



To go to the matNMR project page at source forge:



To go to the source forge homepage:



To go back to the matNMR page:



Search the manual pages directly

1 How to read this manual ?

1.1 Which parts should I read?

2 Installing matNMR

2.1 Installing matNMR

2.1.1 UNIX

2.1.2 MS Windows/MAC

2.2 Setting up matNMR (Options)

2.2.1 General options

2.2.2 Colour scheme settings

2.2.3 Screen settings

2.2.4 Font list

2.2.5 Line properties

2.2.6 Text properties

3 MATLAB and matNMR

3.1 MatNMR and the workspace

3.2 Input expressions

3.3 Important variables

3.4 Where is the NMR?

3.5 matNMR format for spectra

3.6 GUI and MATLAB

3.7 Memory usage

3.8 MATLAB and WYSIWYG

4 Processing in matNMR

4.1 Processing spectra with matNMR

4.2 1D, 2D and 3D mode

4.3 Using the processing history

4.4 Using macro's

4.5 Multi-window matNMR

4.6 Using short-keys

4.7 Apodizing FID's

4.8 Fourier transform

4.9 Phasing spectra

4.10 Baseline correction

5 The main window

- 5.1 General appearance
- 5.2 The menubar
 - 5.2.1 Stop matNMR
 - 5.2.2 Files
 - 5.2.3 1D Processing
 - 5.2.4 2D Processing
 - 5.2.5 Plot manipulations
 - 5.2.6 History/Macro
 - 5.2.7 Create Output
 - 5.2.8 Options
 - 5.2.9 Help
 - 5.2.10 Clear functions
- 5.3 Fitting of CSA tensors (MAS)
 - 5.3.1 General appearance
 - 5.3.2 Buttons
 - 5.3.3 Menubar
- 5.4 Fitting of CSA tensors (Static)
 - 5.4.1 General appearance
 - 5.4.2 Buttons
 - 5.4.3 Menubar
- 5.5 Fitting of Diffusion Curves
 - 5.5.1 General appearance
 - 5.5.2 Buttons
 - 5.5.3 Menubar
- 5.6 Peak Deconvolution
 - 5.6.1 General appearance
 - 5.6.2 Buttons
 - 5.6.3 Menubar
- 5.7 Fitting of Quadrupolar tensors
 - 5.7.1 General appearance
 - 5.7.2 Buttons
 - 5.7.3 Menubar
- 5.8 Fitting of Relaxation Curves
 - 5.8.1 General appearance
 - 5.8.2 Buttons
 - 5.8.3 Menubar

6 The 2D/3D Viewer

- 6.1 General appearance
- 6.2 The panel window
 - 6.2.1 Contours
 - 6.2.2 Mesh 3D
 - 6.2.3 Stack 3D
 - 6.2.4 Raster 2D
 - 6.2.5 Tools
- 6.3 The menubar
 - 6.3.1 Stop 2D/3D Viewer
 - 6.3.2 Plot manipulations
 - 6.3.3 History/Macro
 - 6.3.4 Printing Menu
 - 6.3.5 Copy figure
 - 6.3.6 Options
 - 6.3.7 Help
 - 6.3.8 Clear functions

7 Producing output

- 7.1 Printing menu
- 7.2 Saving figures to disk
- 7.3 Using the Clipboard

8 Script-based processing

[Search the manual pages directly](#)



Chapter 1 - How to read this manual ?

This manual was first written for matNMR 2.6 (October 2000) and describes all features and concept of it. Subsequent releases have been incorporated into the manual. This means that the version that can be found on the website belongs to the latest release of matNMR. I am well aware that this manual is quite limited. It does for example not have an index and it doesn't allow searching through it. If anyone feels up to improving it, then please do so and send me the result!

The manual is divided in several chapters focussed around some subjects. For all features the actions required to perform it and also the implementation are described (in short). The description of the implementation shall contain more information about what is done to a spectrum and how matNMR works. In most cases there will be no explanation of standard MATLAB commands or features; the user is referred to the MATLAB documentation then. Also no explanations will be given for typical NMR operations as these can be found relatively easy in the NMR literature.

In many occasions there will be spoken of "The main window" and the "2D/3D Viewer". These are the two main windows in matNMR. The first is used for processing while the second is used mainly for displaying and analysis. They can be started independently from the MATLAB command line by typing "nmr" or "nmr2d" respectively. More information about these windows is given in chapter 5, "[The main window](#)" and chapter 6, "[The 2D/3D Viewer](#)".

In the manual there will be made no clear distinction between an FID, a spectrum or a matrix. MATLAB does not care about the NMR behind a spectrum as it only knows matrices and vectors (for more information see paragraph 3.4, "[Where is the NMR?](#)"). Therefore whenever it says "FID" or "spectrum" in this manual it could just as well say "matrix" because in the end they're all the same thing.

1.1 Which parts of this manual should I definitely read ?

MatNMR can in principle be used as an independent processing program, just like any other package, making knowledge of MATLAB not really necessary. To use all its features to the fullest however one needs to know some things about it. These things are explained in chapter 3, "[matNMR and MATLAB](#)". There the interaction of matNMR and MATLAB is described in some detail. Even when you don't intend to do fancy stuff however I strongly advise you to read this chapter!

Also I advise you to read Chapter 4, "[Processing in matNMR](#)". This explains how processing in matNMR is thought to be done. Also things like using the processing history and macro's effectively are explained in more detail here.

The installation of matNMR is explained in chapter 2, "[Installing matNMR](#)". How to configure matNMR to your personal likings is described in paragraph 2.2, "[Setting up matNMR \(options\)](#)".

Finally for people who don't want to use the matNMR user interface Chapter 8 , "[Offline Processing with matNMR](#)" explains how to use the processing capabilities in user-defined scripts.



Chapter 2 - Installing matNMR

2.1 Installing matNMR

Installing matNMR only requires copying the files into a directory and setting the search path variable such that it can find it. There are however some small differences between UNIX systems and Windows/MAC systems. For both the preferred way of installation is described below.

2.1.1 UNIX

To install matNMR for a single user only on a UNIX system one must first extract the downloaded code into any directory. Here I will use '/home/jabe/matlab/matNMR' as an example. Then one must change or create the file ~/matlab/startup.m. Specifically, the PATH variable needs to be changed so that MATLAB looks in the directory in which you have just extracted the source code. This is a possible syntax to change the path:

```
path('/home/jabe/matlab/matNMR', path);
```

No further actions should be needed.

To install matNMR globally (i.e. for all users) on a UNIX system it is easiest to add a directory called matNMR to the matlabroot/toolbox directory (note that one needs the proper rights to do this, usually super user). Then extract the downloaded code into this directory. You then need to edit the matlabroot/toolbox/local/pathdef.m file. There you should add the matlabroot/toolbox/matNMR directory to the list of standard directories.

In the matNMR directory there is a file called matnmroptions.mat and this is the configuration file for matNMR. If users have not copied this file into their personal matlab directory ~user/matlab then these settings will be used by matNMR. Note that users usually don't have permission to overwrite this file and therefore not change their settings! For more information on this see paragraph 2.2 "[Setting up matNMR \(Options\)](#)".

2.1.2 Windows / MAC

To install matNMR on a Windows or Macintosh system it is easiest to add a directory called matNMR to the matlabroot\toolbox directory. Then extract the downloaded code into this directory. To make matNMR generally available to all users you need to edit the matlabroot\toolbox\local\pathdef.m file. There you should add the matlabroot\toolbox\matNMR to the list of standard directories.

In the matNMR directory there is a file called matnmroptions.mat and this is the configuration file for matNMR. As users usually do not have a user-specific matlab directory (as on UNIX) all users will use the same configuration file. Only if the system can be set up such that in the pathdef.m a user-specific matlab directory is specified, it can be avoided that all users use the same configuration file. If such a setup is present then users can copy the matnmroptions.mat into their personal matlab directory and have a personal configuration. Note that it is necessary to have the personal matlab directory before the (general) matNMR directory for this to work! How to change the settings is described below.

2.2 Setting up matNMR (Options)

Several user-defined settings can be stored in matNMR, e.g. screen size, UI font size, the fonts the matNMR recognizes, line properties, text properties etc. All these are stored in the file matnmroptions.mat. When matNMR is started it will determine the location of this file and subsequent saving of the settings will be done to this location.

To configure matNMR to your liking start up the main window with "nmr" from the MATLAB prompt. In the resulting window you will find the "Options" in the menubar. Clicking the mouse pointer on this menu will show a list of possible settings that can be changed:

- General Options
- Colour scheme settings
- Screen settings
- Font List
- Line properties
- Text properties
- Restore Defaults

In all of these menus changes of settings are applied either immediately or after pressing the "EXECUTE" button. Without saving these changes they only apply for this session. The next session will then use the default parameters again as saved in the matnmroptions.mat file. Selecting the "Restore Defaults" will change all settings back to their default values.

2.2.1 General Options



The general options menu is used to set some general working parameters. These include:

- whether to show a logo during startup (general)
- whether to ask for confirmation when closing a window (general)
- How many processing steps may be undone in 1D mode
- How many processing steps may be undone in 2D mode ([Beware of memory usage](#))
- whether to plot a x-scale by default (main window)
- whether to plot a y-scale by default (main window)
- which menus to show on startup (main window) - 1D / phase / 2D menus
- whether to use an absolute or a relative(0-1) y-scale by default (main window)
- whether to plot a grid by default (main window)
- whether to multiply the first point of the FID by 0.5 by default
- what type of FFT is used in TD2 by default (main window)
- what type of FFT is used in TD1 by default (main window)
- what default axes should be used for the time and frequency domains in both dimensions, see also [paragraph 3.4 Where is the NMR](#) (main window)
- how many contour levels to use by default (2D/3D Viewer)
- the default lower contour level limit (2D/3D Viewer)
- the default upper contour level limit (2D/3D Viewer)
- whether to use positive, negative, etc contours by default (2D/3D Viewer)
- the default azimuth angle for a mesh plot (2D/3D Viewer)
- the default elevation angle for a mesh plot (2D/3D Viewer)
- the default azimuth angle for a 3D stack plot (2D/3D Viewer)
- the default elevation angle for a 3D stack plot (2D/3D Viewer)
- the default colormap in the 2D/3D Viewer
- the default page orientation for printing
- the default paper size for printing

2.2.2 Colour Scheme Settings

Colour scheme settings for matNMR

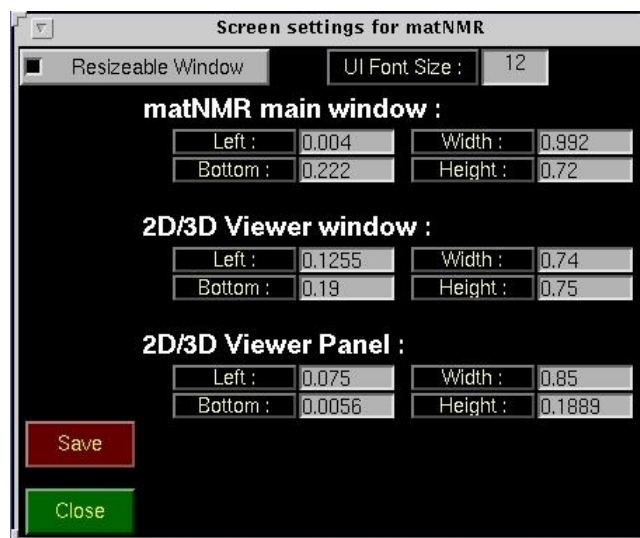
Name	Details	Values Background	Values Foreground	Results
Frame 1	background only	[0.4000 0.0000 0.0000]		
Frame 2	background only	[0.0000 0.0000 0.4000]		
Figure 1	most windows	[0.0000 0.0000 0.0000]		
Figure 2	input windows	[0.2000 0.2000 0.4000]		
UImenu	foreground only		[0.0000 0.0000 0.0000]	test test
Axes	foreground defines labels etc.	none	[1.0000 1.0000 1.0000]	test test
Ulcontrol 1	default Ulcontrol	[0.7000 0.7000 0.7000]	[0.0000 0.0000 0.0000]	test test
Ulcontrol 2		[0.0000 0.0000 0.0000]	[1.0000 1.0000 0.0000]	test test
Ulcontrol 3		[0.4000 0.0000 0.0000]	[1.0000 1.0000 0.0000]	test test
Ulcontrol 4		[0.0000 0.4000 0.0000]	[1.0000 1.0000 0.0000]	test test
Ulcontrol 5		[0.0000 0.0000 0.4000]	[1.0000 1.0000 0.0000]	test test
Ulcontrol 6		[0.3000 0.0000 0.3000]	[1.0000 1.0000 0.0000]	test test
Ulcontrol 7		[0.0000 0.0000 0.0000]	[1.0000 1.0000 1.0000]	test test
Ulcontrol 8		[0.4000 0.0000 0.0000]	[1.0000 1.0000 1.0000]	test test
Ulcontrol 9		[0.0000 0.4000 0.0000]	[1.0000 1.0000 1.0000]	test test
Ulcontrol 10		[0.0000 0.0000 0.4000]	[1.0000 1.0000 1.0000]	test test
Ulcontrol 11	input windows	[1.0000 1.0000 0.0000]	[0.0000 0.0000 0.0000]	test test
Ulcontrol 12	match background to figure 1	[0.0000 0.0000 0.0000]	[1.0000 1.0000 0.0000]	test test
Ulcontrol 13	match background to figure 1	[0.0000 0.0000 0.0000]	[1.0000 1.0000 1.0000]	test test
Ulcontrol 14		[1.0000 1.0000 1.0000]	[1.0000 0.0000 0.0000]	test test
Text 1		none	[1.0000 1.0000 1.0000]	test test
Text 2		none	[0.4900 0.4100 1.0000]	test test
Text 3		none	[1.0000 1.0000 1.0000]	test test
Line (main)	MUST be linespec format for colours!		y	
Line (dual)	MUST be linespec format for colours!		cmrbwg	

Close Save Predefined Schemes

This menu is used to define the colour scheme used in all matNMR windows. Several schemes have been pre-defined from which it is easy to make changes, if necessary. After changing values the results button will change, although for a complete overview of the effect matNMR must unfortunately be restarted. You MUST save the colour scheme

before restarting matNMR! Note that this manual only shows the classic colour scheme.

2.2.3 Screen Settings



This menu is used to define the screen sizes of the three main matNMR windows and also the font size that is used for UI controls. The numbers denote the size of the windows in relative units, i.e. they range from 0 to 1. Changes are applied immediately (just be sure to press "ENTER" after changing a value!).

2.2.4 Font List



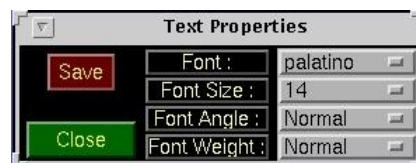
The font list menu species the list of fonts that matNMR recognizes. The default list in the downloaded code is based on a UNIX system which means many Windows/Macintosh fonts are not in this list! It is recommended to push the "import system fonts" button to create a list of fonts recognized on your system.

2.2.5 Line Properties



The line properties hold for both the plots in the main window and the 2D/3D Viewer, i.e. for contour plots and 3D stack plots. This menu can but should not be used for changing the appearance of single plots as all windows will be affected by this. For that the "Plot Manipulations" menu should be used instead.

2.2.6 Text Properties



The text properties hold for both the plots in the main window and the 2D/3D Viewer. This menu can but should not be used for changing the appearance of single plots as all windows will be affected by this. For that the "Plot Manipulations" menu should be used instead.



Chapter 3 - MATLAB and matNMR

3.1 MatNMR and the Workspace

One of the nice features of MATLAB is that it provides the user with a virtual workspace. It is basically memory that can be seen and manipulated from the MATLAB command line. (e.g. the MATLAB *who* or *whos* commands can be used to see what variables are currently in your workspace) Any variable that is defined will be put in this workspace and can at any time be manipulated again. Only variables that are used in functions have their own workspace. These cannot be accessed from the workspace and vice versa. Only when defining a variable to be global one can use this subsequently in a function.

MATLAB allows for two ways of programming: scripts (m-file) or functions. (for more information please read the MATLAB documentation) The first will work in the same workspace as can be seen from the workspace, whereas the second will usually only produce an output to the workspace. MatNMR has been programmed as a script and therefore ALL matNMR variables can be seen (AND manipulated!!) by the user. They are easy to recognize by the way as they ALWAYS start with a "Q"!

This has some consequences:

- High flexibility because a user can invoke certain changes manually, thereby manipulating the output produced by matNMR.
- Easy programming making user-additions to matNMR more easy.
- A huge number of matNMR variables are seen by the user, making it more difficult to see the forest through the trees.
- A risk of saving lots of unwanted matNMR variables to disk when trying to save something.

As the advantages strongly outweigh the disadvantages one just has to be careful when saving. Normally one can save all variables in the workspace by typing:

save filename.mat

This however will include all the matNMR variables which will increase the file size considerably. Therefore for saving some variables one either has to quit matNMR and then save or specify the variables that need to be saved by typing:

save filename.mat a b c d e f g

or by using the "Save Spectrum to Disk" menus in the 1D and 2D processing menus from the main window menubar (See chapter 5, ["The main window"](#)).

As a last remark: needless to say that the user must avoid using private variables that start with a "Q" as this may seriously affect matNMR!

3.2 Input Expressions

Input in matNMR is usually done using either UI-controls from the main window and 2D/3D Viewer panel window OR using the standard input window:



Whenever an edit button is used, i.e. a button in which one can write (for more information of different types of UI-controls please read the MATLAB documentation), the expression is evaluated afterwards using the MATLAB `eval` command. This means that any general function that produces a result that is correct for the given input question, can be given as input. For example in the input window displayed above one can specify "Name" as:

a (where a is a variable in the workspace)
 or **a*10+4./b** (where a and b are variables in the workspace)
 or **1:1024**
 or **diff((1.01).^ (0:511))** (type "help diff" for more information on diff.m)
 etc ...

This again gives a lot of flexibility as the input format is free as long as the result is correct.

Note that the "full width" button allows expansion of the input window to the full width of the screen. This can be useful when the text written in the edit buttons is very long.

3.3 Important Variables

Currently, matNMR is written as a script (m-file) with a large pool of global variables that are used to keep track of all things. These global variables are visible in the workspace as `QmatNMR` and `QmatNMRsettings`. Both are structures that contain many subitems. All items can be accessed in the usual way, i.e. `QmatNMR.Spec1D` will give the vector corresponding to the current 1D spectrum.

For most users it is not really necessary to know which variables are used by matNMR to do what. There are however a few variables of which it is probably not bad to know what they do. Some of these will invariably show up eventually in some of the input windows and then it may be nice to know what they are there for. The following list shows a few important variables used when working with the main window:

Variable Name	Usage in matNMR
Spec1D	The current 1D spectrum
Axis1D	The axis vector belonging to Spec1D
Size1D	The dimension size of Spec1D
SW1D	The spectral width of Spec1D in kHz
SF1D	The carrier frequency of Spec1D in MHz
Spec2D	The RR and RI parts of the current 2D spectrum (hypercomplex dataset)

Spec2Dhc	The IR and II parts of the current 2D spectrum (hypercomplex dataset)
AxisTD2	The axis vector belonging to TD2 of the current 2D spectrum
AxisTD1	The axis vector belonging to TD1 of the current 2D spectrum
SizeTD2	The dimension size of TD2 for the current 2D spectrum
SizeTD1	The dimension size of TD1 for the current 2D spectrum
SWTD2	The spectral width for TD2 of the current 2D spectrum in kHz
SWTD1	The spectral width for TD1 of the current 2D spectrum in kHz
SFTD2	The carrier frequency for TD2 of the current 2D spectrum in MHz
SFTD1	The carrier frequency for TD1 of the current 2D spectrum in MHz
Dim	The current dimension (0=1D, 1=TD2, 2=TD1)
Fig	The figure handle for the main window
History	The processing history belonging to the current 1D OR 2D spectrum
HistoryMacro	The processing macro belonging to the current 1D OR 2D spectrum
QFitResults	Results from the last peak fitting. These are not stored in the QmatNMR structure! For more information on the format of this variable see paragraph 5.3.1, General appearance .
QSSAFitResults	Results from the last MAS CSA fitting. These are not stored in the QmatNMR structure!
QCSAFitResults	Results from the last static CSA fitting. These are not stored in the QmatNMR structure!
QQuadFitResults	Results from the last quadrupolar-tensor fitting. These are not stored in the QmatNMR structure!
QDiffFitResults	Results from the last diffusion-curve fitting. These are not stored in the QmatNMR structure!
QT1FitResults	Results from the last relaxation-curve fitting. These are not stored in the QmatNMR structure!

As can be seen from the table there are different variables for 1D and 2D spectra. The distinction between working in 1D or in 2D mode is quite clear although it cannot be seen directly while working with matNMR. This is explained in more detail in paragraph 4.2, "[1D and 2D mode](#)".

A comparable list can be given for the 2D/3D Viewer window:

Variable Name	Usage in matNMR
Spec2D3D	The last plotted spectrum
Axis2D3DTD2	The axis vector belonging to TD2 of Spec2D3D
Axis2D3DTD1	The axis vector belonging to TD1 of Spec2D3D
PeakList	The peak list belonging to Spec2D3D
History2D3D	The processing history belonging to Spec2D3D
Fig2D3D	The current 2D/3D Viewer window
AxisNR2D3D	The current axis number (i.e. nr of subplot)
AxisHandle2D3D	The axis handle belonging to AxisNR2D3D

3.4 Where is the NMR?

A big difference between working with matNMR and other NMR processing packages is that the user doesn't NEED to work with typical NMR parameters like spectral width, the carrier frequency, etc while processing. Naturally that comes from the fact that MATLAB doesn't know NMR, only matrices, vectors and numbers, and also matNMR doesn't care really about the NMR behind a FID, i.e. **every axis plotted below a spectrum is given in points unless a different axis is defined**.

Having said that, matNMR does show the typical NMR units if wanted and if the relevant parameters have been supplied correctly. If the spectral width and the spectral

frequency for a certain dimension have been supplied (either during loading of a [binary FID](#) or manually, then the default axis, see also [paragraph 2.2.1 General Options](#), will produce proper time and frequency axes (Hz or PPM). The reference for such frequency axes is always set to 0 and in the center of the spectrum. NOTE that by changing the spectral width and/or spectral frequency the default axis will change immediately.

There are two ways of properly referencing spectra: 1) defining an axis for a particular spectrum directly OR 2) by applying an external reference.

1. Axis vectors can be defined from the (Plot Manipulations - ruler X-axis) menubar in the main window. The typical axes in PPM, Hz or time can be defined but also axes in points or other less typical axes can easily be defined (for more information on how to change the axis vector see [paragraph 5.2.5, "Plot manipulations"](#)). As the spectrum is just a numerical matrix it can be manipulated very easily during processing. The axis vectors are just used for plotting, but on the other hand:

NOTE: whenever there is an input concerning coordinates they must usually be given in the unit of the axis vector! (e.g. for extracting parts of a spectrum, integrating, etc....)

2. An external reference may be loaded from external datasets, either as defined in a file on your spectrometer, or by using the reference as made in another variable in matNMR. To use the values set on your spectrometer it is possible to import those values by using the "Import external reference" in the "Plot Manipulations" - "Ruler X-Axis" - "External reference" menus. MatNMR will then try and extract the values from other datasets on disk. Alternatively, you may define a fixed axis for a certain spectrum in matNMR, as described above. Either by saving the external reference directly, or by storing the spectrum in the workspace those values may serve as an external reference. The "Apply external reference" item will ask for a variable containing an external reference, i.e. a spectrum OR a reference only, which may then be applied to the current spectrum.

3.5 matNMR format for spectra

MatNMR can save spectra either as matrices into the workspace (or onto disk directly) or in the matNMR format. This matNMR format makes a structure of the variable with the following fields:

```
VariableName.Spectrum
VariableName.History
VariableName.HistoryMacro
VariableName.AxisTD2
VariableName.AxisTD1
VariableName.SweepWidthTD2
VariableName.SweepWidthTD1
VariableName.SpectralFrequencyTD2
VariableName.SpectralFrequencyTD1
VariableName.Hypercomplex
VariableName.PeakListNums
VariableName.PeakListText
VariableName.FIDstatusTD2
VariableName.FIDstatusTD1
```

(for more information on structures and other MATLAB data types please read the MATLAB documentation)

The spectrum will appear in the workspace as a struct array under the name VariableName. Saving in this format thus allows for easy access to the NMR parameters belonging to this spectrum. Note that for accessing the fields one needs to type "VariableName.FieldName" to obtain the proper result. As many functions will produce error messages when supplying them with the matNMR structure directly, it is important to realize of what type a certain variable is.

3.6 GUI and MATLAB

MatNMR is fully GUI (graphical user interface) programmed. Being able to do this is a nice feature of MATLAB but it is not 100% perfect. Especially when clicking the mouse very fast onto several objects can be the cause of many problems. The problem can often be traced back to the code that is run (bad programming). Often however such

problems arise from the fact that the MATLAB command line parser is slower than the mouse pointer. When using functions like `gcf`, `gca` and `gco` (get current figure, get current axis and get current object respectively) problems can occur because the current object is not the object that the code intended to work on. Therefore a word of caution must be given: don't overdo it, don't try and be the fastest mouse-clicker on earth when working with MATLAB.

3.7 Memory usage

As matNMR does all processing in memory the memory usage can be quite considerable. Especially so because MATLAB will only take continuous blocks of memory when creating a new variable. This means that when it cannot find a block in the currently taken memory of the proper size it will ask for more memory. On a UNIX system this can easily lead to memory usage of 256Mb when processing a 2048x2048 sized spectrum. Under Windows this seems to be less of a problem though. Anyway, it is important to be aware of how much memory is in use and if necessary to stop MATLAB every once in a while, just to free the memory again. Note, starting from version 2.7 matNMR has an optional undo function. Although very useful to have, this function may indeed induce massive memory usage if too many undo steps are allowed as the full matrices are stored for each step. More information on the undo function may be found in paragraph 5.2.11, "[Goodies](#)".

3.8 MATLAB and WYSIWYG

Unfortunately MATLAB in general is not a "what you see is what you get"-kind of program (WYSIWYG). Neither is matNMR therefore. For versions of matNMR older than 2.6 (march 2001) this calls for some improvisation if you really want to use matNMR to create nice plots. The axes, and therefore the spectra, are always plotted at their correct positions. The problem mostly lies with the fonts. For some reason they are not scaled in the same way as the axes. Especially the positioning of floating text, e.g. the super title in the 2D/3D Viewer, must be checked in the resulting plot. In principle it therefore advised to plot to a file on disk first before sending it to a printer. For matNMR 2.6 (march 2001) and newer an option has been added to the printing menu: "enforce WYSIWYG". This tries to ensure WYSIWYG behaviour but it still isn't perfect. The routine works fine as long as the absolute size of the window is not bigger than the paper size. Upon rescaling the window and all its objects and fonts again there is a problem with the fonts in some cases. The best thing to do then is to resize the window manually and make sure that the printing routine does not need to rescale it. A warning message concerning this point will be given by the printing routine though. For more information on how to output plots see chapter 7, "[Producing output](#)".



Chapter 4 - Processing with matNMR

4.1 Processing spectra with matNMR

Like any other program matNMR requires a certain sequence of events to be followed for properly processing spectra. In general it can be said that everything is done stepwise. Although processing macro's can be defined (see paragraph 4.4, "[Using macro's](#)") there are no all-in-one processing steps pre-defined. As an example two typical sets of processing actions are shown below, one for a 1D and one for processing a 2D spectrum. In general for 1D spectra the routine will be something like :

- **load 1D**
- **enter spectral width** (optional)
- **select a proper apodization function**
- **set size 1D**
- **choose Fourier Mode** (Optional --> is set by default options)
- **FT 1D**
- **adjust phase**
- **save spec 1D**

For a 2D this basically looks the same but it needs some steps more. This is because certain steps (like phasing and apodizing) assume working on a 1D spectrum and unless matNMR is instructed to work on the whole 2D matrix only the 1D spectrum in the current view is affected. (For more information see paragraph 4.2, "[1D, 2D and 3D mode](#)"). Therefore it is important not to forget the "Apodize 2D" and "Set Phase 2D" commands during processing. Typically processing a 2D will look like :

- **load 2D**
- **choose Fourier Mode** (Optional --> is set by default options)
- **enter spectral width of TD 2** (optional)
- **select a proper apodization function for TD 2**
- **Apodize 2D** (to apodize the whole 2D matrix !!)
- **set size 2D** (Only for TD 2 --> keeping TD 1 small makes the FT and phasing faster !!)
- **FT 2D**
- **adjust phase for the first row**
- **Set Phase 2D** (to set the phase for the whole 2D spectrum !!)
- **get column** (depends on where the signal is)
- **select a proper apodization function for TD 1**
- **choose Fourier Mode** (Optional)
- **enter spectral width of TD 1** (optional)
- **Apodize 2D** (to apodize the whole 2D matrix !!)
- **set size 2D** (Only for TD 1)
- **FT 2D**
- **adjust the phase for TD 1**
- **Set Phase 2D** (to set the phase for the whole 2D spectrum !!)

and if needed:

- **adjust the phase for TD 2**
- **Set Phase 2D** (to set the phase for the whole 2D spectrum !!)
- **Baseline correction**

4.1.1 Practical Tutorial

As a practical example is the easiest way to learn about matNMR, there are two datasets included in the matNMR distribution. These can be found in a subdirectory called "Examples", in the directory where you have stored matNMR. The datasets are solid-NMR spectra generated with Spinsight. They are called "SpinsightExample1" and "SpinsightExample2".

Loading the data into the Matlab workspace:

In this case there are two ways of importing the two datasets into Matlab: **separately** or as a **series**. To load them separately go into the "files" menu and select "binary FID". Select the file that contains the FID. For Spinsight data this file is always called "data". After selecting this file, an input window comes up. In this window the size of the data must be specified, the name of the variable in the Matlab workspace must be defined, and the format of the data must be specified. When the standard parameter files are present (i.e. the full dataset and not just the FID) then the sizes will be deduced from those files. The data format is in some cases recognized automatically. Finally, when the dataset should not only be stored in the workspace but also loaded into matNMR directly then select the flag.

Since the two datasets are both Spinsight datasets and have a very similar name, they can also be loaded as a series of binary FIDs. Select the appropriate entry in the "files" menu. Select the file that contains the FID. In the input window you can define as common part of the file name, e.g. "/home/jabe/matlab/matNMR/Examples/SpinsightExample\$\$/data" and a range "1:2". Here, the "\$#\$" is used as a code that is substituted once by All elements in the numerical range. Otherwise everything else is the same as above. Clicking OK will load both datasets in one go.

Processing the 1D dataset, SpinsightExample1:

From the main window, press button "load 1D" in the 1D menu, or go to the "1D Processing" menu. Use the name of the variable in the workspace that contains the data for example 1. Load the data as an FID. To process the spectrum use e.g.

- Hamming apodization (phase factor 0).
- Set the size to 2k points. Do a "complex FT", with or without multiplying the first point by 0.5.
- Use the mouse to apply -2.1 degrees zeroth-order phase correction, or type -2.1 in the appropriate edit button (press or mouse mouse pointer out of window to activate).
- To perform a baseline correction, go into the "1D Processing" menu and select "standard processing" and "baseline correction". Use "Define peaks" to exclude the area of the signal from the baseline fit. (left-click=define, right-click=stop). A polynomial of order 0, 1, or 2 should be fine in this case.
- Store the processed spectrum in the workspace by using the "Add to workspace" button.
- Alternatively, you may only store the processing actions with the FID (for large 2D spectra often smaller than storing the spectrum. In that case go into the "History / Macro" menu and select "Connect to FID". Supply the name that contains the example 1. For fun, reload the dataset, for example using the "reload last" button on the right of your screen. Go into the "History / Macro" menu and select "reprocess from history". This should reproduce the spectrum.

Processing the 2D dataset, SpinsightExample2:

From the main window, press button "load 2D" in the 2D menu, or go to the "2D Processing" menu. Use the name of the variable in the workspace that contains the data for example 2. Load the data as an FID. To process the spectrum use e.g.

- This is a 2D dataset with STATES in the indirect dimension for phase sensitivity. The format is already as matNMR requires it to be. First the dataset in made into a hypercomplex set (2 complex matrices). Use the "Start STATES processing" in the "2D Processing" menu.
- Add 40 Hz of Gaussian apodization. Make sure you use the "Apodize 2D" button to apply the apodization to the entire dataset!
- Set the size to 1k points in TD2 using the "set sizes" button in the 2D menu.
- Perform the FT in TD2. The FT mode should say STATES.
- Set the phase properly using the buttons. The reference for the first-order phase correction can be defined by clicking the "Reference Ph1" button. Use the crosshair to indicate the position. Using a "reference ph1" of 776 (assuming 1k points in TD2!), phase values of 51 and -227 degrees for the zeroth and first-order corrections should yield an in-phase spectrum with 4 lines.
- Check the position of the lines. These are useful for process the indirect dimension. Select the "Zoom" function and zoom into the left doublet. The use the "Get position" button to extract the position. Push the left button and move the mouse to follow the lines. The "X index" is the number you want to remember. Click on "Done".
- Repeat the previous for the right doublet in the spectrum. The indices should be e.g. 784 and 373.
- Go to column 784 by typing the number in the appropriate edit button in the window
- Because the pulse program did not conform to the default in Spinsight, the spectral width in TD1 could not be deduced. This was also shown in the Matlab command window (Please look!), where it says that the dw2 parameter could not be found. Hence the spectral width must be supplied manually. Type 25 (kHz) in the edit button next to "Spectr. Width (kHz)".
- Use Hamming apodization with phase factor 0. Make sure you use the "Apodize 2D" button to apply the apodization to the entire dataset!
- Set the size to 1k points in TD1 using the "set sizes" button in the 2D menu.
- Perform the FT in TD1. The FT mode should say STATES. This should show the same 4 lines as before, but reversed.
- Because the STATES modulation in the pulse program was not of the correct sign, the spectrum in TD1 is reversed, compared to TD2. To correct this use the "Flip L/R" in the "2D processing"+"standard processing" menu.
- Now the phase correction can be perfected: select row 784. Set the zero-order phase correction on the most left peak. Set the reference for ph1 on this peak. Select row 373. Adjust the first-order phase correction on the most right peak in the spectrum. Apply the phase to the entire spectrum ("Set Phase 2D"). Do the same in TD1.
- Select row 784 and go into the "2D Processing" menu, "standard processing", and select "baseline correction". use the "define peaks" to exclude the 4 signals from the baseline fit and use a first-order polynomial. Do the same for TD1.
- Store the spectrum in the workspace by using the "Add to workspace" button.

- View the spectrum using the 2D/3D viewer: push the appropriate button on the right side of the 2D menu. Make, for example, a contour plot by selecting relative contours.

4.2 1D, 2D and 3D mode

It is important to realize the difference between working on a 1D and on a 2D spectrum. In principle what is shown in the view, while processing a 2D spectrum, is a certain row or column from the matrix. The variable `QmatNMR.Dim` denotes the current dimension in which we are working: 0=1D spectrum, 1=TD2, 2=TD1. For a certain row the `QmatNMR.Dim` will be set to 1 (in a true 1D mode `QmatNMR.Dim` is set to 0). When changing the phase or apodizing, the current view is changed directly on the screen. The mode is not changed though and therefore `QmatNMR.Dim` will still be either 1 or 2. However, the change is not imposed to the entire 2D matrix until the corresponding buttons "Set Phase 2D" and "Apodize 2D" are pushed!

Then there are several functions (Sum TD2, Sum TD1, Diagonal, etc) that operate on a 2D matrix and which have a 1D spectrum as output. In those cases the mode is switched to a true 1D mode (`QmatNMR.Dim=0`). This is important to realize, e.g. when doing baseline correction: It is often convenient to look at the sum over a certain dimension to see where the intensities are in the spectrum. However then the mode is switched to a 1D mode and the 2D baseline correction routine doesn't know on which dimension to operate then (by default it will assume to be working on TD2 for this routine by the way).

Another way of switching the mode from 2D to 1D is by selecting a 1D function while processing a 2D spectrum. For example doing a 1D FFT will switch the current mode to 1D and do a FFT then. The 2D spectrum is not affected by this at all. (This can be also useful for checking certain processing steps as using 1D functions is usually faster than 2D functions.) To switch back to the 2D mode simply select a row or column from the matrix, using the appropriate UI-controls in the main window.

MatNMR does NOT offer full 3D processing capabilities but rather allows for efficient processing of series of similar 2D spectra, especially using processing macros. When working with 3D matrices in the main window a new window is opened from which an output variable must be declared. This is done in order to conserve memory usage: the 3D matrices are not copied into dedicated matNMR variables but changes to the input variable are stored directly in the output variable. The first dimension of the 3D matrix is the index of the spectrum, whilst the second and third dimension stand for TD1 and TD2 respectively.

This feature is particularly useful for processing sets of simulations. Subsequent plotting of these sets of spectra is facilitated by the 2D/3D viewer support for 3D matrices. Upon detecting a 3D matrix, the 2D/3D viewer will check the number of subplots in the figure window and will try and plot the set of 2D spectra into the available subplots. When combining this with plotting macro's, very efficient plotting of the data may be achieved. For more information on how to optimize the use of plotting macro's, see paragraph 4.4, ["Using macro's"](#).

4.3 Using the processing history

A relatively unique feature of matNMR is the processing history (only recently other packages have started to implement this). MatNMR keeps track of all actions during processing. After processing the spectrum is finished the history can be connected to the FID. When reloading the FID then into matNMR, the spectrum can be obtained by "Reprocessing from History", found in the "History" menu of the menubar in the main window. This has some nice advantages:

- No need to write down processing parameters as they are saved on disk
- No need to save the spectrum to disk (which is usually much bigger than the FID)
- Easy and fast reprocessing of the FID afterwards

Connecting a processing history to an FID induces matNMR to convert the matrix in the workspace to the matNMR format. For more information about this see paragraph 3.5, ["matNMR format for spectra"](#). All necessary information can easily be accessed using this format. The peak list can also be connected to the FID making plotting the only step necessary after reprocessing a spectrum.

History macros can also be converted into independent processing scripts on disk. The scripts can be applied to any dataset, the name of which is the input parameter of the script. See also Chapter 8, [Script-based processing with matNMR](#).

At any time the processing history can be viewed as text in a separate window. The text is stored in the variable `QmatNMR.History`. At the same time however a numerical matrix is stored in the workspace, called `QmatNMR.HistoryMacro`, that contains the processing history as a matNMR macro. This is explained in more detail in paragraph 4.4 below.

Now in principle can every processing step be reprocessed afterwards. Everything except creating a non-linear axis vector! Most axes have linear increments and therefore only the axis offset and increments are saved into the processing history macro. Non-linear axes are not stored into it. This also means they will not be reproduced when reprocessing. It wouldn't be terribly difficult implementing this but it would increase the size of the macro considerably and is therefore not implemented currently.

More explanation on how to use the history can be found in paragraph 5.2.6, "[History/Macro](#)".

4.4 Using macro's

MatNMR allows for the definition of macro's from the menubar in the main window. Currently, both processing and plotting actions can be stored in a macro. Macro's in matNMR are just matrices of numbers encoding for all processing steps. Besides adding general matNMR processing routine steps to a macro, user-defined commands may be defined as well. The command string for these commands will be translated into numbers and is stored as such in the history macro. After finishing recording the macro it can be saved as a normal variable in the workspace (or on disk) for later use. Processing macro's can only be executed in the main window. Plotting macro's can be executed in any of the matNMR windows.

Note that plotting actions are not permanent and are usually all removed when matNMR updates a window. In case of the 2D/3D viewer plotting macro's can be defined in multi-plot situations (see next paragraph), but such macro's can only be executed in 2D/3D viewer windows with exactly the same subplot configuration! When recording a macro in the 2D/3D viewer window then actions that are not intended for all subplots, will only be executed for axes which are selected. If an axis is selected then a box is drawn around it to distinguish it from normal axes. Multiple axes may be selected by right-clicking an axis. That way one can easily create effective plotting macro's.

To avoid warning messages by matNMR to say that the subplot configuration is not correct, one may make plotting macros which only operate on the current axis. These work in all matNMR windows. Only in the 2D/3D viewer window a special step must be taken to create such macros; in all other windows these are made by default. In the 2D/3D viewer window one must deselect all axes before executing a processing action. This will result in the action being executed on the current axis only!

An important point in using plotting macro's effectively is to plan in advance. Rendering of graphical objects is by far the slowest step for anything but simple line plots. Especially in 2D/3D viewer windows with multiple subplots the rendering time can be painfully long. Changing axis parameters in such cases is less than pleasant. But, by planning the plotting macro in advance, the rerendering may be reduced to only once, which can save lots of time. Simply select a particular [subplot configuration](#) and make all changes to all subplots, store the macro and start plotting the spectra.

Finally, there is one more way to execute a macro and that is by calling the function "matNMRRunMacro" from a user-defined script. This requires the matNMR GUI to be opened as is explained in more detail in [Chapter 8, Offline processing](#).

More explanation on how to create and run macro's can be found in paragraph 5.2.6, "[History/Macro](#)".

4.5 Multi-window matNMR

MatNMR is programmed such that all windows can be left open while working with it. It may not be very convenient to look at 100 windows at the same time but matNMR shouldn't crash. For the main window this works quite straightforward as there is only one data window and no subplots currently. For the 2D/3D Viewer a note must be made though: as has been mentioned in paragraph 3.7, ("[Memory usage](#)") the memory usage by MATLAB can be quite considerable. Therefore only one set of variables will be kept in memory for the 2D/3D Viewer! So even when you have several 2D/3D Viewer windows with various numbers of subplots in them, only ONE set of variables is kept in memory. This set of variables (spectral matrix, axis vectors, history, peak list, etc) always belongs to the **last plotted spectrum**.

4.6 Using short-keys

From the main window several short-keys have been defined. They can be seen when selecting an entry on the menubar and generally look like <CTRL> + key. Unfortunately many easy short keys are already reserved by MATLAB so matNMR doesn't use many currently.

4.7 Apodizing FID's

From the main window several apodization functions can be chosen:

Type	Gives
Exponential	The FID is multiplied with a pure exponential decay leading to pure Lorentzian linebroadening. The spectral width must be specified properly for this function else the linebroadening will not correspond to the specified increase in Hz.
Cos ²	The FID is multiplied with a quarter period of a squared(cosine) ranging in intensity from 1 (cos ² (0)) to 0 (cos ² (pi/2)) if the phase factor is set to 0. Any other value for the phase factor will lead to a different shape. Note that this does not give pure Lorentzian or Gaussian linebroadening.
Gaussian	The FID is multiplied with a Gaussian function leading to pure Gaussian linebroadening. The spectral width must be specified properly for this function else the linebroadening will not correspond to the specified increase in Hz.
Hamming	The FID is multiplied with a Hamming function for optimal S/N.
Block + Cos ²	The FID is multiplied with a function consisting of a block of intensity 1 (length of the block must be specified in points) and a Cos ² . Note that this does not give pure Lorentzian or Gaussian linebroadening.
Shifting Gauss	The FID is multiplied with a pure Gaussian function but the maximum of the gaussian is shifted for each TD1 increment. The amount of shifting depends on the respective spectral widths in both dimensions, the Fourier mode (see also paragraph 4.8, " Fourier transform ") and the shearing factor (this is defined by the NMR experiment type). Furthermore the position of the echo maximum must in the first row be given. The direction of the shift depends on whether it is an echo or an anti-echo experiment. This function can only be used for 2D FID's.
Exponential (echo)	Performs an exponential linebroadening on FID's with a full echo in it. It is assumed that the "Swap whole echo" function (see also paragraph 5.2.3, " 1D processing " or paragraph 5.2.4, " 2D processing ") has been performed before applying this apodization.
Gaussian (echo)	Performs a Gaussian linebroadening on FID's with a full echo in it. It is assumed that the "Swap whole echo" function (see also paragraph 5.2.3, " 1D processing " or paragraph 5.2.4, " 2D processing ") has been performed before applying this apodization.
Shifting Gauss (echo)	Performs a shifting-Gauss linebroadening on FID's with a full echo in it. The amount of shifting depends on the respective spectral widths in both dimensions, the Fourier mode (see also paragraph 4.8, " Fourier transform ") and the shearing factor (this is defined by the NMR experiment type). Furthermore the position of the echo maximum must in the first row be given. When the echo maximum is set to 0 (after the "swap whole echo") the apodization consists of two converging gaussian functions (in time) instead of diverging (the usual idea with echo-spectra). This function can only be used for 2D FID's.

4.8 Fourier transform

Fourier transformation in matNMR is done slice-by-slice using the standard MATLAB FFT routine. In general a FFTshift (see MATLAB documentation for more information) is done to end with the common result of all NMR processing programs.

As in general the first point of most FID's needs to be multiplied by 0.5 (because most FID's are apodized to 0) this is automatically selected in the main window. It can however be switched off for FID's which do not need this (e.g. an undamped cosine), as this would otherwise give a baseline offset.

The inverse FT can also be performed, but has to be accessed from the "Additional Features" item in either the "1D Processing" or "2D Processing" menus in the menubar.

Various types of Fourier transform modes corresponding to various types of NMR experiments are supported:

Type	Gives
Complex	Both the real and the imaginary parts of the FID are used.
Real	The imaginary parts of the FID are made 0 before doing the FFT.
States	2D Spectra recorded using the States scheme require this setting both in TD2 and TD1. A hypercomplex dataset will result from this.
TPPI	2D Spectra recorded using the TPPI scheme require this setting ONLY for the processing in TD1. In TD2 the complex FFT should be used. To use this setting in TD2 usually means that the Redfield scheme (i.e. TPPI in 1D) was used and the data have been converted (Bruker qseq mode, see also paragraph 5.2.3, "1D processing" or paragraph 5.2.4, "2D processing"). In 2D FID's hypercomplex dataset will result from this. The spectral width is halved after the Fourier transform to account for the fact that half of the data is thrown away. Note that for qseq data also the direct processing can be done (so without doing the above-mentioned conversion first).
Whole Echo	2D Spectra with phase modulation instead of amplitude modulation (whole echo acquisition) require complex FFT in both dimensions. Using this setting for TD2 avoids having to define the complex FFT for both dimensions.
States-TPPI	2D Spectra recorded using the States-TPPI scheme require this setting both in TD2 and in TD1. A hypercomplex dataset will result from this.
Bruker qseq	This setting performs a similar transform as TPPI does except that it also inverst every second complex point. When the above-mentioned conversion step has been done TPPI must be used and not this mode!
Sine FT	This setting performs a real Fourier transform whilst assuming that the FID is sine-modulated. A sine function will produce an anti-phase spectrum around the carrier. To obtain an in-phase spectrum one half of the spectrum is inverted.

For processing States or States-TPPI spectra special care must be taken to provide matNMR with the proper format of the FID matrices. MatNMR uses the same format as Chemagnetics does and so the FID-matrix should be built up like this :

-----> real part of the matrix
RR RI
----- imaginary part of the matrix
IR II

Here the RR part is the real part in t1 and t2. The RI is the imaginary part in t1 and the real part in t2, etc. (This gives the rather unnatural looking view of 2 FID's in the view) On Bruker and Varian machines the imaginary parts in t1 will probably be alternated with the real parts in t1 :

(RR RI)1
(IR II)1
(RR RI)2
(IR II)2
etc.

After loading a Bruker or Varian States (or States-TPPI) FID into the workspace this first needs to be converted before starting the States processing. This is done from the corresponding entries in the "2D processing" menu: "Convert Bruker States" or "Convert Varian States".

NOTE: Starting processing a States (or States-TPPI) spectrum can only be started by selecting the entry in the 2D Processing menu from the menubar!
This splits the matrix into the proper hypercomplex parts QFT1 and QFT2 and sets the proper flags.

4.9 Phasing spectra

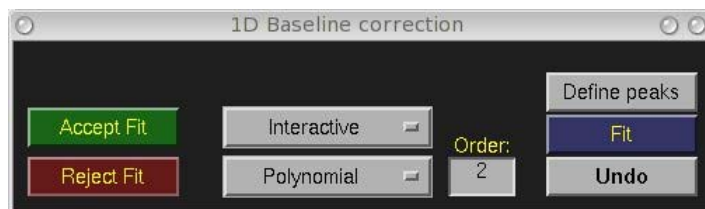
Phase correction of spectra is done using the phase menu (see also paragraph 5.1, "[General appearance](#)"). There are several ways to set the zeroth-order, first-order and second-order phase correction: There is a slider button, an edit button for writing the value in directly and there are step-wise incremental buttons (small and big steps). To set the first-order correction a reference position must be specified. This can be done by clicking on the "Reference Ph1" button. A cross-hair will appear to aid finding the proper peak position. Clicking the left mouse button defines the position. The "zero phases" button will set the phasing values to zero again.

Note that to set the second-order phase correction one needs to push the check button in order to open an extra panel (since it is not commonly used). The reference for the 2nd order phase correction is always in the center of the spectrum since it is used to even out the effects of sharp cut-offs in modern audio filters; these effects are symmetrical around the center of the spectrum.

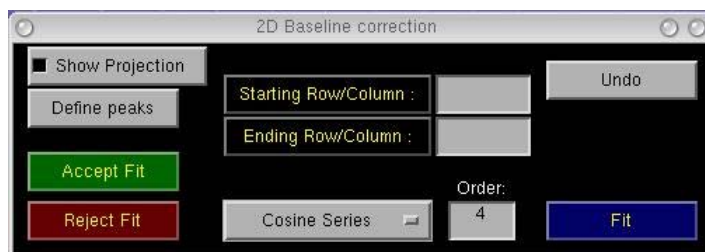
While phase correcting a 2D spectrum it is often necessary to watch different slices to see whether the new phase values are any good. Switching to a new row or column (while not switching dimension!) will show this row/column with the phase values shown in the phase menu on screen. Only when switching dimension will the phase values be reset again. Alternatively to switching between slices one may use the "2D tool", which allows watching two additional slices/columns from which the quality of the phase correction may be estimated. The additional plots are updated with each change of the phase. After having found a good phase correction for a 2D be sure to press the "Set Phase 2D" button to impose the new phase correction on the entire 2D matrix!

4.10 Baseline correction

A semi-interactive baseline correction routine is available both for 1D and 2D spectra. Upon selecting this function a window will appear looking either like:



or



For both the 1D and 2D baseline corrections the areas where peaks can be found in the spectrum must be identified before doing the fit. This can be done using the "Define Peaks" button. Using the left mouse button, click both left and right of a peak region to define it. A red marking will appear over the area. Any mouse click outside the axis or using the right mouse button cancels defining peaks. For 2D spectra it is possible to have the projection onto the axis plotted in the window, to facilitate choosing the position of the peaks. The specified range in the 2D will be taken into account here, i.e. the projection shows the signal from the area that needs baseline correcting only!

Next, the type of function that needs to be used for fitting the baseline must be specified. Currently one can choose from either a cosine series, a polynomial or a Bernstein polynomial. These are defined as:

Polynomial of order 6: $A + Bx + Cx^2 + Dx^3 + Ex^4 + Fx^5$

Cosine series of order 6: $A + B\cos(x) + C\cos(2x) + D\cos(3x) + E\cos(4x) + F\cos(5x)$

Bernstein of order 6: $A(1-x)^5 + Bx((1-x)^4) + Cx^2((1-x)^3) + Dx^3((1-x)^2) + Ex^4(1-x) + Fx^5$

Sine: $A + B \sin(Cx + D)$

Exponential: $A + B \exp(Cx)$

where A-F are the coefficients to be fitted to the baseline and x the coordinate along the spectral axis. After choosing the order to which the fit should go, and pressing "Fit" the fit is performed. Any order of fit is allowed but beware that a fit will become very bad when the order is too high.

The resulting 1D spectrum or row/column from the 2D will be shown in the view. For 1D baseline correction also the fitted baseline is shown in blue as a visual aid. To see the result of the baseline correction for the whole 2D just step through the rows/columns.

Should the baseline be satisfactory then the "Accept Fit" button must be pressed and the fit is made final. If not, either press "Undo" to obtain the original spectrum back or "Reject Fit" to reject the fit and stop the baseline correction routine while retaining the original spectrum.

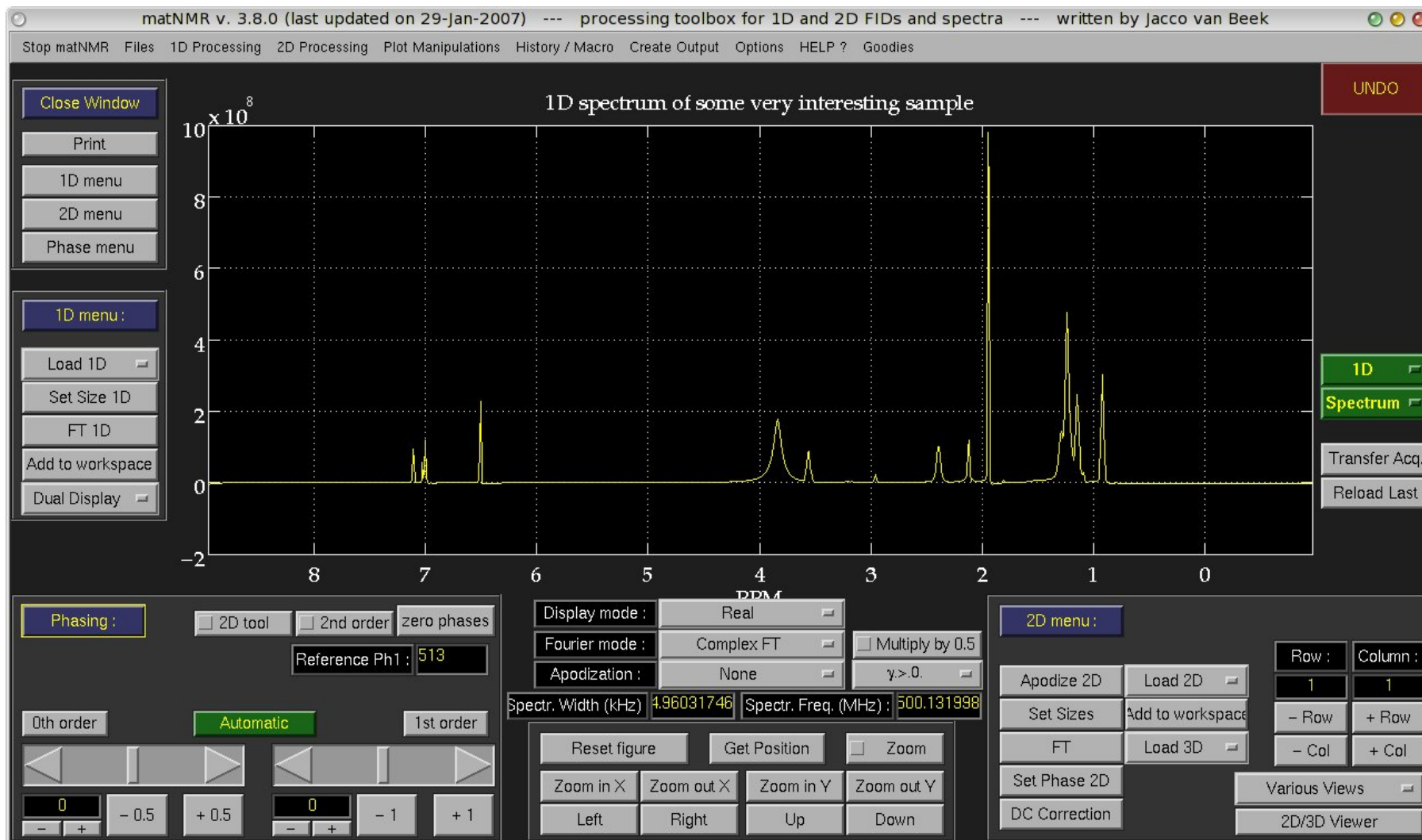


Chapter 5 - The main window

The main window is started using the `nmr.m` script, i.e. type 'nmr' at the Matlab prompt to start it. All processing is done from this window and all other functionality implemented in matNMR can be accessed from the menubar or the Uicontrols.

5.1 General Appearance

The general appearance of the matNMR main window is shown in the figure below:



There are five boxes shaded with a background box around it. Going from top-left to bottom-right they contain

1- a stop-button, a direct-printing button (prints to the default printer as defined in the \$matlabroot/toolbox/local/printopt.m file) and some buttons to open the various menus

2- some often-used buttons for working with 1D spectra. More functions can be found in the menubar.

3- the phasing menu. (see also paragraph 4.9, "[Phasing spectra](#)")

4- some buttons for zooming in discrete steps or continuously (zoom button). Furthermore there is the "reset figure" button which resets the current view (leaving nothing but the current 1D spectrum Qspc2 in the view!) and the "get position" button allows to scan the lines in the plot using the mouse pointer. Upon pressing the left mouse button a crosshair will appear and some buttons that show the coordinate and the intensity of the current trace. When more than one lines are in the plot a popup button will appear to choose which line the routine should follow.

5- some often-used buttons for working with 2D spectra. More functions can be found in the menubar.

Then there are some buttons just above the box with zoom functions.

The "display mode" is for defining whether to show the real, imaginary or absolute value of a FID/ spectrum. Note that this is only a matter of displaying. MatNMR always works with complex values.

The "fourier mode" defines what type of FFT is performed when the FFT-button is pressed (see also paragraph 4.8, "[Fourier transform](#)").

The "apodization" button allows choosing an apodization function. Usually an input window will appear to ask for more information. The "spectr. width" edit button allows to directly enter the spectral width belonging to the 1D FID/spectrum or 2D row/column. In case of a 2D mode the value will be connected to the relevant spectral dimension. Changing the dimension (say from TD2 to TD1) and back will then give the previously entered value back again. The "spectr. freq." edit button allows to directly enter the spectral frequency belonging to the 1D FID/spectrum or the current 2D row/column. In case of a 2D mode the value will be connected to the relevant spectral dimension. Changing the dimension (say from TD2 to TD1) and back will then give the previously entered value back again.

Finally there are five buttons on the right on the plot. The top button calls the [undo function](#). The second button shows the mode matNMR is in, either 1D or 2D. The third button specifies whether the FID/ spectrum in the current plot is an FID or a spectrum. This determines the plotting direction (For non-NMR people: in NMR the convention is to plot a time-scale from left-to-right, a PPM scale from right-to-left and a Hertz (Hz) scale depending on the gyromagnetic ratio of a nucleus). The "Transfer Acq." button has a highly specific function: it uploads the variable QacqFID from the workspace and makes it the current spectrum (1D or 2D). This has been implemented because some people wanted to use MATLAB for a spectrometer interface. The "Reload Last" button reloads the last 1D or 2D FID/spectrum. It just evaluates the Qspct variable that is used by matNMR to denote the name of the last 1D/2D spectrum (for more information on variables used by matNMR see paragraph 3.3, ["Important variables"](#)).

5.2 The menubar

As the menubar contains most of the processing functions implemented in matNMR all entries will be explained here separately. Note that many of the processing functions may also be accessed from the context menus that are defined. Context menus may be accessed by right-clicking anywhere within the figure window (even on top of the UI controls!). The context menus are also accessible on top of the axis, except when the zoom function is switched on. Switching off the zoom will result in the context menus being accessible again.

5.2.1 Stop matNMR

This function allows to stop matNMR or quit MATLAB. Depending on whether the exit safety flag was set (see paragraph 2.2.1, ["General options"](#)) an input window will appear to ask for confirmation when stopping matNMR. For quitting MATLAB the user is always asked for confirmation. Note that when stopping matNMR **ALL** windows belonging to matNMR are closed, including the 2D/3D Viewer. Stopping other routines will not stop any other routines though.

5.2.2 Files

This menu deals with the importing of files into MATLAB:

Name		Function
External MATLAB files		Opens a window which allows selecting the MATLAB file that needs to be uploaded into the workspace from disk.
Series of external MATLAB files		Allows importing a series of similarly-named MATLAB files from disk into the workspace. The files should have a common name and a specific range attached to them. For example the series j011101_1, j011101_2, ... , j011101_20 is imported by supplying 'j011101_\$\$' as the name and '1:20' as the range.
Last series of external MATLAB files		As the previous function but takes the file name to be the same as the previously-chosen file and immediately goes to the second input window.
Import	Binary FID's	Opens a window which allows selecting the FID file that needs to be oploaded into the workspace. After selecting the file a second input window will ask for the sizes of the FID and a name for the new variable (in the workspace). If selected, the FID will be read into matNMR directly, otherwise the variable will only be stored in the workspace and a "Load nD" must be executed to load it into matNMR. The total number of points (real+imaginary!) in the FID file is indicated in this window. This number does not hold for the VNMR (Varian) format as this routine only divides the binary file size by 4 (floating point format uses 4 bytes per number). The structure of the VNMR FID files prevents displaying the correct size. Currently the formats for XWinNMR, UXNMR, winNMR (Bruker), Spinsight (Chemagnetics), VNMR (Varian) NTNMR (TecMag), MacNMR (TecMag), Bruker Aspect 2000/3000 and JEOL Generic format are recognized. Limited support for the SMIS format is also implemented.
		Depending on which format is chosen the time domain sizes must be specified. Whenever it is not necessary to do this the corresponding uicontrols are disabled.

	<p>It is also possible to use the standard spectrometer parameter files to specify all the necessary parameters. This is particularly useful in connection with the default axes, see also paragraph 3.4 Where is the NMR and paragraph 5.2.5 Plot manipulations. Then for all Bruker formats the files "acqu" and "acqu2" are used, for Chemagnetics the "acq" file, for VNMR the "procpars" and for JEOL Generic format the ".hdr" file.</p> <p>It is important to note that the following assumptions are made when using the standard parameter files:</p> <ul style="list-style-type: none"> • only the standard files can be used • if the sizes in the parameter files do not match the size of the FID an error is given • For 2D spectra the values for the spectral width and spectrometer frequency are not generally reliable unless the FID has been recorded following the standard pulse programming style for that spectrometer and using the corresponding standard variables. <p>Even then the following limitations apply:</p> <ul style="list-style-type: none"> ◦ For Bruker SWH and SFO1 are read from acqu2 for the indirect dimension ◦ For Chemagnetics dw and dw2 to calculate SWH by $SWH=1/dw$ (i.e. States is assumed in the indirect dimension!). Arrays are read without problems but the SWH is set to 1 in the indirect dimension. The spectrometer frequency in TD1 is taken to be the same as in TD2. ◦ For VNMR sw and sfrq are read for TD2 and sw1 for TD1. The spectrometer frequency in TD1 is set to the same value as for TD2. ◦ For JEOL x_curr_points, x_freq and x_sweep are used for TD2 and y_curr_points, y_freq and y_sweep for TD1. <p>NOTE: IF YOU WANT TO IMPROVE THIS OVERSIMPLIFIED SCHEME PLEASE LET ME KNOW!!</p>
Get last binary FID	As the previous function but takes the file name to be the same as the previously-chosen file and immediately goes to the second input window (for faster uploading of equal sized spectra). If selected, the FID will be read into matNMR directly, otherwise the variable will only be stored in the workspace and a "Load nD" must be executed to load it into matNMR.
Series of binary FIDs	Allows importing a series of similar binary FIDs into the workspace. They should have a common name and a specific range attached to them. For example the series j011101_1, j011101_2, ... , j011101_20 is imported by supplying 'j011101_\$\$' as the name and '1:20' as the range.
Last series of binary FIDs	As the previous function but takes the file name to be the same as the previously-chosen file and immediately goes to the second input window (for faster uploading of equal sized spectra).
Bruker Spectrum	<p>opens a window which allows selecting the Bruker spectrum file that needs to be oploaded into the workspace. After selecting a second input window will ask for size and name of the new variable. Most of these parameters can be avoided by using the option of reading standard parameter files, which will look for the original parameter file and extract the necessary parameters. Alternatively, they may be entered by hand: The blocking factor is a feature that is used by XWinNMR sometimes (mainly older versions) to save a spectrum in blocks. The number that must be specified here is the XDIM parameter that is written in the procs and proc2s files, in the directory of the dataset. If 0 is entered then no blocking factor is taken.</p> <p>The byte ordering determines how XWinNMR has stored the data. Taking the wrong type will typically result in meaningless noise. This information is in fact stored in the procs file but it's just as fast to select it yourself, than it is to select a file on disk. If the data has been processed on SGI then the byte ordering should be big endian, else it is most likely little endian.</p> <p>The imaginary and hypercomplex parts may be loaded automatically by selecting the option in the input window. This assumes that all file names are standard, i.e. that a standard Bruker dataset is presented.</p> <p>Finally, if selected, the spectrum will be read into matNMR directly, otherwise the variable will only be stored in the workspace and a "Load nD" must be executed to load it into matNMR.</p>
Get last Bruker Spectrum	As the previous function but takes the file name to be the same as the previously-chosen file and immediately goes to the second input window (for faster uploading of equal sized spectra). If selected, the spectrum will be read into matNMR directly, otherwise the variable will only be stored in the workspace and a "Load nD" must be executed to load it into matNMR.
Series of Bruker spectra	Allows importing a series of similar Bruker spectra into the workspace. They should have a common name and a specific range attached to them. For example the series j011101_1, j011101_2, ... , j011101_20 is imported by supplying 'j011101_\$\$' as the name and '1:20' as the range.
Last series of Bruker spectra	As the previous function but takes the file name to be the same as the previously-chosen file and immediately goes to the second input window (for faster uploading of equal sized spectra).
Chemagnetics Spectrum	opens a window which allows selecting the Chemagnetics spectrum file that needs to be oploaded into the workspace. After selecting a second input window will ask for size and name of the new variable. Most of these parameters can be avoided by using the option of reading standard parameter files, which will look for the original parameter file and extract the necessary parameters. If selected, the spectrum will be read into matNMR directly, otherwise the variable will only be stored in the workspace and a "Load nD" must be executed to load it into matNMR.
Get last Chemagnetics Spectrum	As the previous function but takes the file name to be the same as the previously-chosen file and immediately goes to the second input window (for faster uploading of equal sized spectra). If selected, the spectrum will be read into matNMR directly, otherwise the variable will only be stored in the workspace and a "Load nD" must be executed to load it into matNMR.
Series of Chemagnetics spectra	Allows importing a series of similar Chemagnetics spectra into the workspace. They should have a common name and a specific range attached to them. For example the series j011101_1, j011101_2, ... , j011101_20 is imported by supplying 'j011101_\$\$' as the name and '1:20' as the range.

	This function does not need to be used as the qseq mode for FT allows direct processing but is recommended. Direct processing in principal does not allow further manipulation (e.g. apodization is ever so slightly wrong when applying it to the complex vector).	
Standard Processing	Remove Bruker digital filter	Removes the effects of the Bruker digital filter. This filter performs a time-shift on the FID of n points which can be corrected by giving a 360*n degree first order phase correction to the spectrum. This routine does an FFT, 360*n degree first order phase correction and then an IFFT to obtain the FID back. Now apodization can be done as usual.
	Shift data points	Shifts the FID (or spectrum) by a certain number of points. The FID is kept at the same length by adding the same number of zeros to the end. A positive number will shift the FID left (n+1 -> n), a negative number right (n -> n+1).
	Swap whole echo	Performs a data shift such that the maximum of a FID with an echo in it, will appear at the first point afterwards. All points before the maximum are added to the end of the FID. After this the FID can be apodized (see also paragraph 4.7, "Apodizing FID's").
	DC offset correction	Performs an offset correction. A signal-free part of the FID is asked for after which the real and imaginary parts of the FID are corrected independently by substracting the mean of the specified region.
	Solvent Suppression	Performs a solvent deconvolution as was proposed by Marion, Ikura and Bax, JMR, 84, 425-430 (1989). Either a Gaussian, Sine-Bell or Rectangular window' function may be used with a width a (2*K + 1). The jump size M denotes which point to use for the extrapolation of the first and last K points.
	Set size	Changes the size of the current spectrum. A size smaller than the current size will lead to cutting away the end of the spectrum, a bigger size will lead to zeros being added to the end of the spectrum.
	FT	Performs a Fourier transform on the current FID. The type of Fourier transform is defined by the corresponding button in the main window (see paragraph 5.1, "General appearance"). For more information on the differences between the various modes see paragraph 4.8, "Fourier transform".
	FFTshift	Performs a FFTshift on the current spectrum, i.e. a shift of the zero-point of the spectrum to the center (see MATLAB documentation for more information on fftshift).
	Flip L/R	Flips the current spectrum from left to right. Necessary in case the spectrum is flipped (different receiver rotation).
	Baseline correction	Does a 1D baseline correction on the current spectrum (see also paragraph 4.10, "Baseline correction").
Additional Features	Concatenate Matrix	Allows concatenation of the current 1D spectrum into a longer 1D or a 2D with all slices the same.
	Extract from 1D	Extracts part of the current 1D spectrum and makes it the new 1D spectrum. The range that must be specified must be given in the unit of the axis vector. The new axis vector remains in the units of the original vector.
	Integrate	Shows the integral of an area in the current 1D spectrum. Specify a region by clicking left and right of this region. Very basic.
	Inverse FT	Performs the inverse Fourier transform to the current dimension of the 2D. The exact workings depend on the Fourier mode selected, and some of the possible Fourier modes do not allow inverse FT. A notice is given when this is the case.
	Linear prediction	Performs a forward or backward linear prediction. Two methods for linear prediction (back- and forward) are implemented in matNMR: LPSVD and ITMPM. They are described in the following paper: Y.-Y. Lin, P. Hodgkinson, M. Ernst and A. Pines, JMR, 128, 30-41 (1997) . Please refer to this paper for the explanation of the methods. To properly predict points: Cut the FID such that all points that remain are good (so take away the initial bad points or the noise at the end of the FID). Select the method that you want to use and specify the number of points that need to be predicted, the number of frequencies that are assumed in the signal and the S/N-ratio. The ITMPM-method allows the algorithm to determine the number of signals. For that choose either -1 or -2 (different methods).
	Noise filling	This option changes the size of the FID bit instead of appending zeroes, Gaussian noise is appended.
	Regrid Spectrum	Interpolates the current spectrum to a new axis. Different interpolation algorithms may be used to optimize for speed or quality.
	S/N	Very basic routine for determining the signal to noise ratio of a spectrum. Asks for a peak-free region of the spectrum to calculate S/N as the maximum divided by the standard deviation of the noise.
Fitting	CSA tensors MAS	Fits a tensor to the MAS sideband intensities. For a complete discription of this routine see paragraph 5.5, "Fitting of CSA tensors from the MAS sideband manifold".
	CSA tensors Static	Fits tensors to a static powder pattern. For a complete discription of this routine see paragraph 5.5, "Fitting of static CSA tensors".
	Diffusion Curves	(Multi-)exponential fit of diffusion curves. For a complete discription of this routine see paragraph 5.5, "Fitting of Diffusion Curves".
	Peak Deconvolution	Fits mixed Gaussian/Lorentzian oaks to the current spectrum. To reduce the number of data points that the peak-fitting routine has to take into account, first zoom into the region of interest. For a complete discription of this routine see paragraph 5.6, "Spectral Deconvolution").
	Quadrupolar Tensors	Fits a quadrupolar tensor to a MAS spectrum. For a complete discription of this routine see paragraph 5.8, "Fitting of Quadrupolar Tensors under MAS").

Relaxation Curves	(Multi-)exponential fit of relaxation curves. For a complete discription of this routine see paragraph 5.8, "Fitting of Relaxation Curves").
-------------------	---

5.2.4 2D Processing

This menu deals with everything concerning working with 2D spectra:


Name	Function	
Load 2D	Gets a new 2D spectrum. Axis vectors may be specified and also whether the variable is an FID or a spectrum (see also button information in paragraph 5.1, "General appearance").	
Load 3D	Gets a new 3D spectrum (For more information on 3D support see paragraph 4.2, "1D, 2D and 3D mode" . Axis vectors may be specified and also whether the variable is an FID or a spectrum (see also button information in paragraph 5.1, "General appearance").	
Transpose	Transpose the current 2D spectrum. May be needed to convert spectra that were saved with different conventions than used by matNMR.	
Add Spectrum to Workspace	Saves the current 2D spectrum to the workspace. Depending on whether the history and/or axis vector must be included a matNMR structure will be created (also see paragraph 3.5, "matNMR format for spectra").	
Save Spectrum to Disk	As the previous function but saves the variable to disk. In case more than one variable names are given in the input window it will save the variables without doing anything with the current 2D spectrum!	
Convert Bruker qseq	The Bruker qseq acquisition mode is basically TPPI in 1D. This in principle gives a real vector but when loading a binary it the spectrum is assumed to be complex. This function converts the FID into a real vector with all real and imaginary points concatenated and every second complex point inverted: R I R I R I R I R I R I... This function does not need to be used as the qseq mode for FT allows direct processing but is recommended. Direct processing in principal does not allow further manipulation (e.g. apodization is ever so slightly wrong when applying it to the complex vector).	
Convert Bruker/Varian States	Converts the standard Bruker/Varian way of saving States spectra into the matNMR format (Chemagnetics format). For more information on the format see paragraph 4.8, "Fourier transform" .	
Echo / Anti-Echo to States	Converts an echo and an anti-echo experiment into a States-type experiment. The matrix must be in the same format as a States experiment. For more information on the format see paragraph 4.8, "Fourier transform" .	
Start States processing	Splits the States FID into its proper hypercomplex parts and sets all necessary flags for States processing (see also paragraph 4.8, "Fourier transform").	
Full matrix manipulations	Remove Bruker digital filter	Removes the effects of the Bruker digital filter from all rows/columns of the current 2D FID (depending on the current dimension). This filter performs a time-shift on the FID of n points which can be corrected by giving a 360*n degree first order phase correction to the spectrum. This routine does an FFT, 360*n degree first order phase correction and then an IFFT to obtain the FID back. Now apodization can be done as usual.
	Shift data points	Shifts all rows/columns of the current 2D FID (depending on the current dimension) by a certain number of points. Each row/column is kept at the same length by adding the same number of zeros to the end. A positive number will shift the FID left (n+1 -> n), a negative number right (n -> n+1).
	Swap whole echo	Performs a data shift on all rows/columns of the current 2D FID (depending on the current dimension) such that the maximum of a FID with an echo in it, will appear at the first point afterwards. All points before the maximum are added to the end of the FID. After this the FID can be apodized (see also paragraph 4.7, "Apodizing FID's").
	DC offset correction	Performs an offset correction on all rows/columns of the current 2D FID (depending on the current dimension). A signal-free part of the FID is asked for after which the real and imaginary parts of each row/column are corrected independently by substracting the mean of the specified region in each row/column.
	Solvent Suppression	Performs a solvent deconvolution as was proposed by Marion, Ikura and Bax, JMR, 84, 425-430 (1989). Either a Gaussian, Sine-Bell or Rectangular window' function may be used with a width a (2*K + 1). The jump size M denotes which point to use for the extrapolation of the first and last K points.
	Apodize	Performs the currently selected apodization mode on all rows/columns of the current 2D FID (depending on the current dimension). See paragraph 4.7, "Apodizing FID's" for more information on apodizing.
	Set size	Changes the size of the current 2D spectrum. A size smaller in any dimension than the current size will lead to cutting away the end of the dimension, a bigger size will lead to zeros being added to the end of the dimension.
	Zero part of 2D	Sets entire rows and columns of the current 2D matrix to 0. NOTE: input of the ranges is in points! For example: '1:20'. Leaving an input line empty means not zeroing any rows or columns.
	FT	Performs a Fourier transform on all rows/columns of the current 2D FID (depending on the current dimension). The type of Fourier transform is defined by the corresponding button in the main window (see paragraph 5.1, "General appearance"). For more information on the differences between the various modes see paragraph 4.8, "Fourier transform" .

	FFTshift	Performs a FFTshift on all rows/columns of the current 2D FID (depending on the current dimension), i.e. a shift of the zero-point of the row/columns to the center (see MATLAB documentation for more information on fftshift).
	Flip L/R	Flips all rows/columns of the current 2D spectrum (depending on the current dimension) from left to right. Necessary in case the spectrum is flipped (different receiver rotation).
	Set Phase	Performs the currently selected phase correction on all rows/columns of the current 2D FID (depending on the current dimension). See paragraph 4.9, " Phasing spectra " for more information on phase correction.
	Baseline correction	Does a 2D baseline correction on the current spectrum (see also paragraph 4.10, " Baseline correction ").
Single Slice Manipulations	This submenu allows the manipulations of the current row/column of the current 2D, without the rest of the matrix being affected. Look at the entries for the regular functions to see what they do.	
Additional Features	Concatenate Matrix	Allows concatenation of the current 2D spectrum into a bigger 2D spectrum. Concatenation can be done in both directions.
	Extract from 2D	Extracts part of the current 2D spectrum and makes it the new 2D spectrum. The range that must be specified must be given in the unit of the axis vector. The new axis vectors remain in the units of the original vectors.
	Integrate Fixed range	Calculates the integral of a part of a spectrum by selecting a region of the current 2D spectrum (in units of the axis vectors) and adding all intensities. The output is a variable in the workspace with the intensities, the axis in the current dimension and the position of the maximum signal in units of its original axis.
	Integrate around Max	Shows the integral of an area in the current 1D spectrum. This routine searches in each slice in the 2D for the maximum signal in a user-specified range and integrates with a user-specified range around that maximum. This allows integration of peaks with variable positions and possibly overlapping other peaks. The output is a variable in the workspace with the intensities, the axis in the current dimension and the position of the maximum signal in units of its original axis.
	Inverse FT	Performs the inverse Fourier transform to the current dimension of the 2D. The exact workings depend on the Fourier mode selected, and some of the possible Fourier modes do not allow inverse FT. A notice is given when this is the case.
	Linear prediction	Performs a forward or backward linear prediction on all rows/columns of the current 2D spectrum (depending on the current dimension). Two methods for linear prediction (back- and forward) are implemented in matNMR: LPSVD and ITMPM. They are described in the following paper: Y.-Y. Lin, P. Hodgkinson, M. Ernst and A. Pines, JMR, 128, 30-41 (1997) . Please refer to this paper for the explanation of the methods. To properly predict points: Cut the FID such that all points that remain are good (so take away the initial bad points or the noise at the end of the FID). Select the method that you want to use and specify the number of points that need to be predicted, the number of frequencies that are assumed in the signal and the S/N-ratio. The ITMPM-method allows the algorithm to determine the number of signals. For that choose either -1 or -2 (different methods).
	Noise filling	This option changes the size of the FID bit instead of appending zeroes, Gaussian noise is appended.
	Regrid Spectrum	Interpolates the current spectrum to a new axis. Different interpolation algorithms may be used to optimize for speed or quality.
	Set integral	Sets the integral of the spectrum by selecting a region of the current 2D spectrum (in units of the axis vectors) and assigning a value to this.
	Shearing transformation	Performs a shearing transformation on the current 2D spectrum (i.e. frequency domain shearing) or FID (i.e. time domain shearing). In either case the shearing factor (as determined by the NMR experiment) and the spectral widths in both TD1 and TD2 must be specified in the input window. For the frequency domain version also the direction of shearing can be specified (vertical or horizontal). In general a vertical shearing is needed for NMR spectra but the horizontal shearing may have a general use. The frequency domain shearing uses a spline interpolation scheme, i.e. this is not as accurate as the time domain shearing, which is just a TD1-dependent first order phase correction to the matrix after FT in TD2 is done. NOTE: When shearing in frequency domain the imaginary parts are set to zero, i.e. no phase correction will be possible afterwards!
	Symmetrize	Symmetrizes the current 2D spectrum along the diagonal either by averaging the intensities on both sides of the diagonal or by taking the highest value of the two sides. Note that the spectrum must be square to be able to do this.
Fitting	Peak deconvolution	Performs a (absorptive) peak fit on peaks in the current dimension of the 2D spectrum. To reduce the number of data points that the peak-fitting routine has to take into account, first zoom into the region of interest. The difference with the "Peak fitting" entry in the "1D Processing" menu is that now the full 2D matrix is supplied to the peak fitting routine which allows fitting series of 1D spectra consecutively. For a complete discription of this routine see paragraph 5.6, " Peak fitting ".
Various Views	Diagonal:	Extracts the diagonal of the spectrum. The axis vector belonging to the resulting spectrum is in points.
	Anti-diagonal:	Extracts the anti-diagonal of the spectrum. The axis vector belonging to the resulting spectrum is in points.
	Sum TD1:	Extracts the sum over all columns of the spectrum. The range of columns to take into account for this function can be specified in the unit of the axis. The axis vector belonging to the resulting spectrum is the same as the original axis.
	Sum TD2:	Extracts the sum over all rows of the spectrum. The range of rows to take into account for this function can be specified in the unit of the axis. The axis vector belonging to the resulting spectrum is the same as the original axis.

	Skyline:	Extracts a skyline projection of the 2D spectrum at a specified angle. For a direct view of Td2 set the angle to 0, for a direct view of TD1 set it to 90. Currently only the angles of 45 and 135 degrees are allowed besides 0 and 90. The axis vector belonging to the resulting spectrum is equal to the original vector for angles of 0 and 90, else in points.
	Horizontal Stack plot:	Creates a horizontal stack plot from all rows/columns of the current 2D matrix (depending on the current dimension). The zoom limits of the current view are taken into account when creating the 1D stack plot. NOTE: this is a special plot type and will not persist when pushing the "reset figure" button.
	Vertical Stack plot:	Creates a vertical stack plot from all rows/columns of the current 2D matrix (depending on the current dimension). The zoom limits of the current view are taken into account when creating the stack plot. Also, the vertical displacement factor may be changed to create the optimum plot. NOTE: this is a special plot type and will not persist when pushing the "reset figure" button.
2D/3D Viewer	Shortcut to the 2D/3D Viewer window. The current name of the input variable for this routine is set to QFT1, i.e. the current 2D matrix in the main window.	

5.2.5 Plot manipulations

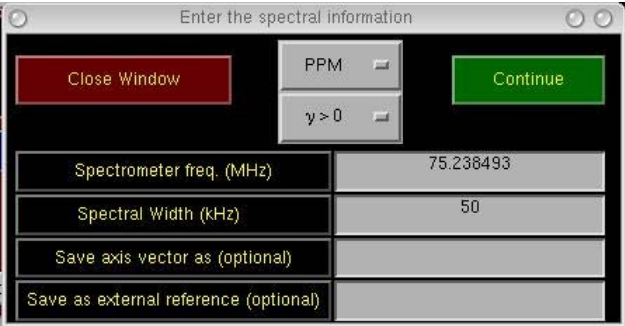
This menu contains all kinds of functions to change the appearance of the current plot.

Name	Function												
Legend	<p>This routine provides a means of adding legends to a plot. A window will appear looking like:</p> <div></div> <p>from which the labels, the text properties and all individual line properties can be specified. The routine checks the number of lines currently in the plot automatically and provides sufficient input boxes. Line properties can be changed separately or in groups: select the desired lines by clicking on the check buttons in front of each line. If there already is a legend present in the plot it is best to remove it first before adding a new one as it may be that the routine does not find the previous legend properly. The resulting legend will be plotted on top of the existing plot in the main window. It can be moved however by keeping the left mouse button pressed over the legend and then moving the mouse.</p>												
Ruler x-axis	<p>This submenu provides a means of changing the axis to a plot. For more information see also paragraph 3.4, "Where is the NMR"</p> <table><tr><td>Use Default Axis</td><td>Switches between the default axis and a fixed axis. A check flag shows when the default axis is on. The unit of the axis is defined by the current settings and whether the current spectrum is an FID or a spectrum.</td></tr><tr><td>Reset Default Axis</td><td>Resets the values for spectral reference, which makes the center of the spectrum 0 ppm/Hz or kHz.</td></tr><tr><td>Change Default</td><td>Changes the current settings of the units for the default axis for either FID or spectrum mode.</td></tr><tr><td>External Reference</td><td><table><tr><th>Name</th><th>Features to manipulate the spectral referencing through external data.</th></tr><tr><td>Apply External Reference</td><td>Allows the application of a spectral reference, as stored in the workspace either as a spectrum or as a reference variable, to the current spectrum. For 2D spectra the reference values may be entered differently for both dimensions by entering two values for each parameter. If only one value is specified then both dimensions are taken to be the same.</td></tr></table></td></tr></table>	Use Default Axis	Switches between the default axis and a fixed axis. A check flag shows when the default axis is on. The unit of the axis is defined by the current settings and whether the current spectrum is an FID or a spectrum.	Reset Default Axis	Resets the values for spectral reference, which makes the center of the spectrum 0 ppm/Hz or kHz.	Change Default	Changes the current settings of the units for the default axis for either FID or spectrum mode.	External Reference	<table><tr><th>Name</th><th>Features to manipulate the spectral referencing through external data.</th></tr><tr><td>Apply External Reference</td><td>Allows the application of a spectral reference, as stored in the workspace either as a spectrum or as a reference variable, to the current spectrum. For 2D spectra the reference values may be entered differently for both dimensions by entering two values for each parameter. If only one value is specified then both dimensions are taken to be the same.</td></tr></table>	Name	Features to manipulate the spectral referencing through external data.	Apply External Reference	Allows the application of a spectral reference, as stored in the workspace either as a spectrum or as a reference variable, to the current spectrum. For 2D spectra the reference values may be entered differently for both dimensions by entering two values for each parameter. If only one value is specified then both dimensions are taken to be the same.
Use Default Axis	Switches between the default axis and a fixed axis. A check flag shows when the default axis is on. The unit of the axis is defined by the current settings and whether the current spectrum is an FID or a spectrum.												
Reset Default Axis	Resets the values for spectral reference, which makes the center of the spectrum 0 ppm/Hz or kHz.												
Change Default	Changes the current settings of the units for the default axis for either FID or spectrum mode.												
External Reference	<table><tr><th>Name</th><th>Features to manipulate the spectral referencing through external data.</th></tr><tr><td>Apply External Reference</td><td>Allows the application of a spectral reference, as stored in the workspace either as a spectrum or as a reference variable, to the current spectrum. For 2D spectra the reference values may be entered differently for both dimensions by entering two values for each parameter. If only one value is specified then both dimensions are taken to be the same.</td></tr></table>	Name	Features to manipulate the spectral referencing through external data.	Apply External Reference	Allows the application of a spectral reference, as stored in the workspace either as a spectrum or as a reference variable, to the current spectrum. For 2D spectra the reference values may be entered differently for both dimensions by entering two values for each parameter. If only one value is specified then both dimensions are taken to be the same.								
Name	Features to manipulate the spectral referencing through external data.												
Apply External Reference	Allows the application of a spectral reference, as stored in the workspace either as a spectrum or as a reference variable, to the current spectrum. For 2D spectra the reference values may be entered differently for both dimensions by entering two values for each parameter. If only one value is specified then both dimensions are taken to be the same.												

Define External Reference	Allows manual definition of a spectral reference. For 2D spectra the reference values may be entered differently for both dimensions by entering two values for each parameter. If only one value is specified then both dimensions are taken to be the same.
Import External Reference	Allows importing of a spectral reference from an external dataset. This requires the default axis mode is on AND that all spectral parameters for the current dataset have been set correctly. Especially the sign of gamma (magnetogyric ratio) is important as this will determine the sign of a frequency axis. NOTE that both XWinNMR and Spinsight do not handle this correctly and that the sign may therefore be different. Currently this only works for the XWinNMR, VNMR and Spinsight formats but should be implemented for other formats as well at some point (kick me to speed up this process!)

Other

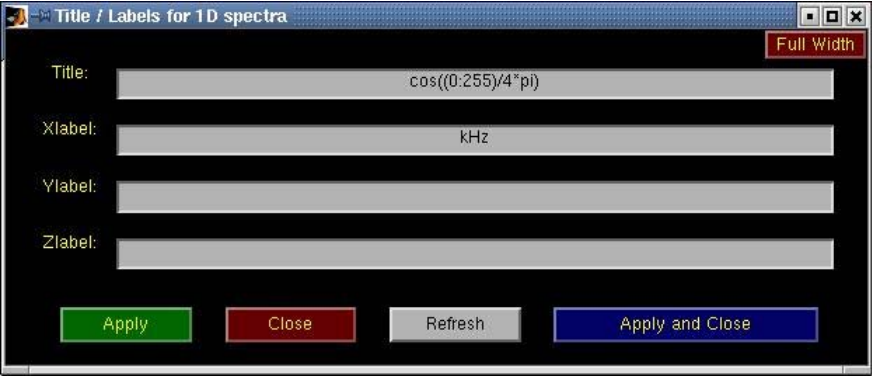
The rest of the menu allows direct access to axes in time, gradient, points or a user-defined vector, but also to the menu-based option "ruler X-axis". A window will appear looking like:



from which the type of axis vector (PPM, kHz, Hz, time, gradient, points, user-defined) can be chosen. The spectral width must be specified for PPM and Hz axes and for a PPM axis also the carrier frequency. For an axis in Hz also the sign of the gyromagnetic ratio of the nucleus that corresponds to the current spectrum must be specified.
NOTE: Gradient axes are defined in T/m and NOT in G/cm. This means you may have to divide the gradient values from your software by 100 to obtain the proper values!!
Pressing the "Continue" button proceeds the routine and one is asked for a reference value. After that the reference peak can be specified by using the crosshair mouse pointer and clicking on the desired reference peak (This may seem strange but is necessary because of a MATLAB bug. Specifying the reference peak before asking its value leads to a mess with the main and input window). If a name is given for the new axis vector then it will be saved in the workspace after calculating it.

Title/axis labels

This routine provides a means of changing the title and axes labels of a plot. A window will appear looking like:




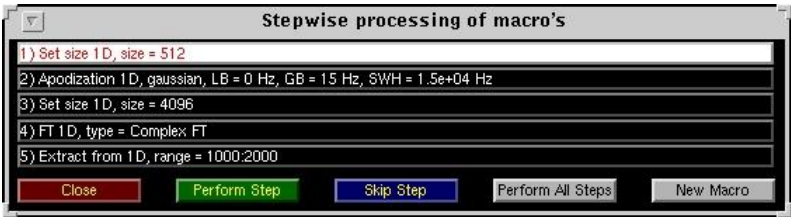
Pressing the "Apply" button will update all labels immediately. The "Refresh" is used for updating the text labels currently written in this window to the labels in the current plot (useful when having this window opened continuously). The "full width" button expands the window to the full width of the screen to give wider edit buttons. This is useful for long texts as they can become inaccessible because the scrolling doesn't work perfectly.

Plotting Functions	This submenu currently contains	
	Name	Function
	1D Bar Plot	a function to create a 1D bar plot from the current view. The current axis is taken and the user may select the various properties of the bars from the input window. NOTE: this is a special plot type and will not persist when pushing the "reset figure" button.
	Errorbar Plot	a function to create a errorbar plot from the current view. The current axis is taken and the user must provide a vector with standard deviations (or similar). NOTE: this is a special plot type and will not persist when pushing the "reset figure" button.
Various axis properties	Show Sidebands	a simple function to indicate MAS sidebands in the spectrum. This requires specifying the central line and the MAS frequency, and various parameters that change the appearance of the markers. The markers can be removed by pressing the "reset figure" button.
	The rest of this menu consists of various axis properties. More explanation concerning these can be found in the MATLAB documentation. Note the difference between changing plot properties using this menu and by changing the properties using the options menu (see also paragraph 2.2, "Setting up matNMR"): changes made with these functions only affect the current plot and not the general matNMR settings.	

5.2.6 History/Macro

This menu deals with the processing history and working with macro's (see also paragraphs 4.3, "Using the processing history" and 4.4, "Using macro's") :

	Name	Function
Show History		This routine shows the processing history for the current spectrum (either 1D or 2D). A window will appear looking like: 
		The "Print" button allows to print the history to a printer or to disk. A separate window will be opened in which the history is plotted as graphical text (Unfortunately this is necessary because it is much more difficult to print text in UI-controls than graphical text in MATLAB).
Connect history to FID		This routine connects the current processing history macro to a FID variable in the workspace. If the variable is not a matNMR structure yet (see also paragraph 3.5, "matNMR format for spectra") it will make a structure out of it. If the variable already is a matNMR structure then only the HistoryMacro field will be changed.
Save history as macro		This routine saves the current processing history macro as a variable in the workspace. It may then at any time bu run as a macro.
Convert history macro to script		This routine converts the current processing history macro to an independent processing script on disk. The script can then be applied to any dataset. The script contains many comments as to what the syntax of the matNMR processing scripts is. This option serves as an alternative to storing the processing history with the FID. (See also Chapter 8, Script-based processing with matNMR)
Reprocess from history		This routine reprocesses the processing history macro that is connected to the current FID. Note that this only works when done directly after loading a FID into matNMR because else the history macro will be overwritten by the new processing steps. (see also paragraph 4.4, "Using macro's")
Reprocess from history stepwise		Same as above but now the reprocessing can be done stepwise. A window is opened in which the information of the current and the four next steps is displayed.

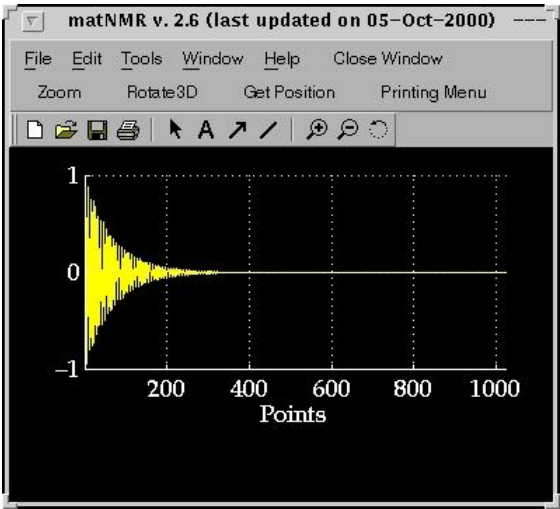


Steps can be performed and skipped. Furthermore the "perform all steps" button performs all next steps without confirmation. The "New Macro" button loads a new macro and displays its processing steps. (see also paragraph 4.4, "Using macro's")

Clear History	Clears the current processing history, i.e. the QmatNMR.History variable.
Start recording macro	Starts the recording of a new macro. An input will appear to ask whether only processing, only plotting or all actions must be stored in the macro. Recording continues until the stop command is given (see below). Note that for user-commands, i.e. commands that are usually executed from the MATLAB command line, to be put into the macro the "execute user-command" function (see below) must be used! (see also paragraph 4.4, "Using macro's")
Stop recording macro	Stop the recording of a macro. An input window will appear to ask for a name of a variable in the workspace to store the new macro in. (see also paragraph 4.4, "Using macro's")
Execute user-command	This function is used to enter user-commands to a macro (and thus also to the processing history macro!). An input window will appear to ask for the command. The resulting string is evaluated using the MATLAB <i>eval</i> command (see the MATLAB documentation for more information about eval) and stored in the history macro in a numeric format. (see also paragraph 4.4, "Using macro's")
Execute macro	This routine executes a macro. (see also paragraph 4.4, "Using macro's")
Execute macro stepwise	This routine executes a macro stepwise. The same window as used for "Reprocess from history stepwise" (see above) will appear .

5.2.7 Create Output

Name	Function
Printing Menu	The printing menu is a general feature available from all matNMR windows that is used for printing. Using a separate window the MATLAB <i>print</i> command (see the MATLAB documentation for more information on print) can be executed. More information on the printing menu is given in paragraph 7.1, "Printing menu".
Save figure to disk	This function will store the current window as a binary figure on disk as an executable m-file. This will allow the user to recall the exact figure by retyping the name of the m-file on the command line. This uses the Matlab mfig data format. See the Matlab manual for more information on the "saveas" command
Copy figure	This function will copy the current view of the main window into a separate window, but without the UI-controls. This will give something like:



The menubar of this new window contains the standard MATLAB menubar which has some useful features for printing and saving the figure. It also offers the possibility to change axis properties and stuff (More information on this menubar can be found in the MATLAB documentation). Additional features in the menubar are the MATLAB zoom and rotate3D functions, the matNMR "Get Position" (for more information see paragraph 5.1, "[General appearance](#)") and the printing menu (for more information see paragraph 7.1, "[Printing menu](#)").

5.2.8 Options

This menu allows changing all default settings of matNMR as is described in detail in paragraph 2.2, "[Setting up matNMR \(options\)](#)".

5.2.9 Help

Besides the simple "About matNMR" and "Copyright" notices this contains a direct link to this manual from inside MATLAB. The browser, defined in the \$matlabroot/toolbox/local/docopt.m file, will be opened and the matNMR manual will appear (See the MATLAB documentation for more information on the MATLAB *web* and *docopt* commands). As the manual is included in the matNMR distribution reading the manual should be faster than using the matNMR website.

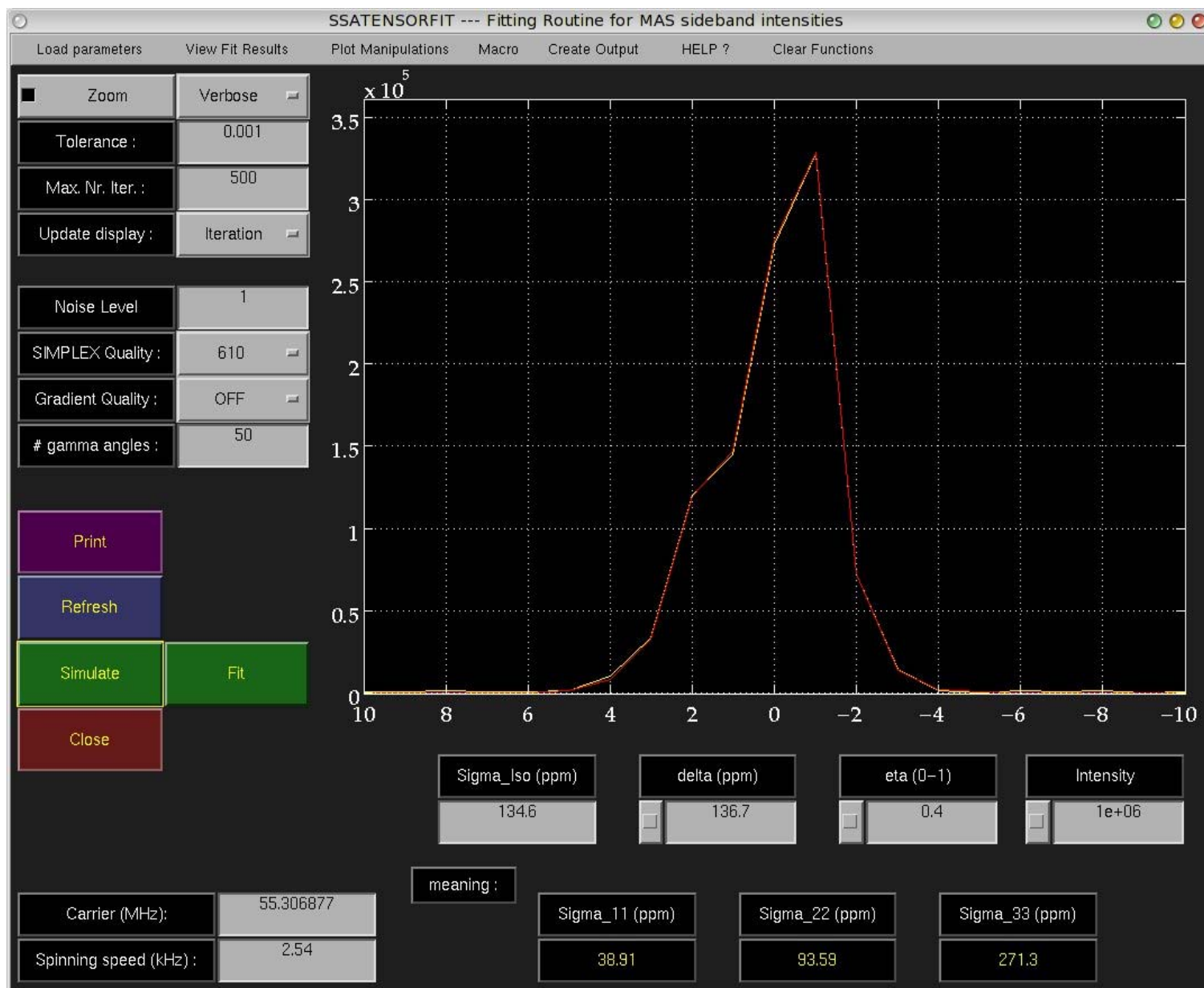
5.2.10 Goodies

Name	Function
Undo	This function undoes the last processing step which may be 1D or 2D mode. There is also an undo button in the main window which does exactly the same. By pressing CTRL-Z in the window the function may also be called. The total number of processing steps that may be undone can be set in the general options menu . Beware though of the extra memory usage especially for the 2D mode! A full copy of the hypercomplex matrices is stored for each step which for a 1024x1024 spectrum already means 32Mb per undo step!
Clear Functions	this performs the MATLAB <i>clear functions</i> command. All compiled functions are deleted from memory forcing MATLAB to recompile each function again upon running it. This is useful when working with your own m-files (or changing the matNMR code) to force MATLAB to use the altered function (See the MATLAB documentation for more information on the MATLAB <i>clear</i> command). Furthermore the mouse pointer is set to an arrow head.
Reset after Error	this performs almost the same as the clear functions command, but also recreates the axis. This is sometimes useful when the view is distorted after an error has appeared (typically a MATLAB error).
Select matNMR Distribution	Allows switching of matNMR distribution. This is only useful if you have multiple versions of matNMR stored on disk and you wish to use another version than the one you are currently using. The path variable is changed in order to make sure the new distribution is used exclusively and matNMR is restarted. No safety is built in so information may be lost. After restarting Matlab, the path variable will be the same as usual

5.3 Fitting of CSA tensors from the MAS sideband manifold

5.3.1 General appearance

MatNMR offers a fitting routine for magic-angle spinning sideband intensities from the "1D Processing" entry in the menubar. The following figure window will appear:



The routine can be operated using the UI-controls in the figure window. The functions of these will be explained in [paragraph 5.3.2](#) just below. The auxiliary functions in the menubar are described in [paragraph 5.3.3](#) below.

Upon starting the routine the current 1D spectrum (main window) is taken and plotted. It is assumed that the current spectrum is a set of sideband integrals and that the axis ruler has been changed to indicate the sideband magnetic deviation from the centerband (=0). Sideband number -3 means that the frequency of the peak is at $\omega_0 - 3\omega_r$. This is independent of the sign of the gyromagnetic ratio!

To best way to obtain the sideband amplitudes is to use the deconvolution routine, [paragraph 5.6](#) to fit the lineshapes of the individual sidebands, and take the values for the integrals and use them for the fit of the sideband amplitudes. This is more accurate than taking only the intensity of the line at a multiple of the spinning speed, since spinning speed instabilities will hardly affect the centerband but the increasingly affect the line for higher sideband numbers. The integral of the line should not be affected. After fitting the 1D spectrum, the results are stored in a structure and one of the fields is called "Integrals". These contain the values for the peaks. Add the correct axis for the sideband numbers, taking care that the sign of the gyromagnetic ratio is correct. Otherwise the sign of the anisotropy delta will invert.

To fit the sideband manifold, the Larmor frequency must be entered in MHz and the magic-angle spinning frequency in kHz. The isotropic shift in ppm may also be entered if tensor values sigma_11, sigma_22 and sigma_33 are required. The routine itself only fits the anisotropy "delta" and the asymmetry "eta", and an intensity. The sideband manifold is calculated using a gammaCOMPUTE-based algorithm which uses an analytical expression for the frequency of a crystallite. Through powder averaging the complete sideband manifold can be calculated.

As holds for all non-linear fits a good initial estimate of the parameters is required for an efficient fit convergence. To facilitate finding such an estimate the "Simulate" button may be used. After having defined the initial guesses the fit can be started with the "Fit" button. Should a parameter not be fitted but kept constant at its current value, then this can be done by selecting the check button (with the square block in it) belonging to the parameter.

Two algorithms may be used for fitting: SIMPLEX and a gradient algorithm. Typically the SIMPLEX algorithm is used for coarse optimization and is followed by a gradient minimization to finish off the analysis. The quality of the simulation of the sideband manifold can be determined for both algorithms separately. The simulation uses a gammaCOMPUTE-based calculation with a two-angle ZCW scheme for angles alpha and beta, and a separate number of angles gamma, which can be selected in the appropriate UIcontrol. By choosing a higher number of powder-averaging angles the quality of the simulation will improve. By default 610 powder-averaging angles are selected for SIMPLEX and none for the gradient algorithm. The value of 610 angles, combined with 50 values for angle gamma seems to provide a stable and fast result for a wide number of conditions (say up to 10 sidebands at least). When more sidebands are required it is important to test the stability of the fit by increasing the number of averaging points.

When the fit has finished by default the results are written into the variable QSSAFitResults which can be read out by hand by the user (Note that this variable will be deleted when matNMR is stopped! This means it must be renamed, if it has to be kept in the workspace for later use, before stopping matNMR). Alternatively, from the "View Fit Results" entry in the menubar a new name for putting the results in can be defined. To be able to access this variable the user must make the variable global by typing "global VariableName" at the MATLAB prompt. A message is written to make sure the user doesn't forget this. The format of the parameters is as follows:

- delta
- eta
- intensity
- Error
- Carrier
- Spinning Speed
- isotropic shift

All these numbers are put in a single row. This variable is combined with the original data, the axis to the data, the fitted spectrum and the error value into a structure variable. The fields in this variable are "Parameters", "Fit", "Data", "Axis" and "Error".

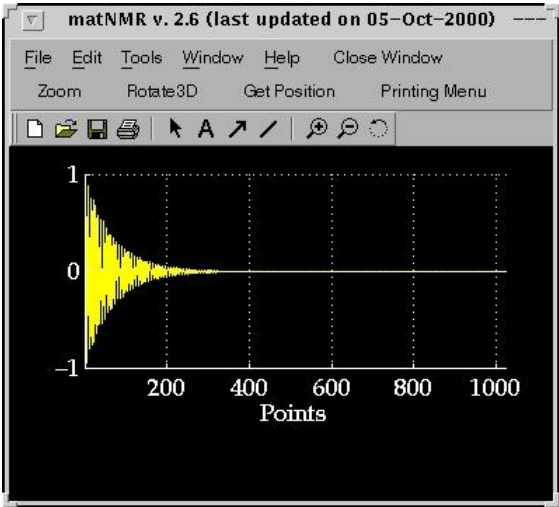
5.3.2 Buttons

Name	Function
Zoom	Switches the zoom function on or off.
Verbose	Determines the level of output generated by the fitting algorithms. This can either be never, after every iteration or only after finishing the fit.
Tolerance	Termination tolerance for the fit.
Max. Nr. Iter.	Maximum number of iterations for each fit.
Update display	Defines whether the plot should be updated after each iteration or only after the fit has finished.
Noise level	In order for the calculated error to be a χ^2 the standard deviation of the noise must be entered here. This can be taken from the 1D spectrum. A proper value for the noise level should ideally result in an error of 1 for a perfect fit, i.e. the residual is nothing more than the standard deviation of the noise then.
SIMPLEX quality	Number of powder-averaging points for the SIMPLEX algorithm. A two-angle ZCW grid is used during the simulation. Selecting "OFF" will result in the algorithm not being used. Increase this value for a more accurate result.
Gradient quality	Number of powder-averaging points for the gradient algorithm. A two-angle ZCW grid is used during the simulation. Selecting "OFF" will result in the algorithm not being used. Increase this value for a more accurate result.
# gamma angles	Defines the number of angles gamma for the powder averaging. This should be larger than the maximum sideband number that needs to be extracted. 50 seems to be a good value for most practical cases (up to +/- 10 sidebands) as it results in a stable simulation. Increase this value for a more accurate result.
Print	Prints the current figure window to the the default printer as defined in the \$matlabroot/toolbox/local/printopt.m file. No UI-controls are printed as the parameter values are put into the plot as white text. The resulting plot can be rather ugly as sometimes the text is right on top of the fitted spectrum.
Refresh	Uploads the current 1D spectrum from the main window into the fitting routine.

Simulate	Performs a simulation of a sideband manifold based on the parameters given in the UIcontrols. This allows finding a good initial estimate of the parameters, which is vital for any non-linear fitting algorithm.
Fit	Starts the fit.
Close	Closes the figure window. No variables are deleted so they can always be accessed afterwards.
Carrier	Carrier frequency of the spectrum from which the sidebands were taken. This value is not fitted and should be taken over from the main window.
Spinning speed	The spinning frequency for the spectrum from which the sidebands were taken. This is not a fit parameter.
Sigma_iso	The isotropic shift in ppm. This is not a fit parameter and is only needed to calculate the principal components sigma_11, sigma_22 and sigma_33.
delta	The anisotropy of the fitted CSA tensor in ppm.
eta	The asymmetry of the fitted CSA tensor.
Intensity	The intensity is merely a scaling factor.

5.3.3 Menubar

Name	Function								
Load parameters	Loads a variable, with the same format as the QSSAFitResults, from the workspace and puts its values into the UI-controls. Please see paragraph 5.3.1, " General Appearance " for more information on the format.								
View fit results	Takes a variable from the workspace, with the same format as the QSSAFitResults, and outputs the fitting results to the MATLAB command window. Furthermore, the parameters are written into the UIcontrols in the window and the data and fit are shown graphically. Please see paragraph 5.3.1, " General Appearance " for more information on the format.								
Plot manipulations	This menu allows changing of various axis properties. More explanation concerning these can be found in the MATLAB documentation. Note the difference between changing plot properties using this menu and by changing the properties using the options menu (see also paragraph 2.2, " Setting up matNMR "): changes made with these functions only affect the current plot and not the general matNMR settings.								
Create Output	<table> <tr> <th>Name</th><th>Function</th></tr> <tr> <td>Printing Menu</td><td>The printing menu is a general feature available from all matNMR windows that is used for printing. Using a separate window the MATLAB <i>print</i> command (see the MATLAB documentation for more information on print) can be executed. More information on the printing menu is given in paragraph 7.1, "Printing menu".</td></tr> <tr> <td>Save figure to disk</td><td>This function will store the current window as a binary figure on disk as an executable m-file. This will allow the user to recall the exact figure by retyping the name of the m-file on the command line. This uses the Matlab mfig data format. See the Matlab manual for more information on the "saveas" command</td></tr> <tr> <td>Copy figure</td><td>This function will copy the current view of the main window into a separate window, but without the UI-controls. This will give something like:</td></tr> </table>	Name	Function	Printing Menu	The printing menu is a general feature available from all matNMR windows that is used for printing. Using a separate window the MATLAB <i>print</i> command (see the MATLAB documentation for more information on print) can be executed. More information on the printing menu is given in paragraph 7.1, " Printing menu ".	Save figure to disk	This function will store the current window as a binary figure on disk as an executable m-file. This will allow the user to recall the exact figure by retyping the name of the m-file on the command line. This uses the Matlab mfig data format. See the Matlab manual for more information on the "saveas" command	Copy figure	This function will copy the current view of the main window into a separate window, but without the UI-controls. This will give something like:
Name	Function								
Printing Menu	The printing menu is a general feature available from all matNMR windows that is used for printing. Using a separate window the MATLAB <i>print</i> command (see the MATLAB documentation for more information on print) can be executed. More information on the printing menu is given in paragraph 7.1, " Printing menu ".								
Save figure to disk	This function will store the current window as a binary figure on disk as an executable m-file. This will allow the user to recall the exact figure by retyping the name of the m-file on the command line. This uses the Matlab mfig data format. See the Matlab manual for more information on the "saveas" command								
Copy figure	This function will copy the current view of the main window into a separate window, but without the UI-controls. This will give something like:								



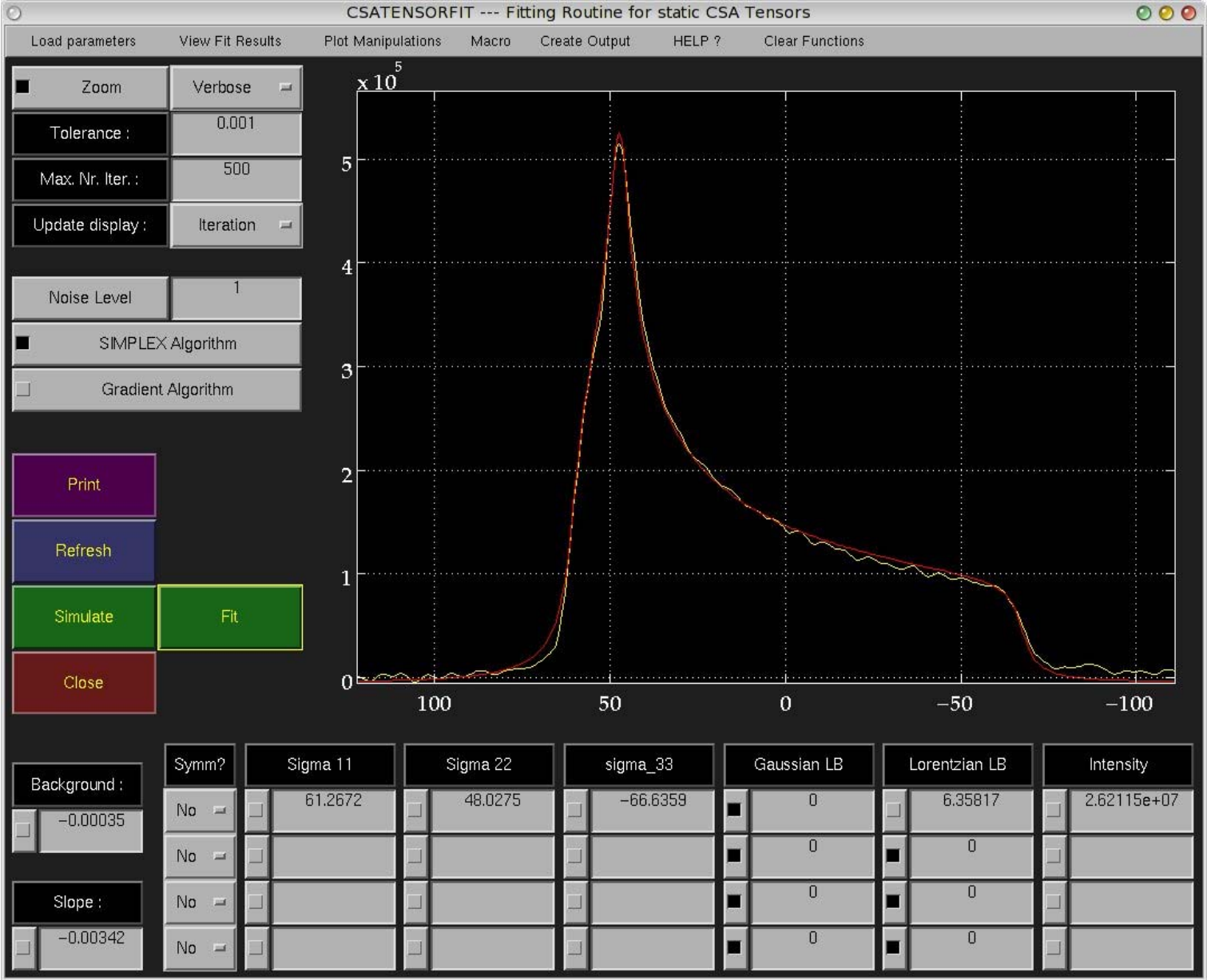
The menubar of this new window contains the standard MATLAB menubar which has some useful features for printing and saving the figure. It also offers the possibility to change axis properties and stuff (More information on this menubar can be found in the MATLAB documentation). Additional features in the menubar are the MATLAB zoom and rotate3D functions, the matNMR "Get Position" (for more information see paragraph 5.1, "General appearance") and the printing menu (for more information see paragraph 7.1, "Printing menu").

Macro	Name	Function
	Start recording macro	Starts the recording of a new macro. An input will appear to ask whether only processing, only plotting or all actions must be stored in the macro. Recording continues until the stop command is given (see below). Note that for user-commands, i.e. commands that are usually executed from the MATLAB command line, to be put into the macro the "execute user-command" function (see below) must be used! (see also paragraph 4.4, "Using macro's")
	Stop recording macro	Stop the recording of a macro. An input window will appear to ask for a name of a variable in the workspace to store the new macro in. (see also paragraph 4.4, "Using macro's")
	Execute macro	This routine executes a macro. (see also paragraph 4.4, "Using macro's")
Help?	Execute macro stepwise	This routine executes a macro stepwise. The same window as used for "Reprocess from history stepwise" (see above) will appear .
	Besides the simple "About matNMR" and "Copyright" notices this contains a direct link to this manual from inside MATLAB. The browser, defined in the \$matlabroot/toolbox/local/docopt.m file, will be opened and the matNMR manual will appear (See the MATLAB documentation for more information on the MATLAB web and docopt commands). As the manual is included in the matNMR distribution reading the manual should be faster than using the matNMR website.	
Clear functions	This performs the MATLAB "clear functions" command. All compiled functions are deleted from memory forcing MATLAB to recompile each function again upon running it. This is useful when working with your own m-files (or changing the matNMR code) to force MATLAB to use the altered function (See the MATLAB documentation for more information on the MATLAB clear command). Furthermore the mouse pointer is set to an arrow head.	

5.4 Fitting of CSA tensors from static powder patterns

5.4.1 General appearance

MatNMR offers a fitting routine for CSA tensors from static powder spectra which can be accessed from the "1D Processing" entry in the menubar. The following figure window will appear:



The routine can be operated using the UI-controls in the figure window. The functions of these will be explained in [paragraph 5.4.2](#) just below. The auxiliary functions in the menubar are described in [paragraph 5.4.3](#) below.

Upon starting the routine the current 1D spectrum (main window) is taken and plotted. It is assumed that the unit of the axis has been set to ppm.

To fit the powder pattern only three principal components are required, an intensity factor and the degree of linebroadening. The routine then calculates the spectrum using the analytical

20 of 42

11/23/07 06:02 PM

lineshape determined by Bloembergen (1949).

As holds for all non-linear fits a good initial estimate of the parameters is required for an efficient fit convergence. To facilitate finding such an estimate the "Simulate" button may be used. After having defined the initial guesses the fit can be started with the "Fit" button. Should a parameter not be fitted but kept constant at its current value, then this can be done by selecting the check button (with the square block in it) belonging to the parameter.

Two algorithms may be used for fitting: SIMPLEX and a gradient algorithm. Typically the SIMPLEX algorithm is used for coarse optimization and is followed by a gradient minimization to finish off the analysis.

When the fit has finished by default the results are written into the variable QCSAFitResults which can be read out by hand by the user (Note that this variable will be deleted when matNMR is stopped! This means it must be renamed, if it has to be kept in the workspace for later use, before stopping matNMR). Alternatively, from the "View Fit Results" entry in the menubar a new name for putting the results in can be defined. To be able to access this variable the user must make the variable global by typing "global VariableName" at the MATLAB prompt. A message is written to make sure the user doesn't forget this. The format of the parameters is as follows:

- for each line:
 - sigma11
 - sigma22
 - sigma33
 - Gaussian LB
 - Lorentzian LB
 - Intensity
- background
- slope

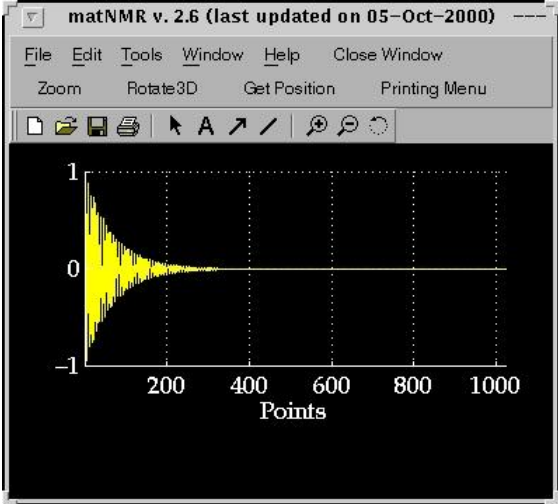
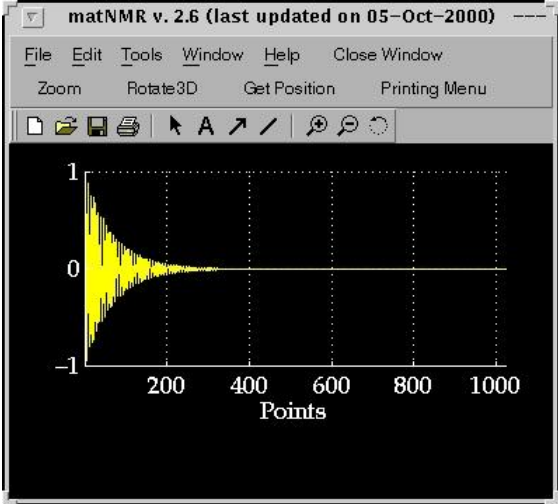
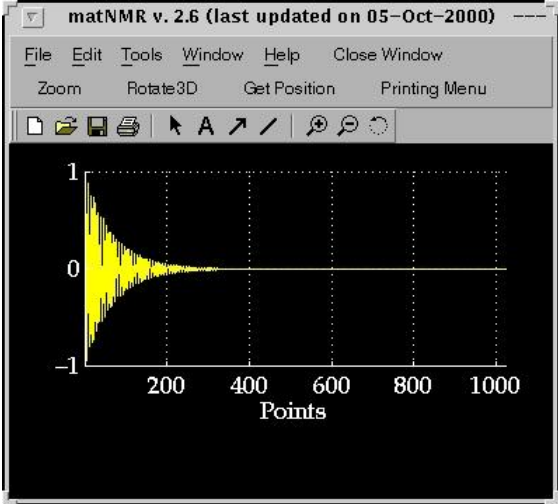
All these numbers are put in a single row. This variable is combined with the original data, the axis to the data, the fitted spectrum and the error value into a structure variable. The fields in this variable are "Parameters", "Fit", "Data", "Axis" and "Error".

5.4.2 Buttons

Name	Function
Zoom	Switches the zoom function on or off.
Verbose	Determines the level of output generated by the fitting algorithms. This can either be never, after every iteration or only after finishing the fit.
Tolerance	Termination tolerance for the fit.
Max. Nr. Iter.	Maximum number of iterations for each fit.
Update display	Defines whether the plot should be updated after each iteration or only after the fit has finished.
Noise level	In order for the calculated error to be a χ^2 the standard deviation of the noise must be entered here. This can be taken from the 1D spectrum. A proper value for the noise level should ideally result in an error of 1 for a perfect fit, i.e. the residual is nothing more than the standard deviation of the noise then.
SIMPLEX algorithm	Selects whether the SIMPLEX algorithm should be used for the fit.
Gradient algorithm	Selects whether the Gradient algorithm should be used for the fit.
Print	Prints the current figure window to the the default printer as defined in the \$matlabroot/toolbox/local/printopt.m file. No UI-controls are printed as the parameter values are put into the plot as white text. The resulting plot can be rather ugly as sometimes the text is right on top of the fitted spectrum.
Refresh	Uploads the current 1D spectrum from the main window into the fitting routine.
Simulate	Performs a simulation of a sideband manifold based on the parameters given in the UIcontrols. This allows finding a good initial estimate of the parameters, which is vital for any non-linear fitting algorithm.
Fit	Starts the fit.
Close	Closes the figure window. No variables are deleted so they can always be accessed afterwards.
Background	A constant baseline offset can be fitted to the spectrum.
Slope	A linearly-sloping baseline offset can be fitted to the spectrum.
Symm?	A tensor can be forced to be symmetric causing two tensor components to be equal.
Sigma11	One of the three tensor components that are fitted.
Sigma22	One of the three tensor components that are fitted.

Sigma33	One of the three tensor components that are fitted.
Gaussian LB	Degree of Gausiaan linebroadening (typically indicative of heterogeneous broadening).
Lorentzian LB	Degree of Lorentzian linebroadening.
Intensity	The intensity is merely a scaling factor.

5.4.3 Menubar

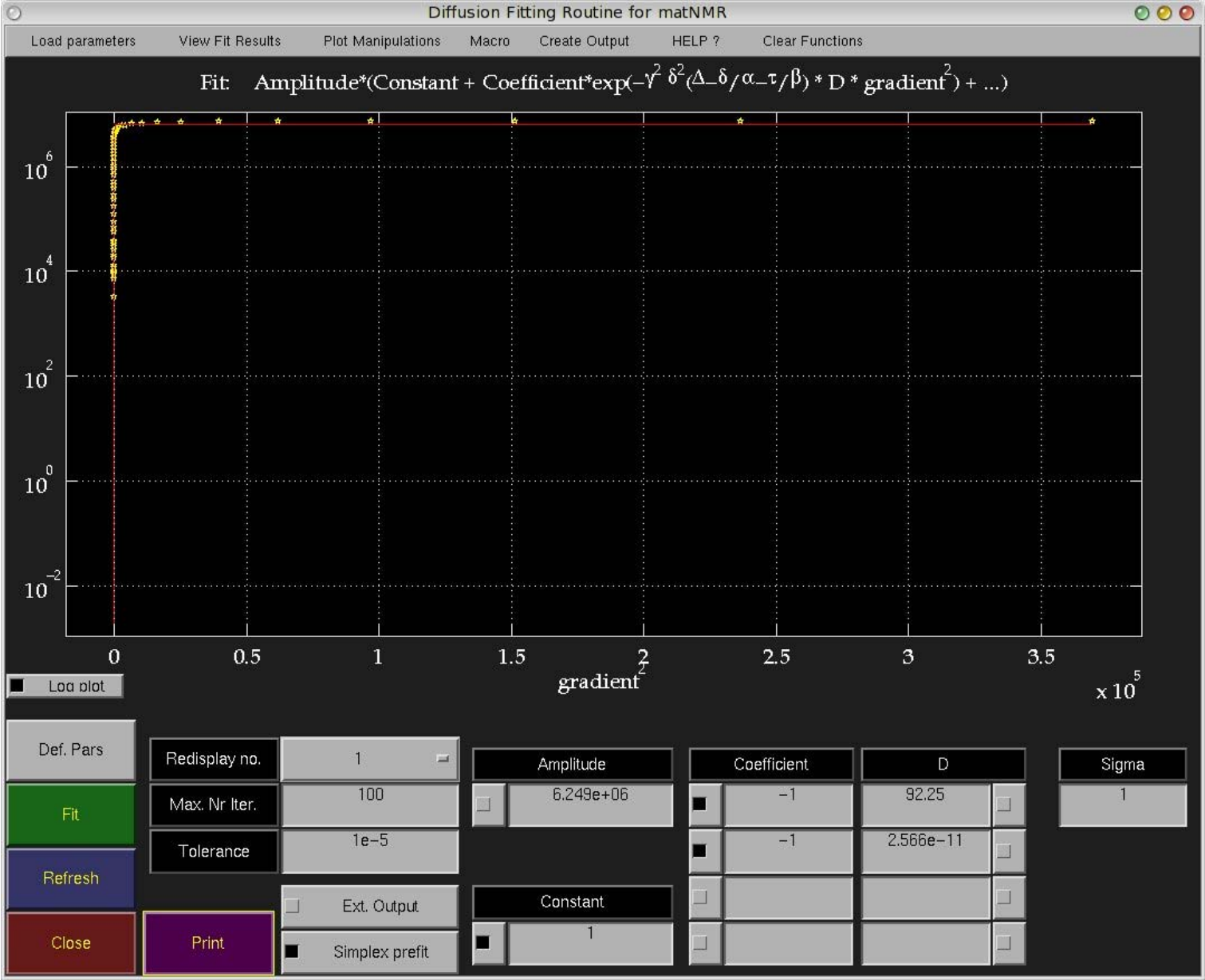
Name	Function								
Load parameters	Loads a variable, with the same format as the QCSAFitResults, from the workspace and puts its values into the UI-controls. Please see paragraph 5.4.1, "General Appearance" for more information on the format.								
View fit results	Takes a variable from the workspace, with the same format as the QCSAFitResults, and outputs the fitting results to the MATLAB command window. Furthermore, the parameters are written into the UIcontrols in the window and the data and fit are shown graphically. Please see paragraph 5.4.1, "General Appearance" for more information on the format.								
Plot manipulations	This menu allows changing of various axis properties. More explanation concerning these can be found in the MATLAB documentation. Note the difference between changing plot properties using this menu and by changing the properties using the options menu (see also paragraph 2.2, "Setting up matNMR"): changes made with these functions only affect the current plot and not the general matNMR settings.								
Create Output	<table><tr><th>Name</th><th>Function</th></tr><tr><td>Printing Menu</td><td>The printing menu is a general feature available from all matNMR windows that is used for printing. Using a separate window the MATLAB <i>print</i> command (see the MATLAB documentation for more information on print) can be executed. More information on the printing menu is given in paragraph 7.1, "Printing menu".</td></tr><tr><td>Save figure to disk</td><td>This function will store the current window as a binary figure on disk as an executable m-file. This will allow the user to recall the exact figure by retyping the name of the m-file on the command line. This uses the Matlab mfig data format. See the Matlab manual for more information on the "saveas" command</td></tr><tr><td>Copy figure</td><td><p>This function will copy the current view of the main window into a separate window, but without the UI-controls. This will give something like:</p></td></tr></table>	Name	Function	Printing Menu	The printing menu is a general feature available from all matNMR windows that is used for printing. Using a separate window the MATLAB <i>print</i> command (see the MATLAB documentation for more information on print) can be executed. More information on the printing menu is given in paragraph 7.1, "Printing menu".	Save figure to disk	This function will store the current window as a binary figure on disk as an executable m-file. This will allow the user to recall the exact figure by retyping the name of the m-file on the command line. This uses the Matlab mfig data format. See the Matlab manual for more information on the "saveas" command	Copy figure	<p>This function will copy the current view of the main window into a separate window, but without the UI-controls. This will give something like:</p> 
Name	Function								
Printing Menu	The printing menu is a general feature available from all matNMR windows that is used for printing. Using a separate window the MATLAB <i>print</i> command (see the MATLAB documentation for more information on print) can be executed. More information on the printing menu is given in paragraph 7.1, "Printing menu".								
Save figure to disk	This function will store the current window as a binary figure on disk as an executable m-file. This will allow the user to recall the exact figure by retyping the name of the m-file on the command line. This uses the Matlab mfig data format. See the Matlab manual for more information on the "saveas" command								
Copy figure	<p>This function will copy the current view of the main window into a separate window, but without the UI-controls. This will give something like:</p> 								

		The menubar of this new window contains the standard MATLAB menubar which has some useful features for printing and saving the figure. It also offers the possibility to change axis properties and stuff (More information on this menubar can be found in the MATLAB documentation). Additional features in the menubar are the MATLAB zoom and rotate3D functions, the matNMR "Get Position" (for more information see paragraph 5.1, " General appearance ") and the printing menu (for more information see paragraph 7.1, " Printing menu ").
Macro	Name	Function
	Start recording macro	Starts the recording of a new macro. An input will appear to ask whether only processing, only plotting or all actions must be stored in the macro. Recording continues until the stop command is given (see below). Note that for user-commands, i.e. commands that are usually executed from the MATLAB command line, to be put into the macro the "execute user-command" function (see below) must be used! (see also paragraph 4.4, " Using macro's ")
	Stop recording macro	Stop the recording of a macro. An input window will appear to ask for a name of a variable in the workspace to store the new macro in. (see also paragraph 4.4, " Using macro's ")
	Execute macro	This routine executes a macro. (see also paragraph 4.4, " Using macro's ")
	Execute macro stepwise	This routine executes a macro stepwise. The same window as used for "Reprocess from history stepwise" (see above) will appear .
Help?	Besides the simple "About matNMR" and "Copyright" notices this contains a direct link to this manual from inside MATLAB. The browser, defined in the \$matlabroot/toolbox/local/docopt.m file, will be opened and the matNMR manual will appear (See the MATLAB documentation for more information on the MATLAB web and docopt commands). As the manual is included in the matNMR distribution reading the manual should be faster than using the matNMR website.	
Clear functions	This performs the MATLAB "clear functions" command. All compiled functions are deleted from memory forcing MATLAB to recompile each function again upon running it. This is useful when working with your own m-files (or changing the matNMR code) to force MATLAB to use the altered function (See the MATLAB documentation for more information on the MATLAB clear command). Furthermore the mouse pointer is set to an arrow head.	

5.5 Diffusion curve fitting

5.5.1 General appearance

MatNMR offers a basic isotropic pulsed-field gradient (PFG) curve fitting routine from the "1D Processing" entry in the menubar. The following figure window will appear:



The routine can be operated using the UI-controls in the figure window. The functions of these will be explained in [paragraph 5.5.2](#) just below. The auxiliary functions in the menubar are described in [paragraph 5.5.3](#) below.

Upon starting the routine the current 1D spectrum (main window) is taken and plotted. It is assumed that the axis ruler has been changed to a gradient axis, see [paragraph 5.2.5](#).
NOTE: Gradient axes are defined in T/m and NOT in G/cm. This means you may have to divide the gradient values from your software by 100 to obtain the proper values!!

For a more convenient view on exponentials the vertical axis is plotted on a logarithmic scale. This can easily be changed using the "scaling types" item in the "plot manipulations" menu on the menubar.

To fit a (multi-)exponential diffusion curve one first has to provide proper values for the three constants: the gyromagnetic ratio, the gradient spacing and the gradient duration. By pushing the "Def. Pars." button an input window will be shown where the values can be given.

The next step is to make an initial guess for all exponential functions that need to be fitted to it. This can be done by typing the coefficient and the diffusion constant (D) for each fraction by hand into the corresponding UI-controls in the window. Also the overall amplitude and a constant may be defined. The routine will include a fraction into the fit if a coefficient value is specified. Note that all parameters are given in the unit of the axis vector belonging to the current 1D spectrum. This means that the axis vector has to be defined before starting the peak-fitting routine! A total of 4 exponentials can be fitted simultaneously.

After having defined the initial guesses the fit can be started with the "Fit" button. Should a parameter not be fitted but kept constant at its current value, then this can be done by selecting the check button (with the square block in it) belonging to the parameter.

When the fit has finished by default the results are written into the variable QDiffFitResults which can be read out by hand by the user (Note that this variable will be deleted when matNMR is stopped! This means it must be renamed, if it has to be kept in the workspace for later use, before stopping matNMR). Alternatively, from the "View Fit Results" entry in the menubar a new name for putting the results in can be defined. To be able to access this variable the user must make the variable global by typing "global VariableName" at the MATLAB prompt. A message is written to make sure the user doesn't forget this. The format of the parameters is as follows:

- nr of iterations
- chi2 for this column
- fit parameters :
 - for each exponential:
 - Coefficient
 - Diffusion constant (D)
 - constant
 - amplitude

All these numbers are put in a single row. This variable is combined with the original data, the axis to the data, the fitted spectrum and the error value into a structure variable. The fields in this variable are "Parameters", "Fit", "Data", "Axis" and "Error".

Last, in case the initial parameters are very far off the "Simplex prefit" check button may be activated. This makes matNMR use a SIMPLEX algorithm first before the gradient algorithm. By default this option is active.

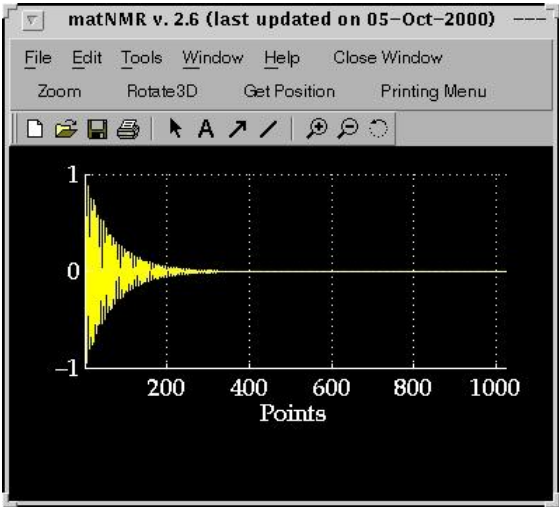
5.5.2 Buttons

Name	Function
log plot	Switches between a linear and a logarithmic y-axis.
Def. Pars.	Allows entering the gyromagnetic ratio, the gradient spacing and the gradient duration and optionally the length of the delay used in a bipolar gradient experiment to allow for the switching of the gradients, all of which are needed for the fit. An input window will appear in which these parameters can be entered. Alternatively, if the data have been recorded on a Bruker spectrometer using the <i>diff</i> script provided for setting up diffusion experiments, instead of typing the values for the gradient spacing and the gradient duration one may select "READ Bruker" in the input window. Gamma still needs to be entered by hand though! NOTE that this only works from XWinNMR 3.0 since before 3.0 the diff script was different. Also, it may mean that this will change at some point. MatNMR will then ask for the corresponding "acqus" file for the data, from which the parameters d20, p17 and p18 are read. d20 corresponds to big DELTA. p17 is the gradient rise time and p18 the length of time at which the gradient is constant.
Fit	Starts the fit.
Refresh	Uploads the current 1D spectrum from the main window into the fitting routine.
Close	Closes the figure window. No variables are deleted so they can always be accessed afterwards.
Print	Prints the current figure window to the the default printer as defined in the \$matlabroot/toolbox/local/printopt.m file. No UI-controls are printed as the parameter values are put into the plot as white text. The resulting plot can be rather ugly as sometimes the text is right on top of the fitted spectrum.
Redisplay no.	Defines after how many iterations the plot must be redrawn. Set this to a high value to save time.
Max. Nr. Iter.	Maximum number of iterations for each fit.
Tolerance	Termination tolerance for the fit.
Sigma	Measure for the noise in the data. When properly defined the error should become a true χ^2 .

Ext. Output	Flag for whether an extended output should be given. A detailed error analysis is presented. NOTE: the basic fitting routine has not been programmed by me and I have never checked the accuracy of the error analysis!
SIMPLEX prefit	Flag for whether the fit should use a SIMPLEX algorithm before the gradient algorithm. This reduces the common problem that the gradient algorithm doesn't find a proper minimum when the parameters are too far off.
Amplitude	Overall amplitude that scales the fit.
Constant	A constant.
Coefficient	Amplitude factor for the exponential fraction.
D	Diffusion constant for the exponential fraction.

5.5.3 Menubar

Name	Function								
Load parameters	Loads a variable, with the same format as the QDiffFitResults, from the workspace and puts its values into the UI-controls. Please see paragraph 5.5.1, " General Appearance " for more information on the format.								
View fit results	Takes a variable from the workspace, with the same format as the QDiffFitResults, and outputs the fitting results to the MATLAB command window. Furthermore, the parameters are written into the UIcontrols in the window and the data and fit are shown graphically. Please see paragraph 5.5.1, " General Appearance " for more information on the format.								
Plot manipulations	This menu allows changing of various axis properties. More explanation concerning these can be found in the MATLAB documentation. Note the difference between changing plot properties using this menu and by changing the properties using the options menu (see also paragraph 2.2, " Setting up matNMR "): changes made with these functions only affect the current plot and not the general matNMR settings.								
Create Output	<table> <tr> <th>Name</th><th>Function</th></tr> <tr> <td>Printing Menu</td><td>The printing menu is a general feature available from all matNMR windows that is used for printing. Using a separate window the MATLAB <i>print</i> command (see the MATLAB documentation for more information on print) can be executed. More information on the printing menu is given in paragraph 7.1, "Printing menu".</td></tr> <tr> <td>Save figure to disk</td><td>This function will store the current window as a binary figure on disk as an executable m-file. This will allow the user to recall the exact figure by retyping the name of the m-file on the command line. This uses the Matlab mfig data format. See the Matlab manual for more information on the "saveas" command</td></tr> <tr> <td>Copy figure</td><td>This function will copy the current view of the main window into a separate window, but without the UI-controls. This will give something like:</td></tr> </table>	Name	Function	Printing Menu	The printing menu is a general feature available from all matNMR windows that is used for printing. Using a separate window the MATLAB <i>print</i> command (see the MATLAB documentation for more information on print) can be executed. More information on the printing menu is given in paragraph 7.1, " Printing menu ".	Save figure to disk	This function will store the current window as a binary figure on disk as an executable m-file. This will allow the user to recall the exact figure by retyping the name of the m-file on the command line. This uses the Matlab mfig data format. See the Matlab manual for more information on the "saveas" command	Copy figure	This function will copy the current view of the main window into a separate window, but without the UI-controls. This will give something like:
Name	Function								
Printing Menu	The printing menu is a general feature available from all matNMR windows that is used for printing. Using a separate window the MATLAB <i>print</i> command (see the MATLAB documentation for more information on print) can be executed. More information on the printing menu is given in paragraph 7.1, " Printing menu ".								
Save figure to disk	This function will store the current window as a binary figure on disk as an executable m-file. This will allow the user to recall the exact figure by retyping the name of the m-file on the command line. This uses the Matlab mfig data format. See the Matlab manual for more information on the "saveas" command								
Copy figure	This function will copy the current view of the main window into a separate window, but without the UI-controls. This will give something like:								



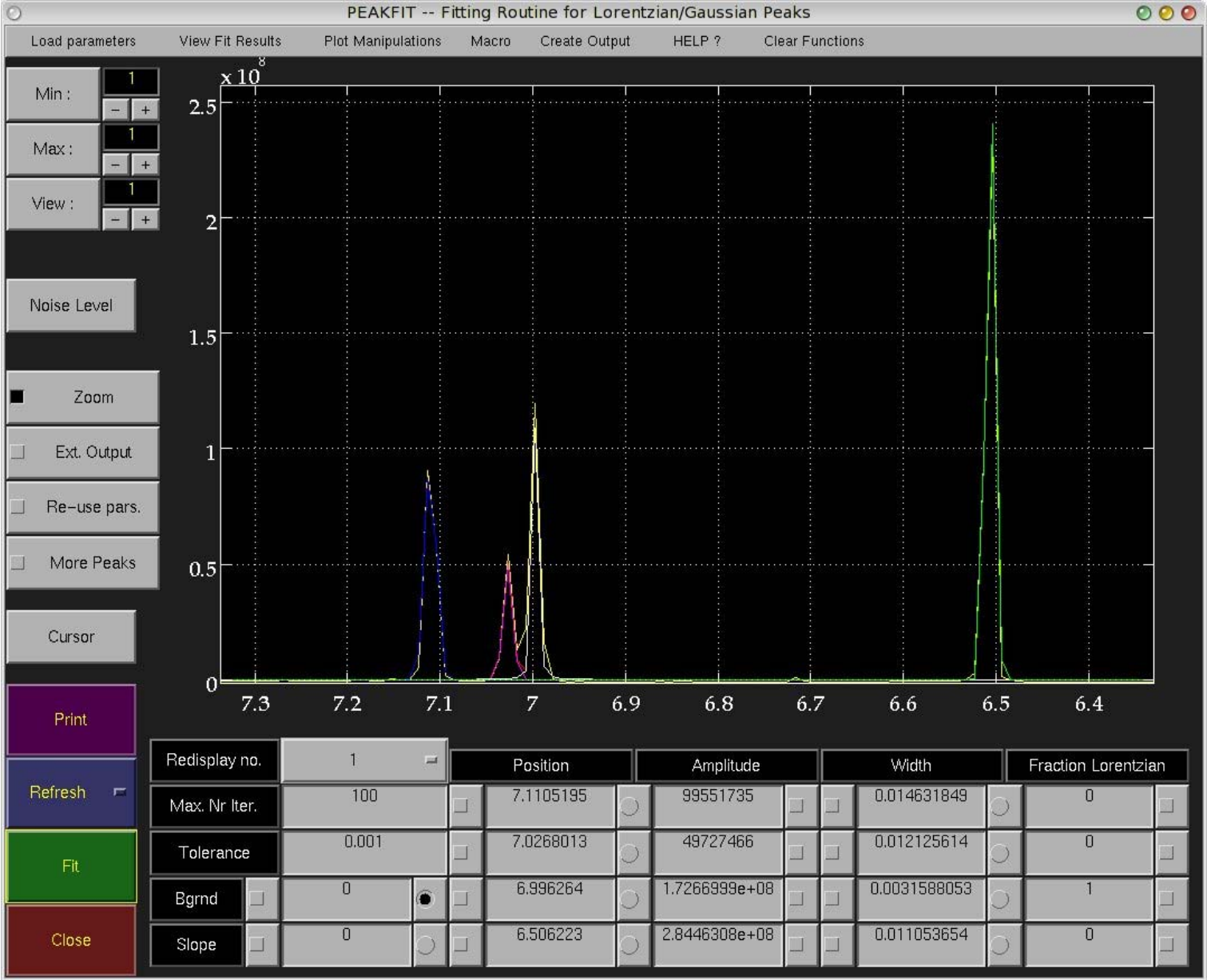
The menubar of this new window contains the standard MATLAB menubar which has some useful features for printing and saving the figure. It also offers the possibility to change axis properties and stuff (More information on this menubar can be found in the MATLAB documentation). Additional features in the menubar are the MATLAB zoom and rotate3D functions, the matNMR "Get Position" (for more information see paragraph 5.1, "General appearance") and the printing menu (for more information see paragraph 7.1, "Printing menu").

Macro	Name	Function
	Start recording macro	Starts the recording of a new macro. An input will appear to ask whether only processing, only plotting or all actions must be stored in the macro. Recording continues until the stop command is given (see below). Note that for user-commands, i.e. commands that are usually executed from the MATLAB command line, to be put into the macro the "execute user-command" function (see below) must be used! (see also paragraph 4.4, "Using macro's")
	Stop recording macro	Stop the recording of a macro. An input window will appear to ask for a name of a variable in the workspace to store the new macro in. (see also paragraph 4.4, "Using macro's")
	Execute macro	This routine executes a macro. (see also paragraph 4.4, "Using macro's")
Help?	Execute macro stepwise	This routine executes a macro stepwise. The same window as used for "Reprocess from history stepwise" (see above) will appear .
	Besides the simple "About matNMR" and "Copyright" notices this contains a direct link to this manual from inside MATLAB. The browser, defined in the \$matlabroot/toolbox/local/docopt.m file, will be opened and the matNMR manual will appear (See the MATLAB documentation for more information on the MATLAB web and docopt commands). As the manual is included in the matNMR distribution reading the manual should be faster than using the matNMR website.	
Clear functions	This performs the MATLAB "clear functions" command. All compiled functions are deleted from memory forcing MATLAB to recompile each function again upon running it. This is useful when working with your own m-files (or changing the matNMR code) to force MATLAB to use the altered function (See the MATLAB documentation for more information on the MATLAB clear command). Furthermore the mouse pointer is set to an arrow head.	

5.6 Peak Deconvolution (mixed Gaussian/Lorentzian lineshapes)

5.6.1 General appearance

MatNMR offers a basic peak-fitting routine from both "1D Processing" and "2D Processing" entries in the menubar. The following figure window will appear:



The routine can be operated using the UI-controls in the figure window. The functions of these will be explained in [paragraph 5.6.2](#) just below. The auxiliary functions in the menubar are described in [paragraph 5.6.3](#) below.

To fit peaks one first has to make an initial guess for the peak parameters for all peaks that need to be fitted. This can be done in two ways: 1) typing the peak center, its amplitude and width and the fraction of Lorentzian contribution to the Voigt lineshape by hand into the corresponding UI-controls in the window or 2) by using the "Cursor" button. This latter function is described in [paragraph 5.6.2](#) below. In any case the routine will think that a peak is a peak if there is a value written into the "peak width" button. Note that the peak parameters are given

29 of 4211/23/07 06:02 PM

in the unit of the axis vector belonging to the current 1D spectrum. This means that the axis vector has to be defined before starting the peak-fitting routine! A total of 36 peaks can be fitted simultaneously (for more UI-controls press the "More Peaks" button). In case the baseline is slightly distorted a baseline offset and linear slope may be included in the fit. In case the baseline is strongly distorted it is advised to first perform a baseline correction.

After having defined the initial guesses the fit can be started with the "Fit" button. Should a parameter not be fitted but kept constant at its current value, then this can be done by selecting the check button (with the square block in it) belonging to the parameter.

When the fit has finished the results are written into the variable QFitResults which can be read out by hand by the user (Note that this variable will be deleted when matNMR is stopped! This means it must be renamed, if it has to be kept in the workspace for later use, before stopping matNMR). Alternatively, from the "View Fit Results" entry in the menubar a new name for putting the results in can be defined. To be able to access this variable the user must make the variable global by typing "global VariableName" at the MATLAB prompt. A message is written to make sure the user doesn't forget this. The format of the parameters is as follows:

- fit parameters :
 - for each peak:
 - Center of peak
 - Amplitude of peak
 - Width of peak
 - Fraction of Lorentzian contribution to the Voigt lineshape
 - background
 - slope

All these numbers are put in a single row. This variable is combined with the original data, the axis to the data, the fitted spectrum and the error value into a structure variable. The fields in this variable are "Parameters", "Fit", "Integrals", "Data", "Axis" and "Error". In case of a 2D being fitted, the structure variable is an array of structures of the same layout.

There is a distinct difference between starting the peak-fitting routine from the "1D Processing" or from the "2D Processing" entry in the menubar: the 1D version will only take the current 1D spectrum or row/column from the 2D into the peak-fitting routine. The 2D version will take the whole 2D in which allows fitting a series of 1D spectra automatically.

Last, as peak fitting can be quite time consuming it is important to reduce the number of points in the spectrum. Therefore be sure to zoom into the region of interest properly before starting the peak-fitting routine. The peak-fitting routine takes the current zoom limits into account when reading the spectrum into the routine. A further reduction in time can be made by reducing the screen output. This can be done by selecting a lower refresh rate using the "redisplay no." button.

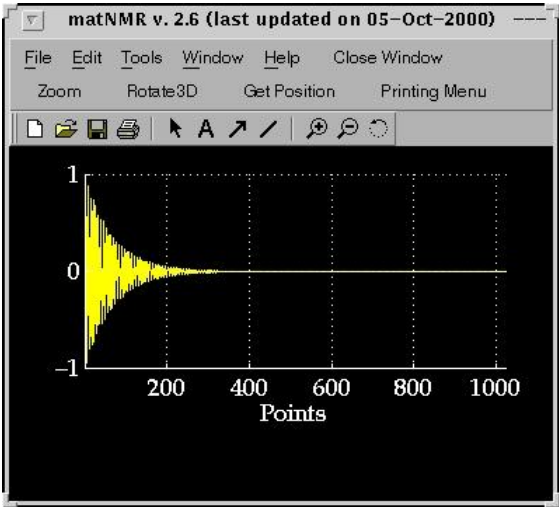
5.6.2 Buttons

Name	Function
Min / Max	Specify the range of spectra from the current 2D (i.e. series of 1D) spectrum that need to be fitted. Numbers can be entered by hand or by clicking the "-" and "+" buttons. Only defined for 2D spectra.
View	Specifies which row/column of the current 2D (i.e. series of 1D) spectrum must be plotted in the figure window. Select a row/column that allows defining the peaks easily. The fit will still start at the row/column as defined by the "Min" button. Only defined for 2D spectra.
Noise level	May be used to obtain a proper χ^2 . Define a signal-free range in the spectrum (preferably without a baseline distortion!) by clicking left and right of the range in the plot. The standard deviation of the use will be used to normalize the χ^2 .
Zoom	Allows zooming into the plot which can be useful for defining the peaks. This uses the standard MATLAB <i>zoom</i> routine.
Ext. Output	Flag for whether an extended output should be given. A detailed error analysis is presented. NOTE: the basic peak-fitting routine has not been programmed by me and I have never checked the accuracy of the error analysis!
Re-use pars.	Flag whether to start each fit of a 2D spectrum with the same initial parameters. If not selected each new row/column will start with the parameters from the last row/column.
More peaks	Opens an additional window with UI-controls for defining peaks. Up to 24 peaks can be fitted simultaneously.
Cursor	Allows defining the peaks using the mouse. Before starting be sure to select the proper radio button (with the diamond in it) belonging to the parameter that needs to be defined. The peak center and amplitudes are defined by a single mouse click (left button). The peak width at half height is defined by two further mouse clicks, one left and one right of the peak center. After a parameter is defined the next radio button is activated. Should another parameter be changed just change the radio button (it is not necessary to stop the "cursor" routine for this!). When all parameters have been defined properly the routine must be stopped by clicking outside the axis or by using the right or middle mouse button.
Print	Prints the current figure window to the the default printer as defined in the \$matlabroot/toolbox/local/printopt.m file. No UI-controls are printed except the parameter buttons. The resulting plot can be rather ugly as the UI-controls don't print nicely.

Refresh	Uploads the current 1D or 2D spectrum from the main window into the peak-fitting routine. Note that this function only checks the current dimension QmatNMR.Dim for determining whether to upload a 1D or 2D spectrum. It does not check whether the previous fitting mode was 1D or 2D. In case only the current row/column of the current 2D spectrum in the main window must be uploaded, use the "Peak fitting" entry from the "1D Processing" menu in the main window.
Fit	Starts the fit.
Close	Closes the figure window. No variables are deleted so they can always be accessed afterwards.
Redisplay no.	Defines after how many iterations the plot must be redrawn. Set this to a high value to save time.
Max. Nr. Iter.	Maximum number of iterations for each fit.
Tolerance	Termination tolerance for the fit.
Backgrnd	Background offset level that is included in the fit.
Slope	Linear slope in the background that is included in the fit.
Xpos	Center of a peak in the unit of the axis vector.
Ampl	Amplitude of a peak.
Width	Full width at half height (FWHH) of a peak in the unit of the axis vector.
Frac. Lor.	Amount of Lorentzian contribution to the Voigt lineshape ranging from 0 to 1.

5.6.3 Menubar

Name	Function								
Load parameters	Loads a variable, with the same format as the QFitResults, from the workspace and puts its values into the UI-controls. Please see paragraph 5.6.1, " General Appearance " for more information on the format.								
View fit results	Takes a variable from the workspace, with the same format as the QFitResults, and outputs the fitting results to the MATLAB command window. Furthermore, the parameters are written into the UIcontrols in the window and the data and fit are shown graphically. Please see paragraph 5.6.1, " General Appearance " for more information on the format.								
Plot manipulations	This menu allows changing of various axis properties. More explanation concerning these can be found in the MATLAB documentation. Note the difference between changing plot properties using this menu and by changing the properties using the options menu (see also paragraph 2.2, " Setting up matNMR "): changes made with these functions only affect the current plot and not the general matNMR settings.								
Create Output	<table> <tr> <th>Name</th><th>Function</th></tr> <tr> <td>Printing Menu</td><td>The printing menu is a general feature available from all matNMR windows that is used for printing. Using a separate window the MATLAB <i>print</i> command (see the MATLAB documentation for more information on print) can be executed. More information on the printing menu is given in paragraph 7.1, "Printing menu".</td></tr> <tr> <td>Save figure to disk</td><td>This function will store the current window as a binary figure on disk as an executable m-file. This will allow the user to recall the exact figure by retyping the name of the m-file on the command line. This uses the Matlab mfig data format. See the Matlab manual for more information on the "saveas" command</td></tr> <tr> <td>Copy figure</td><td>This function will copy the current view of the main window into a separate window, but without the UI-controls. This will give something like:</td></tr> </table>	Name	Function	Printing Menu	The printing menu is a general feature available from all matNMR windows that is used for printing. Using a separate window the MATLAB <i>print</i> command (see the MATLAB documentation for more information on print) can be executed. More information on the printing menu is given in paragraph 7.1, " Printing menu ".	Save figure to disk	This function will store the current window as a binary figure on disk as an executable m-file. This will allow the user to recall the exact figure by retyping the name of the m-file on the command line. This uses the Matlab mfig data format. See the Matlab manual for more information on the "saveas" command	Copy figure	This function will copy the current view of the main window into a separate window, but without the UI-controls. This will give something like:
Name	Function								
Printing Menu	The printing menu is a general feature available from all matNMR windows that is used for printing. Using a separate window the MATLAB <i>print</i> command (see the MATLAB documentation for more information on print) can be executed. More information on the printing menu is given in paragraph 7.1, " Printing menu ".								
Save figure to disk	This function will store the current window as a binary figure on disk as an executable m-file. This will allow the user to recall the exact figure by retyping the name of the m-file on the command line. This uses the Matlab mfig data format. See the Matlab manual for more information on the "saveas" command								
Copy figure	This function will copy the current view of the main window into a separate window, but without the UI-controls. This will give something like:								



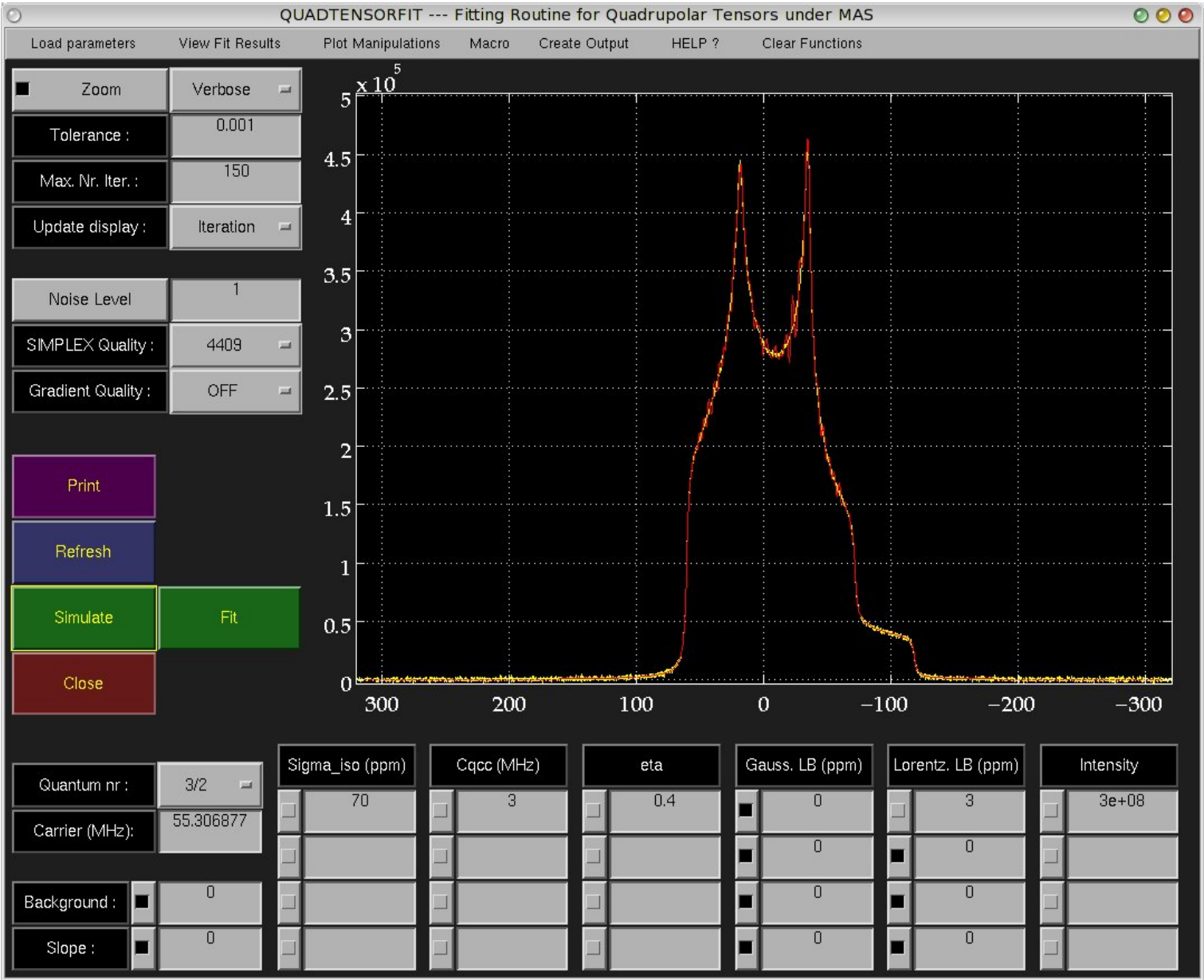
The menubar of this new window contains the standard MATLAB menubar which has some useful features for printing and saving the figure. It also offers the possibility to change axis properties and stuff (More information on this menubar can be found in the MATLAB documentation). Additional features in the menubar are the MATLAB zoom and rotate3D functions, the matNMR "Get Position" (for more information see paragraph 5.1, "General appearance") and the printing menu (for more information see paragraph 7.1, "Printing menu").

Macro	Name	Function
	Start recording macro	Starts the recording of a new macro. An input will appear to ask whether only processing, only plotting or all actions must be stored in the macro. Recording continues until the stop command is given (see below). Note that for user-commands, i.e. commands that are usually executed from the MATLAB command line, to be put into the macro the "execute user-command" function (see below) must be used! (see also paragraph 4.4, "Using macro's")
	Stop recording macro	Stop the recording of a macro. An input window will appear to ask for a name of a variable in the workspace to store the new macro in. (see also paragraph 4.4, "Using macro's")
	Execute macro	This routine executes a macro. (see also paragraph 4.4, "Using macro's")
Help?	Execute macro stepwise	This routine executes a macro stepwise. The same window as used for "Reprocess from history stepwise" (see above) will appear .
	Besides the simple "About matNMR" and "Copyright" notices this contains a direct link to this manual from inside MATLAB. The browser, defined in the \$matlabroot/toolbox/local/docopt.m file, will be opened and the matNMR manual will appear (See the MATLAB documentation for more information on the MATLAB web and docopt commands). As the manual is included in the matNMR distribution reading the manual should be faster than using the matNMR website.	
Clear functions	This performs the MATLAB "clear functions" command. All compiled functions are deleted from memory forcing MATLAB to recompile each function again upon running it. This is useful when working with your own m-files (or changing the matNMR code) to force MATLAB to use the altered function (See the MATLAB documentation for more information on the MATLAB clear command). Furthermore the mouse pointer is set to an arrow head.	

5.7 Fitting of half-integer quadrupolar tensors from MAS spectra

5.7.1 General appearance

MatNMR offers a fitting routine for quadrupolar tensors from MAS spectra which can be accessed from the "1D Processing" entry in the menubar. The following figure window will appear:



The routine can be operated using the UI-controls in the figure window. The functions of these will be explained in [paragraph 5.7.2](#) just below. The auxiliary functions in the menubar are described in [paragraph 5.7.3](#) below.

Upon starting the routine the current 1D spectrum (main window) is taken and plotted. It is assumed that the unit of the axis has been set to ppm.

To fit the isotropic shift, the Cq and and asymmetry eta, the carrier frequency and the spin quantum number are required. The routine then performs a powder averaging of an analytical

34 of 42

11/23/07 06:02 PM

expression of the frequency of a crystallite, assuming an infinite MAS spinning frequency. After adding linebroadening, either Lorentzian or Gaussian (for heterogeneous broadening), and multiplication by an intensity factor the simulation is ready. Note that the simulation is a frequency-space simulation with a fixed resolution. Up to 4 lines can be fit simultaneously.

As holds for all non-linear fits a good initial estimate of the parameters is required for an efficient fit convergence. To facilitate finding such an estimate the "Simulate" button may be used. After having defined the initial guesses the fit can be started with the "Fit" button. Should a parameter not be fitted but kept constant at its current value, then this can be done by selecting the check button (with the square block in it) belonging to the parameter.

Two algorithms may be used for fitting: SIMPLEX and a gradient algorithm. Typically the SIMPLEX algorithm is used for coarse optimization and is followed by a gradient minimization to finish off the analysis. The quality of the simulation of the sideband manifold can be determined for both algorithms separately. By choosing a higher number of powder-averaging angles the quality of the simulation will improve. By default 610 powder-averaging angles are selected for SIMPLEX and none for the gradient algorithm. Increasing the number of averaging points will improve the accuracy of the analysis.

Two algorithms may be used for fitting: SIMPLEX and a gradient algorithm. Typically the SIMPLEX algorithm is used for coarse optimization and is followed by a gradient minimization to finish off the analysis.

When the fit has finished by default the results are written into the variable QQuadFitResults which can be read out by hand by the user (Note that this variable will be deleted when matNMR is stopped! This means it must be renamed, if it has to be kept in the workspace for later use, before stopping matNMR). Alternatively, from the "View Fit Results" entry in the menubar a new name for putting the results in can be defined. To be able to access this variable the user must make the variable global by typing "global VariableName" at the MATLAB prompt. A message is written to make sure the user doesn't forget this. The format of the parameters is as follows:

- for each site in the spectrum:
 - sigma_iso
 - Cq
 - eta
 - Gaussian LB
 - Lorentzian LB
 - Intensity
- background
- slope

All these numbers are put in a single row. This variable is combined with the original data, the axis to the data, the fitted spectrum and the error value into a structure variable. The fields in this variable are "Parameters", "Fit", "AxisFit", "Data", "AxisData" and "Error". Note that the simulated spectrum has a different axis than the experimental spectrum!

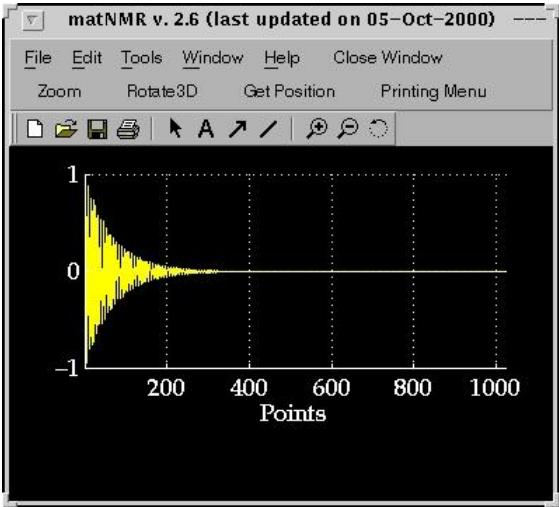
5.7.2 Buttons

Name	Function
Zoom	Switches the zoom function on or off.
Verbose	Determines the level of output generated by the fitting algorithms. This can either be never, after every iteration or only after finishing the fit.
Tolerance	Termination tolerance for the fit.
Max. Nr. Iter.	Maximum number of iterations for each fit.
Update display	Defines whether the plot should be updated after each iteration or only after the fit has finished.
Noise level	In order for the calculated error to be a χ^2 the standard deviation of the noise must be entered here. This can be taken from the 1D spectrum. A proper value for the noise level should ideally result in an error of 1 for a perfect fit, i.e. the residual is nothing more than the standard deviation of the noise then.
SIMPLEX quality	Number of powder-averaging points for the SIMPLEX algorithm. A two-angle ZCW grid is used during the simulation. Selecting "OFF" will result in the algorithm not being used. Increase this value for a more accurate result.
Gradient quality	Number of powder-averaging points for the gradient algorithm. A two-angle ZCW grid is used during the simulation. Selecting "OFF" will result in the algorithm not being used. Increase this value for a more accurate result.
Print	Prints the current figure window to the the default printer as defined in the \$matlabroot/toolbox/local/printopt.m file. No UI-controls are printed as the parameter values are put into the plot as white text. The resulting plot can be rather ugly as sometimes the text is right on top of the fitted spectrum.
Refresh	Uploads the current 1D spectrum from the main window into the fitting routine.
Simulate	Performs a simulation of a sideband manifold based on the parameters given in the UIcontrols. This allows finding a good initial estimate of the parameters, which is vital for any non-linear fitting algorithm.
Fit	Starts the fit.
Close	Closes the figure window. No variables are deleted so they can always be accessed afterwards.

Quantum NR	A quantum number must be provided. This is not a fit parameter.
Carrier	The carrier frequency for the spectrum must be provided, although this is usually imported from the main window when loading the spectrum into the fitting routine. This is not a fit parameter.
Background	A constant baseline offset can be fitted to the spectrum.
Slope	A linearly-sloping baseline offset can be fitted to the spectrum.
Sigma_iso	The isotropic chemical shift for each site. Typically, a good initial estimate of this parameter is the left (downfield) edge of the second-order quadrupolar pattern.
Cq	The quadrupolar coupling constant for each site.
eta	The asymmetry parameter for each site.
Gaussian LB	Degree of Gausiaan linebroadening (typically indicative of heterogeneous broadening).
Lorentzian LB	Degree of Lorentzian linebroadening.
Intensity	The intensity is merely a scaling factor.

5.7.3 Menubar

Name	Function	
Load parameters	Loads a variable, with the same format as the QQuadFitResults, from the workspace and puts its values into the UI-controls. Please see paragraph 5.7.1, " General Appearance " for more information on the format.	
View fit results	Takes a variable from the workspace, with the same format as the QQuadFitResults, and outputs the fitting results to the MATLAB command window. Furthermore, the parameters are written into the UIcontrols in the window and the data and fit are shown graphically. Please see paragraph 5.7.1, " General Appearance " for more information on the format.	
Plot manipulations	This menu allows changing of various axis properties. More explanation concerning these can be found in the MATLAB documentation. Note the difference between changing plot properties using this menu and by changing the properties using the options menu (see also paragraph 2.2, " Setting up matNMR "): changes made with these functions only affect the current plot and not the general matNMR settings.	
Create Output	Name	Function
	Printing Menu	The printing menu is a general feature available from all matNMR windows that is used for printing. Using a separate window the MATLAB <i>print</i> command (see the MATLAB documentation for more information on print) can be executed. More information on the printing menu is given in paragraph 7.1, " Printing menu ".
	Save figure to disk	This function will store the current window as a binary figure on disk as an executable m-file. This will allow the user to recall the exact figure by retyping the name of the m-file on the command line. This uses the Matlab mfig data format. See the Matlab manual for more information on the "saveas" command
	Copy figure	This function will copy the current view of the main window into a separate window, but without the UI-controls. This will give something like:



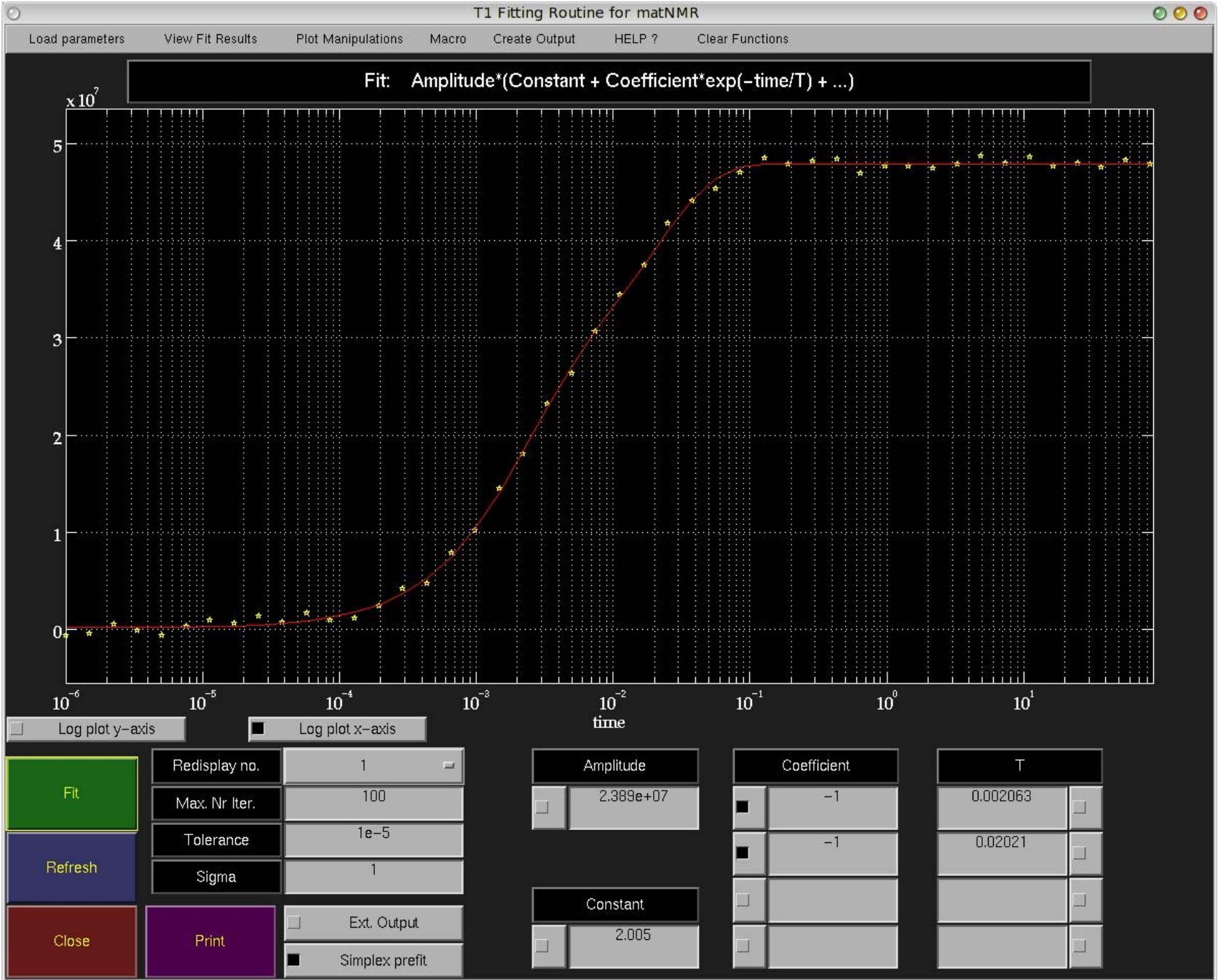
The menubar of this new window contains the standard MATLAB menubar which has some useful features for printing and saving the figure. It also offers the possibility to change axis properties and stuff (More information on this menubar can be found in the MATLAB documentation). Additional features in the menubar are the MATLAB zoom and rotate3D functions, the matNMR "Get Position" (for more information see paragraph 5.1, "General appearance") and the printing menu (for more information see paragraph 7.1, "Printing menu").

Macro	Name	Function
	Start recording macro	Starts the recording of a new macro. An input will appear to ask whether only processing, only plotting or all actions must be stored in the macro. Recording continues until the stop command is given (see below). Note that for user-commands, i.e. commands that are usually executed from the MATLAB command line, to be put into the macro the "execute user-command" function (see below) must be used! (see also paragraph 4.4, "Using macro's")
	Stop recording macro	Stop the recording of a macro. An input window will appear to ask for a name of a variable in the workspace to store the new macro in. (see also paragraph 4.4, "Using macro's")
	Execute macro	This routine executes a macro. (see also paragraph 4.4, "Using macro's")
Help?	Execute macro stepwise	This routine executes a macro stepwise. The same window as used for "Reprocess from history stepwise" (see above) will appear .
	Besides the simple "About matNMR" and "Copyright" notices this contains a direct link to this manual from inside MATLAB. The browser, defined in the \$matlabroot/toolbox/local/docopt.m file, will be opened and the matNMR manual will appear (See the MATLAB documentation for more information on the MATLAB web and docopt commands). As the manual is included in the matNMR distribution reading the manual should be faster than using the matNMR website.	
Clear functions	This performs the MATLAB "clear functions" command. All compiled functions are deleted from memory forcing MATLAB to recompile each function again upon running it. This is useful when working with your own m-files (or changing the matNMR code) to force MATLAB to use the altered function (See the MATLAB documentation for more information on the MATLAB clear command). Furthermore the mouse pointer is set to an arrow head.	

5.8 Relaxation curve fitting

5.8.1 *General appearance*

MatNMR offers a basic multi-exponential fitting routine from the "1D Processing" entry in the menubar. The following figure window will appear:



described in [paragraph 5.8.3](#) below.

Upon starting the routine the current 1D spectrum (main window) is taken and plotted. For a more convenient view on exponentials the vertical axis is plotted on a logarithmic scale. This can easily be changed using the "scaling types" item in the "plot manipulations" menu on the menubar.

To fit a (multi-)exponential curve one first has to make an initial guess for all exponential functions that need to be fitted to it. This can be done by typing the coefficient and the time constant (T1) for each fraction by hand into the corresponding UI-controls in the window. Also the overall amplitude and a constant may be defined. The routine will include a fraction into the fit if a coefficient value is specified. Note that all parameters are given in the unit of the axis vector belonging to the current 1D spectrum. This means that the axis vector has to be defined before starting the peak-fitting routine! A total of 4 exponentials can be fitted simultaneously.

After having defined the initial guesses the fit can be started with the "Fit" button. Should a parameter not be fitted but kept constant at its current value, then this can be done by selecting the check button (with the square block in it) belonging to the parameter.

When the fit has finished by default the results are written into the variable QT1FitResults which can be read out by hand by the user (Note that this variable will be deleted when matNMR is stopped! This means it must be renamed, if it has to be kept in the workspace for later use, before stopping matNMR). Alternatively, from the "View Fit Results" entry in the menubar a new name for putting the results in can be defined. To be able to access this variable the user must make the variable global by typing "global VariableName" at the MATLAB prompt. A message is written to make sure the user doesn't forget this. The format of the parameters is as follows:

- nr of iterations
- chi2 for this column
- fit parameters :
 - for each exponential:
 - Coefficient
 - Time constant (T1)
 - constant
 - amplitude

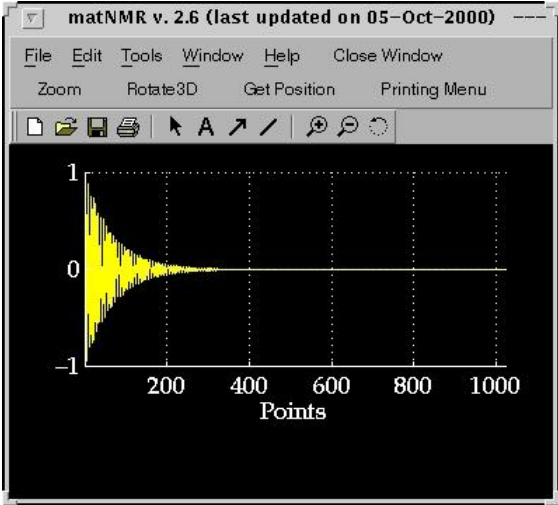
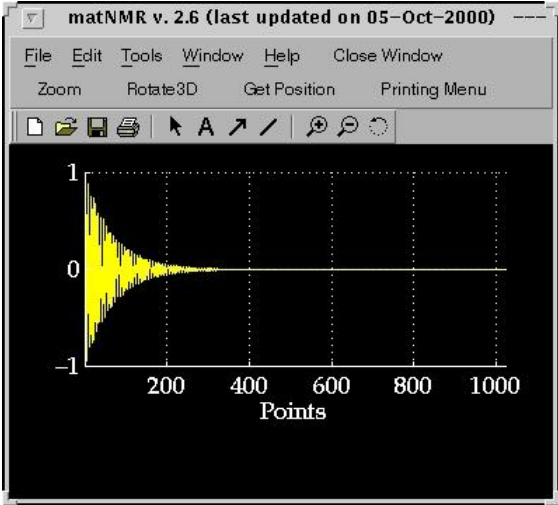
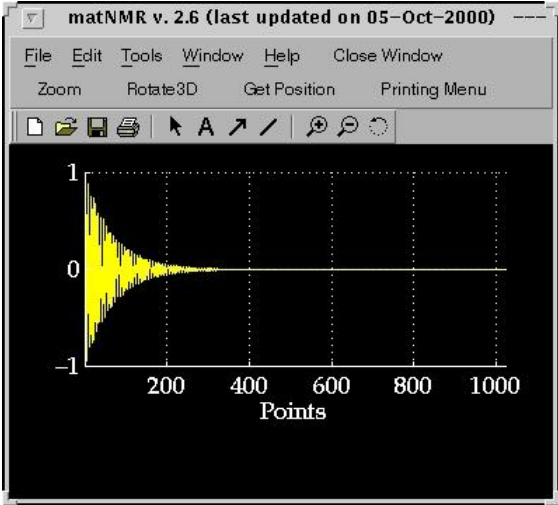
All these numbers are put in a single row. This variable is combined with the original data, the axis to the data, the fitted spectrum and the error value into a structure variable. The fields in this variable are "Parameters", "Fit", "Data", "Axis" and "Error".

Last, in case the initial parameters are very far off the "Simplex prefite" check button may be activated. This makes matNMR use a SIMPLEX algorithm first before the gradient algorithm. By default this option is active.

5.8.2 Buttons

Name	Function
Fit	Starts the fit.
Refresh	Uploads the current 1D spectrum from the main window into the T1-fitting routine.
Close	Closes the figure window. No variables are deleted so they can always be accessed afterwards.
Print	Prints the current figure window to the the default printer as defined in the \$matlabroot/toolbox/local/printopt.m file. No UI-controls are printed as the parameter values are put into the plot as white text. The resulting plot can be rather ugly as sometimes the text is right on top of the fitted spectrum.
Redisplay no.	Defines after how many iterations the plot must be redrawn. Set this to a high value to save time.
Max. Nr. Iter.	Maximum number of iterations for each fit.
Tolerance	Termination tolerance for the fit.
Sigma	Measure for the noise in the data. When properly defined the error should become a true χ^2 .
Ext. Output	Flag for whether an extended output should be given. A detailed error analysis is presented. NOTE: the basic fitting routine has not been programmed by me and I have never checked the accuracy of the error analysis!
SIMPLEX prefite	Flag for whether the fit should use a SIMPLEX algorithm before the gradient algorithm. This reduces the common problem that the gradient algorithm doesn't find a proper minimum when the parameters are too far off.
Amplitude	Overall amplitude that scales the fit.
Constant	A constant.
Coefficient	Amplitude factor for the exponential fraction.
T1	Time constant for the exponential fraction.

5.8.3 Menubar

Name	Function								
Load parameters	Loads a variable, with the same format as the QT1FitResults, from the workspace and puts its values into the UI-controls. Please see paragraph 5.8.1, "General Appearance" for more information on the format.								
View fit results	Takes a variable from the workspace, with the same format as the QT1FitResults, and outputs the fitting results to the MATLAB command window. Furthermore, the parameters are written into the UIcontrols in the window and the data and fit are shown graphically. Please see paragraph 5.8.1, "General Appearance" for more information on the format.								
Plot manipulations	This menu allows changing of various axis properties. More explanation concerning these can be found in the MATLAB documentation. Note the difference between changing plot properties using this menu and by changing the properties using the options menu (see also paragraph 2.2, "Setting up matNMR"): changes made with these functions only affect the current plot and not the general matNMR settings.								
Create Output	<table><tr><th>Name</th><th>Function</th></tr><tr><td>Printing Menu</td><td>The printing menu is a general feature available from all matNMR windows that is used for printing. Using a separate window the MATLAB <i>print</i> command (see the MATLAB documentation for more information on print) can be executed. More information on the printing menu is given in paragraph 7.1, "Printing menu".</td></tr><tr><td>Save figure to disk</td><td>This function will store the current window as a binary figure on disk as an executable m-file. This will allow the user to recall the exact figure by retyping the name of the m-file on the command line. This uses the Matlab mfig data format. See the Matlab manual for more information on the "saveas" command</td></tr><tr><td>Copy figure</td><td><p>This function will copy the current view of the main window into a separate window, but without the UI-controls. This will give something like:</p><div data-bbox="987 786 1543 1292"></div><p>The menubar of this new window contains the standard MATLAB menubar which has some useful features for printing and saving the figure. It also offers the possibility to change axis properties and stuff (More information on this menubar can be found in the MATLAB documentation). Additional features in the menubar are the MATLAB zoom and rotate3D functions, the matNMR "Get Position" (for more information see paragraph 5.1, "General appearance") and the printing menu (for more information see paragraph 7.1, "Printing menu").</p></td></tr></table>	Name	Function	Printing Menu	The printing menu is a general feature available from all matNMR windows that is used for printing. Using a separate window the MATLAB <i>print</i> command (see the MATLAB documentation for more information on print) can be executed. More information on the printing menu is given in paragraph 7.1, "Printing menu".	Save figure to disk	This function will store the current window as a binary figure on disk as an executable m-file. This will allow the user to recall the exact figure by retyping the name of the m-file on the command line. This uses the Matlab mfig data format. See the Matlab manual for more information on the "saveas" command	Copy figure	<p>This function will copy the current view of the main window into a separate window, but without the UI-controls. This will give something like:</p> <div data-bbox="987 786 1543 1292"></div> <p>The menubar of this new window contains the standard MATLAB menubar which has some useful features for printing and saving the figure. It also offers the possibility to change axis properties and stuff (More information on this menubar can be found in the MATLAB documentation). Additional features in the menubar are the MATLAB zoom and rotate3D functions, the matNMR "Get Position" (for more information see paragraph 5.1, "General appearance") and the printing menu (for more information see paragraph 7.1, "Printing menu").</p>
Name	Function								
Printing Menu	The printing menu is a general feature available from all matNMR windows that is used for printing. Using a separate window the MATLAB <i>print</i> command (see the MATLAB documentation for more information on print) can be executed. More information on the printing menu is given in paragraph 7.1, "Printing menu".								
Save figure to disk	This function will store the current window as a binary figure on disk as an executable m-file. This will allow the user to recall the exact figure by retyping the name of the m-file on the command line. This uses the Matlab mfig data format. See the Matlab manual for more information on the "saveas" command								
Copy figure	<p>This function will copy the current view of the main window into a separate window, but without the UI-controls. This will give something like:</p> <div data-bbox="987 786 1543 1292"></div> <p>The menubar of this new window contains the standard MATLAB menubar which has some useful features for printing and saving the figure. It also offers the possibility to change axis properties and stuff (More information on this menubar can be found in the MATLAB documentation). Additional features in the menubar are the MATLAB zoom and rotate3D functions, the matNMR "Get Position" (for more information see paragraph 5.1, "General appearance") and the printing menu (for more information see paragraph 7.1, "Printing menu").</p>								

	Name	Function
Macro	Start recording macro	Starts the recording of a new macro. An input will appear to ask whether only processing, only plotting or all actions must be stored in the macro. Recording continues until the stop command is given. (see below). Note that for user-commands, i.e. commands that are usually executed from the MATLAB command line, to be put into the macro the "execute user-command" function (see below) must be used! (see also paragraph 4.4, " Using macro's ")
	Stop recording macro	Stop the recording of a macro. An input window will appear to ask for a name of a variable in the workspace to store the new macro in. (see also paragraph 4.4, " Using macro's ")
	Execute macro	This routine executes a macro. (see also paragraph 4.4, " Using macro's ")
	Execute macro stepwise	This routine executes a macro stepwise. The same window as used for "Reprocess from history stepwise" (see above) will appear .
Help?	Besides the simple "About matNMR" and "Copyright" notices this contains a direct link to this manual from inside MATLAB. The browser, defined in the \$matlabroot/toolbox/local/docopt.m file, will be opened and the matNMR manual will appear (See the MATLAB documentation for more information on the MATLAB web and docopt commands). As the manual is included in the matNMR distribution reading the manual should be faster than using the matNMR website.	
Clear functions	This performs the MATLAB "clear functions" command. All compiled functions are deleted from memory forcing MATLAB to recompile each function again upon running it. This is useful when working with your own m-files (or changing the matNMR code) to force MATLAB to use the altered function (See the MATLAB documentation for more information on the MATLAB clear command). Furthermore the mouse pointer is set to an arrow head.	



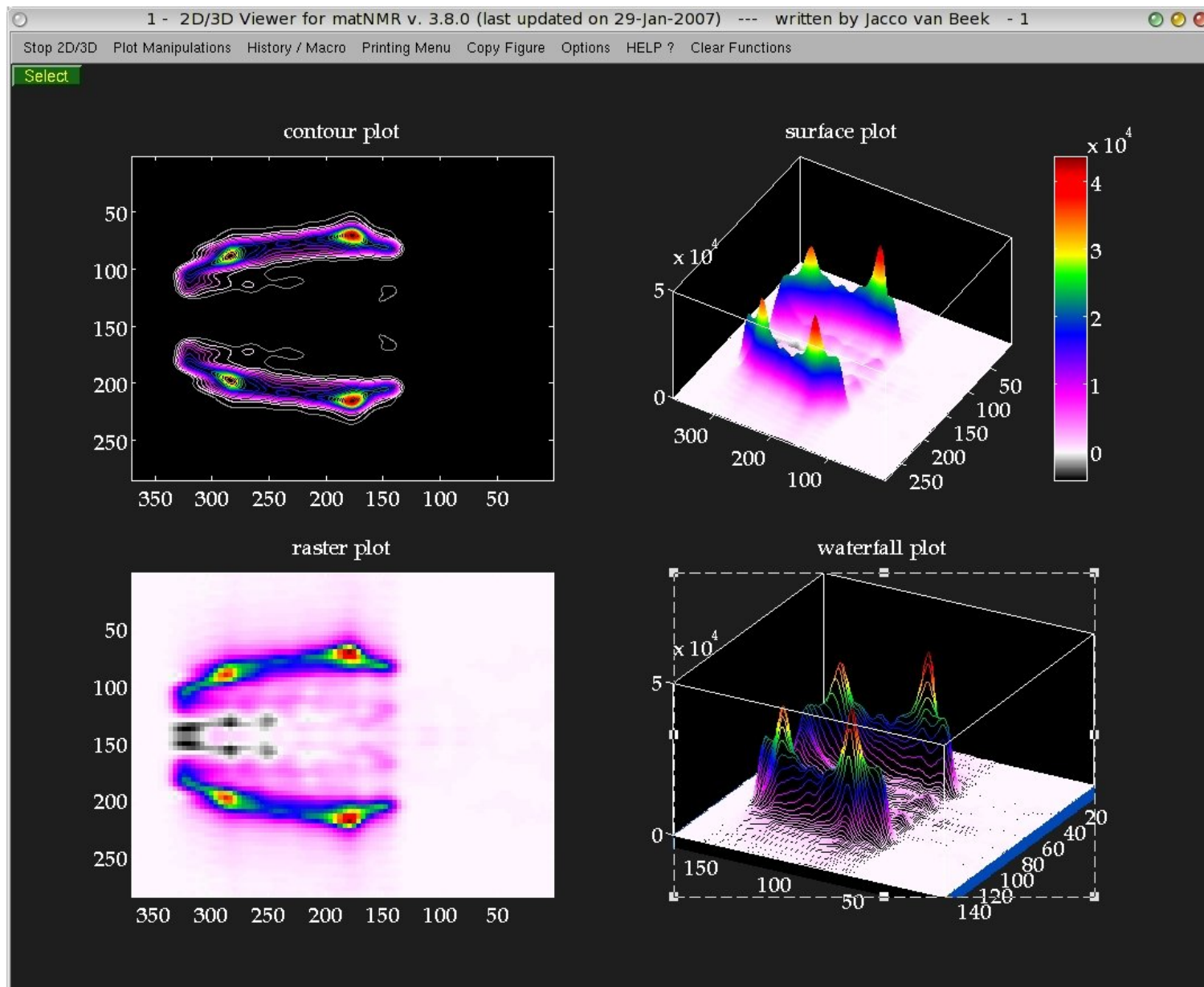
Chapter 6 - The 2D/3D Viewer

The 2D/3D Viewer is started using the nmr2d.m script, i.e. type 'nmr2d' at the Matlab prompt to start it.

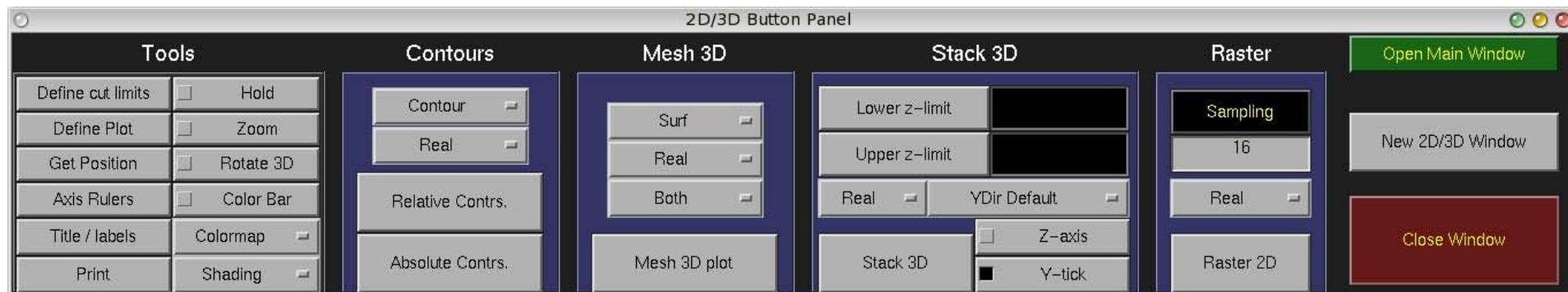
6.1 General Appearance

The general appearance of the 2D/3D Viewer window consists of two parts and is shown in the figures below:

A typical data window:



and the panel window:



The 2D/3D Viewer is a multi-window multi-plot routine: an arbitrary number of windows can be opened (simultaneously) and they can all have an arbitrary number of subplots (only limited by the number of pre-defined subplot configurations matNMR allows). All data windows are controlled from the panel window. Any function executed from here will apply to the current axis in the current 2D/3D Viewer window. The current axis is always marked by a box around it. The current figure though is not always so clearly recognizable. In general the current figure will be pulled in front before executing a plotting command, making it clear what the current figure window is. In general it is **NOT** sufficient to pull the window in front if one wants to make a figure the current figure! The preferred way of selecting a window therefore is to click on the "Select" button in the top left corner of the window, or alternatively inside the window (Not on the menubar!).

Although this routine is multi-window multi-plot only one set of variables is kept in memory (for more information on memory usage see also paragraph 3.7, "[Memory usage](#)"). This means that only the matrix (and axis vectors etc.) belonging to the **last plotted spectrum** is stored. This is important to realize because it affects certain routines that use the matrix for extracting spectral intensities like Integrate, peak picking, get position, etc. These only function properly for the last plotted spectrum. And although they may not produce any errors, the result may still be wrong! Some plot parameters are stored into the figure window and they contain some information on each specific plot in the figure:

- name
- axis vectors: extracting axis coordinates will work properly even when the plot does not belong to the last plotted spectrum!
- processing history

Furthermore certain properties of the current figure are saved:

- number of subplots
- axis handles
- colorbar flags for each axis
- colorbar handles
- zoom flag
- rotate3D flag

All these parameters are stored into the userdata property of the figure (for more information on properties of graphics objects see the MATLAB documentation). Extracting this property (e.g. QTEMP= get(gcf, 'userdata')) will result in a structure variable. Users can add fields to it (in principle the userdata property is free to be manipulated in any way and can therefore be useful for various things) but shouldn't removed the fields that matNMR has created so far as this will produce errors. .

The functions panel window will be explained in paragraph 6.2, just below and the menubar in the data windows in paragraph 6.3, "[The menubar](#)".

6.2 The panel window

The panel window has been subdivided into the four major plot types currently supported by matNMR, contours, mesh, Stack 3D and raster plot and a tools menu. They will be explained separately below. Note that although the functions explained below are separated into discrete categories they can be used as general functions.

Note that all the items in the "Tools" menu can be started also from the Plotmanipulations menu in the menubar.

6.2.1 Contours

Button	Function						
Contour	Popup button for selecting the type of contour plot. Currently supported are the three MATLAB contour routines <i>contour</i> (normal contours), <i>contourf</i> (filled contours) and <i>contour3</i> (contours plotted at their corresponding z-level). See the MATLAB documentation for more information on these routines.						
Real	Popup button for selecting whether to plot the real, the imaginary or the absolute value of a matrix.						
Relative Contours	<div><p>This function allows plotting relative contours, i.e. the contour levels are specified as a percentage of the maximum intensity of a matrix. The following input window will appear:</p><div><div>Display Relative Contours of :</div><div><div>Name Spectrum :</div><div>matNMR</div></div><div><div>Lower countour limit in % of the maximum :</div><div>1</div></div><div><div>Upper countour limit in % of the maximum :</div><div>100</div></div><div><div>Multiplier (1 = linear)</div><div>1</div></div><div><div>Number of contours :</div><div>10</div></div><div><div>Type of contours</div><div>Only Positive</div></div><div><div>Vector for TD2-axis (empty line gives current axis) :</div><div></div></div><div><div>Vector for TD1-axis (empty line gives current axis) :</div><div></div></div><div><div>Linespec (optional) :</div><div></div></div><div><div>OK</div><div>CANCEL</div></div></div></div> <p>The following fields must be specified:</p> <table><tr><th>Field</th><th>Function</th></tr><tr><td>Name Spectrum</td><td>This must be either a name of a matrix variable in the workspace or a function generating a matrix as output. In case the variable is a matNMR structure (see also paragraph 3.5, "matNMR format for spectra") then the routine will extract the corresponding axis vectors (and history etc) that were saved into it. If axis vectors are specified in this input window (see below) for Td2 and/or TD1 then these are used and not the axis vectors as saved in the structure.</td></tr><tr><td>Lower contour limit</td><td>Defines the lower contour limit as a percentage of the maximum intensity of the matrix.</td></tr></table>	Field	Function	Name Spectrum	This must be either a name of a matrix variable in the workspace or a function generating a matrix as output. In case the variable is a matNMR structure (see also paragraph 3.5, " matNMR format for spectra ") then the routine will extract the corresponding axis vectors (and history etc) that were saved into it. If axis vectors are specified in this input window (see below) for Td2 and/or TD1 then these are used and not the axis vectors as saved in the structure.	Lower contour limit	Defines the lower contour limit as a percentage of the maximum intensity of the matrix.
Field	Function						
Name Spectrum	This must be either a name of a matrix variable in the workspace or a function generating a matrix as output. In case the variable is a matNMR structure (see also paragraph 3.5, " matNMR format for spectra ") then the routine will extract the corresponding axis vectors (and history etc) that were saved into it. If axis vectors are specified in this input window (see below) for Td2 and/or TD1 then these are used and not the axis vectors as saved in the structure.						
Lower contour limit	Defines the lower contour limit as a percentage of the maximum intensity of the matrix.						

Upper contour limit	Defines the upper contour limit as a percentage of the maximum intensity of the matrix.
Multiplier	A number that specifies the non-linearity of the increment between contour levels. If (Multiplier = 1) then the increment between contour levels is the same for all contour levels. If (0 < Multiplier < 1) then more contours are drawn towards the high end of the range as defined by the lower and upper limits. If (Multiplier > 1) then more contour levels are drawn towards the lower end of the range.
Number of contours	The total number of contour levels in the plot. When both positive and negative contours are asked for then a vector of two numbers may be given to denote different numbers of contours for positive and negative intensities. If only one number is specified then both positive and negative intensities have the same number of contours.
Type of contours	This determines how the contour levels will be calculated. This can either be a) positive contours (i.e. relative to the positive maximum), (b) negative contours (i.e. relative to the negative maximum), (c) both positive and negative contours (relative to the respective positive and negative maxima), (d) both positive and negative contours but relative to the positive maximum or (e) both positive and negative contours but relative to the negative maximum of the spectrum. Note that for the last three options the number of contours is doubled (if a contour level is set at 0, the number of contours is doubled minus 1). A special colormap will be generated when printing both positive and negative contours.
Vector for TD2-axis	A variable in the workspace corresponding to the axis vector for this dimension. Leaving this empty will give an axis in points unless the "Name Spectrum" is a matNMR structure (see "Name Spectrum"). Specifying an axis vector will overrule any axis vector saved in the variable structure.
Vector for TD1-axis	A variable in the workspace corresponding to the axis vector for this dimension. Leaving this empty will give an axis in points unless the "Name Spectrum" is a matNMR structure (see "Name Spectrum"). Specifying an axis vector will overrule any axis vector saved in the variable structure.
Linespec	Leaving this empty will plot the contours as a MATLAB <i>patch</i> (for more information on this see the MATLAB documentation), meaning it will use a colormap. Specifying the linespec causes MATLAB to plot the contours as line plots. See the MATLAB documentation for more information on colormaps and linespec properties.
Plotting Macro	Allows for a plotting macro to be executed immediately after the plot is made, in order to reduce rendering time. See also paragraph 4.4, "Using macro's".

This function allows plotting absolute contours, i.e. the contour levels are specified directly by the user. The following input window will appear:

Display Absolute Contour of :

Name Spectrum :

matNMR

Vector with contour levels :

Vector for TD2-axis (leave empty for axis in points)

Vector for TD1-axis (leave empty for axis in points)

Linespec (OPTIONAL I) :

OK

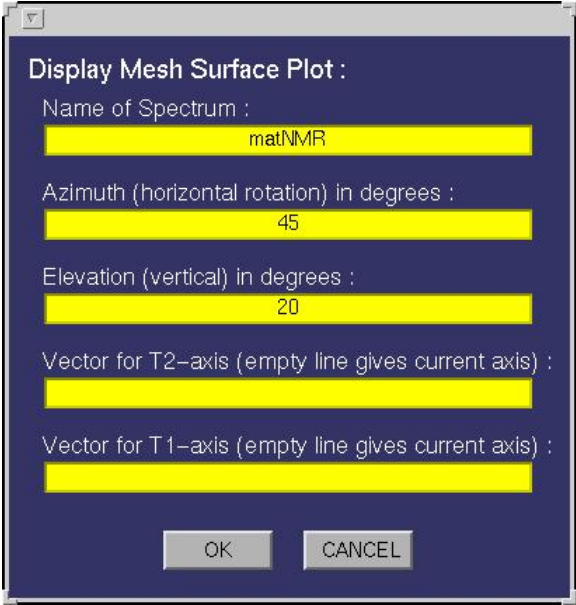
CANCEL

The following fields must be specified:

Field	Function
-------	----------

Name Spectrum	This must be either a name of a matrix variable in the workspace or a function generating a matrix as output. In case the variable is a matNMR structure (see also paragraph 3.5, " matNMR format for spectra ") then the routine will extract the corresponding axis vectors (and history etc) that were saved into it. If axis vectors are specified in this input window (see below) for Td2 and/or TD1 then these are used and not the axis vectors as saved in the structure.
Vector with contour levels	A variable in the workspace containing the contour level intensities.
Vector for TD2-axis	A variable in the workspace corresponding to the axis vector for this dimension. Leaving this empty will give an axis in points unless the "Name Spectrum" is a matNMR structure (see "Name Spectrum"). Specifying an axis vector will overrule any axis vector saved in the variable structure.
Vector for TD1-axis	A variable in the workspace corresponding to the axis vector for this dimension. Leaving this empty will give an axis in points unless the "Name Spectrum" is a matNMR structure (see "Name Spectrum"). Specifying an axis vector will overrule any axis vector saved in the variable structure.
Linespec	Leaving this empty will plot the contours as a MATLAB <i>patch</i> (for more information on this see the MATLAB documentation), meaning it will use a colormap. Specifying the linespec causes MATLAB to plot the contours as line plots. See the MATLAB documentation for more information on colormaps and linespec properties.
Plotting Macro	Allows for a plotting macro to be executed immediately after the plot is made, in order to reduce rendering time. See also paragraph 4.4, " Using macro's ".

6.2.2 Mesh 3D

Button	Function		
Mesh	Popup button for selecting the type of surface plot. Currently supported are the eight MATLAB routines <i>mesh</i> (normal mesh), <i>meshc</i> (mesh + contours underneath), <i>meshz</i> (mesh with curtain), <i>surf</i> (normal surface), <i>surfc</i> (surface with a contour plot underneath), <i>surfl</i> (surface with lighting effects), <i>pcolor</i> (pseudo-color plot, i.e. a top-projection of the surface) and <i>waterfall</i> (a waterfall plot). See the MATLAB documentation for more information on these routines. A word of caution must be made for using the waterfall plot as it has proven be not so stable in the past! Sometimes it used to take ages to finish.		
Real	Popup button for selecting whether to plot the real, the imaginary or the absolute value of a matrix.		
Both	Defines whether to draw mesh grid lines in both matrix dimensions or just along the rows (TD2) or columns (TD1). This property only affects mesh plots, not surface plots.		
Mesh 3D Plot	<div>This function allows plotting mesh and surface plots. The following input window will appear:<div></div><div>The following fields must be specified:</div><table><tr><th>Field</th><th>Function</th></tr></table></div>	Field	Function
Field	Function		

Name Spectrum	This must be either a name of a matrix variable in the workspace or a function generating a matrix as output. In case the variable is a matNMR structure (see also paragraph 3.5, " matNMR format for spectra ") then the routine will extract the corresponding axis vectors (and history etc) that were saved into it. If axis vectors are specified in this input window (see below) for Td2 and/or TD1 then these are used and not the axis vectors as saved in the structure.
Azimuthal	Defines the azimuth angle or better the horizontal rotation angle for the resulting view (see MATLAB documentation for more information on <i>view</i>). The view can always be changed by using the Rotate3D function in the Tools section (described below).
Elevation	Defines the elevation angle or better the vertical rotation angle for the resulting view (see MATLAB documentation for more information on <i>view</i>). The view can always be changed by using the Rotate3D function in the Tools section (described below).
Vector for TD2-axis	A variable in the workspace corresponding to the axis vector for this dimension. Leaving this empty will give an axis in points unless the "Name Spectrum" is a matNMR structure (see "Name Spectrum"). Specifying an axis vector will overrule any axis vector saved in the variable structure.
Vector for TD1-axis	A variable in the workspace corresponding to the axis vector for this dimension. Leaving this empty will give an axis in points unless the "Name Spectrum" is a matNMR structure (see "Name Spectrum"). Specifying an axis vector will overrule any axis vector saved in the variable structure.
Plotting Macro	Allows for a plotting macro to be executed immediately after the plot is made, in order to reduce rendering time. See also paragraph 4.4, " Using macro's ".

6.2.3 Stack 3D

3D Stack plots are plots in which, along one dimension of the matrix, the lines are stacked such that a spatial view is obtained. This can be very useful for watching decays etc.

Button	Function
Lower z-limit	Defines the lower zlimit for the plot. If not defined before plotting a new matrix then matNMR will set it to the default axis settings after plotting the matrix.
Upper z-limit	Defines the upper zlimit for the plot. If not defined before plotting a new matrix then matNMR will set it to the default axis settings after plotting the matrix.
Real	Popup button for selecting whether to plot the real, the imaginary or the absolute value of a matrix.
Y-dir default	Defines the default axis direction for the plot. Usually this is set to right-to-left for NMR spectra but this is not necessarily what is needed for a plot.
Stack 3D	This function allows plotting 3D stack plots. The following input window will appear:

Display 3D Stack Plot :

Name of Spectrum :

matNMR

Dimension along X-axis (1=TD2, 2=TD1) :

1

Azimuth (horizontal rotation) in degrees :

15

Elevation (vertical) in degrees :

15

Vector for T2-axis (empty line gives current axis) :

Vector for T1-axis (empty line gives current axis) :

OK

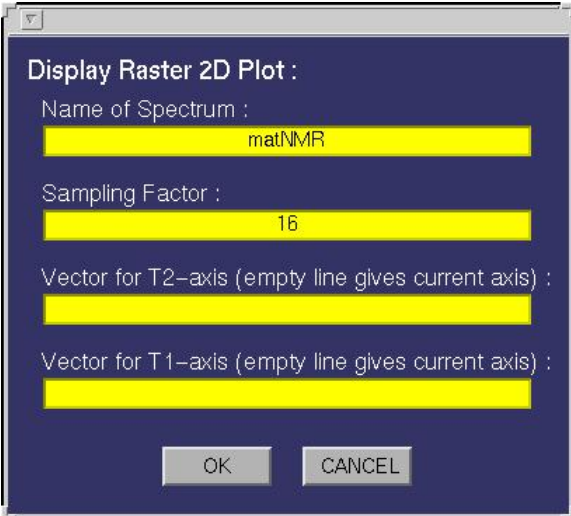
CANCEL

The following fields must be specified:

Field	Function
Name Spectrum	This must be either a name of a matrix variable in the workspace or a function generating a matrix as output. In case the variable is a matNMR structure (see also paragraph 3.5, "matNMR format for spectra") then the routine will extract the corresponding axis vectors (and history etc) that were saved into it. If axis vectors are specified in this input window (see below) for Td2 and/or TD1 then these are used and not the axis vectors as saved in the structure.
Dimension along x-axis	Determines which matrix dimension to put along the x-axis, i.e. in which direction to stack the rows/columns.
Azimuthal	Defines the azimuth angle or better the horizontal rotation angle for the resulting view (see MATLAB documentation for more information on view). The view can always be changed by using the Rotate3D function in the Tools section (described below).
Elevation	Defines the elevation angle or better the vertical rotation angle for the resulting view (see MATLAB documentation for more information on view). The view can always be changed by using the Rotate3D function in the Tools section (described below).
Vector for TD2-axis	A variable in the workspace corresponding to the axis vector for this dimension. Leaving this empty will give an axis in points unless the "Name Spectrum" is a matNMR structure (see "Name Spectrum"). Specifying an axis vector will overrule any axis vector saved in the variable structure.
Vector for TD1-axis	A variable in the workspace corresponding to the axis vector for this dimension. Leaving this empty will give an axis in points unless the "Name Spectrum" is a matNMR structure (see "Name Spectrum"). Specifying an axis vector will overrule any axis vector saved in the variable structure.
Plotting Macro	Allows for a plotting macro to be executed immediately after the plot is made, in order to reduce rendering time. See also paragraph 4.4, "Using macro's".
z-axis	Allows switching the z-axis on or off from displaying.
y-tick	Allows switching the tickmarks along the y-axis on or off from displaying.

6.2.4 Raster 2D

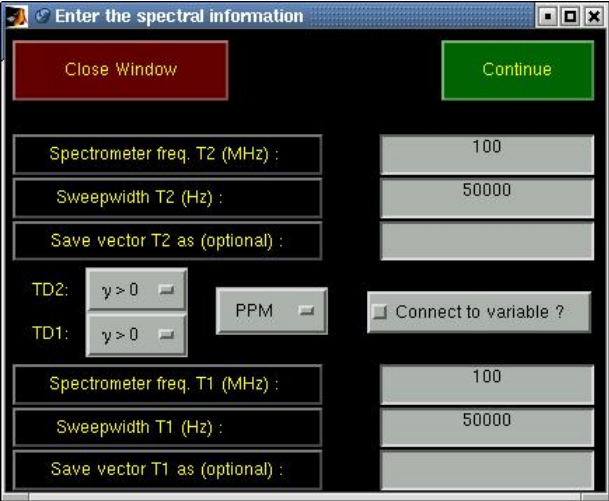
The raster plot can be used for making quick low-resolution pictures of big spectra by undersampling the matrix. The matrix is divided in blocks as big as the sampling size and the intensity over the block is integrated.

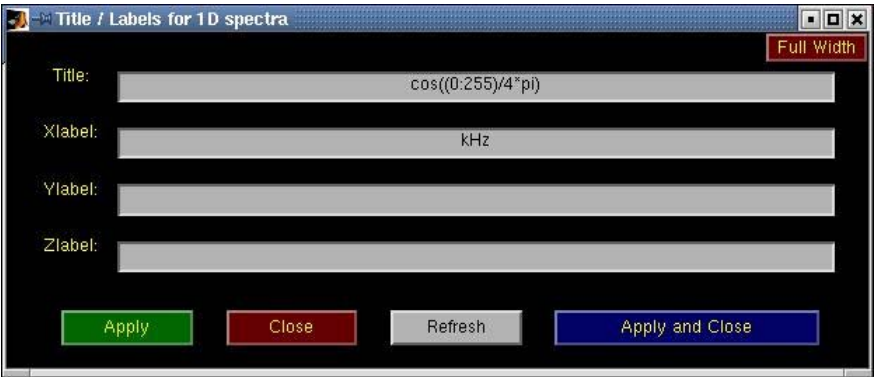
Button	Function																
Sampling	Defines the sampling size (in points) for the current spectrum. Note that setting a value into this edit button immediately takes effect on the current matrix.																
Real	Popup button for selecting whether to plot the real, the imaginary or the absolute value of a matrix.																
Raster 2D	This function allows plotting 2D raster plots. The following input window will appear: <div></div>																
	The following fields must be specified:																
	<table><tr><th>Field</th><th>Function</th></tr><tr><td>Name Spectrum</td><td>This must be either a name of a matrix variable in the workspace or a function generating a matrix as output. In case the variable is a matNMR structure (see also paragraph 3.5, "matNMR format for spectra") then the routine will extract the corresponding axis vectors (and history etc) that were saved into it. If axis vectors are specified in this input window (see below) for Td2 and/or TD1 then these are used and not the axis vectors as saved in the structure.</td></tr><tr><td>Dimension along x-axis</td><td>Determines which matrix dimension to put along the x-axis, i.e. in which direction to stack the rows/columns.</td></tr><tr><td>Azimuthal</td><td>Defines the azimuth angle or better the horizontal rotation angle for the resulting view (see MATLAB documentation for more information on view). The view can always be changed by using the Rotate3D function in the Tools section (described below).</td></tr><tr><td>Elevation</td><td>Defines the elevation angle or better the vertical rotation angle for the resulting view (see MATLAB documentation for more information on view). The view can always be changed by using the Rotate3D function in the Tools section (described below).</td></tr><tr><td>Vector for TD2-axis</td><td>A variable in the workspace corresponding to the axis vector for this dimension. Leaving this empty will give an axis in points unless the "Name Spectrum" is a matNMR structure (see "Name Spectrum"). Specifying an axis vector will overrule any axis vector saved in the variable structure.</td></tr><tr><td>Vector for TD1-axis</td><td>A variable in the workspace corresponding to the axis vector for this dimension. Leaving this empty will give an axis in points unless the "Name Spectrum" is a matNMR structure (see "Name Spectrum"). Specifying an axis vector will overrule any axis vector saved in the variable structure.</td></tr><tr><td>Plotting Macro</td><td>Allows for a plotting macro to be executed immediately after the plot is made, in order to reduce rendering time. See also paragraph 4.4, "Using macro's".</td></tr></table>	Field	Function	Name Spectrum	This must be either a name of a matrix variable in the workspace or a function generating a matrix as output. In case the variable is a matNMR structure (see also paragraph 3.5, "matNMR format for spectra") then the routine will extract the corresponding axis vectors (and history etc) that were saved into it. If axis vectors are specified in this input window (see below) for Td2 and/or TD1 then these are used and not the axis vectors as saved in the structure.	Dimension along x-axis	Determines which matrix dimension to put along the x-axis, i.e. in which direction to stack the rows/columns.	Azimuthal	Defines the azimuth angle or better the horizontal rotation angle for the resulting view (see MATLAB documentation for more information on view). The view can always be changed by using the Rotate3D function in the Tools section (described below).	Elevation	Defines the elevation angle or better the vertical rotation angle for the resulting view (see MATLAB documentation for more information on view). The view can always be changed by using the Rotate3D function in the Tools section (described below).	Vector for TD2-axis	A variable in the workspace corresponding to the axis vector for this dimension. Leaving this empty will give an axis in points unless the "Name Spectrum" is a matNMR structure (see "Name Spectrum"). Specifying an axis vector will overrule any axis vector saved in the variable structure.	Vector for TD1-axis	A variable in the workspace corresponding to the axis vector for this dimension. Leaving this empty will give an axis in points unless the "Name Spectrum" is a matNMR structure (see "Name Spectrum"). Specifying an axis vector will overrule any axis vector saved in the variable structure.	Plotting Macro	Allows for a plotting macro to be executed immediately after the plot is made, in order to reduce rendering time. See also paragraph 4.4, "Using macro's".
	Field	Function															
	Name Spectrum	This must be either a name of a matrix variable in the workspace or a function generating a matrix as output. In case the variable is a matNMR structure (see also paragraph 3.5, "matNMR format for spectra") then the routine will extract the corresponding axis vectors (and history etc) that were saved into it. If axis vectors are specified in this input window (see below) for Td2 and/or TD1 then these are used and not the axis vectors as saved in the structure.															
Dimension along x-axis	Determines which matrix dimension to put along the x-axis, i.e. in which direction to stack the rows/columns.																
Azimuthal	Defines the azimuth angle or better the horizontal rotation angle for the resulting view (see MATLAB documentation for more information on view). The view can always be changed by using the Rotate3D function in the Tools section (described below).																
Elevation	Defines the elevation angle or better the vertical rotation angle for the resulting view (see MATLAB documentation for more information on view). The view can always be changed by using the Rotate3D function in the Tools section (described below).																
Vector for TD2-axis	A variable in the workspace corresponding to the axis vector for this dimension. Leaving this empty will give an axis in points unless the "Name Spectrum" is a matNMR structure (see "Name Spectrum"). Specifying an axis vector will overrule any axis vector saved in the variable structure.																
Vector for TD1-axis	A variable in the workspace corresponding to the axis vector for this dimension. Leaving this empty will give an axis in points unless the "Name Spectrum" is a matNMR structure (see "Name Spectrum"). Specifying an axis vector will overrule any axis vector saved in the variable structure.																
Plotting Macro	Allows for a plotting macro to be executed immediately after the plot is made, in order to reduce rendering time. See also paragraph 4.4, "Using macro's".																
z-axis	Allows switching the z-axis on or off from displaying.																
y-tick	Allows switching the tickmarks along the y-axis on or off from displaying.																

6.2.5 Tools

The Tools section contains a few functions that are often needed while working with the 2D/3D Viewer. All of them can also be found in the "Plot manipulations" entry in the menubar (for more information on the plot manipulations see paragraph 6.3.2, "Plot manipulations").

Button	Function
--------	----------

Cut spectrum	This function allows to make a slice-plot of a 2D spectrum by defining strips of frequencies in both dimenions. A new variable "QmatNMR.Spec2D3DCutSpectrum" will be created which contains these blocks of the spectrum. This variable may then be called by any of the plotting routines and the blocks will be shown separated by a dotted line. After pushing this button use the mouse to define the strips. Afterwards an input window will be opened and the coordinates can be specified more accurately. NOTE that using the corresponding function in the menubar avoids having to define the coordinates by mouse again!
Define plot	It is often most efficient to make a low-quality plot of a spectrum, zoom into the region of interest and then replot it in high quality. This function allows to do so, by creating a new variable "QmatNMR.Spec2D3DDefinePlot" with the zoomed-in part of the spectrum and the axes. This variable may then be used as input for the subsequent plotting action.
Get Position	Like the "Get Position" routine in the main window (see paragraph 5.1, "General Appearance") this routine provides a means of scanning the matrix. The coordinates in both dimensions for the position of the crosshair and the intensity is shown on screen. Note that the intensity is only valid for the last plotted spectrum. The coordinates should always be correct though!
Axis rulers	<div><p>This allows changing the axis vectors for the current matrix. Similar to the way it is done in the main window (see paragraph 5.2.5, "Plot manipulations") the axis vectors can be set to either PPM, Hz, time, points or to a user- defined vector. The following window is opened:</p><p>First the type of axis vector (PPM, Hz, time, points, user-defined) can be chosen. The spectral width must be specified for PPM and Hz axes and for a PPM axis also the carrier frequency. For axes in Hz also the signs of the gyromagnetic ratios of the nuclei that correspond to the respective dimensions of the spectrum must be specified. The button labeled "Connect to variable?" can be checked if the new axes must be connected to the spectrum. If the spectrum wasn't yet a matNMR structure then it will be made into one in order to be able to allow the connection. Pressing the "Continue" button proceeds the routine and one is asked for the reference values. After that the reference peak can be specified by using the crosshair mouse pointer and clicking on the desired reference peak (This may seem strange but is necessary because of a MATLAB bug. Specifying the reference peak before asking its value leads to a mess with the main and input window). If a name is given for the new axis vector(s) then it/they will be saved in the workspace after calculating it.</p></div>
Title/labels	This routine provides a means of changing the title and axes labels of the current axis. A window will appear looking like:



Pressing the "Apply" button will update all labels immediately. The "Refresh" is used for updating the text labels currently written in this window to the labels in the current axis (useful when having this window opened continuously).

Print	Prints the current 2D/3D Viewer window to the default printer as defined in the \$matlabroot/toolbox/local/printopt.m file.
Hold	Switches the hold flag for the current axis. Upon activating this, every consecutive plot is superposed on the existing plots. (see the MATLAB documentation for more information on the <i>hold</i> command).
Zoom	Switches the zoom flag for the current window. Upon activating this, the rotate3D flag is switched off and the standard MATLAB <i>zoom</i> is activated (see the MATLAB documentation for more information on the <i>zoom</i> command). Zooming in can be done using the left mouse button (either by clicking or by dragging a box around the desired area) whereas the right button zooms out.
Rotate3D	Switches the rotate3D flag for the current window. Upon activating this the zoom flag is switched off and the standard MATLAB <i>rotate3d</i> command is activated (see the MATLAB documentation for more information on the <i>rotate3d</i> command). The plot can be rotated by moving the mouse while having the left mouse button pressed in the desired axis.
Colorbar	Adds or removes a colorbar from the current axis.
Colormap	Allows changing of the colormap for the current 2D/3D Viewer window. Beware that colormaps are a figure property and therefore different axes cannot have different color maps. This popup button contains a few of the standard MATLAB colormaps and some others. The QPosNeg colormaps are altered upon plotting a contour plot with both positive and negative contours, such that the central colour always denotes the zero-level. Selecting these in any other case will show the full colormap. To adjust those colormaps according to the degree of positive and negative intensities in the spectrum, select "Adjust PosNeg". All colormaps may be inverted by selecting the corresponding item from the list.
Shading	Sets the shading property (for more information see the MATLAB documentation).

6.3 The menubar

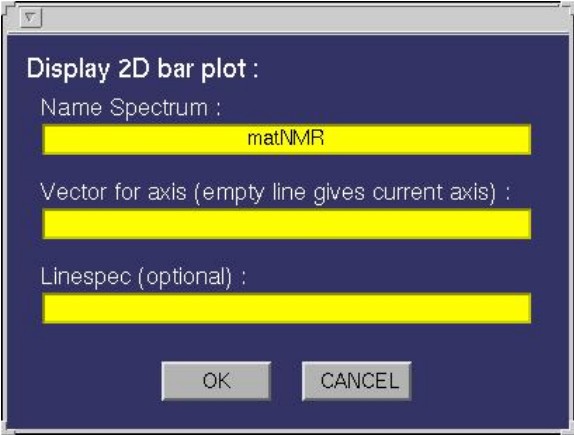
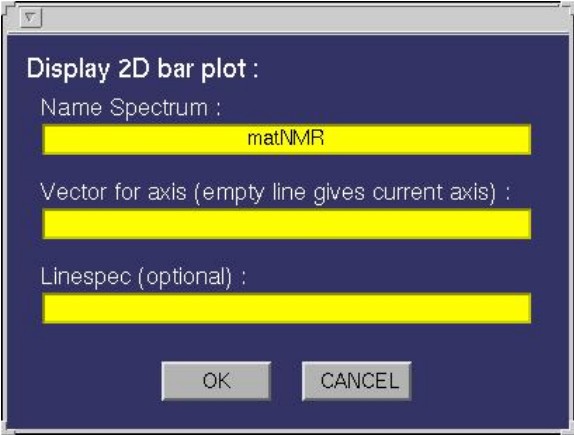
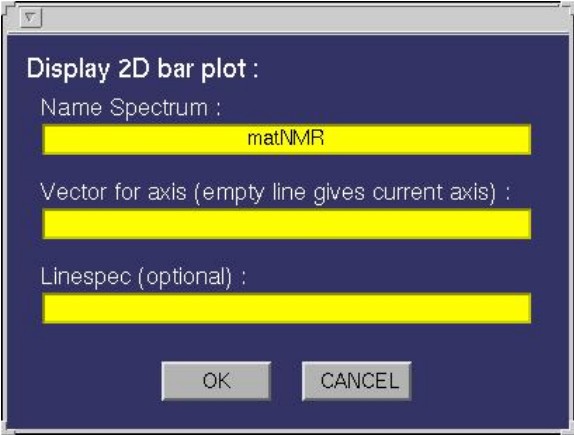
The menubar contains some useful features for the 2D/3D Viewer. All entries will be explained here separately. Note that many of the functions may also be accessed from the context menus that are defined. Context menus may be accessed by right-clicking anywhere within the figure window, but not on top of an axis! To reduce rerendering by Matlab, it is most efficient to move the 2D/3D viewer window to the far left of the screen, and to access the context menu from the far right of the 2D/3D viewer window.

6.3.1 Stop 2D/3D Viewer

This function allows close the current 2D/3D Viewer window, stop the 2D/3D Viewer, stop matNMR completely or quit MATLAB. Depending on whether the exit safety flag was set (see paragraph 2.2.1, "General options") an input window will appear to ask for confirmation. For quitting MATLAB the user is always asked for confirmation. Note that when stopping matNMR **ALL** windows belonging to matNMR are closed, including the main window. Stopping only the 2D/3D Viewer will not affect any other routines though.

6.3.2 Plot manipulations

This menu contains all kinds of functions to change the appearance of the current axis and/or the current 2D/3D Viewer window.

Name	Function																					
New window	This will open a new 2D/3D Viewer window and make it the current figure window. The number of subplots are the same as in the window from which this function was started. The zoom/rotate3D state is also kept. The colormap and colorbar settings are set back to default however.																					
Subplots	This changes the subplot configuration in the current window. An extensive set of subplot configurations are allowed. To efficiently work with subplots please refer to the support of 3D matrices, paragraph 4.2, "1D, 2D and 3D mode", and the use of plotting macros, paragraph 4.4, "Using macro's".																					
Zoom	Switches the zoom flag for the current figure window. If the zoom is activated then the rotate3D is deactivated. This uses the MATLAB <i>zoom</i> function. Please see the MATLAB documentation for more information.																					
Rotate3D	Switches the rotate3D flag for the current figure window. If the rotate3D is activated then the zoom is deactivated. This uses the MATLAB <i>rotate3d</i> function. Please see the MATLAB documentation for more information.																					
Plotting Functions	This contains all plotting routines supported by matNMR and not just those presented in the panel window.																					
	<table><tr><th>Name</th><th>Function</th></tr><tr><td>Bar 2D</td><td><div>Creates a 2D bar plot. The following input window will appear:</div><div></div><div>The following fields must be specified:</div><table><tr><th>Field</th><th>Function</th></tr><tr><td>Name Spectrum</td><td>This must be either a name of a matrix variable in the workspace or a function generating a matrix as output. In case the variable is a matNMR structure (see also paragraph 3.5, "matNMR format for spectra") then the routine will extract the corresponding axis vectors (and history etc) that were saved into it. If axis vectors are specified in this input window (see below) for Td2 and/or TD1 then these are used and not the axis vectors as saved in the structure.</td></tr><tr><td>Vector for x-axis</td><td>A variable in the workspace corresponding to the axis vector for this dimension. Leaving this empty will give an axis in points unless the "Name Spectrum" is a matNMR structure (see "Name Spectrum"). Specifying an axis vector will overrule any axis vector saved in the variable structure.</td></tr><tr><td>Linespec</td><td>Leaving this empty will plot the bars as done using the standard bar plotting routine in Matlab (for more information on this see the MATLAB documentation) , meaning it will use a colormap. Specifying the linespec causes MATLAB to plot the bars as line plots. See the MATLAB documentation for more information on colormaps and linespec properties.</td></tr><tr><td>Plotting Macro</td><td>Allows for a plotting macro to be executed immediately after the plot is made, in order to reduce rendering time. See also paragraph 4.4, "Using macro's".</td></tr></table></td></tr><tr><td>Bar 3D</td><td>Creates a 3D bar plot. The following input window will appear:</td></tr><tr><td></td><td></td></tr><tr><td></td><td></td></tr><tr><td></td><td></td></tr></table>	Name	Function	Bar 2D	<div>Creates a 2D bar plot. The following input window will appear:</div> <div></div> <div>The following fields must be specified:</div> <table><tr><th>Field</th><th>Function</th></tr><tr><td>Name Spectrum</td><td>This must be either a name of a matrix variable in the workspace or a function generating a matrix as output. In case the variable is a matNMR structure (see also paragraph 3.5, "matNMR format for spectra") then the routine will extract the corresponding axis vectors (and history etc) that were saved into it. If axis vectors are specified in this input window (see below) for Td2 and/or TD1 then these are used and not the axis vectors as saved in the structure.</td></tr><tr><td>Vector for x-axis</td><td>A variable in the workspace corresponding to the axis vector for this dimension. Leaving this empty will give an axis in points unless the "Name Spectrum" is a matNMR structure (see "Name Spectrum"). Specifying an axis vector will overrule any axis vector saved in the variable structure.</td></tr><tr><td>Linespec</td><td>Leaving this empty will plot the bars as done using the standard bar plotting routine in Matlab (for more information on this see the MATLAB documentation) , meaning it will use a colormap. Specifying the linespec causes MATLAB to plot the bars as line plots. See the MATLAB documentation for more information on colormaps and linespec properties.</td></tr><tr><td>Plotting Macro</td><td>Allows for a plotting macro to be executed immediately after the plot is made, in order to reduce rendering time. See also paragraph 4.4, "Using macro's".</td></tr></table>	Field	Function	Name Spectrum	This must be either a name of a matrix variable in the workspace or a function generating a matrix as output. In case the variable is a matNMR structure (see also paragraph 3.5, "matNMR format for spectra") then the routine will extract the corresponding axis vectors (and history etc) that were saved into it. If axis vectors are specified in this input window (see below) for Td2 and/or TD1 then these are used and not the axis vectors as saved in the structure.	Vector for x-axis	A variable in the workspace corresponding to the axis vector for this dimension. Leaving this empty will give an axis in points unless the "Name Spectrum" is a matNMR structure (see "Name Spectrum"). Specifying an axis vector will overrule any axis vector saved in the variable structure.	Linespec	Leaving this empty will plot the bars as done using the standard bar plotting routine in Matlab (for more information on this see the MATLAB documentation) , meaning it will use a colormap. Specifying the linespec causes MATLAB to plot the bars as line plots. See the MATLAB documentation for more information on colormaps and linespec properties.	Plotting Macro	Allows for a plotting macro to be executed immediately after the plot is made, in order to reduce rendering time. See also paragraph 4.4, "Using macro's".	Bar 3D	Creates a 3D bar plot. The following input window will appear:					
Name	Function																					
Bar 2D	<div>Creates a 2D bar plot. The following input window will appear:</div> <div></div> <div>The following fields must be specified:</div> <table><tr><th>Field</th><th>Function</th></tr><tr><td>Name Spectrum</td><td>This must be either a name of a matrix variable in the workspace or a function generating a matrix as output. In case the variable is a matNMR structure (see also paragraph 3.5, "matNMR format for spectra") then the routine will extract the corresponding axis vectors (and history etc) that were saved into it. If axis vectors are specified in this input window (see below) for Td2 and/or TD1 then these are used and not the axis vectors as saved in the structure.</td></tr><tr><td>Vector for x-axis</td><td>A variable in the workspace corresponding to the axis vector for this dimension. Leaving this empty will give an axis in points unless the "Name Spectrum" is a matNMR structure (see "Name Spectrum"). Specifying an axis vector will overrule any axis vector saved in the variable structure.</td></tr><tr><td>Linespec</td><td>Leaving this empty will plot the bars as done using the standard bar plotting routine in Matlab (for more information on this see the MATLAB documentation) , meaning it will use a colormap. Specifying the linespec causes MATLAB to plot the bars as line plots. See the MATLAB documentation for more information on colormaps and linespec properties.</td></tr><tr><td>Plotting Macro</td><td>Allows for a plotting macro to be executed immediately after the plot is made, in order to reduce rendering time. See also paragraph 4.4, "Using macro's".</td></tr></table>	Field	Function	Name Spectrum	This must be either a name of a matrix variable in the workspace or a function generating a matrix as output. In case the variable is a matNMR structure (see also paragraph 3.5, "matNMR format for spectra") then the routine will extract the corresponding axis vectors (and history etc) that were saved into it. If axis vectors are specified in this input window (see below) for Td2 and/or TD1 then these are used and not the axis vectors as saved in the structure.	Vector for x-axis	A variable in the workspace corresponding to the axis vector for this dimension. Leaving this empty will give an axis in points unless the "Name Spectrum" is a matNMR structure (see "Name Spectrum"). Specifying an axis vector will overrule any axis vector saved in the variable structure.	Linespec	Leaving this empty will plot the bars as done using the standard bar plotting routine in Matlab (for more information on this see the MATLAB documentation) , meaning it will use a colormap. Specifying the linespec causes MATLAB to plot the bars as line plots. See the MATLAB documentation for more information on colormaps and linespec properties.	Plotting Macro	Allows for a plotting macro to be executed immediately after the plot is made, in order to reduce rendering time. See also paragraph 4.4, "Using macro's".											
Field	Function																					
Name Spectrum	This must be either a name of a matrix variable in the workspace or a function generating a matrix as output. In case the variable is a matNMR structure (see also paragraph 3.5, "matNMR format for spectra") then the routine will extract the corresponding axis vectors (and history etc) that were saved into it. If axis vectors are specified in this input window (see below) for Td2 and/or TD1 then these are used and not the axis vectors as saved in the structure.																					
Vector for x-axis	A variable in the workspace corresponding to the axis vector for this dimension. Leaving this empty will give an axis in points unless the "Name Spectrum" is a matNMR structure (see "Name Spectrum"). Specifying an axis vector will overrule any axis vector saved in the variable structure.																					
Linespec	Leaving this empty will plot the bars as done using the standard bar plotting routine in Matlab (for more information on this see the MATLAB documentation) , meaning it will use a colormap. Specifying the linespec causes MATLAB to plot the bars as line plots. See the MATLAB documentation for more information on colormaps and linespec properties.																					
Plotting Macro	Allows for a plotting macro to be executed immediately after the plot is made, in order to reduce rendering time. See also paragraph 4.4, "Using macro's".																					
Bar 3D	Creates a 3D bar plot. The following input window will appear:																					

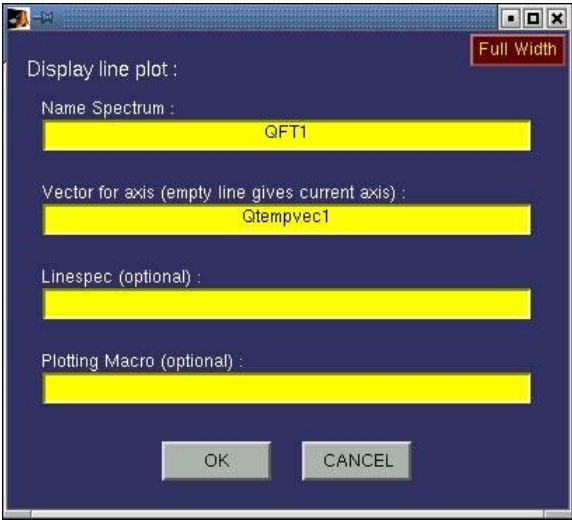


The following fields must be specified:

Field	Function
Name Spectrum	This must be either a name of a matrix variable in the workspace or a function generating a matrix as output. In case the variable is a matNMR structure (see also paragraph 3.5, " matNMR format for spectra ") then the routine will extract the corresponding axis vectors (and history etc) that were saved into it. If axis vectors are specified in this input window (see below) for Td2 and/or TD1 then these are used and not the axis vectors as saved in the structure.
Azimuth	The azimuthal angle for a 3D view, in degrees.
Elevation	The elevation angle for a 3D view, in degrees.
The type of object	Whether square bars are wanted or cylinders.
Vector for TD2	A variable in the workspace corresponding to the axis vector for this dimension. Leaving this empty will give an axis in points unless the "Name Spectrum" is a matNMR structure (see "Name Spectrum"). Specifying an axis vector will overrule any axis vector saved in the variable structure.
Vector for TD1	A variable in the workspace corresponding to the axis vector for this dimension. Leaving this empty will give an axis in points unless the "Name Spectrum" is a matNMR structure (see "Name Spectrum"). Specifying an axis vector will overrule any axis vector saved in the variable structure.
Plotting Macro	Allows for a plotting macro to be executed immediately after the plot is made, in order to reduce rendering time. See also paragraph 4.4, " Using macro's ".
Contours (abs)	More details can be found in the section of the panel window.
Contours (rel)	More details can be found in the section of the panel window.

Line plot

This function creates a line plot of the 1D spectrum that it requires as input. In case a 2D matrix is given then this results in a series of line plots distributed over the available subplots.
The following input window will appear:



The following fields must be specified:

Field	Function
Name Spectrum	This must be either a name of a matrix variable in the workspace or a function generating a matrix as output. In case the variable is a matNMR structure (see also paragraph 3.5, "matNMR format for spectra") then the routine will extract the corresponding axis vectors (and history etc) that were saved into it. If axis vectors are specified in this input window (see below) for Td2 and/or TD1 then these are used and not the axis vectors as saved in the structure.
axis	Whether or not to print an axis system around the polar plot. Only limited control is available over the axis system currently.
Linespec	Leaving this empty will plot the bars as done using the standard line plotting routine in Matlab (for more information on this see the MATLAB documentation). Specifying the linespec may result in different colours and markers and such. See the MATLAB documentation for more information on linespec properties.
Plotting Macro	Allows for a plotting macro to be executed immediately after the plot is made, in order to reduce rendering time. See also paragraph 4.4, "Using macro's" .

Polar plot

This function creates a polar plot using a stereographic projection. A 2D matrix where the dimensions correspond to two polar angles is assumed.
The following input window will appear:

Polar plot :

Matrix :
matNMR

Please Select :
☒ Axis ?

Plot type ?
pcolor

Vector with Contour levels

Lower countour limit in % of the maximum :
1

Upper countour limit in % of the maximum :
100

Multiplier (1 = linear)
1

Number of contours :
10

Type of contours
Only Positive

OK

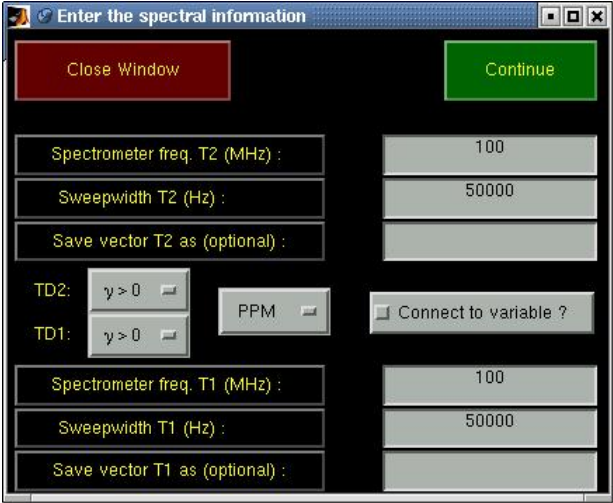
CANCEL

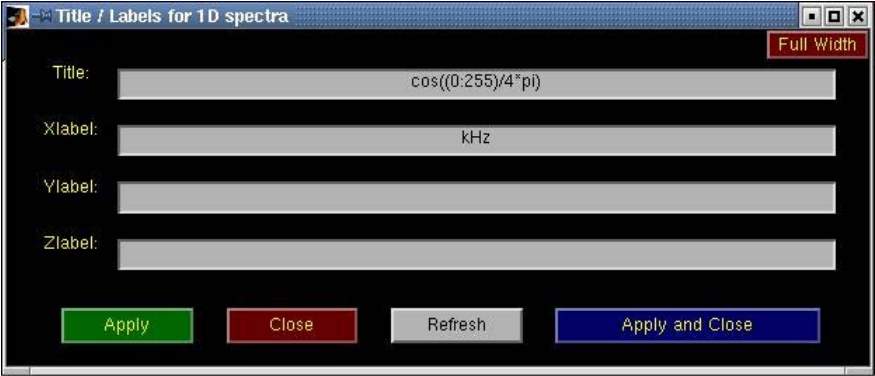
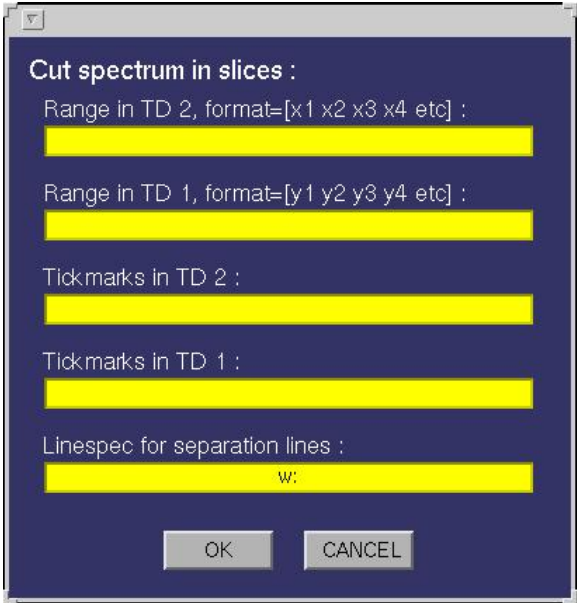
The following fields must be specified:

Field	Function
Name Spectrum	This must be either a name of a matrix variable in the workspace or a function generating a matrix as output. In case the variable is a matNMR structure (see also paragraph 3.5, " matNMR format for spectra ") then the routine will extract the corresponding axis vectors (and history etc) that were saved into it. If axis vectors are specified in this input window (see below) for Td2 and/or TD1 then these are used and not the axis vectors as saved in the structure.
axis	Whether or not to print an axis system around the polar plot. Only limited control is available over the axis system currently.
plot type	This determines whether to print it as a surface (pcolor) or as a contour plot. If the pcolor is chosen then no other options must be given. If absolute contours are asked for then only the vector with contour levels must be given. For relative contours all but the vector of contour levels must be given. Please see the corresponding sections for more details on contour plots and surface plots .
Plotting Macro	Allows for a plotting macro to be executed immediately after the plot is made, in order to reduce rendering time. See also paragraph 4.4, " Using macro's ".

[Raster 2D](#) More details can be found in the section of the panel window.

[Surface](#) More details can be found in the section of the panel window.

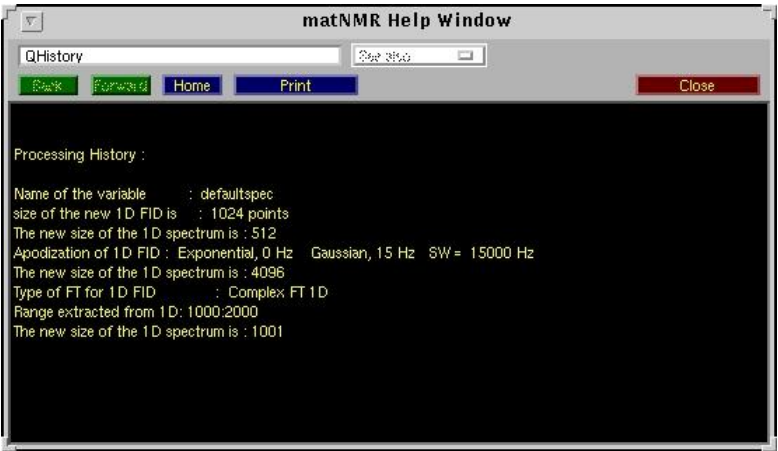
Stack 3D	More details can be found in the section of the panel window.
Field	Function
Axis rulers	<p>This allows changing the axis vectors for the current matrix. Similar to the way it is done in the main window (see paragraph 5.2.5, "Plot manipulations") the axis vectors can be set to either PPM, Hz, time, points or to a user- defined vector. The following window is opened:</p> <div></div> <p>First the type of axis vector (PPM, Hz, time, points, user-defined) can be chosen. The spectral width must be specified for PPM and Hz axes and for a PPM axis also the carrier frequency. For axes in Hz also the signs of the gyromagnetic ratios of the nuclei that correspond to the respective dimensions of the spectrum must be specified. The button labeled "Connect to variable?" can be checked if the new axes must be connected to the spectrum. If the spectrum wasn't yet a matNMR structure then it will be made into one in order to be able to allow the connection. Pressing the "Continue" button proceeds the routine and one is asked for the reference values. After that the reference peak can be specified by using the crosshair mouse pointer and clicking on the desired reference peak (This may seem strange but is necessary because of a MATLAB bug. Specifying the reference peak before asking its value leads to a mess with the main and input window). If a name is given for the new axis vector(s) then it/they will be saved in the workspace after calculating it.</p>
Color Bar	Adds or removes a colorbar from the current axis. If the colorbar must be printable in MIF format then select the corresponding button in the input window. The colorbar will then be plotted as 150 filled contours instead of the default image.
Color Mapping (caxis)	This allows setting the colormapping for the current axis, i.e. what range of values the colormap should cover. The minimum and maximum specified here are connected to the first and last color in the colormap. This uses the MATLAB command <i>caxis</i> . See the MATLAB documentation for more information.
Print	Prints the current 2D/3D Viewer window to the default printer as defined in the \$matlabroot/toolbox/local/printopt.m file.
Super title	Allows writing a line of text into the super-title axis. This axis is invisible and cannot be accessed directly. A text of certain font size and rotation angle will be written into the axis which by default is put in the top of the figure. The position of the text can be changed by dragging the mouse while clicking on the text with the left button. Clicking the right mouse over the text will bring up a menu to change the text properties.
Title/axis labels	This routine provides a means of changing the title and axes labels of a plot. A window will appear looking like:

	<div data-bbox="828 108 1700 485"></div> <p>Pressing the "Apply" button will update all labels immediately. The "Refresh" is used for updating the text labels currently written in this window to the labels in the current plot (useful when having this window opened continuously).</p>
Extract area	Allows extracting part of the current spectrum by mouse selection. Dragging a box using the mouse will define the limits. A window will appear for confirmation. Input is done in the unit of the axis vectors.
Define cut limits	<p>This function is used to make a contour plot of the current spectrum while cutting away certain areas. An input window will appear looking like:</p> <div data-bbox="976 654 1550 1259"></div> <p>Ranges in both TD1 and TD2 must be specified in pairs of coordinates in the unit of the axis vectors, defining the beginning and end of a range. A new variable "QmatNMR.Spec2D3DCutSpectrum" will be generated, which can then be used as input to any of the various plot types. In total this determines the ranges in both dimensions of the spectrum. A new matrix will be generated, using the current matrix, consisting only of the desired areas. An arbitrary number of ranges may be specified for each dimension. In the plot separation lines will be plotted which look is depending on the linespec that can be given in the input window. By default a dotted white line is drawn. Tickmarks can also be specified for each dimension separately.</p>
Define plot limits	It is often most efficient to make a low-quality plot of a spectrum, zoom into the region of interest and then replot it in high quality. This function allows to do so, by creating a new variable "QmatNMR.Spec2D3DDefinePlot" with the zoomed-in part of the spectrum and the axes. This variable may then be used as input for the subsequent plotting action.

	Get position	Like the "Get Position" routine in the main window (see paragraph 5.1, " General Appearance ") this routine provides a means of scanning the matrix. The coordinates in both dimensions for the position of the crosshair and the intensity is shown on screen. Note that the intensity is only valid for the last plotted spectrum. The coordinates should always be correct though!
	Get integral	Allows for getting the absolute integral of a designated part of the spectrum. The range must be specified using the mouse (drag a box using the left mouse button). The sum intensity over the specified region of the spectrum is given in the MATLAB main window (MATLAB prompt).
	Set integral	Allows for setting the absolute integral of the entire spectrum. A range must be specified using either the mouse (drag a box using the left mouse button) or the input window. Input of the coordinate ranges is given in the unit of the axis vectors. The sum intensity over the specified region of the spectrum is set to the integral value given in the input window.
	Create CPMs	Allows the creation of Conditional Probability Matrices (CPMs) from crosspeaks in 2D NMR spectra. For a description of the idea please see the paper JACS 127 (2005), 4466-4475. By selecting the area of a crosspeak, using the left mouse button, the CPMs are defined in a new window. The new window will show the selected area, the projections onto both axes and the corresponding CPMs. Before starting one must define a lower threshold and the number of contours used in the plots of the CPMs. For best results use a low threshold, e.g. 1%, and many contours, e.g. 25. NOTE 1: the inherent numerical instability of CPMs makes it necessary to limit the selected region to fit closely around the peak. Areas that contain much noise will invariably produce lots of artefacts in the CPMs. NOTE 2: should the routine fail to produce nice plots, typically caused by numerical instabilities, then it is foreseen that the CPMs are replotted manually. Directly after making a CPM, click on the corresponding axis that needs to be replotted. Then make a plot in the normal way, for CPMs typically a relative contour plot. The axis vectors for the CPMs are called QmatNMR.CPMvec1 (TD2) and QmatNMR.CPMvec2 (TD1), whilst the CPMs themselves are called QmatNMR.CPM1 and QmatNMR.CPM2. Thus the plots can be redrawn at will. NOTE 3: the super title that is drawn in the window by default can, as usual, be edited by right-clicking on it!
	Peak picking	MatNMR offers a simple (but limited) peak picking routine. Basically peaks can be specified and the coordinates and intensities belonging to these peaks are stored in a list. The variable used by matNMR is QmatNMR.PeakList. Peak lists are saved in the structure (for more information see paragraph 3.5, " matNMR format for spectra ") that contains either a spectrum or an FID. When redrawing a spectrum matNMR will detect the peak list and redraw the lines between and labels for all peaks. Lines can be deleted at any point by clicking the left mouse button on them. Text labels can be changed by clicking the left mouse button on them. There are two peak picking modes: one which just detects the peaks and puts a text label near the peak and one in which also lines are drawn between consecutive peaks. Whether or not a line must be drawn between two peaks is also stored in the list. Peaks are selected by dragging a box, using the left mouse button, over a region in the spectrum in which the peak is situated. Depending on the search specifications a peak is detected and added to the list. Clicking the middle mouse button will remove the last entry of the peak list. Using the "clear list" entry from the menubar will delete the entire list. Peak picking will stay active until it is stopped by clicking the right mouse button. It can be activated again at any time and new peaks will just be appended to the list. The peak list can be connected to either a structure variable in the workspace (note that if the designated variable is not already a structure it will be turned into one) or an ASCII file on disk. Peak picking also works after the "Cut contour plot" function has been executed. Furthermore when a contour plot contained a peak list at the time that the "Cut contour plot" was executed, it will be redrawn into the sliced plot properly. Changing the axis vectors, using the "Axis rulers" option will also not affect the peak list as long as axis vectors with linear increments are used. The coordinates in the peak list will be transformed to the corresponding coordinates in the units of the new axis vectors.
Direct Manipulations	The rest of this menu consists of various axis properties. More explanation concerning these can be found in the MATLAB documentation. Note the difference between changing plot properties using this menu and by changing the properties using the options menu (see also paragraph 2.2, " Setting up matNMR "): changes made with these functions only affect the current plot and not the general matNMR settings.	

6.3.3 History / Macro

Name	Function
Show History	This routine shows the processing history for the current spectrum (either 1D or 2D). A window will appear looking like:



The "Print" button allows to print the history to a printer or to disk. A separate window will be opened in which the history is plotted as graphical text (Unfortunately this is necessary because it is much more difficult to print text in UI-controls than graphical text in MATLAB).

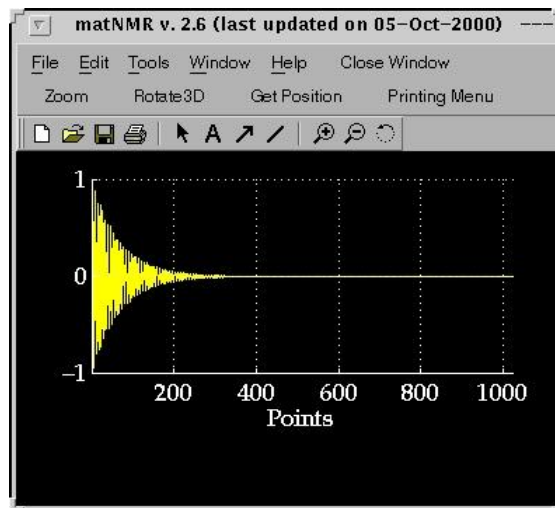
Start recording macro	Starts the recording of a new plotting macro (see also paragraph 4.4, "Using macro's").
Stop recording macro	Stop the recording of a macro. An input window will appear to ask for a name of a variable in the workspace to store the new macro in. (see also paragraph 4.4, "Using macro's")
Execute macro	This routine executes a macro. (see also paragraph 4.4, "Using macro's")
Execute macro stepwise	This routine executes a macro stepwise. The same window as used for "Reprocess from history stepwise" (see above) will appear .

6.3.4 Printing menu

The printing menu is a general feature available from all matNMR windows that is used for printing. Using a separate window the MATLAB print command (see the MATLAB documentation for more information on print) can be executed. More information on the printing menu is given in paragraph 7.1, "Printing menu".

6.3.5 Copy figure

This function will copy the current view of the main window into a separate window, but without the UI-controls. This will give something like:



The menubar of this new window contains the standard MATLAB menubar which has some useful features for printing and saving the figure. It also offers the possibility to change axis properties and stuff (More information on this menubar can be found in the MATLAB documentation). Additional features in the menubar are the MATLAB zoom and rotate3D functions, the matNMR "Get Position" (for more information see paragraph 5.1, "[General appearance](#)") and the printing menu (for more information see paragraph 7.1, "[Printing menu](#)").

6.3.6 Options

This menu contains a subset of matNMR properties that can be changed as for the 2D/3D Viewer not all menus are needed. Please use the menu in the main window for changing all other options. This menu allows changing the default settings of the screen settings and the text properties in matNMR as is described in detail in paragraph 2.2, "[Setting up matNMR \(options\)](#)".

6.3.7 Help

Besides the simple "About matNMR" and "Copyright" notices this contains a direct link to this manual from inside MATLAB. The browser, defined in the \$matlabroot/toolbox/local/docopt.m file, will be opened and the matNMR manual will appear (See the MATLAB documentation for more information on the MATLAB web and docopt commands). As the manual is included in the matNMR distribution reading the manual should be faster than using the matNMR website.

6.3.8 Clear functions

This performs the MATLAB "clear functions" command. All compiled functions are deleted from memory forcing MATLAB to recompile each function again upon running it. This is useful when working with your own m-files (or changing the matNMR code) to force MATLAB to use the altered function (See the MATLAB documentation for more information on the MATLAB clear command). Furthermore the mouse pointer is set to an arrow head.



Chapter 7 - Producing output

What good is a fancy graphics environment when you cannot make decent output? Therefore matNMR offers a direct printing service from every window. This will print the current view to the default printer as defined in the \$matlabroot/toolbox/local/printopt.m file. Furthermore there is the printing menu which provides a GUI interface to the standard MATLAB *print* command. Basically all options available for this command can be accessed from this window. This will be explained further in paragraph 7.2 just below.

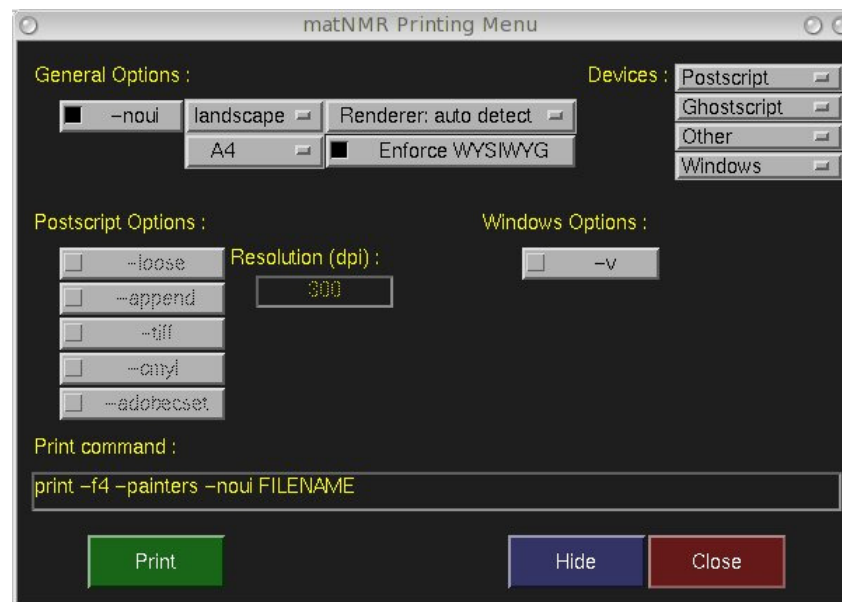
A different way of saving a picture is through the standard MATLAB *saveas* command. This allows to save the entire figure either as a binary or as an m-file. Addressing this file later will reproduce the entire figure as it was before. Some words on this can be found in paragraph 7.2 below.

Finally paragraph 7.3 explains how to use the clipboard for copying figures into other applications for people that are working with Windows.

A note must be made concerning the fact that MATLAB in general is **not** WYSIWIG. For more information on this see paragraph 3.8, [MATLAB and WYSIWIG](#).

7.1 Printing menu

The printing menu provides a GUI interface to the standard MATLAB *print* command. As the use of this routine requires some knowledge on what the MATLAB *print* command does, please see the MATLAB documentation for more information. The following window is opened:



Selection of the appropriate output device and, optionally, other features will change the command line that is displayed in the bottom of the figure window. After the appropriate selection usually a file name is required and this must be written at the end of the print command, separated by a blanc space. Only when using then "-P" option, for direct printing to a printer under UNIX, must the name of the printer device be attached, e.g. -Pvpp_phaser750. Note that any text that is typed into the print command line will be removed again when changing a selection (using the UI-controls).

The specified paper type and orientation override any current setting for the window that needs to be printed. Plots from default matNMR windows should be centered on the printed page. The button "enforce WYSIWYG" tries to obtain what-you-see-is-what-you-get behaviour, i.e. make a plot that looks exactly like the window looks on the screen. In case the absolute size of the window is bigger than the paper size the window will be resized before plotting. As this does not work perfectly in all cases a warning message will be issued. This states that it is better to resize the window by the given amount yourself. Printing the smaller window should then be really WYSIWYG (at least as far as MATLAB allows). For more information see paragraph 3.8, [MATLAB and WYSIWIG](#).

When the print command line is as desired press the "Print" button to execute the command. The figure window is moved into the background and will appear back unchanged when starting the printing menu again.

This menu can also be called from the MATLAB command line by typing "matprint". The routine will assume the current figure as the one to be printed so be sure to make the figure of interest the current figure (for more information on the MATLAB *figure* command see the MATLAB documentation).

7.2 Saving figures to disk

The MATLAB command *saveas* provides a means of saving a figure window to disk in a certain format. As this feature is not specially implemented into matNMR please refer to the MATLAB documentation for more information. Note that saving a figure in this way can create very large files!

7.3 Using the clipboard

To copy figures from MATLAB into other applications under Windows requires the standard MATLAB menubar. In general this has been disabled in matNMR because there are quite a few other items that need to be put into the menubars already. Only in the window created after executing the "Copy Figure" command (can be found in the menubar of all matNMR windows, for a graphical example see paragraph 5.2.8, ["Copy Figure"](#)) will this MATLAB menubar be present. From there objects can be selected and copied using the "Edit" menu.



Chapter 8 - Script-based processing with matNMR

NOTE: These routines still have not been tested enough yet and can contain minor bugs. Please report any bugs that you find!

It is also possible to use the processing facilities of matNMR in user-defined scripts. A number, but not all, processing actions have been created into separate functions that do not require the matNMR GUI to be open. They are, separated by mode:

Mode-independent	1D Processing	2D Processing
<ul style="list-style-type: none">• matNMRReadBinaryFID• matNMRReadBrukerSpectra• matNMRReadSIMPSONASCII• matNMRTtranspose	<ul style="list-style-type: none">• matNMRApodize1D• matNMRBaselineCorrection1D• matNMRConcatenate1D• matNMRConvertBrukerqseq1D• matNMRDCCorr1D• matNMRExtract1D• matNMRFlip1D• matNMRFT1D• matNMRLeftShift1D• matNMRLP1D• matNMRRegridSpectrum1D• matNMRRemoveBrukerDigitalFilter1D• matNMRSetPhase1D• matNMRSetSize1D• matNMRSolventSuppression1D• matNMRSwapEcho1D	<ul style="list-style-type: none">• matNMRApodize2D• matNMRConcatenate2D• matNMRConvertBrukerqseq2D• matNMRConvertEAE2D• matNMRConvertStates2D• matNMRDCCorr2D• matNMRExtract2D• matNMRFlip2D• matNMRFT2D• matNMRLeftShift2D• matNMRLP2D• matNMRRegridSpectrum2D• matNMRRemoveBrukerDigitalFilter2D• matNMRSetPhase2D• matNMRSetSize2D• matNMRShearingTransformationFD• matNMRShearingTransformationTD• matNMRSolventSuppression2D• matNMRSymmetrize2D• matNMRSwapEcho2D

The names should be self-explanatory in terms of what the functions do. The syntax for each function can be obtained by typing **help "Name Function"** in the Matlab command window. This should provide ample information. All the above-mentioned routines operate on matrices only and not on (matNMR) structures! In most cases, processing of hypercomplex matrices is not supported in the sense that both matrices cannot be entered at the same time, but must be called sequentially from the user-defined script.

It is possible to convert macros to scripts directly from the GUI. This currently only works for 1D actions but this will soon be extended to 2D actions as well.

Note that although most things can be done this way, baseline correction is not (yet) supported as this requires input of peak positions in matNMR. Furthermore, no spectral information is maintained by these functions.

Finally, an additional way of using matNMR in user-defined scripts is to record processing and/or plotting macro's, which can then be called by the function **matNMRRunMacro**. Again, the syntax to this function can be called by typing **help matNMRRunMacro** in the Matlab command window. This function does require matNMR windows to be open but doesn't rely on user input whilst processing, i.e. even though changes are made to the matNMR windows they may be hidden from view. This function can be used to either process a spectrum using a macro, OR to execute a plotting macro on the current window.

Here's an example of how one could use matNMRRunMacro in a script:

```
%
%example how to use matNMRRunMacro:
%
%This file loads all files from the current directory with names Epsilon$$$ where $$$ is a range from 0 to 90.
%The FIDs are then processed by macro "m" which was saved in file "Macro.mat". The output variable is called "Spec".
%

MacroFile = 'Macro';
MacroName = 'm';
Range = 0:90;
```

```
%load file with processing macro
eval(['load ' MacroFile]);

%start matNMR
nmr

for tel1 = 1:length(Range)
%load file
eval(['load Epsilon' num2str(Range(tel1))]);

%process the FID
eval(['MatrixOut = matNMRRunMacro(Epsilon' num2str(Range(tel1)) ' ' MacroName ');']);

%create spectral matrix in first iteration
if (tel1 == 1)
[x, y] = size(MatrixOut.Spectrum);
Spec = zeros(length(Range), x, y);
end

%fill in the matrix
Spec(tel1, :, :) = MatrixOut.Spectrum;

%clear FID from memory
eval(['clear Epsilon' num2str(Range(tel1))]);
end
```

