The ssNake reference manual

Wouter Franssen & Bas van Meerten

Version 0.7b (UNDER CONSTRUCTION)

14th June 2017

Contents

1 Running ssNake

1.1 Python versions

ssNake has been programmed to run on both the python 2.x and 3.x branch. We developed ssNake using python 2.7.9 and 3.4.3, but somewhat older version might also be sufficient.

1.2 Numpy versions

We have run into issue while using older versions of numpy. It seems that version 1.7.0 is the oldest that we support (some new commands have been introduced in this version, that ssNake makes use of). Development took place on version 1.8.2.

2 Main window

2.1 Zoom and pan controls

There are several ways to control the display of the current spectrum or FID in the main window. Below, a list of these is given:

- Dragging a box while holding the left mouse button creates a zoombox, and will zoom to that region
- Dragging while holding the right mouse button drags (i.e. pans) the spectrum. Doing this while holding the Control button pans only the x-axis, holding Shift pans only the y-axis.
- Double clicking with the right mouse button resets the view to fit the whole plot (both x and y)
- Scrolling with the mouse wheel zooms the y-axis. Doing this while also holding the right mouse button zooms the x-axis. The standard setting is 'fine'. You can zoom with more coarse steps by holding Control.
- *Special*: Scrolling while holding shift changes the spacing between the different plots in the Array, Stack or Skew plot. In the Contour plot, the contour levels are zoomed.

Having an input window open (e.g. the phasing window, see below) does not prevent the use of these zoom controls. However, some tools require left clicking in the spectrum to select a data point. In these cases, the zoom-box (dragging while holding the left mouse button) is disabled.

2.2 Bottom Frame

The bottom frame is the frame below the current data display. It consist of two parts: one with data parameters (and started by the 'Fourier' button), and the 'Get Position' frame. The features of these frames are explained below.

Data parameters

The data parameter frame features the following elements:

- The Fourier button: performs a forward or inverse fft depending on Time/Frequency toggle. If the current setting is Time, it does a forward transform (with the relevant fft shift), if it is Frequency it does an inverse transform.
- Time/Frequency toggle: displays whether the current data is an FID (Time) or Spectrum (Frequency). It is usually not necessary to manually change this, as this is done automatically. However, you can trick ssNake by making it think the current data is of the other type. This will give unpredicted results, and is not advised (do at your peril).
- Whole echo: specifies if a whole echo transform should be done. This is set by using the swap echo tool from the Tools menu (see below). When enabled, several tools respond differently (e.g. Apodizing and Set Size).
- Freq [MHz]: The centre frequency of the current spectrum dimension in MHz. Note that this value can be different from the Refrence frequency, which specifies the zero frequency of the spectrum.
- Sweepwidth [kHz]: The spectral width of the current spectrum dimension in kHz.
- Plot: A dropdown menu specifying how the data should be plot. It has four options: Real, Imag, Both and Abs. Real: plots only the real part of the complex data. Imag: plots only the imaginary part. Both: plots both the real and imaginary part of the data in different colours. Abs: plots the absolute value of the data. Note that this only toggles the *view* of the current data. Changing the data itself can be done via the Tools menu.
- Axis: Dropdown menu the change the unit of the current x-axis. For a time axis, s, ms and μ are supported. For the frequency axis: Hz, kHz, MHz and ppm.
- *Special*: Axis2: changes the unit of the y-axis in a 2D contour plot.

Get Position Frame

The Get Position frame is the frame in which the properties of a specific data point can be viewed. Pushing the 'Get Position' button and then left clicking on a specific point in the spectrum or FID prints this data. The 'Get Position frame' then displays the number of the data point you clicked on (note that in a spectrum the data goes from right to left!), the x and amplitude (y) value of this data point and the difference with your last pick. In a Contour plot, the amplitude actually codes for the z value, and there now is a separate y value box (and corresponding y difference) as well as the index of the point along the y dimension. In most plot types, a dashed vertical line is shown at the position of the cursor to aid in the selection of a point. In the Contour plot, a horizontal line is also shown.

2.3 Side Frame

On the right side of the spectrum, there is an additional frame in some cases called the Side Frame. In this, special settings for the current plot type can be set, as well as the dimension along which the data should be plot. For a regular 1D spectrum of 1D data there are no additional options, and the Side Frame is not shown. The description of the elements that are shown in the Side Frame are explained in the section on the different plot types. However, the common feature for multidimensional data (i.e. the dimension selector) shall be explained here.

The dimension selector

For data that has more than 1 dimension, a dimension selector is shown in the Side Frame. For each dimension (labelled D1 to D*n*, with *n* the number of dimensions) there is a radio button and an input box. The radio button indicates which dimension is plot in the main window. For 2D plots or arrayed plots, both the direct and the indirect dimension have to be selected. For all the dimensions of which the radio button is not selected, the input box becomes relevant to define which trace should be plot. Take for example a series of ten 1D spectra with 1024 points per spectrum. The total data size is now 10x1024 (D1xD2). Note that the direct spectral dimension is always the last dimension. In a regular view of the first spectrum of this series, the radio button below D2 should be selected (as is the default setting). The number in the box next to D1 then determines the spectrum to be plot. If this value is 0, the first spectrum is plot (note that ssNake uses the Python rules for array indexing, and that 0 is the first point). As we have ten spectra, 9 is the last. Using the up and down buttons next to the input box below D1, the spectra can be cycled. Also, scrolling with the scroll wheel when the mouse is on top of the box can be used.

For data with more dimensions, the idea is essentially the same. Note that ssNake always applies processing commands on the current view (the selected dimension). If two dimensions need to be selected (as in an array or contour plot), the leftmost radio button is the selected dimension for operations.

2.4 Message bar

The bottommost part of the ssNake window is the message bar. Any error messages will be printed here in red. Warnings about wrong inputs will be displayed in black. For both type of messages the same applies: they are shown for several seconds, and then disappear. If some tool is not responding, it is therefore wise to check the message bar for messages.

3 File

Using the ssNake load tools is quite easy. Go to File → Open. Navigating and double clicking on the desired file then loads the data. Many formats supported by ssNake (like Bruker and Varian) have their data in a folder, in which several files with a fixed name are present. For these formats, loading any of the files in this directory is accepted (ssNake searches for the expected files in the selected directory). Loading large data files might take some time, depending on your computer hardware.

When loading the data, the user is prompted to give in a name for the data by which it is identified in the workspace selector (see the Worspaces section for more on this). The default name is the name of the file (or folder for data in which the file name is fixed). If the name already exists as a workspace, the name spectrum*n* is used (with *n* the lowest integer that is not in use).

3.1 Formats

ssNake is able to extract NMR data from a number of formats from several vendors. The following table summarizes the support.

Name	Specification
Varian/Agilent	VnmrJ 2 and newer. `fid` and `data` (spectrum)
Bruker	Topspin and XWinNMR (`fid` & `ser`,`1r/i` & `2rr/ii`)
Bruker minispec	`.sig` file
JEOL Delta	1D only at the moment
Chemagnetics	???
Magritek	Both 1D and 2D data
Simpson	1D and 2D, -ascii and -binary
JCAMP	1D JCAMP-DX file
JSON	ssNake JSON file (ascii)
Matlab	ssNake `.mat` file (binary)

Generally, ssNake loads the data raw data and searches for some variables (e.g. spectral width and spectrometer frequency). The above data formats do not support data with more then two dimensions in a nice way. Getting the higher dimensions is therefore not straightforward. ssNake treats this data as 2D, after which you can split the data yourselves using Matrix → Split.

Varian/Agilent

Loading a Varian/Agilent fid file requires a *procpar* file and a *fid* file. From the *procpar*, the spectral frequency and the spectral width in one or two dimensions is extracted. The *fid* file is then checked for the version (VnmrJ 2 or 3) and the data is extracted from this file.

For processed data (*data*, in the datdir folder for example) it searches for a *procpar* file in that directory or in the parent folder, from which additional parameters are extracted.

Bruker

The Bruker load first checks for the *acqus* and *acqu2s* files and extracts the number of points in both dimensions, the spectral widths and the spectral frequency. Also, the bitorder is checked, which describes the way the data is saved in the *fid* or *ser* file. Using this, the data is extracted.

Bruker processed spectra can be loaded from *1r* and *1i* files for 1D data, *2rr* and *2ii* for 2D data, and *2rr* and *2ir* for hypercomplex data. It additionally takes parameters from *procs* or *proc2s* files in that directory. Moreover, the spectral frequency is extracted from the *acqus* and/or *acqu2s* files located *two directories up*.

Bruker minispec

In some cases, the Bruker minispec also outputs a `.sig` data file. These can be loaded in ssNake, although it is experimental at the moment.

JEOL Delta

JEOL Delta data is contained in a single `.jdf` file. Currently only 1D data is (experimentally) supported. This is simply because we have no access to a JEOL spectrometer for testing purposes. Please send us test files is you have any.

Chemagnetics

Loading Chemagnetics data requires a *acq* (and *acq_2* for 2D) and *data* file. From the former, the number of points in both dimensions is extracted, the spectral widths and the spectral frequency. All the data points are then extracted form the *data* file, and reshaped to a 2D array when necessary.

Magritek

Magritek requires two or three files: *acqu.par* for the spactral variables, and a file that ends with *.1d*. For 2D data, a file that ends with *.2d* is also needed. As with the other formats, the number of data points in the two dimensions is extracted from the parameter file (along with the spectral data) and the binary data in the *.1d* or *.2d* is unpacked.

Simpson

Simpson data files are plain text files, *.fid* or *.spe* as file extension. From the file, the number of points in both dimensions is extracted, and the spectral widths. The spectral frequency is *not* included in the Simpson format, and is put at 0. Simpson files cannot contain a mixture of time and frequency data (both dimensions must be the same type). Simpson binary data is also supported, but not raw binary.

JCAMP

JCAMP (or JCAMP-DX) is a general format for storing spectroscopic data. Although it can be applied to NMR data, it is not convenient, especially for 2D data. ssNake therefore only supports 1D fid and spectrum data at the moment. As the JCAMP format support a wide variety of data formats, it depends on the specific file if it loads in ssNake without errors. When encountering errors, do not hesitate to contact us.

JSON

Loads a JSON file that is saved with ssNake. File must have the *.json* or *.JSON* extension.

Matlab

Loads a Matlab file that is saved with ssNake. File must have the *.mat* or *.MAT* extension. Also supports .mat files that have been resaved to matlab v7.3 format.

Unsupported file formats

Currently, we support all the data formats that we have access to. If you want your favourite data format to be support by ssNake, please send us a request along with some sample data (1D and 2D).

3.2 Open

When using ssNake's load function, ssNake analyses the selected file name and the names of other files in the directory to figure out the format you want to load. ssNake only needs to check for files that are actually needed for loading the data, so removing useless files from the format does not pose a problem. Below, the checks used by ssNake in this function can be found for each format.

Name	Folder contains files	File extension
Varian/Agilent fid	`procpar` and `fid`	
Varian/Agilent spectrum	`data` and `procpar` or `../procpar`	
Bruker fid	`acqus` (and `acqu2s` for 2D) and `fid` or `ser`	
Bruker spectrum	(`1r`/`1i`) or (`2rr` and `2ii`/`2ir`) with `procs`/`proc2s` and `../../acqus` and `../../acqu2s`	
Bruker minispec		`*.sig`
Chemagnetics	`acq` (and `acq_2` for 2D) and `data`	
Magritek	`acqu.par` and `*.1d` or `*.2d`	
JEOL Delta		`.jdf`
NMRpipe		`.fid` and first byte is 0
Simpson		`.fid` or `.spe`
JCAMP		`.jcamp` or `.dx` or `.jdx`
JSON		`.json` or `.JSON`
Matlab		`.mat` or `.MAT`

3.3 Open & Combine

In case you want to load multiple data and combine them into an *n*D array, `Open & Combine` can be used. It opens a window where, via either navigating using the Browse button, or via drag-and-drop, a list of desired data can be made. Pressing OK loads and combines these data files, in a similar way as `Workspaces` → `Combine` does. In essence, `Open & Combine` does the same thing as loading all the data into separate workspaces, and then combining them. However, `Open & Combine` is more convenient, as it does not open separate workspaces for all the individual data files. Do note that only data files with the same shape can be combined (*i.e.* with the same number of data points). Also note that ssNake considers a 1D data file with n points to be different than a 2D data file with only one trace ($n \times 1$ data points).

3.4 Save data

Naturally, ssNake can also be used to save data. When saving data, the current ND data is saved, along with the spectral widths, frequencies and if the axis is in time or frequency units. Also, ppm references are saved, along with the processing history. Undo information is *not* saved.

JSON

JSON (JavaScript Object Notation) is an ascii format (i.e. regular text) used to save data structures. Within ssNake this can be used to save the current data in a clear, human readable

format. As JSON data is in ascii, it is not efficient in file size and in speed. If these are necessary then consider saving the data as a matlab binary.

MATLAB

The MATLAB binary format is a open source format in which many different types of data can be saved. Also, it is the native format of MATLAB, which can be used for more special data manipulation, if necessary. As the format is binary, it is efficient in both speed and file size. The content of the file is the same as the JSON file.

3.5 Export data

Using ssNake you can also export your data. Exporting never saves all the info our own formats do, so there will be loss of information. It is suggested you only use these formats for exchanging data to another program, but not for archiving.

Simpson

Simpson is a general NMR simulation program with its own data format. The format only supports 1D and 2D data, and only if the data in both dimensions is of the same type (i.e. both spectrum, or both time). Apart from that, Simpson is an ascii format, and therefore has the same speed restrictions as the JSON format. The parameters that are saved within the Simpson data are restricted (no offset frequency), so when loading it back into ssNake some information is lost. We therefore advise to use the Simpson format only for transferring data to another program.

ASCII (1D/2D)

This option saves the current data to a bare ascii format (text file). The first column is the axis in units of Hz or seconds. The next columns are alternating real and imaginary for all traces in the 2D data. For 1D data, only a single set of real and imaginary data is printed. The axis of dimension 1 of the 2D data is not saved.

Figure

The Figure option can be used to export any Figure plot within ssNake. The title, axis labels and size of the Figure can be changed. Be aware that choosing a too small height can cause the x-axis label to disappear. After all the titles have been set, pushing ‘Save’ will save the Figure to a directory of your choice. The Figure can be saved in vector format (.eps, .pdf, .svg), in lossless pixel format (.png), and in lossy pixel format (.jpg). We advice to use vector formats to make high quality pictures.

3.6 Quit

Closes the ssNake application. Quiting ssNake always asks for conformation, to prevent data loss.

4 Workspaces

Within ssNake you can have multiple data files opened. Each loaded file gets its own environment in which you can process the data. These are called workspaces. All ssNake workspaces are independent, meaning that the data from workspace 1 cannot be changed while viewing any other workspace. Going back to a workspace from another goes back to the same view as before: nothing is discarded when switching between workspaces. Also: every workspace has its own undo/redo information. Some tools can transfer data from other workspaces to the current workspace. You can find more about these tools later on, in their respective sections.

When loading data, ssNake prompts for a name. This will become the name of the workspace in which the data is loaded. The title of the graph will be set to this name, and the name will be set in the list of opened workspaces in Workspaces → Active. You can also switch workspace using the tabs that are created for every workspace. Note that you cannot have multiple workspaces with the same name.

4.1 Duplicate

Makes a copy of the current workspace to a new workspace, of which the name is asked. All the data and the current view is copied to this new workspace. No undo and redo data is transferred. Also, the current workspace is now the new, duplicated one.

4.2 Delete

Deletes the current workspace and all references to it. The view is changed to the next workspace in the list. Be aware that there is no way to undo this!

4.3 Rename

Renames the current workspace. Note that two workspaces cannot have the same name.

4.4 Active

Shows a list of all workspaces. The current workspaces can be changed by clicking on another workspace name.

4.5 Next

Makes the next workspace the active workspace.

4.6 Previous

Makes the previous workspace the active workspace.

4.7 Combine

Combines the data from several workspaces. This can be used, for example, to create a 2×1024 data file from two spectra with 1024 points (*i.e.* combine several separate spectra into an array). Do note that only workspaces files with the same shape can be combined (*i.e.* with the same number of data points). Also note that ssNake considers a 1D data file with n points to be different than a 2D data file with only one trace ($n \times 1$ data points). Using this tool opens a window in which, via dragging and dropping, a selection can be made from the current workspaces. Pressing OK asks for a name for the new workspace.

4.8 Keyboard shortcuts

ssNake features several keyboard shortcuts to work with the workspaces. Below is a list of all the combinations.

Combination	Effect
Ctrl + w	Close current workspace
Ctrl + n	Duplicate current workspace
F2	Rename current workspace
Ctrl + Page Up	Change current workspace to the one above in the list
Ctrl + Page Down	Change current workspace to the one below in the list

When the current workspace is the last in the list, and Ctrl + Page Down is pressed, the current workspace is set to the first in the list.

5 Macros

Macros can be used to save a particular series of action performed on the data, for later fast reuse. Say that you want to load several data files, and want to set the size of them to 4096 points and perform a Fourier transform. Doing this many times can be annoying. When you create a macro to do this, you only need to execute the macro for every data file. Thus saving time and making sure all are processed in an identical way. Also, macros are shared between workspaces: there is only one pool of macros. How to use macros is explained below.

5.1 Start recording

To create a macros, you must tell ssNake that you want to record a series of action, to be saved in the macro. Executing Macros → Start Recording prompts for a name of the future macro, and

enables the recording. From now on, all the actions you perform on the data are recorded and saved in this macro.

5.2 Stop recording

After some actions have been performed (Fourier, zero fill, phasing...) the recording of the macro can be stopped using Macros → Stop Recording.

5.3 Run

Can be used to run the selected macro on the current data.

5.4 Delete

Delete a macro.

5.5 Save

Can be used to save a macro to the disk as a .json file. This is a plain text file, in which you can see all the steps that are executed when running this macro.

5.6 Load

Loads a saved macro in ssNake. Execution of the macro can then be performed using Macros → Run.

5.7 Notes on usage of macros

Note that ssNake only records the actions that are not undone. If you perform an action when recording is on, and you undo it while recording, the macro has no entries. However, performing two actions that cancel each other *are* recorded. For example, issuing Fourier twice has no net effect, but a macro of this includes both commands.

Note that it is possible to run a macro which has been recorded on a data set to:w be used on a data set with a different number of dimensions. For this, it is important to note that ssNake counts backwards to remember the dimension in which an action is performed. When there are 3 dimensions, ssNake classifies them as the second to last, first to last and last. A recorded macro can therefore be run at data that has all the dimensions in which some actions was performed, counting from the end. Only doing a Fourier transform in the last dimension is therefore possible on every data set, regardless of the fact that it might be D5 in some set, and D1 in another.

6 Edit

6.1 Undo

In ssNake, every action that causes a change in the data can be undone. When such an action is performed, the undo list is increased with one entry , in which the information is stored as how to undo the performed action. Sometimes this involves the inverse operation (e.g. for fft/inverse fft) , but for some actions the old data must be preserved and put back when an undo is requested. Duplicating data to another workspace does not copy the undo list. The undo/redo data can be cleared using Clear Undo/Redo List (see below).

Undo cannot be used the undo the current view to the previous view as this is not an operation that changes the data.

6.2 Redo

When Undo is used, the action that is undone is put in the redo list. Using Undo and then Redo therefore performs no net action. Every time a new action (i.e. something other then redo) is performed the redo list is cleared.

6.3 No Undo Mode

Activating this checkbox starts the No Undo Mode. This clears the current undo and redo list, and prevents any further actions from adding to the undo list. This way, no undo is possible for any further actions. Normally, this is undesirable, however, in some cases the undo information takes a lot of computer memory, especially for large data sets. Having No Undo Mode activated makes it more convenient to processes this data. Also note that No Undo Mode also applies for any run macro's.

6.4 Clear Undo/Redo List

Clears the undo and redo lists of the current data. This clears any memory allocated to these list, which can be a burden when processing large data sets. Naturally, any mistakes made during processing can not be corrected after this. Use with care.

6.5 Reload

This action reloads the data from the original source. All undo and redo information is retained, so you can undo back to before you reloaded the data. Note that the undo information can take quite some memory on your computer, especially for bigger data sets. In case this gives problems: the undo/redo data can be cleared using Clear Undo/Redo List (see above).

6.6 Monitor

The monitor function can be used to automatically reload the data when the source file is changed (for example when measuring on the spectrometer). One or more macro's can be selected to

be applied every time the data is reloaded. The Monitor action opens a window, in which the macro's can be selected (via dragging and dropping). Pushing 'Watch' then starts the monitoring, which can be stopped by returning to this window and pushing 'Unwatch'. Note that ssNake takes a cool-down period of half a second after the macro's have been executed, to avoid a lock-up of the program in the case of heavy macro's and fast changing data.

7 Tools

This section explains all the options included in the Tools menu.

7.1 Real

Real puts the imaginary values of all data points at zero.

Background

In some NMR experiments, only the real value of the magnetization during a specific period is measured. In this case it is convenient to zero the imaginary part to force further processing to ignore this data.

7.2 Imag

Imag puts the real values of all data points at zero.

7.3 Abs

Abs replaces all complex datapoints by the length of the vector they span in the 2D complex plane. That is:

$$F_{\text{new}} = \sqrt{\text{real}(F_{\text{old}})^2 + \text{imag}(F_{\text{old}})^2}$$

for all points in the complex data. These values are put as the real part. The imaginary part is zeroed.

7.4 Complex Conjugate

Multiplies the imaginary part of the data with -1. Performing this in the FID leads to a spectrum which is flipped along that dimension.

7.5 Phasing

Opens a tool that can be use to change the phasing in the current spectrum or fid. Supports zero order, first order and autophasing.

Background

NMR data is usually recorded as complex data. Here the real part can be said to be the x-magnetization, and the imaginary part the y-magnetization. This gives the NMR signal as a complex vector, that is rotating in the complex plane during the signal acquisition. An ideal NMR signal consists of a cosine in the real part, and a sine in the imaginary part. This gives, after Fourier transform, a spectrum were a pure absorption lineshape (i.e. a peak) is seen. Depending on the way the time signal is recorded, there can be a distortion to this line, as a constant phase is added to both the sinusoids. With zero order phase correction, an additional phase is added to both the complex and the real part, to adjust for this effect. When expressed in complex exponentials, the following holds:

$$f_{\text{new}} = f_{\text{old}} \cdot \mathrm{e}^{i\pi\cdot\theta/180} \tag{1}$$

were f is a complex value in the time or frequency domain and θ the desired phase correction in degrees. When using this zero order phase correction, all points in the time or frequency domain get the same phase correction.

In the spectral domain, a *first* order phase correction is also defined. This leads to a phase correction that linearly depends on the frequency offset with respect to a set central frequency.

ssNake implementation

Selecting phasing in the tools menu opens the phasing window. In this windows, the zero and first order phasing can be set, as well as the reference frequency (centre point) for the first order phasing. The values can be either filled in the boxes, changed by pushing the left and right buttons, or set by dragging the slider to the left or the right. For zero order phase correction, the limits are -180 and 180 degrees. For first order correction, these are -540 and 540 degrees. Higher values for the first order correction sometimes make sense, and can be filled in the box directly. Note that the values for the zero order correction are circular, so that -180 and 180 degrees lead to the same effect.

While in this menu, the effect of a certain phasing can be immediately viewed in the ssNake window. When a good setting is reach, ‘Ok’ can be pushed to apply the phasing. Note that for multidimensional data, the preview only shows the current trace. Pushing ‘Ok’ the applies the same phasing correction on all the traces (might take some time for large data). If the phasing should only be applied on the current spectrum or fid, the box ‘Single slice’ can be ticked. Note that you still have to push ‘Ok’ to apply the phasing.

Apart from manual phasing, ssNake also supports two types of autophasing: zero order only, and zero and first order. When pushing these buttons, ssNake will search for the best phasing in the *current* trace, with only first order phasing correction, or with both zero and first order. Do note that the setting that ssNake finds might not be the best: phasing an NMR spectrum is tricky and has no unique mathematical solution. However, for a fast analysis it can be useful.

Also, note that first order phase correction has no meaning in NMR when applied to the time domain. ssNake therefore always applies first order phasing in the *frequency* domain. When first order phasing is applied when viewing the time domain signal, ssNake transforms to the spectral domain, does the phase correction, and transforms back. As Fourier transforms are quite fast, this provides no issues when viewing the effect of the phase correction.

7.6 Autophase 0

Perform a 0th order autophase on the current spectrum (and applies it on all traces). This is identical to doing this via the Phasing window, but requires less effort for the user.

7.7 Autophase 0+1

Perform a 1th order autophase on the current spectrum (and applies it on all traces). This is identical to doing this via the Phasing window, but requires less effort for the user.

7.8 Swap echo

A tool to swap the time domain data symmetrically around a selected point.

Background

In some NMR experiments, the rise and fall of a spin echo is recorded. One way of processing this data is to remove all the data points before the echo maximum, and subsequently treat the data as a regular FID. In some cases it is more useful to use the full data, rise and fall, of the echo signal. Directly Fourier transforming such data leads to massive first order phase distortion. This can be circumvented by swapping the echo around the centre. In this case, the data is split in two parts: the data from the start of the measurement until the top of the echo, and the data from the top of the echo until the last point. The latter data is put leftmost, the former right. The new signal then starts with a decay and rises again at the end. Due to the symmetry of the echo, the Fourier transform of this signal will have zero imaginary signal, while the real signal is still the same as if a regular FID is transformed. This means that phasing such a line can be easily done.

ssNake implementation

Echo swapping is done by left clicking on the echo maximum of the time signal. By default, the swapping is set at the centre of the time signal. After left clicking, a preview is shown. Pushing 'Apply' applies the transformation to all traces of this dimension. Also, the tag 'Whole echo' is set in the footer. When this is on, Fourier transforming does not multiply the first point by 0.5 (as is needed for data that decays to zero), and line broadening is applied symmetrically around zero.

7.9 Offset correction

Subtracts the average value of the selected data points from all data points.

Background

It can sometimes happen that NMR data does not decay to zero, but to another value. In this case, a time signal with zero frequency and no decay can be thought to have been added to the NMR signal. Usually, this is due to issues with the electronics, and does not imply a NMR signal with these properties. For proper analysis of the signal, this offset should be removed, as it leads to a signal at 0 Hz, which can interfere with the spectrum. This can be solved by subtracting the average value of the last points of the time domain data from all the data points.

ssNake implementation

Using the Offset correction tool opens a new window. By default, the last 20% of the time data is selected as containing only noise (of which the average is the offset which should be subtracted). Left clicking two times in the graph sets the limits between which the average value is calculated. Pushing apply subtracts this value from all time data points. If the data has more dimensions then 1, the average value is subtracted from all data points. Take note that the average is only calculated for the current graph. If you want to correct the offset for each graph you can select the 'Single slice' option, and do the correction separately in each desired trace.

7.10 Baseline correction

Fits a polynomial line of a given order through the current spectrum while ignoring the selected parts. This line is then subtracted from the spectrum.

Background

In some NMR spectra, experimental conditions were such that a non-informative baseline is present in the spectrum. This can be due to electronic reasons (e.g. pulse ringing) or due to background signals. This usually means that the first points in the time signal are corrupted. In some cases, removing the first data points is the best option, but this leads to signal loss and a large first order phasing error. In these cases, it can be more convenient to try to subtract the errors in the frequency domain. This is done by fitting a nth degree polynomial through the spectrum, and subtract this line from the spectrum. For this, it is essential that the regions of spectral information (i.e. the NMR peaks) are not included in the fit. Failing to do this will cause the baseline correction to attempt to remove the peaks.

Baseline correction is mostly a visual thing. In the case of ssNake, the degree of the polynomial to be subtracted has to be put in. Usually, putting a very high value leads to strange results. This, however, depends on the spectrum to be corrected.

ssNake implementation

In ssNake, baseline correction is performed by fitting a nth degree polynomial through the spectrum, i.e.:

$$S_{\text{corr}} = \sum_{i=0}^{n} a_i x^i,$$

with a_i the amplitude of the ith degree polynomial. Fitting this to the spectrum is a mathematical operation with a single unique solution. Prior to doing this, specific parts of the spectrum can be selectively ignored in the fitting procedure. This can be done by left clicking in the spectrum at the two limits of the area that is to be ignored. The area is then marked with a red background. This way, multiple areas can be selected. Pressing 'Fit' then performs and shows the fit. Apply subtracts the line from the spectrum. Note that by default this single fit is subtracted from *all* the traces. This is useful if the background is identical in all of them. Alternatively, the box 'Single slice' can be ticked to only apply the correction on the current spectrum.

7.11 Subtract averages

Subtracts the average of a region from the respective data. The analysis is performed independently for every slices in the nD data. This effectively does the same as Offset Correction, but then for each trace independently.

ssNake implementation

Opening this tool gives a window were the left and right limit of the selected data can be set. Alternatively, left clicking on the plot sets the start limit on the first click, and the end limit on the second click. Pushing apply performs the subtraction.

7.12 Reference deconvolution

Uses the FIDDLE algorithm to deconvolute a peak shape in the spectrum.

Background

Due to issues with magnetic field inhomogeneity (i.e. shimming), peaks in an NMR spectrum can have a weird shape with extra ridges and small peaks on the side. In some cases it is known from prior knowledge that the visible peak should be a single Lorentzian line. If so, the line can be deconvoluted to give this Lorentzian line. Naturally, all other lines in the spectrum should have the same distortions. Using the known shape of the easy Lorentzian reference, all other lines can also have their shape corrected.

ssNake implementation

Using the 'Reference deconvolution' opens a window, in which the region of the reference signal has to be set (by left clicking, or by filling in the values). Additionally, the new linewidth of the peak has to be specified. Note that the algorithm cannot increase resolution, only make better lineshapes. Filling in a too small linewidth will result in a heavily distorted spectrum. Pushing apply performs the analysis for all traces.

7.13 Correct Bruker digital filter

Corrects a Bruker fid to start at $t = 0$.

Background

In theory, the recording of an NMR signal starts directly after the excitation point. The signal can then be easily described by a decaying exponential (or another function) and the Fourier transform does not show surprises. NMR data measured on a Bruker device, however, has the peculiar attribute that it starts *before* the physical start of the signal. The origin of this lies in the way Bruker spectrometers use their digital filter, of which the details will not be described here. Direct Fourier transformation of this signal will lead to massive oscillations in the baseline of the resulting spectrum.

To be able to handle Bruker NMR data, this detection error has to be corrected. The preferred way for this seems to apply a strong first order phase correction in the spectrum that (due to Fourier principles) rolls the time domain data in such a way that the first point is now the start of the real signal. Due to this procedure, the points that were at the front of the time signal are now at the end. Apparently this leads to better results than just removing this data.

ssNake implementation

To automate this process, ssNake uses information from a handy text by W. M. Westler and F. Abildgaard. Every Bruker NMR spectrometer has its own characteristics, and therefore needs a different amount of first order phase correction. For older Bruker hardware, the amount of correction is found by extracting two parameters from the Bruker acqus file, and to use a lookup table supplied by Westler & Abildgaard. For newer hardware, this value can be read directly from the acqus file.

Using this tool opens a windows, in which you have to open (again) the current Bruker file. ssNake then finds the desired phase correction and applies it to all traces.

7.14 LPSVD

Uses Linear Prediction (LP) via Single Value Decomposition (SVD) to back or forward predict a time signal.

Background

Measuring an NMR signal right after a pulse is not possible, due to the ring-down time of the pulse, which would interfere with the measurement. Nearly all NMR experiments therefore have a short delay before data acquisition is started. However, in this time the magnetization already starts precessing, causing an offset dependent phase distortion in the spectrum. This can be corrected with a 1st order phase correction, but this can lead to a baseline distortion. An alternative way to compensate the problem of delayed acquisition is by using linear prediction. In this case, the missing data points are reconstructed from the ones that are measured. If done perfectly, the signal is now perfectly phased, as if no delay was used. Via the same method, acquisition that is stopped before the data has died out can be lengthened by forward predicting based on the measured signal.

Most LP methods require the user to supply the number of frequencies that are in the time signal, as this is difficult (impossible?) for the program to recognise.

ssNake implementation

Currently, ssNake supports only LPSVD as a linear prediction algorithm. Using the tool pops up a windows were several things need to be set:

- Whether a forward or backward linear prediction needs to be performed. [1]
- The number of points used for the analysis of the LP. These are taken from the front (backwards prediction) or from the end (forward).
- The number of frequencies in the signal. Putting this too low leads to weird effects. Putting it too high will slow down the process.
- The number of points that need to be predicted.

Pushing OK starts the analysis. The linear prediction is performed independently for every trace. The time it takes can therefore be quite long. Note that increasing the number of points used for the analysis above 200 usually has only a negative effect on the calculation time.

7.15 Hypercomplex

Hypercomplex data is data that has a real and imaginary part along more than 1 dimension. For a 2D data set that is hypercomplex, four 2D data sets are necessary to describe the data: RR, RI, IR, and II (with R = Real, and I = Imag). Generally in NMR, the 2D data is recorded with alternating phase, in such a way that groups of 2 data points in the indirect dimension must be combined to recover the four hypercomplex values. Depending on the way the data is recorded, a different procedure is necessary to construct the hypercomplex data. Currently, ssNake only

[1] Note that ssNake always calculates the LP coefficients via a forward LPSVD. This setting is therefore only for setting the direction in which data points need to be added.

retains one part of the hypercomplex data, which means that phasing must be performed before the hypercomplex transformations described below. All of these methods stat with a alternating data set consisting of [R1,I1,R2,I2], were R1 stands for the first recorded complex point with time 1, I1 for the second recorded complex point with time 1.

7.15.1 States

States conversion takes the alternating data set and makes new complex values following:

$$\mathrm{C}n = \mathrm{R}n + i \cdot \mathrm{I}n \quad .$$

So for example, if the two complex values of time delay 1 (in the indirect dimension) are $0+1i$ and $2+3i$, the result will be $2+4i$. Note that some information is lost in this way (ideally, the difference between the two value is also retained). Therefore, phasing of the data should be done (perfectly) before the States analysis.

7.16 States–TPPI

States–TPPI conversion takes the alternating data set and makes new complex values following:

$$\mathrm{C}n = \mathrm{R}n + i \cdot \mathrm{I}n$$

for odd values of n. And:

$$\mathrm{C}n = -\mathrm{R}n - i \cdot \mathrm{I}n$$

for even values of n.

7.17 Echo–antiecho

Echo–antiecho conversion takes the alternating data set and makes new complex values following:

$$\mathrm{C}n = \mathrm{Re}(\mathrm{R}n + \mathrm{I}n) - i \cdot \mathrm{Im}(\mathrm{R}n - \mathrm{I}n)$$

8 Matrix

This section describes the ssNake matrix manipulation tools. NMR data is nothing more then a data matrix of a specific dimension. Using the matrix tools, you can change the size of a dimension, split the data into more dimension, integrate, multiply, mirror etc. In the following, the different supported matrix manipulations will be introduced.

8.1 Sizing

The sizing tool can be used to edit the number of points in the current dimension.

Background

The number of point taken in an NMR time signal is usually restricted, as increasing the acquisition time leads to more noise (if the signal has decayed) or adds too much strain on the device (if decoupling is used). As the number of points in the spectrum is equal to the number of points in the time signal, this limits the *digital* resolution of the spectrum, making it less easy to extract the desired data. However, if the signal has decayed at the end, the digital resolution in the spectrum can be enhanced by adding zeroes at the end, without effecting the spectrum in terms of signal-to-noise and appearance. This is termed 'zero filling' and is often used in NMR.

If the time signal has not fully decayed, the signal abruptly changes to zero when zero-filling is applied. This leads to distortions in the spectrum (i.e. sinc wiggles) as the time signal is effectively multiplied with a block function. General Fourier theory then states that the spectrum must then be convoluted with the Fourier transform of the block function, which is $\sin(x)/x$, i.e. sinc.

ssNake implementation

Selecting 'Matrix → Sizing' creates a sizing window. In here a single number has to be filled in the box titled 'Size'. Decimal values will be rounded to the nearest integer. Apart from numbers, the command 'k' (or 'K') is also supported, which stands for '1024'. When using this definition there has to be a number before the 'k'. A valid input would therefore be: '2k'. Which sets the number of points to $2 \cdot 1024 = 2048$. Pressing 'Apply' executes the sizing command on all traces in all dimensions.

If the supplied number is larger than the current size, zeroes are added at the right-hand side of the time signal. If the value is lower, points are removed from the right-hand side of the time signal. If the whole echo type is on, the zeroes are added in the centre of the time signal, or points are taken symmetrically from the centre, if the size is reduced. Alternatively, the position of the zero filling can be specified by left-clicking in the spectrum, or by manually filling in the desired offset value in the box 'Offset'.

When applied to the frequency domain, zero filling makes no real sense. Therefore, ssNake always applies it in the time domain. So when the size is changed within a spectrum, it is transformed back to the time domain, the size is changed, and a forward Fourier transform gets the data back to the frequency domain.

8.2 Shift data

Shifts the data to the left or to the right, by removing data points on the left or on the right, and filling with zeroes on the other side. Negative shifts are left shifts, positive right shifts.

Background

Data shifting can be used if some data points on the left or right of the time signal are unnecessary or unwanted. When measuring a spin echo, for example, it is not uncommon to start recording the signal some microseconds before the expected echo top, to make sure the full signal is measured. The data can then be left shifted to remove the data points before the echo maximum, and yield a regular spectrum.

ssNake implementation

Starting the shift data tool creates an input window. Within this window, a integer number must be filled in that equals the number of data points of the desired *right* shift. Left shifts can be made by using negative values. Alternatively, the arrows on the right and left side of the input box can be used to shift the data in the direction of the arrow. The effect of the operation on the current 1D can be seen in the main window. When pushing 'Apply' the shift is performed on all traces of the nD data.

ssNake can also perform the time domain shift from within the frequency domain. When in a spectrum, an inverse Fourier transform is used to go back to the time domain. The data is then shifted with the desired measure, and a forward Fourier transform gives the spectrum of the shifted time signal. The effect of the sift on the current 1D can be seen in the main window. As this calculation is a bit more involved, it can take some time if the data set is large.

8.3 Integrate

Integrates a specific part of the current dimension on all traces.

Background

As NMR is a quantitative technique (when applied correctly), the area under the different peaks in a spectrum is a representative for the relative number of nuclei that resonant at this frequency. Calculating the integral of these peaks can therefore supply the user with an indication on the composition of the material that is analysed. Integration in ssNake using the matrix tool integrates across (specific parts) of the current dimension. Therfore, the resulting data will have one dimension fewer than before. Say, for example, you have a set of 2D data, of which every trace has the same signal, but a different intensity of this signal. When in the second dimension (in which the peak is shown) the peak can be integrated by supplying the left and right limit of the peak. This will result in 1D data, of which every point represents the integral of the selected area of one of the spectra.

ssNake implementation

When using the integration matrix tool, a window pops up, were you have to select the limits of the area you want to integrate in the current dimension. This can be done by typing in the

numbers of the data points, or more conveniently by left clicking in the spectrum. The first click sets the first limit, the second click the second limit. Pushing apply then performs the integration. Be aware that integration values can depend on, amongst others, the amount of zerofilling that is used.

8.4 Sum

Sums a specific part of the current dimension on all traces. Is implemented in the same way as integrating, only no scaling with the step sizes is performed.

8.5 Max

Takes the maximum value of the selected part in the current dimension on all traces. Implementation is the same as with integration.

8.6 Min

Takes the minimum value of the selected part in the current dimension on all traces. Implementation is the same as with integration.

8.7 Max position

Returns the x value of the maximum position (i.e. top of a peak) within the selected area for all traces.

Background

When analysing some experiments, it is useful to make a plot of the position of a peak versus some changing variable (e.g. time, temperature). This features allows the user to extract such data. It finds the maximum value within the selected region, and extracts the corresponding x-value. It does this for all traces, and therefore reduces the number of dimensions by 1. For example, if one has 2D data, of which the most intense peak in D2 shifts linearly in D1, this tool tool can be used to generate 1D data that shows this linear relation.

8.8 Min position

Returns the x value of the minimum position within the selected area for all traces. Works the same as Max position.

8.9 Average

Returns the average of the selected region. Implementation is the same as with integration.

8.10 Diff

Differentiates the data in the current dimension (i.e. returns the difference between each data point and the next data point). This is equal to the slope between the points. Remember that NMR data is the frequency domain is always depicted from right to left, so the direction of differentiating is different in the time domain than in the frequency domain (or, in fact, it is the same but *looks* different).

8.11 Cumsum

Performs a cumulative sum on the data in the current dimension. This means that each point is now equal to its old value, and those of all the previous points added. Remember (as with Diff) that the frequency domain is mirrored, and the direction of Cumsum is also different.

8.12 Extract part

Deletes all parts of the current dimension that are outside the selected region.

ssNake implementation

This function can be used to delete parts from the current spectrum or fid. Left clicking on a prt sets the first limit on the first click, and the second limit on the second click. Apply deletes all parts outside this region on all traces. When doing this in the spectrum, the spectral width and zero point (reference frequency) are also changed to keep all peaks at the same frequency.

8.13 Flip L/R

Flips (i.e. mirrors) the left and right side of the current dimension. The axis is not flipped.

8.14 Delete

Deletes specific data points from the current dimension. Should be entered in array from. For example: [1,2,40] deletes data points 1, 2 and 40. Remember that ssNake uses the python way of array indexing: 0 is the first point. The above code therefore deletes the second, third and forty–first data point. Do remember that the indexing goes from right to left in the case the current dimension is a spectrum (the frequency axis is flipped).

8.15 Split

Splits the current dimension in n parts with an equal number of data points. The newly created dimension is named D1, and all the other dimension numbers are increased by one (shifted one lower). This tool can only spilt the data in parts of equal size. It is therefore necessary to make sure that the size of the data in the current dimension is divisible by n, or in math.

8.16 Multiply

Multiplies the y-values of the data in the current dimension by a given number. If a single number is supplied, all the points are multiplied by this number. Alternatively, an array with the same length as the data can be given to specify a position dependent multiplication. For example, for a dimension with 3 data points: [1,3.34,2]. This multiplies the first data point by 1, second by 3.34, etc. Do not forget the about the mirroring in the spectral domain (see Diff)!

8.17 Reorder

Reorders the data in the current dimension. Input is an array with the new, desired positions. For example, [0,2,1] keeps the position of the first element ('0'), but interchanges the second and the third. It is also possible to supply the input array by loading a filer (as is used in Non-Uniform Sampling). If the new data length is longer than the old, the missing points are filled with zeros. Also, the length of the desired new dimension can be given. If the length of the final data is shorter than this value, some zeros are appended.

8.18 Concatenate

Concatenates (i.e. sticks) that data of dimension n after dimension $n+1$. So if the original data is 10x5x3, and dimension 2 is selected, it becomes 10x15. This can be seen as the opposite of the 'Split' function. Note that after applying this function the numbers of the dimensions in the ssNake window changes accordingly.

8.19 Shearing

Shearing data is a function most commonly used for MQMAS data (in solid state NMR, that is). It is used to correct a frequency slope along a specific dimension. The amount of shearing necessary is defined by a shearing constant, which needs to be supplied by the user. For every trace in the current spectrum, the data is shifted (left or right, depending on the sign of the shearing constant) a specific amount. Any data point that goes outside the range of the data rolls to the other side (aliasing). The size of the shift is directly related to the x-value of that specific trace in the shearing direction.

ssNake implementation

Activating Shearing opens a menu, in which the shearing constant, direction and axis must be set.

- *Constant*: The amount of shearing necessary. The shift for each trace depends on this constant, and the spectral width in the shearing direction. For MQMAS data, the constant depends on the spin quantum number, and the quantum level that is used. A selection menu with these values can be found under Shearing Constant.

- *Direction*: The direction is the axis number, over which the amount of shearing differs. Usually, this is this indirect dimension (for 2D data this is D1).
- *Axis*: The axis over which the data must be rolled. In a 2D case, this is D2.

The actual shearing can be done in two ways: in the time, or in the frequency domain. Shearing in the frequency domain is intuitive: it just rolls the spectrum to the left or right. However, if the shift is not an integer number of data points, some interpolation is required. We therefore opted to apply the shearing in the time domain. This involves an inverse Fourier transform , after which each trace gets a time dependent phasing (1st order phasing in the time domain). A regular Fourier transform gives back the sheared spectrum. In this way, no interpolation is necessary.

9 Transforms

9.1 Fourier Transform

Performs a Fourier Transform (equal to the 'Fourier' button in the bottom frame). When the current data is in time units, it performs a regular Fourier transform, when in spectrum mode, it performs an inverse Fourier transform. The type (i.e. time or spectrum) of the data is modified accordingly.

For the Fourier transform, ssNake uses the Fast Fourier Transform (FFT) form the numpy library. It also executes the corresponding fftshift after the transformation. Note that FFT is the most efficient when the data size is a power of 2. It is always wise to keep NMR data in this shape (by setting the size to a number of 'k' as explained at Matrix → Sizing).

9.2 Real Fourier Transform

Performs a real Fourier transform. This is the same as performing a Fourier transform with only the real part of the data, as can be done by using Tools → Real. In this case, signals with a lower frequency than the offset cannot be distinguished from signals with a higher frequency. This makes the spectrum symmetric around the centre. In some Figures in the literature, only the positive frequencies are showed for this reason. We have opted to keep the negative frequency part.

9.3 Fftshift

Performs an fftshift. This interchanges the left and right part of the data. Normally, this function is not needed, as ssNake keeps track of the shifting when performing a Fourier transform.

9.4 Inv fftshift

Performs an inverse fftshift.

9.5 Hilbert Transform

Performs a Hilbert Transform on the current data. This removes the imaginary data, and creates new imaginary data from the real data via the Kramers–Kronig relations. The sign of the imaginary data is lost in this way. Most commonly, a Hilbert transform is used in algorithms that take some part of the real data, and the imaginary part needs to be reconstructed.

9.6 NUS

Non-uniform sampling (NUS) is a way to reduce measurement time of a 2D (or higher) experiment. It involves measuring only selected t_1 delays, and using an algorithm to reconstruct the data. Reconstruction is necessary, as a regular Fourier transform can only deal with linearly sampled data. Most commonly, NUS is performed by choosing a regular delay time and number of points (i.e. a linear grid), and deleting selected values from this list. Processing NUS data is more involved then regularly measured data: the reconstruction process is not unique, and requires optimization.

Processing NUS data in ssNake requires the data to be reordered (using Matrix → Reorder, and selecting the NUS list), in this way points that are not measured are taken as a 0. Now, phasing should be performed. The easiest way to do this is to do a regular Fourier transform. This will lead to ugly data, but is sufficient to perform some 0th order phasing. Do another Fourier transform to get back to the time domain data. After this, NUS processing using the following methods is possible.

9.6.1 FFM

Fast Forward Maximum entropy (FFM, [1]) is a method to reconstruct the missing point is the FID by searching for the highest entropy. In our case, the entropy of the spectrum is defined as the sum of the absolute values of the complex data. FFM changes the values of the missing points in search for the highest entropy. This means that FFM reduces the noise in the spectrum (but adds uncertainty to the peak shapes and amplitudes) and favours narrow lines. Naturally, the points that *are* measured force the FFM analysis to create a spectrum that resembles the actual data.

For FFM, no additional parameters are needed, except for the NUS sampling scheme. Note that although the noise seems to be reduced by FFM, it is now equally contained in the peak shapes and amplitudes.

9.6.2 CLEAN

The CLEAN algorithm attempts to take single point in the spectrum, transfer part of it to a separate reconstructed spectrum, and subtract it from the main spectrum. The Fourier transform of the FID input mask (1's for measured points, 0's for skipped points) is used to not only subtract part of the spectrum maximum, but also the associated artifacts, that were introduced by the

NUS. After a maximum number of iterations (or if the residual spectrum maximum falls below a value) the loop is stopped, and the remainder is added to the reconstruction data. An inverse Fourier transform returns the reconstructed FID.

In short:

- Fourier transform to get the spectrum
- Select the maximum
- Multiply the maximum with the gamma factor and add it to the reconstructed spectrum (at that data point)
- Remove the max * gamma value (multiplied by the Fourier transform of the mask, and shifted for a peak at that position) form the spectrum

Steps 2–4 are repeated. Finally, the remained of the spectrum is added to the reconstruction.

The user needs to supply the sampling scheme, the gamma value (lower is better, but slower) and the threshold (stopping criterion in %, should be equal to the noise band for best results).

9.6.3 IST

Iterative Soft Thresholding (IST) is one of the methods to reconstruct the full FID from NUS sampled data. It involves the following steps:

- Fourier transform the FID
- Cut off the part of the spectrum that is above the threshold (in absolute mode)
- Add the cut off part to a separate array, and remove it from the spectrum
- Reconstruct the imaginary part of the spectrum via a Hilbert transform
- Inverse Fourier transform to get the FID
- Set the not measured parts back to zero
- (Repeat)

This is repeated for a specific maximum number of iterations, or until the residual spectrum is below a certain value. Then finally:

- The separate array is considered to be the reconstructed spectrum
- The remained of the spectrum is added to this (i.e. the noise)
- A Hilbert transform is used to reconstruct the imaginary data
- Inverse Fourier transform gives the reconstructed FID

The settings that need to be supplied by the user are:

- Positions of the spectra: list with the indices of the values that *have* been measured (the others will be zeroed every iteration).
- Threshold: at this relative intensity, the spectrum is cut (above goes to the separate array). Higher value (closer to 1) are more accurate, but more iterations are needed.
- Max. iterations: Maximum number of iterations, higher is better but slower.
- Stop when residual is below: stop when the spectrum has no points above value relative to the first spectrum (before any threshold steps). Value should be equal to the noise maximum. It is no problem if this value is too low (it might take longer). Having it too high leads to a bad reconstructions (if at 100%, no reconstruction is done at all!).

Reconstruction is done separately for every trace in the *n*D data. This also goes for the settings: the residual is compared to the first spectrum of *that* trace.

10 Fitting

In NMR spectroscopy, the shape of the resonance lines can provide information on the strength of interactions that the nucleus experiences. ssNake can, as many other programs, fit some idealized lineshapes with are often encountered in real life spectra. Fitting in ssNake ranges from trivial 'full width at half maximum' (FWHM) fitting, to the complex quadrupolar Czjzek distribution. The following section will explain the way ssNake handles the different types of spectra it can fit, and which equations are used to simulate the different spectra.

10.1 Fitting basics

Roughly speaking, ssNake uses two types of fitting procedures: those with a mathematical solution, and those who have to be approximated iteratively. An example of the former type is the fitting of the width of a peak at halve height (FWHM). This involves nothing more than finding the left and right side of the peak (at halve height) and to calculate the width of this interval. This can be done in one easy mathematical procedure, and involves no iterative loop. The second type of fitting procedure is used for the powder lineshapes (e.g. quadrupolar and chemical shift lineshapes). With these problems, there is no direct mathematical solution. We are searching a parameter space for the solution that best fits the experimental spectrum. This is done by trying different settings, and qualifying how well these fit. Based on these attempts, a new setting is tested. This is continued until an optimum has been found. There are several ways to do such an iterative fitting: ssNake uses the Nelder-Mead algorithm [2] (also known as "Simplex").

Powder distributions and line shape fitting

ssNake supports fitting procedures for several idealized lineshapes commonly encountered in solid state NMR. In the following sections, the equations for these shapes fill be given. In all cases, the fitted shape represents an ideal version assuming perfect excitation, and no mixed lineshape effects (e.g. quadrupolar and CSA at the same time). Note that the quality of the fit is determined by the digital resolution, which is the same as in the spectrum that is fitted. In many cases, convergence towards to optimum solution is difficult, as our fitting procedure is sensitive for local optima. It is therefore good practice to have a good initial guess. Use the 'Sim' button to show the current guess, and put all variables to reasonable values.

Simulation of powder spectra is done by getting intensities and frequencies of all the crystallite angles contained in a crystal file. These crystal files (which are not files in our case, but merely matrices) are made on the fly by ssNake depending on the 'Cheng' number supplied by the user from the fitting window. Higher values of this number lead to more powder orientations and more accurate results. Note that the number of angles increases exponentially with the Cheng number, which can lead to very slow fits. The resulting frequency/intensity pairs will be add to a vector of the current intensities. Nearly always, the found frequency is not a value of the x-axis, in which case it is shifted to the nearest point that is. This introduces a small error in the simulation, but if the digital resolution is good enough to show the pattern nicely, this error will be very small. Also note that values that are not within the limits of the x-axis are discarded, no aliasing is performed.

ZCW crystal files

As explained in the previous section, simulation of powder patterns requires averaging over a sphere. In practice, only a finite amount of angles (the polar angles θ and ϕ) is selected and saved in the crystal file. There is no perfect way to do this, and therefore there exist multiple methods to do this. We use the ZCW method (named after Zaremba, Conroy, and Wolfsberg [3–5]), being both fast and accurate. The required input is a single number: the 'Cheng' number. This number specifies the amount of crystal orientations via the Fibonacci series, with Cheng(0) being '1', Cheng(1) '2' and subsequent values the sum of the two previous:

$$F_M = F_{M-1} + F_{M-2} \quad . \tag{2}$$

From this, a list js is made, going from 0 to 1 (not including) with the number of steps equal to the Fibonacci number. Phi is equal to:

$$\phi = \frac{2\pi}{c_2} \cdot \mathrm{mod}(F_{M-1} \cdot js, 1) \tag{3}$$

with c_2 a scaling factor (see below) and F_{M-1} the value of the Fibonacci series for Cheng -1. In practice this means that (due to the 'mod'), the list makes F_{M-1} passes through the region $0{:}2\pi/c_2$, with slightly different values for every pass. Theta is equal to:

$$\theta = \mathrm{acos}(c_0(c_1 \cdot js - 1)) \quad . \tag{4}$$

In the above, the c parameters define the limits over which the angles are calculated. ssNake provides three options for this: a full sphere, a hemisphere (the upper half) and an octant (a quarter of the upper half). In the simulations that ssNake performs, the symmetry of the NMR interactions is high, and only the octant needs to be sampled. The c values are:

$$\mathbf{c} = \begin{cases} (1,2,1) & \text{full sphere} \\ (-1,1,1) & \text{hemisphere} \\ (-1,1,4) & \text{octant} \end{cases} . \tag{5}$$

The source code of this can be found in `zcw.py`. For more information on powder averaging see the paper by Mattias Edén [6].

10.2 S/N

S/N (or SNR) calculates the signal-to-noise ratio between a defined section of noise, and a peak. SNR is a useful way to describe the quality of a spectrum, with too low values being a sign for inaccurate data. In ssNake, opening the S/N tool creates a window, in which the limits of the section of noise has to be set, as well as the region with the peak of interest. Left clicking in the plot can set these values (noise limits first, then signal limits). Note that it is essential that the region that is specified as 'noise' really is noise. Any remaining signal will disrupt the SNR calculation. When the limits are set, ssNake directly calculates the SNR, using:

$$\mathrm{SNR} = a_{\max}/\mathrm{std}(\mathrm{NoiseRange}) \tag{6}$$

with $a_{\max}$ the maximum value within the range with the signal, and std(NoiseRange) the standard deviation of the noise.

10.3 FWHM

FWHM, or Full Width at Half Maximum, is a tool to calculated the width of a peak. Selecting the tool in the menu creates a window, in which the boundaries of the peak of interest have to be set, for ssNake to search for the width of the peak. Make sure that there is only one peak in the selected window, otherwise the result might be inaccurate. The easiest way to select the regions is by left clicking in the spectrum. After valid input is given, ssNake directly calculates the FWHM in the units of the current axis. The order of input of the two limits is not important for the eventual calculation.

10.4 Centre of Mass

Calculates the centre of mass of a selected part of a spectrum or FID. For a symmetric peak, this is equal to the centre of the peak. However, for an asymmetric peak, identifying the centre is not possible via visual methods and a centre of mass calculation is required. Opening the tool

creates an input window, in which the limits have to be set (as with the FWHM tool). After these are set, the centre of mass is calculated in units of the current axis, using:

$$R = \frac{1}{M} \sum_i y_i \cdot x_i \quad , \tag{7}$$

with $M = \sum y_i$ and y_i the y-value corresponding to x-value x_i.

10.5 Second order quadrupole static

This fitting routine can be used to fit a static powder lineshape that is caused by the second order quadrupolar effect. Only the central transition is considered. A simplex method is used to find the (local) optimal solution. It is therefore necessary to make sure the initial parameters are somewhat close to the final fit, especially for fits with multiple sites. General values that need to be set are:

- The Cheng number (number of powder averages)
- The spin quantum number I (only half-integer $> 1/2$)
- The background (constant value added to all data points)
- The slope (linear factor added to all data points, is zero at $x = 0$ and uses the x-values in Hz).

And for each site:

- Position: centre of the lineshape. This value is added to all x-values from the simulation. Scale is in Hz.
- C_Q: quadrupolar coupling constant in MHz.
- η: the asymmetry parameter
- Integral: total integral of the lineshape
- Lorentz: amount of Lorentzian linebroading in Hz
- Gauss: amount of Gaussian linebroading in Hz.

A maximum of 10 sites can be fitted simultaneously.

Formulae

The second order quadrupole static relies on some relatively standard formulae: those of the second order quadrupolar coupling. This equation can be split up in two parts: an angle independent factor, and the angular factors. This latter part is then split in three parts depending on the influence of the asymmetry parameter η. In short:

$$\nu = -\frac{3(C_Q/2\pi)^2}{6\nu_0(2I(2I-1))^2}(I(I+1)-3/4)(A+B\cdot\eta+C\cdot\eta^2) \tag{8}$$

With the angular factors:

$$A = -\frac{27}{8}\cos(\theta)^4 + \frac{15}{4}\cos(\theta)^2 - \frac{3}{8} \tag{9}$$

$$B = \left(-\frac{9}{4}\cos(\theta)^4 + 2\cos(\theta)^2 + \frac{1}{4}\right)\cos(2\phi) \tag{10}$$

$$C = -\frac{1}{2}\cos(\theta)^2 + \frac{1}{3} + \left(-\frac{3}{8}\cos(\theta)^4 + \frac{3}{4}\cos(\theta)^2 - \frac{3}{8}\right)\cos(2\phi)^2 \tag{11}$$

and the regular definitions for the quadrupolar coupling constant C_Q (in Hz) and asymmetry parameter η. For a given cheng number (i.e. number of powder orientations), the A, B and C parameters are constant, and only have to be calculated once for the fitting procedure.

10.6 Second order quadrupole MAS

This fitting routine works the same as `Second order quadrupole static`, but now for a case with magic angle spinning (MAS) at 'infinite' rate. The general workings are the same, only the equations differ.

Formulae

$$\nu = -\frac{3(C_Q/2\pi)^2}{6\nu_0(2I(2I-1))^2}(I(I+1)-3/4)(A+B\cdot\eta+C\cdot\eta^2) \tag{12}$$

With the angular factors:

$$A = \frac{21}{16}\cos(\theta)^4 - \frac{9}{8}\cos(\theta)^2 + \frac{5}{16} \tag{13}$$

$$B = \left(-\frac{7}{8}\cos(\theta)^4 + \cos(\theta)^2 - \frac{1}{8}\right)\cos(2\phi) \tag{14}$$

$$C = \frac{1}{12}\cos(\theta)^2 + \left(\frac{7}{48}\cos(\theta)^4 - \frac{7}{24}\cos(\theta)^2 + \frac{7}{48}\right)\cos(2\phi)^2 \tag{15}$$

11 Plot

The plot menu allows the user the change the view and nature of the current data. Several types, for both 1D and 2D views are supported. Apart from the view, the current x-axis can be changed, and a frequency reference can be set and saved.

11.1 1D Plot

Shows the default 1D plot with a solid line. The side window only shows the dimension selector, as there are no additional options.

11.2 Scatter Plot

This plot type is essentially the same as the 1D Plot. However, the data is now plot as points without a line. Scatter plots can be useful for viewing plots with only a couple of points.

11.3 Stack Plot

A stack plot is a way to view 2D data. Every trace is plot as a separate line, with a certain y-offset depending on its number. The y-axis in this view must be view with care, due to the offset given to each trace. In a stack plot, the dimension selector show two bullets: a left one for the main view, and the right one for the stack. For a T1 array, for example, having D2 as the first, and D1 as the right bullet shows a plot of the data along D2, with a separate spectrum (or FID) for every point in D1 (the arrayed recovery time in this case).

The side frame shows several options:

- From: index from where to start with the stack (zero is the start).
- To: index where to stop with the stack. Note that this should be 1 higher than the last point (it is to not including the shown value)
- Step: step size of the From to To array. 1 shows every spectrum, 2 only every second spectrum, etc.
- Spacing: the distance in units of y between each spectrum (or FID). This can also be scrolled using <shift – scroll>.

11.4 Array Plot

This plot is essentially the same as the Stack Plot, only now the spectra are plot next to each other. In this case, the labels of the x-axis are not shown anymore, as the do not make any sense. The label of the x-axis is changed to 'Frequency' or 'Time' to indicate a series of spectra or FID's. The x-limits of every subplot are taken from the view before changing to the Array Plot, and cannot be changed from the array plot.

The settings in the side frame are the same as with the Stack Plot.

11.5 Contour Plot

A contour plot shows the height profile of the data using constant height lines (like altitude lines on a map). This is the most common way to view true 2D data (data in which there are two NMR dimension, which are both Fourier transformed). Calculation of the contour lines takes some time, especially for larger data sets. As with the Stack and Array plot, the dimension selector shows two bullets. Now they stand for the x-axis (left) and y-axis (right). The Get Position frame now shows some extra features, as each data point now has a x-value, y-value and amplitude (z-value).

The side frame has several options, which are described below.

Contour type

- *Number*: the number of contour lines plot. Positive and negative lines are treated separately (so the real maximum is twice this number). Also, when the multiplier type is chosen, the total number of lines depends on whether or not the level maximum is exceeded.
- *Sign*: whether or not to plot only the positive contours (+ only), only the negative (− only) or both (Both).
- *Type*: Linear or multiplier. Linear has even spacing between the contour lines, and goes in 'Number' steps from the contour minimum to the maximum (for both the negative and positive contours). Multiplier takes the lowest contour and multiplies it with the Multiplier to get the 2nd contour. This is continues until the contour maximum, or the maximum number of lines is reached.
- *Multiplier (optional)*: Supplies the multiplier value as described above.

Contour limits [%]

Sets the maximum and minimum contour. Note that the maximum always refers to the absolute maximum of the data that is plot (real or imaginary). The maximum cannot be higher than 100%, and the minimum cannot be lower than 0%. Whenever the minimum is higher than the maximum, the meaning (but not the labels) is interchanged. The minimum contour level can be scrolled using the <shift – scroll> with the mouse wheel.

Projections

The projections are the regular 1D plots shown at the top and right of the contour plot. These can be of help in viewing the data. For both, several types are available:

- Sum: the sum of the data
- Max: the maximum of the data
- Min: the minimum of the data
- Off: no projection is displayed

Sum, Max, and Min are applied along the axis opposite to the projection. So the top projection (which shows the x-axis) is summed (max/min) along the y-axis.

12 Combine

The combine menu options allow the interaction between different workspaces like multiplication and addition. For some of these tools it is essential that the workspaces are the same size, or can be broadcasted. Broadcasting is described below.

12.1 Broadcasting

ssNake makes use of the general broadcasting rules of the numpy python package. Generally, data that has less dimensions than the target can be added (multiplied, etc) if the last dimensions agree on size. For example, 3x1024 can be added to 2x3x1024 (the data is just stretched along a new third dimension). Alternatively, data that has a size 1 along a specific dimension can also be added: 2x1x1024 added to 2x10x1024 (again, the data is copied in the second dimension to reach length 10). Note that always the last dimensions are checked, so adding 1024x3 to 1024x3x10 is not possible: checking is done from the right.

12.2 Combine Workspaces

Using Combine Workspaces opens a window that allows for the selection of multiple workspaces via dragging and dropping in the right-hand list. Only workspaces with the same size can be combined (and error will be given otherwise). The selected workspaces will be combined and copied to a new workspaces, of which the name will be asked. Combining multiple sets of data this way requires the creation of a new dimension, which is always the new dimension D1. For example, combining two 1D data sets with 1024 points will lead to a combined data set of 2x1024 points (D1xD2). Combining two sets of 10x512 data points leads to a 3D set of 2x10x512.

12.3 Insert From Workspace

Insert From Workspace can be used to append data from another workspace to the current. For this, it is *not* necessary to have two Workspaces of exactly the same size. However, appending must be possible in some way. The way the data is appended depends on the current dimension. For example: the current data has size 2x1000 and D2 is viewed. Inserting a 1D data set of 500 points leads to a 2x1500 data set. Doing the same when viewing the D1 dimension gives an error. It would be possible to append a 1D set of 1000 points to this data from D1. This gives back a 3x1000 data set.

12.4 Add

Add the data of the selected workspace to the current workspace. The active dimension is not considered. Data of both workspaces must be the same size, or must be able to be broadcasted (see above).

12.5 Subtract

Same as add, but now subtracting.

12.6 Multiply

Same as add, but now multiplying the data sets. Note that ssNake by default has complex data. Using Multiply therefore does a complex multiplication. If this is not desirable, first use Tools → Real to get rid of the imaginary part of the data.

12.7 Divide

Same as add, but now dividing. Note that the data is complex, as described for Multiply.

13 History

13.1 History

Whenever ssNake performs an action on the data, a line is added to the history list to be able to show the processing history. In the History menu, these entries can be viewed. Saving the data (to a JSON or MATLAB file) also saves the history (but not the undo information). Note that changing the view of the data is not an action, nor is fitting the data. Changing the spectral width and such *is* considered an action.

13.2 Error Messages

Whenever a wrong input is given to ssNake, a message is printed in the status bar (on the bottom of the main window). In Error Messages, these messages can also be found (printed in black), with a time stamp of their occurrence. Apart from these warning messages, there can also be more serious issues. These are python errors, and indicate that we did something wrong, or forgot to check if the input is correct. These messages are not displayed in the status bar, but only in the Error Messages window. They are printed in red, and when selected show some additional information in the bottom area of this window. Whenever a tools gives no, or an unpredictable result, consult this window for any errors. If such an error occurs you can email us with the specifics. For this, it is useful to be able to reproduce the error, otherwise it becomes difficult to find the cause.

14 Utilities

14.1 Chemical shift conversion tool

In NMR literature there are several definitions in use for describing the chemical shift tensor. This tool provides an easy way to inter convert them. The definitions are based on the very useful website of Klaus Eichele (http://anorganik.uni-tuebingen.de/klaus/nmr/index.php?p=conventions/csa/csa). In the most useful conventions, the three tensor values δ_{xx},

δ_{yy} and δ_{zz} are rewritten to an average value, a width and an asymmetry. Below follow the definitions of the different conventions.

The primary convention is called the 'Standard Convention'. This convention just gives the three principal components of the tensor, without any restructuring, called δ_{11}, δ_{22} and δ_{33}. They are ordered in such a way that: $\delta_{11} \geq \delta_{22} \geq \delta_{33}$.

The second convention we call the 'xyz Convention'. This is essentially the same as the Standard Convention, only the ordering is different. We now have δ_{xx}, δ_{yy} and δ_{zz}. With $|\delta_{zz}-\delta_{iso}| \geq |\delta_{xx}-\delta_{iso}| \geq |\delta_{yy}-\delta_{iso}|$. With δ_{iso} the isotropic value, defined as the average of the three principal components. In this definition, δ_{zz} is the value furthest away from the isotropic value, δ_{yy} the closest, and δ_{xx} the remaining value.

The third definition is the Haeberlen definition (or Haeberlen-Mehring-Spiess). Here, the principal components are ordered as in the xyz convention, only now they are redefined into a width, asymmetry and isotropic value. With $\delta_{iso} = (\delta_{xx} + \delta_{yy} + \delta_{zz})/3$, $\delta_{aniso} = \delta_{zz} - \delta_{iso}$ and $\eta = (\delta_{yy} - \delta_{xx})/\delta_{aniso}$. This way, the asymmetry parameter η always lies between 0 and 1. Not that due to the ordering of the principal components, the sign of the anisotropy (i.e. the width) is ill defined for $\eta = 1$.

The final definition is the Herzfeld-Berger definition. This convention uses the Standard Convention as a basis, and then defines the isotropic value as $\delta_{iso} = (\delta_{11} + \delta_{22} + \delta_{33})/3$, the span as $\Omega = \delta_{11} - \delta_{33}$ and the skew as $\kappa = 3(\delta_{22} - \delta_{iso})/\Omega$. This way, the span is always positive, while the skew lies between -1 and 1.

Use of the tool

Using the chemical shift conversion tool is quite easy. You can fill in all the necessary values for one convention, and push the 'Go' button next to it to convert it to all the other definitions. Note that ssNake also checks the definitions. So if you mess up the order of the Standard Convention, for example, it will be put in the correct order. The same is done for values that are outside the defined limits (e.g. η and κ). Pushing 'Reset' will reset all the values to 0.

14.2 Quadrupole coupling conversion tool

Nuclei with a spin quantum number greater than 1/2 have an asymmetric charge distribution. This leads to a quadrupolar moment, and thus to a coupling between the nucleus and the electronic field gradient at the nuclear site. Just as for chemical shift, there are multiple convention in use to describe the resulting quadrupolar tensor. Apart from being a symmetric tensor, Laplace relation states that the sum of the field gradients in all direction should be zero (otherwise the nuclei would experience a net force). Due to this, there is no isotropic term, and the quadrupolar coupling can be described by two numbers: the strength and the asymmetry. Of these there are two conventions in use: C_Q and ω_Q. When the field gradients in the three

independent directions are V_{xx}, V_{yy} and V_{xx} (with $|V_{zz}| \geq |V_{yy}| \geq |V_{xx}|$), C_Q is defined as:

$$C_Q = \frac{eV_{zz}Q}{\hbar} = \frac{e^2qQ}{\hbar} \tag{16}$$

or the more commonly used $C_Q/2\pi$ (which has Hz as unit):

$$C_Q/2\pi = \frac{eV_{zz}Q}{h} = \frac{e^2qQ}{h} \quad . \tag{17}$$

With e the elementary charge, Q the quadrupole moment (in m^2), h the Planck constant (and $\hbar$ the reduced Planck constant). Alternatively, a value that is scaled with the spin quantum number I is used:

$$\omega_Q/2\pi = \frac{3e^2qQ}{2I(2I+1)h} = \frac{3}{2I(2I+1)}C_Q/2\pi \quad . \tag{18}$$

For both definitions, the asymmetry is defined as:

$$\eta = \frac{V_{xx} - V_{yy}}{V_{zz}} \tag{19}$$

which, due to the ordering of the fields gradient components, is always between 1 and 0.

Generally, only the C_Q and ω_Q definitions are used, and the field gradients themselves are not shown. However, if the quadrupolar coupling for a nucleus in a material is known, the field gradients can be calculated by using the known quadrupolar moment of the nucleus. This way, the quadrupole coupling for another isotope at the same site in the material can be calculated by reversing the equation with a new moment and spin quantum number.

Use of the tool

Using the quadrupole conversion tool is quite easy. You can fill in all the necessary values for one convention, and push the ‘Go’ button next to it to convert it to all the other definitions. Note that ssNake also checks the definitions. So if you have an asymmetry below 0, it will reorder the field gradients in such a way it lies between 1 and 0 again. To calculate the field gradients from the quadrupole coupling constant (or the other way around), the quadrupolar moment should be given (which starts at ‘ND", being ‘not defined’). When starting from the field gradients, ssNake checks if they sum to 0 (which they should). If not, the difference is subtracted from the three values, to make sure it is. Note that only the absolute quadrupolar coupling is found in NMR, so pay no heed to the absolute sign of the field gradients.

14.3 NMR table

This tool provides an interactive way to find and calculate NMR frequencies and other characteristic values. The main view is the periodic table, showing the NMR frequency in MHz at a specific magnetic field for the most common isotope of each element. Changing one of the values (and pressing enter) updates all other values. Most commonly, either the B_0 field (in the top left corner) or the ^1H frequency is supplied by the user. Each element also has a coloured box

representing the spin quantum number (of the most common isotope). Apart from the B_0 field, the top left corner also shows the electron frequency in GHz.

Additional details on specific element can be obtained by double clicking on the element title. This creates a pop-up box showing these details. Navigating between the elements can be done from this box via the element number or abbreviation. Show values are:

- Mass
- Spin quantum number (I)
- Natural abundance and radio-active life-time (when applicable)
- Gyromagnetic ratio (γ) in 10^7 rad s^{-1} T^{-1}
- Quadrupole moment (Q) in fm^2 (when applicable)
- NMR frequency at the given magnetic field in MHz
- Sensitivity relative to a specific nucleus. The sensitivity is taken to be proportional to $\gamma^3 I(I+1)$, scaled with the natural abundance.
- Reference sample for ppm scale
- Reference sample condition
- Line-width factor for quadrupole nuclei ($I > 1/2$). It is defined as $\frac{Q^2(2I+3)}{I^2(2I-1)}$

The leftmost entry of the details window is a bullet, which can be used to set the preferred isotope of this element in the periodic table view.

15 Contact

To contact the `ssNake` team write to `ssnake@science.ru.nl`.

Bibliography

[1] N. M. Balsgart and T. Vosegaard. *Journal of Magnetic Resonance*, **223**, 164 (2012).

[2] J. A. Nelder and R. Mead. *The computer journal*, **7**, 308 (1965).

[3] S. Zaremba. *Annali di matematica pura ed applicata*, **73**, 293 (1966).

[4] H. Conroy. *The Journal of Chemical Physics*, **47**, 5307 (1967).

[5] V. B. Cheng, H. H. Suzukawa Jr and M. Wolfsberg. *The Journal of Chemical Physics*, **59**, 3992 (1973).

[6] M. Edén. *Concepts in Magnetic Resonance Part A*, **18**, 24 (2003).