avoid giving the Ntuple variables the same name as one of the functions, as Mn_Fit may well get confused! The `VAXTIME` form is not available for Ntuples.

If you want to make an HBOOK histogram and then plot it as a function of date you should book it in terms of the numbers of days since 1-Jan-1980 00:00. For plots as a function of time book it in terms of the number of hours since 1-Jan-1980 00:00.

At present reading in plots as a function of date and/or time only works for the x-axis. While you can specify date or time mode for other axes it is not fully implemented.

1.17.1 Examples

1. You have an Ntuple which contains dates, times and temperature. You want to plot the temperature vs date from the beginning to the end of 1995. Here is a simple ASCII file (`file.mnd`), but you could store the same thing in an HBOOK Ntuple:

```
id 1
title Temperature vs. time
ntuple 3 ndate ntime temp
data
950203 120000 13
950303 080000 -4
950503 040000 0
950603 165700 19
950703 201000 30
950903 120000 18
end
```

   You read in the file and make and plot a projection:

```
dat_fetch file.mnd
set x mode date
project 1 date_tim(ndate,ntime) temp &1 0 950101 960101
! Plot from 1-Jan-1995 to 1-Jan-1996
plot 1&1
! Plot from 1-Jan-1995 to 1-Jun-1995
set x lim 950101 950601
plot 1&1
```

1.18 Y2K

For Mn_Fit the magic year is 1980! In most of the code I stick with a 2 digit year. If the year is less than 80 I put it in the 21st century. If it is more than 80 it is assumed to be in the 20th century. In addition a year such as 101 should be treated at 2001 in most cases. Unfortunately I cannot cope properly with 4 digit years. You should use 020410 and not 20020410 to specify the 10th April 2002. The reason is that Mn_Fit internally uses real numbers and single precision is not sufficient for an 8 digit date.

Note that for the L3 database interface year 2000 MUST be given as 100 etc.

If histograms have been or are created in 2000 or later using CERN library versions 98 or earlier, then the date within the histogram file will be screwed up. If Mn_Fit has been compiled with a 1998 or earlier version of the CERN library, you may also have problems with the display of dates.

1.19 Root

Mn_Fit can fetch ROOT histograms so that its fitting (and plotting) features can also be used in conjunction with ROOT. The interface is very new and so probably has a number of bugs and I am sure that several necessary checks for errors are not yet made. Please also look out for memory leaks.

I only plan to support the reading in of histograms (1-D, 2-D and 3-D) as well as profile plots (3-D histogram support will be added soon). I do not plan to make it possible to read in Ntuples - this is too much work. Create your histograms from Ntuples within ROOT and then read them in with Mn_Fit.

It seems as if the Mn_Fit executable including root is not really transportable, so you should compile Mn_Fit yourself if you want to use this feature. The environment varibale `ROOTSYS` must be set before you run the Makefile. The Mn_fit startup script will set `ROOTSYS` to whatever it was when the executable was made.

In order to use the ROOT interface Mn_Fit must have been compiled with the ROOT option turned on. Use the command `make all_root` to make both nomral and ROOT interface excutables. The environment variable `ROOTSYS` must also be set before starting Mn_Fit.

1.20 Vectors

While vectors as such do not exist in Mn_Fit, you can use registers and/or a series of points for almost all such purposes. A series of x and y values can be read in from a table using the `DAT_FETCH` command. If you want to store a series of x and y values, e.g. the results of a fit to a series of histograms, book an unbinned histogram and then fill it with the results.

1.20.1 Examples

1. Fit a series of histograms 1&1 to 1&20 with a Gaussian and plot the area as a function of the histogram number:

```
fun del 0
fun add gaus s
1000 100
0 0.5
1 2
! Book the results plot
book/unbin/err 1 'Fit results' 1 20
do i=1,20
  fit/like 10&@i
  minimize
  display
  fill 1 @i p1(1) 0.5 err1(1)
enddo
! Set the limits of the result histogram properly
```

```
part 1 0 21
plot 1
```

1.21 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` (VMS) or by the directory `$MN_FIT/help` (Unix). 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` (Unix) or `exec mn_fit_help:filename` (VMS):

```
demonn.mnf        - Demonstration macro number nn (nn = 01, 02 ...)
symbol.mnf        - Shows the available symbols
font.mnf          - Shows a sample of the HIGZ Postscript fonts
font_gksgral.mnf  - Shows a sample of the GKSGRAL fonts
hatch.mnf         - Shows HIGZ hatchings
hatch_gksgral.mnf - Shows GKSGRAL hatchings
pattern.mnf       - Solid and hollow fill + grey shading
key.mnf           - Different styles of keys to describe the plot symbol
ps_char.mnf       - All the Postscript characters
```

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

```
* Unix
  file $MN_FIT/cmz/mn_util -r
  sel UNIX
* VMS
  file mn_fit_cmz:mn_util -r
  sel VMS
*
  set *.mnf -d
  ctot -y demo
```

A series of test files are available in the directory referred to by the by `$MN_FIT/test` (Unix) or the logical name `mn_fit_test` (VMS). 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 first give the command `exec $MN_FIT/test/init_test` (Unix) or `exec mn_fit_test:init_test` (VMS) and then `exec $MN_FIT/test/filename` (Unix) or `exec mn_fit_test:filename` (VMS). 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 extracting and plotting HBOOK slices etc.
time_test.mnf    - Tests plotting vs. date or time
fit_test.mnf     - Tests the fitting
fun_check.mnf    - Checks many of the Mn_Fit built-in functions
frag_test.mnf    - Tests fragmentation functions
dipi_test.mnf    - Tests the dipion invariant mass functions
```

```
char_test.mnf    - Tests parsing commands, plus comments and continuations
do_test.mnf      - Tests DO loops
if_test.mnf      - Tests if blocks
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
ntuple_test.mnf  - Tests operations on Ntuples without cuts
cut_test.mnf     - Tests cut and project operations
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.mnf    - Tests calculations, use of registers, parameters etc.
var_test.mnf     - Tests defining user variables
smooth_test      - Tests smoothing of histograms
spline_test.mnf  - Test spline fitting
draw_test.mnf    - Tests drawing lines etc.
symb_test.mnf    - Tests the available symbols
pattern_test.mnf - Shows the available patterns (DECGKS, DEC GKS3D)
font_test.mnf    - Tests out changing fonts
pfont_test.mnf   - Tests Postscript fonts
```

Note that `plot_test`, `fit_test`, `ntuple_test` and `cut_test` run many other macros that test different aspects of plotting, fitting and Ntuple handling.

If you are not at Cornell, CERN or DESY and do not have the files please ask me to send them to you. 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. See:

```
http://www-zeus.physik.uni-bonn.de/~brock/mn_fit/pro/mn_fit.doc
```

for details of what to expect.

1.22 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 = DECGKS, DG = DEC GKS-3D, DI = DI3000, X = X Windows, 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.

In X Windows you can specify the display in the following ways when you are prompted inside Mn_Fit. However this does not work for Decnet communication on VMS:

```
<CR>          uses the DISPLAY name and line 10 of higz_windows.dat
n.hostname    DISPLAY is set to hostname and added to higz_windows.dat
n             DISPLAY is taken from line n of higz_windows.dat if available
n.            DISPLAY uses environment variable DISPLAY and line n of
              higz_windows.dat
```

For the X windows version you can define the DISPLAY variable before you start Mn_Fit or in the command line (using `-n hostname` (Unix) or `-n hostname -t transport -s server` (VMS)). The transport is usually either decnet or tcpip (for Multinet tcpip). The host name is either the decnet name or the tcp/ip name. If you specify the display in one of these ways just hit <CR> when prompted for the screen device. Line 10 of `higz_windows.dat` will be used to set the display size. If you start the X windows version and the environment variable `DISPLAY` is not defined and if you do not give the `hostname` then `` `hostname` `` (Unix) will be used. Under VMS if the logical `decw$display` is not defined, then the display will be set to `sys$node`.

`hostname` is of the form `host:m` where `host` is the host name (e.g. `hpl3cmu3.cern.ch`) and `m` is the display number (usually 0). The `higz_windows.dat` file is searched for in your current directory and then in your home directory. If it does not exist it is created in your home directory. It contains 10 lines with the format `4(1X,I4),1X,A`. The four numbers are the x and y sizes of the display and the offset from the top left corner (in pixels). The character string is the `DISPLAY` name. `higz_windows.dat` is initially created with display sizes of 600 pixels and 0 offsets. If you are logged directly onto a computer or use `ssh` to connect to another computer `DISPLAY` should always be set correctly.

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.
Display     X            X Windows Display
Xterm       X            Xterm Tektronix window
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/P       Decwindows input and output device
Motif       V/DG         Motif 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/X        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
```

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.

1.23 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 = DECGKS, DG = DEC GKS-3D, DI = DI3000, X = X Windows, 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 HIGZ/GKS versions). This command is also useful if you run Mn_Fit in a script and want to output all pictures to a file. Use the sequence of commands:

```
mn_fit
  none
  set hard filename.ps
  capture postscript
  fetch ...
  plot ...
  close
  exit
```

If you are reading commands from a file and omit the device a `METAFILE` will be assumed. Postscript (Landscape, Portrait and Encapsulated) is now always HIGZ Postscript. To get other Postscript interfaces precede the device name with G (GKSGRAL) or D (DECGKS), e.g. GPostscript. The following hardcopy devices are supported:

```
Metafile     G/V/DG        plot.meta  GKS Metafile
Postscript   All           plot.ps    Postscript file (portrait)
EPostscript  G/V/DG/X      plot.eps   Encapsulated Postscript file
LPostscript  G/V/DG/X      plot.ps    Postscript file (landscape)
CPostscript  G/X           plot.ps    Colour Postscript file (portrait)
CLPostscript G/X           plot.ps    Colour 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)
```

If you try to put more than one page into an encapsulated postscript file, then the pages will each be put into separate files with the names `file.eps, file.eps_1, file.eps_2` etc.

You can set the paper size using the `SET PAPER` command.

1.24 Condition Handler

Mn_Fit uses the KUIP condition handler which traps errors and `CTRL/C` interrupts. If an error is detected control is returned to the `MN_CMD>` level. This means that overflows and any other arithmetic errors should not cause the program to exit. It will return to the `MN_CMD>` level instead. WARNING: A `CTRL/C` during FORTRAN I/O can cause strange effects, particularly on VMS. To suppress the output during an `INDEX` command for example use `CTRL/O` and then you can try `CTRL/C` with care.

If you interrupt with `CTRL/C` inside MINUIT, Mn_Fit sometimes gets confused and refuses to start fitting again, so you have to exit and restart.

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 44 (MINUIT).

EXIT	Exits from program or back to previous level.
2DIM	Plots 2-D histograms in various ways.
ADD	Adds 2 histograms together.
ALIAS	Defines an alias.
UNALIAS	Undefines one or all aliases.
ATTACH	Attaches you to another process (VMS only).
AVERAGE	Makes a weighted average of the contents of 2 histograms.
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 or ROOT directory immediately.
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 4.19 on page 59 (CUT) for more).
NO_CUT	Deletes all cuts.
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 entry.

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 or ZEUS display version.

DIVIDE Divides 2 histograms.

DO Starts a `DO` loop (only available inside macro's or `DEFINE`d commands).

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.

ELIF Else if in an `IF` block.

ELSE Else in an `IF` block.

ENDDO Terminates a `DO` loop.

ENDIF End of an `IF` block.

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.

FI End of an `IF` block.

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 4.48 on page 89 (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.

HINDEX Gives the HBOOK index for histograms currently in memory.

HISTOGRAM Plots, overlays or dumps a histogram (see section 4.59 on page 110 (HISTOGRAM) for more information).

HISTORY	History of commands read by GNU readline (cannot be abbreviated and must be in lowercase).
HMAKE	Makes an HBOOK histogram from another sort.
HMERGE	Merges the HBOOK histograms from one or more RZ files.
HPRINT	Does an HPRINT on the given histogram.
HRENAME	Renames an HBOOK histogram.
HRECOVER	Calls HRECOV to recover an Ntuple in a RZ file.
HY_FETCH	Fetches one or more HYBRID histograms.
IF	Conditional control of statement execution.
IGTABLE	Plots 2-D histograms in various ways (almost the same as `2DIM`).
INDEX	Gives an index of the histograms read in.
INQUIRE	Gets the value for a macro parameter (only available inside macros or DEFINEd commands).
INTEGRATE	Integrates a function over a range.
KEY	Gives a key for the meaning of symbols.
LDIRECTORY	Lists the contents of the current HBOOK or ROOT directory.
LEGO	Makes a lego plot of a 2-dimensional histogram.
LS	Lists the contents of the current HBOOK or ROOT directory.
MADD	Adds a series of histograms together.
MDIRECTORY	Makes a new HBOOK directory in memory.
MESSAGE	Writes a message to the current output device.
MN_FETCH	Reads in plots in Mn_Fit format.
MN_STORE	Stores plots in Mn_Fit format.
MSUBTRACT	Subtracts a series of histograms from another histogram.
MULTIPLY	Multiplies 2 histograms.
NORMALIZE	Normalizes the total contents of a histogram.
NTUPLE	Operations on Ntuples and multi-dimensional histograms (see section 4.86 on page 128 (NTUPLE) for more).
OPEN	Opens an HBOOK version 4 histogram file.
OVERLAY	Overlays one histogram on another (alias for `HIST OVERLAY`).
PARTITION	Cuts out part of a histogram.
PARSE	Parses and executes a command line with formatting.
PLOT	Plots a histogram (alias for `HIST PLOT`).
2DIM	Plots 2-D histograms in various ways.
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`).
PWD	Prints the current HBOOK or ROOT directory.

QUIT	Emergency stop of Mn_Fit.
READ	Reads in commands or data (see section 4.95 on page 141 (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.
ROOT_FETCH	Fetches one or more ROOT histograms.
ROOT_OPEN	Opens a ROOT histogram file.
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 4.106 on page 145 (SET) for more).
SHELL	Executes a single DCL command or spawns a subprocess (VMS). Executes a shell command or starts a new shell (Unix).
SHOW	Shows plot drawing parameters (see section 4.108 on page 180 (SHOW) for more).
SMOOTH	Smooths a histogram.
SPAWN	Executes a single DCL command or spawns a subprocess (VMS). Executes a shell command or starts a new shell (Unix).
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.
TEXT	Adds text to a picture or plot - not yet implemented.
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.
WDIRECTORY	Sets the current working directory for opening files.
WRITE	Writes out a list of commands or data points.
ZDIRECTORY	Makes an RZLDIR of the current HBOOK directory.

2.2 Menu of Commands inside CUT

The following is the list of `CUT` commands. For more details see section 4.19 on page 59 (CUT) and the individual subtopics:

NEW	Specifies a new cut.
NAME	Specifies or changes a named 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 DRAW

The following is the list of `DRAW` commands. For more details see section 4.33 on page 73 (DRAW) and the individual subtopics:

ARC	Draws an arc of a circle.
ARROW	Draws an arrow.
BOX	Draws a box.
CIRCLE	Draws a circle.
ELLIPSE	Draws an ellipse.
GLUON	Draws a gluon line.
LINE	Draws a line.
POLYGON	Draws a polygon.
POLYLINE	Draws a polyline.
SEGMENT	Draws a segment of a circle.
SINE	Draws a sine wave.
SYMBOL	Draws a symbol.
TRIANGLE	Draws a triangle.
CHANGE	Changes a drawn item.
DELETE	Deletes one or more drawn items.
LIST	Lists the items to draw.
FETCH	Fetches a drawing.
STORE	Stores a drawing in a file .
END	End of drawing.

2.4 Menu of Commands inside FUNCTION

The following is a list of the available commands. For more details see section 4.48 on page 89 (FUNCTION) and the individual subtopics:

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.

`SET FUNCTION` has a number of subcommands that control how the functions are used. See section 4.106.33 on page 154 (SET FUNCTION) for more details:

AREA	Specifies whether the AREA is that inside the plot or over the whole valid range.
BIN_WIDTH	Controls whether the bin width is used when calculating the value of a function.
POINTS	Sets the number of points used when drawing a function.
INTEGRATE	Sets the number of intervals used when integrating a function.
STEP	Sets the step size used for differentiation of a function.

2.5 Menu of Commands inside HISTOGRAM

The following is a list of the available commands. For more details see section 4.59 on page 110 (HISTOGRAM) and the individual subtopics:

BOOK	Books a new plot inside Mn_Fit.
DISPLAY	Makes an L3 or ZEUS 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.
IGTABLE	Interface to HIGZ IGTABL routines for 2-D histograms.
LEGO	Makes a lego plot of a 2-D histogram.

OVERLAY	Overlays a histogram on the present plot.
PLOT	Plots a histogram.
SURFACE	Makes a surface plot of a 2-dimensional histogram.
2DIM	Interface to HIGZ IGTABL routines for 2-D histograms.

2.6 Menu of Commands inside NTUPLE

The following is a list of the available commands. For more details see section 4.86 on page 128 (NTUPLE) and the individual subtopics:

DUMP	Dumps the contents of an Ntuple.
EPROFILE	Makes a profile plot from an Ntuple - errors are error on mean.
FILTER	Filter an Ntuple by applying cuts.
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.
SPROFILE	Makes a profile plot from an Ntuple - errors are r.m.s. spread.

2.7 Menu of Commands inside READ

The following is a list of the available commands. For more details see section 4.95 on page 141 (READ) and the individual subtopics:

COMMAND	Reads a series of commands and executes them from a file.
DATA	Reads data points or Ntuples from an ASCII file.

2.8 Menu of Commands inside SET

The following is a list of the available commands. For more details see section 4.106 on page 145 (SET) and the individual subtopics:

EXIT	Exits from SET (deprecated).
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.
ABORT	Turns on or off the aborting of macros when an error occurs.
ALIAS	Turns on or off alias translation.
AUTOFETCH	Automatically refetch Ntuples when they are projected or scanned, or fetches histograms when they are plotted.
AUTOSCALE	Scale text sizes when windowing.
AUTOSWITCH	Switches between plotting and alphanumeric without asking for a <CR>.

AUTOTRIM Do not draw the last scale value when windowing with 0 spacing between the windows or for the 1st y-axis scale value in a lego plot.

AXIS Sets axis labelling text, position, size and angle (deprecated).

BACKGROUND Specifies which functions should be considered as background.

BIN Sets an scale factor or offset for each bin when plotting.

BOX Turns on or off drawing a box round the plot.

BREAK Turns on or off the condition handler.

CHARACTER Sets continuation and comment characters.

COLOR Same as `SET COLOUR` for Americans.

COLOUR Changes the colour of all or part of a picture.

DBASE Sets database access parameters.

DEBUG Turns on/off the debug flag.

DEFAULT Sets parameters back to default values or sets the default plot.

DIRECTORY Sets the current HBOOK4 or ROOT directory.

DISPLAY Changes the mode or defaults for displaying fit results.

DSIZE Sets the scale factor for the dot size.

DUMP Changes the output unit for the printout from 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 (see section 4.106.28 on page 152 (SET FIT) for more details).

FONT Sets the font for text in pictures.

FOOTER Turns on or off the footer on a picture and allows a user footer to be specified.

FRAME Sets where you want a frame round the plot.

FSIZE Sets the text size of the footer.

FUNCTION Specifies options for calculating function values (see section 4.106.33 on page 154 (SET FUNCTION) for more details).

GRID Turns on or off a grid.

GSIZE Sets the global/user title size.

HATCH Sets the hatching number.

HARDCOPY Sets the name of the file that hardcopies are written to.

HEADER Turns on or off displaying the histogram identifiers and and other information on the picture.

HIGZ Sets IGSET or IGTABLE parameters.

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.

IGARC Uses HIGZ IGARC routine for arcs and circles.

IGTABLE Sets the IGTABLE parameters for `IGTABLE` or `2DIM` command.

LABEL Sets axis labelling text, position, size and angle.

LOG Turns on or off logging of all commands - for a log scale in a histogram see section 4.106.53 on page 160 (SET MODE).

LSIZE Sets the segment length for drawing lnes.

MANUAL Turns on or off special command echoing used to make Mn_Fit manual.

MOUSE Turns on or off using the mouse to position comments/keys.

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.

PAGER Gives the command to invoke your favourite pager.

PAPER Sets the paper size for Postscript files.

PARAMETER Sets parameters for the L3 detector displays.

PATH Sets the list of directories searched when opening existing files.

PATTERN Sets the pattern number (only works with GKS).

PI Turns on or off the ability to use pi as a scale symbol.

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.

ROTATION Rotates a plot - not yet fully implemented.

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 Unix systems.

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 the number of ticks and where to put them.
TIME	Sets the time mode for storing data and the reference time.
TITLE	Sets the user title or changes the title position or size for a histogram.
TKTCL	Turns on/off TK/TCL interface.
TSIZE	Sets the title size.
USIZE	Sets the size for printing the fit results.
WAIT_CR	Turns on or 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.
WORKING_DIR	Sets the current working directory for files.

Parameters specific to an axis. The commands FRAME, TICKS, SCALE and LABEL|AXIS can either be preceded by the axis name X, Y or Z or followed by the postion on the picture that they refer to ALL, LEFT, RIGHT, BOTTOM, TOP or VERTICAL. The other commands must be preceded by the axis name:

AXIS	Sets axis labelling text and position, size and angle.
GRID	Turns on or off a grid.
LABEL	Sets axis labelling text and position, size and angle.
LIMITS	Sets the lower and upper plotting limits for axis.
MARGIN	Sets size of lefthand or lower margin.
MODE	Sets mode to draw scale, Real, Integer, Log, or Time.
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 picture.
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.

2.9 Menu of Commands inside SHOW

The following is a list of the available commands for which there are not SET equivalents. For more details see section 4.108 on page 180 (SHOW) and the individual subtopics:

ALL	Shows everything there is to show.
ALIAS	Lists the currently defined aliases.
CHAR	Lists contents of one or more character array elements (0-100).
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 or ROOT directory.
EXCLUSIONS	Lists the parts of histograms excluded or included in the fit.
FILES	Lists the filenames for hardcopy, dump and currently open.
FLAGS	Lists the logical flags that control what is plotted.
FRAME	Lists the parameters affecting the scale and frame.
INCLUSIONS	Lists the parts of histograms included or excluded in the fit.
KEYS	Lists the current keys.
LABELS	Lists the picture labels.
LIMIT	Lists the parameters affecting the scale and frame.
LOG	Lists the last 100 commands given.
MODE	Lists the parameters affecting the scale and frame.
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 (0-500).
SCALE	Lists the parameters affecting the scale and frame.
SEGMENTS	Lists the segments currently in use (GKS version, mainly for experts).
SIZES	Lists the parameters relating to plot sizes, margins, windows.
TICK	Lists the parameters affecting the tick marks.
UNITS	Lists FORTRAN units and who has grabbed them (mainly for experts).
VARIABLE	Gives the value of one or all user defined variables.

2.10 Menu of Commands inside WRITE

The following is a list of the available commands. For more details see section 4.124 on page 189 (WRITE) and the individual subtopics:

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

At one time there were 2 versions of the MINUIT package interfaced to Mn_Fit. The default is now the CERN library MINUIT, while the previous version was based on a much older version of MINUIT which was extensively modified by Chris Rippich. This older version is no longer supported.

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 following extra commands are defined when you are running MINUIT: `MODIFY`, `DISPLAY`, `EXCLUDE`, `INCLUDE`, `NO_EXCLUDE`, `NO_INCLUDE`, `DUMP`, `ITERATIONS`, `MAX_CALLS`, `PRECISION`, `FLOAT`, `INFO`, `FIT_INFO`, `BACK_SUB`, `CONSTRAIN`, `SCAN`, `STRATEGY` and `RELEASE`.

You can also 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 1.7 on page 11 (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 standard outupt.
PRECISION	Sets the convergence criteria.
PRINTOUT	Changes the printout level.
PUNCH	Saves the current parameter values in a file.
RELEASE	Floats a previously fixed variable (CERN version of MINUIT only).
RESTORE	Floats variable(s) which were fixed.
SAVE	Saves the current parameter values in a file
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.
CHI_PLOT	Plots the χ^2 vs. a variable even if you are doing a likelihood fit.
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
```

Emergency exit of Mn_Fit. This is useful if you get an error when trying to exit normally. Note that no files are closed if you use QUIT. The minimal abbreviation for this command is QUI to avoid accidental exits.

4.3 2DIM

```
Syntax: 2DIM option[/qual] id [&idb] [par1 par2 ...]
where: option  is the 2-dimensional histogram plotting mode.
       id      is the histogram identifier
       idb     is the (optional) secondary identifier
       par     are any parameters
```

Interface to the HIGZ IGTABL routines for plotting 2-dimensional histograms in different ways. This command is an alias for HISTOGRAM 2DIM. The 2DIM command is useful if you want to change the parameters on the command line. Use the command HISTOGRAM IGTABLE or IGTABLE if you have specified the option and any relevant parameters with the SET IGTABLE command. The SET IGTABLE and IGTABLE commands are useful if you want to make many plots with the same parameters.

The following options are available:

```
ARROW       Arrows with magnitude and direction proportional to
            rate of change.
BOX         Boxes with area proportional to the number of entries.
CHAR        Number of letter with number of entries.
COLOUR      Colour plot.
```

```
CONTOUR     Contour plot joining points with equal values.
SCATTER     Scatter plot.
TEXT        Number of entries drawn in each bin.
LEGO        Lego plot with different styles.
SURFACE     Surface plot with different styles.
```

For LEGO and SURFACE plot additional qualifiers can be given that specify the coordinate system to be used and the style of the plot. The possible coordinate systems are:

```
POL         Polar coordinates,
CYL         Cylindrical coordinates,
SPH         Spherical coordinates,
PSD         Pseudo-rapidity coordindates.
```

Some of the options allow qualifiers which further control the style. See the LEGO and SURFACE subtopics for the other options on colour and shading which only apply to such plots. Fine control of the contours is also possible. If you want to make a contour plot with solid colours you can use the SURFACE/C2 command.

There are also a number of options available for drawing 2-dimensional histograms using the normal PLOT command and setting the appropriate symbol number. See section 4.106.83 on page 174 (SET SYMBOL) for more details.

The normal procedure is to FETCH the histograms from a file and then plot them using the above syntax. You can also plot histograms directly from an HBOOK RZ file by opening the file (OPEN or HB_OPEN) and then giving the plot command. This feature can be suppressed using the SET AUTOFETCH OFF command. Note that this only works for single histograms and not for a range nor for all histograms.

4.3.1 ARROW

```
Syntax:  2DIM ARROW id[&idb]
```

Draws a 2-D histogram with an arrow for each bin. The size and magnitude of the arrow show the slope from each bin to its neighbour.

An alternative syntax is:

```
SET IGTABLE ARROW
IGTABLE id[&idb]
```

4.3.2 BOX

```
Syntax:  2DIM BOX id[&idb]
```

Draws a 2-D histogram with a box for each bin, the area of which is proportional to the number of entries.

An alternative syntax is:

```
SET IGTABLE BOX
IGTABLE id[&idb]
```

4.3.3 CHAR

```
Syntax:  2DIM CHAR id[&idb]
```

Draws a 2-D histogram with the a number or letter for the number of entries given for each bin. The font is set using the `SET FONT SYMBOL` command.

An alternative syntax is:

```
SET IGTABLE CHAR
IGTABLE id[&idb]
```

4.3.4 COLOUR

```
Syntax:  2DIM COLOUR[/qual] id[&idb] mode
where:   qual     Z means show the colour scale
                 NZ means do not show the colour scale (default)
         mode     0 means uses the standard 8 colours
```

Draws a 2-D histogram with colours for the number of entries. By default the scale is not given on the right-hand side. Use the qualifier `/Z` to turn the scale on.

You can change the size of the font on the colour scale using the command `SET Z SCALE = = size`.

An alternative syntax is:

```
SET IGTABLE COLOUR
IGTABLE id[&idb]
```

With this syntax the colour scale is not shown.

4.3.5 CONTOUR

```
Syntax:  2DIM CONTOUR id[&idb] nlevel mode [level1 level2 ...]
where:   nlevel   is the number of levels to show.
         mode     is the mode to use:
                  0     = Use colour to distinguish the contours
                  1.nnn = Use the lines type with colour n
                  2.nnn = Use the same line type and colour for all
                          contours
         level1   is the z-value for the contour
Default: nlevel=20, mode=0
```

Draws a 2-D histogram as a contour plot. To use the default number of levels give `nlevel` as 0. Different modes are available to distinguish between the contour lines. The numbers of the colours are the same as for the `SET COLOUR` command. Note that you have to give the full 3 digits for the colour, e.g. 1.004, to use the line type and draw the contour in blue. You can set the line type used for the contours with mode 2.nnn by using the command `SET HIGZ LTYP n`.

An alternative syntax is:

```
SET IGTABLE CONTOUR nlevel mode [level1 level2 ...]
IGTABLE id[&idb]
```

Use the `2DIM SURFACE/C2` command to make a contour plot with solid colours. See the examples for how this is done.

4.3.6 LEGO

```
Syntax:  2DIM LEGO[/qual] id[&idb] theta phi [ncol1 ncol2 ncol3 ...]
where:   qual      describes the style of the lego plot
         theta     is the theta viewing angle
         phi       is the phi viewing angle
         ncol1...  are the colours to use for the colour options
Default: theta=30, phi=30
```

Draws a LEGO plot. An alternative syntax is `LEGO/IGTABLE[/qual]`.

The possible qualifiers and their equivalent `SET IGTABLE` commands are:

```
/C1    LEGOC1   Colour lego plot - mode 1.
/C2    LEGOC2   Colour lego plot - mode 2 (most common).
/BAR   LEGOBAR  Reduced the bin width plot. Use the commands
                'SET HIGZ BARW frac' and 'SET HIGZ BARO frac' to
                adjust the bin width and offset.
/POL   LEGOPOL  LEGO plot in polar coordinates.
/CYL   LEGOCYL  LEGO plot in cylindrical coordinates.
/SPH   LEGOSPH  LEGO plot in spherical coordinates.
/PSD   LEGOPSD  LEGO plot in pseudo-rapidity coordinates.
/NFB            Do not draw the front box.
/NBB            Do not draw the back box.
```

One of the options `C1,C2,BAR` can be combined with one of the coordinate system options.

While any angles can be specified for theta and phi, theta between 0 and 90 degrees works best. If you want to look at the bottom of your surface, use a theta angle between 90 and 180 degrees.

You can specify your own colour scheme by giving the colour numbers. Use the `SET COLOUR REPRESENT ncol r g b colname` command to define new colours or redefine existing ones.

4.3.7 SCATTER

```
Syntax:  2DIM SCATTER id[&idb] marker maxpnt
where:   marker   is the marker type to use
         maxpnt   is the maximum number of random points in a cell
```

Draws a 2-D histogram with random points per bin, where the number of points is proportional to the number of entries.

An alternative syntax is:

```
SET IGTABLE SCATTER
IGTABLE id[&idb]
```

4.3.8 SURFACE

```
Syntax:  2DIM SURFACE[/qual] id[&idb] theta phi [ncol1 ncol2 ncol3 ...]
where:   qual     describes the style of the surface plot
         theta    is the theta viewing angle
         phi      is the phi viewing angle
         ncol1...  are the colours to use for the colour options
Default: theta=30, phi=30
```

Draws a SURFACE plot. An alternative syntax is `SURFACE/IGTABLE[/qual]`.
The possible qualifiers and their equivalent `SET IGTABLE` commands are:

```
/C1    SURFC1    Surface plot with colours and contour lines.
/C2    SURFC2    Surface plot with colours only.
/CONT  SURFCONT  Surface plot with colour contour plot on top.
/SHADE SURFSHADE Surface plot with shading.
/POL   SURFPOL   Surface plot in polar coordinates.
/CYL   SURFCYL   Surface plot in cylindrical coordinates.
/SPH   SURFSPH   Surface plot in spherical coordinates.
/PSD   SURFPSD   Surface plot in pseudo-rapidity coordinates.
/NFB             Do not draw the front box.
/NBB             Do not draw the back box.
```

One of the options `C1,C2,CONT,SHADE` can be combined with one of the coordinate system options.

The `2DIM SURFACE/C2 id 90 0` command can be used to draw a contour with solid colours. You can control the values for each colour by specifying the `Z LIMIT` and then giving a corresponding number of colours on the command line. See the examples for how this is done.

While any angles can be specified for theta and phi, theta between 0 and 90 degrees works best. If you want to look at the bottom of your surface, use a theta angle between 90 and 180 degrees.

You can specify your own colour scheme by giving the colour numbers. Use the `SET COLOUR REPRESENT ncol r g b colname` command to define new colours or redefine existing ones.

4.3.9 TEXT

```
Syntax:  2DIM TEXT id[&idb]
```

Draws a 2-D histogram with the number of entries given for each bin. The font is set using the `SET FONT SYMBOL` command.

An alternative syntax is:

```
SET IGTABLE TEXT
IGTABLE id[&idb]
```

4.3.10 Examples

1. Make a lego plot of 2-D histogram 10:

```
2dim lego 10 30 30
! Use colour option 2
lego/c2   10 30 30
```

2. Make a contour plot with contours at 1,4 and 9:

```
! Take default parameters
2dim contour 10 0
! 2dim command with parameters on the command line
2dim contour 10 3 1 1 4 9
! Using the set igtable command for plots 10 and 11
set igtable contour 10 3 1 1 4 9
igtable 10
igtable 11
```

3. Make a contour plot with solid colours using the SURFACE command. Note that the scale is not drawn properly here:

```
! Default colours
2dim surface/c2/nfb/nbb 10 90 0
! Specify my own colours
! Violet
set col rep 8 0.60 0.0 0.9 violet
! Orange
set col rep 9 1.0 0.5 0.0 orange
! Set limit as 0 to 10 and specify the 10 colours
! The standard colours are:
! 1 black
! 2 red
! 3 green
! 4 blue
! 5 yellow
! 6 magenta
! 7 cyan
set z lim 0 10
2dim surface/c2/nfb/nbb 10 90 0 7 7 7 9 8 4 3 2 1 7
```

4. Make a contour plot with solid colours using the SURFACE command. Draw the frame at the centre of the first bin rather than the edge, so that the whole plot is filled. Note that you could also plot the 2-D Gaussian using `FUNCTION PLOT`, rather than fitting it, as is done here. This example also shows how to use the registers containing the number of bins and the bin boundaries as well as those containing the corners of the plot:

```
fet $MN_FIT/help/tutorial/hbook_example.his 10
!
! Make a nice Gaussian by fitting the data
!
fun del 0
fun add 2dim sig
1000
0
0
1
1
fit/like 10
mini
display
exit
! Define some new colours
! Violet
set col rep 8 0.60 0.0 0.9 violet
! Orange
set col rep 9 1.0 0.5 0.0 orange
!
! Draw the Gaussian without any frame
!
set frame all off
! Set the z limit and give the 10 colours
set z lim 0 10
```

```
! The 2-D Gaussian is in plot 10&981
2d surface/c2/nbb/nfb 10&981 90 0 7 7 7 9 8 4 3 2 1 7
!
! Move the scale and frame half a bin inside the plot
!
! Plot coordinates - these are set with the set plot default command
! Use the bin boundaries and the number of bins
set plot 10&981 default
dep r01 = r205 + 0.5*(r133-r132)/r131
dep r02 = r206 - 0.5*(r133-r132)/r131
dep r03 = r207 + 0.5*(r138-r137)/r136
dep r04 = r208 - 0.5*(r138-r137)/r136
dep r05 = r02 - r01
dep r06 = r04 - r03
!
! In cm - these are only set once the histogram has been plotted
dep r11 = r201 + 0.5*(r202-r201)/r131
dep r12 = r202 - 0.5*(r202-r201)/r131
dep r13 = r203 + 0.5*(r204-r203)/r136
dep r14 = r204 - 0.5*(r204-r203)/r136
dep r15 = r12 - r11
dep r16 = r14 - r13
dep r17 = r11 - r201
dep r18 = r13 - r203
!
! The number of bins is effectively reduced by 1
dep r21 = r131 - 1
dep r22 = r134 - 1
book/bin/noerr 9999 'Frame' 2 r21 r01 r02 r22 r03 r04
set x lim r01 r02
set y lim r03 r04
!
! Move the frame in by half a bin and make it one bin smaller
set x wmarg r17
set y wmarg r18
set x wsize r15
set y wsize r16
!
! Draw the frame
!
set frame all on
set title off
plot/noclear/empty 9999
```

4.4 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.

4.5 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.

Aliases can be up to 20 characters long and translate into strings that are up to 80 characters long.

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 to show the translation use `SHOW ALIAS @name`.

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

4.5.1 Examples

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

```
alias junk show
junk all                -> 'show all'
show alias all
unalias @junk
```

   Note the use of `@junk` to remove the alias.

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

```
alias name1 'John Smith'
alias name2 son
message name1//name2  -> 'John Smithson'
```

```
alias name  'Jane Jones'
message name1's real @name is name
                      -> 'John Smith's real name is Jane Jones'
set alias off
message What is your name?
                      -> 'What is your name?'
```

You can now remove the alias with `unalias name`.

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 VMS.

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 AVERAGE

```
Syntax: AVERAGE 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
        idb3      is the (optional) output secondary identifier
```

Makes a weighted average of the contents of 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).

To average 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, `AVERAGE 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.

4.9 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 4.59.1 on page 111 (HISTOGRAM BOOK) for more details.

4.10 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 4.28 on page 69 (DEPOSIT) for more details.

4.11 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.

4.12 CAPTURE

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

Captures a new output device for the plots. See section 1.22 on page 30 (Screen Devices) and Hardcopy_Devices 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`.

4.13 CDIRECTORY

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

Sets the current HBOOK or ROOT directory name to `dirname`. Note that the HCDIR command is executed immediately, whereas it is only executed when you do `FETCH`, `LDIR`, `ZDIR`

or SHOW DIR after a SET DIR command. You should therefore always use SET DIR before a FETCH command.

Note that the directory name should not be preceded by a '/'. However '//' is allowed so you can go to the top level. You can use either \ or '..' to go up a directory level.

ROOT directory names are always with respect to the top level directory. Therefore it make no sense to try to use '..' to go up a level. Use the command 'cd //root' to get to the top level. ROOT directories are case sensitive. HBOOK directories always get converted to upper case.

Note that the Mn_Fit HBOOK file top level directory name is //MN_HBIN, while the ROOT top level directory name is '//root'.

4.14 CLEAR

```
Syntax: CLEAR
```

Clears the currently selected screen device. It also deletes any items to be drawn and existing comments and keys. The command is most useful for drawing things without using the PLOT command.

Note that in order to use registers > 200, which contain the plot corners in cm and plot coordindates, you first have to draw something.

4.15 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. On Unix machines the HARDCOPY command will overwrite an existing file. Use the SET HARDCOPY command to change the filename.

4.16 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 1.16 on page 24 (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.

4.17 COMMENT

```
Syntax: 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 centered
                  RIGHT  Comment is right adjusted (default)
mode      can be: CM   = Position is in cm (default)
                  PLOT = Position in terms of plot coordinates
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, 2DIM, LEGO, SURFACE, DISPLAY or OVERLAY/DIFF commands - OVERLAYS on the same scale do not count), the syntax is the first line. 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 formatting text and the other fonts available see section 1.12 on page 18 (Text). The default colour and thickness of the comments are the same as the title.

If you use the option SET MOUSE ON, then the position of the comment is specified by the mouse. Move the mouse to where you want to have the comment and click on the left button. To keep the comment in the same position (if you are changing it) click on the right mouse button. Note that using the mouse only works properly in CM mode.

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.

4.18 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.

Note that COPY only copies the Mn_Fit histogram and not any associated HBOOK histogram. Use the HCOPY command to copy the HBOOK histogram also.

4.19 CUT

```
Syntax: CUT
     or CUT command
where:      command is one of ?|NEW|NAME|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` or `CUT NAME` 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 4.86.6 on page 131 (NTUPLE PROJECT) for more details).

For a short description of the commands see section 2.2 on page 37 (CUT Menu). See the individual topics for more information.

WARNING: In versions before 4.07/30, if you combined simple cuts (where the expression2 (see section 4.19.1 on page 59 (CUT NEW)) was a number or a register) with more complicated ones that had to be parsed (Ntuple variables, or arithmetic expressions), and the more complicated expression was specified later in the `CUT USE` command the cut value of another cut could be overwritten.

4.19.1 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 <, <=, >, >=, =, <>
        expression2  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 1.8 on page 14 (Expressions).

Note that you cannot use the form `x.lt..4` for a cut, although `x<.4` is OK. In general it makes it easier if you leave spaces between the expressions and the conditions. This problem may be fixed in a future version of Mn_Fit.

If your cut expression starts with a parenthesis then you should enclose the whole expression in parentheses:

```
cut new ((a+b)/(c+d))
```

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.

4.19.2 NAME

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

Specifies or modifies a named cut. The name of the cut can be use later with the `CUT USE` command. 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 1.8 on page 14 (Expressions).

The name of the cut should start with a letter. All names will be converted to upper case and only the first 10 letters are significant. If the name already exists you will modify an exisiting cut. If the name does not exist it will be added as a new cut. Names cannot be abbreviated in either the `CUT NAME` or the `CUT USE` commands.

Note that you cannot use the form `x.lt..4` for a cut, although `x<.4` is OK. In general it makes it easier if you leave spaces between the expressions and the conditions. This problem may be fixed in a future version of Mn_Fit.

If your cut expression starts with a parenthesis then you should enclose the whole expression in parentheses:

```
cut new ((a+b)/(c+d))
```

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.

4.19.3 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 == if you want to keep the same 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 in making a projection. You should use the `PARSE` command if you want the values to be translated immediately.

4.19.4 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 4.86 on page 128 (NTUPLE).

4.19.5 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.

4.19.6 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`.

4.19.7 DELETE

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

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

4.19.8 LIST

```
Syntax: CUT LIST
```

Lists the cuts already specified.

4.19.9 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
```

4.19.10 END

```
Syntax: CUT END
```

Exits cut.

4.19.11 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 one of the cuts:

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

Make a second projection:

```
PROJECT id X &idb2
```

2. Use a more complicated expression for a cut. Also illustrate how to use `CUT NAME`. Assume you have an Ntuple with 3 variables, gaus1, gaus2 and flat:

```
cut del 0
cut name gauss gaus1 > gaus2
cut new abs(flat-0.5) < 0.5*(gaus1+gaus2)
cut use -1
ntuple plot 31 (gaus1-gaus2) &1 50 -5 5
cut use gauss
ntuple plot 31 (gaus1-gaus2) &2 50 -5 5
```

4.20 CWN

This is an abbreviation for ColumnWise Ntuple. See section 1.15 on page 24 (ColumnWise Ntuples) and section 4.86 on page 128 (NTUPLE) for more details.

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 should 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:

```
!              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 4.106.60 on page 162 (SET ORDER) for more details). The following ways of giving the time are recognized:

```
DATE_TIM     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 VMS, 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.

If you want to ignore some columns in the data, give DUMMY as the name of the column. This works on both the ORDER and the NTUPLE card.

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 dummy THETA PHI
TITLE Ntuple containing particle parameters
0.35 0.498 Angle  0.6 3.6
1.20 0.480 Angle -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 to a file, in ASCII 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.

Ntuples or scatter plots with up to 10 variables can be stored. Interactively you are warned if you try to write more than 1000 points!

You can give a list of plot numbers to store, but all id's must be on 1 line.

4.23 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.

Following the L3 database conventions years after 1999 should be given in the form (100 + YY), i.e. 2000 is year 100 etc.

4.23.1 DB_HISTORY

```
Syntax:  DB_HISTORY[/qual] directory chan1 chan2 start stop mode [version]
where:   qual       is PLOT|NOPLOT|PEDESTAL|NOPEDESTAL
         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=today)
         stop       is the finishing date and time (default=today)
         mode       specifies the mode to plot (default is S)
                    U  plots the numbers as is
                    S  scales them by a reference entry
                       - you give the date/time
                    C  scales them by the latest entry
         version    is the FLUM database version or
                    specifies the detector part for the ECAL database
```

Plots one or more elements of a database vs. time. The default qualifiers are `/PLOT/NOPEDESTAL`. 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 `yYYMMDD.HHMMSS`. Year 2000 should be given as 100 etc. The default starting time and finishing times are the current time. If you want the database entries prior to some time, give that time first and the number of days, months, years before as a negative number. e.g. 0 -20 will get the database entries for the past 20 days and 941030 -100 will get the database entries between 940930 and 941030. If the time is omitted it is 000000 by default.

You can plot the data as is or scaled by either a reference entry, for which you give the date and time, or the latest entry.

The version number is required for the ECAL database and optional for all the others.

The ECAL database has a separate bank for each of the half-barrels and each of the endcaps. The version number is in the form kji where:

```
k =  0  means get the top-level data (e.g. the pedestals)
  =  1  means get the data from the down link (e.g. the pedestal widths)
j =  0  means low energy data
  =  1  means low energy data
  =  2  means high energy data
i = -1  means get the -Z detector data (barrel + endcap)
  = -2  means get the +Z detector data (barrel + endcap)
  =  0  means get all the data - does not work!
  =  1  means get the data for -Z half-barrel
  =  2  means get the data for +Z half-barrel
  =  3  means get the data for -Z endcap 1
  =  4  means get the data for +Z endcap 2
```

For example version 111 for directory //DBEC/ELECTRONICS/PEDESTALS/BEAM will get the pedestals widths for the -Z half-barrel, for the low energy range of the Level-1 board. i=0,-1 and -2 do not work yet, nor does pedestal subtraction.

The channel number for the ECAL database is interpreted as the box number.

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

4.23.2 DB_SNAP

```
Syntax: DB_SNAP[/qual] directory chan1 chan2 time mode [version]
where:  qual       is PLOT|NOPLOT|PEDESTAL|NOPEDESTAL
        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)
                   U  plots the numbers as is
                   S  scales them by a reference entry
                      - you give the date/time
                   P  scales them by the previous entry
                   D  gives the difference between the previous entry
                   RD gives the difference between a reference entry
                      - you give the date/time
        version    is the FLUM database version or
                   specifies the detector part for the ECAL database
```

Plots part or all of the contents of a database directory. By default the qualifiers are `/PLOT/NOPEDESTAL`.

Time should be given in the form (`yYYMMDD.HHMMSS`). Year 2000 should be given as 100 etc. The default time gets you the last entry. You can plot the data as is or scaled by either a reference entry, for which you give the date and time, or the previous entry.

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

The version number is required for the ECAL database and optional for all the others.

The ECAL database has a separate bank for each of the half-barrels and each of the endcaps. The version number is in the form kji where:

```
k =  0  means get the top-level data (e.g. the pedestals)
  =  1  means get the data from the down link (e.g. the pedestal widths)
j =  0  means low energy data
  =  1  means low energy data
  =  2  means high energy data
i = -1  means get the -Z detector data (barrel + endcap)
  = -2  means get the +Z detector data (barrel + endcap)
  =  0  means get all the data - does not work!
  =  1  means get the data for -Z half-barrel
  =  2  means get the data for +Z half-barrel
  =  3  means get the data for -Z endcap 1
  =  4  means get the data for +Z endcap 2
```

For example version -111 for directory //DBEC/ELECTRONICS/PEDESTALS/BEAM will get the pedestals widths for the -Z detector (half-barrel + endcap), for the low energy range of the Level-1 board.

For the ECAL data base the data always gets stored in a 160x41 2-dimensional histogram for now, so that it is easily interfaced with the BGEO ECAL display. See section 4.106.64 on page 164 (SET PARAMETER ECAL) for more details.

4.24 DB_HISTORY

```
Syntax:  DB_HISTORY[/qual] directory chan1 chan2 start stop mode [version]
where:   qual       is PLOT|NOPLOT|PEDESTAL|NOPEDESTAL
         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)
                    U  plots the numbers as is
                    S  scales them by a reference entry
                       - you give the date/time
                    C  scales them by the latest entry
         version    is the FLUM database version or
                    specifies the detector part for the ECAL database
```

Alias for `DATABASE DB_HISTORY`. See section 4.23.1 on page 65 (DATABASE DB_HISTORY) for more details.

4.25 DB_SNAP

```
Syntax:  DB_SNAP[/qual] directory chan1 chan2 time mode [version]
where:   qual       is PLOT|NOPLOT|PEDESTAL|NOPEDESTAL
         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)
                    U  plots the numbers as is
                    S  scales them by a reference entry
                       - you give the date/time
```

```
                P  scales them by the previous entry
                D  gives the difference between the previous entry
                RD gives the difference between a reference entry
                   - you give the date/time
        version    is the FLUM database version or
                   specifies the detector part for the ECAL database
```

Alias for `DATABASE DB_SNAP`. See section 4.23.2 on page 66 (DATABASE DB_SNAP) for more details.

4.26 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 4.42 on page 82 (EXECUTE).

4.26.1 Examples

1. Define the command LOOK:

```
MN_CMD> DEFINE LOOK
DEFINE> PRINTOUT -5
DEFINE> SIMPLEX @1 50 @2
DEFINE> MIGRAD @1 @2
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> INQUIRE 1 'Give histogram id to fit'
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.

4.27 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.

4.28 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 1.8 on page 14 (Expressions) for more details. See section 1.7 on page 11 (Numbers) for details on the meaning of all the parameters. The parameter can also be a character string that one can change (`CHAR`, `CHTITLE`, or `CHNAME`). Use the `EXAMINE` command to look at any of the parameters described

below. You can also use the SHOW CHARACTER command to look at the character arrays and the usual DIRECTORY, INDEX, DUMP commands to look at titles and variable names.

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. User variables can be deleted using the REMOVE command. If you give a user variable with a length longer than 8 characters it will be truncated.

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 1.7 on page 11 (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 id)
               m is the bin number
```

Note that the x values and their errors can only be changed for a series of data points. For plots with asymmetric errors DX and DY are interpreted as the negative errors.
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 id)
              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 id)
              m    is the event number
              nvar is the variable number
              tvar is the variable name
```

If the histogram identifier is in a register use the syntax `PARSE DEPOSIT X{IRn,(I6.6)}(m,nvar)` etc. to convert the register to a number.

You can change the value or set a value of a character array element using the command:

```
DEPOSIT CHAR(i) = 'string'
```

You can change the title of a histogram or Ntuple using the command:

```
DEPOSIT CHTITLE(n) = 'string'
```

You can change the variable name of an Ntuple using the command:

```
DEPOSIT CHNAME(n,nvar) = 'string'
or      CHNAME(n,tvar) = 'string'
```

If the string should contain blanks enclose it in quotes.

4.28.1 Examples

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

```
DEPOSIT P1(2) = 1000
DEPOSIT ERR1(2) = 500
EXAMINE P1(2)
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)
examine pi
show register 1:10
```

4. Change 2 histogram titles in 2 different ways:

```
title 31 'A New Title'
dep chtitle(32) = 'A Different New Title'
```

4.29 DIRECTORY

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

Alias for INDEX. Gives a directory 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 `DIRECTORY 0 &idb`. To get all the histograms in a particular range, give the command `DIRECTORY id1:id2 &idb1:idb2`.

4.30 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`, `FSIL` and `FWCH`. For the ZEUS experiment it has been implemented for the `FTD` and `TRD` detectors.

Alias for `HIST DISPLAY`. See section 4.59.2 on page 112 (HISTOGRAM DISPLAY) for more details.

Note that there is also a `DISPLAY` command in MINUIT for showing the results of fits. See section 5.8 on page 194 (MINUIT DISPLAY) for more details.

4.31 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.

If you really want to calculate an efficiency and the numerator is a subset of the denominator you should use the `EFFICIENCY` command.

4.32 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 1.7 on page 11 (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. Remember that the DO loop variable must be a single character.

If you use the loop variable as a number, remember that all comparisons use floating point numbers.

4.32.1 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
   set plot 4&@i default
   DO J=1,r131
        dep R5 = R5 + 1
        DEP R6 = Y3&@I(@J) ** 2
        FILL 4&@I R5 R6
   ENDDO
ENDDO
```

4.33 DRAW

```
Syntax: DRAW item|command ...
 where: item       is ARC, ARROW, BOX, CIRCLE, ELLIPSE, GLUON, LINE,
                   POLYGON, POLYLINE, SEGMENT, SINE, SYMBOL or TRIANGLE
        command    is CHANGE, DELETE, LIST, FETCH, STORE, or END
```

Draws an item on the plot. See section 2.3 on page 37 (DRAW Menu) for a short description of each comand. 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 an item.

If you have the mouse turned on, SET MOUSE ON, the coordinates of the items are defined by using the mouse. In the case of a CIRCLE give the centre and a point on the radius; for an ARC give the centre, the bottom right corner and the top left corner; for an ellipse give the centre, the major axis and the minor axis.

The default is for items to be drawn after all the plots. If you set the fl option to 1, the items will be drawn after the 1st plot when you give a REDRAW or HARDCOPY command. You can use this option together with PLOT/EMPTY to draw an empty frame and then the items you want before any histograms. This option is useful if you use hatch -3 for example that is not transparent. The option is ignored if you are drawing without a plot.

The hatch and pattern colours are by default the same as the outline colour, but can also be given on the command line. If you want to solidly fill an object use pattern 100.

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.

If you want to draw without a plot, you should use SET X|Y PSIZE to set the size of the picture and then give the CLEAR command to make sure the coordinate system is setup properly.

If you want your coordinate system to be more flexible you can also draw in plot coordinates, even if you have not made a plot. Set the X and Y limits, and then specify plot coordinates. Note that the coordinate transformation is done as if you had made a plot, i.e. SET X|Y SIZE sets the physical size, and the X|Y MARGIN are also respected. If the X and Y limits are set to default, then the plot coordinates are the same as the physical coordinates.

If you want to give the coordinates of the item to draw on the same line as the option you must give all the parameters (8 for POLYLINE and POLYGON, 7 for the rest). For those that you do not want to change give an =. If you give the coordinates on a new line, then you only need to give those parameters for which you are not happy with the default.

4.33.1 ARC

```
Syntax:  DRAW ARC symbol colour thick units hatch pattern fl hcol pcol
         x,y
         r1,r2
         phi1,phi2
where:   symbol     is the symbol number
         colour     is the colour of the line
         thick      is the line thickness (scale factor)
         units      are the units to use (CM or PLOT units)
         hatch      is the hatch symbol to fill the shape
         pattern    is the pattern symbol to fill the shape
         fl         is whether to draw the item first (1) or last (0)
         hcol       is the hatch colour
         pcol       is the pattern colour
         x,y        are the coordinates of the centre of the circle(s)
         r1,r2      are the inner and outer radii of the circles
         phi1,phi2  are the phi angles of the arc (in degrees)
```

Draws an arc of one or two circles. If the hatch or pattern is not zero the area between the two arcs is filled. If you want to solidly fill an object use pattern 100. If you want to draw just one circle and the lines from the centre to the ends of the arc, give the first radius as zero (using

IGARC, see section 4.106.45 on page 158 (SET IGARC) for more details). If you just want one arc, give the same value for both radii.

The HIGZ arc drawing routine IGARC sometimes ignores what you set for colour and thickness. As an alternative use `SET IGARC OFF` which will let Mn_Fit draw the arc.

4.33.2 ARROW

```
Syntax:  DRAW ARROW symbol colour thick units type dummy fl
         x1,y1
         x2,y2
 where:  symbol     is the symbol number
         colour     is the colour of the line
         thick      is the line thickness (scale factor)
         units      are the units to use (CM or PLOT units)
         type       is the type of the arrow
                    1  = tip of arrow on (x2,y2)
                    2  = tip of arrow on (x1,y1)
                    3  = tip on both ends
                    4  = tip in middle
                    -n = arrow is filled
         dummy      is an unused parameter
         fl         is whether to draw the item first (1) or last (0)
         x1,y1      is the coordinate of the tail of the arrow
         x2,y2      is the coordinate of the head of the arrow
```

Draws an arrow.

4.33.3 BOX

```
Syntax:  DRAW BOX symbol colour thick units hatch pattern fl hcol pcol
         x1,y1
         x2,y2
where:   symbol     is the symbol number
         colour     is the colour of the box
         thick      is the line thickness (scale factor)
         units      are the units to use (CM or PLOT units)
         hatch      is the hatch symbol to fill the shape
         pattern    is the pattern symbol to fill the shape
         fl         is whether to draw the item first (1) or last (0)
         hcol       is the hatch colour
         pcol       is the pattern colour
         x1,y1      is the coordinate of one corner
         x2,y2      is the coordinate of the opposite corner
```

Draws a box using the lower left and upper right coordinates. If you want to solidly fill the box use pattern 100.

4.33.4 CIRCLE

```
Syntax:  DRAW CIRCLE symbol colour thick units hatch pattern fl hcol pcol
         x,y
         r1,r2
where:   symbol     is the symbol number
         colour     is the colour of the line
         thick      is the line thickness (scale factor)
```

```
          units      are the units to use (CM or PLOT units)
          hatch      is the hatch symbol to fill the shape
          pattern    is the pattern symbol to fill the shape
          fl         is whether to draw the item first (1) or last (0)
          hcol       is the hatch colour
          pcol       is the pattern colour
          x,y        are the coordinates of the centre of the circle(s)
          r1,r2      are the inner and outer radii of the circles
```

Draws one or two circles. If the hatch or pattern is not zero the area between the two circles is filled. If you want to solidly fill the circle use pattern 100. If you want to draw just one circle give the first radius as zero.

If you use the mouse to give the position and size of the circle, you can only draw a single circle.

The HIGZ arc drawing routine IGARC sometimes ignores what you set for colour and thickness. As an alternative use `SET IGARC OFF` which will let Mn_Fit draw the arc.

4.33.5 ELLIPSE

```
Syntax:  DRAW ELLIPSE symbol colour thick units hatch pattern fl hcol pcol
         x,y
         r1,r2
         phi
where:   symbol     is the symbol number
         colour     is the colour of the line
         thick      is the line thickness (scale factor)
         units      are the units to use (CM or PLOT units)
         hatch      is the hatch symbol to fill the shape
         pattern    is the pattern symbol to fill the shape
         fl         is whether to draw the item first (1) or last (0)
         hcol       is the hatch colour
         pcol       is the pattern colour
         x,y        are the coordinates of the centre of the ellipse
         r1,r2      are the major and minor axis half-lengths
         phi1       is the rotation angle of the ellipse (in degrees)
```

Draws an ellipse. If the hatch or pattern is not zero the ellipse is filled. If you want to solidly fill the ellipse use pattern 100.

4.33.6 GLUON

```
Syntax:  DRAW GLUON symbol colour thick units amplitude period fl
         x1,y1
         x2,y2
where:   symbol     is the symbol number
         colour     is the colour of the line
         thick      is the line thickness (scale factor)
         units      are the units to use (CM or PLOT units)
         amplitude  is the amplitude of the gluon line
         period     is the length of 1 period
         fl         is whether to draw the item first (1) or last (0)
         x1,y1      is the coordinate of one end of the line
         x2,y2      is the coordinate of the other end of the line
```

Draws a gluon line. If the period of a gluon oscillation is not given, the amplitude will be used.

4.33.7 LINE

```
Syntax:  DRAW LINE symbol colour thick units dummy dummy fl
         x1,y1
         x2,y2
where:   symbol     is the symbol number
         colour     is the colour of the line
         thick      is the line thickness (scale factor)
         units      are the units to use (CM or PLOT units)
         fl         is whether to draw the item first (1) or last (0)
         x1,y1      is the coordinate of one end of the line
         x2,y2      is the coordinate of the other end of the line
```

Draws a line.

4.33.8 POLYGON

```
Syntax:  DRAW POLYGON npoint symbol colour thick units hatch patt fl hcol pcol
         x1,y1
         x2,y2 ...
where:   npoint     is the number of points on the polygon
         symbol     is the symbol number
         colour     is the colour of the line
         thick      is the line thickness (scale factor)
         units      are the units to use (CM or PLOT units)
         hatch      is the hatch symbol to fill the shape
         patt       is the pattern symbol to fill the shape
         fl         is whether to draw the item first (1) or last (0)
         hcol       is the hatch colour
         pcol       is the pattern colour
         x1,y1      is the coordinate of the first point
         x2,y2      is the coordinate of the second point etc.
```

Draws a polygon. If you want to solidly fill the polygon use pattern 100.

4.33.9 POLYLINE

```
Syntax:  DRAW POLYLINE npoint symbol colour thick units dummy dummy fl
         x1,y1
         x2,y2 ...
where:   npoint     is the number of points on the line
         symbol     is the symbol number
         colour     is the colour of the line
         thick      is the line thickness (scale factor)
         units      are the units to use (CM or PLOT units)
         dummy      is an unused parameter
         fl         is whether to draw the item first (1) or last (0)
         x1,y1      is the coordinate of the first point
         x2,y2      is the coordinate of the second point etc.
```

Draws a polyline.

4.33.10 SEGMENT

```
Syntax:  DRAW SEGMENT symbol colour thick units hatch pattern fl hcol pcol
         x1, y1
         x2, y2
         sagitta
where:   symbol     is the symbol number
         colour     is the colour of the line
         thick      is the line thickness (scale factor)
         units      are the units to use (CM or PLOT units)
         fl         is whether to draw the item first (1) or last (0)
         x1,y1      are the coordinates of one end
         x2,y2      are the coordinates of the other end
         sagitta    is the sagitta
```

Draws a segment of a circle. This form is useful if you know the 2 endpoints and the sagitta, rather than the centre and angles. If you want to fill the area you have to use `DRAW ARC`.

The HIGZ arc drawing routine IGARC sometimes ignores what you set for colour and thickness. As an alternative use `SET IGARC OFF` which will let Mn_Fit draw the arc.

4.33.11 SINE

```
Syntax:  DRAW SINE symbol colour thick units amplitude period fl
         x1,y1
         x2,y2
where:   symbol     is the symbol number
         colour     is the colour of the line
         thick      is the line thickness (scale factor)
         units      are the units to use (CM or PLOT units)
         amplitude  is the amplitude of the sine wave
         period     is the length of 1 period
         fl         is whether to draw the item first (1) or last (0)
         x1,y1      is the coordinate of one end of the line
         x2,y2      is the coordinate of other end of the line
```

Draws a sine curve. If the period of the sine wave is not given, the amplitude will be used.

4.33.12 SYMBOL

```
Syntax:  DRAW SYMBOL symbol colour size units dummy dummy fl
         x1,y1
where:   symbol     is the symbol number
         colour     is the colour of the line
         size       is the size of the symbol
         units      are the units to use (CM or PLOT units)
         fl         is whether to draw the item first (1) or last (0)
         x1,y1      is the coordinate of the symbol
```

Draws a symbol.

4.33.13 TRIANGLE

```
Syntax:  DRAW TRIANGLE symbol colour thick units hatch pattern fl hcol pcol
         x1,y1
         x2,y2
```

```
         x3,y3
where:   symbol     is the symbol number
         colour     is the colour of the line
         thick      is the line thickness (scale factor)
         units      are the units to use (CM or PLOT units)
         size       is the amplitude of the sine wave
         fl         is whether to draw the item first (1) or last (0)
         hcol       is the hatch colour
         pcol       is the pattern colour
         x1,y1      is the coordinate of the first point
         x2,y2      is the coordinate of the second point
         x3,y3      is the coordinate of the third point
```

Draws a triangle. If you want to solidly fill the triangle use pattern 100.

4.33.14 CHANGE

```
Syntax:  DRAW CHANGE number
 where:  number     is the item number to change
```

Changes a drawn item.

4.33.15 DELETE

```
Syntax:  DRAW DELETE number
 where:  number     is the item number to change
```

Deletes one or more items to draw. `DELETE 0` deletes them all.

4.33.16 FETCH

```
Syntax:  DRAW FETCH filename
 where:  filename   is the file with a stored drawing
```

Fetches a list of drawing commands made with `DRAW STORE`. The new items are added to any that have already been made.

4.33.17 LIST

```
Syntax:  DRAW LIST
```

Lists the items to draw.

4.33.18 STORE

```
Syntax:  DRAW STORE filename number
 where:  filename   is the file with a stored drawing
         number     is the item number to store
```

Stores one or more drawn items in a file. Item number 0 means store all items.

4.33.19 END

```
Syntax:  DRAW END
```

Finish drawing. Only used if you are already in `DRAW`.

4.33.20 Examples

1. Draw a red arrow in cm with a filled arrow symbol in the middle with ends (1,1), (3,4) Put the complete syntax on one line:

```
set x psize 10
set y psize  5
clear
DRAW ARROW 1 red 4 cm -4 = = 1 1 3 4
```

2. Draw an arc in cyan. The centre of the arc is at (5,5). It has radii of 2 and 4 cm, and goes from 50 to 100 degrees. The arc is filled with hatch 345. Split things across several lines to avoid giving all parameters:

```
set x psize 10
set y psize 10
clear
DRAW arc 3 cyan = 345
5 5
2 4
50 100
```

3. Make a square in plot coordinates. Set the margins to a non-zero small value, as 0 will set them to the default:

```
set x psize 10.2
set y psize 10.2
set x margin 0.1
set y margin 0.1
set x size 10
set y size 10
clear
set x lim -10 10
set y lim -10 10
DRAW box 3 magenta = pl
-6 -6
+6 +6
! Now distort it
set y lim -8 8
redraw
```

4.34 DUMP

```
Syntax: DUMP id [&idb] Y|N|npnt1 [npnt2]
 where: id     is the histogram identifier
        idb    is the (optional) secondary identifier
        npnt1  is the 1st point to dump
        npnt2  is the last point to dump
```

Alias for `HIST DUMP`. See section 4.59.3 on page 113 (HISTOGRAM DUMP). Inside MINUIT you must use `HIST DUMP` as `DUMP` dumps the fit results point by point.

4.35 EDIT

```
Syntax: EDIT filename
```

Edits a file. The default `EDIT` command is `emacs` on Unix, `EDIT/TPU` on VMS, and `dm` on the Apollo machines. If the environment variable `EDITOR` is defined it is used instead. To change it to your favourite editor use the `SET EDIT` command.

4.36 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.

The error on the efficiency is calculated using form derived by Paul Avery for MULFIT. This form makes some assumptions about the prior distribution, but is better than the naive one derived from a binomial. The form is:

```
sigma = sqrt [(n+1) * (N-n+1)] / [(N+3) * (N+2)^2]
```

If the efficiency is 0 or 1 then the unbiased estimate of `n/N` is added to the error. This gives a better estimate of the confidence level:

```
sigma = sigma + 1/(N+2)
```

You should only use the `EFFICIENCY` command if the errors on the number of entries in each bin are Poisson distributed and the numerator is a subset of the denominator. If this is not the case you should use the `DIVIDE` command. It assumes that the numerator and denominator are uncorrelated when calculating the error.

4.37 ELIF

```
Syntax:   IF expression
           block of statements
          [ELIF expression]
           [block of statements]
          [ELSE]
           [block of statements]
          ENDIF
```

Else if clause in an `IF` block. See section 4.66 on page 120 (IF) for more details.

4.38 ELSE

```
Syntax:   IF expression
           block of statements
          [ELIF expression]
           [block of statements]
          [ELSE]
           [block of statements]
          ENDIF
```

Else clause in an IF block. See section 4.66 on page 120 (IF) for more details.

4.39 ENDDO

```
Syntax: ENDDO
```

Terminates a DO loop. See section 4.32 on page 72 (DO) for more details.

4.40 ENDIF

```
Syntax:   IF expression
           block of statements
          [ELIF expression]
           [block of statements]
          [ELSE]
           [block of statements]
          ENDIF
```

End of an IF block. You can also use the FI command. See section 4.66 on page 120 (IF) for more details.

4.41 EXAMINE

```
Syntax: EXAMINE parameter
 where: parameter  is a register, function parameter, bin,
                   character array, title, Ntuple variable 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 1.7 on page 11 (Numbers) and section 4.28 on page 69 (DEPOSIT) for details on the meaning of all the parameters.

You can use the same command to find the content of a character array element, a histogram title or an Ntuple variable name or charcter variable. Alternative commands are SHOW CHAR to look at the character arrays and the usual DIRECTORY, INDEX, DUMP commands to look at titles and variable names. See section 1.12 on page 18 (Text) and section 4.28 on page 69 (DEPOSIT) for details on the meaning of all the parameters.

4.42 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 and/or specify a default value for a parameter. See section 4.69 on page 122 (INQUIRE) for more details.

The default filename qualifier is .mnf. You can exit a command file at any point by putting the RETURN command in it. Comment lines begin with a ! by default. Use the SET CHARACTER COMMENT command to change the comment character. Continuation lines are denoted by a - as the last character of the line. Use the SET CHARACTER CONTINUATION command to change the continuation character.

Within macros you can have DO loops (see section 4.32 on page 72 (DO) for more details) and IF blocks (see section 4.66 on page 120 (IF) 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. Use the SET WAIT_CR OFF command to avoid this.

If you do not want to give the directory name with the filename, you can use the SET PATH command to specify a search path. See section 4.106.65 on page 170 (SET PATH) for more details.

WARNING: In Mn_Fit versions before 4.07/32 you should not include comment lines inside DEFINE commands, if there is a DO loop inside the same file. Inline comments are OK. Mn_Fit keeps track of which line is being read for future use in IF blocks and DO loops. If you have comment lines inside the DEFINE command, the counting of the line number gets out of sync. As a result any DO loops that are executed after the definitions (within the same macro) will not work properly. After the first run through the loop Mn_Fit will rewind the file and then jump to the wrong line for the second and future runs through the loop.

4.42.1 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
```

4.43 EXTRACT

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

Alias for `HIST EXTRACT`. See section 4.59.5 on page 114 (HISTOGRAM EXTRACT).

4.44 FETCH

`Syntax: FETCH [filename] id1 [:id2] [&cycle] [id3...] [&cycle]`

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.

You can use the secondary identifier to specify which cycle to fetch from HBOOK RZ files. The cycle number will be added to the current setting for the secondary identifier. This is useful if you store a series of Mn_Fit histogram which have the same primary identifier, but different secondary identifiers, as they then get stored in an HBOOK RZ file with the same identifier, but different cycles. Use the `ZDIR` command to see which cycle numbers exist.

It is a good idea to set the record length for direct access HBOOK version 4 files, if it is known. Otherwise you will get an error message and then Mn_Fit will try to open the file again with the correct record length. The default is 1024 words. 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, profile plots and variable bin width plots can also be fetched using the above syntax. RowWise Ntuples of <50000 words will be read directly into memory. The data for longer Ntuples which have been made with the disk option, will not be read in.

You also plot histograms directly from an HBOOK RZ file by opening the file (`OPEN` or `HB_OPEN`) and then giving the plot command. This feature can be suppressed using the `SET AUTOFETCH OFF` command.

If your histograms are in several subdirectories, you should use `SET DIRECTORY` 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.

It is a good idea to set the record length for direct access HBOOK version 4 files, if it is known. Otherwise you will get an error message and then Mn_Fit will try to open the file again with the correct record length. The default is 1024 words. 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.

If the histogram file is locked, then Mn_Fit will try to remove the lock so that it can be read properly. This feature has not been fully tested yet, as I cannot find a locked histogram file!

4.44.1 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. Use a `DO` loop to calculate and fetch the histograms that are required:

```
open filename.rz
do i=1,10
  fetch @i
  deposit r1 = 1000 + @i
  parse fetch {ir1,(i6)}
enddo
```

The `PARSE` command is needed here, because `FETCH ir1` would try to open file `ir1.his`, which is not what you want!

4.45 FI

```
Syntax:   IF expression
           block of statements
          [ELIF expression]
           [block of statements]
          [ELSE]
           [block of statements]
          FI
```

End of an `IF` block. You can also use the `ENDIF` command. See section 4.66 on page 120 (IF) for more details.

4.46 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 4.59.6 on page 114 (HISTOGRAM FILL) for more details.

4.47 FIT

```
Syntax: FIT[/qual] id1 [&idb1] [-nfun1 [-nfun2 ...]] id2 [&idb2] ...
        nmode
 where: qual       are (optional) qualifiers:
                   CHI         means do a chi**2 fit
                   LIKELIHOOD  means do a likelihood fit
                   SLIKELIHOOD means do a likelihood fit including
                               Poisson function errrors
                   NORMALIZE   means use an overall normalization factor
                   FRACTION    normalizations are fractional
                   GAUSS       means fit a Gaussian
                   FLAT        means fit a flat line
                   LINE        means fit a line with a slope
        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 chi**2 fit
                   =  1 likelihood fit
                   =  2 likelihood fit including function statistics
                   = 1n parameters are fractions + overall normalization
```

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. If you have not given the qualfier that says which sort of fit you want to do, 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.

A fit including the function statistics assumes that you are fitting with a histogram or some other function where the error on the function for each bin is given by the Poisson error on the value of the function, e.g. the function is a Monte Carlo histogram of the background without any reweighting or scale factor applied. This sort of fit implies that the parameters are fractions and an overall normalization is included.

See the section in the manual "Fitting With Mn_Fit" for more complete details on how to fit. See section 1.5.1 on page 8 (Errors) for more information on MINUIT errors and how to interpret them.

The normal way to fit is by using the `FUNCTION ADD` command to add the function(s) that you want to use and then using the `FIT` command. However, if you want to do a simple fit of a Gaussian, flat line or a line with a slope you can use the `FIT/qual` syntax. This form will add the correct function(s), disable all others and start fitting. The default MINUIT commands are `MINIMIZE` followed by `DISPLAY`. You can change these using `SET FIT COMMANDS`.

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 fitting mode 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 44 (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
  DEFAULT           Sets fitting option to default
  INTEGRATE ON|OFF  Turns on/off integrating functions across each bin.
  CONVOLUTE ON|OFF  Turns on/off convoluting functions with a Gaussian.
  AREA              Calculate the AREA for dipion and
                    fragmentation functions.
  COMMANDS          Commands used in the quick fitting mode
```

4.47.1 Likelihood

The formula used for the likelihood is:

```
-log(L(i)) = 2 * n(i) * log(n(i) / mu(i)) - (n(i) - mu(i))

where n(i)   is the number of entries in bin i
and   mu(i)  is the expected number of entries
```

If either n(i) or mu(i) are <= 0, the log term is set to 0. If the log likelihood is negative then I take 1000 * abs(-log(L(i))) for that point.

This formula can be derived as follows. The probability of observing n(i) events in bin i, when the expected number is mu(i) is:

```
L(i) = exp(-mu(i)) * mu(i)**n(i) / n(i)!
```

In the fit one wants to determine mu(i). For the total likelihood the product of the individual likelihoods is taken so an arbitrary normalization can be applied to the likelihood function that can depend on n(i), but not on mu(i). I therefore define:

```
       exp(-mu(i)) * mu(i)**n(i)   *             n(i)!
L(i) = -------------------------       ----------------------
                  n(i)!                exp(-n(i)) * n(i)**n(i)

       exp(-mu(i)) * mu(i)**n(i)
L(i) = -------------------------
       exp(-n(i))  * n(i)**n(i)

-log(L(i)) = -(-mu(i) - n(i)) - n(i) * log(mu(i) / n(i))

           = n(i) * log(n(i) / mu(i)) - (n(i) - mu(i))
```

This value is multiplied by 2 so that a log likelihood change of 1 corresponds to a 1 sigma error. The likelihoods are summed and then this is the value that is minimized.

4.47.2 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/like 401 501&10
```

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/chi 1 -2 -3 17 -2  -4
```

4.48 FUNCTION

```
Syntax: FUNCTION command
```

Does something with a function. See section 2.4 on page 38 (FUNCTION Menu) for a short description of each command. For a detailed help on the functions available see section 4.48.9 on page 91 (FUNCTION LIST) and its subtopics. When you are asked to give function number(s), 0 means all the defined or selected functions.

Note that all Mn_Fit functions by default include the bin width in the calculation of the function value. Thus, if you want to fit a histogram that is actually stored as a series of data points, then you have to make sure that `DX` for each point corresponds to the bin width. You can use the `SET FUNCTION BIN_WIDTH ON|OFF` command to change whether the bin width is taken into account.

The AREA under a fucntion is by default the area obtained by integrating from -infinity to +inifinity. You can change this AREA to be that of the plot (or othogonality) limits using the command `SET FUNCTION AREA LIMIT`. This command applies to all forms of Gaussian, Landau, Breit-Wigner and Crystal Ball functions.

4.48.1 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 4.48.9 on page 91 (FUNCTION LIST) for a list of the available functions and the various subtopics for details of the syntax for USER and COMIS functions and how to fit using histograms as functions.

In general I recommend using the function name rather than the number, as the number may change at some point, if I add new functions.

Several functions have names consisting of more than one word. If you add a new function using the name, then just give the first word. For 2-D functions use the syntax `FUNCTION ADD 2D GAUSS SIGMA`.

For the lepton and hadron spectra you will be given a list of the functions and must then give the relevant number. This is the same as the function number.

4.48.2 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 COMIS` command.

4.48.3 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. See section 4.35 on page 81 (EDIT) for details on the default editor. You can change the editor with the `SET EDIT` command.

4.48.4 DELETE

```
Syntax: FUNCTION DELETE nfun1 [nfun2 ...]
 where: nfun1  is the 1st function number to delete as in FUNCTION INFO
 where: nfun2  is the 2nd function number to delete etc.
```

Only valid in `MN_CMD>`.

Deletes one or more functions from the list of those you have added. `nfun = 0` will delete all functions that have been added.

4.48.5 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.

4.48.6 HISTOGRAM

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

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.

4.48.7 HOVERLAY

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

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.

4.48.8 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.

4.48.9 LIST

```
Syntax: FUNCTION LIST
```

Note that all Mn_Fit functions by default include the bin width in the calculation of the function value. You can use the `SET FUNCTION BIN_WIDTH ON|OFF` command to change this behaviour.

List of 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
14 Lepton Spectrum V+A
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 Two Bifurcated Gaussians (sigma)
24 Sum Two Bifurcated Gaussians (FWHM)
25 ARGUS background function - phase space going to 0 at beam energy
26 Pexp(Q)  A polynomial multiplied by the exponential of a polynomial
27 Trigonometric Functions
28 CB Line Shape
29 Power Law Function
30 Lifetime Function
31 OChebyshev Polynomial - parameter 1 is not overall normalization
32 OLegendre Polynomial - parameter 1 is not overall normalization
33 Fermi Function (positive going)
34 Fermi Function (negative going)
35 Threshold Function
36 rCB Line Shape (reversed CB line shape with a high-energy tail)
37 rARGUS background function (reversed ARGUS background - phase
   space going to 0 at a certain energy)
38 Dipion Inv Mass
39 Fragmentation Functions
40 Resonance Cross Section
41 Continuum Cross Section
42 User Defined Function (see subtopic USER for details)
43 Histogram used as a function
44 Smooth Histogram used as a function
45 COMIS defined function (see subtopic COMIS for details)
46 Convolution function
47 2D Gaussian Distribution (sigma)
48 2D Gaussian Distribution (FWHM)
```

Polynomial

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

```
BIN_WIDTH * (NORM + POLY01*(x-OFFSET) + POLY02*(x-OFFSET)**2 + ...)

The parameters are:
NORM
POLY01
```

```
POLY02...
OFFSET
```

The parameter OFFSET is redundant and should usually be fixed to a convenient value.

Exponential

Falling exponential with an offset. The form used is:

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

The parameters are:

```
NORM
SLOPE
OFFSET
```

The parameter OFFSET is redundant and should usually be fixed to a convenient value.

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
```

The parameters are:

```
CONST
SLOPE1
SLOPE2
JOIN1
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.

The first parameter is the overall normalization. This is new for version 4 of Mn_Fit. The previous form of the Chebyshev is in the function OChebyshev.

The parameters are:

```
NORM
CHEB01
CHEB02...
```

Legendre

The Legendre 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 first parameter is the overall normalization. This is new for version 4 of Mn_Fit. The previous form of the Legendre is in function OLegendre.

The parameters are:

```
NORM
PARM01
PARM02...
```

Gaussian

Gaussians can be given either in terms of `SIGMA` or `FWHM`. They are normalised, so that the area is what it says. The form in terms of `SIGMA` is:

```
AREA / (SQRT(2*PI) * SIGMA) * EXP( (X-MEAN)**2/(2*SIGMA**2)
```

The parameters are:

```
AREA
MEAN
SIGMA
```

In terms of `FWHM` it is the same except for the scale factor between the two.

Landau

Gives the Landau density function in the form:

```
AREA * DENLAN( (X - XLAN)/ WIDTH)
```

`DENLAN` is the CERN library routine for the Landau density function of a normalized variable.

The parameters are:

```
AREA
XLAN
WIDTH
```

Note that `XLAN` is not actually the peak of the distribution, which is why its name was changed in Mn_Fit version 4.07/22. This is discussed in the long write-up for the `DENLAN` routine. The actual peak is at -0.222782*WIDTH below the parameter `XLAN`.

Breit Wigner

This is the Breit-Wigner signal shape. Its form is:

```
(1/2*pi) * (AREA * WIDTH) / ( (X-MEAN)**2 + (WIDTH**2)/4 )
```

The function is bin-width normalized.

The parameters are:

```
AREA
MEAN
WIDTH
```

x-a Gaussian

Gaussians can be given either in terms of `SIGMA` or `FWHM`. They are normalised, so that the area is what it says. The form in terms of `SIGMA` is:

```
AREA * (X-OFFSET) / (SQRT(2*PI) * SIGMA) * EXP( (X-MEAN)**2/(2*SIGMA**2)
```

The parameters are:

```
AREA
MEAN
SIGMA
OFFSET
```

In terms of `FWHM` it is the same except for the scale factor between the two.

Lepton Spectra

A series of lepton momentum spectra for semileptonic b and c meson decays are available. For all the spectra the parameters are:

```
AREA
MS/MC   Ratio of strange/charm mass, or charm/bottom mass
SCALE1
SCALE2
```

Although the second parameter is called `MS/MC` it means the ratio of the relevant quark masses, i.e. `MC/MB`, `MU/MB` etc.

The form for the phase space is:

```
NORM * VAL * (1 - (MS/MC)**2 - VAL) / (1 - VAL)
where VAL = X*SCALE2/SCALE1
```

i.e. the spectrum is by default between 0 and `(1-(MS/MC)**2)`. You should set `SCALE1` and `SCALE2` so that their ratio gives the b or c quark momentum for your data.

The forms for (V-A), (V+A) and Spin 0 -> Spin 0 are a bit more complicated.

I am not sure why there are 2 parameters `SCALE1` and `SCALE2`. As far as I can tell only the ratio is used. You should therefore decide what ratio is needed and fix the parameters accordingly.

Hadron Spectra

A series of hadron momentum spectra for semileptonic b and c meson decays are available. For all the spectra the parameters are:

```
AREA
MS/MC   Ratio of strange/charm mass, or charm/bottom mass
SCALE1
SCALE2
```

Although the second parameter is called `MS/MC` it means the ratio of the relevant quark masses, i.e. `MC/MB`, `MU/MB` etc.

The form for the phase space is:

```
NORM * VAL * SQRT(VAL**2 - 4*(MS/MC)**2)
where VAL = X*SCALE2/SCALE1
```

You should set `SCALE1` and `SCALE2` so that their ratio gives the b or c quark momentum for your data.

It is not clear to me that the form for the hadron spectra is correct or where it comes from. In particular for the phase space and Spin 0 -> Spin 0 form it rises monotonically, while for the (V-A) and (V+A) functions it goes negative. I do not know where these forms came from (I originally got them from Paul Avery's MULFIT). Any input would be welcome.

The forms for (V-A), (V+A) and Spin 0 -> Spin 0 are a bit more complicated.

I am not sure why there are 2 parameters `SCALE1` and `SCALE2`. As far as I can tell only the ratio is used. You should therefore decide what ratio is needed and fix the parameters accordingly.

Two Gaussians

Gaussians can be given either in terms of `SIGMA` or `FWHM`. They are normalised, so that the area is what it says. The parameters for the 2 Gaussians are such that it is easy to fix the difference in the means, or the ratio of the widths. The parameters for the second Gaussian are `AREA2/AREA`, `(MEAN2-MEAN1)` and `SIGMA2/SIGMA1`. The form of each Gaussian in terms of `SIGMA` is:

```
AREA / (SQRT(2*PI) * SIGMA) * EXP( (X-MEAN)**2/(2*SIGMA**2)
```

The parameters are:

```
AREA
MEAN
SIGMA1
AR2/AREA
DELM
SIG2/SIG1
```

For the first Gaussian the area is `AREA * (1-AR2/AREA)`. For the second Gaussian the area is `AREA * (AR2/AREA)`. The mean of the first Gaussian is `MEAN`, while the mean of the second Gaussian is `MEAN + DELM`. The width of the first Gaussian is `SIGMA1` and the width of the second is `SIGMA1 * (SIG2/SIG1)`.

In terms of `FWHM` everything is the same except for the scale factor between the two.

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
```

The parameters are:

```
AREA
MEAN
SIGA
SIGB
```

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**2)                     for X >= MEAN
```

The parameters are:

```
AREA
MEAN
SIGA1
SIGB/SIGA
AR2/AREA
SIG2/SIG1
```

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 widths. In addition the ratio of both of the widths for the two Gaussians is same and is the parameter `SIG2/SIG1`.

ARGUS Background

This function is the form ARGUS used 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)
```

The parameters are:

```
NORM
OFFSET
EBEAM
EFACT
```

Use the command `SET FIT AREA ON` to get a normalised function, so that NORM is actually the area within the range of the histogram you are fitting.

rARGUS Background

This function is an inverted version of the form ARGUS used to parametrize their background for fitting B mass peaks. The form used is:

```
P(1) * (x + P(2)) * SQRT(((x + P(2)) / P(3))**2 - 1) *
 EXP(P(4) * (((x + P(2)) / P(3))**2) - 1)
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 value at which the function goes to zero
      P(4) is a scale factor for the exponential (0 to not use it)
```

The parameters are:

```
NORM
OFFSET
EZERO
EFACT
```

Use the command `SET FIT AREA ON` to get a normalised function, so that NORM is actually the area within the range of the histogram you are fitting.

Pexp(Q)

This function can be used to combine a polynomial with the exponential of a polynomial. In each case the polynomial for is:

```
NORM * (1 + POLY01*(X-OFFSET) + POLY02*(X-OFFSET)**2 + ...)
```

The parameters are:

```
NORM
OFFSET
POLY01
POLY02...
EPOLY01
EPOLY02...
```

The parameter `OFFSET` is redundant and should usually be fixed to a convenient value.

Trigonometric

The form used is:

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

The parameters are:

```
ASINE
ACOSINE
OMEGA
```

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.
```

The parameters are:

```
AREA
MEAN
SIGMA
ALPHA
N
```

`N` controls the length of the tail to lower energies. Both `ALPHA` and `N` should be none-zero and `N` should I think be smaller than the `MEAN`. `ALPHA` should be positive (maybe fixed?) and says where the function starts to diverge from a normal Gaussian in units of `SIGMA`.

rCB Line Shape

This function uses the NaI line shape as obtained by the Crystal Ball. However it is reversed from the CB line shape (see 'HELP FUNCTION LIST CB_LINE_SHAPE for more details) and has a high-energy rather than a low energy tail. 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
```

Power Law

The form used is:

```
NORM * (x / SLOPE)**POWER
```

Note that there is no protection against `(x / SLOPE)**POWER` going into overflow.

The parameters are:

```
NORM
POWER
SLOPE
```

Lifetime

Falling exponential with an offset for lifetime fitting. The form used is:

```
AREA * ABS(SLOPE) * EXP(-SLOPE * (x - OFFSET))
for (x - OFFSET) >= 0
```

The parameters are:

```
AREA
LIFETIME
OFFSET
```

The parameter `OFFSET` is redundant and should usually be fixed to a convenient value.

The function can also be combined with a resolution function by using the `SET FIT CONVOLUTE` command.

Fermi

A Fermi function. This form is useful for threshold or efficiency turn ons. It can be selected as either positive or negative going. The positive going form used is:

```
NORM / (1 + EXP((X-X0)/SLOPE))
```

with `(X0-X)` for the negative going function.

The parameters are:

```
NORM
X0
SLOPE
```

The function can also be combined with a resolution function by using the `SET FIT CONVOLUTE` command.

Threshold

Threshold function. The form has a sharp rise followed by an exponential tail. The form used is:

```
NORM * (X-OFFSET)**POWER * EXP(COEFF1*(X-OFFSET) + COEFF2*(X-OFFSET)**2)
where NORM    is the normalization
      OFFSET  is the threshold
      POWER   is the power of the turnon
      COEFF1  is the coefficient linear in x
      COEFF2  is the coefficient quadratic in x
```

As the function is not smooth at the threshold it seems to be better to fix `OFFSET`, if all other parameters are left free. Alternatively it should be possible to fix the `POWER` and leave the offset free.

Dipion Inv Mass

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

By default, the `AREA` has an arbitrary normalization. Use the `SET FIT AREA ON` command to make it meaningful. A Simpson integration of the function is then performed if any of the parameters change.

These functions all prompt you for the parent and daughter resonance names and the pion type, and the following fit parameters, whose significance depends on the model chosen:

```
AREA
PARM
```

Phase Space This is the dipion invariant mass according to phase space arguments. It takes the form:

```
AREA *
SQRT(((Ep + Ed)**2 - x**2)*((Ep - Ed)**2 - x**2)*(x**2 - 4*mpi**2))
where
      Ep  is the energy of the parent resonance,
      Ed  is the energy of the daughter resonance,
      mpi  is the mass of the pion.
```

Yan Model This is the first order dipion invariant mass from the Yan model. It takes the form:

```
AREA * PHASE SPACE * [(x**2 - 2*mpi**2) + (PARM/3)*(x**2 - 2*mpi**2)*
                      (x**2 - 4*mpi**2)   +   2*(x**2 + 2*mpi**2)*
                      (((Ep**2 + x**2 - Ed**2)/(2*Ep))**2)/x**2]
where
  Ep  is the energy of the parent resonance,
  Ed  is the energy of the daughter resonance,
  mpi is the mass of the pion,
  PHASE SPACE  has the functional form of the Phase_Space model.
```

`x` is limited to the range `(2 * mpi)` $<$ `x` $<$ `(Ep - Ed)`.

Voloshin Model This is the dipion invariant mass from the Voloshin-Zakharov model. It takes the form:

```
AREA * PHASE_SPACE * ...
```

Novikov Model This is the dipion invariant mass from the Novikov-Shifman model. It takes the form:

```
AREA * PHASE_SPACE * ...
```

Full Yan Model This is the second order calculation of the dipion invariant mass from the Yan model. It takes the form:

```
AREA * PHASE_SPACE * [ YAN_MODEL + ( ( PARM**2 ) / 20 ) *
               ((X**2 - 4*mpi**2) + 4/3 * (X**2 - 4*mpi**2)
    * (X**2 + 6*mpi**2)*(((Ep**2 + X**2 - Ed**2)/(2*Ep))**2)/X**2)
    + ( 8/3 * ( X**4 + 2 * ( X**2 ) * ( mpi**2 ) + 6*mpi**4 ) *
      ( ( Ep**2 + X**2 - Ed**2 ) / ( 2 * Ep ) )**4 / X**2 )]
where
  Ep  is the energy of the parent resonance,
  Ed  is the energy of the daughter resonance,
  mpi is the mass of the charged pion (0.1396 GeV/c).
  PHASE_SPACE  has the functional form of the Phase_Space model.
  YAN_MODEL is the first order Yan model calculation.
```

HW problem: derive this for yourself some weekend.

Peskin Model This is the dipion invariant mass from the Peskin model, the corrected formula.

Isovector Model This is the dipion invariant mass from the Voloshin Isovector model.

BW x Phase Space This model for the dipion invariant mass takes the form of a Breit-Wigner function times the Phase_Space model.

Moxhay Resonance This is the Moxhay resonance form for the dipion invariant mass.

Fragmentation Functions

This is a set of functions containing various theoretical predictions for the fragmentation of quarks and gluons. The following models are available:

```
1: Peterson         2: Kartvelishveli   3: Andersson        4: Collins
5: Bowler
```

By default, the `AREA` has an arbitrary normalization. Use the `SET FIT AREA ON` command to make it meaningful. A Simpson integration of the function is then performed if any of the parameters change.

Resonance Cross Section

Parametrizes a resonance cross section such as an Upsilon state.

The parameters are:

```
AREA
MEAN
SIGMA
AXIS
```

Continuum Cross Section

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

The parameters are:

```
LEVEL
THRESH
HEIGHT
```

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. The errors will not be included if you do a normal likelihood fit even if you ask them to be. However you can use the `SLIKELIHOOD` fitting option (see section 4.47 on page 87 (FIT)) if you want them to be taken into account.

This sort of function is very useful if you want to fit a complicated shape you got from Monte Carlo or an efficiency corrected function to a plot. It can also be used to fit the data to a number of different components which are described by histograms. If you want to be have the x coordinate free in the fit you should use a `Smooth Histogram` function.

Smooth Histogram

Use a smooth 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. The errors are assumed to be the square root of the value of each point. The errors will not be included if you do a normal likelihood fit even if you ask them to be. However, you can use the `SLIKELIHOOD` fitting option (see section 4.47 on page 87 (FIT)) if you want them to be taken into account.

The possible modes for using the histogram are:

```
 0  Do not use the errors on the histogram function
 1  Errors are the square root of the value of each point
1n  The ordinate is scaled by a free parameter
```

This function is very useful if you want to fit a complicated shape you got from Monte Carlo, or an efficiency corrected function to a plot, or if you have a mass peak from Monte Carlo and want to determine its position and/or its relative width (mode 10 or 11).

Note that what this function actually does is to smoothly interpolate using a cubic spline between the centres of the histogram bins. If your histogram also includes statistical fluctuations, then maybe you should first smooth the histogram itself. Use the `SMOOTH` command to do this.

2D Gaussian

Gaussians can be given either in terms of SIGMA or FWHM. They are normalised, so that the area is what it says. The form in terms of SIGMA is:

```
AREA / (SIGMA1 * SIGMA2) * EXP( (X-MEAN1)**2/(2*SIGMA1**2)
                         * EXP( (Y-MEAN2)**2/(2*SIGMA2**2)
```

The parameters are:

```
AREA
MEAN1
MEAN2
SIGMA1
SIGMA2
```

In terms of FWHM it is the same except for the scale factor between the two.

COMIS

```
Syntax: FUN ADD COMIS filename [nfun] Y|N
        nparm
        Title
        Name(s)
 where: filename  is the file containing the function
        nfun      is the (optional) user function number
        Y|N       is whether you want to edit the file or not
        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. If you give the command in a macro you must either put Y|N on a separate line or include the function number (which can be 0). 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. Note that if you want to fit with 2 COMIS functions at the same time, the functions must have different names:

```
DOUBLE PRECISION FUNCTION XMNCMI(XX,YY,NP,DPAR,NFUSER,WFERR)
where XX     is the current x value for which the function is to be
             evaluated (REAL)
      YY     is the current y value for which the function is to be
             evaluated (REAL)
      NP     is the number of parameters
      DPAR   is a DOUBLE PRECISION array containing the parameters
      NFUSER is the Comis function number (set by the command
             FUN ADD COMIS n, where n is the Comis 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 (VMS) or $MN_FIT/test/comis_test.for (Unix).

If you add two or more COMIS functions and want to use them simultaneously, then the functions must have different names, otherwise whichever one you compile last will be used for all of them.

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 can call any of the Mn_Fit predefined functions from within your own COMIS function. The calling sequence is:

```
WVAL = XMNCLC(NFUN,DPAR,NP,NPAR2,X,Y)
where NFUN   is the function number
      DPAR   are the function parameters (in DOUBLE PRECISION)
      NP     is the number of parameters
      NPAR2  only applies to function 26 - polynomial multiplied by
             the exponential of a polynomial
      X      is the X value for which the function returns its
             value (REAL)
      Y      is the Y value - not used in this routine (REAL)
```

The 2-dimensional functions (only Gaussians at present) can also be called using the same calling sequence with function name XMNC2D.

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. See section 4.35 on page 81 (EDIT) for more details on editing. 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>`.

The same Cernlib subroutines and functions are available in Mn_Fit as in PAW with a few additions. For complete list of the routines available see section 1.16 on page 24 (Using COMIS).

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)`.

You should do all relevant calculations in the COMIS functon in `DOUBLE PRECISION`. The `XX` and `YY` values are passed in single precision for historical reasons only.

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`. There also exist single precision versions of the same functions, with the name RMN... (instead of XMN...).

Note that all Mn_Fit functions include the bin width in the calculation of the function value.

The arguments generally used are:

```
NPAR=  Number of parameters
P=     Parameters (DOUBLE PRECISION)
X, Y=  X and Y (REAL)
```

Y is only appropriate for 2D functions.

In order to find out exactly what is calculated for each function you can look at the source code: $MN_FIT/src/function/xmnclc.fpp etc. If you do not have the source code you can get it from the Mn_Fit homepage:

```
http://www-zeus.physik.uni-bonn.de/~brock/mn_fit.html
```

Other arguments are listed with the functions:

```
1) General Functions (1-37):  XMNCLC(NFUN,DPAR,NPAR,NPAR2,X,Y)
   NFUN=  Function number
   NPAR2= Number of secondary parameters

2) Continuum cross section:  XMNCNT(DPAR,NPAR,X)

3) Resonance cross section:  XMNRES(DPAR,NPAR,X)

4) Fragmentation functions:  XMNFRG(MODEL,BIN,DPAR,NPAR,X)
   BIN=   Unused

5) 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

6) Smooth Histogram:  XMNHSM(IDHA,IDHB,OFFSET,NHS,MODE,X,Y,WMNHER)
   IDHA=   Primary Identifier of Histogram
   IDHB=   Secondary Identifier of Histogram
   OFFSET= X Shift of histogram
   NHS=    Element number that stores the spline parameters
   NMODE=  0= No Errors, 1= Errors from histogram
   WMNHER= Error returned from XMNHSM - DOUBLE PRECISION
   The element number is defined if you do FUNCTION ADD SMOOTH id.
   You should therefore add the smooth histogram as a function, do
   a FUNCTION INFO and the element number is given after the id.

6) 2D Functions: XMNC2D(NFUN,DPAR,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 4.48.9 on page 104 (FUNCTION LIST COMIS) for details.

To link Mn_Fit with a user function (Unix) you can use `Makefile.user` in the Mn_Fit top level directory:

```
export MN_FIT=/usr/lib/mn_fit (bash, zsh, ksh, etc.)
setenv MN_FIT /usr/lib/mn_fit (csh, tcsh, etc.)
make -f $MN_FIT/Makefile.user all (Unix).
```

In order to do this, you must have installed the mn-fit-dev Debian package. Your user function must be in the file `xmnusr.fpp` in your current directory and the executable name will be `mn_user_x.exe`. Replace `/usr/lib/mn_fit` with wherever Mn_Fit is located on your system. You may have to change compile options in `Makefile.user` if your system is not the same as I expect. `make -f $MN_FIT/Makefile.user` will give more information on the options available.

Example functions and scripts to run them can be found in `$MN_FIT/test/xmnusr1.fpp` and `$MN_FIT/test/xmnusr1.mnf` and `$MN_FIT/test/xmnusr2.fpp` and `$MN_FIT/test/xmnusr2.mnf`.

The first example contains a quadratic function. The second is probably more interesting and shows how to convolute a Breit-Wigner with a Gaussianconvolute}.

Under VMS use the script `mn_fit_dir:mn_user.lnk` You can give the executable name (default is `mn_user_x.exe`) the filename for your compiled function and any extra libraries or object files that you need, e.g.:

```
$MN_FIT/mgr/mn_user.lnk /home/brock/exe/mn_user fittest.f
```

To run with your user version give the command:

```
mn_fit -u [/home/brock/exe/mn_user]
```

Note that Mn_Fit must have been compiled on your system or installed on your system with the the Mn_Fit libraries available for this to work.

4.48.10 OVERLAY

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

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. You can change the number of points with the SET FUNCTION POINT command.

4.48.11 PLOT

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

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. You can change the number of points with the SET FUNCTION POINT command.

4.48.12 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.

4.48.13 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.

4.49 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. Use the SET HARDCOPY command to change the filename.

Note that what HARDCOPY actually does is to repeat the plotting commands that were used to make the picture. Thus if you change a histogram, by fetching in a new one with the

same identifier or performing some operation on it after that histogram has been plotted, the new histogram will get drawn when you give the HARDCOPY command. Another, sometimes unexpected, side-effect is that if you are windowing and project an Ntuple several times with the same secondary identifier each time, but different cuts, the HARDCOPY command will plot the last histogram only for all the projections. If your hardcopy comes out differently from the picture on the screen this is probably the reason. Try the REDRAW command before HARDCOPY to see what the hardcopy will actually look like.

Different hardcopy devices are supported for different graphics packages. See section 1.23 on page 31 (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.

4.50 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.

4.51 HB_FETCH

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

Alias for FETCH. See section 4.44 on page 84 (FETCH) for more details.

4.52 HB_MN_FIT

```
Syntax:   HB_MN_FIT [id1:[id2]]
where:    id1  is the lower limit fo the range of identifers to fetch
          id2  is the (optional) upper limit on the range
```

Converts all or selected 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:

```
SET DIR newdir     or CD newdir
HB_MN_FIT
```

to get those histograms into Mn_Fit memory. In addition if you have created more than 200 new histograms you will have to use this command with a range of identifiers to fetch the extra ones.

4.53 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.

It is a good idea to set the record length for direct access HBOOK version 4 files, if it is known. Otherwise you will get an error message and then Mn_Fit will try to open the file again with the correct record length. The default is 1024 words. 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.

4.54 HB_STORE

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

Alias for `STORE`. See section 4.114 on page 187 (STORE) for more details.

4.55 HCLOSE

```
Syntax: HCLOSE
```

Closes all open histogram files. This enables you to write to the same filename from another program without having conflicts.

4.56 HCOPY

```
Syntax: HCOPY id1 id2
```

Calls subroutine HCOPY to copy one HBOOK histogram to another.

4.57 HDELETE

```
Syntax: HDELETE id
```

Calls subroutine HDELET to delete an HBOOK histogram.

4.58 HINDEX

```
Syntax: HINDEX
```

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

4.59 HISTOGRAM

Does something with a plot. See section 2.5 on page 38 (HISTOGRAM Menu) for a short description of each command. See section 4.3 on page 47 (2DIM) for more commands on different ways of plotting 2-dimensional histograms.

4.59.1 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.
```

Books 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 4.59.6 on page 114 (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 prompted 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.

Examples

1. Book a binned histogram with errors:

```
BOOK/BIN/ERR 1 'A New Histogram' 1 100 0 100
```

2. Book a series of points with asymmetric errors and allow a maximum of 1000 points:

```
BOOK/UNBIN/ASYM 1 'A New Histogram' 1 1000
```

4.59.2 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 `FSIL` and `FWCH`. For the ZEUS experiment it has been implemented for the `FTD` and `TRD` detectors. You have to use the `SET PARAMETER det` command to set the display options.

/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 4.106.93 on page 178 (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.

ECAL

Displays the L3 BGO electromagnetic caloriter. This command must be preceded by the command `SET PAR ECAL mode xlo xhi ylo yhi`. See section 4.106.64 on page 164 (SET PARAMETER ECAL) for more information.

FBGO

Makes a 2-D display of the L3 forward luminosity monitor BGO. See section 4.106.64 on page 166 (SET PARAMETER FBGO) for more information on what parameters can be changed.

To overlay the forward luminosity monitor chamber hits see section 4.59.2 on page 113 (HISTOGRAM DISPLAY FWCH) and to overlay the silicon tracker hits see section 4.59.2 on page 112 (HISTOGRAM DISPLAY FSIL).

FSIL

Makes a 2-D display of the L3 forward luminosity monitor silicon tracker. See section 4.106.64 on page 167 (SET PARAMETER FSIL) for more information on what parameters can be changed.

You can overlay the silicon tracker on the BGO by using the following syntax:

```
SET PAR FBGO 4 0 0
DISP FBGO id1
SET PAR FSIL 1 0 1 0
DISP/NOCLEAR FSIL id
SET PAR FSIL 1 0 2 0
DISP/NOCLEAR FSIL id
SET PAR FSIL 1 0 3 0
DISP/NOCLEAR FSIL id
```

FWCH

Makes a 2-D display of the L3 forward luminosity monitor planar chambers. See section 4.106.64 on page 168 (SET PARAMETER FWCH) for more information on what parameters can be changed.

You can overlay chambers and also the BGO by using the following syntax:

```
SET PAR FWCH mode side 1 0
DISP FWCH id
SET PAR FWCH mode side 2 0
DISP/NOCLEAR id
etc.
```

Note that this will only work properly if you display one chamber at a time.

FTD

```
Syntax: HISTOGRAM DISPLAY[/CLEAR|/NOCLEAR] FTD id[&idb] nevt
 where: id      is the histogram identifier
        idb     is the (optional) secondary identifier
        nevt    is the Ntuple event number
```

Makes a display of the ZEUS forward tracking detector. See section 4.106.64 on page 168 (SET PARAMETER FTD) for more information on what parameters can be changed.

The display assumes that you are using the standard FTD Ntuple format and that there is one event per Ntuple event.

You can use the `SET X|Y LIMIT` commands to show only part of the detector. However this only works if you are displaying a single chamber. The default limits are -100 to +100cm.

TRD

```
Syntax: HISTOGRAM DISPLAY[/CLEAR|/NOCLEAR] TRD id[&idb] nevt
 where: id      is the histogram identifier
        idb     is the (optional) secondary identifier
        nevt    is the Ntuple event number
```

Makes a display of the ZEUS forward tracking detector. See section 4.106.64 on page 169 (SET PARAMETER TRD) for more information on what parameters can be changed.

The display assumes that you are using the standard TRD Ntuple format and that there is one event per Ntuple event. If you want to see both the TRD and the FTD information you should use the combined Ntuple format.

You can use the `SET X|Y LIMIT` commands to show only part of the detector. However this only works if you are displaying a single chamber. The default limits are -100 to +100cm.

4.59.3 DUMP

```
Syntax: HIST DUMP id [&idb] Y|N|npnt1 [npnt2]
 where: id     is the histogram identifier
        idb    is the (optional) secondary identifier
        npnt1  is the 1st point to dump
        npnt2  is the last point to dump
```

Dumps the contents of a histogram. If you omit the secondary identifier the default is used. If you answer `Y` when asked whether you want to see the points they will all be dumped. You can also give the range of point numbers that should be dumped. This is most useful for Ntuples.

4.59.4 ERRORS

```
Syntax: HIST ERROR id [&idb] nmode [val]
 where: id    is the histogram identifier
        idb   is the (optional) secondary identifier
        nmode is the mode: -1 = remove errors altogether,
                            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.
                            3 = errors on all points are set to val
        val   is the value of the error for mode 3
```

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.

4.59.5 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.

4.59.6 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 1.7 on page 11 (Numbers) for more details on the things available and see section 4.28 on page 69 (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
```

4.59.7 IGTABLE

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

Interface to HIGZ IGTABL routine. Specify the parameters for how to show the table using the command `SET IGTABLE`. See section 4.106.39 on page 157 (SET HIGZ TABLE) and the HIGZ/HPLOT manual for more details. IGTABL allows you to display 2-dimensional histograms as contours, lego plots, surface plots with and without colour and in different coordinate systems.

If you omit the secondary identifier the default is used.

4.59.8 LEGO

```
Syntax: HIST LEGO[/qual] id [&idb] theta phi (default theta=30, phi=30)
 where: qual     can be /C1|/C2|/BAR|/POL|/CYL|/SPH|/PSD|/NFB|/NBB
        id       is the plot identifier
        idb      is the (optional) secondary identifier
        theta    is the viewling angle in theta
        phi      is the viewing anlge in phi
```

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. This limit does not apply if the IGTABL interface is used. The current lego plotting code will soon be replaced by that in the `2DIM LEGO` command, which has no restrictions on the rotation angles and has a number of diffferent colour options.

If you give one of the qualifiers, the HIGZ IGTABL routine will be used for the drawing. See section 4.3.6 on page 50 (2DIM LEGO) for more details. The `2DIM` command can also be used to have many more options on how to plot 2-D histograms. See section 4.3 on page 47 (2DIM) for more details.

One of the qualifiers `/C1,/C2,/BAR` can be given and one of the coordinate system qualifiers `/POL,/CYL,/SPH,/PSD` can be given. The qualfiers `/NFB, /NBB` turn off the font and back boxes respectively.

4.59.9 OVERLAY

```
Syntax: HIST OVERLAY[/option] id [&idb] [symb/col hatch/col patt/col]
    or  HIST OVERLAY[/option] id part
    or  HIST OVERLAY/NEXT [symb/col hatch/col patt/col]
 where: option   can be /SAME|/DIFFERENT|/NEXT/|/NTUPLE|/SMOOTH
        id       is the plot identifier
        idb      is the (optional) secondary identifier
        symb     is the symbol number to use (default = previous + 1)
        hatch    is the hatching to use (default = none)
        patt     is the pattern to use (default = none)
        col      is the (optional) colour for the symbol, hatch or pattern.
```

Overlays a histogram on the last one you plotted. To plot the next histogram in memory use the form `OVERLAY/NEXT`. If you omit the secondary identifier, the default is used. If you omit the symbol the default is taken as 1 greater than the current symbol number. If you are reading the command from a macro or `DEFINE`d command the symbol number must be included on the same line, otherwise the defaults will be used. If you give the command interactively, you have to hit <CR> to get the default values. Hatching and patterns only work with the HIGZ versions of Mn_Fit and are device dependent (see section 4.106.37 on page 156 (SET HATCH) and section 4.106.66 on page 171 (SET PATTERN) for more details).

You can add the symbol, hatch or pattern colour to the symbol number as a qualifier, e.g. `32/red` instead of just 32.

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.

Use `OVERLAY/NTUPLE` to plot variables from an Ntuple. You have to give the command `SET NTUPLE ...` first to specify the variables. Note that this form does not apply any cuts and is most useful for plotting different columns from a table of entries read in with `DAT_FETCH` for example.

You can also plot histograms directly from an HBOOK RZ file by opening the file (`OPEN` or `HB_OPEN`) and then giving the overlay command. This feature can be suppressed using the 'SET AUTOFETCH OFF' command.

/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.

/NEXT

Draws the next plot that is in memory. Useful for scanning a series of plots, following an `id` that you know.

/NTUPLE

Draws the specified variables from an Ntuple. Use the `SET NTUPLE ...` command to specify which variables to put on which axis.

/SMOOTH

Draws the plot as a smooth curve using a cubic spline interpolation between the points.

4.59.10 PLOT

```
Syntax: HIST PLOT[/option] id1[:id2] [&idb1:idb2]] [sym/col hat/col pat/col]
    or  HIST PLOT[/option] id part
    or  HIST PLOT/NEXT [sym/col hat/col pat/col]
 where: option   can be /CLEAR|/NOCLEAR|/NEXT/|/NTUPLE|/SMOOTH|/EMPTY
        id       is the plot identifier
```

```
idb     is the (optional) secondary identifier
sym     is the symbol number to use (default from 'SET SYMBOL')
hat     is the hatching to use (default from 'SET HATCH')
pat     is the pattern to use (default from 'SET PATTERN')
part    is the part of an HBOOK histogram that you want to plot
        e.g. BANX, FUN etc.
col     is the (optional) colour for the symbol, hatch or pattern.
```

Plots 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. To plot all histograms use the syntax `PLOT 0`. To plot the next histogram in memory use the form `PLOT/NEXT`. 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.

Use `PLOT/NTUPLE` to plot variables from an Ntuple. You have to give the command `SET NTUPLE ...` first to specify the variables. Note that this form does not apply any cuts and is most useful for plotting different columns from a table of entries read in with `DAT_FETCH` for example.

If you want to plot an empty frame, without any error messages, book a new histogram with `HISTOGRAM BOOK` and then plot it with `PLOT/EMPTY`.

If you omit the symbol, hatch or pattern the default will be used. If you specify a symbol this only applies to the current plot command and will not change the default. Use `SET SYMBOL` to change the default.

If you are using a colour as a shading for an overlay and need to get the ticks redrawn use the `PLOT/NOCLEAR` command and redraw the 1st histogram.

2-D histograms can also be plotted using the `LEGO` or `SURFACE` commands, or by using the interface to the HIGZ IGTABL routine, via the `2DIM` or `IGTABLE` commands. See section 4.3 on page 47 (2DIM) for more details.

The normal procedure is to `FETCH` the histograms from a file and then plot them using the above syntax. You can also plot histograms directly from an HBOOK RZ file by opening the file (`OPEN` or `HB_OPEN`) and then giving the plot command. This feature can be suppressed using the `SET AUTOFETCH OFF` command. Note that this only works for single histograms and not for a range nor for all histograms.

If you want to check whether a histogram exists in a macro before deciding what to do you can use the following commands:

```
SET PLOT ida&idb DEFAULT
IF r121 = ida & r122 = idb
  ...
ENDIF
```

Note that this only works for histograms that you have tried to fetch. It does not yet work for histograms in an RZ file.

/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 4.106.93 on page 178 (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). It can also be used if you make a plot with a pattern (or hatch -3), which obscures the scale. You can then say `PLOT/NOCLEAR id` for the 1st histogram again.

/EMPTY

Draws an empty plot without giving an error message. This is useful if you want the frame of a plot for drawing in etc.

/NEXT

Draws the next plot that is in memory. Useful for scanning a series of plots, following an `id` that you know.

/NTUPLE

Draws the specified variables from an Ntuple. Use the `SET NTUPLE ...` command to specify which variables to put on which axis.

/SMOOTH

Draws the plot as a smooth curve using a cubic spline interpolation between the points.

4.59.11 SURFACE

```
Syntax: HIST SURFACE[/qual] id [&idb] theta phi (default theta=30, phi=30)
 where: qual  can be /C1|/C2|/CONT|/SHADE|/POL|/CYL|/SPH|/PSD|/NFB|/NBB
        id    is the plot identifier
        idb   is the (optional) secondary identifier
        theta is the viewling angle in theta
        phi   is the viewing angle in phi
```

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. This limit does not apply if the IGTABL interface is used. The current surface plotting code will soon be replaced by that in the `2DIM SURFACE` command, which has no restrictions on the rotation angles and has a number of diffferent colour options.

If you give one of the qualifiers, the HIGZ IGTABL routine will be used for the drawing. See section 4.3.8 on page 50 (2DIM SURFACE) for more details. The `2DIM` command can also be used to have many more options on how to plot 2-D histograms. See section 4.3 on page 47 (2DIM) for more details.

One of the qualifiers `/C1,/C2,/CONT,/SHADE` can be given and one of the coordinate system qualifiers `/POL,/CYL,/SPH,/PSD` can be given. The qualfiers `/NFB, /NBB` turn off the font and back boxes respectively.

4.59.12 2DIM

```
Syntax: HIST 2DIM option id [&idb] [par1 par2 ...]
 where: id      is the histogram identifier
        idb     is the (optional) secondary identifier
        option  is the 2-D display option
        par     are parameters needed
```

Interface to HIGZ IGTABL routine. The parameters needed can either be given on the command line or specified using the command `SET IGTABLE`. IGTABL allows you to display 2-dimensional histograms as contours, lego plots, surface plots with and without colour and in different coordinate systems. See section 4.3 on page 47 (2DIM) for more details.

If you omit the secondary identifier the default is used.

The 2-D plotting options available are `SCATTER, BOX, ARROW, CONTOUR, COLOUR, TEXT, CHAR, LEGO` and `SURFACE`, with extra options for the style and coordinate system for `CONTOUR`, `COLOUR`, `LEGO` and `SURFACE` plots.

4.60 HISTORY

```
Syntax:  history
```

Gives a history of the commands read by GNU readline. This is implemented for Unix versions only. You can then use the command numbers given to recall commands.

If this command does not work, you have a version of Mn_Fit without readline. On Unix machines this means that command recall and filename completion are not available.

Note that this command cannot be abbreviated and must be given in lowercase.

4.61 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.

4.62 HMERGE

```
Syntax: HMERGE filout filein1 [filein2 [filein3 ...]]
 where: fileout  is the name of the output file
        filein1  is the first filename etc.
```

Merges one or more HBOOK RZ files into a single output file. This command calls the HBOOK routine `HMERGE` which combines Ntuples and adds histograms. The files should have identical structures. A <`CR`> signifies that the list of files to merge has finished.

A maximum of 100 files can be merged in one command.

4.63 HRECOVER

```
Syntax: HRECOVER id
```

Recovers an Ntuple in a RZ file by calling HRECOV. Before giving the `HRECOV` command you should open the file with `HB_OPEN`:

```
HB_OPEN filename
HRECOV 1
HB_FETCH 1
```

4.64 HRENAME

```
Syntax: HRENAME id1 id2
```

Renames an HBOOK histogram by calling HCOPY and HDELET.

4.65 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.

4.66 IF

```
Syntax:   IF expression
           block of statements
          [ELIF expression]
           [block of statements]
          [ELSE]
           [block of statements]
          ENDIF
```

This is a rudimentary conditional block structure. If the expression is true, then the first block of statements is performed, if it is false, then the control goes to the block of alternate statements following the `ELSE`. A second block can be started with the `ELIF` statement. If there is no `ELSE` block, execution picks up normally after the `ENDIF`.

Note the result of an expression is always returned as a floating point number, i.e. also the results of `INT` and `NINT` are converted back to floating point immediately.

The `IF` statement can only be used from a command file or in a defined command.

WARNING: In Mn_Fit versions before 4.07/32 you should not include comment lines inside `DEFINE` commands, if there is a `DO` loop inside the same file. Inline comments are OK. Mn_Fit keeps track of which line is being read for future use in IF blocks and DO loops. If you have comment lines inside the DEFINE command, the counting of the line number gets out of sync.

As a result any DO loops that are executed after the definitions (within the same macro) will not work properly. After the first run through the loop Mn_Fit will rewind the file and then jump to the wrong line for the second and future runs through the loop.

4.66.1 expression

The expression can involve registers, parameters, or real numbers. At this time, you are limited to no more than four simple logical expressions per `IF` statement, linked by `.AND.` or `.OR.`, and the whole expression must appear on the same line as the `IF`. The whole expression may be surrounded by a pair of parentheses, but individual expressions within the whole may not. You should not use the word `THEN` after the expression.

4.66.2 Examples

1. Here is a simple IF which decides whether to use a Maximum Likelihood fit or a Chi-square fit based on the area of the histogram:

```
SET PL 10 DEFAULT
IF R125 .LT. 100
  MESSAGE USING MAXIMUM LIKELIHOOD
  FIT 10 1
ELSE
  MESSAGE CHI-SQUARE
  FIT 10 0
ENDIF
```

2. Here is a rather mundane example which illustrates Nesting of IF statements:

```
DEFINE BIGGEST
IF ( @1 .GT. @2 )
  MESSAGE First is biggest
ELSE
  IF @2 .GT. @1
    MESSAGE Second is biggest
  ELIF @2 .LT. 1
    MESSAGE 'If is screwed up, as this has already been checked'
  ELSE
    MESSAGE They are equal
  ENDIF
ENDIF
ENDDEF
```

3. This example illustrates linking of conditionals in a single IF statement:

```
IF ( @1 > @2 & R4 > R6 | R4 .GT. 25 )
  block of statements
ENDIF
```

   Conditionals in MN_FIT are evaluated left-to-right, so the above would be evaluated as:

```
     IF (( @1 > @2   &   R4 > R6 ) | ( R4 > 25 ))
NOT  IF (( @1 > @2 ) & ( R4 > R6   |   R4 > 25 ))
```

Notice that you CANNOT use parentheses to overide the order of evaluation. Only ONE set of parentheses is allowed - around the whole expression - individual subexpressions should not be surrounded by parentheses!

4.67 IGTABLE

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

Interface to the HIGZ IGTABL routines for plotting 2-D histograms in different ways. See section 4.59.7 on page 115 (HISTOGRAM IGTABLE) for more details. Use the command `HISTOGRAM 2DIM` or `2DIM` to pass the option and parameters on the command line.

4.68 INDEX

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

Alias for `DIRECTORY`. 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`.

4.69 INQUIRE

```
Syntax:   INQUIRE par [prompt] [default]
where:    par      is the parameter number
          prompt   is the optional prompt
          default  is the optional default value
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
```

You can specify a default value for a parameter after the prompt. The default is read in as a string. If spaces exist in the string you should enclose it in quotes:

```
inquire 1 'Give extra offset' 0.5
inquire 2 'Give title' 'Dummy Title'
```

If you do not give the parameter value with the above form the default will be taken. If you give a negative parameter number, then you will still be prompted for a new value. If you want a parameter to take its default value type =:

```
exec test 1 = -3
test.mnf:
  inquire 1 'Give first value'  2
```

```
inquire 2 'Give second value' 4
inquire 3 'Give third value'  6
dep r1 = @1 * @2 * @3
message 'The result is {ir1,(i6)}'
```

would have as answer -12.

If you are prompted for a parameter and want the default value also type =. Note that no check is made on whether a default value exists.

4.70 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 intervals to sum over is 100 and can be changed using the `SET FUNCTION INTEGRATE` command. 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`.

4.71 KEY

```
Syntax: KEY command [nkey]
    or  KEY id [&idb] command [nkey]
        nsymb
        text
        x, y, size, angle, opt, mode, font, colour, thick, unit, tcolour
 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 +/-(n000 + abs(nhat|npat))
                    where the sign is the sign of the hatch or
                    pattern number if you want a symbol which
                    includes a hatch or pattern
        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
        opt       can be: LEFT   Key is left adjusted (default)
                          CENTRE Key is centered
                          RIGHT  Key is right adjusted
        mode      can be: CM   = Position is in cm (default)
                          PLOT = Position in terms of plot coordinates
        font      is the font to use
        colour    is the colour number for the key
        thick     is the text thickness factor
        unit      is the segment size for drawing lines (default = 0.01)
                  Only works for symbols 5->8
        tcolour   is the text colour.
```

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, 2DIM, LEGO, SURFACE, DISPLAY or OVERLAY/DIFF commands - OVERLAYS on the same scale do not count), the syntax is the first line. 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. If you do not give the text colour it will be the same as symbol colour. The text colour is useful for plots where you are using colour as a shading to print in black and white.

If you want a symbol that includes a hatch or pattern use the symbol number in the form +/-(n000 + abs(nhat|npat)). n can be between 1 and 8:

```
+/-(1000+nhat) square symbol with hatch nhat
+/-(2000+npat) square symbol with pattern npat
+/-(3000+nhat) histogram symbol with hatch nhat
+/-(4000+npat) histogram symbol with pattern npat
+/-(5000+nhat) square symbol with hatch nhat
+/-(6000+npat) square symbol with pattern npat
+/-(7000+nhat) histogram symbol with hatch nhat
+/-(8000+npat) histogram symbol with pattern npat
```

The special values, 1000,3000,5000,7000 draw a filled symbol and the values 2000,4000,6000,8000 draw a hollow symbol. For values >=5000 an edge is drawn round the symbol in the same colour as the text. Try the macro exec MN_FIT_HELP:key (VMS) or exec $MN_FIT/help/key (Unix) for some examples.

If you use the option SET MOUSE ON, then the position of the key is specified by the mouse. Move the mouse to where you want to have the key and click on the left button. To keep the key in the same position (if you are changing it) click on the right mouse button. Note that using the mouse only works properly in CM mode.

By default the text will be written using the HIGZ routine IGTEXT. For details on formatting text and the other fonts available see section 1.12 on page 18 (Text). The default colour and thickness of the comments are the same as the title.

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.

4.72 LDIRECTORY

```
Syntax: LDIRECTORY
```

Alias for SHOW DIRECTORY or LS.

List of the contents of the currently selected HBOOK directory in an HBOOK4 file or the contents of the currently selected ROOT directory in a ROOT file.

4.73 LEGO

```
Syntax:  LEGO[/qual] id [&idb] theta phi      (default theta=30, phi=30)
 where:  qual  can be /C1|/C2|/BAR|/POL|/CYL|/SPH|/PSD|/NFB|/NBB
```

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. If you give the command without any qualifier the angles must lie between 0 and 90. If you give a qualifier the IGTABLE interface will be used. The current lego plotting code will soon be replaced by that in the `2DIM LEGO` command, which has no restrictions on the rotation angles and has a number of diffferent colour options. See section 4.3 on page 47 (2DIM) for more details.

4.74 LS

```
Syntax: LS
```

Alias for `SHOW DIRECTORY` or `LDIRECTORY`.

List of the contents of the currently selected HBOOK directory in an HBOOK4 file or the contents of the currently selected ROOT directory in a ROOT file.

4.75 MADD

```
Syntax: MADD ido&idbo id1[:id1n][&idb1[:idbn1]] id2[:id2n][&idb2[:idbn2]] ...
 where: ido              is the output histogram identifier
        idbo             is the (optional) output secondary identifier
        id1,id2,...      are the input histogram identifiers
        idb1,idb2,...    are the (optional) input secondary identifiers
        id1n,id2n,...    are the upper limits of an (optional) range
        idb1n,idb2n,...  are the upper limits of an (optional) range
```

Adds a series of histograms (id1, id2) together to make a new 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).

To add a range of histograms, you can specify a range of primary and/or secondary identifiers. For example, `MADD 500&1 300:400&1` will add all histograms with primary identifiers 300 to 400 and secondary identifiers 1 and create histogram 500&1. If you give primary identifier 0, the operation will be performed on all plots with the given secondary identifier.

4.76 MDIRECTORY

```
Syntax:   MDIRECTORY dirnam
where:    dirnam  is the name of the new directory in memory
```

Creates a new HBOOK directory in memory and changes the default directory to that one. Subsequent `STORE` commands maintain the same directory structure as in memory.

4.76.1 Examples

1. Make a new directory, copy some histograms and then store them in that directory. Plots 1,2 will be stored in //MN_HBOUT, the top level directory, while 101 and 102 will be stored in //MN_HBOUT/NEWDIR:

```
store file.hsto 1,2
mdir newdir
copy 1 101
copy 2 102
store/update file.hsto 101,102
```

4.77 MERGE

```
Syntax: 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. Alias for `NTUPLE MERGE`. See section 4.86.4 on page 130 (NTUPLE MERGE) for more details.

4.78 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 1.12 on page 18 (Text) for more details. You can get more fancy and even use `MESSAGE` to write fit results to a file in the form of a LaTeX table, for example!

4.79 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`.

4.80 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).

4.81 MSUBTRACT

```
Syntax: MSUBTRACT ido&idbo id1[&idb1] id2[:id2n][&idb2[:idbn2]] ...
 where: ido              is the output histogram identifier
        idbo             is the (optional) output secondary identifier
        id1              is the histogram identifier from which the others
                         are subtracted
        idb1             is the (optional) input secondary identifier
        id2,...          are the histogram identifiers subtracted from id1
        idb2,...         are the (optional) secondary identifiers
        id2n,...         are the upper limits of an (optional) range
        idb2n,...        are the upper limits of an (optional) range
```

Subtracts a series of histograms (id2,...) from id1 to make a new 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).

To subtract a range of histograms, you can specify a range of primary and/or secondary identifiers. For example, `MSUBTRACT 500&1 200&1 300:400&1` will subtract all histograms with primary identifiers 300 to 400 and secondary identifiers 1 from 200&1 and create histogram 500&1. If you give primary identifier 0, the operation will be performed on all plots with the given secondary identifier.

4.82 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.

4.83 NO_CUT

```
Syntax: NO_CUT
```

Deletes all the specified cuts on a particular histogram. Same effect as `CUT DELETE 0`.

4.84 NO_WINDOW

```
Syntax: NO_WINDOW
```

Alias for `SET NO_WINDOW`. Turns off windowing.

4.85 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.

4.86 NTUPLE

```
Syntax: NTUPLE DUMP|MERGE|PLOT|PROJECT|EPROFILE|SPROFILE|FILTER|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`, `PROJECT`, `EPROFILE`, `SPROFILE` and `FILTER` commands.

ColumnWise Ntuples (CWN's) are supported, but do not yet work with the `FILTER` or `MERGE` commands. If you use a COMIS function for a cut on a CWN then you have to give the list of variables that you want to use in the COMIS function using the `SET NTUPLE VARIABLE` command. For general help on CWN's see section 1.15 on page 24 (ColumnWise Ntuples).

Each time you project or scan an Ntuple that is stored with the disk option, it is usually refetched from the HBOOK file. This ensures that everything is synchronised properly. However, for very large Ntuples, this can take a long time. You can use the `SET AUTOFETCH OFF` command to suppress the extra fetches. In this case you have to make sure that you have not opened another file and that the HBOOK directory is set properly.

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 five common blocks are available to you. `MNTPL2` is only available from version 4.03 onwards:

```
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.

For CWN's the variables are also available in the same common blocks as for PAW:

```
COMMON /PAWCR4/ rvar1, rvar2
COMMON /PAWCR8/ zvar1, zvar2
COMMON /PAWCCH/ cvar1, cvar2
```

where PAWCR4 contains REAL,INTEGER and LOGICAL variables, PAWCR8 contains DOUBLE PRECISION variables, and PAWCCH contains CHARACTER variables.

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 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.

4.86.1 DUMP

```
Syntax: NTUPLE DUMP id [&idb] Y|N|npnt1 [npnt2]
where   id   is the ntuple identifier
and     idb  is the (optional) secondary identifier
```

Dumps the contents of an Ntuple. If you omit the secondary identifier the default is used. If you answer Y when asked whether you want to see the points they will all be dumped. You can also give the range of point numbers that should be dumped.

4.86.2 EPROFILE

```
Syntax: NTUPLE EPROFILE id[&idb] var1 var2 &idb2
        nbinx [xlo xhi]
        ylo yhi [nevt1 nevt]
where   id       is the ntuple identifier
```

```
         var1     is the x-axis variable or expression to project onto
         var2     is the y-axis variable or expression
         idb2     is the secondary identifier of the projection
         nbinx    are the number of bins for the x-axis of the profile
                  plot if the input is an Ntuple.
                  -1 means automatic binning
         xlo      is the lower limit for the x-axis
         xhi      is the upper limit for the x-axis
         ylo      is the lower limit for the y-axis
         yhi      is the upper limit for the y-axis
         nevt1    is the first event to scan
and      nevt     is the number of events to scan
```

Makes a profile plot from an Ntuple. Either variables or expressions can be used for both axes. The mean of the y-axis variable is calculated. The error on the y-axis variable is the error on its mean. Only events that have y-axis variables between the y limits will be used. If `ylo` and `yhi` are both 0, no checking will be done.

Note that for technical reasons the profile plot has both negative and positive errors.

4.86.3 FILTER

```
Syntax: NTUPLE FILTER id&idb [&]idb2 [nevt1 nevt]
    or: NTUPLE FILTER id idb2
where   id     is the ntuple identifier
        idb    is the (optional) secondary identifier
        idb2   is the secondary identifier of the output Ntuple
        nevt1  is the first event to scan
and     nevt   is the number of events to scan
```

Scans an Ntuple applying the currently selected cuts. Events that pass the cuts are stored in a second Ntuple with secondary identifier `idb2`.

This command is useful for applying a preliminary set of cuts on an Ntuple to decrease its size.

This command does not work with CWN's.

4.86.4 MERGE

```
Syntax: NTUPLE MERGE id fileout filein1 [filein2 [filein3 ...]]
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.

This command does not work with CWN's.

If you want to merge complete RZ files use the command `HMERGE` which calls the HBOOK routine `HMERGE` directly.

4.86.5 PLOT

```
Syntax: NTUPLE PLOT id[&idb] var1 [var2 ...] [#weight [#dweight]] &idb2
        nbinx [xlo xhi]
```

```
        [nbiny] [ylo yhi] [nevt1 nevt]
where   id       is the ntuple identifier
        var1     is the first variable or expression to project onto
        var2     is the second etc.
        weight   is a variable name or number to be used as the weight
        dweight  is a variable to be used as the error on the weight
        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
        nevt1    is the first event to scan
and     nevt     is the number of events to scan
```

Projects and plots the projection of an Ntuple or an n-dimensional histogram. For more details see section 4.86.6 on page 131 (NTUPLE PROJECT).

4.86.6 PROJECT

```
Syntax: NTUPLE PROJECT id[&idb] var1 [var2 ...] [#weight [#dweight]] &idb2
        nbinx [xlo xhi]
        [nbiny] [ylo yhi] [nevt1 nevt]
where   id       is the ntuple identifier
        var1     is the first variable or expression to project onto
        var2     is the second etc.
        weight   is a variable name or number to be used as the weight
        dweight  is a variable to be used as the error on the weight
        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
        nevt1    is the first event to scan
and     nevt     is the number of events to scan
```

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. While this also works for single variables in CWNs, it is not a recommended way of proceeding, as it does not work for arrays. 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.

If you want to project onto an element of a CWN, use the form var(n), where n is the element number you want to project onto. To project many elements of a CWN use the form var(m,n1:n2) for the case when var is a 2-dimensional array. You can also use this form inside expressions (see example 8), or when projecting onto more than 1 axis (see example 9).

Note that the limits of each variable of a CWN are not known. Therefore you always have to give the limits for plotting yourself.

If you want one of the Ntuple variables to be the weight precede the variable name or number by #. For the error on the weight give another variable name or number preceded by #. This form is useful if you store a series of points in an Ntuple and then want to manipulate them as a series of points with errors.

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 give the number of bins as -1 automatic binning will be used, taking as default lower and upper limits the lower and upper limits of the relevant variables in the Ntuple. However, as CWN's do not have the limits stored you are always given the opportunity to specify your own limits. If you want to keep the suggested limits just hit <CR> when asked for new limits. If you want to plot the result directly you can use the NTUPLE PLOT command.

If you project onto a scatter plot then you can also give the limits on 1 or more of the axes for the projections - see example 4. These limits will be used as cuts, i.e. any events outside the limits will be dropped.

The first event and the number of events to be scanned can also be given. If you omit them all events in the Ntuple will be scanned. If you specify 0 bins (i.e. you want to make a scatter plot, then you must give the limits for both axes if you also want to give the event numbers. Use limits of 0 0 if they should be taken from the Ntuple.

If you want to make a plot vs. date or time, you can use the various time expressions that are available (see section 1.8 on page 14 (Expressions)). Note that the reference time is always set to 80/01/01 00:00 for such plots. This should be changed in the future when I have a bit more experience with such plots. Use the command SET X MODE DATE|TIME to decide is the plot should be stored in terms of days or minutes. Somewhat more flexibility is available if the information is stored in an ASCII file and can then be read in using the DAT_FETCH 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. Let the limits be calculated internally. Start scanning at event 100 and do 1000 events:

```
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 0 0 0 0 100 1000
```

4. Make a scatter plot of an expression and a variable using a COMIS function as a cut plus a cut on another variable. Specify the limits. Start scanning at event 20 and do 2000 events. This is the same as Example 3, except for the limits:

```
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 -3 3 0 10 20 2000
```

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

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

6. 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. This is usually not what you want and so you should set the limits afterwards by hand:

```
NTUPLE PROJ 1 (X+Y) X-Y &4  -1
partition 1&4 -5 5 -10 10
```

7. Project an Ntuple onto a series of points and use one of the variables as the y value and another as its error:

```
NTUPLE PROJ 1 X #WEIGHT #ERR &5 0
```

8. Project the absolute value of a range of CWN array entries onto a 1-dimensional histogram (note that if nval2 = 0, no entry will be made in the projection):

```
ntuple plot 34 abs(rval2(1:nval2)) &11 75 -5 10
```

9. Project a range of CWN entries into a scatter plot. Note that the scatter plot limits must be given as the limits for CWN variables are not known::

```
ntuple plot 34 abs(rval2(2:nval2)) rval3(2,2:nval2) &12 0 -
 -5 10 -10 10
```

4.86.7 SCAN

```
Syntax: NTUPLE SCAN id[&idb] filename[(arg)] Y|N [nevt1 nevt]
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
        nevt1     is the first event number to scan
        nevt      is the number of events to scan
```

Loops over all the events in an Ntuple and calls a function 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 function 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 4.52 on page 109 (HB_MN_FIT) for more details).

4.86.8 SPROFILE

```
Syntax: NTUPLE SPROFILE id[&idb] var1 var2 &idb2
        nbinx [xlo xhi]
        ylo yhi [nevt1 nevt]
where   id       is the ntuple identifier
        var1     is the x-axis variable or expression to project onto
        var2     is the y-axis variable or expression
        idb2     is the secondary identifier of the projection
        nbinx    are the number of bins for the x-axis of the profile
                 plot if the input is an Ntuple.
                 -1 means automatic binning
        xlo      is the lower limit for the x-axis
        xhi      is the upper limit for the x-axis
        ylo      is the lower limit for the y-axis
        yhi      is the upper limit for the y-axis
        nevt1    is the first event to scan
and     nevt     is the number of events to scan
```

Makes a profile plot from an Ntuple. Either variables or expressions can be used for both axes. The mean of the y-axis variable is calculated. The error on the y-axis variable is its r.m.s. Only events that have y-axis variables between the y limits will be used. If `ylo` and `yhi` are both 0, no checking will be done.

Note that for technical reasons the profile plot has both negative and positive errors.

4.87 OPEN

```
Syntax: 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.

It is a good idea to set the record length for direct access HBOOK version 4 files, if it is known. Otherwise you will get an error message and then Mn_Fit will try to open the file again with the correct record length. The default is 1024 words. 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.

4.88 OVERLAY

```
Syntax: OVERLAY[/option] id [&idb] [symb/col hatch/col patt/col]
    or  OVERLAY[/option] id part
    or  OVERLAY/NEXT [symb/col hatch/col patt/col]
 where: option   can be /SAME|/DIFFERENT|/NEXT/|/NTUPLE|/SMOOTH
        id       is the plot identifier
        idb      is the (optional) secondary identifier
        symb     is the symbol number to use (default = previous + 1)
        hatch    is the hatching to use (default = none)
        patt     is the pattern to use (default = none)
        col      is the (optional) colour for the symbol, hatch or pattern.
```

Alias for HIST OVERLAY. Overlays a histogram on the last one you plotted. To plot the next histogram in memory use the form OVERLAY/NEXT. If you omit the secondary identifier, the default is used. If you omit the symbol the default is taken as 1 greater than the current symbol number. If you are reading the command from a macro or DEFINEd command the symbol number must be included on the same line, otherwise the defaults will be used. If you give the command interactively, you have to hit <CR> to get the default values. Hatching and patterns only work with the HIGZ versions of Mn_Fit and are device dependent (see section 4.106.37 on page 156 (SET HATCH) and section 4.106.66 on page 171 (SET PATTERN) for more details).

You can add the symbol, hatch or pattern colour to the symbol number as a qualifier, e.g. 32/red instead of just 32.

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.

Use OVERLAY/NTUPLE to plot variables from an Ntuple. You have to give the command SET NTUPLE ... first to specify the variables. Note that this form does not apply any cuts and is most useful for plotting different columns from a table of entries read in with DAT_FETCH for example.

4.88.1 /SAME

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

4.88.2 /DIFFERENT

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

4.88.3 /NEXT

Draws the next plot that is in memory. Useful for scanning a series of plots, following an `id` that you know.

4.88.4 /NTUPLE

Draws the specified variables from an Ntuple. Use the `SET NTUPLE ...` command to specify which variables to put on which axis.

4.88.5 /SMOOTH

Draws the plot as a smooth curve using a cubic spline interpolation between the points.

4.89 PARSE

```
Syntax:   PARSE command
where:    command  is a Mn_Fit command that can include formatting
```

Parses and then executes a command line. This command is useful if you want to convert a register, parameter etc. into a number to pass to another macro or defined command for example.

4.89.1 Examples

1. Pass the contents of a series of registers that contain the histograms to be plotted and the limits to be used:

```
! Histogram id's
deposit r1 = 8162
deposit r2 = 8180
deposit r3 = 7296
! Plot limits
deposit r11 = 200
deposit r12 = 400
deposit r13 =  50
show register 1:3
show register 11:13
! Plotting command
undef test_plot
define test_plot
set y limit 0 @2
plot @1
enddef
!
window 1 3 2 2
do i=1,3
  parse test_plot {ir@i,(i4)} {r2,(f6.1)}
enddo
```

test_plot will be executed 3 times with the following parameters:

```
test_plot 8162  200.0
test_plot 8180  400.0
test_plot 7296   50.0
```

2. This example illustrates how you can store the histogram identifiers that you want to work with in registers and then use the PARSE command to put the content of the registers into the command line. In this example the errors of all bins are set to 10:

```
dep r1 = 102
dep r2 = 1102
do i=1,2
  ! Get the number of bins etc. for the histogram into registers
  set plot r@i default
  do j=1,r131
    parse deposit dy{ir@i,(i4.4)}(@j) = 10.0
  enddo
enddo
```

Note the use of i4.4 to avoid spaces identifiers which have fewer than 4 digits.

4.90 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-D 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.

The commmand is also useful to set the limits and fill the arrays that contain the mean etc. for histograms and scatter plots that have been booked and filled using the BOOK and FILL commands, as Mn_Fit does not know when you have finished filling a plot.

4.91 PLOT

```
Syntax: PLOT[/option] id1[:id2] [&idb1:idb2]] [symb/col hatch/col patt/col]
    or  PLOT[/option] id part
    or  PLOT/NEXT [symb/col hatch/col patt/col]
 where: option   can be /CLEAR|/NOCLEAR|/NEXT/|/NTUPLE|/SMOOTH/|/EMPTY
        id       is the plot identifier
        idb      is the (optional) secondary identifier
        symb     is the symbol number to use (default from 'SET SYMBOL')
        hatch    is the hatching to use (default from 'SET HATCH')
        patt     is the pattern to use (default from 'SET PATTERN')
        part     is the part of an HBOOK histogram that you want to plot
                 e.g. BANX, FUN etc.
        col      is the (optional) colour for the symbol, hatch or pattern.
```

Alias for HIST PLOT. 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. To plot all histograms use the syntax PLOT 0. To plot the next histogram in memory use the form PLOT/NEXT. 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.

Use PLOT/NTUPLE to plot variables from an Ntuple. You have to give the command SET NTUPLE ... first to specify the variables. Note that this form does not apply any cuts and is most useful for plotting different columns from a table of entries read in with DAT_FETCH for example.

If you want to plot an empty frame, without any error messages, book a new histogram with HISTOGRAM BOOK and then plot it with PLOT/EMPTY.

If you omit the symbol, hatch or pattern the default will be used. If you specify a symbol this only applies to the current plot command and will not change the default. Use SET SYMBOL to change the default. You can add the symbol, hatch or pattern color to the symbol number as a qualifier, e.g. 32/red instead of just 32.

If you are using a colour as a shading for an overlay and need to get the ticks redrawn use the PLOT/NOCLEAR command and redraw the 1st histogram.

The normal procedure is to FETCH the histograms from a file and then plot them using the above syntax. You can also plot histograms directly from an HBOOK RZ file by opening the file (OPEN or HB_OPEN) and then giving the plot command. This feature can be suppressed using the SET AUTOFETCH OFF command. Note that this only works for single histograms and not for a range nor for all histograms.

If you want to check whether a histogram exists in a macro before deciding what to do you can use the following commands:

```
SET PLOT ida&idb DEFAULT
IF r121 = ida & r122 = idb
  ...
ENDIF
```

Note that this only works for histograms that you have tried to fetch. It does not yet work for histograms in an RZ file.

Many other options for plotting 2-D histograms exist using the 2DIM command. See section 4.3 on page 47 (2DIM) for more details.

4.91.1 /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 4.106.93 on page 178 (SET WINDOW)) this option will be overridden, except for the plot in the top lefthand corner (WINDOW 1 1).

4.91.2 /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). It can also be used if you make a plot with a pattern (or hatch -3), which obscures the scale. You can then say PLOT/NOCLEAR id for the 1st histogram again.

4.91.3 /EMPTY

Draws an empty plot without giving an error message. This is useful if you want the frame of a plot for drawing in etc.

4.91.4 /NEXT

Draws the next plot that is in memory. Useful for scanning a series of plots, following an `id` that you know.

4.91.5 /NTUPLE

Draws the specified variables from an Ntuple. Use the `SET NTUPLE ...` command to specify which variables to put on which axis.

4.91.6 /SMOOTH

Draws the plot as a smooth curve using a cubic spline interpolation between the points.

4.91.7 Examples

1. Use color shading as an overlay and put a black edge on all plots. Also put on a key in the same form:

```
set colour symbol yellow
plot 7 1 0 100
set colour symbol green
scale 6 6&1 0.9
overlay 6&1 1 0 100
set colour symbol black
plot/noclear 7
overlay 6&1 1
! Add a key with yellow symbol and black text
key 7 new 8100 'Signal + background'
  10 15 0.4 = = cm -1004 yellow = = black
! This form works, but the above method is simpler
key 7 new 2100 'Background'
  10 14 0.4 = = cm -1004 green = = black
! Draw a black edge around the symbol
draw symbol 12 black 0.4
  10 14
```

2. Same as part of the previous example with the colours in the `PLOT` command:

```
! Yellow symbol and black pattern - probably not what is wanted
plot 7 1/yellow 0 100
! Other way round
plot 7 1 0 100/yellow
scale 6 6&1 0.9
! All in green
overlay 6&1 1/green 0 100/green
! Default symbol colour is still black
plot/noclear 7
overlay 6&1 1/red
```

4.92 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 `SHELL` or `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.

4.93 PROJECT

```
Syntax: NTUPLE PROJECT id[&idb] var1 [var2 ...] [#weight [#dweight]] &idb2
        nbinx [xlo xhi]
        [nbiny ylo yhi] [nevt1 nevt]
where   id       is the ntuple identifier
        var1     is the first variable or expression to project onto
        var2     is the second etc.
        weight   is a variable name or number to be used as the weight
        dweight  is a variable to be used as the error on the weight
        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
        nevt1    is the first event to scan
and     nevt     is the number of events to scan
```

Projects a plot (either a scatter plot or an n-dimensional histogram) onto 1 or more axes. Alias for `NTUPLE PROJECT`. See section 4.86.6 on page 131 (NTUPLE PROJECT) for more details.

4.94 PWD

```
Syntax: PWD
```

Prints the currently selected HBOOK directory in an HBOOK4 file or the currently selected ROOT directory in a ROOT file.

4.95 READ

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

Reads in a series of commands from a file or a histogram stored in card image format. See section 2.7 on page 39 (READ Menu) for a short description of the commands.

4.95.1 COMMAND

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

Alias for `EXEC`. See section 4.42 on page 82 (EXECUTE) for more details.

4.95.2 DATA

```
Syntax: READ DATA filename
```

Alias for `DAT_FETCH`. See section 4.21 on page 63 (DAT_FETCH) for more details.

4.96 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.

4.97 REDRAW

```
Syntax: REDRAW
```

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

Note that what `REDRAW` actually does is to repeat the plotting commands that were used to make the picture. Thus if you change a histogram, by fetching in a new one with the same identifier or performing some operation on it after that histogram has been plotted, the new histogram will get drawn when you give the `REDRAW` command. Another, sometimes unexpected, side-effect is that if you are windowing and project an Ntuple several times with the same secondary identifier each time, but different cuts, the `REDRAW` command will plot the last histogram only for all the projections.

4.98 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 1.7 on page 11 (Numbers) and section 4.28 on page 69 (DEPOSIT) for details on defining user variables.

4.99 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`.

Note that `RENAME` only copies the Mn_Fit histogram and not any associated HBOOK historam. Use the `HRENAME` command to copy the HBOOK histogram also.

4.100 RETURN

```
Syntax: RETURN
```

Returns from a macro or `DEFINE`d command.

4.101 ROOT_FETCH

```
Syntax: ROOT_FETCH tid1[;id1][,tid2[;id2]]
    or  ROOT_FETCH 0
    or  ROOT_FETCH id1:id2
where   tid1,tid2  are the root histogram identifiers
        id1,id2    are the (optional) Mn_Fit identifiers to use.
```

Fetches ROOT histograms from a file. Note that you must use the command `ROOT_OPEN` to open the file before trying to fetch some histograms.

1-D, 2-D and 3-D histograms and prifle plots (1-D and 2-D) can be fetched. Ntuples cannot be fetched.

ROOT histograms have string rather than integer identifiers. If the identifiers is of the for `h102`, etc. then the first character will be stripped and the identifier converted to an integer. If the identifier is a string you can specify the integer identifier that it should be given using the syntax `tid;id`. At present, if you want to use a register of something for the identifier you have to use the `PARSE` command - see Example 4, as the identifier must be given as a number. You can use the `SET IDR` or 'SET ROOT_ID' commands to give a default identifier. See section 4.106.42 on page 158 (SET IDR) or SET ROOT_ID for more details.

If you give the command `ROOT_FETCH 0` all histograms in the current ROOT directory will be fetched. The will be given consecutive identifiers starting with the 1 or the value specified in the last `SET IDR` command.

If you want to fetch a range of histograms, then their identifiers must be of the form `hNNN`, e.g. `h101, h1, h998`.

Note that Mn_Fit calculates the number of entries, the lower and upper limits on the number of entries, as well the mean and RMS for each axis from the bin contents. At present it is not possible to access the root statistics, which are calculated also using underflows and overflows.

If the histogram number you ask to fetch already exists it will be overwritten. All the histograms will be given the default secondary identifier.

NOT SUPPORTED YET: You can use the secondary identifier to specify which cycle to fetch from ROOT files. The cycle number will be added to the current setting for the secondary identifier. Use the `LDIR` command to see which cycle numbers exist.

If your histograms are in several subdirectories, you should use 'SET DIRECTORY' or 'CDIR' command 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 `LDIR` and `CDIR` commands to list what is in the directory, before giving the `ROOT_FETCH` command.

Note that the top level directory name for the file is `//root`. Use the command `CDIR //root` (`//root` is case sensitive) to get to the top-level directory.

4.101.1 Examples

1. Get all plots from a file:

```
root_open filename
ROOT_FETCH 0
```

2. Get some plots from a file where the IDs are h101, h102 etc:

```
root_fetch h101,h102,h103
```

3. Get some plots from a file where the IDs are strings:

```
root_fetch hpx;10,hpy;11,hpz;12
```

4. Get all the plots from various subdirectories:

```
ROOT_OPEN filename
SET DIRECTORY dir1 IDB 100 ENDSET  FETCH h101,h102
SET DIRECTORY dir2 IDB 200 ENDSET  FETCH h101,h102
SET DIRECTORY dir3 IDB 300 ENDSET  FETCH hpx;101,hpy;102
```

   Note that the directory name you give is relative to the top directory.

5. Use a `DO` loop to calculate and fetch the histograms that are required:

```
root_open filename.root
do i=1,10
  fetch h@i
  deposit r1 = 1000 + @i
  parse fetch h{ir1,(i4)};{ir1,(i4)}
enddo
```

   The `PARSE` command is needed here, because the integer identifier must be given as a number.

4.102 ROOT_OPEN

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

Opens an ROOT file. Unlike HB_OPEN this command is necessary for ROOT, as the filename cannot be specified in the ROOT_FETCH command. Use the commands CDIR and LDIR (or SET DIRECTORY and SHOW DIRECTORY) to change directories or list directroy contents. The root file will have the directory name //root, so use the command cd //root to chnage to the top-level directory. You can then find out what histograms are in a file and fetch histograms from several subdirectories and give them separate secondary identifiers.

This command is only available is Mn_Fit has been compiled with the ROOT interface turned on.

4.103 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.

4.104 SCAN

```
Syntax: SCAN id[&idb] filename[(arg)] Y|N [nevt1 nevt]
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
        nevt1     is the first event number to scan
        nevt      is the number of events to scan
```

Loops over all the events in an Ntuple and calls a function which is contained in filename. Alias for NTUPLE SCAN. See section 4.86.7 on page 134 (NTUPLE SCAN) for more details.

4.105 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`.

4.106 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 1.7 on page 11 (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 4.106.68 on page 171 (SET PLOT) and section 4.106.19 on page 150 (SET DISPLAY) for more details.

If a parameter has to be preceded by the axis it applies to, you do not 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 `ALL`, `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.

4.106.1 ABORT

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

Turns on or off the aborting of macros when an error occurs. The default behaviour is to ask if the macro or defined command should be aborted. If this option is turned off the macro will continue whether an error occurs or not.

4.106.2 ALIAS

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

Turns on or off alias translation on the command line. See section 4.5 on page 54 (ALIAS) for full details on aliases.

4.106.3 AUTOFETCH

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

Automatically refetches Ntuples when they are projected or scanned. This means you are sure that everything is setup OK before the operation on the Ntuple. However for very big Ntuples, even though they have been made with the disk option, this can take a long time. If you are sure you know what you are doing you can turn this option off, but you must be sure not to change the HBOOK file or current directory after you have fetched the Ntuple.

With this option turned on, if you try to plot a histogram that does not exist, Mn_Fit will try to fetch it from an open HBOOK RZ input file.

4.106.4 AUTOSCALE

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

Enables automatic rescaling of text sizes, if they are set to their default values, when windowing. By default the sizes are reduced by a factor of 2/3 of the number of windows in the x direction. This is sometimes too much, especially when preparing a plot for a publication.

4.106.5 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`.

4.106.6 AUTOTRIM

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

Enables automatic dropping of the last scale value when windowing with 0 spacing between the plots.

4.106.7 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 4.106.47 on page 159 (SET LABEL) for more details.

4.106.8 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.

4.106.9 BIN

```
Syntax:  SET BIN SCALE|OFFSET value
where:   value  is the scale factor or offset
```

Sets a scale factor or an offset for each bin. This command works only for 1-D histograms when they are plotted as such, i.e. symbols 1->10. The command is useful if you want to show each bin of a binned histogram, or have several histograms with the same binning that you want to compare next to each other. If you want to shift the points of a histogram you can also use `XSCALE` and `XSHIFT` commands, but these permanently modify the histogram.

4.106.10 BOX

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

Turns on or off drawing a box around the plot.

4.106.11 BREAK

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

Turns on or off the condition handler.

4.106.12 CHARACTER

```
Syntax:  SET CHARACTER COMMENT|CONTINUATION char
where:   char   is the comment or continuation character
Default: Comment is '!', continuation is '-'
```

Sets the comment or continuation line characters. The defaults are the same as in VMS DCL: '!' for comments and '-' for continuation lines. Command lines in Mn_Fit can have a maximum of 255 characters.

You should usually include the character in quotes to avoid it being interpreted as a comment character, which would mean that it would be ignored.

You can list the comment and continuation characters using the command `SHOW CHARACTER`. This can only be abbreviated as fas as `SHOW CHARA` as otherwise it will show character array elements - see section 4.28 on page 69 (DEPOSIT) for more details on the character array.

Examples

1. Set the comment character and continuation line character to be the same as Unix:

```
SET CHARACTER COMMENT '#'
SET CHARACTER CON     '\'
```

4.106.13 COLOR

```
Syntax: SET COLOR [item] color|ON|OFF|DEFAULT
    or: SET COLOR REPRESENTATION index red green blue name
where:  item    is the name of what you want to change
        color   is the name or number of the color you want
        name    is the (optional) name for the colour
```

Exactly the same as `SET COLOUR` for Americans!

4.106.14 COLOUR

```
Syntax: SET COLOUR [item] colour|ON|OFF|DEFAULT
    or: SET COLOUR REPRESENTATION index red green blue name
where:  item    is the name of what you want to change
        colour  is the name or number of the colour you want
        name    is the (optional) name for the colour
```

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

You can also turn on or off the use of colour with the command `SET COLOUR ON|OFF`. This is useful to convert a colour macro into a black and white one for including figures in a paper.

The default colour map is that of HIGZ:

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

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

```
FRAME              TICK               SCALE
LABEL              HEADER             TITLE
SYMBOL             HATCH              PATTERN
ZERO_LINE          COMMENT            FIT
BACKGROUND         REPRESENTATION
```

Note that `SET COLOUR 0` sets colours to default (i.e. black) while `SET COLOUR WHITE` will set the colour to the background colour.

The command `SET COLOUR REPRESENTATION index red green blue name` changes the colour representation for that index. This enables you to customize your colour indices if you wish. `index` is the colour index, `red, green, blue` are the fractions of red, green and blue for that index. They must be between 0 and 1. If the index is larger than 7 then you must also give a name. For lower indices, you can give a name. The name MUST be a single word. Note that the `REPRESENTATION` command may only change the colours for the currently active output device, i.e. the screen or the file. If this is the case, then to change things for both (assuming you are using a X display and a Postscript file) you have to give the commands:

```
capture display
set col rep 1 0.3 0.4 0.5
cap post
set col rep 1 0.3 0.4 0.5
capture display
```

You can define up to a total of 20 colours. The colour indices should be continuous. To help you get started with your own definitions, the colour fractions for the default HIGZ colour scheme are:

Index	Colour	Red	Green	Blue
0	Background (white)	1.0	1.0	1.0
1	Foreground (black)	0.0	0.0	0.0
2	Red	1.0	0.0	0.0
3	Green	0.0	1.0	0.0
4	Dark blue	0.0	0.0	1.0
5	Yellow	1.0	1.0	0.0
6	Magenta	1.0	0.0	1.0
7	Cyan	0.0	1.0	1.0

Note that the `SET COLOUR DEFAULT` command now sets all colours back to the HIGZ colour map. This has changed with Mn_Fit version 4.07. For earlier version the command `SET COLOUR ON` also reset the colour map.

4.106.15 DBASE

```
Syntax:   SET DBASE YEAR|LIMITS
```

Sets the database year to access or the limits on the values that are included in calculating the averages for `DATABASE DB_HISTORY`.

4.106.16 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 much at the moment, it is a useful debugging tool for me.

4.106.17 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 4.106.40 on page 157 (SET HISTOGRAM) for more details), and fills registers 121-199 as well as 231-257 with information on the histogram and defines the default plot for the DEPOSIT command. See section 1.7 on page 11 (Numbers) for a list of what is in the registers.

4.106.18 DIRECTORY

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

Sets the current HBOOK or ROOT directory name to dirname. Note that the HCDIR (or ROOT equivalent) 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 SET DIR 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. You can use either \ or '..' to go up a directory level.

ROOT directory names are always with respect to the top level directory. Therefore it make no sense to try to use '..' to go up a level. Use the command 'cd //root' to get to the top level. ROOT directories are case sensitive. HBOOK directories always get converted to upper case.

Note that the Mn_Fit HBOOK file top level directory name is //MN_HBIN, while the ROOT top level directory name is '//root'.

4.106.19 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.

4.106.20 DSIZE

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

Controls the scale factor for the dot size. This command only works in the HIGZ version of Mn_Fit. The effect is only usually seen when you make a Postscript version of the figure.

4.106.21 DUMP

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

Change the output unit for the DISPLAY, FIT_INFO, DUMP, HIST DUMP, INDEX and MESSAGE commands. If you specify LPT or > 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. Alternatively you can use the syntax SET DUMP > filename to both reroute the output and start sending it to a file. If a file is already open you must close it first using SET DUMP CLOSE, otherwise the output will just be appended to the open file.

4.106.22 ECHO

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

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

4.106.23 EDIT

```
Syntax: SET EDIT command
```

Changes the command used to invoke the editor. The default editor is EDIT/TPU on VMS, dm on the Apollo and emacs on Unix. If the environment variable EDITOR exists (Unix), this will be used as the default edit command.

4.106.24 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.

4.106.25 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, EFFICIENCY, AVERAGE is applied to the plot.

4.106.26 EXCLUSIONS

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

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

4.106.27 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.

4.106.28 FIT

```
Syntax: SET FIT option
 where: option  is the fit option
```

Sets options for fitting.

DEFAULT

```
Syntax: SET FIT DEFAULT
```

Sets all fitting and function options associated with fitting to their default values.

AREA

```
Syntax: SET FIT AREA ON|OFF
```

Uses Simpson integration to calculate the area of fragmentation functions, dipion invariant mass functions and ARGUS background functions. Slows down the fitting, but means that the `AREA` is meaningful.

COMMANDS

```
Syntax:   SET FIT COMMANDS
          command1
          command2 ...
          END
 where:   comand1, comand2, ...   are a list of commands
```

Defines the list of commands to be used when using the quick fitting option, e.g. `FIT/GAUSS`, where the functions to use are included in the `FIT` command. The default commands are `MINIMIZE`, `DISPLAY`.

CONVOLUTE

```
Syntax:   SET FIT CONVOLUTE ON|OFF [width ninterval nsigma]
where:    width      is the width of the Gaussian
          ninterval  is the number of intervals to integrate over
          nsigma     is the number of sigmas of the Gaussian
Defaults: width = 1, ninterval = 100, nsigma = 3
```

WARNING: This command's results should be carefully checked to ensure that they are correct. Its syntax could also change in response to suggestions.

Convolutes the fitting function(s) with a Gaussian resolution function. Useful for things such as lifetime fits, where you want to include the experimental resolution.

You should normally set the Gaussian width. Increasing the number of intervals slows down the fitting, but gives a more precise result. The same applies to the number of sigma. At present it is not worth going beyond 6 sigma, as I set a Gaussian to 0 there.

INTEGRATE

```
Syntax: SET FIT INTEGRATE ON|OFF [ninterval]
where   ninterval  is the number of intervals in the integration.
Defaults: ninterval = 100
```

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.

4.106.29 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 = -1004
```

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 omit the item, the font of all items will be changed. The following items are valid:

```
HEADER              TITLE               COMMENT
SYMBOL              SCALE               LABEL
```

Font 2000 uses the IGTEXT routine (see section 1.12 on page 18 (TEXT) 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 -1004` will choose font -4 (Helvetica) with precision 1.

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.

4.106.30 FOOTER

```
Syntax: SET FOOTER ON|OFF|USER [text]                    (default = OFF)
```

Controls whether to print the footer that shows the date and time that a picture was made or user specified text. For user specified footer give the text that should be in the footer. Use the command `SET FSIZE` to change the size of the text.

In a user specified footer the following strings have special meaning:

```
@file    Filename for the hardcopy
@date    The date
@time    The time
@|       Splits the text at this point, putting the text before at
         the bottom left-hand corner of the picture and the text
         after at the bottom right-hand corner.
```

For example to get the date and time at the bottom left-hand corner and the hardcopy filename at the bottom right-hand corner give the command:

```
SET FOOTER USER 'Mn_Fit @date @time@|@file'
```

4.106.31 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]`.

4.106.32 FSIZE

```
Syntax: SET FSIZE size                                    (default = 0.2cm)
```

Controls the size of the footer text if the footer is turned on (command `SET FOOTER`).

4.106.33 FUNCTION

```
Syntax:   SET FUNCTION option
 where:   option  is the function option
```

Set options used in calculating functions. See section **??** on page **??** (SET FUNCTION Menu) for a short description of each command. Use the `SET BACKGROUND` and `SET SIGNAL` commands to specify which functions are background and which are signal for the different `DISPLAY` modes.

The `SET FUNCTION AREA` command controls whether the `AREA` given is evaluated inside the plot limits or from -infinity to +infinity. In fact what is actually used are the orthogonality limits used to also define Chebyshev and Legendre polynomials. These are by default the plot limits when fitting and can be set using the `SET ORTHOGONAL` command.

Use the command `SET FIT AREA ON|OFF` to turn on/off use of Simpson integration to calculate area of fragmentation functions, dipion invariant mass functions and ARGUS background functions. Slows down the fitting, but means that the `AREA` is meaningful.

The options are:

```
AREA      Specifies whether the AREA is that inside the plot or over
          the whole valid range.
BIN_WIDTH Controls whether the bin width is used when calculating
          the value of a function.
POINTS    Sets the number of points used when drawing a function.
INTEGRATE Sets the number of intervals used when integrating a function.
STEP      Sets the step size used for differentiation of a function.
```

AREA

```
Syntax:   SET FUNCTION AREA LIMIT|FULL
where:    LIMIT means the area is calculated using the plot limits
          FULL  means the area is the integral from -infinity to +infinity
Default:  FULL
```

Specifies if a function will be integrated over the plot limits or from -infinity to +infinity when the area is calculated. For functions with long tails, such as the Breit-Wginer or Crystal

Ball functions, the area outside the plot limits can be significant and leads to the AREA parameter being apparently too high.

This parameter is used with all versions of a Gaussian, Breit-Wigner Landau and the Crystal Ball functions (normal and reversed). The area inside the plot limits is evaluated numerically using Simpson integration. The number of intervals used is specified with the SET FUNCTION INTEGRATE command.

In fact what is actually used for the limits are the orthogonality limits used to also define Chebyshev and Legendre polynomials. These are by default the plot limits when fitting and can be set using the SET ORTHOGONAL command.

This option is set to the default value using the SET FIT DEFAULT command.

BIN_WIDTH

```
Syntax:   SET FUNCTION BIN_WIDTH ON|OFF
```

Specifies whether the bin width should be used when calculating the value of a function. This now applies to all functions, as they all include the bin width in their calculation.

INTEGRATE

```
Syntax:   SET FUNCTION INTEGRATE nint
where:    nint  is the number of intervals
Default:  nint=100
```

Specifies how many intervals will be used when integrating a function. This is used in the INTEGRATE command and the expression FINT.

POINTS

```
Syntax:   SET FUNCTION POINTS npnt
where:    npnt  is the number of points
Default:  npnt=500
```

Specifies how many points to use when drawing a function.

STEP

```
Syntax:   SET FUNCTION STEP size
where:    size  is the step size used when differentiating a function
Default:  step = 1.0
```

Specifies the step size used when differentiating a function. It is also 1000000 times the step size used when evaluating the value or derivative of a function just below the given parameter value (FNEG).

4.106.34 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]`.

4.106.35 GSIZE

```
Syntax: SET GSIZE value                    (default = 0.4cm)
```

Changes the size of the global and/or user 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 GPOSITION` command to do this.

4.106.36 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.

4.106.37 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: Specifies the distance between each hatch;
j: Specifies the angle between 90 and 180 degrees;
k: Specifies the 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.

4.106.38 HEADER

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

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; `DEFAULT` also shows the area, mean and r.m.s. for the plot; `COMPLETE` also show the number of underflows and overflows. Alias for `SET IDSHOW`.

4.106.39 HIGZ

```
Syntax: SET HIGZ name value
    or: SET HIGZ TABLE option val1 [val2...]
 where: name       is the IGSET parameter name
        value      is its value
        option     is the IGTABL option
        val1,val2  are IGTABL parameters
```

Changes the HIGZ IGSET parameters or sets the plotting mode and parameters for the 2-dimensional histogram plotting routine IGTABL. The command `SET HIGZ TABLE` is an alias for `SET IGTABLE`. See section 4.106.46 on page 158 (SET IGTABLE) for more details.

TABLE

```
Syntax: SET HIGZ TABLE option val1 [val2...]
```

Alias for `SET IGTABLE`. See section 4.106.46 on page 158 (SET IGTABLE) and 2DIM for more details.

4.106.40 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 as well as 231 - 257 with information on the histogram. See section 1.7 on page 11 (Numbers) for a list of the variables stored.

4.106.41 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`.

4.106.42 IDR

```
Syntax: SET IDR n
```

Sets the default ROOT histogram identifier. This is used if the ROOT identifier string cannot be converted to a number and if the identifier is not give with the ROOT_FETCH command. Alias for SET ROOT_ID.

4.106.43 IDSHOW

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

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; DEFAULT also shows the area, mean and r.m.s. for the plot; COMPLETE also show the number of underflows and overflows. Alias for SET HEADER.

4.106.44 IDSIZE

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

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

4.106.45 IGARC

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

Uses the HIGZ IGARC routine for drawing arcs and circles. This routine appears to have a bug and sometimes ignores what you set for the line thickness and colour.

4.106.46 IGTABLE

```
Syntax:  SET IGTABLE option [parameters]
where:   option      is the option for drawing the table
         parameters  are the associated parameters
```

Specifies how to draw a 2-D histogram. All HIGZ IGTABL modes are available: SCATTER, BOX, ARROW, CONTOUR, COLOUR, TEXT, CHAR, LEGO, LEGOC1, LEGOC2, SURF, SURFC1, SURFC2, SURFCONT, SURFSHADE with extra options for the coordinate system for LEGO and SURFACE plots: POL, CYL, SPH, PSD. Note that the extra LEGO and SURFACE options cannot be abbreviated.

The parameters such as the minimum and maximum bin numbers and the lower and upper limits on the z axis are set using the normal SET X|Y|Z LIMIT commands. The parameters you should give here are the lego or surface plots viewing angles, the number of contours, the contour distinguishing mode and the z values for the contours (if required).

See section 4.3 on page 47 (2DIM) for a fuller description of the options and parmaeters. Even more details can be found in the HIGZ manual [3]. Certain features are not in the manual! For example you can specify the colours of coloured lego and surface plots by giving the numbers of the colours as parameters after the viewing angles.

4.106.47 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.

Using `option` left puts the label aligned with the left-hand or bottom edge of the plot and left adjusted. With `option` right it is aligned with the right-hand or top edge and right adjusted (as in the PAW default).

For the lego plots axes the offsets for the x and y axes are flipped to match the scale offsets. In addition the x and y offsets for the y-axis are flipped, so that by default the label will be put at the x offset below the centre of the axis.

4.106.48 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]`.

Note that when you start a fit all limits get reset. This means that you have to give the command `SET X|Y|Z MODE LOG ...` or `SET X|Y|Z LIMIT ...` if you want to get the limits that you were using again.

In addition when you switch to or from a log scale (`SET X|Y|Z MODE LOG`) the appropriate limits also get reset, in order to avoid trying to calculate the log of 0.

If you want to specify date and/or time limits you must first set the plotting mode to `DATE` or `TIME` and then you can give the limits in the form `YYMMDD.HHMMSS`, `YYMMDD` or `HH:MM`.

If you want to set or change the limits for fit parameters use the command `MODIFY` when inside MINUIT, or `DEPOSIT LOLIMn(m)` and `DEPOSIT HILIMn(m)`. See section 5.31 on page 199 (MINUIT MODIFY) and section 4.28 on page 69 (DEPOSIT) for more details.

4.106.49 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`.

The commmand for a log scale on a plot is `SET X|Y|Z MODE LOG`.

4.106.50 LSIZE

```
Syntax:  SET LSIZE size                              (default = 0.01)
```

Sets the segment size for drawing lines - calls IGSET('BASL',size). The segment size is in Normalized Device Coordinates (NDC), which means that effectively the size is a fraction of the picture size.

4.106.51 MANUAL

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

Turns on or off the special logfile syntax for making the Mn_Fit manual.

4.106.52 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.

4.106.53 MODE

```
Syntax:  SET X|Y|Z MODE REAL|INTEGER|LOG|DATE|TIME [low ndecade]
Default: INTEGER
```

Changes the way the scale is drawn on the x or y axis. For a log scale you can give the lower limit and the number of decades that you want to plot. The following modes are available:

```
REAL    = real numbers
INTEGER = integer numbers
LOG     = log scale. You can give the lower limit  and  the  number of
          decades you want to plot.
DATE    = day of month, with month also printed.
TIME    = time of day.
```

To change the mode for a particular plot, precede the command with `SET PLOT id [&idb]`. For a log scale with automatic scaling the nearest decade above and below the calculated plot limits is used. If the lower limit is thus zero, then 3 decades are plotted.

Note that when you start a fit all limits get reset. This means that you have to give the command `SET MODE LOG ...` or `SET X|Y|Z LIMIT ...` if you want to get the limits that you were using again.

In addition when you switch to or from a log scale (`SET X|Y|Z MODE LOG`) the appropriate limits also get reset, in order to avoid trying to calculate the log of 0. `SET Z MODE LOG` works for lego and surface plots and when the area of the symbol is proportional to the number of entries.

If you want to specify date and/or time limits you must first set the plotting mode and then you can give the limits in the form `YYMMDD.HHMMSS` or `HH:MM`.

4.106.54 MOUSE

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

Turns on or off using the mouse to specify the position of comments/keys and items that you draw. See section 4.17 on page 57 (COMMENT) and section 4.71 on page 123 (KEY) for more details. There is also a general section on usage of the mouse at the beginning of the Mn_Fit manual.

4.106.55 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.

4.106.56 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.

4.106.57 NTUPLE

```
Syntax: SET NTUPLE NAME|PLOT|VARIABLE
```

Commands for setting options for Ntuples. `PLOT` specifies the Ntuple variables to be plotted with the `PLOT/NTUPLE` command. `VARIABLE` specifies the extra CWN variables that have to be fetched. `NAME` gives the names of Ntuple variables that are used in subsequent `HISTOGRAM BOOK` commands.

NAME

```
Syntax: SET NTUPLE NAME var1,var2,...
where   var1  is an Ntuple variable name
```

Gives a list of variable names that are used for subsequent `HISTOGRAM BOOK` commands for Ntuples.

PLOT

```
Syntax: SET NTUPLE PLOT id[&idb] X|Y|Z var1 [X|Y|Z var2 [...]]
 where: id    is the  Ntuple  identifier  (0 = all Ntuples)
        idb   is the (optional) Ntuple secondary identifier
        var1  is the Ntuple variable name to be associated
              with  the  specified  plot  axis
```

Specifies which Ntuple variables will be plotted on which axes when using `PLOT/NTUPLE` command. As usual the Ntuple variable can be specified as a name or a number.

VARIABLE

```
Syntax: SET NTUPLE VARIABLE var1,var2,...
where   var1  is an Ntuple variable name or '$CLEAR'
```

Gives a list of CWN variable names that should be fetched when a projection of a CWN is made. This command is used to list the variables that need to be accessed in a COMIS function used to make cuts on a CWN.

4.106.58 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 4.106.98 on page 180 (SET ZERO) for more details.

4.106.59 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.

4.106.60 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_TIM     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 DNX DPX DNY DPY
```

4.106.61 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`.

These limits are also used for evaluating the area under the Gaussian, Landau, Breit-Wigner and Crytal Ball functions if the command `SET FUNCTION AREA LIMIT` has been given.

4.106.62 PAGER

```
Syntax: SET PAGER command
```

Changes the command used to invoke the pager. The default pager is `TYPE/PAGE` on VMS, `dm` on the Apollo and `more -e` on Unix. If the environment variable `PAGER` exists (Unix), this will be used as the default edit command.

The pager is used with the `HELP` command on Unix machines and the `DUMP` command. If you give the command 'SET PAGER '", then no pager will be invoked.

4.106.63 PAPER

```
Syntax: SET PAPER type
 where: type  is the paper type A4,A3,A2,A1,A0,Letter,Legal or Ledger
```

Sets the paper type for future `HARDCOPY` commands. This command must be given before you open the output file:

```
set paper letter
hard post
close
```

The command also works with the `CAPTURE` command. Plots will be centred on the paper.

4.106.64 PARAMETER

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

Sets parameters for the L3 or ZEUS 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 =.

ECAL

```
Syntax: SET PARAMETER ECAL mode xlo xhi ylo yhi [nres]
 where: mode    selects the display mode
        xlo     minimum box|crystal number in phi
        xhi     maximum box|crystal number in phi
        ylo     minimum crystal number in theta
        yhi     maximum crystal number in theta
        symbol  is the symbol to use
        nres    is the number of subdivisions for the BGEO display
```

There are 2 ECAL displays available. The original is a grid of the BGO barrel written by Anne Heavey. A modified version of Vinod Gupta's BGEO package is also available. This shows either the +Z or -Z side of the BGO in the form of concentric circles.

Anne Heavey's version is described in the `Barrel` subtopic. The following describes the BGEO display mode.

The command `SET PARAMETER` must precede the command `DISPLAY ECAL id[&idb]`. The mode for display is given in the form `+/-kji` where:

```
k = 1   means show the -Z or +Z detector using BGEO.
        It needs a 160x41 2-D histogram or a 160x82 histogram
        or an Ntuple.
j = 0   means length of side proportional to entries
  = 1   means area proportional to entries
  = 2   means use a log scale with length of side proportional to
        the log of the number of entries
i = 0   means use line thickness - quicker, but not visible on all
        screens
  = 1   means use filled areas
  = 2   means use a symbol for each crystal with an entry.
```

Lower and upper limits in theta can be set for the BGEO display, but the only allowed combinations are:

```
-41  -1 (-z)           or     1 41 (+z)
-41 -25 (-z endcap)    or    25 41 (+z endcap)
-24  -1 (-z barrel)    or     1 24 (+z barrel)
```

If you have a 2-D histogram with all the BGO detector, i.e. 160x82, you must give the theta range to show. For an Ntuple you must also give the theta range.

If the data are in the form of a 2-D histogram it should always have 160 bins for the x-axis. If the crystal number in theta is `it`, the y-axis should be booked and filled with one of the following ranges:

```
41 bins   0.0 -> 41.0      Fill with float(iabs(it))-0.5
82 bins -41.0 -> 41.0      Fill with sign(float(iabs(it))-0.5,it)
83 bins -41.0 -> 42.0      Fill with float(it)+0.5
84 bins -42.0 -> 42.0      Fill with sign(float(iabs(it))+0.5,it)
```

In the case of 83 bins, the 42nd bin will be ignored. In the case of 84 bins the 42nd and 43rd bin will be ignored.

If the data is in the form of an Ntuple, the Ntuple must have 2 or 3 variables in the order phi, theta (energy). If your Ntuple has more variables or a different order use the `NTUPLE PROJECT` command to get the right number and order.

The BGO detector is divided into 16 boxes in phi, and you can select which box(es) you want to display. Box 1 starts at crystal phi=2 in the barrel and phi=1 in the endcap.

A positive number for the mode means averages will be calculated for the display. A negative number means totals will be calculated.

For modes i=0,1 a colour code is used to distinguish the value of each crystal. It is evaluated as a fraction of the allowed range:

```
0.00 -> 0.25   Black
0.25 -> 0.50   Blue
0.50 -> 0.75   Green
0.75 -> 1.00   Red
```

The lower and upper limits can be set using the `SET Z LIMIT` command.

For mode 2 a symbol is drawn for each crystal with an entry above the lower limit. The symbol size is set with the `SET SSIZE` command and the colour with the `SET COLOUR SYMBOL` command.

The BGEO display can be combined with the ECAL database interface to plot database entries in a more useful form than just as histograms. The display also fills HBOOK histograms 98001-98016 for each readout box and 98100 with all the data in a 1-dimensional histogram. You can use the command `HB_MN_FIT` to copy these histograms into Mn_Fit memory and then plot them.

For the BGEO display the following `SET` commands can be used to control the frame etc:

```
SET FRAME BOTTOM ON|OFF    Turns on/off drawing of the box boundaries
SET FRAME LEFT   ON|OFF    Turns on/off drawing of the ring boundaries
SET LABEL BOTTOM ON|OFF    Turns on/off printing -Z or +Z label
SET LABEL TOP    ON|OFF    Turns on/off printing of phi and theta numbers
SET SCALE BOTTOM ON|OFF    Turns on off mean or total values for
                           each quadrant and all crytals
SET SCALE LEFT   ON|OFF    Turns on/off mean or total values for
                           each box
```

The sizes of the text are 2/3 of the corresponding `LABEL` or `SCALE` sizes, except for the `-/+Z` label which is drawn with the title size. The thicknesses and colours of the lines or text are also set with the corresponding `SET COLOUR` or `SET THICKNESS` commands, except for the ring boundaries whose colour and thickness is set using the `SET COLOUR|THICKNESS ZERO_LINE` command.

Barrel

```
Syntax: SET PARAMETER ECAL mode xlo xhi ylo yhi
 where: mode  10     displays a status histogram using a color scale
              21-24  displays a 2-D histo using a color scale:
```

```
                21 for energy
                22 for ADC
                23 for number of events
                24 for temperature
          -2 displays a 2-D histo as a table with (real) values
             printed in the grid squares
xlo   minimum crystal number in phi that you choose to display
xhi   maximum crystal number in phi
ylo   minimum crystal number in theta
yhi   maximum crystal number in theta
```

This mode is only able to display the barrel and because of that has mostly been superceded by the BGEO display. However some of the options are still not fully implemented in the BGEO display, e.g. the status display.

For `mode = 10` there are specially filled 2-D histograms in `ECAL$HIST:ECAL0000.HFILE`, directory: `ECAL/DETECTOR_STATUS`.

For modes 2 and -2, 2-D histograms can be found in the directory `ECAL/DATA`. `xlo` and `xhi` (the phi values) should be between 1 and 160 inclusive, and `ylo` and `yhi` (the theta values) should be between -24 and -1, and 1 and 24. `xhi` > `xlo` and `yhi` > `ylo` are expected. The y region to be displayed may span the two regions, for example `ylo=-12, yhi=12` is acceptable. Entering 0 for any of the four last parameters yields the default value, which for `xlo` and `ylo` is the lowest crystal number for which the histogram was booked, and for `xhi` and `yhi` is the highest.

Examples

1. Get the -Z detector low energy pedestal widths and compare them with the previous entry in the database. Plot the difference in MeV (note the database stores the entries in uV and the conversion factor is 1 MeV = 50 uV):

```
set idb 2
! The first 2 zeros mean get all channels, the 3rd means get the
! data for now.
database db_snap/noplot //dbec/electronics/beam/pedestals 0 0 0 d 102
zscale 98767 988767&12 0.02
set default
set
  y psize 25
  y size 21
  wind 1 2 0 0
  y wmarg 4
  y wsize 17
endset
set par ecal 111
disp ecal 98767&12
hb_mn_fit
set y wmarg 0
set y wsize 4
plot 98100
```

FBGO

```
Syntax: SET PAR FBGO mode side sector emin emax
```

```
where:  mode   can be -3 -> 4           (default = 1)
               -5   Floating point numbers (value and error) (F7.3)
               -4   Integers (value and error)
               -3   Status (Integers -9 -> 9)
               -2   Floating point numbers (F7.3)
               -1   Integers
                1   Traffic light -9 -> -1 red
                               0 ->  0 no colour
                               1 ->  7 red
                               8 ->  9 yellow
                2   Energy colour scale
                3   Status in colour
                4   Trapezia with area proportional to energy
                    (default scale is 0 -> 50)
                5   Trapezia with area proportional to energy
                    (default scale is 0 -> maximum number of entries)
                6   Trapezia with area proportional to energy, not filled
                    (default scale is 0 -> 50)
                7   Trapezia with area proportional to energy, not filled
                    (default scale is 0 -> maximum number of entries)
                8   Trapezia with area proportional to energy
                    Filled if > emin, not filled if < emin
                    (default scale is 0 -> maximum number of entries)
        side    Side to show,
                0 means show -z and +z sides  (default)
                1 means -z
                2 means +z
        sector  sector number to show
                0     means show all sectors      (default)
                nnss  means show nn sectors starting at sector ss
        emin    is the minimum energy to show (default = 0)
        emax    is the maximum energy (default = 50 - modes 4,6
                                              = max # entries - others)
```

Sets the parameters for the forward luminosity monitor display. The colour of the frame can be set with `SET COLOUR FRAME`, that of the contents with `SET COLOUR SYMBOL`, that of the title and `+Z, -Z` indicators with `SET COLOUR TITLE` and that of the sector numbers with `SET COLOUR SCALE`. The size of the `+Z, -Z` labels is controlled with the `SET TSIZE` command. The size of the sector numbers can be controlled with `SET USIZE`, of the text with the contents with `SET SSIZE`. The font for the title is set with the `SET FONT TITLE nfont` command, that of the sector numbers with `SET FONT SCALE nfont`. This font is also used for the contents. The key colours and fonts are the same as those for `COMMENTS`. The line thickness can be set in the same way as the colours using the `SET THICKNESS` commands. The font used for the symbols can be set using the `SET SYMBOL FONT` command. The frame can be turned on or off using the `SET FRAME` command and the sector labels can be turned on or off using the `SET X SCALE` command. The format used for modes -2 and -5 can be controlled by using the number of digits and the number of decimal places in the `SET X SCALE` command.

FSIL

```
Syntax: SET PAR FSIL mode side layer sector
where:  mode   can be  1
        side    Side to show,
                0 means show -z and +z sides
```

```
                1 means -z
                2 means +z
        layer   is the layer number to show
        sector  is the sector number to show
                0 means show all sectors
Defaults: 1 0 0 0
```

Sets the parameters for the forward luminosity monitor silicon strip display. The default is to show all wafers. Switching on and off of the frame etc. and also the colours, fonts and line thicknesses are controlled in a way very similar to the FBGO display.

FWCH

```
Syntax: SET PAR FWCH mode side chamber strip emin emax
where:  mode   can be  -1 -> 4           (default = 1)
               -1   Integers
                1   Traffic light -9 -> -1 red
                                   0 ->  0 no colour
                                   1 ->  7 red
                                   8 ->  9 yellow
                2   Pulse height colour scale
                4   Hatching scale
        side    Side to show,
                0 means show -z and +z sides
                1 means -z
                2 means +z
        chamber is the chamber number to show
        strip   is the strip type to show
                0 means show all strips
                1 means r strips
                2 means phi strips
                3 means x strips
                4 means y strips
        pmin    is the minimum pulse height
        pmax    is the maximum pulse height
Defaults: 1 0 0 0 0.0 60.0
```

Sets the parameters for the forward luminosity monitor planar chamber display. The default is to show all 8 chambers and all strip types. Note that chambers 1 and 2 have strip types 3 and 4, and 3 and 4 have strip types 1 and 2.

The colour code is blue, green, yellow, orange, red, where blue is below the limit set and red is above. Similarly hatching increases in intensity, blank means below limits and solid means above the limits.

Note that modes -3,-2 and 3 have not yet been implemented.

FTD

```
Syntax: SET PAR FTD mode chamber layer
where:  mode     Viewing mode - the various parts can be combined
                 Front view:
                  1 VC tracks
                  2 FTD segments
                  4 FTD LTEs
                 Side view:
```

```
                  101 VC tracks
                  102 FTD segments
                  104 FTD LTEs
                  108 FTD hits
                  111 VC tracks + FTD hits
                  112 FTD segments + FTD hits
                  114 FTD LTEs + FTD hits
         chamber  The chamber number
                  1,2,3 mean FTD1, FTD2, FTD3
                  4 means RTD
         layer    is the layer number to show (not yet implemented)
Defaults: 3 0 0
```

Sets the parameters for the ZEUS forward tracking chambers. The default is to show the extrapolated VC tracks and the FTD segments. It is also possible (but not yet implemented) to show the LTEs. The `mode` controls what is shown. A bit pattern is used:

```
Bit 0: VC tracks
    1: Segments
    2: LTEs
    3: Hits
```

Thus mode 3 means show the VC tracks and the segments.

If you specify `chamber 0` all chambers are shown in a 2x2 display (only front view). Otherwise only a single chamber is shown.

You can use the `SET X|Y LIMIT` commands to show only part of the detector. However this only works if you are displaying a single chamber. The default limits are -100 to +100cm for the front view mode. The default limits for the side view:

```
FTD     x-axis      y-axis( in cm)
 1      0 to 140    -71.25 to 71.25
 2      0 to 182.5  -91.25 to 91.25
 3      0 to 217.5  -108.75 to 108.75
```

The colour of the frame can be set with `SET COLOUR FRAME` (only front view), that of the contents with `SET COLOUR SYMBOL`, that of the title and detectors with `SET COLOUR TITLE`. The size of the `FTDn` labels is controlled with the `SET TSIZE` command, that of the symbols with `SET SSIZE`. The font for the title is set with the `SET FONT TITLE nfont` command. This font is also used for the contents. The key colours and fonts are the same as those for `COMMENTS`. The line thickness can be set in the same way as the colours using the `SET THICKNESS` commands. The frame can be turned on or off using the `SET FRAME` command.

TRD

```
Syntax: SET PAR TRD mode chamber ancat
where:  mode     Viewing mode - the various parts can be combined
                  1 Wires hit - XY view
                  2 Clusters - XY view
                 10 VC tracks
                 20 FTD segments
        chamber  The chamber number
                 1,2,3,4 mean TRD1, TRD2, TRD3, TRD4
        ancat    1 means show the anodes
                 2 means show the cathodes
```

```
                3 means show both
Defaults: 1 0 3
```

Sets the parameters for the ZEUS transition radiation detectors. The default is to show the hit wires. At the moment showing the clusters gives the same picture, as a side view has not yet been implemented. The extrapolated VC tracks and the FTD segments can also be shown, assuming that you have a combined Ntuple. In order to show the VC tracks and FTD segments use the same bit pattern as for the FTD display multiplied by 10:

```
Bit 0: VC tracks
    1: Segments
    2: LTEs
```

Thus mode 31 means show the VC tracks and the segments overlayed on the TRD. For the VC tracks the extrapolated hit positions in TRD1 and 2 are taken as a linear extrapolation between those in FTDs 1 and 2, while for TRDs 3 and 4 they are between FTDs 2 and 3. For the segments the slope and position of the segments are used to extrapolate linearly to the relevant TRD. (26.08.97 This is not yet the case, but is being worked on!).

If you specify `chamber 0` all chambers are shown in a 2x2 display. Otherwise only a single chamber is shown.

You can use the `SET X|Y LIMIT` commands to show only part of the detector. However this only works if you are displaying a single chamber. The default limits are -100 to +100cm.

The colour of the frame can be set with `SET COLOUR FRAME`, that of the contents with `SET COLOUR SYMBOL`, that of the title and detectors with `SET COLOUR TITLE`. The size of the `TRDn` labels is controlled with the `SET TSIZE` command, that of the symbols with `SET SSIZE`. The font for the title is set with the `SET FONT TITLE nfont` command. This font is also used for the contents. The key colours and fonts are the same as those for `COMMENTS`. The line thickness can be set in the same way as the colours using the `SET THICKNESS` commands. The frame can be turned on or off using the `SET FRAME` command.

4.106.65 PATH

```
Syntax: SET PATH [+]dir1, dir2
        dir3, ...
where   dir  are the directory names to be include in the path.
```

Sets a list of directories that are used when trying to open existing files, e.g. macros. When trying to open a file Mn_Fit first tries the filename exactly as given with ~ and environment variable expansion. It then loops over the list of directories to see if the file exists in one of them. You must include the / at the end of the directory name on Unix machines. The default list of directories is:

```
$HOME/mnf/, $MN_FIT/help/        (Unix)
mn_fit_help:                     (VMS)
```

This means that you can execute the demonstration macros etc. by just giving the command, e.g. `exec symbol`, independent of the directory you are in. You can also collect useful macros in a single directory, e.g. `$HOME/mnf`.

You can add directories to the current path by prefixing the first directory name with a +.

In the case of files such as COMIS functions which may or may not exist already, a loop is first made over the `PATH`. If no existing file is found the working directory (`WDIRECTORY`) is added as a prefix.

A <CR> terminates the list of directories. If you set the path in a macro do not forget to include an empty line after the list.

Note that filename completion no longer works if the file is not in the current directory. If you give a filename of the form `./file` (Unix) then the path search will not be executed.

4.106.66 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.

4.106.67 PI

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

Turns on or off using pi as a symbol for the scale. This is useful if you make plots in radians and want to have an easy to understand scale.

4.106.68 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.

4.106.69 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.

4.106.70 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.

4.106.71 RECL

```
Syntax: SET RECL n
```

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

4.106.72 REDRAW

```
Syntax:  REDRAW
```

Alias for REDRAW so that you can REDRAW without exiting SET. See section 4.97 on page 141 (REDRAW) for more details.

4.106.73 ROOT_ID

```
Syntax: SET ROOT_ID n
```

Sets the default ROOT histogram identifier. This is used if the ROOT identifier string cannot be converted to a number and if the identifier is not give with the ROOT_FETCH command. Alias for SET IDR.

4.106.74 ROTATION

```
Syntax:   SET ROTATION angle
where:    angle  is the angle to rotate a plot by
```

Rotates a plot - not yet fully implemented.

4.106.75 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, 0, 0
          y axis -0.2,  0.0, 0.4, 0.0, 6, -1, 0, 0
          z axis -0.2,  0.0, 0.4, 0.0, 6, -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].

For the y-axis of a lego or surface plot the x and y offsets are flipped, as the scale should be vertically below the tick. The scales are also offset by the scale text size and the big tick size.

Use the command SET X|Y|Z MODE to change the numbering of the scale. To get a log scale for example use the commmand SET X|Y|Z MODE LOG.

4.106.76 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.

4.106.77 SHOW_ZERO

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

Controls showing zero points which also have zero errors.

4.106.78 SHELL

```
Syntax:   SET SHELL command
where:    command  is the shell command
Defaults: command = $SHELL
```

Sets the shell used by the SPAWN or SHELL commands. This option applies to Unix machines. Examples of command are /com/sh, /bin/sh, /bin/ksh, /bin/csh, /bin/tcsh. The shell given by the command will be invoked when SHELL or SPAWN is given.

4.106.79 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.

4.106.80 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].

4.106.81 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].

4.106.82 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-D 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').

4.106.83 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
```

Note that histograms always get converted to a series of points when fitting, so the default symbol for DISPLAY is always -32.

To get a picture of the available symbols issue the command exec $MN_FIT/help/symbol.mnf (Unix) or EXEC MN_FIT_HELP:SYMBOL.MNF (VMS). 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 4.106.37 on page 156 (SET HATCH) for details on some of the hatching available and 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
12        Square
13        Triangle
14        Inverted triangle
15        Diamond
16        Plus  (+)
17        Cross (x)
18        Asterix (*)
19        Octogon (used to be 11)
20-29     Show x error bars for histograms
30-39     Show y error bars
40-49     Show x and y error bars
60-69     Show x error bars with line at end (symbol size, SET SSIZE)
70-79     Show y error bars with line at end (symbol size, SET SSIZE)
80-89     Show x and y error bars with line at end (symbol size, SET SSIZE)
-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
10-19     As for histograms
20-49     As for 10-19
-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
10-19     Area of symbol is proportional to number of entries in
```

```
           the bin
 20-49     As for 10-19
 -n        Show symbol filled
```

If the minimum weight is negative and the maximum weight is positive for 2-D histograms and you use a symbol number greater than 10 or less than -10 you will get the symbol for positive weights and its inverse for negative weights. This mode is also used for displaying the result of a 2-D fit. This mode is only used if either the lower or upper plotting limit is 0.0. Use the `SET Z LIMIT` command to set the limits. Otherwise only those entries within the specified range are shown.

2-D histograms can also be plotted using the `LEGO` or `SURFACE` commands or preferably using the interface to the HIGZ IGTABL routines, `2DIM` or `IGTABLE`.

I use the HIGZ circle routine to draw circles. It is also possible to use large dots (symbol 10 or -10) and set the dot scale size using the `SET DSIZE` command. A scale factor of 10 or 20 is usually good. Note that you only see the big dots when you print the picture.

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

4.106.84 TEXT

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

Controls whether the header text with the fit results is shown on your screen device. Use the command `SET HEADER` if you want to change it for hardcopies also.

4.106.85 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 HIGZ version) To change the pen width for a particular plot, precede the command with `SET PLOT id [&idb]`.

If you omit the item, all line thicknesses will be changed. The following items are valid:

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

Note that you may not see the effect of the thickness on the screen. Postscript lines are usually very thin and a thickness factor of 3 or 4 looks a lot better, with 6 or so for the frame thickness.

4.106.86 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|OUT] 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
```

4.106.87 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`. Give a time of 0 if you do not want to have a reference time.

Use the `SET X|Y|Z MODE DATE|TIME` commnd to show the scale in date or time format.

4.106.88 TITLE

```
Syntax: SET TITLE USER|GLOBAL title
    or  SET TITLE DEFAULT|ON|OFF
    or  SET TITLE POSITION  [xoff yoff size angle option font]
    or  SET TITLE GPOSITION [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. A global title is also put at the top of the picture, but the individual plot titles are shown inside the plot. `SET TITLE POSITION` controls the position of title, which is relative to the top centre of the plot, and the font used for it. `SET TITLE GPOSITION` controls the position of the global and/or user 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]`.

Note that if the title size is set to default and you have more than 1 plot per page (`WINDOW` command), the title size for the individual histograms is scaled down. To turn off this option use the `SET AUTOSCALE` command.

4.106.89 TKTCL

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

Turns on/off the TK/TCL interface for Mn_Fit. At present this only involves sending a special string at the end of any output.

4.106.90 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.

4.106.91 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]`.

4.106.92 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. It is also useful if you want to put up a new plot every so often and use it in conjunction with the `WAIT` command.

4.106.93 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, scale and label 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. This can be turned off with the SET AUTOTRIM OFF command.

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.

If you change the windowing the next plot will be put in window 1,1 by default. Use the SET NEXT_WINDOW command to change this.

If you plot Ntuple projections in windows with different cuts, but with the same secondary identifier you will usually get a surprise when you make a HARDCOPY or REDRAW. This is because these commands repeat the commands used to make the plots, thus you will get the same hisotgram plotted several times. It is therefore better to give a different secondary identifer to each projection that has different cuts.

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
SET NEXT_WINDOW 1 2    !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
```

4.106.94 NO_WINDOW

```
Syntax: SET NO_WINDOW
```

Turns off any windowing. All tick, title, label and scale sizes that may have been rescaled will be set back to default.

4.106.95 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 defined by the x and y margins. 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. You can also use it in connection with the PLOT/NOCLEAR command to make inserts.

4.106.96 WORKING_DIR

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

Alias for WDIR. Sets the working directory name to dirname. If you then try to open a file without a directory name dirname will be prepended.

4.106.97 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. You can also use it in connection with the PLOT/NOCLEAR command to make inserts.

4.106.98 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].

4.107 SHELL

```
Syntax: SHELL [command]
```

Alias for SPAWN. Spawns a DCL (VMS) or Shell (Unix) command. You can specify the shell on Unix machines using the SET SHELL command. If no command is given a new process is created. Use the command Return (VMS) or exit (Unix) to return to Mn_Fit when you have finished in the subprocess. On VMS you can also ATTACH to the main process and then use ATTACH in Mn_Fit to re-enter the subprocess if you wish.

4.108 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>.

4.108.1 ALL

```
Syntax: SHOW ALL
```

Lists everything there is to show.

4.108.2 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 4.5 on page 54 (ALIAS) for more information on use of aliases.

4.108.3 CHAR

```
Syntax:  SHOW CHAR r1:[r2]|ALL
where:   r1  is a character array element number
         r2  is the upper limit of a range of characters to show
```

Lists the contents of one or more character array elements. If you give the command `SHOW CHAR ALL` all array elments will be listed (0-99). You can also list a range of elements e.g. `SHOW CHAR 11:20`. The `DEPOSIT` and `CALCULATE` commands can be used to set the contents of the array. For example, `DEP CHAR(4) = ABC` will set the contents of element 4 to `ABC`.

Use the command `SHOW CHARACTER` (abbreviated to no shorter than `SHOW CHARA` to show the comment and continuation characters.

4.108.4 COMMANDS

```
Syntax: SHOW COMMANDS
```

Lists the commands defined using the `DEFINE` command. Alias for `SHOW DEFINITION`.

4.108.5 COMMENT

```
Syntax: SHOW COMMENT
```

Lists the current comments associated with plots.

4.108.6 CONSTRAINT

```
Syntax: SHOW CONSTRAINT
```

Lists the current constraints on parameters that are being fit. See section 5.5 on page 193 (MINUIT CONSTRAIN) for more details on how to constrain parameters.

4.108.7 CUTS

```
Syntax: SHOW CUT
```

Lists the cuts set for a plot.

4.108.8 DEFINITION

`Syntax: SHOW DEFINITION`

Lists the commands defined using the `DEFINE` command. Alias for `SHOW COMMANDS`.

4.108.9 DIRECTORY

`Syntax: SHOW DIRECTORY.`

Alias for `LDIRECTORY` or `LS`.

List of the contents of the currently selected HBOOK directory in an HBOOK4 file or the contents of the currently selected ROOT directory in a ROOT file.

4.108.10 EXCLUSIONS

`Syntax: SHOW EXCLUSIONS`

Lists the parts of histograms excluded and included in the fit.

4.108.11 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.

4.108.12 FLAGS

`Syntax: SHOW FLAGS`

Lists the logical flags that control what is plotted.

4.108.13 FRAME

`Syntax:  SHOW FRAME`

Lists the plot limits and the parameters affecting the scale and frame.

4.108.14 INCLUSIONS

`Syntax: SHOW INCLUSIONS`

Lists the parts of histograms included and excluded in the fit.

4.108.15 KEYS

`Syntax: SHOW KEYS`

Lists the current keys associated with plots.

4.108.16 LABEL

`Syntax: SHOW LABEL`

Lists the plot labels and the parameters affecting them.

4.108.17 LIMIT

```
Syntax:  SHOW LIMIT
```

Lists the plot limits and the parameters affecting the scale and frame.

4.108.18 LOG

```
Syntax: SHOW LOG [n]
```

Lists the last `n` commands given. If `n` is not given the last 100 commands are shown.

4.108.19 MODE

```
Syntax:  SHOW MODE
```

Lists the plot limits and the parameters affecting the scale and frame.

4.108.20 ORDER

```
Syntax: SHOW ORDER
```

Lists the expected order of variables when executing `DAT_FETCH`.

4.108.21 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`).

4.108.22 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 4.28 on page 69 (DEPOSIT) for more details). A more general command that enables you to look at the values of registers, parameters etc. is the `EXAMINE` command.

4.108.23 SCALE

```
Syntax:  SHOW SCALE
```

Lists the plot limits and the parameters affecting the scale and frame.

4.108.24 SEGMENTS

```
Syntax:  SHOW SEGMENTS
```

Lists the current segments in use. Only applies to the GKS version and is mainly for expert use.

4.108.25 SIZES

```
Syntax:  SHOW SIZES
```

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

4.108.26 TICK

```
Syntax:  SHOW TICK
```

Lists the parameters affecting the tick marks.

4.108.27 UNITS

```
Syntax:  SHOW UNITS
```

Lists FORTRAN units and who has grabbed them (mainly for experts). The name of the subroutine that booked the unit is shown.

4.108.28 VARIABLE

```
Syntax: SHOW VARIABLE name|ALL
where   name  is a user variable name
```

Lists the value of a user variable or all user variables.

4.109 SMOOTH

```
Syntax:   SMOOTH[/option] id [&idb] parameter [&]idb2 [mode]
 where:   option     is the type of smoothing:
                     /HBOOK
                     /NAGLIB
                     /IMSL
                     /TOPDRAW
          parameter  is a steering parameter
          [&]idb2    is the secondary id of the smoothed histogram
          mode       plotting mode - NAGLIB and IMSL versions only
Defaults: idb2 = idb + 1, mode = 1,
```

Smooths a histogram. There are a number of possibilities available depending on whether you have the NAGLIB and/or IMSL libraries. The default smoothing type is NAGLIB (if available), IMSL (if available), Topdraw.

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).

Mode 0 is only available for IMSL and NAGLIB smoothing. You can use the `HISTOGRAM ERROR` command to change or remove the errors on the points.

If you just give the command `SMOOTH[/option] id` you enter into an interactive mode and can iterate on the smoothing by varying the smoothing parameter. If you give the command this way in a macro then leave a blank line (or the command `YES`) at the end of your fitting.

4.109.1 /HBOOK

Uses the HBOOK routine HSMOOF. This routine returns the χ^2 of the smoothing, but does not give you any control on how much smoothing to do and does not take into account the errors on the bins.

4.109.2 /IMSL

Uses the IMSL library routine ICSSCU. See the IMSL manual for complete details. The smoothing parameter is the χ^2 for the original histogram 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 errors on each point are taken into account. Points with 0 error are not included in the smoothing. Use the `HIST ERROR id 0` command to remove the errors if you do not want to use them.

4.109.3 /NAGLIB

Uses the NAGLIB library routine E02BEF. See the NAGLIB manual for complete details. The smoothing parameter is the χ^2 for the original histogram 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 errors on each point are taken into account. Points with 0 error are not included in the smoothing. Use the `HIST ERROR id 0` command to remove the errors if you do not want to use them.

4.109.4 /TOPDRAW

Uses the routine SMCTRL coded by Roger Chaffee from an algorithm by J.W. Tukey and A. Tubillo. You can specify the level of smoothing, which must lie between 0 and 5. 0 does not seem to work very well, 1 produces the least amount of smoothing and 5 produces the most.

The points are assumed to be equally spaced and monotonically increasing for this routine, but this is not checked by Mn_Fit.

4.110 SPAWN

```
Syntax: SPAWN [command]
```

Alias for `SHELL`. Spawns a DCL (VMS) or Shell (Unix) command. You can specify the shell on Unix machines using the `SET SHELL` command. If no command is given a new process is created. Use the command `Return` (VMS) or `exit` (Unix) to return to Mn_Fit when you have finished in the subprocess. On VMS you can also `ATTACH` to the main process and then use `ATTACH` in Mn_Fit to re-enter the subprocess if you wish.

4.111 SPLINE

```
 Syntax:   SPLINE[/option] id [&idb]
           nknot
           knot1,knot2,...
           [&]idb2
           mode
  where:   option     is the type of spliine fitting
                      /HBOOK
                      /NAGLIB
                      /IMSL
           nknot      is the number of knots
           knot1      is the 1st knot position - not HBOOK verison
           knot2      is the 2nd knot position etc.
           [&]idb2    is the secondary id of the smoothed histogram
           mode       plotting mode - NAGLIB and IMSL versions only
 Defaults: idb2 = idb + 1, mode = 0,
```

Spline fits a histogram. There are a number of possibilities available depending on whether you have the Naglib and/or IMSL libraries. The default spline fitting type is NAGLIB (if available), IMSL (if available), HBOOK.

The minimum number and maximum number of knots and their minimum and maximum values depend on the type of fitting - see the relevant section for details.

The result of the fit 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). Mode 0 is only available for IMSL and NAGLIB smoothing

If you just give the command `SPLINE[/option] id` you enter into an interactive mode and can iterate on your fit by varying the numbers of knots and/or their positions. If you give the command this way in a macro then leave a blank line (or the command `YES`) at the end of your fitting.

4.111.1 /HBOOK

Uses the HBOOK routine HBSPLI1. You specify the number of knots, and the routine returns the χ^2 of the spline, but does not give you any control on the knot positions. The errors on the bins are not taken into account..

4.111.2 /IMSL

Uses 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.

4.111.3 /NAGLIB

Uses the NAGLIB library routine E02BAF. See the NAGLIB manual for complete details. The number of knots must be less than or equal to the number of points - 4, and the knots must be monotonically increasing. The first knot must be greater than first point in the histogram and

the last knot must be less than the last point in the histogram. The routine adds 4 knots at the minimum and maximum points.

The errors on each point are taken into account. Points with 0 error are not included in the fit. Use the `HIST ERROR id 0` command to remove the errors if you do not want to use them.

4.112 SQUEEZE

```
Syntax: SQUEEZE
```

Gets back unused space in the array where the histograms are stored. You `DELETE 0` to remove all histograms. This is much faster than `SQUEEZE`.

4.113 STAT

```
Syntax:   STAT
```

Calls RZSTAT for the current HBOOK input file, to give statistics on space usage.

4.114 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 `SET DIR` 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`.

If you want to keep the secondary identifiers you can use the `MN_STORE` command, together with `MN_FETCH` to read them in again. Note that this format is not machine independent.

As of version 4.01 of Mn_Fit the histograms will be stored in RZ exchange format. This means that they cannot be read in properly by older versions of Mn_Fit or PAW. The big advantage of exchange format is that you can just binary ftp files from one machine to another.

4.114.1 /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.

4.114.2 /UPDATE

```
Syntax: STORE/UPDATE ...
```

Stores the histograms in an already existing HBOOK direct access file.

4.115 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.

4.116 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.

4.117 SURFACE

```
Syntax:  SURFACE id [&idb] theta phi          (default theta=30, phi=30)
where    qual  can be /C1|/C2|/CONT|/SHADE|/POL|/CYL|/SPH|/PSD|/NFB|/NBB
```

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. If you give the command without any qualifier the angles must lie between 0 and 90. If you give a qualifier the IGTABLE interface will be used. The current surface plotting code will soon be replaced by that in the `2DIM SURFACE` command, which has no restrictions on the rotation angles and has a number of diffferent colour options. See section 4.3 on page 47 (2DIM) for more details.

4.118 TITLE

```
Syntax: TITLE id [&idb] new_title
```

Changes the title on a histogram.

4.119 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 4.5 on page 54 (ALIAS) for a complete description of the alias mechanism.

4.120 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`.

4.121 WAIT

```
Syntax: WAIT sec
 where  sec  is the time to wait in seconds
```

Tells Mn_Fit to wait for a time.

4.122 WDIRECTORY

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

Sets the working directory name to `dirname`. If you then try to open a file without a directory name `dirname` will be prepended.

4.123 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 4.106.93 on page 178 (SET WINDOW) for more details.

4.124 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.

4.124.1 DATA

```
Syntax: WRITE DATA filename id [&idb]
```

Alias for DAT_STORE. See section 4.22 on page 65 (DAT_STORE) for more details.

4.124.2 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.

4.125 XSCALE

```
Syntax:   XSCALE id1 [:id1n] [&idb1] id2 [:id2n] [&idb2]
          [scale]
 where:   id1    is the input histogram identifiers
          idb1   is the (optional) input secondary identifiers
          id2    is the output histogram identifier
          idb2   is the (optional) output secondary identifier
          scale  is the amount to scale the x-axis by
```

Scales the x-axis of a plot by scale. This is useful if you want to say change a plot scale from seconds to hours. Note that XSCALE makes a new histogram. If you just want to scale the values in a picture you can use the SET BIN command instead for 1-D histograms.

4.126 XSHIFT

```
Syntax:   XSHIFT id1 [:id1n] [&idb1] id2 [:id2n] [&idb2]
          [shift]
 where:   id1    is the input histogram identifiers
          idb1   is the (optional) input secondary identifiers
          id2    is the output histogram identifier
          idb2   is the (optional) output secondary identifier
          shift  is the amount to shift the x-axis by
```

Shifts the x-axis of a plot by shift. This is useful if you want to overlay plots with error bars that have the same x values for all points. Note that XSHIFT makes a new histogram. If you just want to shift the values in a picture you can use the SET BIN command instead for 1-D histograms.

4.127 YSCALE

```
Syntax:   YSCALE id1 [:id1n] [&idb1] id2 [:id2n] [&idb2]
          [scale]
 where:   id1    is the input histogram identifiers
          idb1   is the (optional) input secondary identifiers
```

```
          id2    is the output histogram identifier
          idb2   is the (optional) output secondary identifier
          scale  is the amount to scale the y-axis by
```

Scales the y-axis of a 2-D plot, or the bin contents or point values of a plot by `scale`. For 1-D plots this is identical to the `SCALE` command.

4.128 YSHIFT

```
Syntax:   YSHIFT id1 [:id1n] [&idb1] id2 [:id2n] [&idb2]
          [shift]
 where:   id1    is the input histogram identifiers
          idb1   is the (optional) input secondary identifiers
          id2    is the output histogram identifier
          idb2   is the (optional) output secondary identifier
          shift  is the amount to shift the y-axis by
```

Shifts the y-axis of a 2-D plot, or the bin contents or point values of a 1-D plot by `shift`. This is useful if you need to modify the bin contents of a complete plot by the same amount.

4.129 ZDIRECTORY

```
Syntax: ZDIRECTORY
```

List of the contents of the currently selected HBOOK directory in an HBOOK4 file using the routine RZLDIR.

4.130 ZSCALE

```
Syntax:   ZSCALE id1 [:id1n] [&idb1] id2 [:id2n] [&idb2]
          [scale]
 where:   id1    is the input histogram identifiers
          idb1   is the (optional) input secondary identifiers
          id2    is the output histogram identifier
          idb2   is the (optional) output secondary identifier
          scale  is the amount to scale the z-axis by
```

Scales the z-axis of a 3-D plot, or the bin contents of a 2-D plot by `scale`. For 2-D plots this is identical to the `SCALE` command.

4.131 ZSHIFT

```
Syntax:   ZSHIFT id1 [:id1n] [&idb1] id2 [:id2n] [&idb2]
          [shift]
 where:   id1    is the input histogram identifiers
          idb1   is the (optional) input secondary identifiers
          id2    is the output histogram identifier
          idb2   is the (optional) output secondary identifier
          shift  is the amount to shift the z-axis by
```

Shifts the z-axis of a 3-D plot, or the bin contents of a 2-D plot by `shift`. This is useful if you need to modify the bin contents of a complete plot by the same amount.

Chapter 5

Reference Manual for MINUIT

5.1 MINUIT

```
Syntax: MINUIT minuit_command
```

Direct interface to MINUIT commands. Only works with the CERN version of MINUIT.

5.2 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.

5.3 CALLS

Command has been renamed to MAX_CALLS.

5.4 CHI_PLOT

```
Syntax: CHI_PLOT name|number xlo xhi ylo yhi
 where: name    is the name of the variable you want to plot
        number  is the parameter number
        xlo     is the minimum value
        xhi     is the maximum value
        ylo     is the minimum value of the chi**2
        yhi     is the maximum value of the chi**2
```

Draws the χ^2 as a function of one of the fit parameters. If you are doing a likelihood fit the proper χ^2 is calculated using the square root of the value of the function as the error. NO check is done on exceeding limits on variable x (if any), so be careful! You can give either

the parameter number or its name. See section 5.15 on page 196 (MINUIT FCN_PLOT) and section 5.39 on page 200 (MINUIT PROB_PLOT) for other ways to plot the fit probability as a function of the value of a variable. See section 5.30 on page 199 (MINUIT MNCONTOUR) for contour plots.

5.5 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 1.8 on page 14 (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.

You can specify other parameters that are being fit using the syntax `Pn` (where `n` is the MINUIT parameter number), or `Pn(m)` (where `n` is the function number and `m` is the parameter number in the function).

WARNING: 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 minimization (i.e. when you give the `MINIMIZE` command) will be used.

This command is only available in the CERN version of MINUIT.

5.5.1 Examples

1. Constrain the mean and sigma of 1 Gaussian with respect to a second:

```
fun add gaus sig             !Add 2 Gaussians
1000
1
0.5
fun add gaus sig
1000
2
0.5
fit 1
0
!Constrain the mean of the second Gaussian to be 1 > the first
!Use the MINUIT parameter number
constrain 5 p2+1
!Constrain the sigma of the second Gaussian to be 2x that of the first
!Use the Mn_Fit parameter number
constrain 6 2*p1(3)
minimize
show constrain
unconstr 5                  !Remove the constraint on parameter 5
minimize
```

5.6 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. Use the `MNCONTOUR` command to get a graphical contour.

Makes a 2-D 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.

5.7 COVARIANCE

```
Syntax: COVARIANCE
```

Writes the covariance matrix to the current output unit.

5.8 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-D 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-D plots, so if, for example, you want to change the limits, you must use the syntax `SET PLOT id&982 X LIMIT` etc.

5.9 DUMP

```
Syntax: DUMP
```

Gives the data and fit values for each point and their contribution to the χ^2 or likelihood.

5.10 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.

5.11 ERROR_DEF

```
Syntax: ERROR_DEF up
 where: up   is the change in the chi**2/likelihood
```

Sets the change in the χ^2 or likelihood that is used to define the error on a parameter.

5.12 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-D 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`.

5.13 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.

5.14 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 +/- 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. You can give either the parameter number or its name.

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!

5.15 FCN_PLOT

```
Syntax: FCN_PLOT name|number xlo xhi ylo yhi
 where: name    is the name of the variable you want to plot
        number  is the parameter number
        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
```

Plots the χ^2 or likelihood as a function of one of the fit parameters. NO check is done on exceeding limits on variable `x` (if any), so be careful! You can give either the parameter number or its name. See section 5.39 on page 200 (MINUIT PROB_PLOT) and section 5.4 on page 192 (MINUIT CHI_PLOT) for other ways to plot the fit probability as a function of the value of a variable. See section 5.30 on page 199 (MINUIT MNCONTOUR) for contour plots.

5.16 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.

5.17 FIX

```
Syntax: FIX var1 [var2 ...]
```

Fixes the list of variables you give. You can specify up to 7.

5.18 FLOAT

```
Syntax: FLOAT var1 [var2 ...]
```

Restores up to 7 parameters (reverse of `FIX`).

5.19 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.

5.20 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.

If the `PRINTOUT` level is 0 or greater the covariance matrix is also shown.

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 INCLUDE

```
Syntax: INCLUDE xlo xhi
    or  INCLUDE axis xlo xhi
 where: axis  is the axis name (defaults are x and y)
        xlo   is the lower limit of the inclusion
        xhi   is the upper limit of the inclusion
```

Includes a region of the histogram to be fit. If you are fitting a 2-D histogram, you must specify which axis the inclusion 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 inclusion applies to. If you just give the command `INCLUDE`, you can give several regions to include, then hit <`CR`> to exit `INCLUDE`.

5.23 INFO

```
Syntax: INFO
```

Provides full info of current fit status. The value of FCN is the current value of the χ^2 or likelihood.

5.24 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!

5.25 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.

If you make a plot of your fit parameter with MINOS errors and then you want to store the histogram, you should not use the STORE command, as HBOOK does not know about asymmetric errors in its normal histograms. Instead use the DAT_STORE or MN_STORE commands that write the plots out in an ascii file or in Mn_Fit format.

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 NO_EXCLUDE

```
Syntax: NO_EXCLUDE
```

Removes any exclusions.

5.33 NO_GRADIENT

```
Syntax: NO_GRADIENT
```

Turns off GRADIENT.

5.34 NO_INCLUDE

```
Syntax: NO_INCLUDE
```

Removes any inclusions.

5.35 NO_ITERATIONS

```
Syntax: NO_ITERATIONS
```

Turns off showing of fit iterations.

5.36 PAGE

```
Syntax: PAGE
```

Starts a new page on unit 6 [ISYSWR].

5.37 PRECISION

```
Syntax: PRECISION edm          (default = 0.1)
 where: edm  is the convergence criterium for the minimizers
```

Sets the convergence criterium for SIMPLEX and MIGRAD.

5.38 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!

5.39 PROB_PLOT

```
Syntax: PROB_PLOT name|number xlo xhi ylo yhi
 where: name    is the name of the variable you want to plot
        number  is the parameter number
        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 one of the fit parameters. 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! You can give either the parameter number or its name. See section 5.15 on page 196 (MINUIT FCN_PLOT) and section 5.4 on page 192 (MINUIT CHI_PLOT) for other ways to plot the fit probability as a function of the value of a variable. See section 5.30 on page 199 (MINUIT MNCONTOUR) for contour plots.

5.40 PUNCH

```
Syntax:  PUNCH
```

Saves the current MINUIT parameter values in a file. Inside Mn_Fit the `FUNCTION STORE` commmand is much more useful (see section 4.48.12 on page 108 (FUNCTION STORE) for more details).

5.41 RELEASE

```
Syntax: RELEASE var1 [var2 ...]
```

Standard `MINUIT` command for `FLOAT`. Restores up to 7 parameters (reverse of `FIX`).

5.42 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 5.18 on page 196 (MINUIT FLOAT).

5.43 SAVE

```
Syntax:  SAVE
```

Saves the current MINUIT parameter values in a file. Inside Mn_Fit the `FUNCTION STORE` commmand is much more useful (see section 4.48.12 on page 108 (FUNCTION STORE) for more details).

5.44 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.

5.45 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.

5.46 SIMPLEX

```
Syntax: SIMPLEX
```

Invokes the fairly crude SIMPLEX minimizer.

5.47 STANDARD

```
Syntax: STANDARD
```

Calls routine `STAND` (without argument list) whose default version sits in the library as a simple `RETURN+END`. To get your own, include it in the LINK command.

5.48 STOP

```
Syntax: STOP
```

Executes a `STOP` right now.

5.49 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.

5.50 UNCONSTRAIN

```
Syntax:   UNCONSTRAIN npar
where:    npar  is the MINUIT parameter number
```

Removes the constraint on a parameter. See section 5.5 on page 193 (MINUIT CONSTRAIN) for details on the use of constraints.

This command is only available in the CERN version of MINUIT.

5.51 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. Dave Brown has tested the new versions that I sent him, fixed a number of bugs and implemented some new features. Jon Lewis has implemented new commands and is always a source of new ideas. He has also read through an older version of the manual several times and made a number of suggestions. Bijan Nemati used to support Mn_Fit at Cornell and was always ready to try out new versions and try to get them to work! Dan Bliss then took over that responsibility. The current Mn_Fit tsar for CLEO is Horst Severini. Roy Wang suggested the list of commands to help newcomers get started.

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. I also use HIGZ for the graphics interface, to minimize the numbeers of changes I have to make when I want to implement a new interface.

The TYPSCN package, originally written by D.G. Cassel and T.J. Killian, later updated by Chris Bebek, is straightforward to use and relieves the programmer of many of the usual chores associated with deciding which command has been given, or interpreting numbers.

Most of the code for calculating expressions and also the use of registers, parameters etc. came from Paul Avery's MULFIT package.

Jon Lewis has implemented the DI3000 interface using HIGZ/DI3000.

Many other Mn_Fit users have sent in their comments and bug reports. The list of good ideas for new features to implement never seems to get shorter!

Appendix B

Examples of Symbols, Fonts, Hatching and Keys

The complete set of Mn_Fit symbols is shown in Fig. B.1 and the standard Mn_Fit registers are given in Table B.1.

The same fonts are available in Mn_Fit as in PAW. For some examples see Fig. B.4. 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. -1013 to get font -13 with precision 1. The fonts available depend on the version of the graphics package used. In Fig. B.4 HIGZ Postscript fonts are shown. In Fig. B.5 those for GKSGRAL and GKSGRAL Postscript are shown.

The complete character set for the device independent HIGZ font (font 0) is shown in Fig. B.2, while that for HIGZ Postscript font -13 is shown in Fig. B.3. Note that you only see the Zapfdingbats fonts when you look at or print the Postscript file. Use the `HARDCOPY` or `CAPTURE` command to make such a file.

HIGZ Postscript fonts can be obtained using precision 0,1, or 2. They are identical when you make a hardcopy. The differences are how they appear on the screen. If you use Postscript fonts and precision 2 you get the hardware default font on the screen. If you use precision 1 then the standard IGTEXT font (font 2000) is used on the screen and the Postscript font is used when printing. If you use precision 0, you get the Postscript fonts on the screen, but not superscripts, subscripts, Greek characters etc.

The default font everywhere is now (Version 4.06) -1004 i.e. IGTEXT on the screen and Helvetica Postscript font when printing.

Note that there is an offset between the HIGZ/Postscript font numbers and those in GKSGRAL. Times-Roman is -1 in GKSGRAL and -13 in HIGZ/Postscript, Times-Italic is -2 and -1 respectively.

Device independent hatchings are available in all HIGZ interfaces and these are shown in Fig. B.6. Each graphics package usually has its own hatchings and patterns (DECGKS) and these are often different for screen and hardcopy devices. Some examples for GKSGRAL are shown in Fig. B.7. Consult the relevant graphics package documentation for more details.

Hatch -3 is often used, because it gives a nice shaded overlay. However note that this is really a pattern (because you cannot see through it) and if you use it in an overlay it obscures the ticks. The solution is to do a `plot/noclear` on the 1st id again to get the scale redrawn. The solid fill patterns (100) and examples of hatch -3 are shown in Fig. B.8.

The `KEY` command can be used to provide a legend of the different symbols and shading used in a plot. Some examples are given in Fig. B.9. See Section 4.71 on page 123 for more details.

A picture showing the standard Mn_Fit sizes and the appropriate commands is shown in Fig. B.10.

A complete list of the Postscript characters that are available is shown in Fig. B.11. The first column shows the character code. The second shows what you get if you give the character code using the form `\nnn`. The other columns show what you get in Greek (`[\nnn]`), Special (`"\nnn#`) and Zapfdingbats (`~\nnn#`) modes respectively. `nnn` indicates the character code given in the first column.

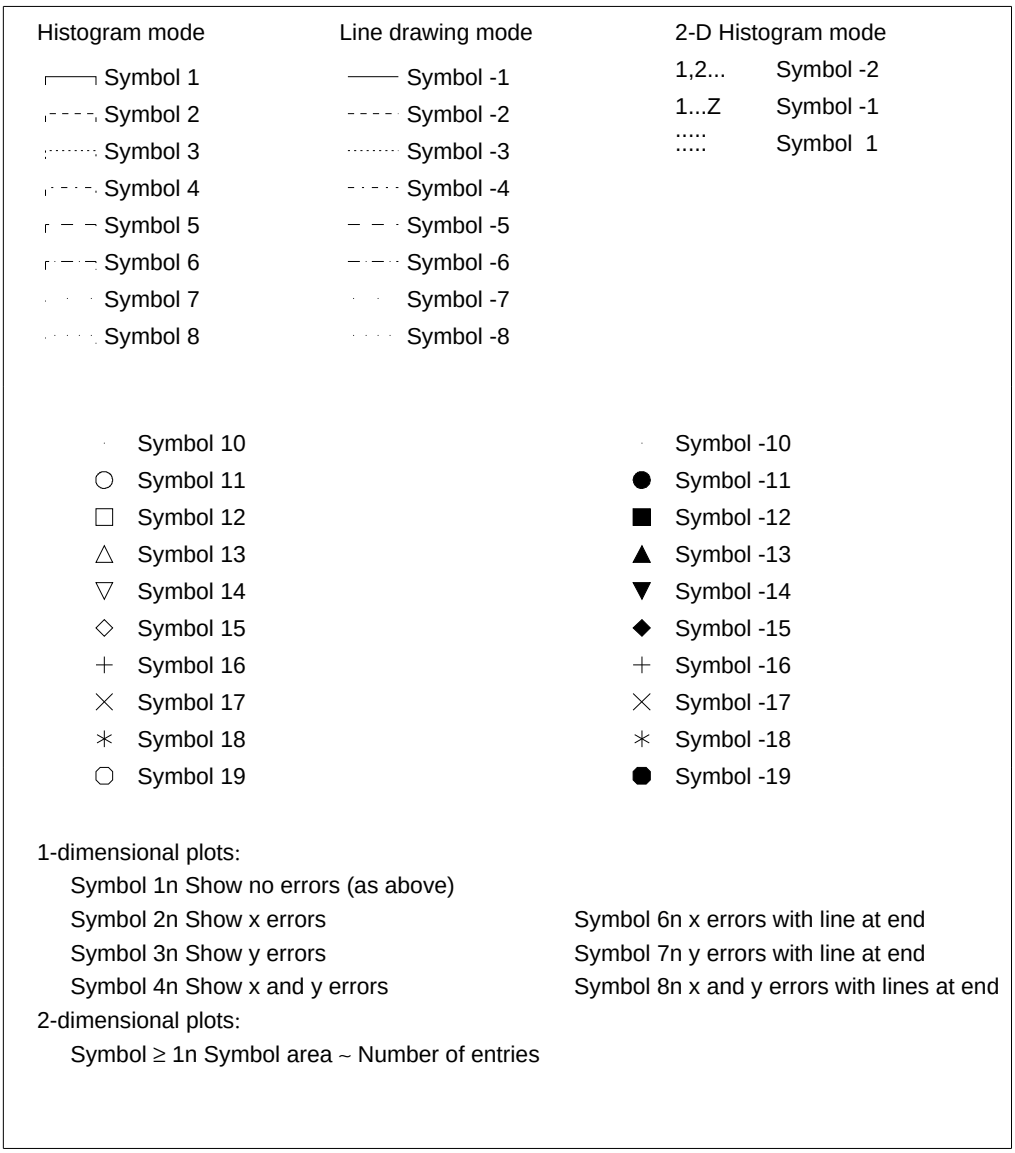


Figure B.1: Mn_Fit Symbols

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
113	The fit status (3 means converged properly)
114	The EDM (estimated distance to minimum) of the fit
Registers 121 to 200 and 231 to 257 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)
128	The creation date of the histogram (yymmdd)
129	The creation time of the histogram (hhmm)
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 mean value for the x-axis
135	The sigma for the x-axis
136	The number of bins on y-axis (0 for Ntuples and points)
137	The lower limit of the y-axis
138	The upper limit of the y-axis
139	The mean value for the y-axis
140	The sigma for the y-axis
	etc. up to 14th dimension of an Ntuple.
231	Underflows x-axis
232	Contents x-axis
233	Overflows x-axis
For 2 dimensional histograms 9 registers are filled (contents in register 235), while for 3-dimensional histograms 27 registers are filled (contents in register 244).	
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 They are in plot co-ordinates	
205	x minimum
206	x maximum
207	y minimum
208	y maximum
209	z minimum
210	z maximum
Registers > 300 contain extra variable names that you have defined. Up to 200 variables are allowed.	

Table B.1: The list of standard Mn_Fit registers. See Section 1.7 for more details on what you can give when Mn_Fit asks for a number.

Font 2000, i.e. IGTEXT

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	o	≧	≦
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	⸮	⸮
.	.	.	.	◇	□
,	,	,	,	⸮	⸮
+	+	+	+	+	+
−	-	−	-	¯	-
*	*	*	*	∗	∗
/	/	/	/	/	/
=	=	=	=	‾	‾
(	(	(	(	(	(
)	)	)	)	)	)

[	Switch to Greek	]	Switch back from Greek
"	Switch to special	#	Switch back from special
?	Switch to subscript	!	Switch back from subscript
^	Switch to superscript	!	Switch back from superscript
&	Backspace one character	@	Print an escape character

Figure B.2: HIGZ Portable Software Characters and the escape characters to switch modes.

Font -1004, i.e. Postscript - Helvetica

Upper Roman	Lower Roman	Upper Greek	Lower Greek	Upper Special	Lower Special	Upper Zapf	Lower Zapf
A	a	A	α	±	≈	✡	❁
B	b	B	β	|	≅	✢	✲
C	c	H	η	∃	⊥	✣	✵
D	d	Δ	δ	∀	∂	✤	✺
E	e	E	ε	!	ƒ	✥	✹
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	Y	υ	♦	⇒	❁	◆
V	v	X	χ	♣	⇓	✶	❖
W	w	Ω	ω	≤	&	✷	◗
X	x	Ξ	ξ	×	ϖ	✸	|
Y	y	Ψ	ψ	%	~	✹	❙
Z	z	Z	ζ	∞	ℵ	✺	❚
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	↔	↔	✙	✙
.	.	.	.	.	.	✚	✚
,	,	,	,	,	,	✛	✛
+	+	+	+	+	+	✜	✜
-	-	−	−	∠	∠	✝	✝
*	*	*	*	〈	〈	✞	✞
/	/	/	/	÷	÷	✟	✟
=	=	=	=	≠	≠	†	†
(	(	(	(	≡	≡	✠	✠
)	)	)	)	″	″	✡	✡

[	Switch to Greek	]	Switch back from Greek
"	Switch to special	#	Switch back from special
?	Switch to subscript	!	Switch back from subscript
^	Switch to superscript	!	Switch back from superscript
~	Switch to Zapf Dingbats	#	Switch back from Zapf Dingbats
&	Backspace one character	@	Print an escape character

Figure B.3: Postscript Version of HIGZ Portable Software Characters and the escape characters to switch modes.

HIGZ Postscript Fonts

Font/Prec	
- 1/0	*abcdefghijklmno ABCDEFGHIJKLMNO 123*
- 2/0	**abcdefghijklmno ABCDEFGHIJKLMNO 123**
- 3/0	***abcdefghijklmno ABCDEFGHIJKLMNO 123***
- 4/0	abcdefghijklmno ABCDEFGHIJKLMNO 123
- 5/0	*abcdefghijklmno ABCDEFGHIJKLMNO 123*
- 6/0	**abcdefghijklmno ABCDEFGHIJKLMNO 123**
- 7/0	***abcdefghijklmno ABCDEFGHIJKLMNO 123***
- 8/0	abcdefghijklmno ABCDEFGHIJKLMNO 123
- 9/0	*abcdefghijklmno ABCDEFGHIJKLMNO 123*
-10/0	**abcdefghijklmno ABCDEFGHIJKLMNO 123**
-11/0	***abcdefghijklmno ABCDEFGHIJKLMNO 123***
-12/0	αβηδεφγχικλμνο ΑΒΗΔΕΦΓΧΙΙΚΛΜΝΟ 123
-13/0	abcdefghijklmno ABCDEFGHIJKLMNO 123
-14/0	❁✲❃❄❅✳✺✼✻●❍■❑ ✡✣⌘✠✤✥✧★☆✪✯✭✮✬ 🖉🖋✓
-15/0	*abcdefghijklmno ABCDEFGHIJKLMNO 123*
-16/0	abcdefghijklmno ABCDEFGHIJKLMNO 123
-17/0	*abcdefghijklmno ABCDEFGHIJKLMNO 123*
-18/0	abcdefghijklmno ABCDEFGHIJKLMNO 123
-19/0	*abcdefghijklmno ABCDEFGHIJKLMNO 123*
-20/0	abcdefghijklmno ABCDEFGHIJKLMNO 123
-21/0	*abcdefghijklmno ABCDEFGHIJKLMNO 123*
-22/0	αβχδεφγηιφκλμνο ΑΒΧΔΕΦΓΗΙϑΚΛΜΝΟ 123
-23/0	abcdefghijklmno ABCDEFGHIJKLMNO 123
-24/0	❁❂❃❄❅✳✺✼✻✸○❍□❑ ✡✣✧✠✤✥✧✦☆✯✰✭✯✬ ✓

Figure B.4: Examples of HIGZ Postscript Fonts

GKSGRAL Software Fonts				Postscript	
Font/Prec		Font/Prec		Font/Prec	
− 1/2	abcABC 123	−201/2	abcABC 123	− 1/0	abcABC 123
− 2/2	abcABC 123	−202/2	abcABC 123	− 2/0	*abcABC 123*
− 3/2	abcABC 123	−203/2	abcABC 123	− 3/0	**abcABC 123**
− 4/2	abcABC 123	−204/2	abcABC 123	− 4/0	***abcABC 123***
− 5/2	*abcABC 123*	−205/2	*abcABC 123*	− 5/0	abcABC 123
− 6/2	***abcABC 123***	−206/2	***abcABC 123***	− 6/0	*abcABC 123*
− 7/2	*abcABC 123*	−207/2	*abcABC 123*	− 7/0	**abcABC 123**
− 8/2	*abcABC 123*	−208/2	*abcABC 123*	− 8/0	***abcABC 123***
− 9/2	abcABC 123	−209/2	abcABC 123	− 9/0	abcABC 123
−10/2	abcABC 123	−210/2	abcABC 123	−10/0	*abcABC 123*
−11/2	abcABC 123	−211/2	abcABC 123	−11/0	**abcABC 123**
−13/2	αβχABX 123	−213/2	αβχABX 123	−12/0	***abcABC 123***
−101/2	*abcABC 123*	−301/2	*abcABC 123*	−13/0	αβχABX 123
−102/2	***abcABC 123***	−302/2	***abcABC 123***		
−103/2	*abcABC 123*	−303/2	*abcABC 123*		
−104/2	***abcABC 123***	−304/2	***abcABC 123***		
−105/2	*abcABC 123*	−305/2	*abcABC 123*		
−106/2	***abcABC 123***	−306/2	***abcABC 123***		
−107/2	*abcABC 123*	−307/2	*abcABC 123*		
−108/2	*abcABC 123*	−308/2	*abcABC 123*		
−109/2	***abcABC 123***	−309/2	***abcABC 123***		
−110/2	***abcABC 123***	−310/2	***abcABC 123***		
−111/2	***abcABC 123***	−311/2	***abcABC 123***		
−113/2	*αβχABX 123*	−313/2	*αβχABX 123*		

Figure B.5: Examples of GKSGRAL Fonts

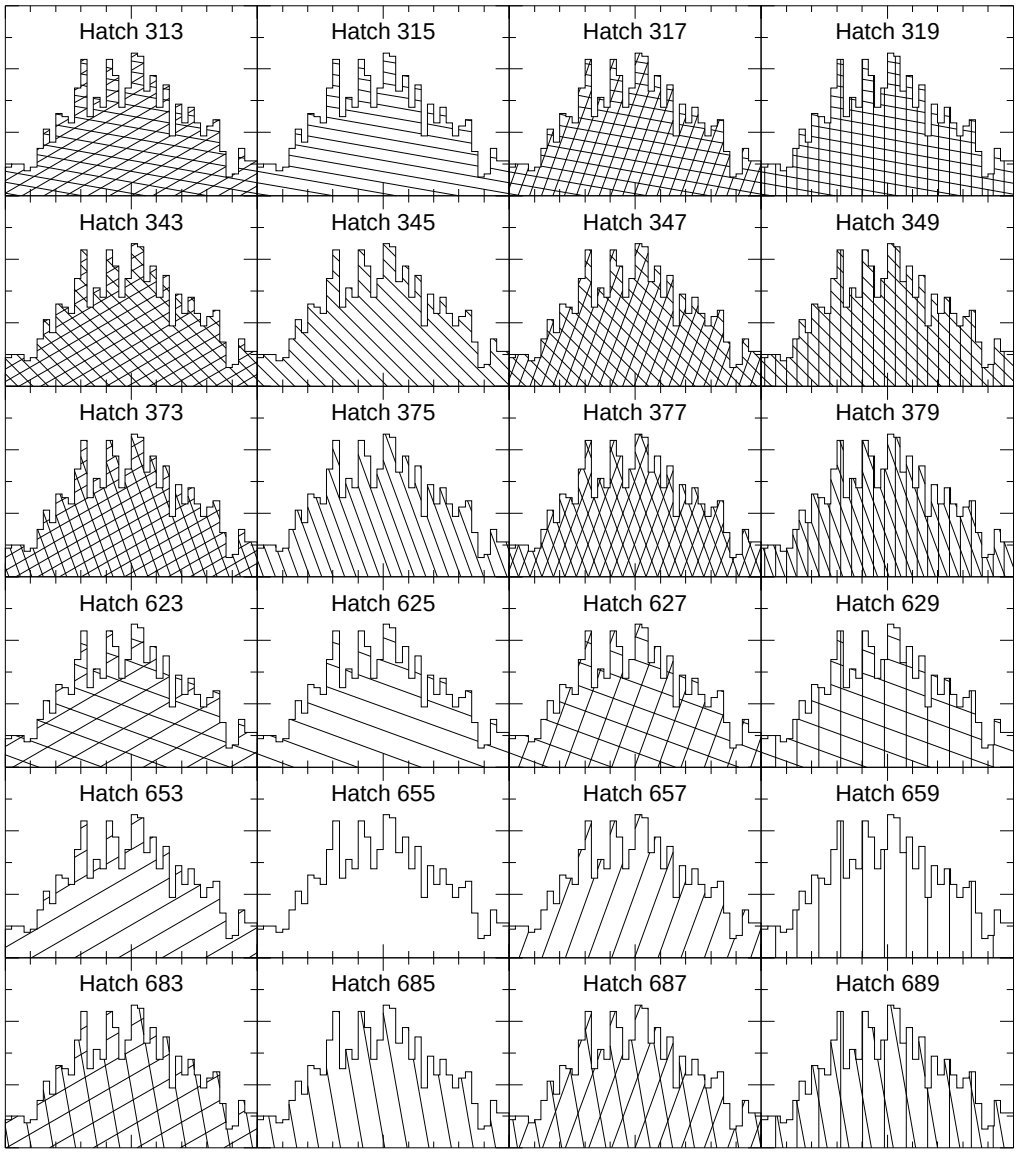


Figure B.6: Examples of HIGZ Hatching

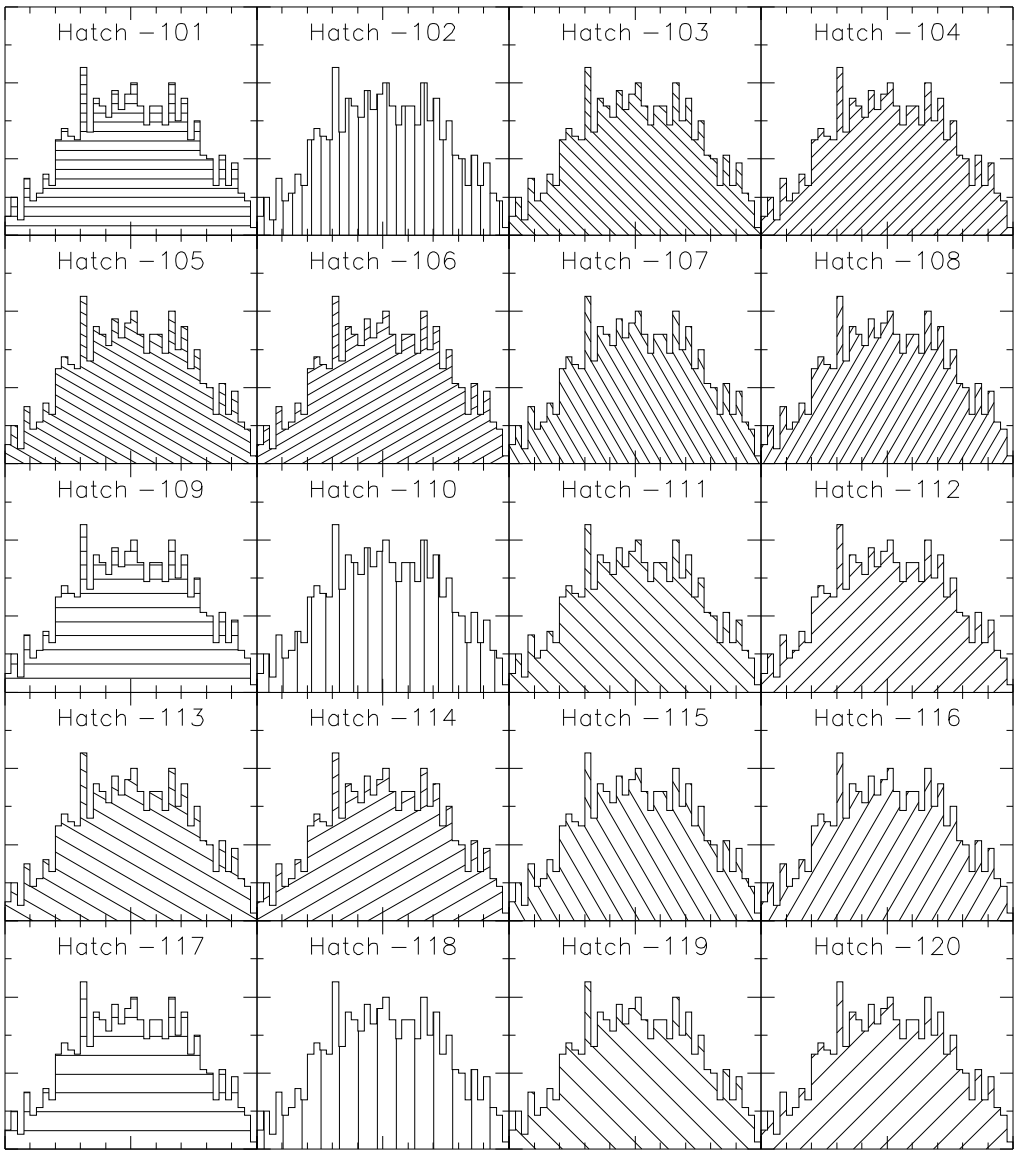


Figure B.7: Examples of GKSGRAL Hatching

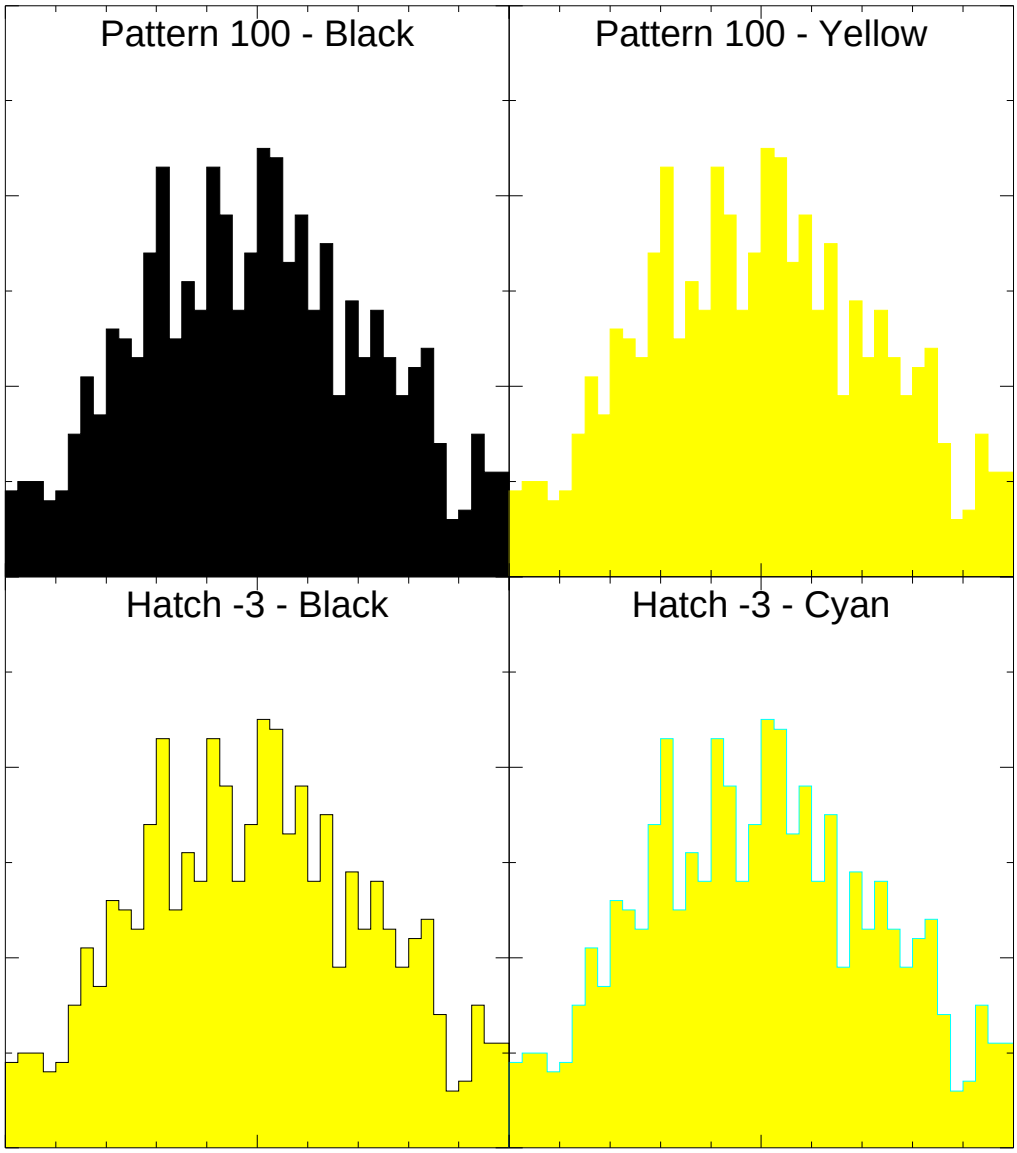


Figure B.8: Examples of Patterns

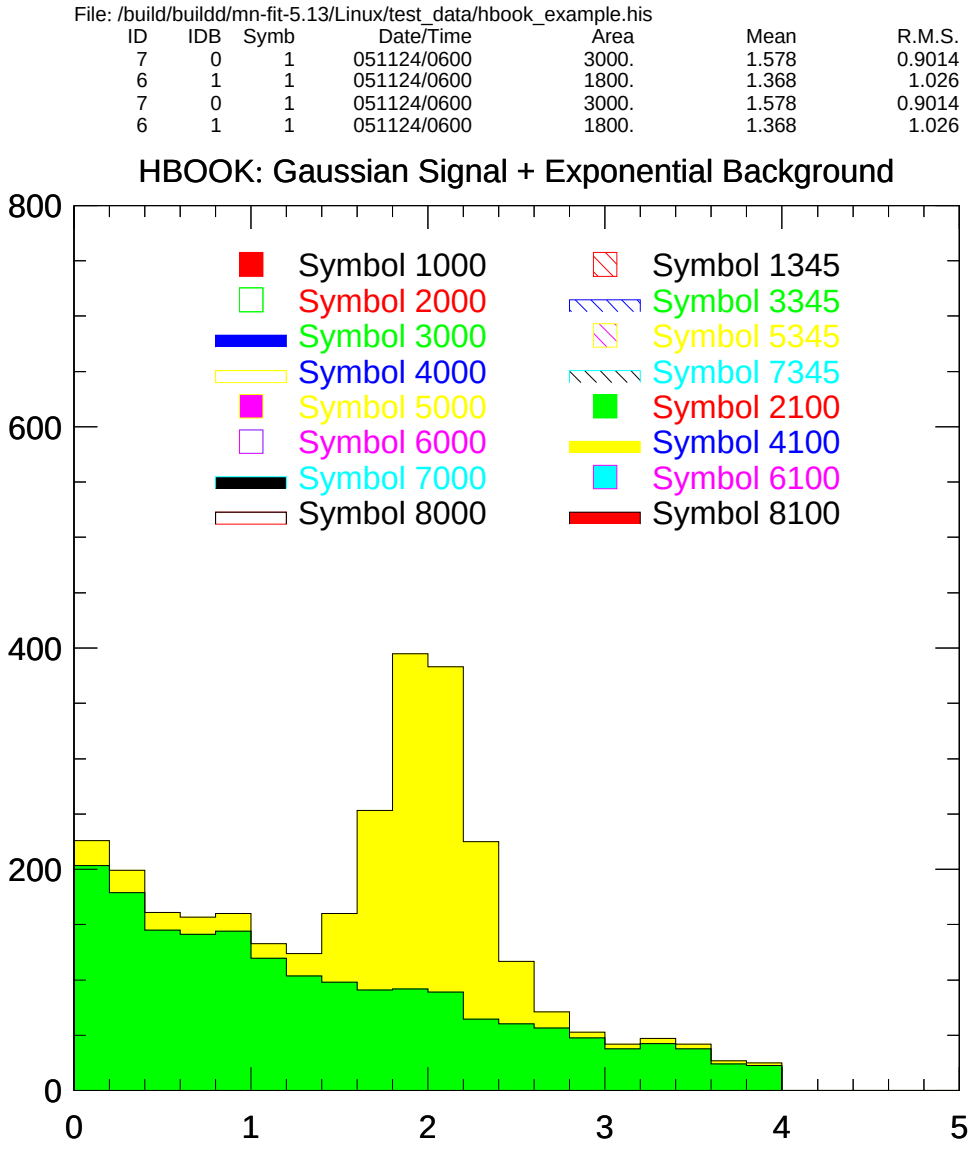


Figure B.9: Examples of Keys

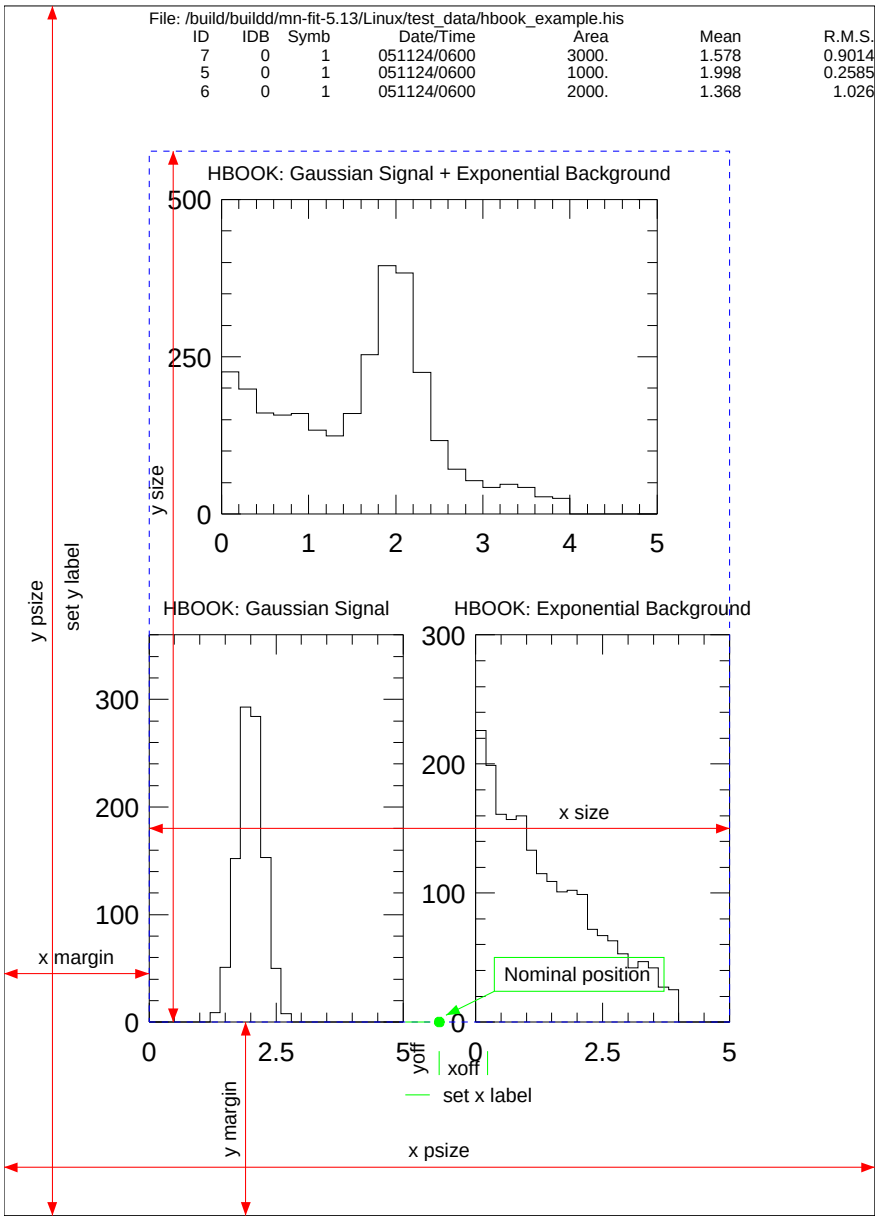


Figure B.10: Picture sizes and the commands to set them

Input	Roman	Greek	Spec	Zapf
\0	\0	∴0	∴⊗	✲✐
\1	\1	∴1	∴⊕	✲✑
\2	\2	∴2	∴∅	✲✒
\3	\3	∴3	∴◊	✲✓
\4	\4	∴4	∴•	✲✔
\5	\5	∴5	∴→	✲✕
\6	\6	∴6	∴↑	✲✖
\7	\7	∴7	∴←	✲✗
\8	\8	∴8	∴↓	✲✘
\9	\9	∴9	∴↔	✲✙
\10	\10	∴10	∴⊕⊗	✲✑✐
\11	\11	∴11	∴⊕⊕	✲✑✑
\12	\12	∴12	∴⊕∅	✲✑✒
\13	\13	∴13	∴⊕◊	✲✑✓
\14	\14	∴14	∴⊕•	✲✑✔
\15	\15	∴15	∴⊕→	✲✑✕
\16	\16	∴16	∴⊕↑	✲✑✖
\17	\17	∴17	∴⊕←	✲✑✗
\18	\18	∴18	∴⊕↓	✲✑✘
\19	\19	∴19	∴⊕↔	✲✑✙
\20	\20	∴20	∴∅⊗	✲✒✐
\21	\21	∴21	∴∅⊕	✲✒✑
\22	\22	∴22	∴∅∅	✲✒✒
\23	\23	∴23	∴∅◊	✲✒✓
\24	\24	∴24	∴∅•	✲✒✔

Input	Roman	Greek	Spec	Zapf
\25	\25	∴25	∴∅→	✲✒✕
\26	\26	∴26	∴∅↑	✲✒✖
\27	\27	∴27	∴∅←	✲✒✗
\28	\28	∴28	∴∅↓	✲✒✘
\29	\29	∴29	∴∅↔	✲✒✙
\30	\30	∴30	∴◊⊗	✲✓✐
\31	\31	∴31	∴◊⊕	✲✓✑
\32	\32	∴32	∴◊∅	✲✓✒
\33	\33	∴33	∴◊◊	✲✓✓
\34	\34	∴34	∴◊•	✲✓✔
\35	\35	∴35	∴◊→	✲✓✕
\36	\36	∴36	∴◊↑	✲✓✖
\37	\37	∴37	∴◊←	✲✓✗
\38	\38	∴38	∴◊↓	✲✓✘
\39	\39	∴39	∴◊↔	✲✓✙
\40				
\41	!	!	!	✁
\42	"	∀	∀	✂
\43	#	#	#	✃
\44	$	∃	∃	✄
\45	%	%	%	✆
\46	&	&	&	✇
\47	’	϶	϶	✈
\48	8	8	↓	✘
\49	9	9	↔	✙

Input	Roman	Greek	Spec	Zapf
\50	(	(	≡	✉
\51	)	)	″	✌
\52	*	∗	⟨	✍
\53	+	+	+	✎
\54	,	,	,	✏
\55	-	−	∠	✐
\56	.	.	.	✑
\57	/	/	÷	✒
\58	8	8	↓	✘
\59	9	9	↔	✙
\60	0	0	⊗	✐
\61	1	1	⊕	✑
\62	2	2	∅	✒
\63	3	3	◊	✓
\64	4	4	•	✔
\65	5	5	→	✕
\66	6	6	↑	✖
\67	7	7	←	✗
\68	8	8	↓	✘
\69	9	9	↔	✙
\70	8	8	↓	✘
\71	9	9	↔	✙
\72	:	:	:	✚
\73	;	;	;	✛
\74	<	<	<	✜

Figure B.11: Postscript characters - Codes 0 to 75

Input	Roman	Greek	Spec	Zapf
\75	=	=	≠	✝
\76	>	>	>	✞
\77	?	?	?	✟
\78	\78	∴78	∴←	✲✗✘
\79	\79	∴79	∴↔	✲✗✙
\80	\80	∴80	∴⊗	✲✘✐
\81	\81	∴81	∴⊕	✲✘✑
\82	\82	∴82	∴∅	✲✘✒
\83	\83	∴83	∴◊	✲✘✓
\84	\84	∴84	∴•	✲✘✔
\85	\85	∴85	∴→	✲✘✕
\86	\86	∴86	∴↑	✲✘✖
\87	\87	∴87	∴←	✲✘✗
\88	\88	∴88	∴↓	✲✘✘
\89	\89	∴89	∴↔	✲✘✙
\90	\90	∴90	∴⊗	✲✙✐
\91	\91	∴91	∴⊕	✲✙✑
\92	\92	∴92	∴∅	✲✙✒
\93	\93	∴93	∴◊	✲✙✓
\94	\94	∴94	∴•	✲✙✔
\95	\95	∴95	∴→	✲✙✕
\96	\96	∴96	∴↑	✲✙✖
\97	\97	∴97	∴←	✲✙✗
\98	\98	∴98	∴↓	✲✙✘
\99	\99	∴99	∴↔	✲✙✙

Input	Roman	Greek	Spec	Zapf
\100	@	≅	≅	✠
\101	A	Α	±	✡
\102	B	Β	|	✢
\103	C	Χ	∃	✣
\104	D	Δ	∀	✤
\105	E	Ε	!	✥
\106	F	Φ	#	✦
\107	G	Γ	>	✧
\108	8	8	↓	✘
\109	9	9	↔	✙
\110	H	Η	?	★
\111	I	Ι	∫	☆
\112	J	ϑ	:	✪
\113	K	Κ	;	✫
\114	L	Λ	<	✬
\115	M	Μ	[	✭
\116	N	Ν	]	✮
\117	O	Ο	≥	✯
\118	8	8	↓	✘
\119	9	9	↔	✙
\120	P	Π	{	✰
\121	Q	Θ	}	✱
\122	R	Ρ	√	✲
\123	S	Σ	♠	✳
\124	T	Τ	♥	✴

Input	Roman	Greek	Spec	Zapf
\125	U	Υ	♦	✵
\126	V	ς	♣	✶
\127	W	Ω	≤	✷
\128	8	8	↓	✘
\129	9	9	↔	✙
\130	X	Ξ	×	✸
\131	Y	Ψ	%	✹
\132	Z	Ζ	∞	✺
\133	[	[	[	✻
\134	\	∴	∴	✼
\135	]	]	]	✽
\136	^	⊥	⊥	✾
\137	_	_	_	✿
\138	8	8	↓	✘
\139	9	9	↔	✙
\140	‘	¯	¯	❀
\141	a	α	≈	❁
\142	b	β	≅	❂
\143	c	χ	⊥	❃
\144	d	δ	∂	❄
\145	e	ε	f	❅
\146	f	ϕ	∩	❆
\147	g	γ	∪	❇
\148	8	8	↓	✘
\149	9	9	↔	✙

Figure B.12: Postscript characters - Codes 75 to 149

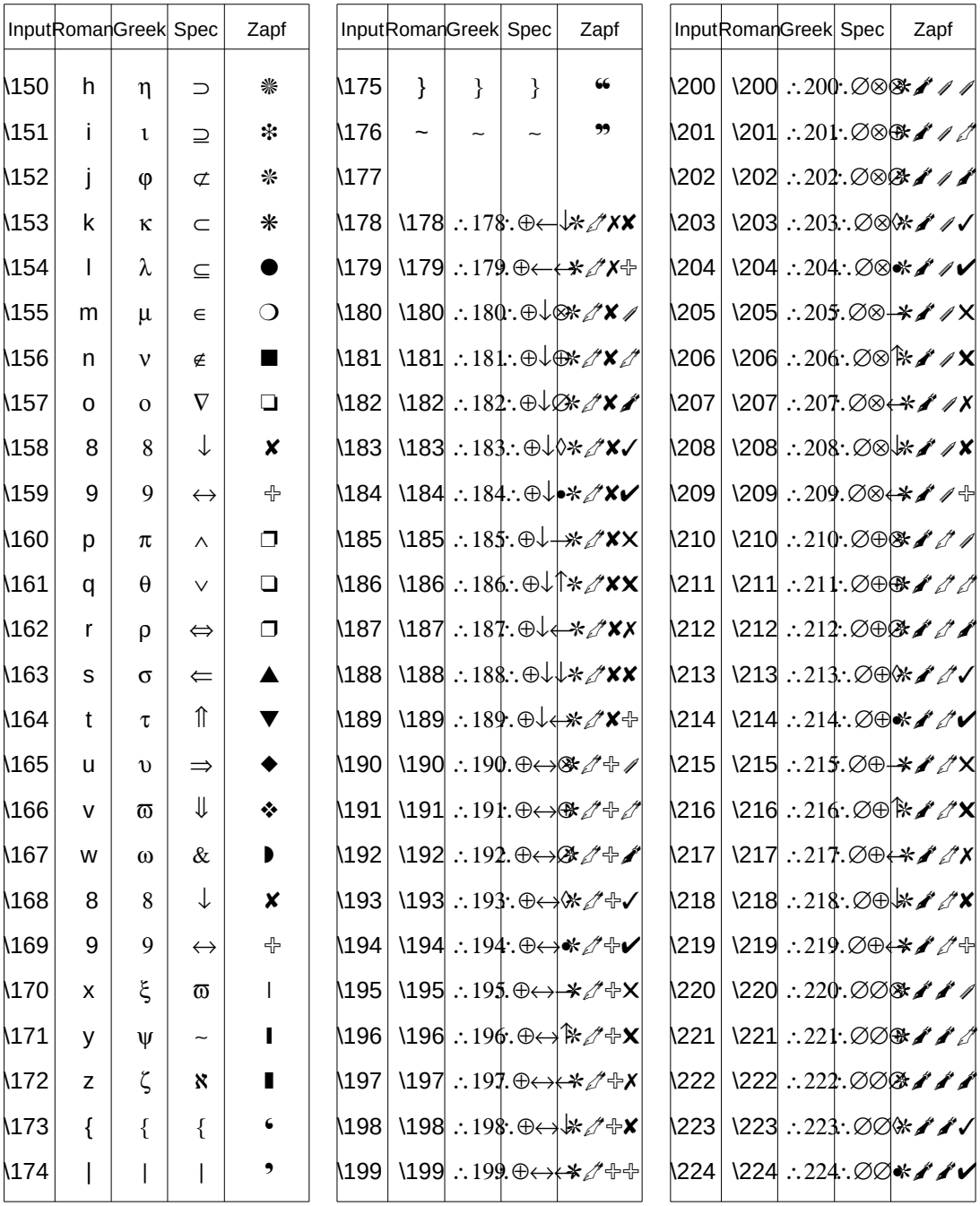


Figure B.13: Postscript characters - Codes 150 to 224

Input	Roman	Greek	Spec	Zapf
\225	\225	∴	∅	✁
\226	\226	∴	∅	✂
\227	\227	∴	∅	✃
\228	\228	∴	∅	✄
\229	\229	∴	∅	✆
\230	\230	∴	∅	✇
\231	\231	∴	∅	✈
\232	\232	∴	∅	✉
\233	\233	∴	∅	☛
\234	\234	∴	∅	☞
\235	\235	∴	∅	✌
\236	\236	∴	∅	✍
\237	\237	∴	∅	✎
\238	\238	∴	∅	✏
\239	\239	∴	∅	✐
\240	\240	∴	∅	✑
\241	¡	ϒ	ϒ	✒
\242	¢	′	′	✓
\243	£	≤	≤	✔
\244	⁄	⁄	⁄	✕
\245	¥	∞	∞	✖
\246	ƒ	ƒ	ƒ	✗
\247	§	♣	♣	✘
\248	8	8	↓	✙
\249	9	9	↔	✚

Input	Roman	Greek	Spec	Zapf
\250	¤	♦	♦	✛
\251	'	♥	♥	✜
\252	“	♠	♠	✝
\253	«	↔	↔	✞
\254	‹	←	←	✟
\255	›	↑	↑	✠
\256	fi	→	→	✡
\257	fl	↓	↓	✢
\258	8	8	↓	✣
\259	9	9	↔	✤
\260	à	°	°	✥
\261	–	±	±	✦
\262	†	″	″	✧
\263	‡	≥	≥	★
\264	·	×	×	✩
\265	À	∝	∝	✪
\266	¶	∂	∂	✫
\267	•	•	•	✬
\268	8	8	↓	✭
\269	9	9	↔	✮
\270	,	÷	÷	✯
\271	„	≠	≠	✰
\272	”	≡	≡	✱
\273	»	≈	≈	✲
\274	…	…	…	✳

Input	Roman	Greek	Spec	Zapf
\275	‰	⏐	⏐	✴
\276	â	—	—	✵
\277	¿	↵	↵	✶
\278	8	8	↓	✷
\279	9	9	↔	✸
\280	80	80	↓⊗	✹
\281	81	81	↓⊕	✺
\282	82	82	↓∅	✻
\283	83	83	↓◊	✼
\284	84	84	↓•	✽
\285	85	85	↓→	✾
\286	86	86	↓↑	✿
\287	87	87	↓←	❀
\288	88	88	↓↓	❁
\289	89	89	↓↔	❂
\290	90	90	↔⊗	❃
\291	91	91	↔⊕	❄
\292	92	92	↔∅	❅
\293	93	93	↔◊	❆
\294	94	94	↔•	❇
\295	95	95	↔→	❈
\296	96	96	↔↑	❉
\297	97	97	↔←	❊
\298	98	98	↔↓	❋
\299	99	99	↔↔	❋❋

Figure B.14: Postscript characters - Codes 225 to 299

Input	Roman	Greek	Spec	Zapf
\300	Â	א	א	①
\301	`	ℑ	ℑ	②
\302	´	ℜ	ℜ	③
\303	ˆ	℘	℘	④
\304	˜	⊗	⊗	⑤
\305	¯	⊕	⊕	⑥
\306	˘	∅	∅	⑦
\307	˙	∩	∩	⑧
\308	8	8	↓	✘
\309	9	9	↔	✙
\310	¨	∪	∪	⑨
\311	ä	⊃	⊃	⑩
\312	°	⊇	⊇	❶
\313	¸	⊄	⊄	❷
\314	Ä	⊂	⊂	❸
\315	˝	⊆	⊆	❹
\316	˛	∈	∈	❺
\317	ˇ	∉	∉	❻
\318	8	8	↓	✘
\319	9	9	↔	✙
\320	—	∠	∠	❼
\321	ç	∇	∇	❽
\322	Ç	®	®	❾
\323	é	©	©	❿
\324	É	™	™	➔

Input	Roman	Greek	Spec	Zapf
\325	è	∏	∏	→
\326	È	√	√	↔
\327	ê	·	·	↕
\328	8	8	↓	✘
\329	9	9	↔	✙
\330	Ê	¬	¬	➘
\331	ë	∧	∧	➔
\332	Ë	∨	∨	➘
\333	î	⇔	⇔	→
\334	Î	⇐	⇐	➔
\335	ï	⇑	⇑	→
\336	Ï	⇒	⇒	➞
\337	ñ	⇓	⇓	➟
\338	8	8	↓	✘
\339	9	9	↔	✙
\340	Ñ	◊	◊	➟
\341	Æ	〈	〈	➡
\342	ô	®	®	➢
\343	ª	©	©	➢
\344	Ô	™	™	➤
\345	ö	∑	∑	➥
\346	Ö	⎛	⎛	➦
\347	û	⎜	⎜	➧
\348	8	8	↓	✘
\349	9	9	↔	✙

Input	Roman	Greek	Spec	Zapf
\350	Ł	⎝	⎝	➨
\351	Ø	⎡	⎡	➩
\352	Œ	⎢	⎢	➪
\353	º	⎣	⎣	➫
\354	Û	⎡	⎡	➬
\355	ü	⎧	⎧	➭
\356	Ü	⎩	⎩	➮
\357	å	⎪	⎪	➯
\358	8	8	↓	✘
\359	9	9	↔	✙
\360	ð			
\361	æ	〉	〉	➱
\362	Å	⌡	⌡	➲
\363	ÿ	⎛	⎛	➳
\364	Ÿ	⎜	⎜	➴
\365	ı	⎭	⎭	➵
\366	á	⎞	⎞	➶
\367	Á	⎟	⎟	➷
\368	8	8	↓	✘
\369	9	9	↔	✙
\370	ł	⎭	⎭	➸
\371	ø	⎤	⎤	➹
\372	œ	⎟	⎟	➺
\373	ß	⎦	⎦	➻
\374	ù	⎤	⎤	➼

Figure B.15: Postscript characters - Codes 300 to 374

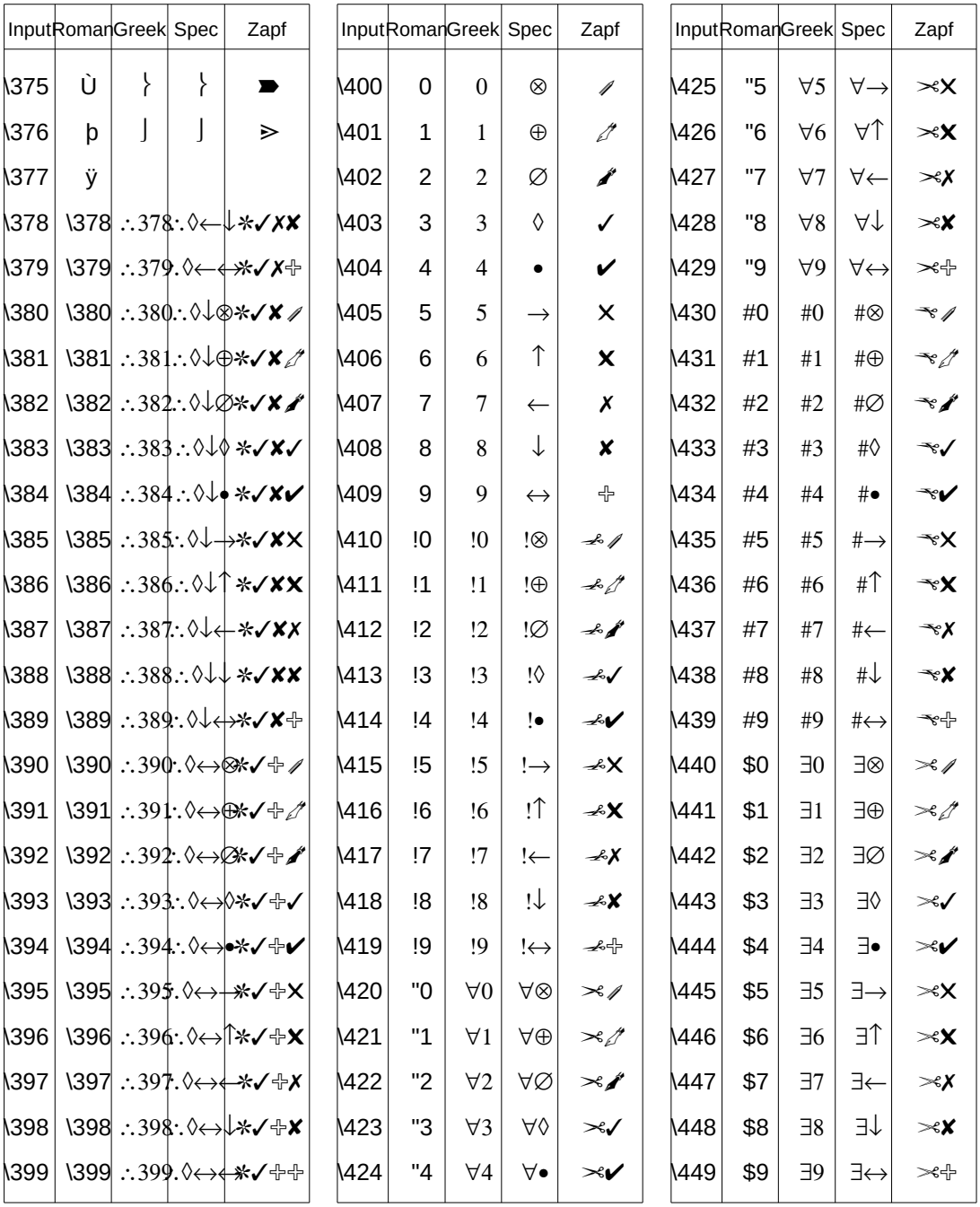


Figure B.16: Postscript characters - Codes 375 to 449

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. The demonstration files can be found in the directory 'mn_fit_help:' (VMS) or '$MN_FIT/help' (Unix). They are called 'demo01, demo02, ...' These files can also be found in the CMZ file 'mn_fit_cmz:mn_util.cmz' (VMS) or '$MN_FIT/cmz/mn_util.cmz' (Unix) in the directory '//mn_util/demo'. In the same directories you can find the command files used to produce the figures in Appendix B. All the demonstration files except `demo07.mnf` were run on a Linux PC using the X Windows version of Mn_Fit. `demo07` was run on an L3 HP workstation. On VMS 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:

```
* VMS
  file mn_fit_cmz:mn_util -r
  sel VMS
* Unix
  file $MN_FIT/cmz/mn_util -r
  sel UNIX
*
  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.
`demo06`	Drawing possibilities.
`demo07`	Display of an L3 Luminosity Monitor Event.

In the following examples the prompt is shown in **boldface**, user typin is underlined, and comments are shown in *italics*.

Font -1004 (i.e. Helevetica) is the default on all machines.

C.1 Demo 1: Simple Fetch and Plot

```
MN_CMD> exec demo01
 Reading commands from unit  11
 File: demo01.mnf
MN_CMD> !-------------------------------------------------------------------------
MN_CMD> ! Mn_Fit Demonstration file number 1.
MN_CMD> ! Simple fetch and plot.
MN_CMD> !-------------------------------------------------------------------------
MN_CMD> !
MN_CMD>
MN_CMD>
MN_CMD> fet $MN_FIT_SYS/test_data/hbook4_test.his 1
 Reading histograms from unit   1
 File: /home/brock/mn_fit/Linux/test_data/hbook4_test.his
 MN_HBF: A total of     1 plots have been read in
MN_CMD>
MN_CMD>
MN_CMD> set default             !Set everything back to default
MN_CMD> set y mode integer      !Show y scale as an integer
MN_CMD> set footer on           !Turn on the footer
MN_CMD> !
MN_CMD> plot 1
MN_CMD> !
MN_CMD>
MN_CMD>
MN_CMD> !Make a hardcopy file
MN_CMD> exec $MN_FIT_SYS/help/demohard 01
 Reading commands from unit  12
 File: /home/brock/mn_fit/Linux/help/demohard.mnf
MN_CMD> !
MN_CMD> ! Macro to make a Postscript hardcopy
MN_CMD> !
MN_CMD> inquire 1 'Give demonstration number'
MN_CMD> set hard demo01.ps      !Set the hardcopy filename
MN_CMD> hardcopy post           !Make a Postscript hardcopy
 *** MN_FIL: File /home/brock/mn_fit/Linux/help/demo01.ps will be overwritten
 TVCAP: Hardcopy to unit 13, File: /home/brock/mn_fit/Linux/help/demo01.ps
 Postscript selected
 None selected
MN_CMD> close                   !Close the hardcopy file
 End of Macro demohard.mnf, Unit= 12. Exiting
MN_CMD>
MN_CMD>
 End of Macro demo01.mnf, Unit= 11. Exiting
```

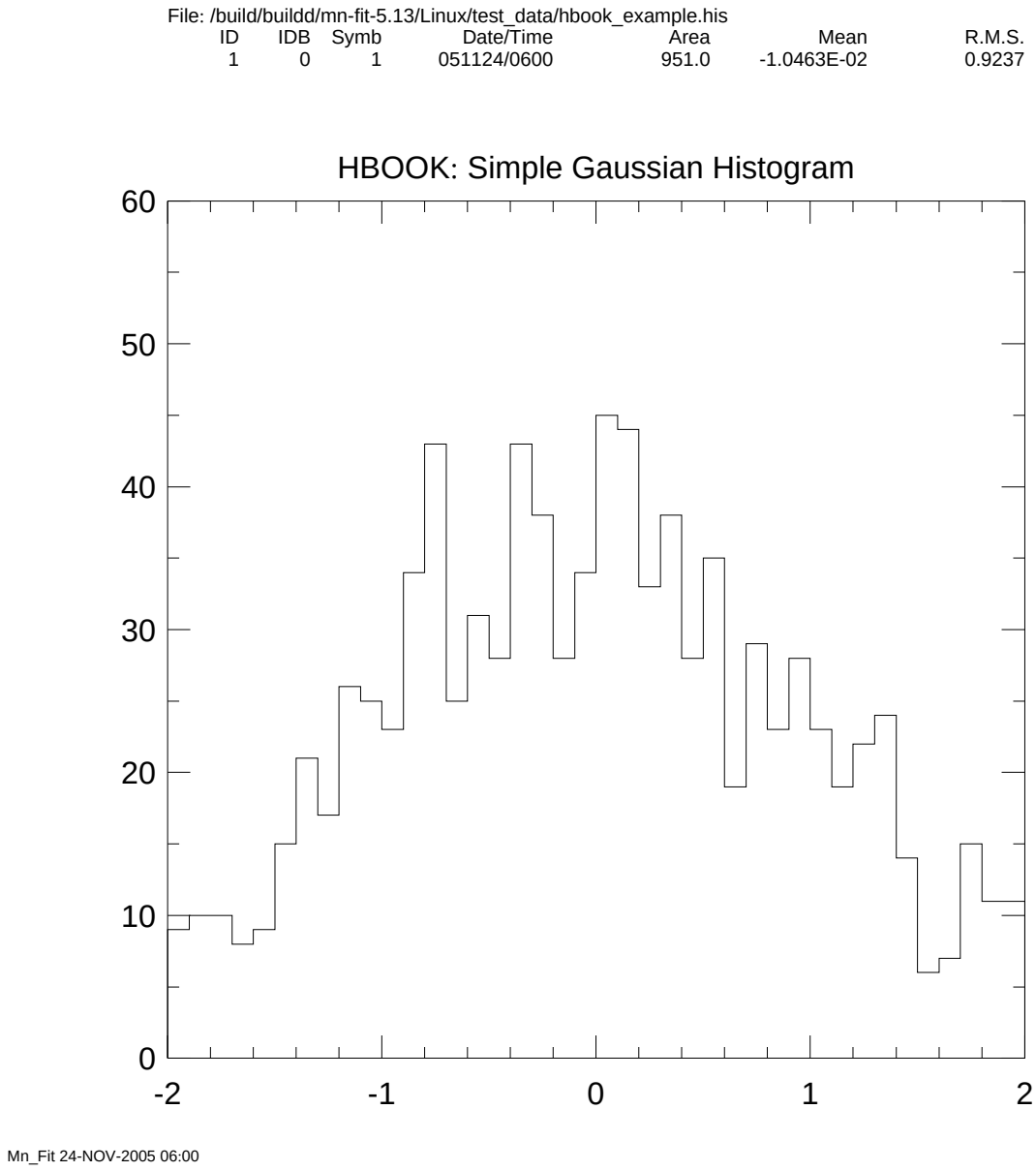


Figure C.1: Demo 1 — Simple Fetch and Plot.

C.2 Demo 2: Simple Gaussian Fit

```
MN_CMD> exec demo02 '!'
 Reading commands from unit  11
 File: demo02.mnf
MN_CMD> !------------------------------------------------------------------------------
MN_CMD> ! Mn_Fit Demonstration file number 2.
MN_CMD> ! Gaussian Fit to Histogram 1
MN_CMD> !------------------------------------------------------------------------------
MN_CMD> !
MN_CMD>
MN_CMD>
MN_CMD> fet $MN_FIT_SYS/test_data/hbook4_test.his 1
 Closing MN_HBIN: /home/brock/mn_fit/Linux/test_data/hbook4_test.his
 PAWC             File: COMMON /PAWC/ in memory
 Reading histograms from unit   1
 File: /home/brock/mn_fit/Linux/test_data/hbook4_test.his
 MN_HBF: A total of     1 plots have been read in
MN_CMD>
MN_CMD>
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 (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/lik 1
 Will do a likelihood fit to histogram:       1   0;
 ================================================================================
  MINUIT RELEASE 96.03  INITIALIZED.   DIMENSIONS 100/ 50  EPSMAC=  0.89E-15
 ================================================================================
 **********
 **    1 **SET PRINTOUT           0.000
 **********
 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=   794.3265     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.
 MIGRAD WILL VERIFY CONVERGENCE AND ERROR MATRIX.
 FCN=   45.21896     FROM MIGRAD    STATUS=CONVERGED     81 CALLS       82 TOTAL
                     EDM=  0.18E-06    STRATEGY= 1      ERROR MATRIX ACCURATE
  EXT PARAMETER                                   STEP         FIRST
  NO.   NAME        VALUE          ERROR          SIZE      DERIVATIVE
   1    AREA        1023.2        35.564       0.11004      -0.18093E-05
   2    MEAN      -0.15000E-01   0.42962E-01   0.14263E-03   0.47329E-02
   3    SIGMA       1.1062       0.45725E-01   0.14169E-03  -0.11606E-01
MINUIT> display                   !Show the results
 Results of Fit to Plot(s):        1   0;
 Likelihood =    45.2
 Chi**2 =     44.1  for  40 -   3 d.o.f.                  C.L. = 19.7    %
 Plot Area Total/Fit  951.000    /  951.000               Fit Status  3
 Func Area Total/Fit  950.902    /  950.902               E.D.M. 1.777E-07
            Name       Value                     Errors
                                          Parabolic              Minos
 Function  1: Gaussian (sigma)
   1( 1)  1 AREA        1023.2     +/-  35.564     -  0.0000     +  0.0000
   1( 2)  2 MEAN      -1.50003E-02 +/- 4.29621E-02 -  0.0000     +  0.0000
   1( 3)  3 SIGMA       1.1062     +/- 4.57245E-02 -  0.0000     +  0.0000
MINUIT> !
MINUIT>
MINUIT>
MINUIT> !Make a hardcopy file
MINUIT> exec $MN_FIT_SYS/help/demohard 02
 Reading commands from unit  12
 File: /home/brock/mn_fit/Linux/help/demohard.mnf
MINUIT> !
MINUIT> ! Macro to make a Postscript hardcopy
MINUIT> !
MINUIT> inquire 1 'Give demonstration number'
MINUIT> set hard demo02.ps        !Set the hardcopy filename
MINUIT> hardcopy post             !Make a Postscript hardcopy
 *** MN_FIL: File /home/brock/mn_fit/Linux/help/demo02.ps will be overwritten
 TVCAP: Hardcopy to unit 13, File: /home/brock/mn_fit/Linux/help/demo02.ps
 Postscript selected
 None selected
MINUIT> close                     !Close the hardcopy file
```

```
 End of Macro demohard.mnf, Unit= 12. Exiting
```
MINUIT>
MINUIT>
MINUIT> *!*
MINUIT> `inquire 1 'Type !  if you do not want to exit from MINUIT'`
MINUIT> *! exit !Exit from MINUIT*
```
 End of Macro demo02.mnf, Unit= 11. Exiting
```

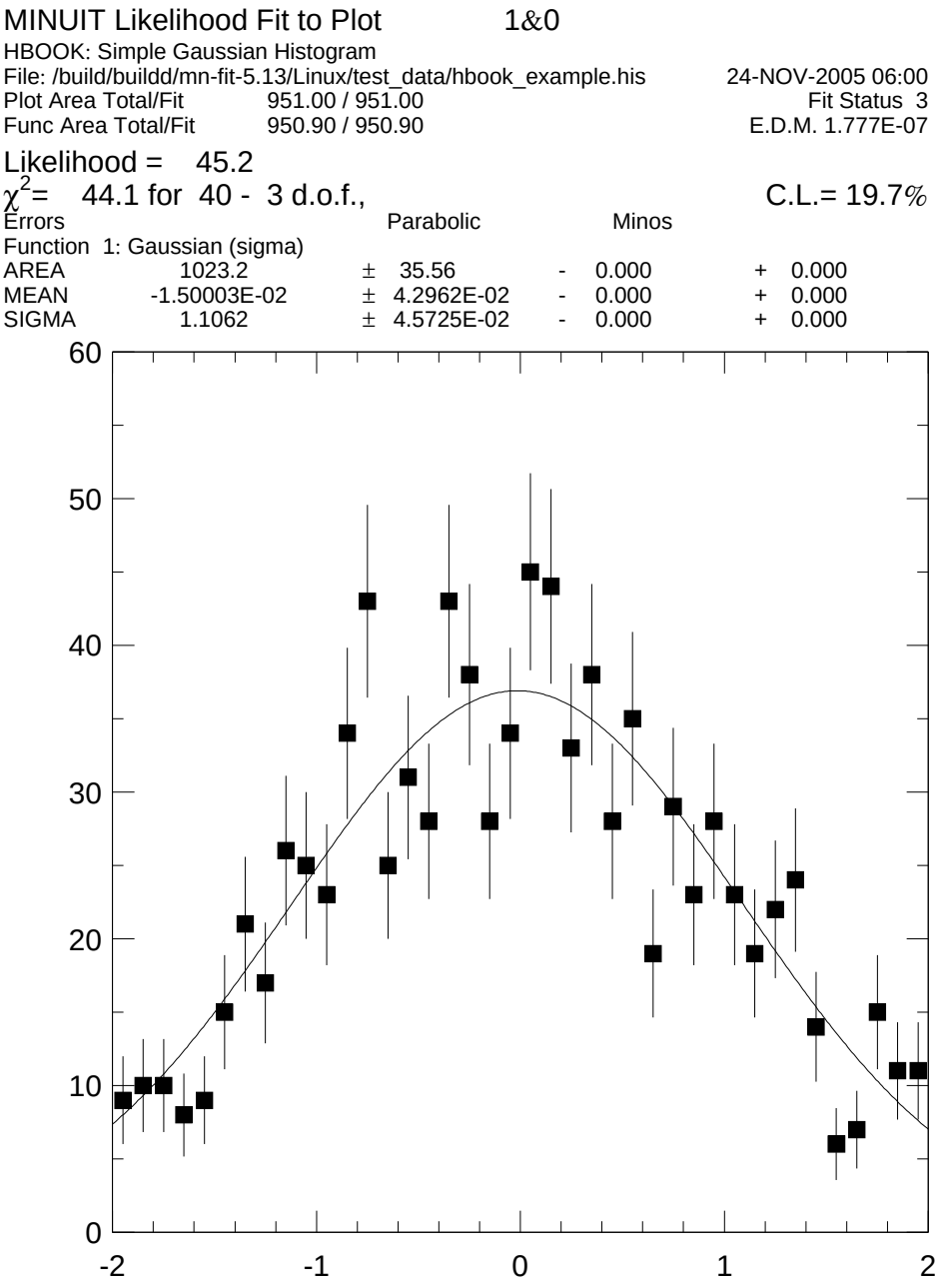


Figure C.2: Demo 2 — Simple Gaussian Fit.

C.3 Demo 3: Plotting with overlays, inserts, colours and fonts

```
MN_CMD> exec demo03
 Reading commands from unit  11
 File: demo03.mnf
MN_CMD> !-----------------------------------------------------------
MN_CMD> ! Mn_Fit Demonstration file number 3.
MN_CMD> ! Demonstration of Windowing, with COMMENTS, KEYS and use of options
MN_CMD> ! for axes. You can print this picture on a colour printer 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>
MN_CMD>
MN_CMD> fetch $MN_FIT_SYS/test_data/hbook4_test.his 0
 Closing MN_HBIN: /home/brock/mn_fit/Linux/test_data/hbook4_test.his
 PAWC             File: COMMON /PAWC/ in memory
 Reading histograms from unit   1
 File: /home/brock/mn_fit/Linux/test_data/hbook4_test.his
 MN_HBF: A total of    26 plots have been read in
MN_CMD>
MN_CMD>
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 ?: x psize 20
SET> Give variable name or ?: y psize 21
SET> Give variable name or ?: x marg 2.5
SET> Give variable name or ?: y size 18
SET> Give variable name or ?: thick 2
SET> Give variable name or ?: thick frame 4
SET> Give variable name or ?: header off
SET> Give variable name or ?: sym 32
SET> Give variable name or ?: ! Make 3 plots with 0 y separation
SET> Give variable name or ?: ! Only draw the frame on the bottom and left
SET> Give variable name or ?: ! 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' = = = = = -1004
SET> Give variable name or ?: X SCALE = = 0.3 = = -1003
SET> Give variable name or ?: Y SCALE = = 0.3 = = -1003
SET> Give variable name or ?: TSIZE 0.3
```

```
SET> Give variable name or ?: endset
MN_CMD> !
MN_CMD> ! Set up the colours. User variable to store default symbol colour.
MN_CMD> !
MN_CMD> dep csymbol = 2
 CSYMBOL =   2.00000
MN_CMD> set
SET> Give variable name or ?: colour frame black
SET> Give variable name or ?: colour header black
SET> Give variable name or ?: colour label 4
SET> Give variable name or ?: colour scale green
SET> Give variable name or ?: colour symbol red
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 = = -1003
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 14
SET> Give variable name or ?: sym 1
SET> Give variable name or ?: hatch 343
SET> Give variable name or ?: colour symbol magenta
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 csymbol
SET> Give variable name or ?: Y LABEL 'Number of Events' -1.5 = = = = -1004
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> ! 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, 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, 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 19.3 0.3 = = = -6
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 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 ?: 0.9 300 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: 1,320
DRAW> Point 2 Give x,y: 1.9 160
MN_CMD> END
MN_CMD> *!*
MN_CMD>
MN_CMD>
MN_CMD> *!Make a hardcopy file*
MN_CMD> exec $MN_FIT_SYS/help/demohard 03
```
 Reading commands from unit  12
 File: /home/brock/mn_fit/Linux/help/demohard.mnf
```
MN_CMD> *!*
MN_CMD> *! Macro to make a Postscript hardcopy*
MN_CMD> *!*
MN_CMD> inquire 1 'Give demonstration number'
MN_CMD> set hard demo03.ps *!Set the hardcopy filename*
MN_CMD> hardcopy post *!Make a Postscript hardcopy*

```
 *** MN_FIL: File /home/brock/mn_fit/Linux/help/demo03.ps will be overwritten
 TVCAP: Hardcopy to unit 13, File: /home/brock/mn_fit/Linux/help/demo03.ps
 Postscript selected
 TVRNG: Device Postscript  Scaling factor 0.900 will be used
 None selected
MN_CMD> close                    !Close the hardcopy file
 End of Macro demohard.mnf, Unit= 12. Exiting
MN_CMD>
MN_CMD>
 End of Macro demo03.mnf, Unit= 11. Exiting
```

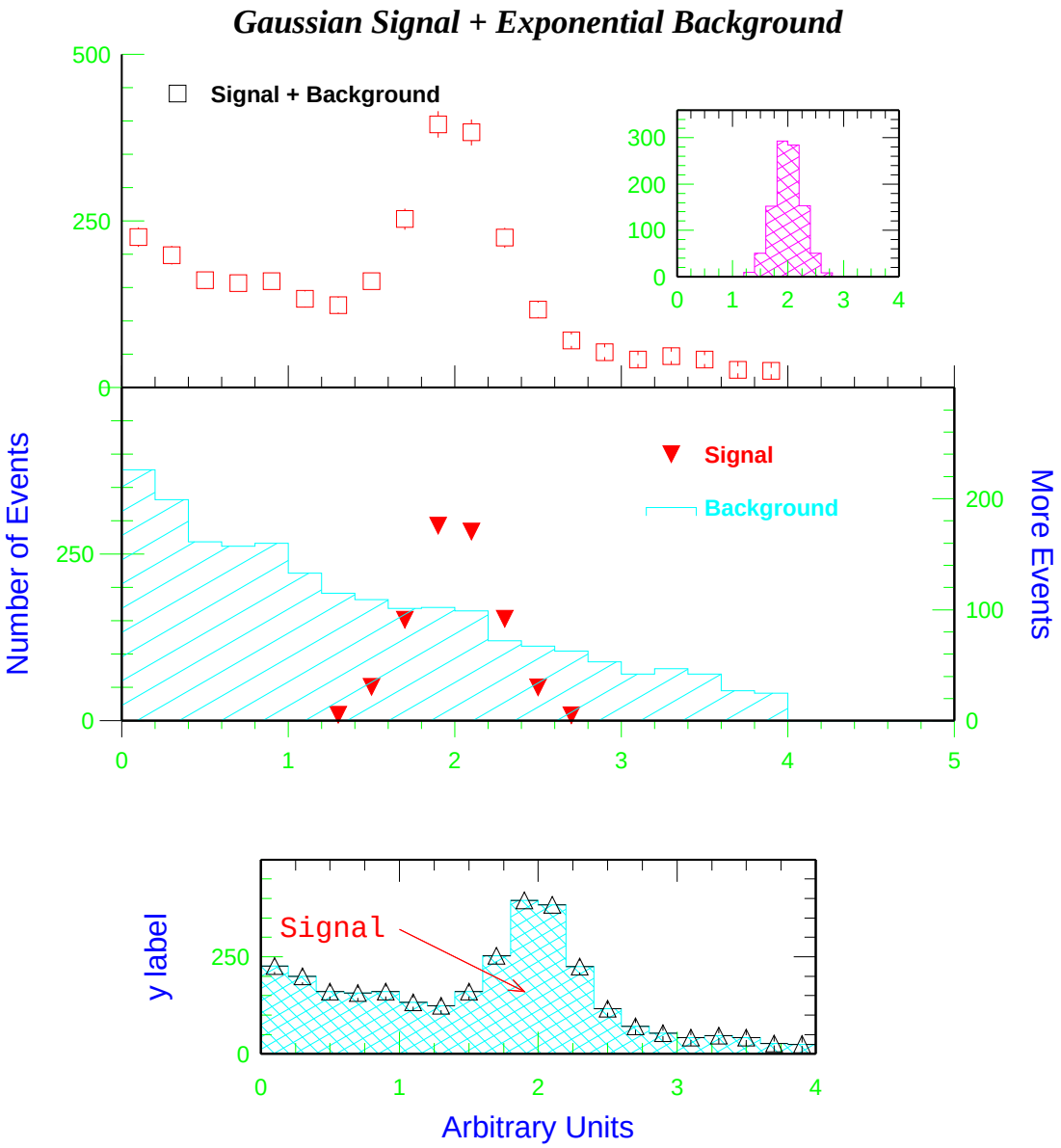


Figure C.3: Demo 3 — Plotting with overlays, inserts, colours and fonts.

C.4 Demo 4: Different ways of displaying 2-D histograms

```
MN_CMD> exec demo04
 Reading commands from unit  11
 File: demo04.mnf
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>
MN_CMD>
MN_CMD> fetch $MN_FIT_SYS/test_data/hbook4_test.his 10
 Closing MN_HBIN: /home/brock/mn_fit/Linux/test_data/hbook4_test.his
 PAWC             File: COMMON /PAWC/ in memory
 Reading histograms from unit   1
 File: /home/brock/mn_fit/Linux/test_data/hbook4_test.his
 MN_HBF: A total of     1 plots have been read in
MN_CMD> set dir ntuple
MN_CMD> fetch $MN_FIT_SYS/test_data/hbook4_test.his 51
 Closing MN_HBIN: /home/brock/mn_fit/Linux/test_data/hbook4_test.his
 PAWC             File: COMMON /PAWC/ in memory
 Reading histograms from unit   1
 File: /home/brock/mn_fit/Linux/test_data/hbook4_test.his
 M_SDIR: Creating HBOOK directory: //PAWC/NTUPLE
 HBOOK directory set to: //MN_HBIN/NTUPLE
 MN_HBF: A total of     1 plots have been read in
MN_CMD>
MN_CMD>
MN_CMD> set
SET> Give variable name or ?: default
SET> Give variable name or ?: !Only give plot id and symbol
SET> Give variable name or ?: header brief
SET> Give variable name or ?: ! User title
SET> Give variable name or ?: title user 'Different ways of showing 2-D Plots'
SET> Give variable name or ?: ! Windows 2x2 with 1.5cm spacing
SET> Give variable name or ?: wind 2 2 1.5 1.5
SET> Give variable name or ?: endset
MN_CMD> cop 10 10&1
 HBOOK directory set to: //MN_HBIN
 *** WARNING in HRIN : Already existing histogram replaced : ID=      10
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/c1 10&2 45 45      !Rebinned version as a colour surface plot
 Axis vertices: x1,x2,y1,y2,z1,z2:   2   3   2   1   1   5
MN_CMD> !
MN_CMD> ! Plot the Ntuple directly as a table
MN_CMD> !
MN_CMD> set ntuple plot 51 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 51
MN_CMD> !
MN_CMD>
MN_CMD>
MN_CMD> !Make a hardcopy file
MN_CMD> exec $MN_FIT_SYS/help/demohard 04
 Reading commands from unit  12
 File: /home/brock/mn_fit/Linux/help/demohard.mnf
MN_CMD> !
MN_CMD> ! Macro to make a Postscript hardcopy
MN_CMD> !
MN_CMD> inquire 1 'Give demonstration number'
MN_CMD> set hard demo04.ps      !Set the hardcopy filename
MN_CMD> hardcopy post           !Make a Postscript hardcopy
 *** MN_FIL: File /home/brock/mn_fit/Linux/help/demo04.ps will be overwritten
 TVCAP: Hardcopy to unit 13, File: /home/brock/mn_fit/Linux/help/demo04.ps
 Postscript selected
 Axis vertices: x1,x2,y1,y2,z1,z2:   2   3   2   1   1   5
 None selected
MN_CMD> close                   !Close the hardcopy file
 End of Macro demohard.mnf, Unit= 12. Exiting
MN_CMD>
MN_CMD>
 End of Macro demo04.mnf, Unit= 11. Exiting
```

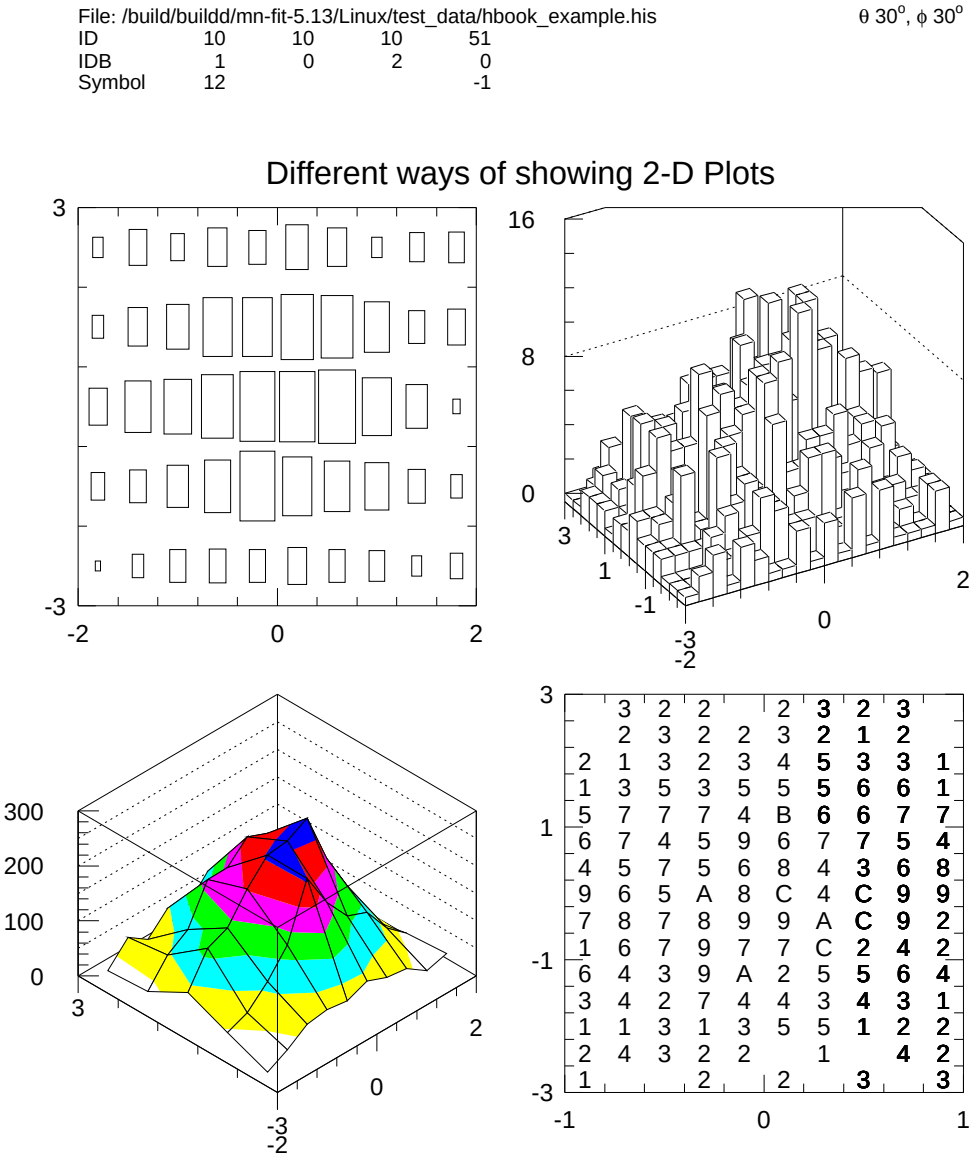


Figure C.4: Demo 4 — Different ways of displaying 2-D histograms.

C.5 Demo 5: Simultaneous fit of 2 plots with function overlays

```
MN_CMD> exec demo05 '!'
 Reading commands from unit  11
 File: demo05.mnf
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>
MN_CMD>
MN_CMD> fetch $MN_FIT_SYS/test_data/hbook4_test.his 0
 Closing MN_HBIN: /home/brock/mn_fit/Linux/test_data/hbook4_test.his
 PAWC             File: COMMON /PAWC/ in memory
 Reading histograms from unit   1
 File: /home/brock/mn_fit/Linux/test_data/hbook4_test.his
 MN_HBF: A total of    26 plots have been read in
MN_CMD>
MN_CMD>
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 (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> add 8 5 8&1 1 1
MN_CMD> part 8&1 1 5
MN_CMD> add 7 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 2
SET> Give variable name or ?: thick frame 4
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

```
 Possible fit types are:
  0 - Chi**2
  1 - Likelihood
  2 - Likelihood including function statistics
 1n - Parameters are fractions + overall normalization
```

Give fit type (0, 1, 10, 11 or 12, <CR>=0): 0

```
 Will do a chi**2 fit to histograms:       7   1;      8   1;
 ================================================================================
  MINUIT RELEASE 96.03  INITIALIZED.   DIMENSIONS 100/ 50  EPSMAC=  0.89E-15
 ================================================================================
 **********
 **    1 **SET PRINTOUT          0.000
 **********
 PARAMETER DEFINITIONS:
    NO.   NAME         VALUE      STEP SIZE      LIMITS
     1 'AREA      '    2000.0       500.00         no limits
     2 'AR1/AREA  '   0.50000      0.10000         0.0000       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 'CHEB01    '    0.0000       100.00         no limits
     8 'CHEB02    '    0.0000       100.00         no limits
     9 'NORM      '    100.00       50.000         no limits
    10 'NRM1/NORM '    0.0000      constant
    11 'CHEB01    '    0.0000       10.000         no limits
 **********
 **    2 **CALL FCN              1.000
 **********
```

```
 FCN=   2188.044     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        0.0000        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        0.0000        100.00
   8  CHEB02        0.0000        100.00
   9  NORM          100.00        50.000
  10  NRM1/NORM     0.0000       constant
  11  CHEB01        0.0000        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> ! Turn off the display text for screen device - saves time
MINUIT> set text off
MINUIT> display
 Results of Fit to Plot(s):       7   1;       8   1;
 Chi**2 =     31.3  for  28 -   5 d.o.f.                  C.L. = 11.6    %
 Individual chi**2 =    23.0,     8.3,
 Hist       7   1
 Plot Area Total/Fit  10000.0    /  5834.00               Fit Status  3
 Func Area Total/Fit  9417.90    /  6251.41               E.D.M. 4.643E-07
 Hist       8   1
 Plot Area Total/Fit  2270.00    /  720.000
 Func Area Total/Fit  1667.35    /  833.954
            Name       Value                    Errors
                                        Parabolic              Minos
 Function  1: Gaussian (sigma)
 * 1( 1)  1 AREA        2000.0     +/-  0.0000     -  0.0000     +  0.0000
 * 1( 2)  2 AR1/AREA   0.80000     +/-  0.0000     -  0.0000     +  0.0000
 * 1( 3)  3 MEAN        1.9000     +/-  0.0000     -  0.0000     +  0.0000
```

```
 * 1( 4)  4 SIGMA      0.30000     +/-  0.0000     -  0.0000     +  0.0000
 Function  2: Chebyshev Polynomial of Order 2
   2( 1)  5 NORM        1822.3     +/-  53.940     -  0.0000     +  0.0000
 * 2( 2)  6 NRM1/NORM   1.0000     +/-  0.0000     -  0.0000     +  0.0000
   2( 3)  7 CHEB01     -1.0587     +/- 6.21672E-02 -  0.0000     +  0.0000
   2( 4)  8 CHEB02     0.28225     +/- 4.51100E-02 -  0.0000     +  0.0000
 Function  3: Chebyshev Polynomial of Order 1
   3( 1)  9 NORM        397.97     +/-  16.647     -  0.0000     +  0.0000
 * 3( 2) 10 NRM1/NORM   0.0000     +/-  0.0000     -  0.0000     +  0.0000
   3( 3) 11 CHEB01     -1.0176     +/- 3.00878E-02 -  0.0000     +  0.0000
MINUIT> float 1 2 3 4
MINUIT> !
MINUIT> no_excl 0                !Remove all exclusions
 Exclusions for plot      7   1 removed
 There are no exclusions for plot      7   1
 Exclusions for plot      8   1 removed
 There are no exclusions for plot      8   1
 +++ 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 =     27.8  for  40 -   4 d.o.f.                    C.L. = 83.3    %
 Individual chi**2 =    19.2,     8.7,
 Hist       7   1
 Plot Area Total/Fit  10000.0    /  10000.0                 Fit Status  3
 Func Area Total/Fit  9899.09    /  9899.09                 E.D.M. 1.996E-07
 Hist       8   1
 Plot Area Total/Fit  2270.00    /  2270.00
 Func Area Total/Fit  2251.73    /  2251.73
            Name       Value                    Errors
                                          Parabolic              Minos
 Function  1: Gaussian (sigma)
   1( 1)  1 AREA        3065.0     +/-  101.09     -  0.0000     +  0.0000
   1( 2)  2 AR1/AREA   0.67901     +/- 1.24901E-02 -  0.0000     +  0.0000
   1( 3)  3 MEAN        2.0027     +/- 8.57366E-03 -  0.0000     +  0.0000
   1( 4)  4 SIGMA      0.24696     +/- 8.21562E-03 -  0.0000     +  0.0000
 Function  2: Chebyshev Polynomial of Order 2
 * 2( 1)  5 NORM        1822.3     +/-  0.0000     -  0.0000     +  0.0000
 * 2( 2)  6 NRM1/NORM   1.0000     +/-  0.0000     -  0.0000     +  0.0000
 * 2( 3)  7 CHEB01     -1.0587     +/-  0.0000     -  0.0000     +  0.0000
 * 2( 4)  8 CHEB02     0.28225     +/-  0.0000     -  0.0000     +  0.0000
 Function  3: Chebyshev Polynomial of Order 1
 * 3( 1)  9 NORM        397.97     +/-  0.0000     -  0.0000     +  0.0000
 * 3( 2) 10 NRM1/NORM   0.0000     +/-  0.0000     -  0.0000     +  0.0000
 * 3( 3) 11 CHEB01     -1.0176     +/-  0.0000     -  0.0000     +  0.0000
MINUIT> float 5 7 8 9 11
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	904.0		42.5	825.4	3.421	3.421
2	0.3000		0.100	796.0		39.9	765.8	0.573	3.993
3	0.5000		0.100	644.0		35.9	708.7	-3.247	7.240
4	0.7000		0.100	628.0		35.4	653.9	-0.536	7.776
5	0.9000		0.100	640.0		35.8	601.7	1.149	8.925
6	1.100		0.100	532.0		32.6	552.4	-0.390	9.315
7	1.300		0.100	478.0		30.9	514.1	-1.360	10.675
8	1.500		0.100	538.0		32.8	535.9	0.004	10.679
9	1.700		0.100	708.0		37.6	720.2	-0.105	10.784
10	1.900		0.100	994.0		44.6	983.7	0.053	10.837
11	2.100		0.100	964.0		43.9	952.7	0.066	10.903
12	2.300		0.100	594.0		34.5	617.1	-0.448	11.351
13	2.500		0.100	368.0		27.1	351.2	0.383	11.734
14	2.700		0.100	268.0		23.2	250.1	0.599	12.333
15	2.900		0.100	212.0		20.6	212.0	0.000	12.333
16	3.100		0.100	168.0		18.3	185.6	-0.919	13.252
17	3.300		0.100	188.0		19.4	162.2	1.769	15.021
18	3.500		0.100	168.0		18.3	141.3	2.125	17.146
19	3.700		0.100	108.0		14.7	122.8	-1.010	18.156
20	3.900		0.100	100.0		14.1	106.7	-0.222	18.378

Histogram 8 1 Number of points 20

Pnt	X	+/-	DX	Y	+/-	DY	Yfit	Chi**2	Total
1	1.100		0.100	137.0		11.7	129.1	0.450	18.828
2	1.300		0.100	125.0		11.2	126.8	-0.026	18.854
3	1.500		0.100	158.0		12.6	152.0	0.227	19.081
4	1.700		0.100	250.0		15.8	253.6	-0.052	19.133
5	1.900		0.100	394.0		19.8	391.9	0.011	19.144
6	2.100		0.100	390.0		19.7	388.4	0.006	19.150
7	2.300		0.100	233.0		15.3	238.4	-0.124	19.274
8	2.500		0.100	125.0		11.2	120.5	0.162	19.437
9	2.700		0.100	81.00		9.00	80.09	0.010	19.447
10	2.900		0.100	69.00		8.31	68.64	0.002	19.449
11	3.100		0.100	58.00		7.62	61.59	-0.223	19.671
12	3.300		0.100	55.00		7.42	54.86	0.000	19.672
13	3.500		0.100	44.00		6.63	48.13	-0.388	20.059
14	3.700		0.100	31.00		5.57	41.40	-3.491	23.551
15	3.900		0.100	44.00		6.63	34.68	1.975	25.526
16	4.100		0.100	27.00		5.20	27.95	-0.034	25.559
17	4.300		0.100	27.00		5.20	21.22	1.235	26.795
18	4.500		0.100	14.00		3.74	14.50	-0.018	26.812
19	4.700		0.100	8.000		2.83	7.773	0.006	26.819
20	4.900		0.100	0.000		0.00	1.046	0.000	26.819

MINUIT> set back 2 3
MINUIT> set display mode 3

```
MINUIT> display &11 &11          !Give secondary id's for background subtracted plots
 Results of Fit to Plot(s):       7   1;       8   1;
 Chi**2 =     26.8  for  40 -   9 d.o.f.                  C.L. = 68.1    %
 Individual chi**2 =    18.4,     8.4,
 Hist       7   1
 Plot Area Total/Fit  10000.0    /  10000.0               Fit Status  3
 Func Area Total/Fit  9965.25    /  9965.25               E.D.M. 1.274E-06
 Hist       8   1
 Plot Area Total/Fit  2270.00    /  2270.00
 Func Area Total/Fit  2262.61    /  2262.61
            Name       Value                    Errors
                                          Parabolic             Minos
 Function  1: Gaussian (sigma)
   1( 1)  1 AREA        2982.7     +/-  131.60     -  0.0000     +  0.0000
   1( 2)  2 AR1/AREA   0.67690     +/- 1.48066E-02 -  0.0000     +  0.0000
   1( 3)  3 MEAN        2.0033     +/- 8.65565E-03 -  0.0000     +  0.0000
   1( 4)  4 SIGMA      0.24215     +/- 9.48215E-03 -  0.0000     +  0.0000
 Function  2: Chebyshev Polynomial of Order 2
   2( 1)  5 NORM        1821.8     +/-  53.793     -  0.0000     +  0.0000
 * 2( 2)  6 NRM1/NORM   1.0000     +/-  0.0000     -  0.0000     +  0.0000
   2( 3)  7 CHEB01     -1.0906     +/- 6.17438E-02 -  0.0000     +  0.0000
   2( 4)  8 CHEB02     0.25880     +/- 4.49273E-02 -  0.0000     +  0.0000
 Function  3: Chebyshev Polynomial of Order 1
   3( 1)  9 NORM        408.80     +/-  16.667     -  0.0000     +  0.0000
 * 3( 2) 10 NRM1/NORM   0.0000     +/-  0.0000     -  0.0000     +  0.0000
   3( 3) 11 CHEB01     -1.0283     +/- 2.90657E-02 -  0.0000     +  0.0000
 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> set text on              !Turn text back on for final display
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>
MINUIT>
MINUIT> !Make a hardcopy file
MINUIT> exec $MN_FIT_SYS/help/demohard 05
```

```
 Reading commands from unit  12
 File: /home/brock/mn_fit/Linux/help/demohard.mnf
MINUIT> !
MINUIT> ! Macro to make a Postscript hardcopy
MINUIT> !
MINUIT> inquire 1 'Give demonstration number'
MINUIT> set hard demo05.ps        !Set the hardcopy filename
MINUIT> hardcopy post             !Make a Postscript hardcopy
 *** MN_FIL: File /home/brock/mn_fit/Linux/help/demo05.ps will be overwritten
 TVCAP: Hardcopy to unit 13, File: /home/brock/mn_fit/Linux/help/demo05.ps
 Postscript selected
 None selected
MINUIT> close                     !Close the hardcopy file
 End of Macro demohard.mnf, Unit= 12. Exiting
MINUIT>
MINUIT>
MINUIT> !
MINUIT> inquire 1 'Type !  if you do not want to exit from MINUIT'
MINUIT> ! exit !Exit from MINUIT
 End of Macro demo05.mnf, Unit= 11. Exiting
```

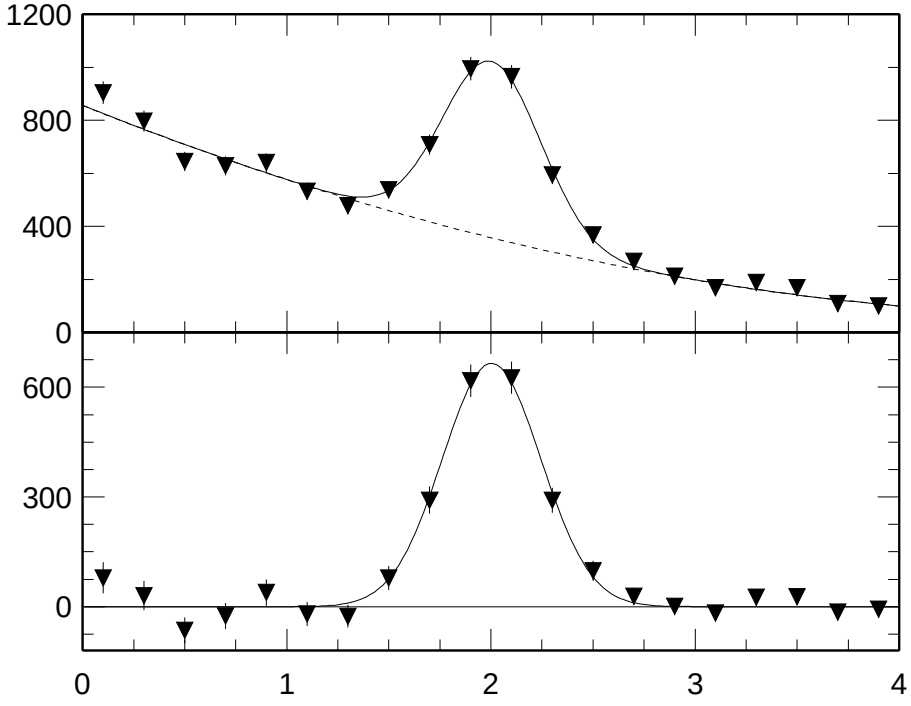
MINUIT χ^2 Fit to Plots 7&1; 8&1

Plot 7&1: HBOOK: Gaussian Signal + Exponential Background
File: /build/buildd/mn-fit-5.13/Linux/test_data/hbook_example.his 24-NOV-2005 06:00
Plot Area Total/Fit 10000. / 10000. Fit Status 3
Func Area Total/Fit 2019.0 / 2019.0 E.D.M. 1.274E-06

χ^2= 26.8 for 40 - 9 d.o.f., C.L.= 68.1%

Individual χ^2: 18.4, 8.4

	Errors		Parabolic		Minos		
	Function 1: Gaussian (sigma)						
	AREA	2982.7	± 131.6	-	0.000	+	0.000
	AR1/AREA	0.67690	± 1.4807E-02	-	0.000	+	0.000
	MEAN	2.0033	± 8.6556E-03	-	0.000	+	0.000
	SIGMA	0.24215	± 9.4821E-03	-	0.000	+	0.000
	Function 2: Chebyshev Polynomial of Order 2						
	NORM	1821.8	± 53.79	-	0.000	+	0.000
*	NRM1/NORM	1.0000	± 0.000	-	0.000	+	0.000
	CHEB01	-1.0906	± 6.1744E-02	-	0.000	+	0.000
	CHEB02	0.25880	± 4.4927E-02	-	0.000	+	0.000
	Function 3: Chebyshev Polynomial of Order 1						
	NORM	408.80	± 16.67	-	0.000	+	0.000
*	NRM1/NORM	0.0000	± 0.000	-	0.000	+	0.000
	CHEB01	-1.0283	± 2.9066E-02	-	0.000	+	0.000



MINUIT χ^2 Fit to Plots 7&1; 8&1

Plot 8&1: HBOOK: Straight Line Background

File: /build/buildd/mn-fit-5.13/Linux/test_data/hbook_example.his 24-NOV-2005 06:00

Plot Area Total/Fit 2270.0 / 2270.0 Fit Status 3

Func Area Total/Fit 963.71 / 963.71 E.D.M. 1.274E-06

χ^2= 26.8 for 40 - 9 d.o.f., C.L.= 68.1%

Individual χ^2: 18.4, 8.4

	Errors		Parabolic		Minos		
	Function 1: Gaussian (sigma)						
	AREA	2982.7	± 131.6	-	0.000	+	0.000
	AR1/AREA	0.67690	± 1.4807E-02	-	0.000	+	0.000
	MEAN	2.0033	± 8.6556E-03	-	0.000	+	0.000
	SIGMA	0.24215	± 9.4821E-03	-	0.000	+	0.000
	Function 2: Chebyshev Polynomial of Order 2						
	NORM	1821.8	± 53.79	-	0.000	+	0.000
*	NRM1/NORM	1.0000	± 0.000	-	0.000	+	0.000
	CHEB01	-1.0906	± 6.1744E-02	-	0.000	+	0.000
	CHEB02	0.25880	± 4.4927E-02	-	0.000	+	0.000
	Function 3: Chebyshev Polynomial of Order 1						
	NORM	408.80	± 16.67	-	0.000	+	0.000
*	NRM1/NORM	0.0000	± 0.000	-	0.000	+	0.000
	CHEB01	-1.0283	± 2.9066E-02	-	0.000	+	0.000

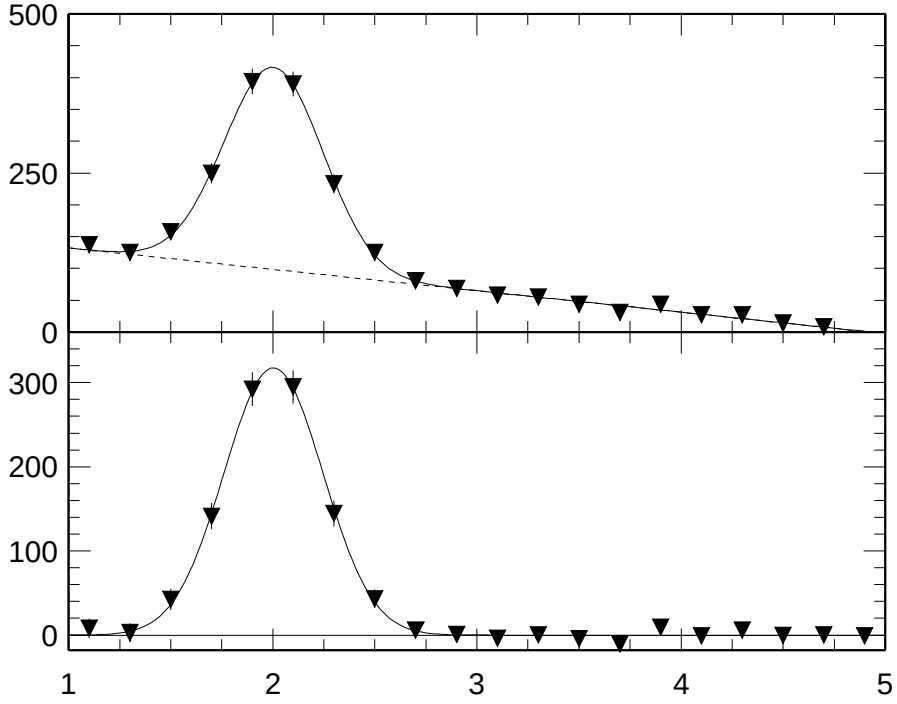


Figure C.5: Demo 5 — Simultaneous fit of 2 plots with function overlays.

C.6 Demo 6: Drawing Possibilities

```
MN_CMD> exec demo06
 Reading commands from unit  11
 File: demo06.mnf
MN_CMD> !----------------------------------------------------------------------
MN_CMD> ! Mn_Fit Demonstration file number 6
MN_CMD> ! Example of drawing - Picture of a Bhabha in colour
MN_CMD> !----------------------------------------------------------------------
MN_CMD> !
MN_CMD> set
SET> Give variable name or ?: default
SET> Give variable name or ?: x psize 20
SET> Give variable name or ?: y psize 18
SET> Give variable name or ?: thick 4
SET> Give variable name or ?: endset
MN_CMD> clear
MN_CMD> !
MN_CMD> !----------------------------------------------------------------------
MN_CMD> ! Schematic of a Bhabha Event
MN_CMD> !----------------------------------------------------------------------
MN_CMD> !
MN_CMD> ! Incoming particle
MN_CMD> !
MN_CMD> draw arrow -1 black = = -4
DRAW> Point 1 Give x,y: 2 4
DRAW> Point 2 Give x,y: 10 4
MN_CMD> draw arrow -1 black = = -4
DRAW> Point 1 Give x,y: 18 4
DRAW> Point 2 Give x,y: 10 4
MN_CMD> comment
COMMENT> Give command or ?: new 'e^-!'  6 3.4 0.4 = c
COMMENT> Give command or ?: new 'e^+!'  14 3.4 0.4 = c
COMMENT> Give command or ?: end
MN_CMD> !
MN_CMD> ! Detector
MN_CMD> !
MN_CMD> draw line 1 3
DRAW> Point 1 Give x,y: 16 4.5
DRAW> Point 2 Give x,y: 16 7.0
MN_CMD> draw line 1 3
DRAW> Point 1 Give x,y: 16 3.5
DRAW> Point 2 Give x,y: 16 1.0
MN_CMD> draw line 1 3
DRAW> Point 1 Give x,y: 4 4.3
DRAW> Point 2 Give x,y: 4 7.2
MN_CMD> draw line 1 3
DRAW> Point 1 Give x,y: 4 3.7
DRAW> Point 2 Give x,y: 4 0.8
```

MN_CMD> *!*
MN_CMD> *! Minimum and maximum angles*
MN_CMD> *!*
MN_CMD> draw line 2 6
DRAW> Point 1 Give x,y: 10 4
DRAW> Point 2 Give x,y: 16 4.5
MN_CMD> draw line 2 6
DRAW> Point 1 Give x,y: 10 4
DRAW> Point 2 Give x,y: 16 7.0
MN_CMD> draw arc 1 6
DRAW> Give centre: 10 4
DRAW> Give radii: 1 1
DRAW> Give angles: 0 4.76
MN_CMD> draw arc 1 6
DRAW> Give centre: 10 4
DRAW> Give radii: 1.5 1.5
DRAW> Give angles: 0 26.57
MN_CMD> comment
COMMENT> Give command or ?: new '[q]?min!' 10.6 3.5 0.4 = c = = 6
COMMENT> Give command or ?: new '[q]?max!' 11.1 4.9 0.4 = c = = 6
COMMENT> Give command or ?: *!*
MN_CMD> *! Bhabha*
MN_CMD> *!*
MN_CMD> draw arrow 1 2 = = -1
DRAW> Point 1 Give x,y: 10 4
DRAW> Point 2 Give x,y: 17 6.0
MN_CMD> draw arrow 1 2 = = -1
DRAW> Point 1 Give x,y: 10 4
DRAW> Point 2 Give x,y: 3 2.0
MN_CMD> draw arc 1 2
DRAW> Give centre: 10 4
DRAW> Give radii: 2 2
DRAW> Give angles: 0 15.95
MN_CMD> *!*
MN_CMD> draw box 1 yellow 1 cm 0 100
DRAW> Point 1 Give x,y: 6.0 6.9
DRAW> Point 2 Give x,y: 14.0 7.6
MN_CMD> comment
COMMENT> Give command or ?: new 'A Bhabha Event' 10 7 0.5 = c
COMMENT> Give command or ?: new 'e^-!' 17 6.1 0.4 = c = = 2
COMMENT> Give command or ?: new 'e^+!' 3 2.1 0.4 = c = = 2
COMMENT> Give command or ?: new '[q]' 11.8 3.5 0.4 = l = = 2
COMMENT> Give command or ?: end
MN_CMD> *!*
MN_CMD> *! Extra text*
MN_CMD> *!*
MN_CMD> comment
COMMENT> Give command or ?: new 'Tight' 17 4.7 0.4 = c
COMMENT> Give command or ?: new 'Cut' 17 4.1

COMMENT> Give command or ?: new 'Loose' 3 4.7
COMMENT> Give command or ?: new 'Cut' 3 4.1
COMMENT> Give command or ?: new 'Luminosity' 14.5 1.6 0.4 = r = = 3
COMMENT> Give command or ?: new 'Monitor' 14.5 1.0 0.4
COMMENT> Give command or ?: end
MN_CMD> draw arrow 1 3
DRAW> Point 1 Give x,y: 14.6 1.5
DRAW> Point 2 Give x,y: 15.9 1.8
MN_CMD> *!*
MN_CMD> *!————————————————————————-*
MN_CMD> *! Detector tilt and offset*
MN_CMD> *!————————————————————————-*
MN_CMD> *!*
MN_CMD> draw arrow 1 1
DRAW> Point 1 Give x,y: 4.0,13
DRAW> Point 2 Give x,y: 16.0,13
MN_CMD> draw arrow 1 1
DRAW> Point 1 Give x,y: 10.0, 9.1
DRAW> Point 2 Give x,y: 10.0,16.9
MN_CMD> draw arc 1 2
DRAW> Give centre: 11.0 13.5
DRAW> Give radii: 1 3
DRAW> Give angles: -80 100
MN_CMD> draw line 1 4
DRAW> Point 1 Give x,y: 11.0 13
DRAW> Point 2 Give x,y: 11.0 13.5
MN_CMD> draw line 1 4
DRAW> Point 1 Give x,y: 10.0 13.5
DRAW> Point 2 Give x,y: 11.0 13.5
MN_CMD> draw line 1 4
DRAW> Point 1 Give x,y: 11.2 12.5
DRAW> Point 2 Give x,y: 11.2 9.9
MN_CMD> draw line 1 4
DRAW> Point 1 Give x,y: 11.2 12.5
DRAW> Point 2 Give x,y: 11.6 9.9
MN_CMD> draw box 1 yellow 1 cm 0 100
DRAW> Point 1 Give x,y: 6.0 17.1
DRAW> Point 2 Give x,y: 14.0 17.8
MN_CMD> comment
COMMENT> Give command or ?: new 'Detector Tilt and Offset' 10 17.2 0.5 = c
COMMENT> Give command or ?: new '[Df]' 11.4 9.2 0.6 = centre = = 4
COMMENT> Give command or ?: new '[D]x' 10.6 12.4 0.6 = centre = = 4
COMMENT> Give command or ?: new '[D]y' 9.5 13.3 0.6 = centre = = 4
COMMENT> Give command or ?: new 'x' 15.5 12.3 0.6 = = = = 1
COMMENT> Give command or ?: new 'y' 9.5 16.5 0.6 = = = = 1
COMMENT> Give command or ?: end
MN_CMD> *!*
MN_CMD> draw box 1
DRAW> Point 1 Give x,y: 2.5 8.9

```
DRAW> Point 2 Give x,y: 17.5 17.1
MN_CMD> !
MN_CMD>
MN_CMD>
MN_CMD> !Make a hardcopy file
MN_CMD> exec $MN_FIT_SYS/help/demohard 06
 Reading commands from unit  12
 File: /home/brock/mn_fit/Linux/help/demohard.mnf
MN_CMD> !
MN_CMD> ! Macro to make a Postscript hardcopy
MN_CMD> !
MN_CMD> inquire 1 'Give demonstration number'
MN_CMD> set hard demo06.ps      !Set the hardcopy filename
MN_CMD> hardcopy post           !Make a Postscript hardcopy
 *** MN_FIL: File /home/brock/mn_fit/Linux/help/demo06.ps will be overwritten
 TVCAP: Hardcopy to unit 13, File: /home/brock/mn_fit/Linux/help/demo06.ps
 Postscript selected
 TVRNG: Device Postscript  Scaling factor 0.900 will be used
 None selected
MN_CMD> close                   !Close the hardcopy file
 End of Macro demohard.mnf, Unit= 12. Exiting
MN_CMD>
MN_CMD>
 End of Macro demo06.mnf, Unit= 11. Exiting
```

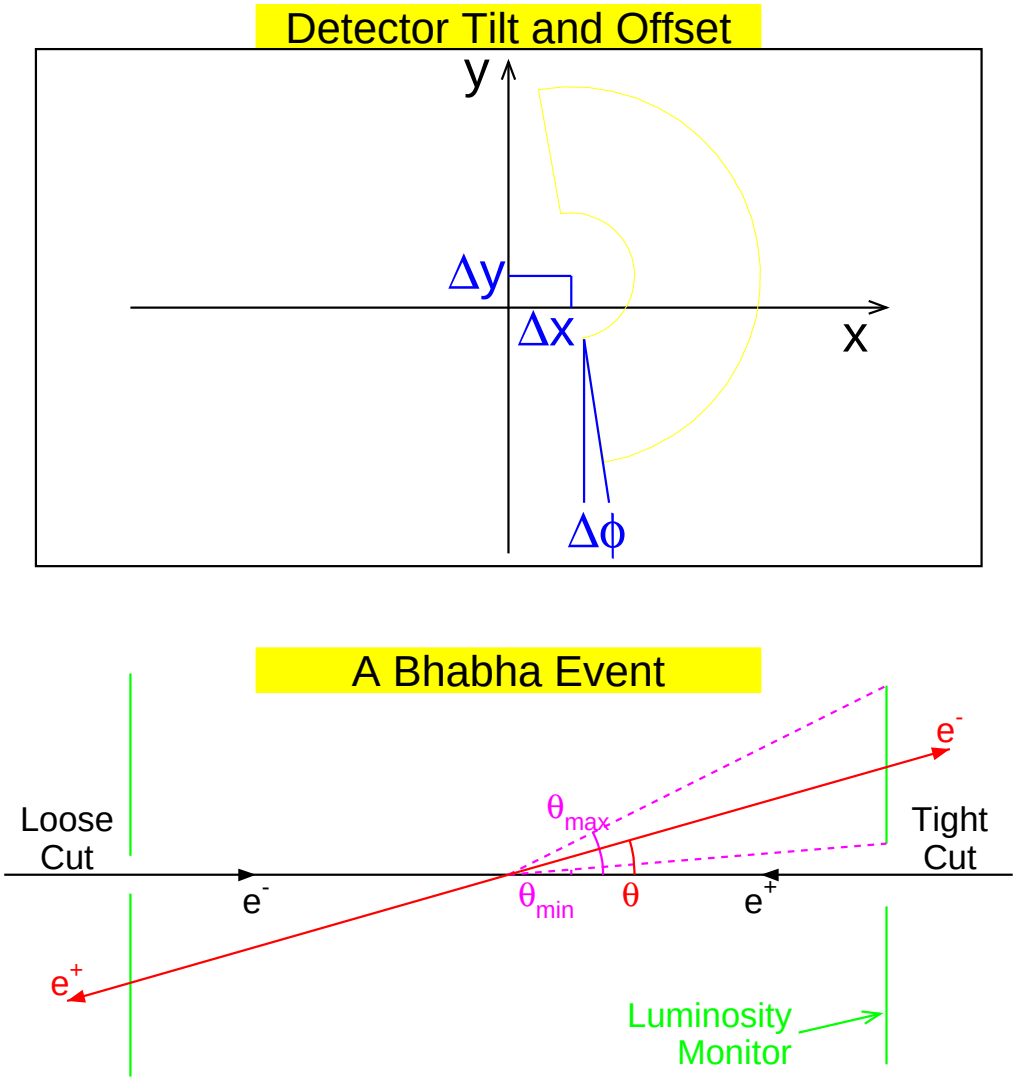


Figure C.6: Demo 6 — Drawing Possibilities.

C.7 Demo 7: Display of an L3 Luminosity Monitor Event

```
MN_CMD> exec demo07
 Reading commands from unit  13
 File: demo07.mnf
MN_CMD> !------------------------------------------------------------------------
MN_CMD> ! Mn_Fit Demonstration file number 7
MN_CMD> ! L3 Luminosity Monitor Event display with SLUM overlayed on BGO
MN_CMD> !------------------------------------------------------------------------
MN_CMD> set
SET> Give variable name or ?: default
SET> Give variable name or ?: head off
SET> Give variable name or ?: x psize 18
SET> Give variable name or ?: y psize 25.5
SET> Give variable name or ?: x size 17
SET> Give variable name or ?: x marg 0.5
SET> Give variable name or ?: y size 24
SET> Give variable name or ?: y marg 0.5
SET> Give variable name or ?: thickness 3
SET> Give variable name or ?: ! frame all off
SET> Give variable name or ?: x label ' ' = = = = = -1013
SET> Give variable name or ?: title pos = = = = = -1013
SET> Give variable name or ?: endset
MN_CMD> !
MN_CMD> dat_fetch $MN_FIT/test/slum_evt.mnd
 Reading histograms from unit   4
 File: /hp/cmu2a/user/brock/mn_fit/test/slum_evt.mnd
 Plot   1012   0  Data for   608 points read in
 Plot   6012   0  Data for    10 points read in
 MN_CDF: End of file reading in data.
 MN_CDF: A total of     2 plots have been read in
MN_CMD> !
MN_CMD> title 1012 'Bhabha Event in the L3 Luminosity Monitor'
MN_CMD> title 6012 ' '
MN_CMD> dep r1 = 1012
 r1 =   1012.00
MN_CMD> dep r2 = r1 + 5000
 r2 =   6012.00
MN_CMD> !
MN_CMD> ! Draw the Lumi Monitor BGO
MN_CMD> !
MN_CMD> set par fbgo 4 0 0 0 50
MN_CMD> set colour sym 2
MN_CMD> set col frame cyan
MN_CMD> display fbgo r1
MN_CMD> !
MN_CMD> ! Overlay the 3 layers of the SLUM
MN_CMD> !
MN_CMD> set colour sym blue
```

```
MN_CMD> set col frame green
MN_CMD> do i=1,3
MN_CMD> set par fsil 1 0 1 0
MN_CMD> display/noclear fsil r2
MN_CMD> enddo
MN_CMD> set par fsil 1 0 2 0
MN_CMD> display/noclear fsil r2
MN_CMD> enddo
MN_CMD> set par fsil 1 0 3 0
MN_CMD> display/noclear fsil r2
MN_CMD> enddo
MN_CMD> !
MN_CMD> !Make a hardcopy file
MN_CMD> exec $MN_FIT/help/demohard 07
 Reading commands from unit  14
 File: /hp/cmu2a/user/brock/mn_fit/help/demohard.mnf
MN_CMD> !
MN_CMD> ! Macro to make a Postscript hardcopy
MN_CMD> !
MN_CMD> inquire 1 'Give demonstration number'
MN_CMD> set hard demo07.ps      !Set the hardcopy filename
MN_CMD> hardcopy post           !Make a Postscript hardcopy
 *** MN_FIL: File /hp/cmu2a/user/brock/mn_fit/help/demo07.ps will be overwritten
 TVCAP: Hardcopy to unit 15, File: /hp/cmu2a/user/brock/mn_fit/help/demo07.ps
 Postscript selected
 Number of colours   8
 DISPLAY selected
MN_CMD> close                   !Close the hardcopy file
 End of Macro demohard.mnf, Unit= 14. Exiting
 End of Macro demo07.mnf, Unit= 13. Exiting
```

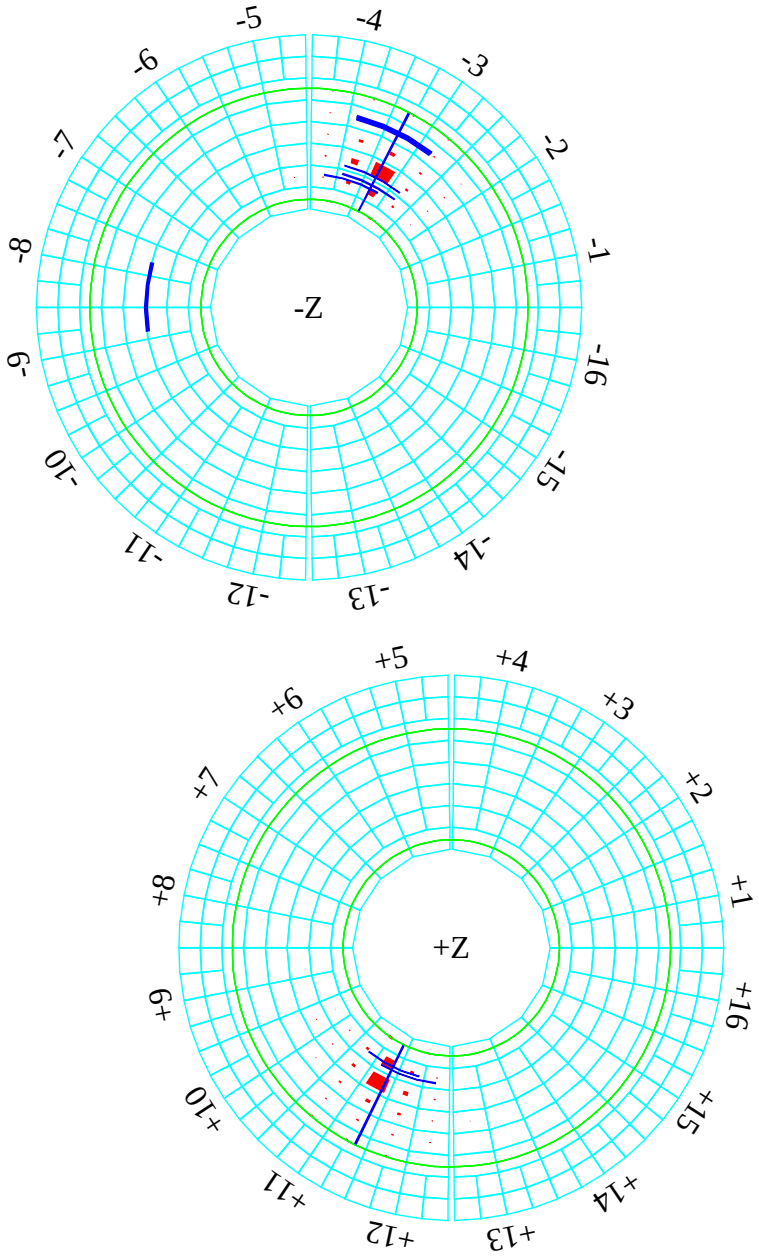


Figure C.7: Demo 7 — Display of an L3 Luminosity Monitor Event.

Appendix D

List of Changes

The following subtopics list the changes made for the version indicated. The top name Vn_mm used to correspond to the CMZ version number Vn.mm/ll which is given when you start up Mn_Fit. Since version 5_00 CVS is used to maintain Mn_Fit and the number refers to the CVS tag. Note that all changes before 01/12/89 are under topic V2_00_00. The manual only includes changes made in version 4.01 and later.

D.1 Version 5.13 07/11/2005

The following version numbers apply to this version of Mn_Fit:

```
Mn_Fit      v5_13
ZEUS_DSP    1.00/05
```

This section lists the changes that have been made since Mn_Fit version 5_06. Check `HELP CHANGES V5_06` for a list of changes made in version 5_06.

This release has been tested with the 2002, 2003 and 2004 releases of the CERN libraries. It has also been linked to the 98 version for CLEO. Depending on your version of gcc you may only be able to link with one of the versions. Versions earlier than 98 are known to have date/time problems.

Known Bugs or Problems:

- The text size when you use `SET HIGZ TABLE TEXT` or `SET HIGZ TABLE CHAR` followed by `IGTABL` is sometimes wrong. If this is a problem you can use `SET SYMBOL -1` or `SET SYMBOL -2` followed by `PLOT` instead.
- `DRAW` commands with the mouse on only work in CM.
- You cannot give a range of array elements to use in a `CUT` command. This only works in the variables to be projected. If you need to do something fancy like this you have to use a COMIS function for the cut.
- You can only access a variable in a CWN using the `DEPOSIT/EXAMINE` commands. It is not possible to access array elements this way.
- COMIS has occasional problems on OSF1 machines (Cernlib 98).
- The ZEUS forward detector display has not yet been integrated into this version.

New Features:

- ROOT interface is reasonably complete. Separate the name and number of a ROOT histogram with `;` rather than `:`.
- Commands `MADD` and `MSUBTRACT` are available to add or subtract a series of histograms from each other.
- You can specify whether the full range or the plot limits should be used to calculate the area under a function using the `SET FUNCTION AREA` command.
- You can try to recover Ntuples where the job creating them fail - use the `HRECOVER` command.
- The fit status and estimated distance to minimum are stored in registers 113 and 114.
- An array of character variables is now available `DEP CHAR...` to set them, `SHOW CHAR...` to look at them and {`CHAR(i)`} to use them - see section 1.12 on page 18 (Text) for more details. You can also set histogram titles and Ntuple variable names with the `DEPOSIT` command.
- Cernlib version 2002 is now the default, but another version may be used depending on the location.

New Commands:

FUNCTION ADD RARGUS Adds a reverse ARGUS function - useful for phase space turn on.

HRECOVER Tries to recover an Ntuple from a failed job.

LS Alias for `LDIRECTORY`.

MADD Adds a series of histograms together.

MSUBTRACT Subtracts a series of histograms from another histogram.

PWD Prints the current HBOOK or ROOT directory.

SET FUNCTION AREA Use the plot limits or the full range to calculate the area under a function.

SHOW CHAR Prints the one or more character array elements.

Modified Commands:

ROOT_FETCH Fetches 3-D histograms, profile plots and variable bin width histograms properly.

DEPOSIT Can now set character array, histogram title and Ntuple variable names

EXAMINE Can also look at character array elements, histogram title and Ntuple variable names and character values.

Numbers The fit status and EDM are in registers 113 and 114

SHOW CHAR Shows one or more character array elements.

SHOW CHARACTER Shows the comment and continuation characters.

Improvements and Bug Fixes:

- Cernlib version 2002 is now the default.
- Complete ROOT interface is available.
- The character used to separate the name from the number of a ROOT histogram was changed from `:` to `;` to avoid problems when specifying a range.
- 2-D histogram size is checked before trying to use IGTABL. Font size was not set correctly in the `2DIM TEXT` and `2DIM TABLE` commands.
- It is not possible to abbreviate the the `SHOW CHARACTER` command beyond `SHOW CHARA`, as `SHOW CHAR` is used to show character array elements. `EXAMINE` can now also be used for the various character variables: title, variable name, character variable value and the character array.
- Test macros adapted to use a single directory for all output files. You are prompted for the directory with `all_test`. `init_test` assumes that it is the current working directory.

D.2 Version 5.06 16/05/2005

The following version numbers apply to this version of Mn_Fit:

```
Mn_Fit      v5_06
ZEUS_DSP    1.00/05
```

This section lists the changes that have been made since Mn_Fit version 4.07/36. Check `HELP CHANGES V4_07` for a list of changes made in version 4.07, some of which you may have missed if you have been running a version earlier than 4.07/36.

This release has been tested with the 2002, 2003 and 2004 releases of the CERN libraries. It has also been linked to the 98 version for CLEO. Depending on your version of gcc you may only be able to link with one of the versions. Versions earlier than 98 are known to have date/time problems.

Known Bugs or Problems:

- `DRAW` commands with the mouse on only work in CM.
- You cannot give a range of array elements to use in a `CUT` command. This only works in the variables to be projected. If you need to do something fancy like this you have to use a COMIS function for the cut.
- You can only access a variable in a CWN using the `DEPOSIT/EXAMINE` commands. It is not possible to access array elements this way.
- COMIS has occasional problems on OSF1 machines (Cernlib 98).
- The ZEUS forward detector display has not yet been integrated into this version.

New Features:

- Mn_Fit has moved to cpp and make. The VMS version is frozen.
- A first interface to ROOT histograms is available. This requires Mn_Fit to have been compiled with this option turned on and for you to have the root libraries available. At present you can only fetch 1-D and 2-D histograms - profile plots will be available soon. Variable bin-width histograms are also not recognised properly yet. You must give each histogram identifier (0 and a range of histograms do not yet work).
- Cernlib version 2002 is now the default, but another version may be used depending on the location.

New Commands:

ROOT_OPEN	Opens a ROOT file.
ROOT_FETCH	Reads in ROOT histograms.
FUNCTION ADD RCB	Adds a reversed CB lineshape function.

Modified Commands:

NTUPLE MERGE	Internal changes to avoid segmentation violations.
SET FIT AREA	Can also calculate area for ARGUS background function.

Improvements and Bug Fixes:

- Cernlib version 2002 is now the default.
- First version of a ROOT interface is available.
- You can now use `SET FIT AREA ON` to also normalise the ARGUS background function.
- A new function `rCB` has a Crystal Ball lineshape, with an upper rather than a lower energy tail. Useful for beam-constrained D and B mass peak to include effect of initial state radiation, even though the parametrisation is Ad Hoc.
- Small bug in `SET FIT AREA` fixed. If you turned it off and then on again, and no parameters changed, e.g. in `FUNCTION PLOT` the integration was not redone.
- `SHOW FILE` now shows names of current HBOOK and ROOT files.
- Can now fetch up to 1000 plots from an ASCII file without losing track of which were fetched.
- The Linux version has been made and tested most using make. Linux, OSF1 and SunOS versions have been compiled for CLEO using the 98 version of the CERN libraries and seem to work fine.
- Many improvements to Makefile.
- Add Makefile.user in order to make a user version of Mn_Fit.
- Some fixes to include files and a few source files to remove C type comments that came from conversion from CAR files to fpp files.

- The font was not set correctly for 2-D plots using the `COL/Z` option.
- `NTUPLE MERGE` uses separate HBOOK top-level directory names. This will hopefully avoid segmentation violations - does not seem to do the trick all the time.

D.3 Version 4.07 02/12/2002

The following version numbers apply to this version of Mn_Fit:

```
Mn_Fit      4.07/33
Mn_Util     1.12/33
TYPSCN      1.02/18
CGR_MINUIT  1.00/15
DBASE       1.05/06
L3_DSP      1.02/16
ZEUS_DSP    1.00/05
```

This section lists the changes that have been made since Mn_Fit version 4.06/14. Check `HELP CHANGES V4_06` for a list of changes made in version 4.06, some of which you may have missed if you have been running a version earlier than 4.06/14.

This release has been tested with the 99 and 2000 releases of the CERN libraries. Versions earlier than 98 are known to have date/time problems.

Known Bugs or Problems:

- `DRAW` commands with the mouse on only work in CM.
- You cannnot give a range of array elements to use in a `CUT` command. This only works in the variables to be projected. If you need to do something fancy like this you have to use a COMIS function for the cut.
- You can only access a variable in a CWN using the `DEPOSIT/EXAMINE` commands. It is not possible to access array elements this way.

New Features:

- Cernlib version 2000 is now the default.
- This version of Mn_Fit is Y2K compatible.
- CWN variable names of up to 32 characters are now supported.
- `CUT` expressions can now be up to 255 characters long.
- The colour of symbols, hatchs and patterns can be given with the `PLOT` command and for the `FUNCTION PLOT` and `OVERLAY` commands. You can also define your own colour names for up to 20 different colours.
- `DRAW SEGMENT` draws the segment of a circle using the endpoints and sagitta (useful in Feynman graphs etc.). `SET IGARC` turns on/off using HIGZ IGARC routine for circles, as it appears to have a bug that cause it to ignore what you set for colour and thickness.

- Font -1004 is default for all sites and platforms. This means that the text stays as it was on the screen, but uses Helvetica Poscript fonts when you print the pictures - see section 1.12 on page 18 (Text) for more information.
- The underflows and overflows for the default plot are now also stored in registers >230. 200 user variable names are now available.
- You can use `..` instead of backslash to go up 1 HBOOK directory level. Multiple `..` will not work. HBOOK directory names can contain spaces. Enclose the name in quotes.
- You can use the syntax `SET DUMP SCREEN` to direct dump output to the screen and `SET DUMP > [filename]` to direct dump output to a file (alternatives to `TTY` and `LPT`, as these terms are no longer in common use).
- The command `HMERGE` has been introduced to allow the merging of CWN's and complete RZ files.
- You can set the paper size using the `SET PAPER` command.
- The `HELP` text (and therefore also the manual) has been proofread and many corrections and improvements have been included.

New Commands:

DRAW SEGMENT Draws a circle segment using endpoints and sagitta.

HMERGE Merge one or more HBOOK RZ files into a single file.

HPRINT Calls HPRINT for an HBOOK histogram.

TEXT Same as `COMMENT` command without specifying the plot identifier (planned, but not yet implemented).

SET COLOUR DEFAULT Set the colours to the default HIGZ colour scheme.

SET COLOUR BACKGROUND Give a background colour for plots.

SET DUMP SCREEN Alias for `SET DUMP TTY`.

SET DUMP > Alias for `SET DUMP LPT` with optional filename.

SET IGARC Uses HIGZ IGARC routine for circles or Mn_Fit code.

SET PAPER Set the paper size.

Modified Commands:

CUT Expressions can now be up to 255 characters long.

PLOT Allow colour to be given with the symbol, hatch and pattern.

OVERLAY Allow colour to be given with the symbol, hatch and pattern.

FUNCTION PLOT Allow colour to be given with the symbol.

FUNCTION OVERLAY Allow colour to be given with the symbol.

SET COLOUR ON Turns back on colour, but does not change the colour map.

SET PLOT id DEFAULT Also fills registers with underflows and overflows.

Improvements and Bug Fixes:

- Cernlib version 2000 is now the default.
- A serious bug was found when combining simple cuts (numbers or registers used for cut values) with more complicated expressions. The cut values could be overwritten - see section 4.19 on page 59 (CUT) for more details. This has been fixed in version 4.07/30. Thanks to Ahren Sadoff for spotting this and providing an example.
- You can given the symbol, hatch and pattern colours with the `PLOT` and `OVERLAY` commands. You can give the symbol colour with the `FUNCTION PLOT` and `FUNCTION OVERLAY` commands.
- Colour handling has been improved. You can name your additional colours and up to a total of 20 colours are allowed. From version 4.07/29 onwards 50 colours are allowed.
- Polygons can have up to 200 points (used to be 50). You can draw up to 2000 items now. Gluon drawing in plot coordinates should now be correct (the endpoint was drawn in slightly the wrong place.
- Drawing of first/last items should now be much improved. There were bugs in the old handling of items that should be drawn first in `PLOT` coordinates.
- User variable lengths of 8 characters are properly enforced and a warning is given if you try to make one that is longer.
- As a result of the above, the function `DATE_TIME` has been renamed to be `DATE_TIM`.
- Added MNSTAT to the list of known routines, so that one can find out how many parameters are being fit etc in COMIS routines.
- Along with the support for variable names of up to 32 characters in CWNs, the checking of the variable name has been improved, so that abbreviations do not work anymore.
- Length of many character variables increased to cope with long filenames etc. HBOOK subdirectories can be up to 16 characters. This is an RZ restriction.
- The `SET PLOT id DEFAULT` commands now also fills registers with underflows and overflows. Before version 4.07/20 it also overwrote registers 1 to 27. `SET HIST` did not recognise correclty that a command line was finished.
- The number of user variables names has been increased to 200.
- HBOOK and Mn_Fit commons increased in size for CLEO.
- Opening of an RZ filename with uppercase letters and the wrong record length should now work.
- It should now be possible to open and close many HBOOK RZ files. The command `HCLOSE` should close all files properly. It was not clear if it was really doing its job before.
- You can use variable numbers instead of the name for CWNs in the `NTUPLE PROJECT` command. However, this only works for single variables and not arrays, so is not recommended.
- Dumping of row-wise Ntuples gave screwed up numbers.

- Up to 100 subdirectories are shown with the `LDIR` command. Previous limit was 20.
- Help for polynomial function corrected. Help for ARGUS background function also corrected.
- Spurious lines drawn in overlays should no longer happen.
- Internal names of GETLUN routines have CLEO_ added as a prefix to avoid conflicts with ADAMO.

D.4 Version 4.06 06/04/2000

The following version numbers apply to this version of Mn_Fit:

```
Mn_Fit      4.06/14
Mn_Util     1.11/14
TYPSCN      1.02/16
CGR_MINUIT  1.00/15
DBASE       1.05/06
L3_DSP      1.02/16
ZEUS_DSP    1.00/05
```

This section lists the changes that have been made since Mn_Fit version 4.05/34.

This release has been tested with the 99 and 2000 releases of the CERN libraries. Versions earlier than 98 are known to have date/time problems. It will only work with versions 94b or later.

Known Bugs or Problems:

- `DRAW` commands with the mouse on only work in CM.
- You cannnot give a range of array elements to use in a `CUT` command. This only works in the variables to be projected. If you need to do something fancy like this you have to use a COMIS function for the cut.
- You can only access a variable in a CWN using the `DEPOSIT/EXAMINE` commands.

New Features:

- Cernlib version 99 is now the default. Version 2000 is recommended.
- This version of Mn_Fit should be Y2K compatible.
- Font -1004 is default for all sites and platforms. This means that the text stays as it was on the screen, but uses Helvetica Poscript fonts when you print the pictures - see section 1.12 on page 18 (Text) for more information.
- The axis labels are now aligned with the left (bottom), centre or right (top) of the plot as well as being appropriately adjusted. This makes it easy to make them the same as default PAW position.
- Symbols 50-79 added which put a small line (symbol size) at the end of an error bar.
- The ZEUS display for FTD has been improved significantly and new options have been added.

New Commands:

SET TKTCL	Turns on/off the TK/TCL interface.
HELP Y2K	Explains how years from 2000 onwards are implemented.

Modified Commands:

DATABASE DB_SNAP	Can now handle years from 2000 onwards.
DATABASE DB_HISTORY	Can now handle years from 2000 onwards.
SET X|Y|Z LABEL	Alignment option implemented fully.
SET SYMBOL	New symbols added to put cross-bars on errors.
SET PAR FTD	More options added.

Improvements and Bug Fixes:

- Cernlib version 99 is now the default. 2000 is recommended.
- This version of Mn_Fit should be Y2K compatible.
- The axis labels are now aligned with the left (bottom), centre or right (top) of the plot as well as being appropriately adjusted. This makes it easy to make them the same as default PAW position.
- Symbols 50-79 added which put a small line (symbol size) at the end of an error bar.
- Trimming of the last number on the y-axis was done whatever the window spacing.
- `CUT LIST` crashed if a cut filename was longer than 55 characters.
- Added TOPDRAW smoothing routines (SMCTRL) to the list of routines known by COMIS.
- Select options structured better. New experiments DESYAFS and CERNAFS introduced. @ removed from experiment name. BONNCIP added instead of CIP being an extra option. L3 and ZEUS displays can be explicitly selected instead of being implicit in ZEUS and L3. The flags L3DSP and ZEUSDSP in the Mn_Fit code have been renamed to L3DSP_SRC and ZEUSDSP_SRC, as L3DSP and ZEUSDSP are used in the scripts.
- Several bug fixes to allow Mn_Fit to be compiled and run with bounds checking and debug.
- L3 database interface did not work with 99, 2000 versions of the CERN libraries. Call to MZDIV now added, I/O is now with CIO and some other minor fixes.
- A bug in the making of the test histograms meant that the test script for `DAT_STORE` also had to be corrected.

D.5 Version 4.05 16/06/99

The following version numbers apply to this version of Mn_Fit:

```
Mn_Fit      4.05/34
Mn_Util     1.10/42
TYPSCN      1.02/15
CGR_MINUIT  1.00/15
DBASE       1.05/04
L3_DSP      1.02/16
ZEUS_DSP    1.00/04
```

This section lists the changes that have been made since Mn_Fit version 4.04/15.

This release has been tested with the 96a and 97a releases of the CERN libraries. It will only work with versions 94b or later. With version 96a and earlier you cannot use the routines FUNLUX and FUNLXP in COMIS functions, as these are new routines in the CERN libraries.

Known Bugs or Problems:

- `DRAW` commands with the mouse on only work in CM.
- You cannnot give a range of array elements to use in a `CUT` command. This only works in the variables to be projected. If you need to do something fancy like this you have to use a COMIS function for the cut.
- You can only access a variable in a CWN using the `DEPOSIT/EXAMINE` commands.

New Features:

- Cernlib version 97a is now the default.
- You can access histograms directly from an HBOOK histogram file provided you have already opened the file using the `OPEN` or `HB_OPEN` command. You can use the command `SET AUTOFETCH OFF` to suppress the automatic fetching of histograms or refetching of Ntuples when you project or scan them. While this is normally not necessary, it can take a long time for very big Ntuples.
- `CUT`s with names can now be defined and the names can be used in the `CUT USE` command. This should simplify the writing of macros and defining which cuts to use.
- All Mn_Fit functions now include the bin width. This was always the case for Gaussians, Breit-Wigners etc. However polynomials did not include the bin width. This gives problems when trying to integrate the functions. Where appropriate, the names of `NORM` parameters have been changed to `AREA` to reflect this.
- The `FUNCTION PLOT` and `FUNCTION HISTOGRAM` etc. commands now also work for 2-D functions (only Gaussian is available directly - for other 2-D functions you have to write them yourself).
- It is now possible to specify more resonances for the dipion invariant mass spectra (`FUNCTION ADD DIPION`). It is also possible for the user to give his/her own masses. More help has been added on a number of the functions. For functions that include extra parameters, such as resonance masses, these are now shown in the `FUNCTION INFO` command. Some function names have been modified, so it is now possible to add all functions by name as well as number. In order to add a 2-D Gaussian you now have to give the command `FUNCTION ADD 2DIM GAUSS` (`FUNCTION ADD 2DIM` used to be sufficient). See section 4.48.9 on page 91 (FUNCTION LIST) for the latest list of functions.

- ELIF is now allowed in IF blocks. See section 4.66 on page 120 (IF) for more details. You can also use the command FI instead of ENDIF. In earlier 4.05 versions of Mn_Fit ELIF and ELSE do not work together properly. This has been fixed from version 4.05/29 onwards.
- You can give the default value of a parameter after the prompt in the INQUIRE command.
- You can add directories to the path using the form SET PATH +dir.
- CWN variables can now be accessed in expressions. However, this does not yet work for arrays. DEPOSIT IRn works.
- German characters with umlauts can now be given with the syntax ä German characters with umlauts can now be given with the syntax
"a etc. Extra information has been added to HELP Text.
- The global/user title size and positions can now be set separately from the normal histogram titles. See section 4.106.88 on page 177 (SET TITLE) and section 4.106.35 on page 156 (SET GSIZE) for more details. The global/user title size is not automatically scaled when windowing. The automatic scaling of sizes set a maximum rather than a minimum size. The title offset is now also scaled.
- The making of an inserted plot with PLOT/NOCLEAR only worked if you were windowing at the same time. It is now possible to use it also when not windowing. This also enables you to put a frame around a plot in a different place than the plot boundaries set by SET X|Y LIMITS. See section 4.59.10 on page 116 (HISTOGRAM PLOT) for more details.
- 2-D histogram plotting has ben improved. The AUTOTRIM option is no longer used in lego and surface plots, as the scales do not overlap. The labels and scales should now get drawn in the right place and at the right angle for phi rotations outside the 0 to 90 degree range.
- It is now possible to make filled colour contour plots using the 2DIM SURFACE command. See section 4.3.8 on page 50 (2DIM SURFACE) for more details. The colours for coloured surface and lego plots can now be specified on the command line. Contour and surface plots usually start in the centre of the first bin rather than at the edge of the bin. With the help of one of the examples I show how one can put the frame at the centre of the bin (thanks to Jim Smith for the suggestion).
- SET Z MODE LOG also works for 2-D histograms which are shown with the area proportional to the number of entries.
- CLEAR also deletes any existing comments and keys. With the option SET BOX ON a box will also be drawn around a picture that does not include any PLOT commands.
- You can use SET NTUPLE NAME to specify the names of Ntuple variables to be used if you book an Ntuple inside Mn_Fit with the HISTOGRAM BOOK commmand.
- The SET FOOTER USER command has been extended to allow the user to put in the date, time and hardcopy filename. You can also split the footer between the bottom left and bottom right of the page.
- If you create many histograms in a COMIS function you can now give HB_MN_FIT a range of identifiers to convert. The maximum number of histograms that can be converted has been increased from 100 to 200.

- The new CERNLIB random number generators have been added to the list of routines that can be called from COMIS functions. These include `RANLUX, RANMAR, RANECU, RNORML, FUNLUX`. See section 1.16 on page 24 (Using COMIS) for more details on which routines are available.
- You can now use the secondary identifier to specify which cycle to fetch from HBOOK RZ files. The cycle number will be added to the current setting for the secondary identifier. This is useful if you store a series of Mn_Fit histogram which have the same primary identifier, but different secondary identifiers, as they then get stored in an HBOOK RZ file with the same identifier, but different cycles. Use the `ZDIR` command to see which cycle numbers exist.
- It may be that you do not want to have to tell Mn_Fit to continue if there is an error when executing a macro or defined command. Use the `SET ABORT` command to set this option.
- You now have to type at least `QUI` to quit Mn_Fit.
- Event display for ZEUS Forward Tracking Detector and Transition Radiation Detector added.

New Commands:

OPEN Synonymous with `HB_OPEN`.

CUT NAME Defines or modifies a `CUT` with a name.

ELIF `ELIF` can now be used in `IF` blocks.

FI Alias for `ENDIF`.

SET TITLE GPOSITION Sets the position and size of the global/user title.

SET GSIZE Sets the size of the global/user title.

SET AUTOFETCH Turns on/off the automatic fetching of histograms or refetching of Ntuples.

SET NTUPLE NAME Sets the names of Ntuples variables that are then used with the `HISTOGRAM BOOK` command.

SET ABORT Turns on/off the aborting of macros when an error occurs.

Modified Commands:

CUT USE Either CUT names or numbers can be used.

HB_MN_FIT You can optionally specify which histograms to convert.

FETCH You can use the secondary identifier to specify the cycle to fetch.

HB_FETCH You can use the secondary identifier to specify the cycle to fetch.

2DIM LEGO Colours can be given on the command line.

2DIM SURFACE Colours can be given on the command line. Can be used to make a filled contour plot.

PLOT/NOCLEAR The window size and margin (`WSIZE` and `WMARGIN`) can be used even when not windowing.

FUNCTION PLOT Works for 2-D functions also.

FUNCTION HIST	Works for 2-D functions also.
INQUIRE	Default values for parameters can be given.
WAIT	Works for Unix machines also.
DEPOSIT	Expression can now access CWN variables.
CLEAR	Also deletes any existing comments and keys.
SET PATH	Directories can be added to the path.
SET FOOTER	More control over user format, including hardcopy filename.
QUIT	You must now type at least `QUI` to quit Mn_Fit.
Functions	All Mn_Fit functions now include the bin width.

Improvements and Bug Fixes:

- Cernlib version 97a is now the default. Linux version now available.
- `NTUPLE EPROF` made a profile plot with the wrong errors.
- `SET NTUPLE VARIABLE` should work properly for CWNs. Use of a CWN variable as a weight works. Ntuple skeleton for logicals in CWNs fixed. Ntuple skeleton was wrong if it had more than 30 lines. The limit is now 80 lines.
- CWNs with only a one or two variables or arrays should now be correctly recognised.
- The trigonometric and Crystal ball lineshape functions were missing RETURNs. This caused unpredictable behaviour.
- Improved scales when the scale includes zero.
- Small improvements to `DUMP`. HPRINT removed from `DUMP` as it causes crashes on the ZEUS FDET DQM history Ntuple (also in PAW). There are still some strange problems that appear to be associated with `DUMP` and CWN's that have to be investigated more.
- `CAPTURE DISPLAY` caused a segmentation violation if the DISPLAY was :0.0. For other reasons it also caused a segmentation violation on Decstations.
- There were some problems when combining `IF` blocks and `DO` loops. These should now be fixed.
- The modification of the `higz_windows.dat` file that controls the size of your display now conforms to the HIGZ specifications again.
- Improved cleaning up of Ntuples before reading a new file. Caused strange problems on Alphas and Decstations.
- The automatic scaling of sizes set a maximum rather than a minimum size. The title offset is now also scaled.
- Up to 100 cuts are now allowed (used to be 40).
- Up to 5000 histograms can now be stored in Mn_Fit memory for Unix machines for CLEO and 2000 for others/
- There was an error when opening the 10th EPS file.

- The mean and RMS for scatter plots was not stored in the right place, so always appeared to be zero. The mean and average calculation was skipped for 4.05 versions earlier than 4.05/21, if you had already fetched a Columnwise Ntuple.
- Space for integers and reals in CWNs increased to 50000 words.
- `HB_MN_FIT` can now fetch up to 200 histograms at once (used to be 100).
- You can now plot 200 histograms and still `HARDCOPY` or `REDRAW` them.
- The maximum length for macro arguments was 40 characters. This has been increased to 80.
- The length of commands in `DEFINE` blocks has been increased to 255 characters (including continuation lines).
- Using FORTRAN `INQUIRE` to find out if a file exists did not always work on Decstations (Alphas also?). Changed to use `ACCESSF` from the CERN libraries for Unix machines. Should mean that the Mn_Fit help files no longer need to be world writeable. Some further checking on whether `INQUIRE` returned something sensible added.
- One too few characters was passed to shell command (only appears to have affected Linux).
- There was a bug in the setting of options for 3-D plots.
- The z scale size on `2DIM COL/Z` plots can now be set properly.
- A number of local variables that should be saved were not saved (only affected machines for which the static option was not default or not set - probably only Linux).
- The font for 2-D plots for symbols -1 and -2 was not set properly.
- The COMIS user function number was not initialised to zero properly. Too many histogram operations required entries in a histogram before the operation would be performed.
- If the same histogram is plotted twice the header will only be printed once.
- Better error handling when accessing CWN variables in expressions.
- A <`CR`> after an `ALIAS` command was not properly protected.
- CWN only read in once when projecting. It used to be read in twice. A number of bugs connected with CWN's not being initialized properly fixed.
- The error message about 'Run out of room to store plots for hardcopy' will only be printed once, and also only printed if the active device is a screen device.
- The HELP subtopics were sometimes not listed if the last subtopic was a level lower than those that were being listed.
- STRATEGY command did not work.
- The font setting for individual plots for scales and labels did not work properly. Some more information added to SHOW LABEL and SCALE.
- Error returns from shell commands are now caught. The correct shell was not used for single command.

- The Cernlib functions GAMMA and DGAMMA have been added to the list of functions that can be used in COMIS.
- Bug when adding a dipion invariant mass function, could cause a segmentation violation.
- The server number can be set in the VMS version (-SERVER).
- The flag @CERN has been replaced by @CERNLIB as it is only used to say if the cernlib command is available.
- SOLARIS and Linux added as machine types. @ZEUS, @CIP added as flags. ZEUSDSP added as flag. $L3_ONL renamed as L3DSP. Cleanup/fixing of the libraries needed for Solaris.
- Some modifcations had to be made to the flags for compiling under IRIX 6.2 and system IRIX64. IRIX verison should now also be able to use cc.

D.6 Version 4.04 12/06/96

The following version numbers apply to this version of Mn_Fit:

```
Mn_Fit      4.04/15
Mn_Util     1.09/24
TYPSCN      1.02/12
CGR_MINUIT  1.00/15
DBASE       1.05/04
L3_DSP      1.02/14
```

This section lists the changes that have been made since Mn_Fit version 4.03/49.

This release has been tested with the 95a release of the CERN libraries. It will only work with versions 94b or later.

Known Bugs or Problems:

- `DRAW` commands with the mouse on only work in CM.
- The `IGTABL` scale is not always drawn on the right axes, or exactly in the right place if rotations of more than 90 degrees are used.

New Features:

- Plotting with a date or time as axis labels is now implemented.
- New functions for expressions - DATE, TIME, TIME_MIN, DATE_TIME and DATE_MIN. These are mainly to be used for plots vs date or time and can be used to convert dates in the form YYMMDD etc. See section 1.17 on page 26 (Time) and HELP Expressions for more details.
- New help topics on time and vectors added.
- You can now `DRAW` in plot coordinates, even if you have not plotted anything! To do this you must set the x and y limits. See section 4.33 on page 73 (DRAW) for more details.

- The functions which evaluate functions at a particular point, FPOS, FNEG, etc. have been tested and now work properly. See section 1.8 on page 14 (Expressions) for more details.
- A cut can have an expression on both sides of the condition.
- `IF` statements allow arithmetical expressions.
- Projections of 1-D histograms now work. This is useful if you want to calculate an expression of the x-axis variable, e.g. cos(x).
- 2-D histogram plotting has many new options easily available and documented. All IGTABLE options are now built into Mn_Fit. The command `2DIM` can be used as a simple interface, or the old method using `SET IGTABLE` followed by `IGTABLE` also works. `LEGO` and `SURFACE` plots, including `IGTABLE`, work properly with log scales.
- A list of directory names, similar to the Unix PATH, can now be given using the `SET PATH` command. These directories are used when trying to open existing files.
- A footer can be added to a picture, either with default text that gives the date and time that the plot was made, or with user text.
- The key for symbols 5->8 now works correctly and the line segment size can also be set. Hatched histogram symbols are also available.
- The automatic rescaling of text sizes when windowing, the dropping of the last scale value when windowing and in lego plots can now be controlled. See section 4.106.5 on page 146 (SET AUTOSWITCH) and section 4.106.6 on page 146 (SET AUTOTRIM) for more details.
- You can plot the χ^2 vs. a variable even if you are doing a likelihood fit using the `CHI_PLOT` command. Otherwise the command is the same as `FCN_PLOT`.
- The `COPY` and `RENAME` commands copy the associated HBOOK histogram, only if it is not an Nutple.
- The `HELP` and `HISTOGRAM|NTUPLE DUMP` commands now invoke a pager by default. This means the help will not scroll off your screen. See section 4.106.62 on page 163 (SET PAGER) for more details on how to control the paging.
- New options have been added to the ECAL display. It is now possible to draw single boxes, rescale the picture easily, overlay displays and make a display directly from an Ntuples. See section 4.106.64 on page 164 (SET PARAMETER ECAL) for more details.

New Commands:

HISTOGRAM 2DIM	2-D histogram plotting using IGTABL.
2DIM	2-D histogram plotting using IGTABL.
CHI_PLOT	Plot the χ^2 vs. a variable.
SET IGTABLE	Alias/replacement for SET HIGZ TABLE.
SET FOOTER	Turns on or off footer and user text.
SET FSIZE	Change the text size for the footer.

SET AUTOSCALE Turns on or off the auotmatic resclaing of text sizes when windowing.

SET AUTOTRIM Turns on or off the automatic dropping of the last scale value when windowing or in lego plots.

SET COLOR Alias for SET COLOUR for Americans!

SET PAGER Sets the paging command used by `HELP` and `DUMP`.

SET PATH Sets list of directories used in file search..

SHOW FRAME Alias for `SHOW SCALE`.

SHOW LIMIT Alias for `SHOW SCALE`.

SHOW MODE Alias for `SHOW SCALE`.

SIGN Now allowed in expressions.

Modified Commands:

LEGO Qualifiers allowed for IGTABL options.

SURFACE Qualifiers allowed for IGTABL options.

KEY Hatched histogram symbols now possible.

KEY Line segment size and the text colour can now be set.

DEPOSIT Date/time functions added.

SET HIGZ TABLE Some options renamed and better prompts added.

SET MODE Date and time modes added.

SET PARAMETER ECAL New options and mode for L3 ECAL display.

FCN_DRAW The parameter name or number can now be given

FCN_PLOT The parameter name or number can now be given

PROB_PLOT The parameter name or number can now be given

Improvements and Bug Fixes:

- Cernlib version 95a is now the default.
- Bug in calculation for `AVERAGE` command fixed.
- Filled and unfilled symbols for 2-D histograms did not work as advertised. If both of the limits were not zero, the entries outside the range still got drawn.
- `SHOW FRAME`, `SHOW LIMIT` and `SHOW MODE` commands added as aliases for `SHOW SCALE`, because `SHOW SCALE` really shows all 4 settings.
- Evaluation of function with FPOS, FNEG, etc. functions now works.
- If you were projecting a histogram with an expression things could get confused and you got error messages from `AMNE`.
- Dumping of Ntuples improved further.
- The contents of a series of points are filled properly, including underflows and overflows if they are outside the set limits.

- The default spline and smooth options for NAGLIB and IMSL were not set properly.
- Histograms with no entries should now get drawn completely straight away.
- Copying and renaming of projections of Ntuples should work properly. The associated HBOOK histograms no longer get copied or renamed if the histogram is an Ntuple.
- If you tried to fetch a range of HBOOK histograms the range had be given as the last of a list of histogram numbers. This is now fixed.
- Associated functions etc. of HBOOK histgorams get the filename attached. The filename of projections is that of the original histogram with a * prepended.
- Either the parameter number or name can be given for the `FCN_DRAW`, `FCN_PLOT` and `PROB_PLOT` commands.
- Plot coordinates work properly for L3 Lumi BGO and SLUM displays. Commands for font setting modified slightly.
- Plotting of L3 Lumi BGO sectors 15 -> 1 did not show any energy in sector 1.

D.7 Version 4.03 15/09/95

The following version numbers apply to this version of Mn_Fit:

```
Mn_Fit      4.03/49
Mn_Util     1.08/60
TYPSCN      1.02/12
CGR_MINUIT  1.00/15
DBASE       1.05/03
L3_DSP      1.02/09
```

This section lists the changes that have been made since Mn_Fit version 4.02/19.

This release has been tested with the 94a, 94b and 95a releases of the CERN libraries. However the CWN interface only works with 94b or later.

Known Bugs or Problems:

- `DRAW` commands with the mouse on only work in CM.
- The `IGTABL` scale is not always drawn on the right axes, or exactly in the right place if rotations of more than 90 degrees are used.
- Naglib smoothing and spline fitting is not available on Apollo DN10000's.
- There is an error in the calculation of the `AVERAGE` command.

New Features:

- ColumnWise Ntuples are now supported. `NTUPLE FILTER` and `MERGE` do not work yet with CWN's. Suggestions on improvements to the CWN interface are very welcome. See section 1.15 on page 24 (ColumnWise Ntuples) and NTUPLE for more details.

- If you need to pass the contents of registers, parameters etc. to another macro or defined command you can use the `PARSE` command. See section 4.89 on page 136 (PARSE) for more details and an example.

- Plots in radians can now be displayed with the scale in fractions of pi.

- Continuation lines are allowed. The maximum line length is a total of 255 characters. The comment and continuation line characters can be set. Note that comments are now removed immediately after reading a command line. If you have to include a comment character in a command (e.g. an `!` in a text string), the text string must be in single or double quotes.

- `STAT` command does an RZSTAT on the current input file.

- You can set the limits for a scatter plot when projecting from an Ntuple in the command line.

- Character information, such as Ntuple title and tags, is now available in COMIS functions.

- Gluon lines can be drawn. Amplitude and period of sine waves and gluons can be given.

- Polyline and ellipse drawing is available.

- `SET NTUPLE` command has new options and a modified syntax. You have to use `SET NTUPLE PLOT` to specify Ntuple variables to plot.

- You can show underflows and overflows in a plot with the new `SET HEADER COMPLETE` command. `SET HEADER FULL` has been renamed to `SET HEADER DEFAULT` (although `FULL` still works).

- `DUMMY` is allowed as a variable name for Ntuples read in with `DAT_FETCH` allowing columns to be ignored.

- COMIS memory, defined subroutines and common blocks can be listed using the `SHOW COMIS` command. FORTRAN units that have been opened or reserved can be listed using the `SHOW UNITS` command.

- The segment length for drawing lines can now be set (SET LSIZE).

- For the database interface options to plot the results and/or to subtract pedestals can be given. Database history the the L3 ECAL is available. Various parameters, such as the year, can be set for database access. For database history limits on acceptable values for including in the average can be set.

- In database history you can now give a negative 2nd time to ask for number of days, months or years that you want to see before the first time.

- L3 ECAL display can cope with histograms of half and whole detector. New plotting modes are available.

- Some installation flags have changed - should only affect installers.

New Commands:

PARSE Parses and executes a command line.

STAT Gives the statistics on the current HBOOK input file.

DRAW ELLIPSE Draws an ellipse.

DRAW POLYLINE Draws a polyline.

SET PI Turns on or off using pi as a scale symbol.

SET NTUPLE PLOT List of variables to plot (was `SET NTUPLE`).

SET NTUPLE VARIABLE List of variables to fetch for a CWN.

SET CHARACTER Sets the command and continuation line characters.

SET DBASE Sets parameters for database access.

SET FONT SYMBOL Sets the font for symbols.

SET LSIZE Sets the segment length for lines.

SHOW COMIS Lists COMIS memory, routines and common blocks.

SHOW UNITS Lists FORTRAN units and who has grabbed them.

MOD Can be used in expressions.

Modified Commands:

DRAW GLUON Really draws a gluon now.

DRAW SINE Amplitude and period can be given.

DUMP Points to dump can be given.

HISTOGRAM DUMP Points to dump can be given.

HISTOGRAM ERROR New mode which allows error on all points to be set.

COMMENT Command must always be given.

KEY Command must always be given.

SET HEADER New option, `COMPLETE`, `DEFAULT` is new name for `FULL`.

SET NTUPLE Options added.

NTUPLE PROJECT Setting limits for a scatter plot works.

DATABASE Qualifiers added and new options.

SET PAR ECAL New options.

Improvements and Bug Fixes:

- Cernlib version 94b is now the default.
- The X11 version of Mn_Fit is the default for all platforms. Any other version can be selected at install time.
- Readline is now the default for Unix versions. Readline can now be taken from /usr/local/lib or wherever it happens to be.
- For plots in degrees the scale should be improved.

- `NTUPLE MERGE` was always putting zeroes for the 1st file.
- Fetching of variable bin width histograms had a bug.
- Fetching of big Ntuples was not working sometimes.
- Standard Chebyshev, with overall normalization, did not give reproducible results.
- The Landau function now includes the bin width.
- Setting of x and y limits when projecting onto a scatter plot now works. If you project an Ntuple onto a 1-dimensional histogram, it will default to automatic binning.
- An expression for the projection of an Ntuple can now be longer than 80 characters. The full expression for a cut is shown if you use the `CUT LIST` command.
- You must always give the command for `COMMENT` and `KEY`. If you were adding a comment or key interactively and no comment or key existed, then the command `NEW` used to be assumed. Now you have to give the command.
- `DRAW SYMBOL` drew a symbol a factor of 2 smaller than the size asked for.
- Better checking is made of whether a file atually exists. This should reduce the number of inadvertently overwritten files with COMIS functions. Filenames can be upto 80 characters.
- Dumping of Ntuples and listing of cuts improved.
- Expressions for projections can be longer than 80 characters.
- COMIS functions gave intermittent problems on Alpha VMS. Fixed by using a special load option. However comis converts all filenames to lowercase, so you cannot have uppercase characters in your filename.
- `FIT/SLIKE` command caused a format mismatch error.
- Tick positions should be somewhat better for strange scales.
- Adding of a leading 0 to scales was not correct.
- Points at the limit of a plot were not drawn. This meant that bins with no entries were not drawn, even if the `SHOW_ZERO` option was on.
- Underflows and overflows for plots of dimension -1 (a series of data points) are now kept. This is mainly useful for variable binned histograms.
- If you asked for a 2nd help topic after the 1st topic had no sub-topics you used to get an error. This is now fixed.
- Comments are now stripped immediately after the command is read. Thus if you need a comment character in a command it must be in single or double quotes. Improved checking inside loops. A `RETURN` inside a `DO` loop that was not being executed was not ignored.
- Help files opened with readonly if possible on non-VMS machines. They should now be accessible by everyone, without having to give write permission.
- Length of shell string increased (/usr/local/bin/bash should now be OK).

- Number of drawn items can be up to 100 and is now checked.
- User and global titles should no longer be overwritten by the header.
- Use of backslash to quote an `!` did not work properly on some machines - Decstations, IBM RS6000. This has been fixed in TYPSCN.
- `FUNCTION STORE` did not free its unit, leading eventually to a crash of Mn_Fit if you gave this command many times. If you tried to `EXECUTE` a non-existing macro the unit number was not released. The units used for `DRAW STORE` and `DRAW FETCH` had the same problem.
- Error return when adding COMIS functions improved.
- VAX Help replaced by VMS Help.
- Lego y label positioning was controlled by the x label.
- The file for the C. Rippich version of MINUIT, minuit.cmz (or .car), has been renamed to cgr_minuit.cmz to avoid confusion with the CERN library version of MINUIT.
- Installation of Mn_Fit can now be done using patchy instead of cmz.
- A new flag VMS has been added. ALPHAOSF and ALPHAVMS have been removed. AIX370 has been replaced by IBMAIX. IBMRT has been added as a flag, but is not yet in install. All flags for OS and MACHINE should now be the same as for the CERN libraries.
- The XGKS graphics option has been completely dropped.
- VAXGKS has been replaced by DECGKS.
- RS6000 installation and readline fixed.
- Some fixups for CLEO, expecially for the ALPHA.

D.8 Version 4.02 25/07/94

The following version numbers apply to this version of Mn_Fit:

```
Mn_Fit  4.02/19
Mn_Util 1.07/14
MINUIT  1.00/13
TYPSCN  1.02/06
L3_DSP  1.02/02
DBASE   1.04/04
```

This section lists the changes that have been made since Mn_Fit version 4.01/21.

This release has been tested with the 93d and 94a releases of the CERN libraries.

Known Bugs or Problems:

- `DRAW` commands with the mouse on only work in CM.
- The `IGTABL` scale is not always drawn on the right axes, or exactly in the right place if rotations of more than 90 degrees are used.

- NTUPLE MERGE may fill the first files entries with 0 if you get an RZOUT not authorized message.
- Fetching of variable bin width histograms is wrong.
- Standard Chebyshev doe not give reproducible parameters.
- Naglib smoothing and spline fitting is not available on Apollo DN10000's.

New Features:

- The FIT command syntax has been improved to allow immediate specification of the fit type. It is also possible to likelihood fit taking into account the limited function statistics.
- A new function type Smoothed Histogram has been added. It has 2 parameters: the NORMALIZATION and the OFFSET and can also have multiply the ordinate by a SCALE factor.
- A directory name can be specified, which is then used for all files if you do not give a directory specification. However this means that filename completion does not work for such files. The command is WDIR or SET WORKING_DIR.
- Points with no entries and zero error only have the error replaced by 1 when doing operations involving more than 1 histogram (ADD, SUBTRACT, MULTIPLY, DIVIDE, EFFICIENCY, AVERAGE), provided that ERR_ZERO is .TRUE..
- New spline fitting and smoothing routines have been added. HBOOK routines HSMOOF and HSPLI1 can be called (SMOOTH/HBOOK and SPLINE/HBOOK commands). A smoothing routine from Topdrawer is available (SMOOTH/TOP) and Naglib spline fitting and smoothing (SPLINE/NAG and SMOOTH/NAG) are available.
- Smooth curves through a series of points can be drawn using the command PLOT/SMOOTH.
- The interface to IGTABL has been improved substantially. All options are now available and axis ticks, scales and labels are controlled in the normal Mn_Fit way. The positioning of the scale in 3-d plots (LEGO and SURFACE) still needs some improvement and the scale and label are sometimes drawn on the wrong axis if you rotate by more than 90 degrees. All options also work with windows.
- A global title is available (SET TITLE GLOBAL title).
- Interface to ECAL database and a new display for the L3 BGO added.

New Commands:

FUN ADD SMOOTH Smoothed histogram used as a function.

PLOT/SMOOTH Plots a histogram as a smooth curve.

PLOT/EMPTY Plots an empty histogram without giving an error.

SET BIN Specfies an offset or scale factor for histograms.

WDIRECTORY Sets the working directory.

SET WORKING_DIR Sets the workinh directory.

SET FUNCTION POINTS Specifies the number of points to use when drawing a function.

SET FUNCTION INTEGRATE Specifies the number of intervals to use when integrating a function.

SET FUNCTION STEP Specifies the step size to use when differentiating a function.

SET COLOUR ON|OFF Turns on or off use of colour.

SET TITLE GLOBAL Adds a global title to each picture.

DATABASE DBN_SNAP Database snapshot without a picture.

DATABASE DBN_HISTORY Database history without a picture.

Modified Commands:

FIT	Qualifiers added and new options.
HIST ERROR	New mode (-1) to delete errors, rather than just setting to 0 (mode 0).
SPLINE	HBOOK and Naglib spline fitting added.
SMOOTH	HBOOK, Naglib and Topdraw smoothing added.
IGTABLE	Interface much improved - HELP added.
SET HIGZ	Interface much improved.
DRAW	New option for whether items should be at front or back.
DB_SNAP	Differences between database entries can be plotted.
COVARIANCE	The CERN version shows the covariance matrix.
DATABASE DB_SNAP	Options to to differences between entries.

Improvements and Bug Fixes:

- You can now delete a list of functions.
- `DAT_STORE` can now store Ntuples.
- `HISTOGRAM ERROR id -1` deletes the errors on a histogram, so `DAT_STORE` will only store `X,Y` values.
- The spurious lines in `DISPLAY` when using the `INCLUDE` command are now fixed.
- `FIT_INFO` and `DISPLAY` sometimes gave floating point errors on the hp. This is fixed.
- Wrong frames when making lego or surface plots should be fixed.
- After a `HARDCOPY` command an xterm did not switch back from the Tektronix to the VT100 window properly.
- The return status from `lib$spawn` on VMS is printed.
- An attempt is made to unlock locked histogram files if they cannot be opened properly.
- `mn_fit filename` did not work on Unix machines.
- `INQUIRE` statements inside "Here Documents' caused problems with the readline version.
- The automatic switching between Ntuples on disk did not work properly.

- The HESSE command shows the covariance matrix if the PRINTOUT level is 0 or greater.
- There was no protection on defining more than the maximum number of cuts.
- If you aborted a macro inside an IF block things were not reset properly.
- If you try to draw more plots than the buffer, you get a warning message, but are not asked for a <CR>, nor is the window number reset.
- The macro name is also given when the file is closed.
- The PRECISION command did not show you the current precision properly.
- Protection added to histogram operations if I run out of space.
- ALPHAVMS, ALPHAOSF and ALPHA added as flags.
- The C. Rippich version of MINUIT should also work on HP's.

D.9 Version 4.01 17/12/93

The following version numbers apply to this version of Mn_Fit:

```
Mn_Fit  4.01/21
Mn_Util 1.06/25
MINUIT  1.00/11
TYPSCN  1.02/02
L3_DSP  1.01/00
DBASE   1.03/00
```

This section lists the changes that have been made since Mn_Fit version 3.03/05.

Known Bugs or Problems:

- SEEK gives underflows in some of my test fits on HP's.
- NTUPLE MERGE only allows you to put 1 Ntuple in the merged file.
- Ntuples will always be stored in the top level directory. Other plot types should be stored in the correct directories.
- SQUEEZE sometimes gives errors.
- As ! is used for command line history in Unix versions, you must quote the ! with a backslash if you want a ! interactively.
- This release works only with version 93 or later of the CERN libraries.
- If you use the mouse with the DECGKS version you get error message 51 from GINLC. This is not a problem, so you can ignore it.
- The SHOW command calls the corresponding SET command if the SHOW does not exist. If you are asked to set a value you wanted to show, just hit <CR>.

New Features:

- Command recall and history are available on all Unix machines using GNU readline and history routines. The `history` command is available on versions that include GNU readline.
- `DRAW` has `CIRCLE, ARC, SINE, SYMBOL` and `GLUON` added. It is possible to fetch and store items made with `DRAW`. `DRAW BOX` only needs 2 points. You can use `DRAW` and `COMMENT` after a clear, without plotting a histogram first.
- `CDIR` now changes the HBOOK directory immediately rather than waiting until the next `LDIR, ZDIR, FETCH` etc. command. This means that sequential `CDIR` commands will work. `SET DIRECTORY` works as before, so should be used before a `FETCH` command.
- `MIN, MAX` are allowed in expressions.
- `SET COLOUR` can now use the colour name and you can set the colour of the overlayed fit. The colour name can also be used in `DRAW`.
- Symbols 11,21,31,41 are now real circles instead of octogons. The octogons are symbols 19,29,39,49.
- Chebyshev and Legendres now have the overall normalization as the first parameter. The previous forms are called `OChebyshev` and `OLegendre`. A threshold function has been added.
- Interface to `IGTABL` and `IGSET` added for contour plots etc.

New Commands:

DRAW CIRCLE	Draws a circle
DRAW ARC	Draws an arc
DRAW SINE	Draws a sine function
DRAW GLUON	Command built in, but draws a straight line!
DRAW SYMBOL	Draws a symbol
DRAW STORE	Store drawn items
DRAW FETCH	Fetch drawn items
MIN	Can use in expressions e.g min(r1,r2)
MAX	Can use in expressions e.g max(r3,r4)
AVERAGE	Weighted average of the contents of 2 histograms
FUN ADD THRESHOLD	Threshold function
FUN ADD OCHEBYSHEV	Old form of Chebyshev
FUN ADD OLEGENDRE	Old form of Legendre
HISTOGRAM IGTABLE	Direct interface to the HIGZ IGTABL routine
IGTABLE	Direct interface to HIGZ IGTABL routine
SET HIGZ	Direct interface to HIGZ IGSET routine or IGTABL parameters
DUMP	Short form for HISTOGRAM DUMP
history	History of commands for Unix versions

Modified Commands:

CDIRECTORY Changes the HBOOK directory immediately

SET COLOUR You can use the colour name

FUN ADD CHEBYSHEV The first parameter is now the overall normalization

FUN ADD LEGENDRE The first parameter is now the overall normalization

Improvements and Bug Fixes:

- Fetch of a range of HBOOK histograms only reads in those that are needed. The histograms (and profile plots) are now fetched using hunpak and hunpke rather than hi and hie. This should be faster.
- `STORE` is much improved and can now handle series of points and scatter plots. They are stored as Ntuples.
- `DUMP` can be used instead of `HISTOGRAM DUMP` at the `MN_CMD>` level.
- Several fixes to improve `IF` statements.
- The users SHELL and EDITOR should now be picked up properly.
- Setting of HBOOK directories improved. You should no longer get error messages just before the creation of a new directory. After a `FETCH` the default directory will be the input file for commands such as `LDIR`.
- `MDIR` did not recognize correctly if a directory already existed.
- `FILL` put entries 1 bin below the lower limit in the first bin.
- A scale `-2.5, -1.25, 0, 1.25, 2.5` will now be written correctly and not with `1.25` as `1.3`.
- You now always get a 0 in front of the decimal point (was not the case on the HP before).
- `RETURN` statements inside `IF` blocks that are not being executed should now work OK. They used to always be executed.
- `DISPLAY` sometimes gave a floating point error on the HP. This is now fixed. `DISPLAY` and `FIT_INFO` also caused a divide by 0 in 1000 bin histograms.
- If you `HARDCOPY` and then try to plot another histogram while windowing the normalization transformation was not correct. This is fixed.
- Fetching profile plots sometimes caused a divide by 0. This was because I was fetching the errors on the underflows and overflows. The bug is fixed.
- If you had `SET X LIMIT` and then used hatches or patterns the results were not always correct.
- There was an error in the error calculation in the `AVERAGE` command.
- Initialization of the resonances for dipion mass functions just after adding the function was not correct.

- The Falco interface for DECGKS has been dropped as it is available in the X Windows version.
- All Unix scripts are now written in Bourne shell.
- The main Mn_Fit routines are now written in C and use the latest version of KUIP condition handling.
- All flags (except a few special L3 ones) have been renamed to remove the `$` at the front. This is because the latest version of cmz tries to interpret `$XXXX` as an environment variable, symbol or logical name.

Bibliography

[1] CERN Program Library. MINUIT – Function Minimization and Error Analysis. D506.

[2] CERN Program Library. COMIS – Compilation and Interpretation System. L210.

[3] CERN Program Library. HIGZ – High level Interface to Graphics and Zebra. Q120.

[4] I.J. Scott. *A Measurement of the Polarization τ Leptons in Z Decays with the L3 Detector at LEP*. PhD thesis, Harvard University, May 1993.

[5] R. Barlow and C. Beeston. Fitting using Finite Monte Carlo Samples. *Comp. Phys. Comm.*, 77:219, 1993.

Index

2DIM, **47**, 110, 115, 117–119, 125, 138, 158, 188
  ARROW, **48**
  BOX, **48**
  CHAR, **49**
  COLOUR, **49**
  CONTOUR, **49**
  Examples, **51**
  LEGO, **50**, 115
  SCATTER, **50**
  SURFACE, **50**, 118, 266
  TEXT, **51**

ADD, **53**
ALIAS, 2, **54**, 146, 181, 189
  Examples, **54**
ATTACH, **55**
AVE_FETCH, **55**
AVERAGE, **55**

BOOK, **55**

CALCULATE, **56**
CALL_COMIS, 25, **56**
CAPTURE, **56**
CDIRECTORY, **56**
CLEAR, **57**
CLOSE, **57**
ColumnWise Ntuples, **24**, 63, 128, 273
COMIS, 25, **57**
  Mn_Fit Functions in COMIS, **106**
Commands, **6**
COMMENT, **57**, 161
Condition Handler, **32**
convolute, 7
COPY, **58**
CUT, 1, 33, 37, **59**, 262
  CHANGE, **60**
  COMPILE, **61**
  DELETE, **61**
  EDIT, **61**
  END, **62**
  Examples, **62**
  FILE, **61**
  LIST, **61**
  Menu, 59
  NAME, **60**
  NEW, 59, **59**
  USE, **62**
CWN, **63**

DAT_FETCH, **63**, 141
  Examples, **64**
DAT_STORE, **65**, 190
DATABASE, **65**
  DB_HISTORY, **65**, 67
  DB_SNAP, **66**, 68
DB_HISTORY, **67**
DB_SNAP, **67**
DEFINE, **68**
  Examples, **68**
DELETE, **69**
DEPOSIT, 12, 56, **69**, 82, 114, 142, 147, 159, 183
  Examples, **71**
Dipion Inv Mass
  BW x Phase Space, **102**
  Full Yan Model, **102**
  Isovector Model, **102**
  Moxhay Resonance, **102**
  Novikov Model, **102**
  Peskin Model, **102**
  Phase Space, **101**
  Voloshin Model, **102**
  Yan Model, **101**
DIRECTORY, **72**
DISPLAY, **72**
DISPLAY environment variable, 3
DIVIDE, **72**
DO, **72**, 82, 83
  Examples, **73**
DRAW, 37, **73**, 270
  ARC, **74**

ARROW, **75**
BOX, **75**
CHANGE, **79**
CIRCLE, **75**
DELETE, **79**
ELLIPSE, **76**
END, **79**
Examples, **80**
FETCH, **79**
GLUON, **76**
LINE, **77**
LIST, **79**
Menu, 73
POLYGON, **77**
POLYLINE, **77**
SEGMENT, **78**
SINE, **78**
STORE, **79**
SYMBOL, **78**
TRIANGLE, **78**
DUMP, **80**

ECAL
Barrel, **165**
Examples, **166**
EDIT, **81**, 90, 105
EFFICIENCY, **81**
ELIF, **81**
ELSE, **82**
ENDDO, **82**
ENDIF, **82**
Errors, 8, 87
Including Monte Carlo, 10
MINUIT, 8
Small Numbers of Events, 9
Special Commands, 10
EXAMINE, **82**
Examples, **29**
EXECUTE, 11, 68, **82**, 141
Examples, **83**
EXIT, **47**
Expressions, **14**, 27, 59, 60, 69, 132, 193, 271
EXTRACT, **84**

FETCH, **84**, 109
Examples, **85**
FI, **86**
FILL, **86**
FIT, 2, **87**, 103
Examples, **89**
Likelihood, **88**
Fitting, **6**
FUNCTION, 34, 38, **89**
ADD, **89**
COMPILE, **90**
DELETE, **90**
EDIT, **90**
FETCH, **90**
HISTOGRAM, **91**
HOVERLAY, **91**
INFO, **91**
LIST, 2, 7, 89, 90, **91**, 265
2D Gaussian, **104**
ARGUS Background, **97**
Bifurcated Gaussian, **97**
Breit Wigner, **95**
CB Line Shape, **99**
Chebyshev, **93**
COMIS, 2, 8, 25, **104**, 107
Continuum Cross Section, **103**
Dipion Inv Mass, **101**
Exponential, **93**
Fermi, **100**
Fragmentation Functions, **102**
Gaussian, **94**
Hadron Spectra, **96**
Histogram, **103**
Landau, **94**
Legendre, **94**
Lepton Spectra, **95**
Lifetime, **100**
Pexp(Q), **98**
Polynomial, **92**
Power Law, **100**
Quadratic Joining Two Lines, **93**
rARGUS Background, **98**
rCB Line Shape, **99**
Resonance Cross Section, **103**
Smooth Histogram, **103**
Sum Two Bifurcated Gaussians, **97**
Threshold, **101**
Trigonometric, **99**
Two Gaussians, **96**
USER, 7
User, 2, 8, **106**
x-a Gaussian, **95**
Menu, 89
OVERLAY, **107**
PLOT, **108**

STORE, **108**, 201
USE, **108**
Function parameters, 13

HARDCOPY, **108**
Hardcopy Devices, **31**, 109
HB3_FETCH, **109**
HB_FETCH, **109**
HB_MN_FIT, **109**, 134
HB_OPEN, **110**
HB_STORE, **110**
HCLOSE, **110**
HCOPY, **110**
HDELETE, **110**
HINDEX, **110**
HISTOGRAM, 34, 38, **110**
2DIM, **119**
BOOK, 56, **111**
/ASYMMETRIC, **111**
/BINNED, **111**
/ERRORS, **111**
/NOERRORS, **111**
/UNBINNED, **111**
Examples, **111**
DISPLAY, 72, **112**
/CLEAR, **112**
/NOCLEAR, **112**
ECAL, **112**
FBGO, **112**
FSIL, 112, **112**
FTD, **113**
FWCH, 112, **113**
TRD, **113**
DUMP, 80, **113**
ERRORS, **114**
EXTRACT, 84, **114**
FILL, 86, 111, **114**
Examples, **114**
IGTABLE, **115**, 122
LEGO, **115**
Menu, 110
OVERLAY, **115**
/DIFFERENT, **116**
/NEXT, **116**
/NTUPLE, **116**
/SAME, **116**
/SMOOTH, **116**
PLOT, **116**, 266
/CLEAR, **117**
/EMPTY, **118**
/NEXT, **118**
/NOCLEAR, **118**
/NTUPLE, **118**
/SMOOTH, **118**
SURFACE, **118**
Histogram contents, 13
HISTORY, **119**
HMAKE, **119**
HMERGE, **119**
HRECOVER, **120**
HRENAME, **120**
HY_FETCH, **120**

Identifiers, 2, **16**
IF, 11, 81–83, 86, **120**, 266
Examples, **121**
expression, **121**
IGTABLE, **122**
INDEX, **122**
INQUIRE, 83, **122**
INTEGRATE, **123**

KEY, **123**, 161

LDIRECTORY, **124**
LEGO, **125**
limits, 199
LS, **125**

Macros, 10
MADD, **125**
MDIRECTORY, **125**
Examples, **125**
Menus, **33**
CUT, 37
DRAW, 37
FUNCTION, 38
HISTOGRAM, 38
MINUIT, 44
NTUPLE, 39
READ, 39
SET, 39
SHOW, 42
WRITE, 43
MERGE, **126**
MESSAGE, **126**
MINUIT, 2, 33, 88
BACK_SUB, **192**
CALLS, **192**

CHI_PLOT, **192**, 196, 200
CONSTRAIN, 2, 7, 181, **193**, 202
  Examples, **193**
CONTOUR, **194**
COVARIANCE, **194**
DISPLAY, 72, **194**
DUMP, **194**
END, **195**
ERROR_DEF, **195**
EXCLUDE, **195**
EXIT, **195**
FCN_DRAW, **195**
FCN_PLOT, 193, **196**, 200
FIT_INFO, **196**
FIX, **196**
FLOAT, **196**, 201
GRADIENT, **196**
HESSE, **197**
IMPROVE, **197**
INCLUDE, **197**
INFO, **197**
ITERATIONS, **197**
MATOUT, **198**
MAX_CALLS, **198**
MIGRAD, **198**
MINIMIZE, **198**
MINOS, **198**
MINUIT, **192**
MNCONTOUR, 193, 196, **199**, 200
MODIFY, 159, **199**
NO_EXCLUDE, **199**
NO_GRADIENT, **199**
NO_INCLUDE, **200**
NO_ITERATIONS, **200**
PAGE, **200**
PRECISION, **200**
PRINTOUT, **200**
PROB_PLOT, 193, 196, **200**
PUNCH, **201**
RELEASE, **201**
RESTORE, **201**
SAVE, **201**
SCAN, **201**
SEEK, **201**
SIMPLEX, **202**
STANDARD, **202**
STOP, **202**
STRATEGY, **202**
UNCONSTRAIN, **202**
  UNIT, **202**
MN_FETCH, **126**
MN_STORE, **126**
Mouse, **23**
MSUBTRACT, **127**
MULTIPLY, **127**

Newcomers, **4**
NO_CUT, **127**
NO_WINDOW, **128**
NORMALIZE, **128**
NTUPLE, 23, 25, 35, 39, 61, 63, **128**
  DUMP, **129**
  EPROFILE, **129**
  FILTER, **130**
  MERGE, 126, **130**
  PLOT, **130**
  PROJECT, 1, 24, 25, 59, 131, **131**, 140
    Examples, **132**
  SCAN, 25, **134**, 144
  SPROFILE, **134**
Numbers, 2, **11**, 19, 44, 69, 70, 73, 82, 114, 142, 145, 150, 157

OPEN, **134**
OVERLAY, **135**
  /DIFFERENT, **135**
  /NEXT, **136**
  /NTUPLE, **136**
  /SAME, **135**
  /SMOOTH, **136**

PARSE, 19, **136**, 274
  Examples, **136**
PARTITION, **137**
PLOT, **137**
  /CLEAR, **138**
  /EMPTY, **139**
  /NEXT, **139**
  /NOCLEAR, **138**
  /NTUPLE, **139**
  /SMOOTH, **139**
  Examples, **139**
PRINT, **140**
PROJECT, **140**
PWD, **140**

QUIT, **47**

READ, 36, 39, **141**

COMMAND, **141**
DATA, **141**
Menu, 141
readline, **6**
REBIN, **141**
Recalling Commands, 6
REDRAW, **141**, 172
registers, 12
REMOVE, **142**
RENAME, **142**
RETURN, **142**
Root, **28**
ROOT_FETCH, **142**
Examples, **143**
ROOT_OPEN, **144**
Running Mn_Fit, **2**

SCALE, **144**
SCAN, **144**
Screen Devices, **30**, 56
SCT_FETCH, **144**
Secondary Identifiers, 1, 16, **16**
SET, 1, 36, 39, **145**
ABORT, **145**
ALIAS, **146**
AUTOFETCH, **146**
AUTOSCALE, **146**
AUTOSWITCH, **146**, 271
AUTOTRIM, **146**, 271
AXIS, **146**
BACKGROUND, **147**
BIN, **147**
BOX, **147**
BREAK, **147**
CHARACTER, **147**
Examples, **148**
COLOR, **148**
COLOUR, **148**
DBASE, **149**
DEBUG, **149**
DEFAULT, **149**
DIRECTORY, **150**
DISPLAY, 145, **150**
MODE, 7, **150**
DSIZE, **150**
DUMP, **151**
ECHO, **151**
EDIT, **151**
ENDSET, **151**
ERR_ZERO, **151**
EXCLUSIONS, **151**
EXIT, **151**
FIT, 7, 40, **152**
AREA, **152**
COMMANDS, **152**
CONVOLUTE, 2, **152**
DEFAULT, **152**
INTEGRATE, **152**
FONT, **153**
FOOTER, **153**
FRAME, **154**
FSIZE, **154**
FUNCTION, 38, 40, **154**
AREA, **154**
BIN_WIDTH, **155**
INTEGRATE, **155**
Menu, 154
POINTS, **155**
STEP, **155**
GRID, **155**
GSIZE, **156**, 266
HARDCOPY, **156**
HATCH, 17, 116, 135, **156**, 174
HEADER, **157**
HIGZ, **157**
TABLE, 115, **157**
HISTOGRAM, 150, **157**
IDB, **157**
IDR, 142, **158**
IDSHOW, **158**
IDSIZE, **158**
IGARC, 75, **158**
IGTABLE, 157, **158**
LABEL, 147, **159**
LIMITS, **159**
LOG, **159**
LSIZE, **160**
MANUAL, **160**
MARGIN, **160**
MODE, 41, **160**
MOUSE, **161**
NEXT_WINDOW, **161**
NO_WINDOW, **179**
NORMALIZE, **161**
NTUPLE, **161**
NAME, **161**
PLOT, **162**
VARIABLE, **162**

NULL, **162**
OPT_ZERO, **162**
ORDER, 63, **162**
  Examples, **163**
ORTHOGONAL, **163**
PAGER, **163**, 271
PAPER, **163**
PARAMETER, **164**
  ECAL, 67, 112, **164**, 271
  FBGO, 112, **166**
  FSIL, 112, **167**
  FTD, 113, **168**
  FWCH, 113, **168**
  TRD, 113, **169**
PATH, 2, 83, **170**
PATTERN, 116, 135, **171**
PI, **171**
PLOT, 145, **171**
PSIZE, **171**
RATIO, **172**
RECL, **172**
REDRAW, **172**
ROOT_ID, **172**
ROTATION, **172**
SCALE, **172**
SECONDARY_ID, **173**
SHELL, **173**
SHOW_ZERO, **173**
SIGNAL, **173**
SIZE, **174**
SSIZE, **174**
STATISTICS, **174**
SYMBOL, 48, **174**
TEXT, **176**
THICKNESS, **176**
TICKS, **176**
  Examples, **177**
TIME, **177**
TITLE, **177**, 266
TKTCL, **178**
TSIZE, **178**
USIZE, **178**
WAIT_CR, **178**
WINDOW, 112, 117, 138, **178**, 189
  Examples, **179**
WMARGIN, **179**
WORKING_DIR, **180**
WSIZE, **180**
ZERO, 162, **180**
SHELL, **180**
shift, *see* XSHIFT
SHOW, 36, 42, **180**
    ALIAS, **181**
    ALL, **181**
    CHAR, **181**
    COMMANDS, **181**
    COMMENT, **181**
    CONSTRAINT, **181**
    CUTS, **181**
    DEFINITION, **182**
    DIRECTORY, **182**
    EXCLUSIONS, **182**
    FILES, **182**
    FLAGS, **182**
    FRAME, **182**
    INCLUSIONS, **182**
    KEYS, **182**
    LABEL, **182**
    LIMIT, **183**
    LOG, 11, **183**
    MODE, **183**
    ORDER, **183**
    PLOT, **183**
    REGISTER, **183**
    SCALE, **183**
    SEGMENTS, **184**
    SIZES, **184**
    TICK, **184**
    UNITS, **184**
    VARIABLE, **184**
SMOOTH, **184**
    /HBOOK, **185**
    /IMSL, **185**
    /NAGLIB, **185**
    /TOPDRAW, **185**
SPAWN, **185**
SPLINE, **186**
    /HBOOK, **186**
    /IMSL, **186**
    /NAGLIB, **186**
SQUEEZE, **187**
Startup files, 3
STAT, **187**
STORE, 110, **187**
    /NEW, **187**
    /UPDATE, **187**
SUBTRACT, **188**
SUM, **188**

SURFACE, **188**
Symbols, **17**

TEXT, 153
Text, 11, 12, **18**, 58, 82, 124, 126, 257, 261, 263
Time, **26**, 270
TITLE, **188**

UNALIAS, **189**
UNDEFINE, **189**
Using COMIS, 2, **24**, 57, 105, 267
Using Ntuples, **23**

Variables, *see* Numbers
Vectors, **28**

WAIT, **189**
WDIRECTORY, **189**
WINDOW, **189**
WRITE, 43, **189**
  DATA, **190**
  LOG, **190**

XSCALE, **190**
XSHIFT, **190**

Y2K, 26, **27**
YSCALE, **190**
YSHIFT, **191**

ZDIRECTORY, **191**
ZSCALE, **191**
ZSHIFT, **191**