QCPMG processing in ssNake

17th January 2019

1 Introduction

This tutorial will explain how QCPMG NMR data can be processed with ssNake. The tutorial provided with the ssNake program is considered as prior knowledge. If you have not yet studied this, please do so before continuing with this example.

QCPMG stands for Quadrupolar Carr-Purcell-Meiboom-Gill. The sequence is used to record a series of consecutive echoes. Via careful processing the signal-to-noise ratio of this data can be higher than when only a single echo would have been recorded. The experiment is particularly helpful for broad quadrupolar lineshapes, where the decay of the signal is much faster than the loss of magnetization by relaxation (T_2). In this case, a large number of echoes can be recorded in one go.

QCPMG experiments can be processed in a number of ways to produce either a spikelet pattern, or a regular NMR spectrum. Both have advantages in specific circumstances.

2 Data

The data provided with this tutorial is a ^{35}Cl spectrum of magnesium chloride ($MgCl_2$) recorded at 20 T using 15.6 kHz MAS.

3 Processing

3.1 Sum echo spectrum

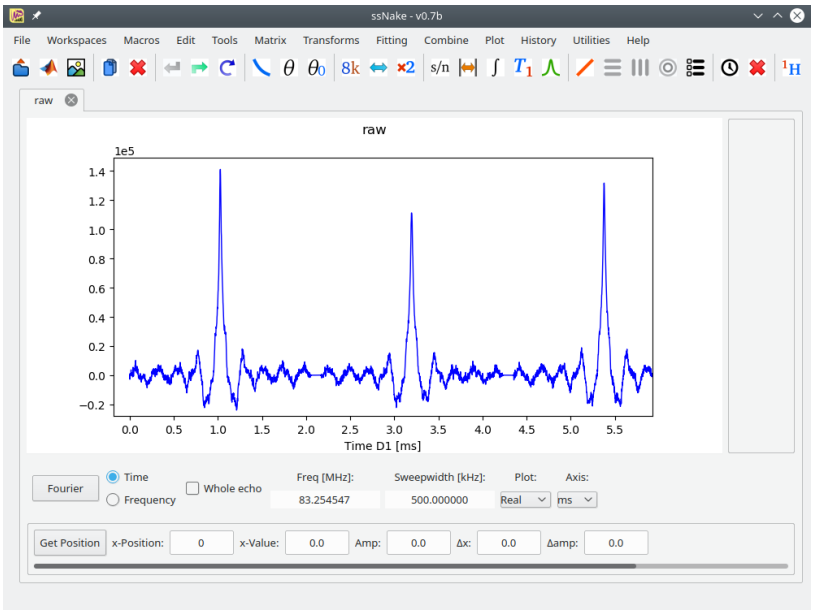
The most general way of analysing the series of echoes that are recorded in a QCPMG experiment is by adding all the individual echoes, and process it as you would a single echo experiment. This results in a spectrum which is easy to interpret and has a higher signal-to-noise ratio compared to a regular echo experiment. However, processing can be a bit tricky.

The data supplied in this tutorial has 137 recorded echoes, consisting of 1088 data points each. In order to sum these, we have to split the data in 137 parts to create a pseudo 2D data set with a shape of 137×1088. The number of echoes is chosen when setting up the experiment, but can also be determined afterwards. Determining the number of points per echo can be more

tricky. Dividing the total number of data points by 137 should provide this number, but in our case the QCPMG data has zeroes appended to the echoes. What we do know is that only with the correct number of points (1088) the splitting of the data results in aligned echos.

- Open the data in the `Raw` directory

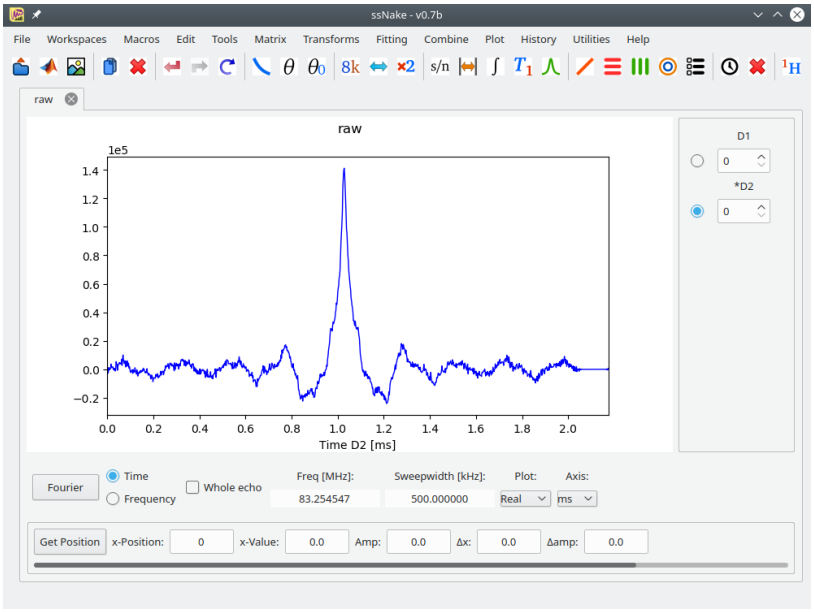
Each echo has some zeroes appended to it, which is a consequence of how the Varian equipment acquires the QCPMG data. This can be seen in the following Figure (from the start of the FID):



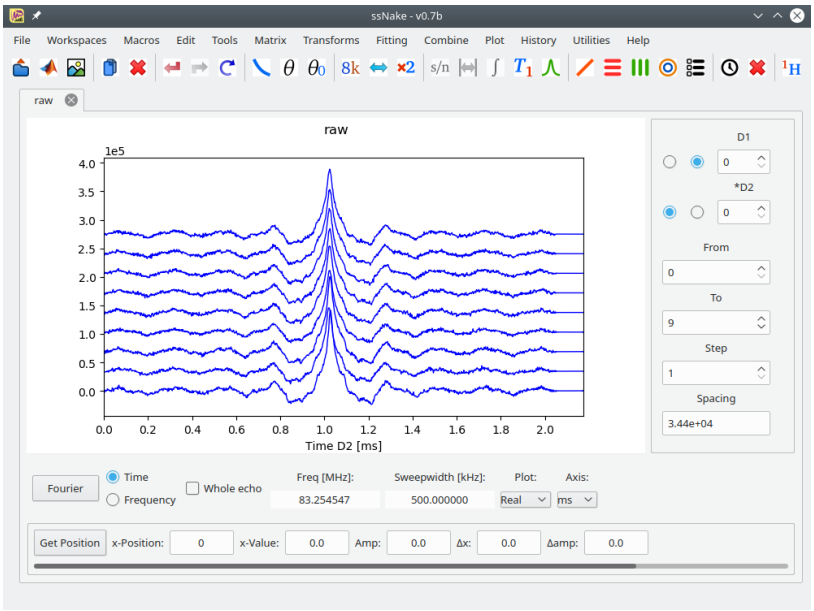
which shows the first 2 and a half echoes. We now must split the data in the 137 echoes.

- Set the size to $137 \cdot 1088 = 149056$ data points (Matrix $\longrightarrow$ Sizing)
- Split the data: Matrix $\longrightarrow$ Split (137 sections)

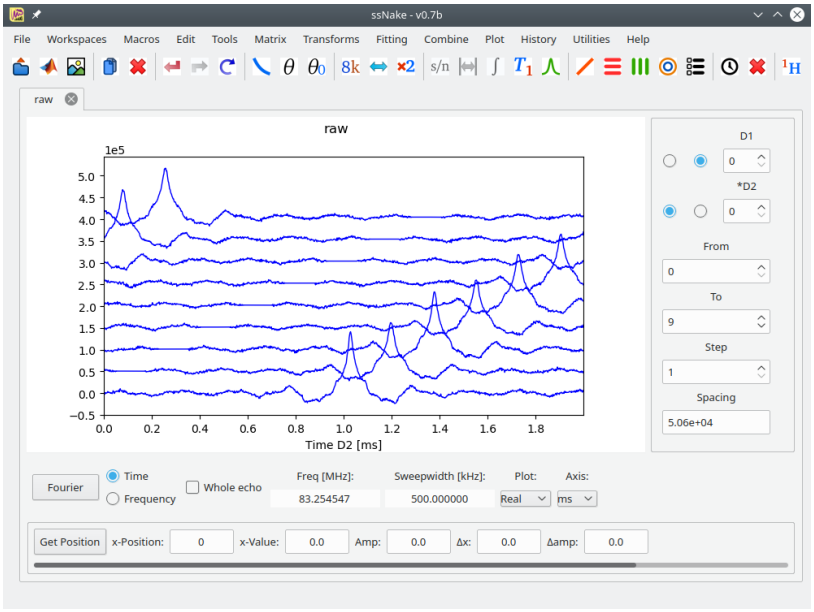
Now we have all the separate echoes. Viewing D2 shows:



which is the first echo. Scrolling through D1, we can see that for each echo, the position is the same: the splitting has been performed correctly. A stack plot of the first 10 echoes for example looks like this:



If we had used 1000 points per echo we would see:



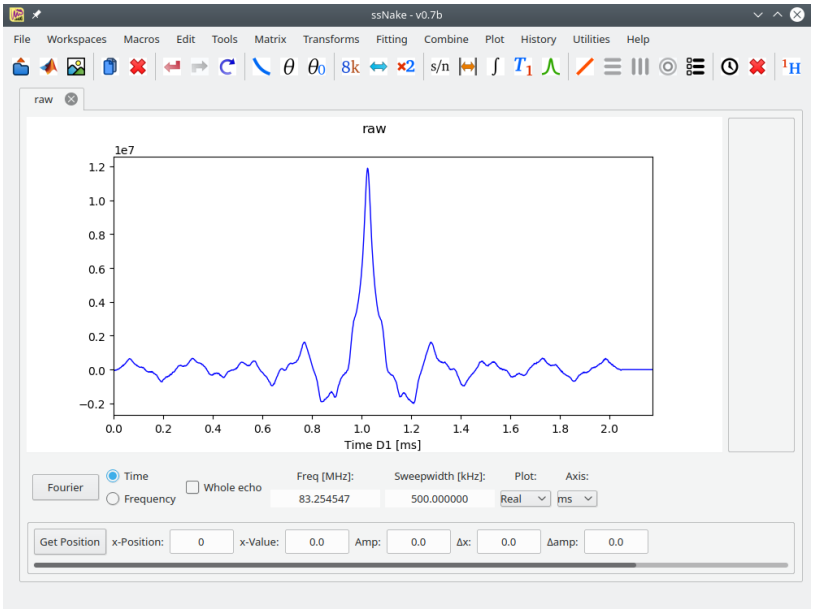
which is clearly very, very, wrong. A good check is to overlay the first and last echo. The 'zero' part at the end should be at the same position.

3.1.1 Option 1: Directly summing the echoes

Using the properly split data we can sum the echoes:

- Go to D1 (using the radio button in the side frame)
- Sum along this dimension using: Matrix ⟶ Region ⟶ Sum (with no input, just push Ok).

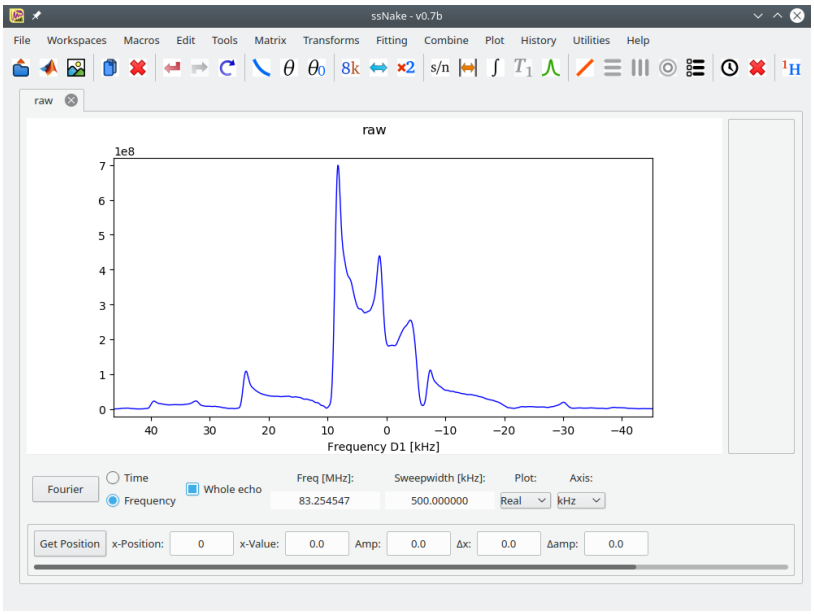
This results in:



We will now process the data using whole echo processing. More on this subject can be found in the Whole Echo processing tutorial. In short:

- Swap the echo using Tools ⟶ Swap echo at point 512
- Zero fill to 8192 points using Matrix ⟶ Sizing
- Fourier transform with the Fourier button in the bottom frame
- Phase a bit using Tools ⟶ Phasing (0th order: 0.220, 1st order: 68)
- 800 Hz Gaussian apodization (Tools ⟶ Apodize)

This should result in (zoomed):

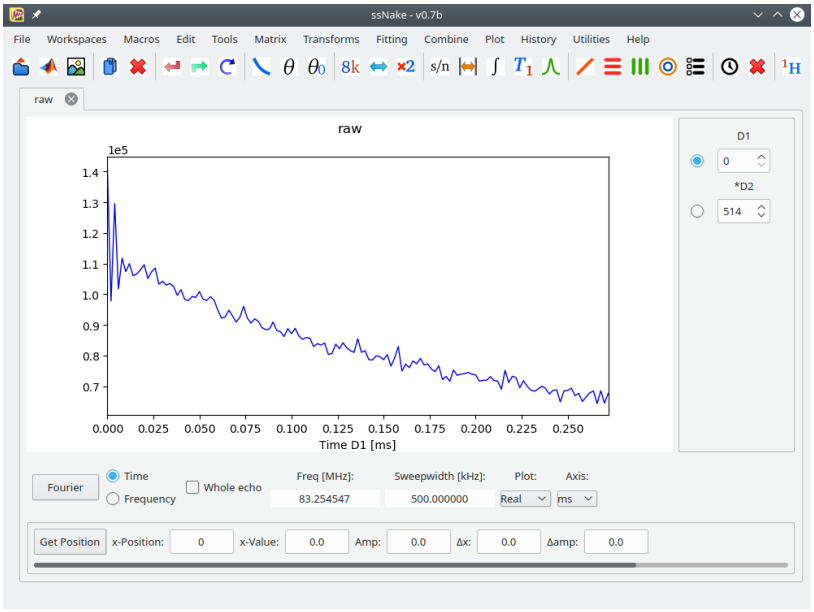


Which is the final spectrum using this processing method. It is saved as `EchoSumOption2.mat` in the data directory.

3.1.2 Option 2: T2 weighting

Summing all the echoes as has been done in Option 1 is fine in some cases, but could lead to a suboptimal signal-to-noise ratio. Imagine for instance that the signal would have almost completely decayed at the last echo. Clearly, including this echo in the sum will only increase the noise. So we would like to add the echoes with a weighting factor which equals the amount of signal present in the echo. As the echo intensity is reducing due to T_2 relaxation, this scaling method is referred to as 'T_2 weighting'.

To perform T_2 weighting, we should first obtain the T_2 value. We start with the data which we have just split in 137 parts (before the Option 1 section processing). The echo maxima are in this case located at data point 514. Put this number in the D2 box and press enter. This should result in a curve like this:

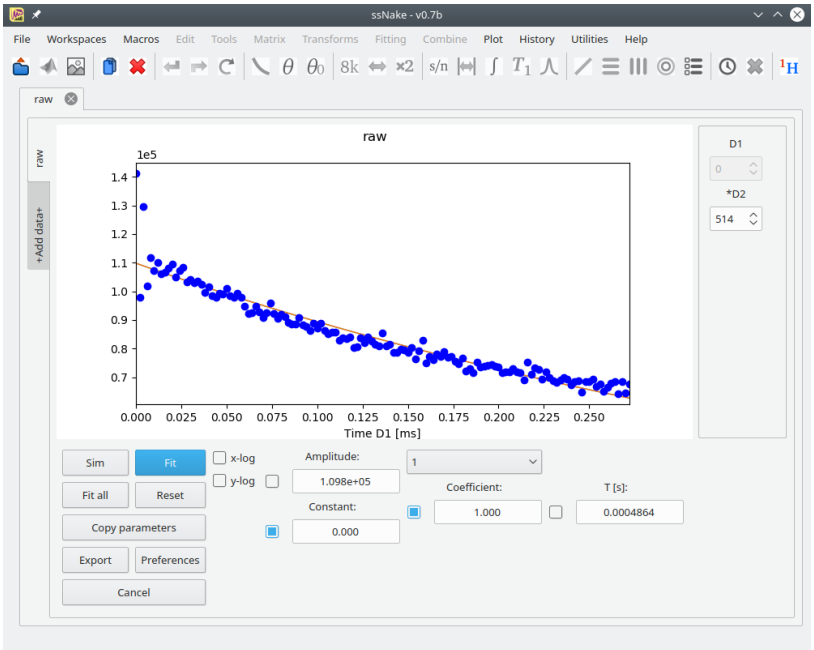


We now want to fit this line with a T_2 decay curve. Do note that the spectral width in this dimension is not sensible (as it was copied from the other dimension when we split the data). This means that our fitted T_2 will be in arbitrary units, but as long as we apply the weighting in the same units, we should be fine[1]. Let's fit a $T2$ curve:

- Open the relaxation fitting window (Fitting ⟶ Relaxation Curve)
- Set the 'Constant' to 0, and the 'Coefficient' to 1
- Set the 'T' variable to 0.001 as an initial guess
- Tick the boxes next to the 'Constant' and 'Coefficient' (this fixes them, so that they are not optimized during the fitting)
- Click 'Fit' (and perhaps a second time to see if the result changes significantly)

This results in:

[1] Alternatively, we could define the spectral width as the inverse of the time between the echo tops.



with a T_2 of 0.0004864 seconds. Close the Window using the 'Cancel' button

We now would like to use this T_2 curve to scale our echos. We will do this using Lorentzian apodization, which has the same function as a T_2 decay (exponential decay). The value that we will use is: LB $= 1/(T_2 \cdot \pi) = 654$ Hz.

- Use Tools $\longrightarrow$ Apodize and apply 654 Hz Lorentzian apodization along D1

The data can now be further processed using the methods of Option 1 above. This T_2 weighted spectrum has a better signal-to-noise ratio that that of Option 1, although in this particular example the data has only high-intensity echoes, so the difference will be small. The final spectrum is saved in the data directory as `EchoSumOption2.mat`.

3.2 Spikelet spectrum

A wholly different method for processing the data from a QCPMG experiment is the spikelet method. It features a direct Fourier transform of the FID, with no splitting and summing of the echoes. The resulting spectrum has a series of spikes (i.e. spikelets) in the spectrum, with a distance of $1/T$, with T the time between two echoes. The advantage of the spikelet method is that all the signal is concentrated in the spikes, leading to a huge increase in signal-to-noise ratio. The disadvantage is that, while the maxima of the spikes follow the intensity distribution of the 'regular' echo spectrum, the area between the spikes has no information. If only a couple of spikelets are present, the shape of the quadrupolar line becomes obscured.

- Open the data in the `Raw` folder
- Zero fill to 524288 (Matrix $\longrightarrow$ Sizing)

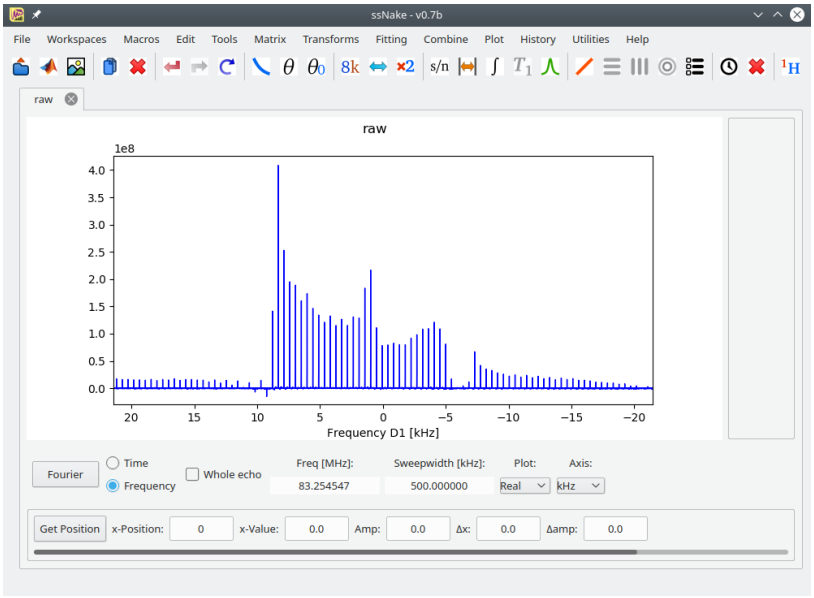
We have to apply a first order phase correction, to make sure the FID starts at the first echo maximum. The position of this top is data point 514. To correct this we require $\theta = 360 \cdot n = 185040$ of first order phasing:

- Phase with 185040 first order phasing (Tools ⟶ Phasing)

And to obtain a decent spectrum:

- Fourier transform (using the 'Fourier' button)
- Gaussian apodization of 4 Hz
- Phase (0th order: -104, 1st order: -618)

The final spectrum should then look like:



which is saved as `spikelets.mat` in the data folder.

When we performed the Gaussian apodization, we also removed the tail of the first echo (which was shifted to the end of the FID by the large first order phase correction). This suppresses the formation of a regular echo line shape under our QCPMG spikelets at the cost of intensity. In this case, we have many intense echoes, and the added baseline of this tail does not help us. If we do wish to retain this, we must make sure that the apodization is performed *before* the large first order phase shift.