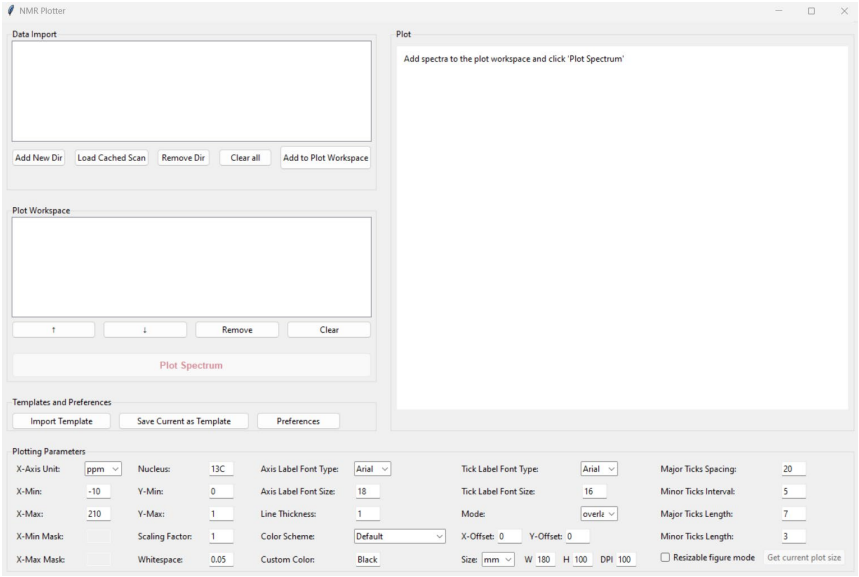
Tutorial and examples (2025/08/19)

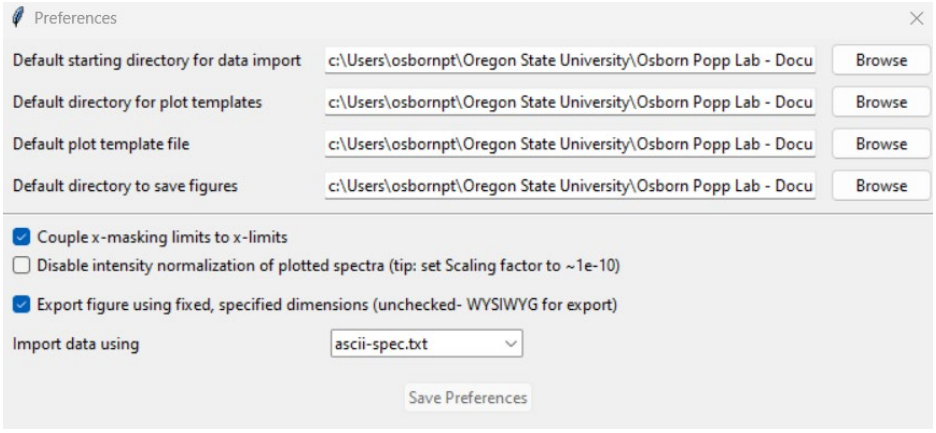
Before you begin: setting preferences

On first startup, you should see a GUI window that looks like the image below (see *GUI Functions* section below for detail about each frame in the GUI). Before importing any data though, it's highly recommended that you set your preferences for default data import directories, plot templates, and figure exports.



Click the "Preferences" button in the Templates and Preferences Frame:

A second window will pop up that looks like this:



If this is your first time running the program, you likely do not have a “preferences.txt” file in your current working directory. As a result, the default paths were set based on wherever your NMR_Plotter folder is located. The default directory for each item is initially set as:

- *Default starting directory for data import:* *\NMR_Plotter (the folder that NMR_Plotter.py is contained within)
    - **For the tutorial- set it to *\NMR_Plotter\exampledata_directories**
- *Default directory for plot templates:* *\NMR_Plotter\plot_templates
    - **Keep it as this directory for the tutorial**
- *Default plot template file:* *\NMR_Plotter\plot_templates\default.txt
    - **Keep it as this directory for the tutorial**
- *Default directory to save figures:* *\NMR_Plotter\figures
    - **Keep it as this directory for the tutorial**

Clicking “Browse” will open a file browser where you can then select a different folder/file as default. For example, if you store all your top-level directories containing NMR data in a particular folder, it may make sense to set that as your default data import directory. Or, if you’re working on figures for a manuscript, you may want to set the default save figure directory to a folder on your machine associated with your manuscript files. The plot templates directory/default template file may not need to be changed immediately, but for example, if your group has a collection of plotting templates stored in a centralized location, you could set it to that folder. If you know you will always be plotting figures with a particular set of settings, you can change the default to a different template file, or you could just save a new set of parameters to the default.txt file.

The three check-box preference options are:

- *Couple x-masking limits to x-limits:* This option prevents data from being drawn outside of the plot window, so that when processing the exported plot files in a vector graphics program, the plot trace is not restricted by a clipping mask, but instead exactly matches the width of the plot frame. It is checked by default, **and it is recommended for most typical plotting cases that you leave this checked**. One of the examples below gives a scenario where it is useful to have this option unchecked **(but if you are following along the tutorial, please leave it checked to start)**
- *Disable intensity normalization of plotted spectra:* When unchecked, the program normalizes all spectra to their maximum value, setting this maximum to 1. This is useful for direct comparisons of spectra where intensity/signal to noise is not important, as it produces consistent plot ranges for the y-axis between different spectra. If, however, intensity values or signal to noise comparisons are important for your figure, then go ahead and check this box, which will mean that the program will no longer apply any normalization, and just use the raw intensity values from TopSpin. These can be very large numbers though, so it’s recommended you set the scaling factor in “Plot Parameters” to around 1e-10 so that you can work with reasonable ymin/ymax range values **(for the tutorial, leave this option unchecked).**
- *Export figure using fixed, specified dimensions (unchecked – WYSIWYG for export)*- Left checked by default- this means that figures will export according to fixed, specified dimensions in the plot parameters. When unchecked, this option allows you to export spectra according to how they exactly look in the Plot canvas view (what you see is what you get- WYSIWYG). **It is generally recommended that you leave this checked** unless

you prefer to have sizing changes made using “resizable mode” or by changing the GUI dimensions/scaling reflected in your figure export **(for the tutorial, leave this option checked).**

The final preference option is to import data using either ascii-spec.txt files, generated using the *convbin2asc* AU program in TopSpin, or to use pdata (processed data binary files, like ‘1r’ files) as the data import choice. Note that with the pdata option, you will need to install the nmrglue package. Note that in the example data included in the repository, both ascii-spec.txt and pdata files are provided, so you don’t need to process anything in TopSpin to test the program’s functionality. **Either option is fine for the tutorial.**

After changing any preferences, the “Save Preferences” button on the bottom of the window will become active, and **you should click it to save your changes.**

Tutorial 0: Adding data directories

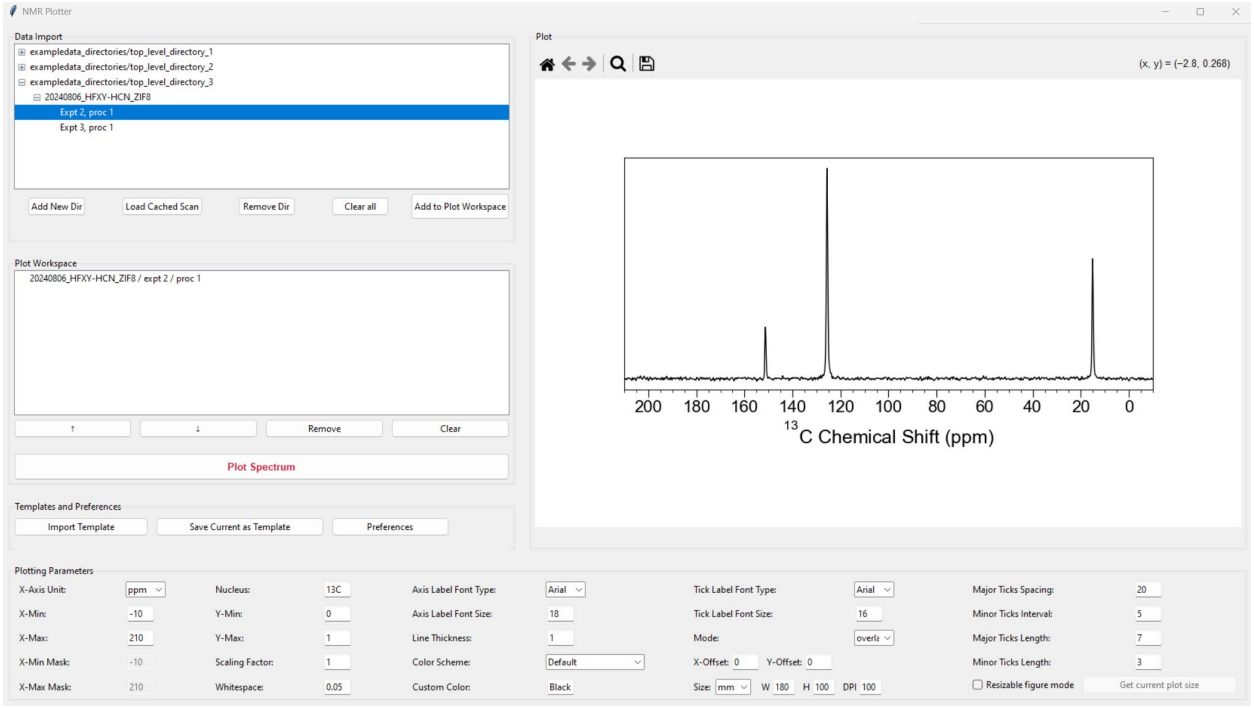
Once your preferences for data import are set, click “Add New Dir” in the Data Import Frame. From the example data directories, select “top_level_directory_1”, then it should appear in the Data Import window with a “+” button next to it. Also add “top_level_directory_2” and “top_level_directory_3” to the Data Import window. Clicking the “+” button on each will reveal the sample folders contained within each directory. Clicking the “+” button on each sample folder will reveal the processed data below that, labeled by experiment number and proc number (Bruker TopSpin convention). You can scroll using the mouse scroll wheel to navigate.

A directory scan is “cached” the first time its imported. To show what this means, click “Clear all” to clear the window, then click “Load Cached Scan”. This will quickly reload all the directories you just previously added. The “Remove Dir” just removes the directory from the window, but not the cache. A cached scan of a directory does not show any recent changes that may have occurred after the cache was created, for example if an ascii-spec.txt file was generated in a new folder. To refresh a directory cache so it is up to date, simply “Add New Dir” again on the directory you’d like to update. Separate caches are saved for both ascii-spec.txt data import and for pdata import. To clear a cache, simply delete the cache_ascii.txt or cache_pdata.txt file that lives in */NMR_Plotter.

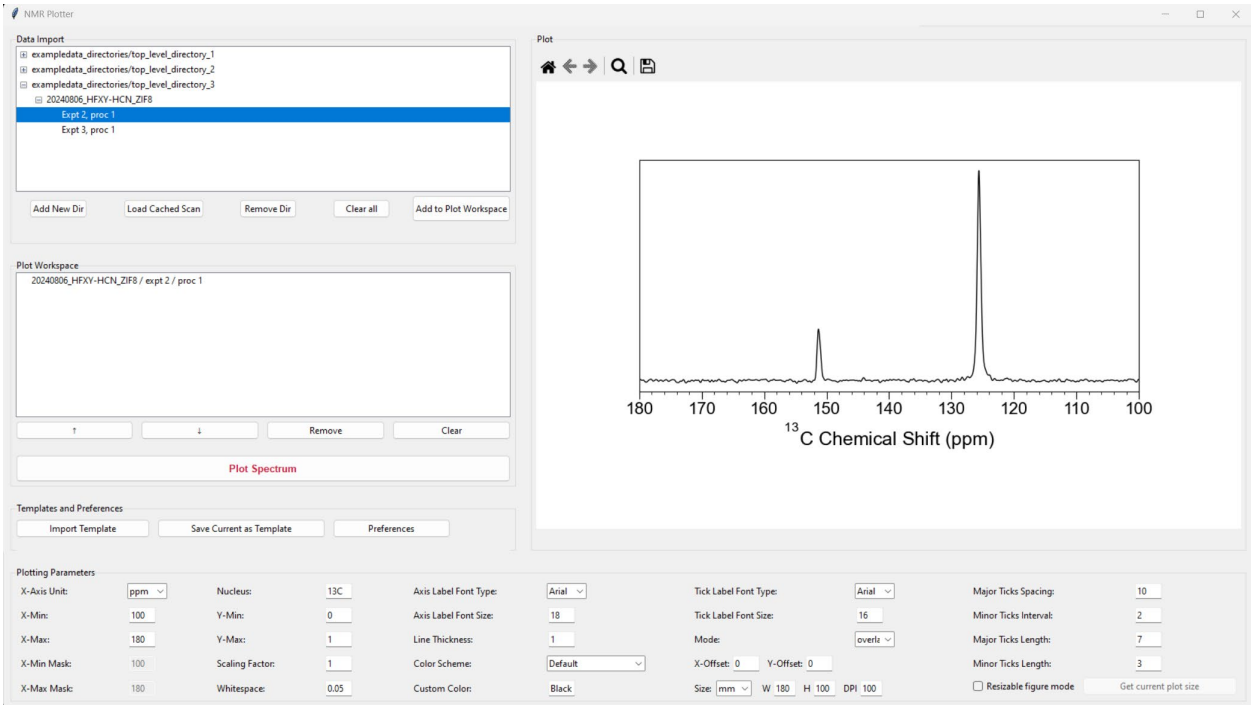
Now that you’ve set your preferences and added your directories, let’s proceed with the tutorial examples.

Tutorial 1: Plotting using the default template, and generating a new template

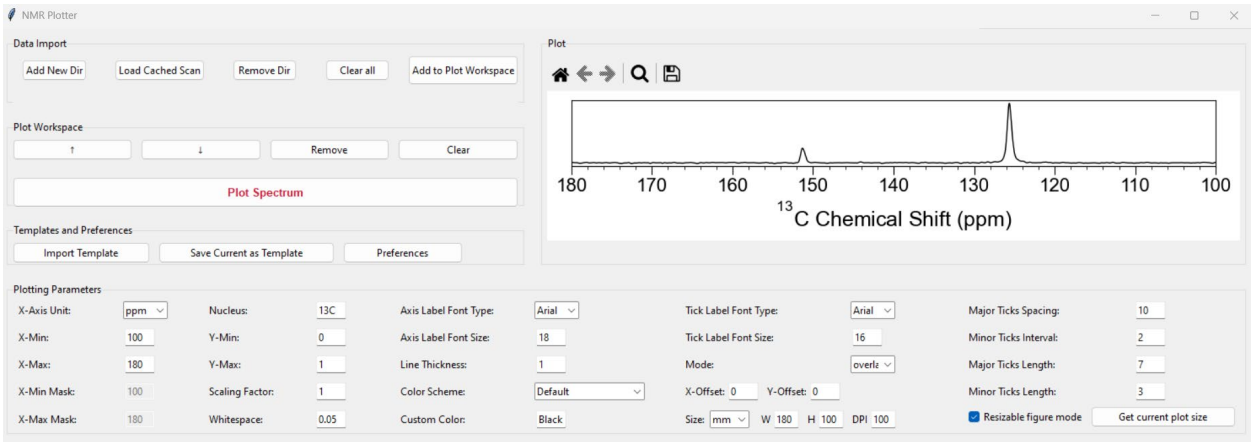
Open top_level_directory_3 and the sample folder contained within it, then select “Expt 2, proc 1”. Once highlighted, select “Add to Plot Workspace”. Once it’s in the plot workspace, click the red “Plot Spectrum” button. This will apply the default template, which was loaded in on startup. Your window should look something like this:



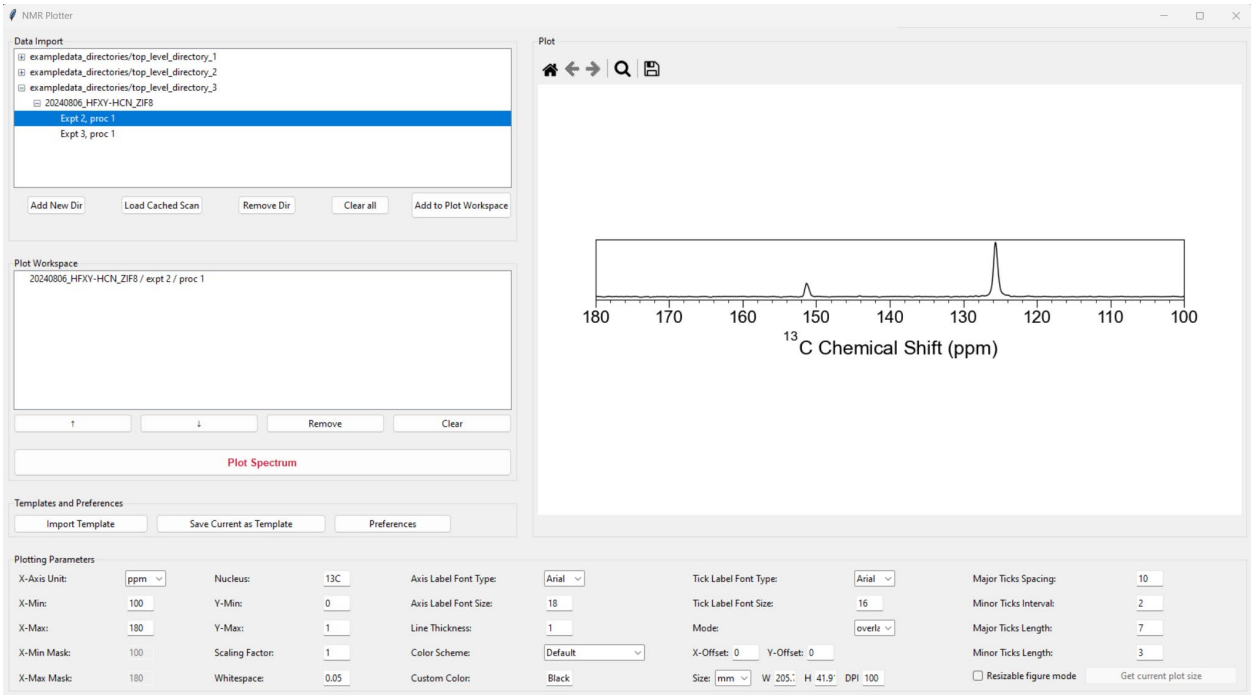
To modify plotting parameters, simply change any of the values, then click “Plot Spectrum” to see how it looks. For example, change xmin to 100, xmax to 180, major ticks spacing to 10, minor ticks interval to 2:



To adjust the sizing, you can manually change the width (W), height (H) and DPI values. You can also click the checkbox “Resizable figure mode” which will have the figure bounds snap to the edges of the Plot canvas. Resizing the window will change the dimensions of the figure as well- for an extreme example, see below:



Let's say you like this aspect ratio for your figure- to get these dimensions so you can reproduce this sizing consistently, click the "Get current plot size button," which will populate the W, H, fields with the current plot size. In this case, it captured it at 205.7 x 41.97 mm. To implement these new dimensions directly, just uncheck "Resizable figure mode". Now, resizing the window leaves the figure according to those captured dimensions:



Let's try to formalize this into a reasonable plot parameters template- so let's round to W:200, H:40, then click Plot Spectrum. Now, to save these parameters as a template, click the "Save Current as Template" button in the Templates and Preferences frame. This should open up your plot_templates folder, where you can then save this template as whatever name you'd like to give it. In this case, name it something like 13C_100-180_wide.txt (you can leave the .txt on or off, it will always save as .txt file).

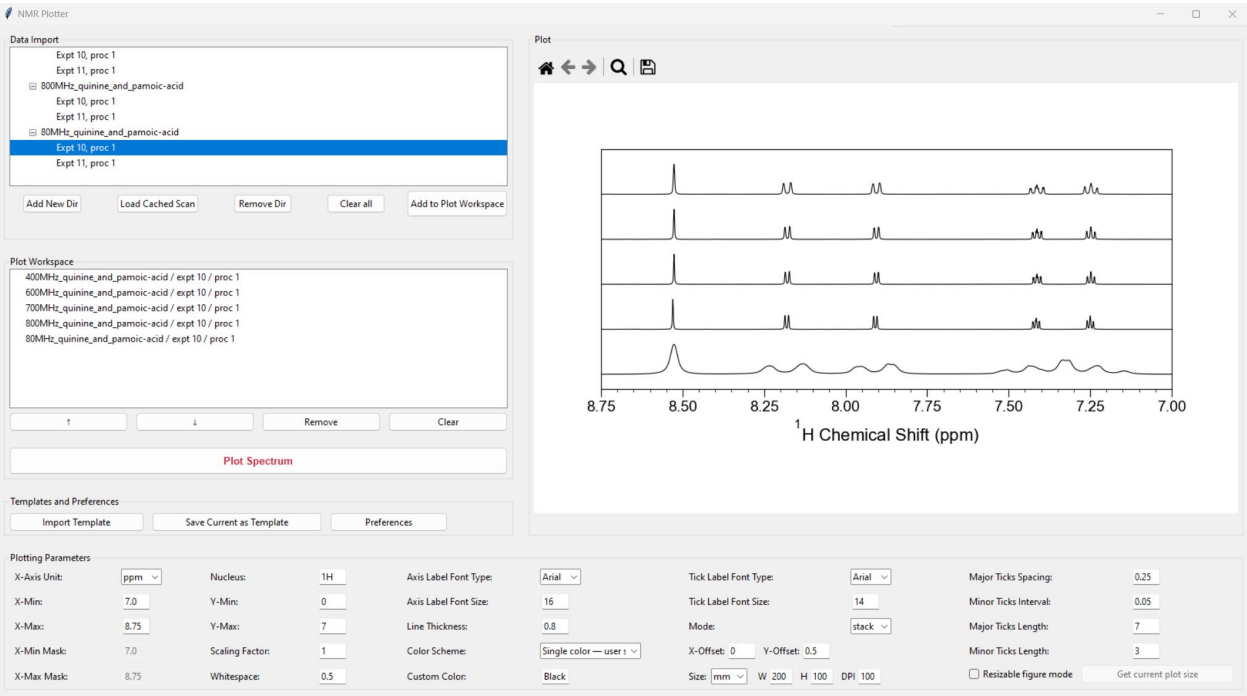
After saving, you can test re-applying it by clicking "Import Template" then selecting default.txt, then pressing "Plot Spectrum," which will return the plot to the default template. Reload your wide template file by going back to the import menu and selecting it, then Plot Spectrum.

To export your figure, click the “Save” floppy disk Icon above the Plot canvas, where you can now save it as a PDF, SVG, or PNG file.

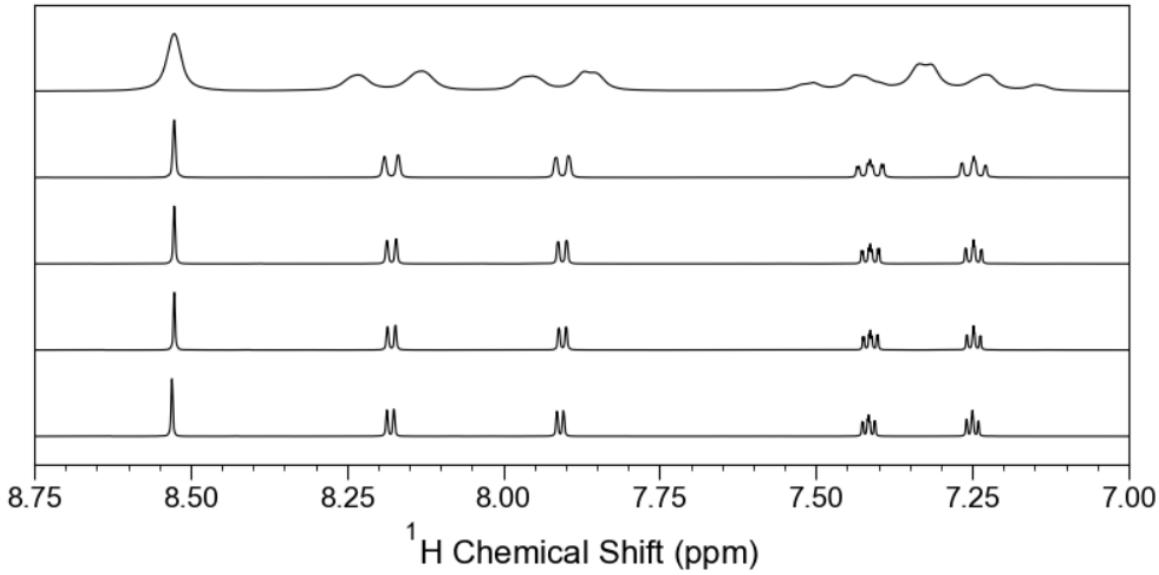
Note, while we’re at the plot toolbar- most of the usual Matplotlib icons have been removed, except for zoom. Zoom is really just meant to help you find x-axis and y-axis ranges that work for you, so that you can zoom to a target area and then hover the mouse over coordinates to get the coordinate values. The view showed in a zoom does not reflect in the final figure export, as the export pulls from the plotting parameters. However, if the WYSIWYG option is chosen in preferences, then you can export a zoomed view.

Tutorial 2: Plotting stackplots using the “stack” vs. “overlay” mode

Clear your Plot Workspace. Navigate to “top_level_directory_1” and expand all folders within it. You should see that each one contains “Expt 10, proc 1” and “Expt 11, proc 1”. Select Expt 10, proc 1 from each of the folders and add them to the Plot Workspace. Ctrl+clicking on Windows and cmd+clicking on Mac will allow you to select multiple datasets at a time to add simultaneously. Import the template file called: “1H_tutorial2_stackplot.txt”, then click “Plot Spectrum.” Your plot should look like this:

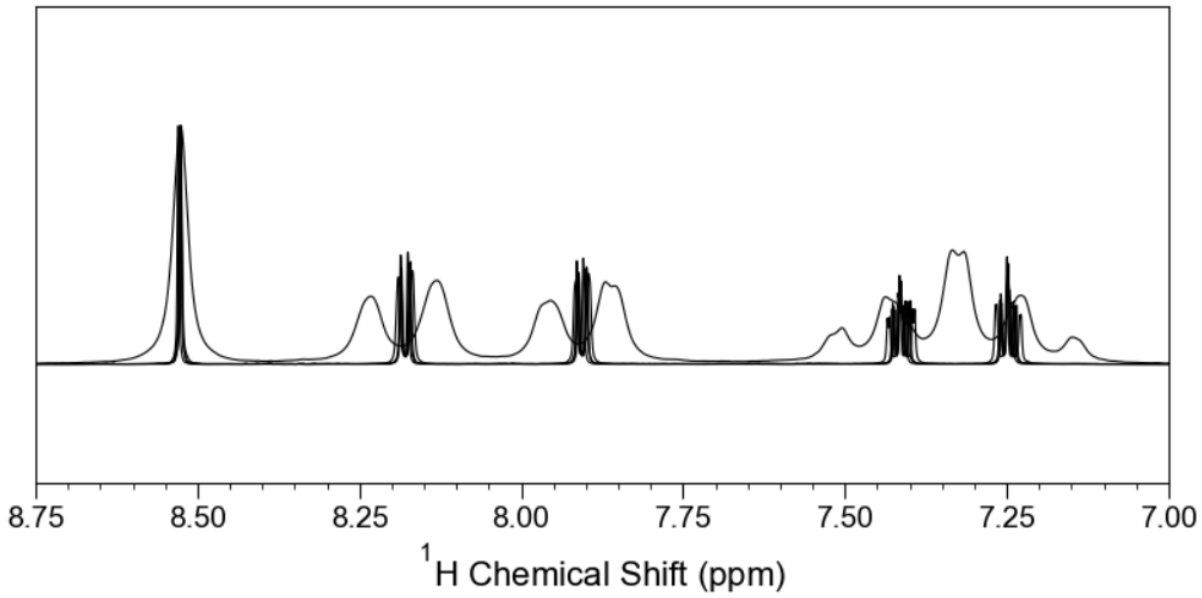


However, because this data shows a trend—the dependence of the spectrum on the strength of the magnetic field—we want to order in a logical way, either from highest to lowest field or vice versa. Right now, the lowest field (given as the Larmor frequency of 80 MHz) is at the bottom, but the rest are in order. To fix this, you can select a spectrum in the Plot Workspace and then use the arrow buttons to move it up and down to reorder the data. Reordering the data to put the 80 MHz spectrum on top yields the following figure:

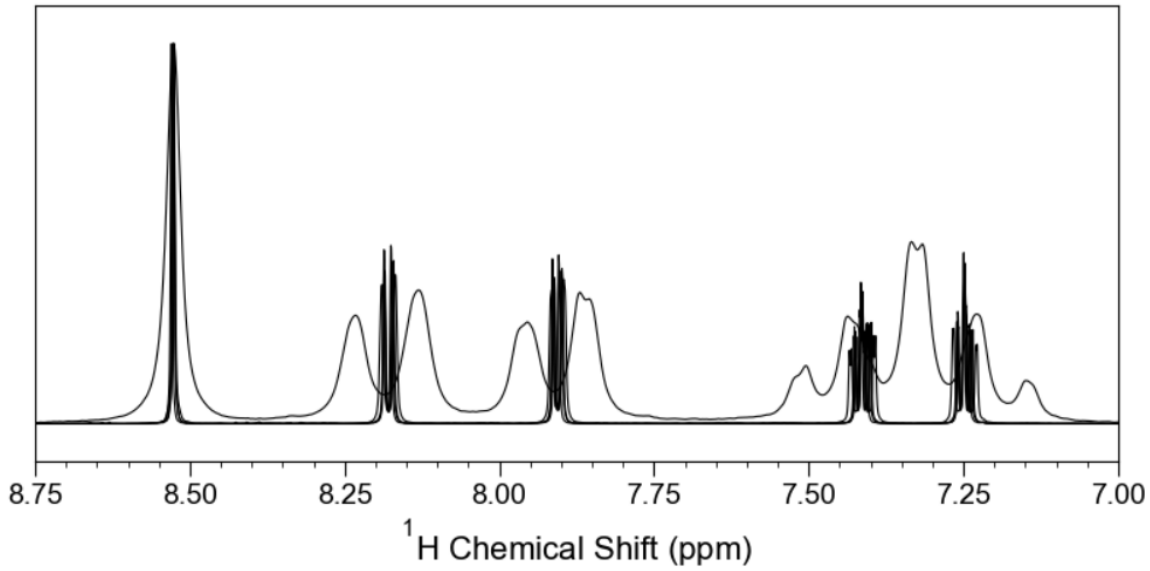


Note that the order in the Plot Workspace corresponds to the order in the plot- i.e. if the 80 MHz spectrum is at the top of the list in the workspace, it is at the top of the stackplot in the figure.

To illustrate the difference between the “Stack” mode and the “Overlay” mode, while you have the data loaded in the Plot Workspace, set your y-max to 1, Y-offset to 0 and change the mode from “Stack” to “Overlay.” Now, your plot should look like this:

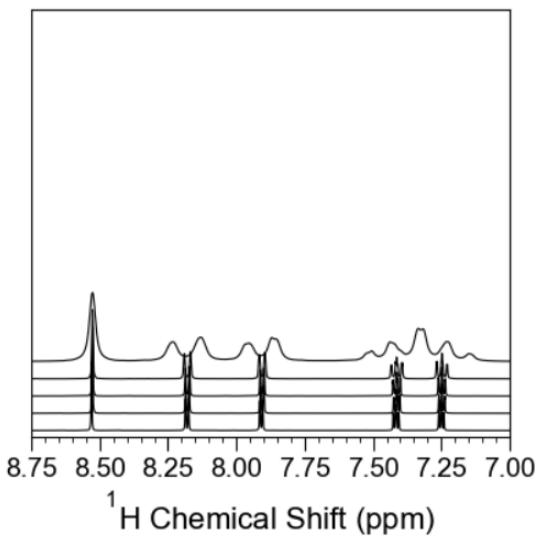


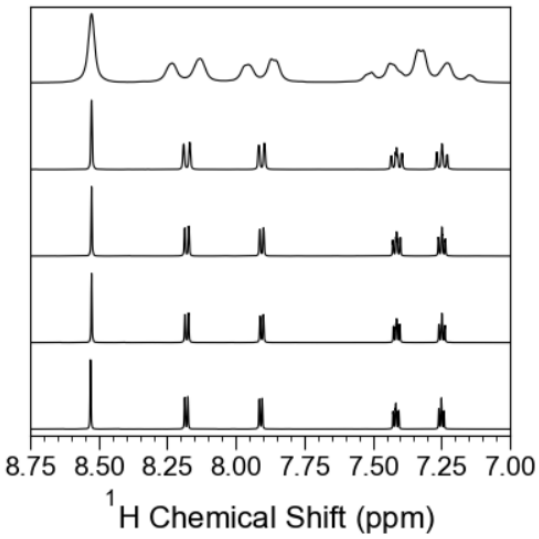
Now, all the spectra are overlaid on top of one another. Note that the “whitespace” parameter is still set to 0.5 here, which is why there is a lot of vertical space above and below the data. Set this to 0.1:



The whitespace guarantees that there is some breathing room between your data and the plot frame, so that you can think in terms of data limits, rather than in terms of figure limits (i.e., no need to set your ymin to -0.1 and ymax to 1.1 to get that extra space—setting y-limits to 0 and 1, with a whitespace of 0.1, guarantees you will fit the full height spectrum in your figure).

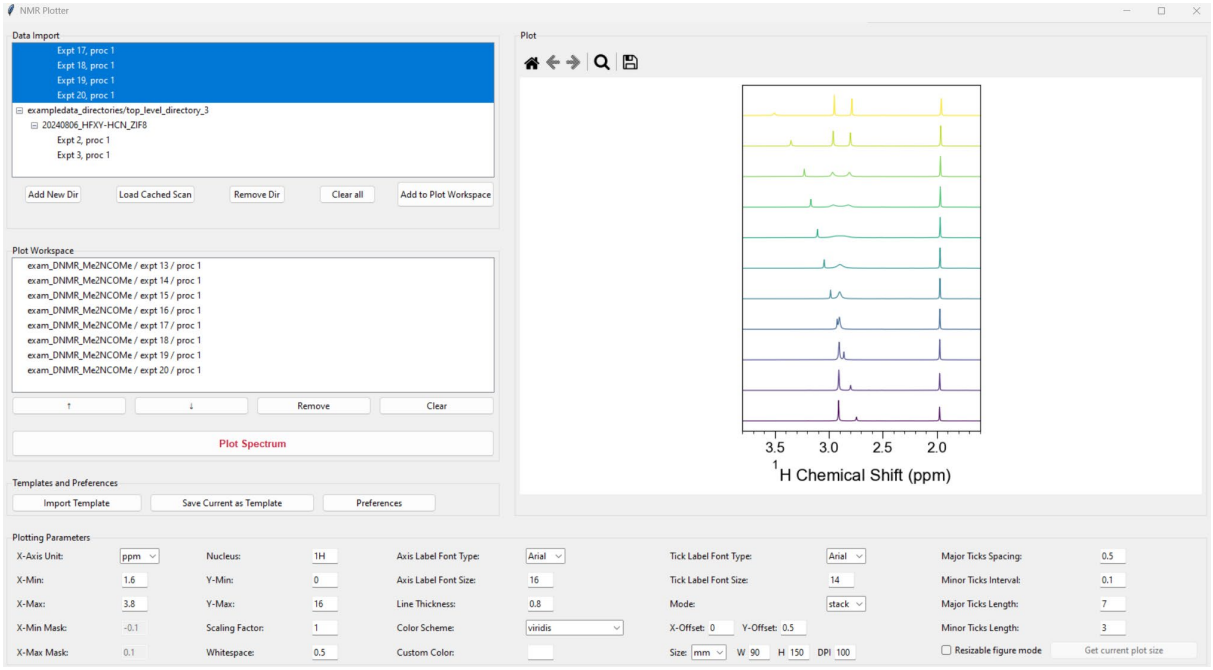
While still in overlay mode, set your y-max to 5, then gradually increase your y-offset value, from 0.25, to 0.5, to 1. Setting it higher than 1 will send some of your spectra off the viewable area, requiring you to increase your y-max limit. Overlay-based stackplots take the y-offset from a common origin, while “stack” based stack plots will use the y-offset as a spacing between the top and bottom of each successive spectrum. For example, on the left is an overlay stackplot with a y-offset of 0.25, y-max of 6, and a whitespace of 0.1, and on the right is a stack stackplot with the same parameters:





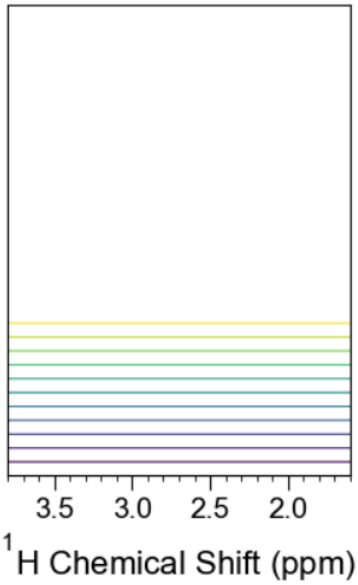
Tutorial 3: Plotting stackplots in color, without normalization, and as diagonal plots

Clear your Plot Workspace, and then expand the “top_level_directory_2” folder and the subfolder beneath it. Click the first entry, Expt 1, Proc 1, then shift click Expt 20, proc 1 to select all of the data in this folder and add to the Plot Workspace. Import the “1H_tutorial3_stackplot.txt” template, then click “Plot Spectrum,” yielding a colorful stackplot:

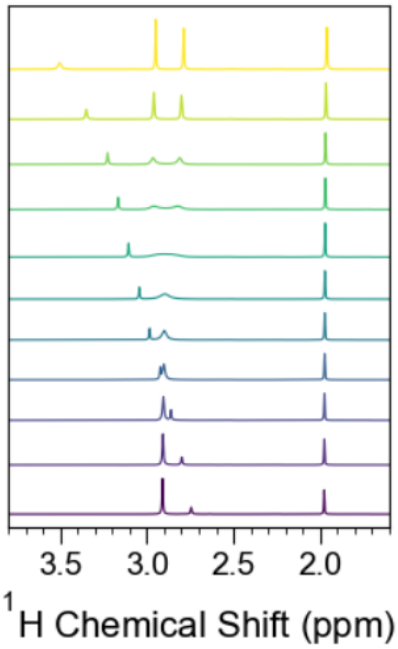


This is just an example that shows that you can use different color schemes to visualize your data, in this case, the “viridis” color palette.

However, this is a series of spectra taken at different temperatures on the same sample, so intensity changes are significant. The tallest peak in the spectrum changes from one spectrum to the next, which means normalizing to the max intensity makes it seem like some peaks are growing and shrinking in a way that isn’t true to the dataset. To represent the data in a quantitative way, head to the preferences window and check “Disable intensity normalization of plotted spectra.” Before plotting the spectrum, it is suggested that you set the scaling factor to a small number so that you can work within reasonable y-limits, whitespace values, and y-offsets. The tip in the preferences window suggests 1e-10, but this number will vary between different datasets depending on the exact intensities- you’ll usually need to play with this to get it right. For example, in this dataset, changing none of the plotting parameters except for the scaling factor to 1e-10 results in the following output:



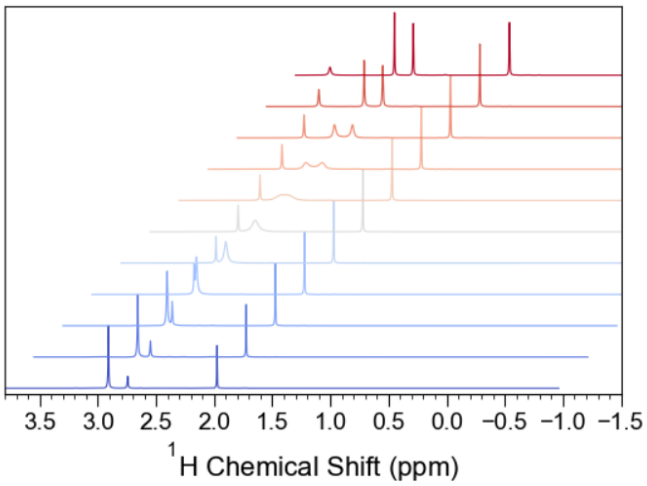
Seems like we'll need to make the scaling factor slightly less extreme—setting it to 1e-7 with a y-max of 18 gives:



Although visually similar to the normalized stackplot, this version preserves quantitative data about the peak height relations between spectra, making it clear that the peak at ~2.0 ppm decreases in intensity as you go from yellow to purple, and that the peaks are more intense overall in the yellow plot compared to the purple plot.

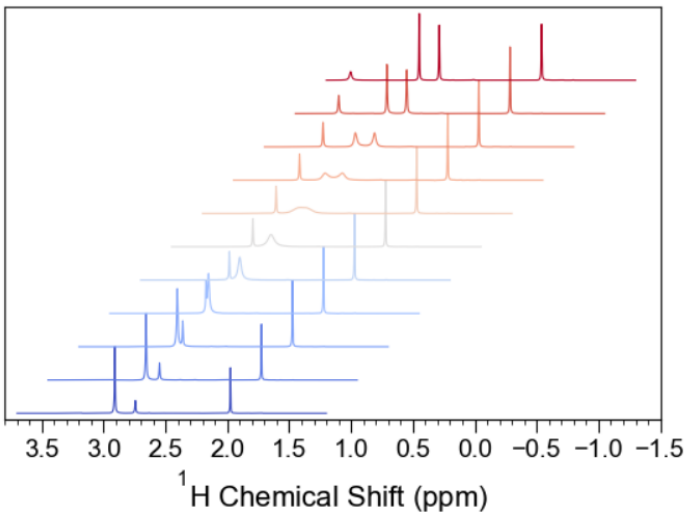
Make sure to uncheck "Disable intensity normalization" in preferences before proceeding with the next part of the tutorial.

To expand on the concept of overlay vs. stack mode introduced in Tutorial 2, now select “1H_tutorial3_diagonal.txt” and replot. Your figure should look something like this:

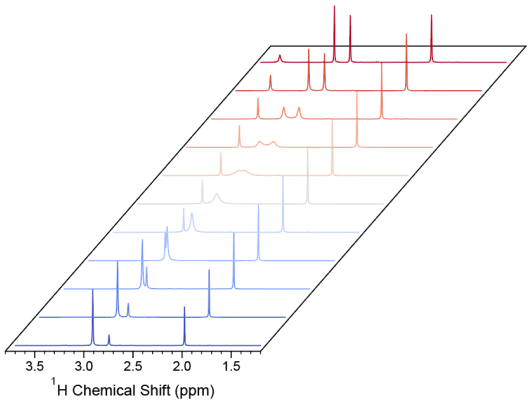


This is an overlay plot with both a y-offset and an x-offset, creating a diagonal-style, pseudo-3D plot. This kind of plot is only possible in overlay mode. Note though that the only spectrum with the correct correspondence between the x-axis and the data is the bottom blue spectrum- the rest have all been offset by -0.25 ppm. Because of this, to show the top plot without it going out of frame, the x-min had to be set to -1.5 ppm. However, this also results in a weird effect where all the other plots show data as far out as -1.5 ppm as well. We can fix this by decoupling the x-masking limits from the plotting limits.

In preferences, uncheck “Couple x-masking limits to x-limits” and save preferences. You’ll now see that the x-min mask and x-max mask are editable parameters where they previously were not. Set your x-min mask max to 3.7, and your min to 1.2 to get the following:



In this state, it’s ready for export as a vector graphic to be converted quickly into a more presentable format, such as the figure below which can be made in just a minute or two in Adobe Illustrator or another vector graphics program, as the bulk of the work involving spacing and alignment has already been done:



Whether you use the exports directly, or if you use them as inputs to a vector graphics program for further processing, the key is that NMR_Plotter can save time in a figure-making workflow, giving the user a lot of flexibility in this process, without requiring coding/programming knowledge.