

2020 HEXT Workshop

Tutorial in Data Processing at the Beamline

Chris Pollock
2020.06.10

Processing Data at the Beamline

Software Needed

PyMCA (<http://pymca.sourceforge.net/>)

Part 1: Processing and Assessing XANES Data using PyMCA

- 1) PyMCA is a program commonly used to process x-ray spectroscopy data that is found at many beamlines around the world, including here at CHESS. We'll be using PyMCA to process all of the data in this tutorial, so go ahead and open it up. It may take a few seconds, especially if this is your first time using the program, but when it does you should see a window like Figure 1-1.

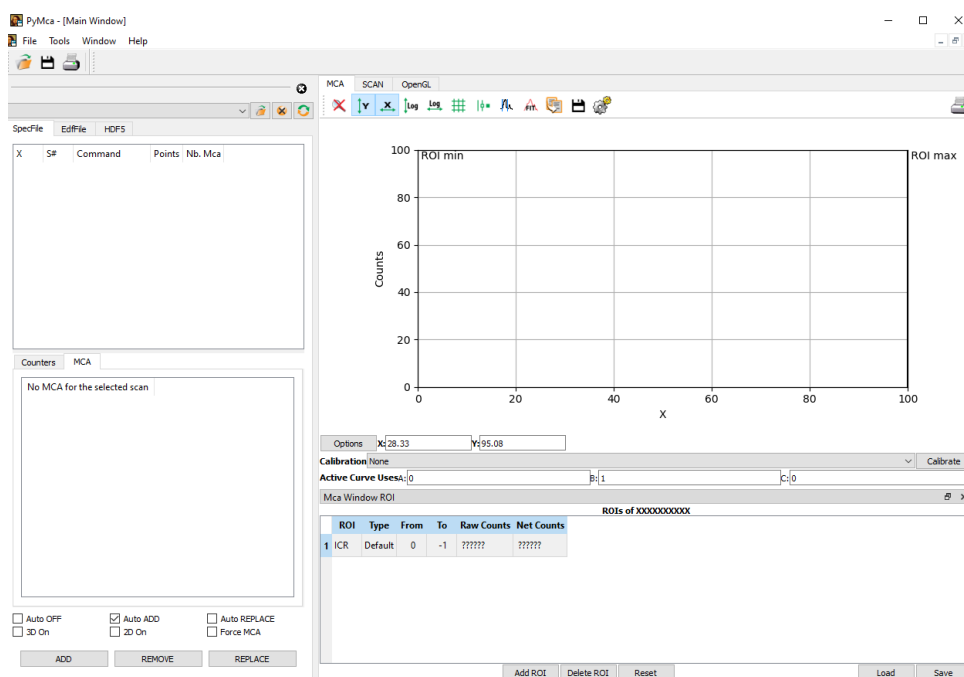


Figure 1-1: The PyMCA window should look like this.

- 2) Open up the data file named "tutorialData" that is available on Box by going to File > Open > Data Source. Since SPEC files do not have extensions, in order to see this file, you will need to set the "File type" filter to "All files (*)" in order to see the file. If all worked well, the upper left panel of your PyMCA window should appear as in Figure 1-2.

X	S#	Command	Points	Nb. Mca
	1.1	chess_escan 1.8...	553	0
	2.1	chess_escan 1.8...	553	0
	3.1	chess_escan 1.8...	553	0
	4.1	a2scan xesdan...	201	0
	5.1	a2scan xesdan...	201	0
	6.1	a2scan xesdan...	201	0
	7.1	a2scan xesdan...	156	0
	8.1	a2scan xesdan...	156	0
	9.1	a2scan xesdan...	156	0

Figure 1-2: The upper left panel of the PyMCA window after opening the tutorialData file.

- 3) This file contains several different spectral scans, including both XAS and XES, which we'll go through sequentially during this tutorial. With the file open, go ahead and click on the first line shown in Figure 1-2 (the one beginning with "1.1"). PyMCA should update to plot something in panel E (don't worry if your plot doesn't match Figure 1-3). Before we go any farther, let us first explore the PyMCA window a little bit. Using the labels from Figure 1-3, the following are useful locations within PyMCA:

- A. This panel contains a list of all scans contained within the data file.
- B. This panel displays all of the channels containing data for a given scan.
- C. Here are buttons to control whether spectra are added or replaced in the current view.
- D. This toolbar contains tools to manipulate the spectra.
- E. This panel displays the active spectrum / spectra.

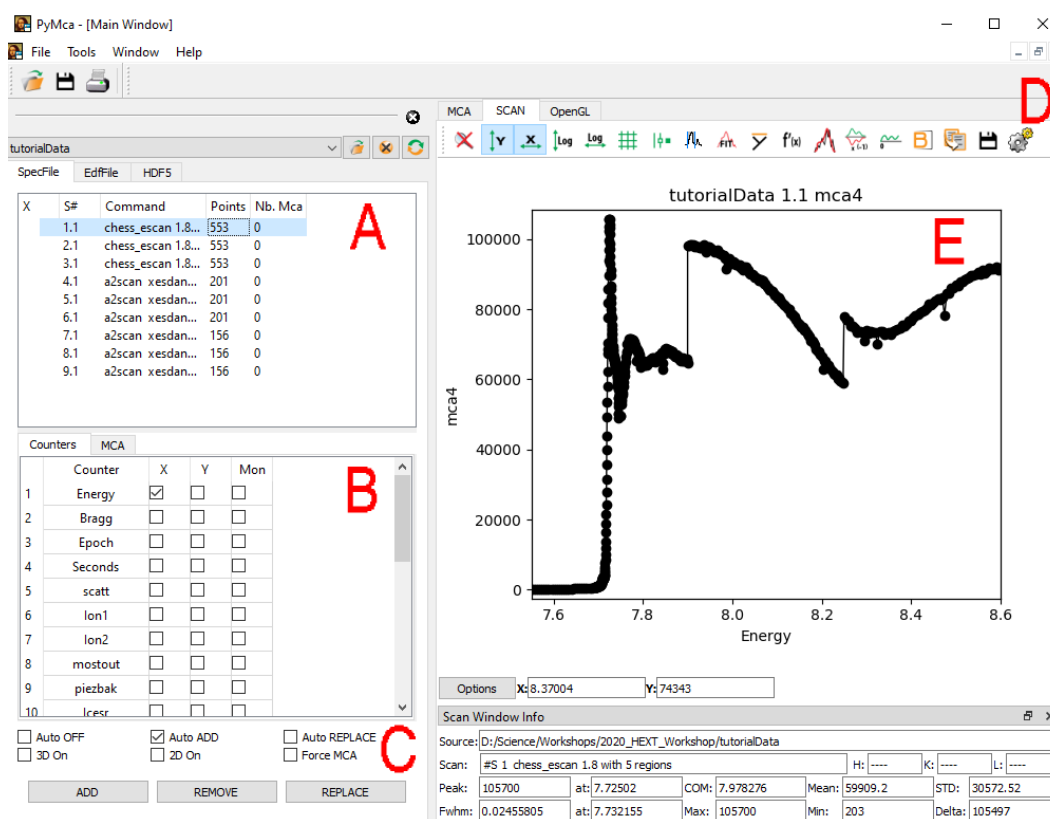


Figure 1-3: The most useful places of the PyMCA window are highlighted.

- 4) PyMCA allows for data to be quickly displayed and manipulated in a variety of ways that are useful when looking at x-ray spectra. Many of these useful tools and functions are easily accessible in panel D, shown in Figures 1-3 and 1-4. We'll be using several of these soon, so below are more detailed descriptions of what the various buttons do (Figure 1-5).



Figure 1-4: The PyMCA toolbar has many useful functions.



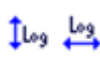










	This button zooms out of the current spectrum, adjusting the axis limits so that the entire spectrum can be viewed.
	These buttons control the x- and y-axis autoscaling; when clicked (as they are in Figure 1-3), the axes will automatically adjust their limits to accommodate newly added spectra.
	These buttons plot the y- and x-axes (respectively) on a logarithmic scale.
	This button toggles the display of a background grid behind the displayed data.
	This button controls whether the data is plotted as a line only, data points only, or a line with data points. Clicking the button will cycle between these various options.
	This button averages all currently displayed data.
	This button takes the derivative of the active curve and plots it along with the data.
	This button smooths the active curve and can be clicked repeatedly to increase the amount of smoothing applied.
	This button multiplies the active curve by -1.
	This button sets the lowest y value of the active curve to 0; in many cases this can be used for rough background subtraction.
	This button will subtract the active curve from all other displayed data.
	This button is a shortcut to save the active curve.
	This button allows access to additional functions within PyMCA, such as deglitching and data normalization.

Figure 1-5: Detailed descriptions of the various PyMCA buttons are included for reference here.

- 5) Now that we have that out of the way, it is helpful to understand how SPEC records information in its data files. When you perform a typical x-ray spectroscopy experiment, you generally scan one or more motors at the beamline while recording signals from various detectors at each point along the scan. For every spectrum you collect, SPEC records the positions of the motors that move as well as the signals coming from all of the detectors at each and every point. Since each scan generally has a few hundred points and there can easily be >20 motors and detectors, these files can contain a lot of data! We'll explore things in more detail soon, but for now simply know that the scans are located in panel A while the motor positions and detector signals are found in panel B (referred to as "channels" in PyMCA).

- 6) Alright, turning to the data itself, the first few scans are XAS data collected in fluorescence mode using a Vortex solid state detector, which means the beamline was configured similar to the cartoon in figure 1-6. Typical for fluorescence experiments, the Vortex detector was placed near the sample and at 90° relative to the incident beam.

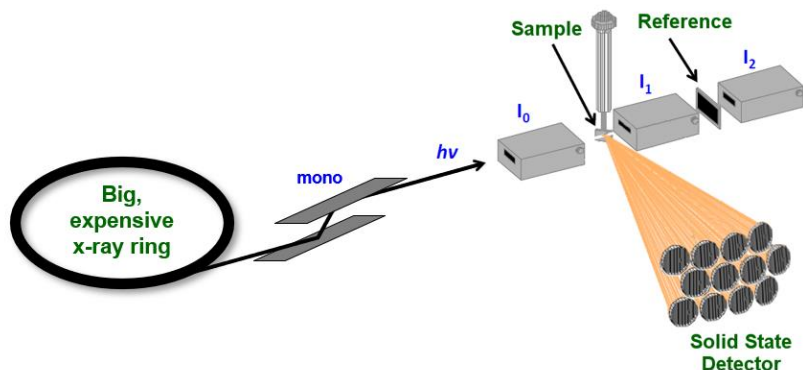


Figure 1-6: Schematic representation of the beamline setup for fluorescence XAS measurements.

- 7) Our Vortex detector has four separate elements, each of which can collect fluorescence signal. This means that the detector records four spectra each time we do an energy scan, quadrupling the signal we can collect (some beamlines have detectors with 30 or even 100 element detectors!). In PyMCA, these four detector elements have their data each saved to separate channels (**mca1**, **mca2**, **mca3**, and **mca4**) and we can plot each one individually. To do this, go into panel B of the PyMCA window and make sure that “Energy” box is checked in the “X” column, as it is in Figure 1-3. After that is done, scroll farther down in panel B until you find the rows labeled mca1, mca2, mca3, and mca4 (they’re at the bottom) and check each of those boxes in the “Y” column. When you’re done, the PyMCA window should look like Figure 1-7.

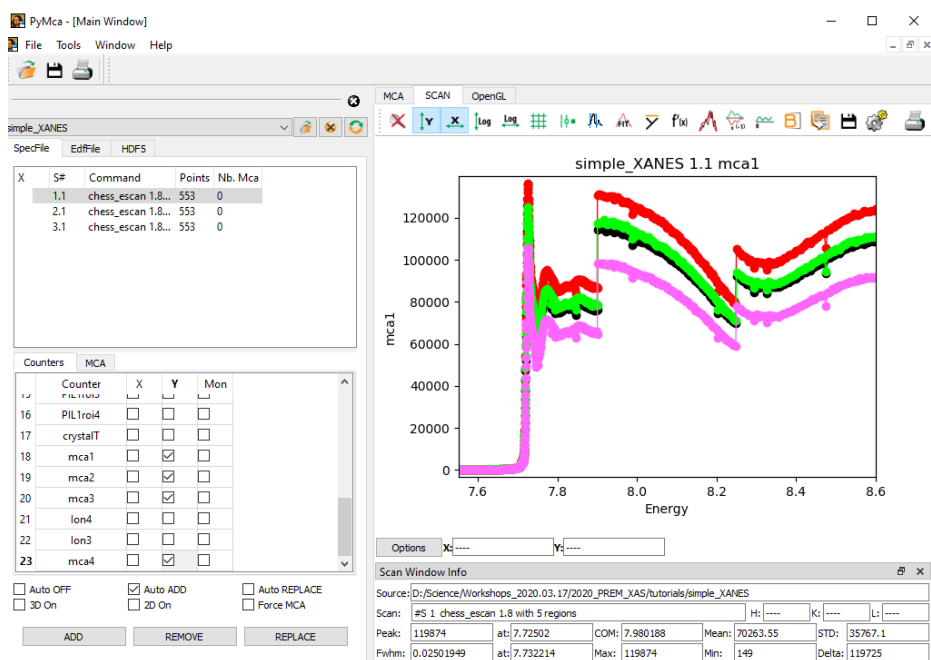


Figure 1-7: The four elements of the Vortex detector are now plotted in PyMCA.

- 8) It's important to note that at this point the y-axis is plotted in terms of detector counts—which is the raw signal coming from the detector—which explains why the spectra look funny and have strange discontinuities. This happens both because the intensity of the incident beam changes with energy and also because different points in the spectrum were collected using different counting times, resulting in different amounts of total signal collected. To correct for these effects, it's critical to normalize the detector signal to the incident beam intensity. We do this by dividing the detector signal by the intensity of the incident beam; at the beamline, we had an ion chamber set up just before the sample to record the incident beam intensity (the “**Ion3**” channel in PyMCA), so to normalize the data check the “Ion3” box in the “mon” column. Your PyMCA window should look like Figure 1-8.

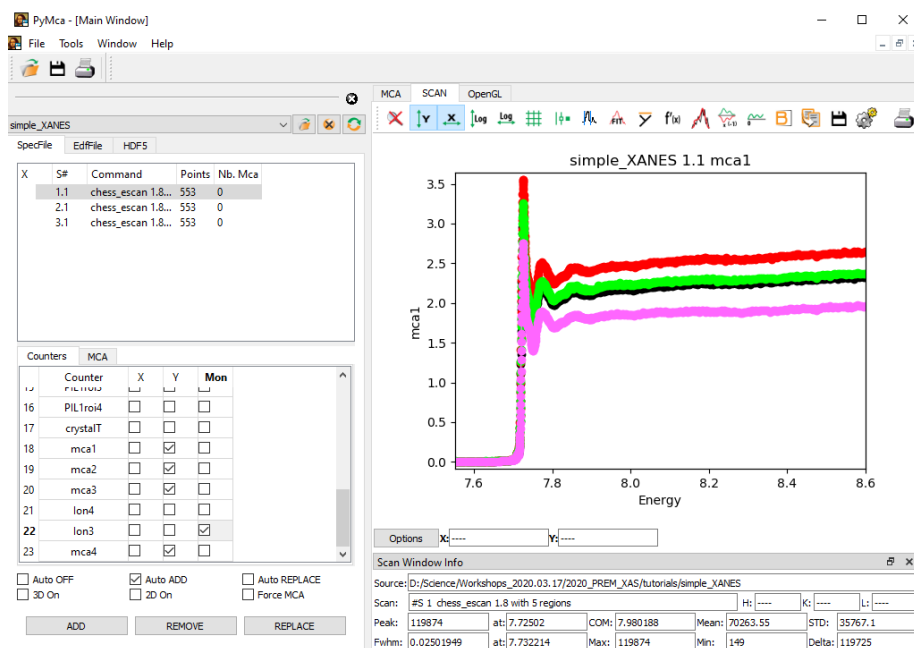


Figure 1-8: This is the XAS data after normalization to the incident beam intensity.

Question 1: Using the energy reported on the x-axis (in units of keV), the edge we are looking at corresponds to what element?

Question 2: The four Vortex detector elements do not have the same intensity, even though they are all normalized and were all looking at the same sample; why could this be?

- 9) Congratulations, you've plotted your first XAS spectrum!
- 10) Since this is a XANES tutorial, most of the information we're interested in is concentrated near the edge, which is only a ≈ 50 eV region within this 1000 eV scan. Thankfully, PyMCA makes it easy to zoom in on certain parts of the spectra; to zoom, click somewhere just below the spectra around 7700 eV and, holding the mouse button down, draw a box around the edge region, stopping around 7800 eV. The box you draw should be similar to Figure 1-9a. When you're done, release the mouse button and PyMCA should zoom to the region you chose, hopefully resulting in something like Figure 1-9b.

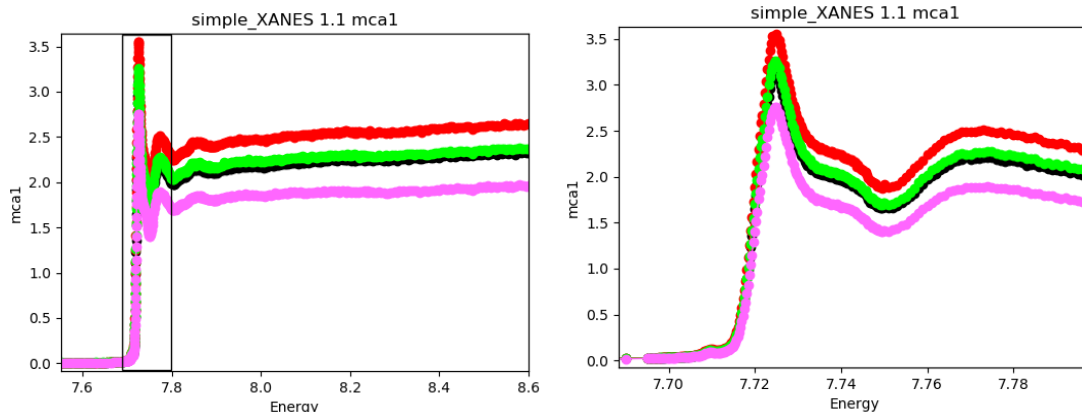




Figure 1-9: a) This is the region we want to zoom in on. b) Here is the result of the zoom. Notice how much more detail is now visible.

- 11) If you don't like your zoom selection or simply want to see the whole spectrum again, you can reset the view by either clicking the Autoscale Graph icon in panel D () or by simply right clicking somewhere on the graph.
- 12) Since we want to see small details in these spectra, having the data plotted with big dots at every data point is suboptimal. Thankfully we can change this easily in PyMCA by clicking on the "Toggle Points/Lines" button in panel D (). You may need to click this a few times, but eventually you should get to a plot with just lines (Figure 1-10).

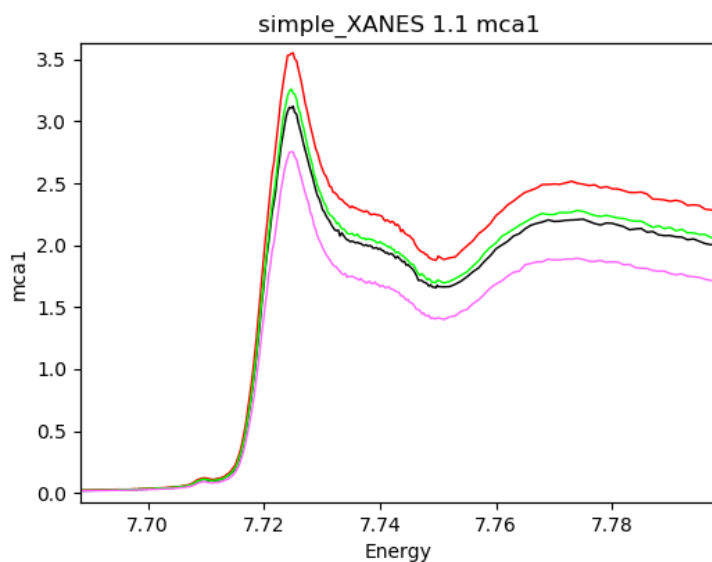


Figure 1-10: By just plotting lines we can see much more detail in the plots.

- 13) Now that we can plot the data, it's time to start doing some real processing. For doing so, there's a general strategy that looks something like:
 - Make sure there's no evidence of damage occurring to the sample between various scans
 - Assess all channels of all scans for glitches and other artifacts

- If no damage or other artifacts, average all channels / scans together
- Export for further analysis / processing

Starting with the first bullet point, we'll begin by looking for any evidence of sample damage. One of the clearest signs of damage is a change in the position or shape of the edge, so we'll assess damage by overlaying the three scans to make sure their edges all appear at the same energy. To do this, it's easiest to uncheck all of the "mca" channel buttons in panel B except for "mca1" so that only a single spectrum will be plotted per scan (otherwise the plot gets too messy to see anything). At first nothing will happen, so press the "Replace" button in panel C to remove all but the mca1 spectrum. Your PyMCA plot should now look like Figure 1-11.

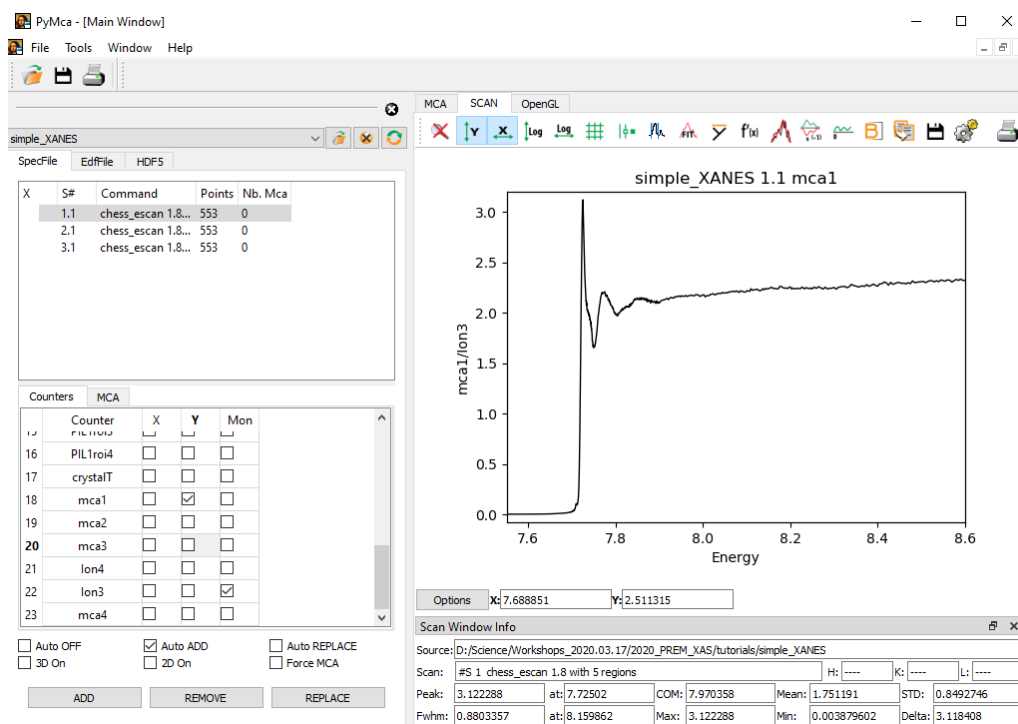


Figure 1-11: Only the mca1 channel of the scan is being plotted.

- 14) At this point we are ready to begin overlaying and comparing the three scans. Make sure that the "Auto ADD" box in panel C is checked, then click on each of the first three scans in panel A to add them to the plot. When you're done, the results should look like Figure 1-12.

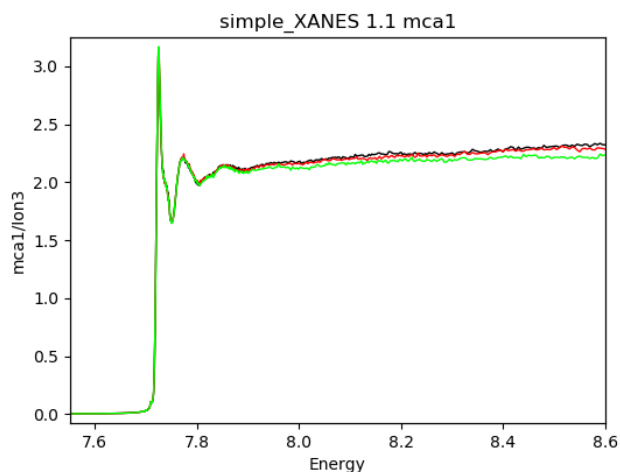


Figure 1-12: The mca1 channels of all three scans are overlaid here.

- 15) You'll notice that the spectra are similar, but that the backgrounds differ slightly (from 8000 to 8600 eV). This is a somewhat common observation in XAS, especially when collecting spectra from multiple spots on a sample, so it isn't terribly concerning.
- 16) Before we can compare the spectra to assess damage, there's one last thing we need to do: Normalize the intensities to account for any differences in intensity between spots. If we don't, differences in intensity can easily be misinterpreted as differences in energy, which is not a mistake we want to make. PyMCA offers a number of ways to normalize spectra, each with its own strengths and weaknesses. For the spectra here, the best choice is to go to panel D and select 1D Plugins > Normalization > $y/\max(y)$, which will divide each of the spectra by its largest value. With that done, zoom in to the edge so you can focus on the region from 7705 – 7735 eV (Figure 1-13).

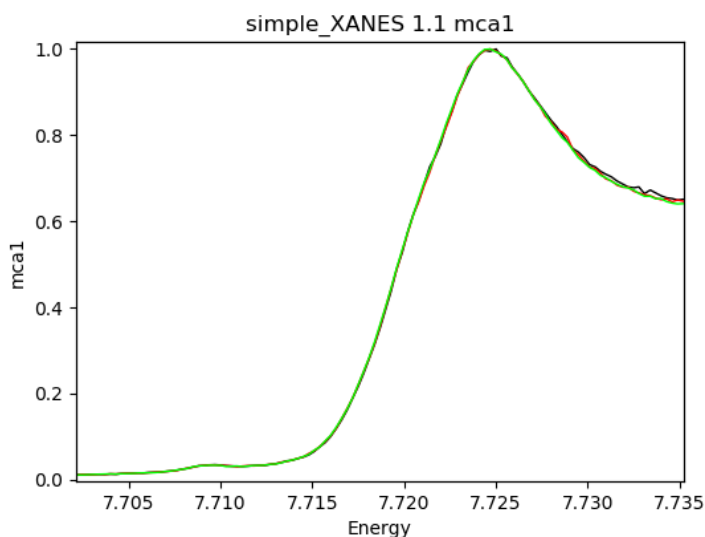


Figure 1-13: The normalized edges of the three spectra are shown here.

Question 3: Based on the edges, do you think this sample was damaging during measurement? Why or why not?

- 17) Hopefully we can all agree that these edges look essentially identical, so we can safely say that no damage is occurring. With that decided, we can now go ahead and average together all of the channels from all of the spectra. Back in panel B, check the boxes for mca1, mca2, mca3, and mca4, click the “Replace” button to get rid of the normalized spectra, and then click on each of the three scans to plot all 12 spectra. Finally, click on the average icon (☑) in panel D to average all of these together. To make the resulting spectrum less ugly, you can click the “Toggle Points / Lines” icon in panel D until you get a smooth line; your average should look like Figure 1-14.

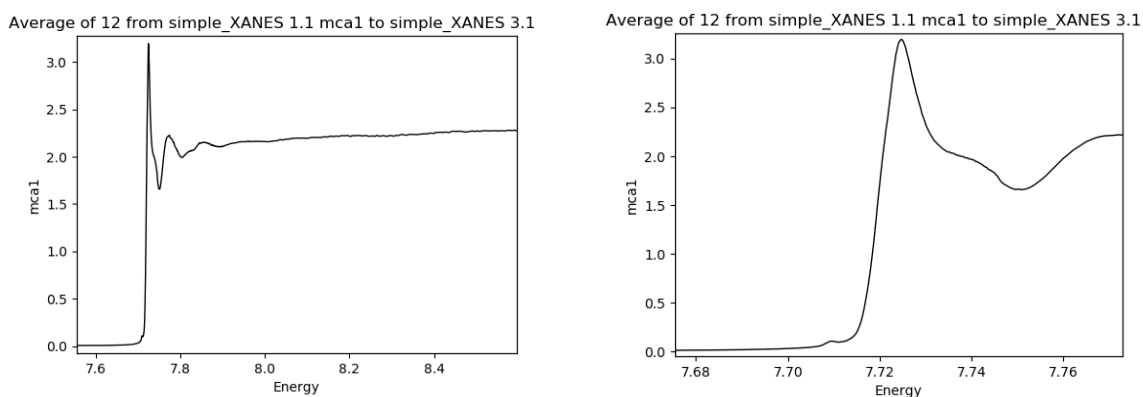


Figure 1-14: All mca channels from all three spectra are averaged together, showing the whole spectrum (left) and just the edge (right).

Question 4: We did not normalize the spectra before averaging them together. Why do you think we didn't?

- 18) At the beamline, you'll likely find yourself doing the steps we just did quite often to assess how data collection is going and to see when it's ok to stop scanning on a sample. PyMCA is great for this kind of work as it allows all of these common data processing steps to be done quickly and easily. Advancing into deeper data processing, however, generally requires additional, specialized, programs. In order to use those, you would need to export your processed PyMCA data into a form that can be read by other software. Thankfully this step is easy: With your averaged spectrum plotted, click the “Save” icon in panel D to bring up the save dialogue box. Navigate to where you want to save the file and—**importantly!**—change the file type from *.mca to comma-separated *.csv file (Figure 1-15); save the file and you're done.

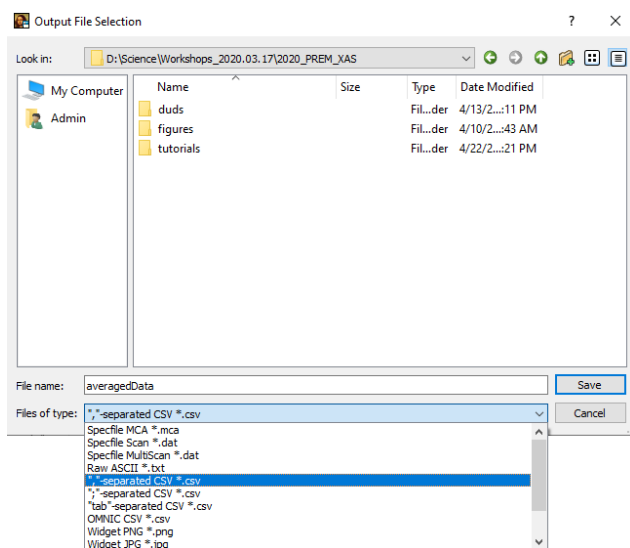


Figure 1-15: Save the file as a *.csv file to export from PyMCA.

Part 2: Processing and Assessing EXAFS

- 1) With that out of the way we can move on to processing some EXAFS. The first few steps here will hopefully be at least somewhat familiar from the XANES tutorial since the first steps of processing XANES and EXAFS data are exactly the same; in fact, we'll be using the exact same scans for EXAFS that we used for the XANES tutorial. Since we've already established that those spectra are not damaged, we can go ahead and average the data from the three XAS scans if they aren't already (if you hadn't already done this, you would want to do it in the same way we did in the XANES tutorial). Plot the data as we did before, selecting the "Energy" channel for the x axis, the "mca1", "mca2", "mca3", and "mca4" channels for the y axis, and the "ion3" channel for mon. With that setup done, plot all three spectra together and then go ahead and average all mca channels for all three scans together. Your plot should look like Figure 2-1.

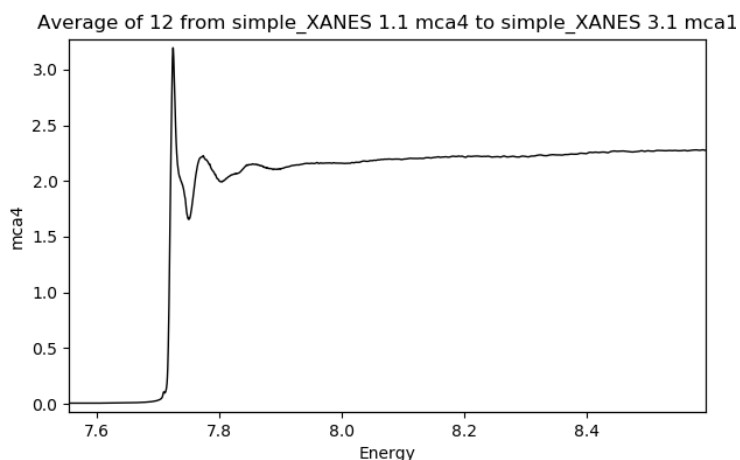


Figure 2-1: The averaged spectrum is shown.

- 2) When processing and analyzing EXAFS data for publications, you'll want to use a dedicated program such as Athena / Artemis, but for quick data assessment at the beamline PyMCA works very well and so that's how we'll proceed. As we discussed during the lectures, there are several different ways to plot EXAFS data, including various k -weightings (e.g. k^2 , k^3) and Fourier transforming the data, all of which are available in PyMCA. We'll start by plotting the k^2 -weighted EXAFS to enhance the EXAFS signal at high k (remember that the signal drops off as energy increases); click the "1D Plugins" icon in panel D and then navigate to 1D Plugins > XAS > EXAFS * k^2 . Doing so should result in the plot shown in Figure 2-2.

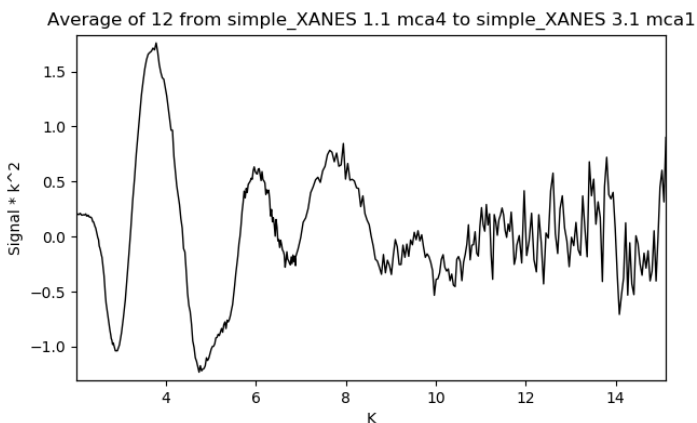


Figure 2-2: The k^2 -weighted EXAFS are plotted here.

Question 1: Assess the data quality as k increases—why do you think you observe this behavior?

- 3) Let's compare this to the k^3 -weighted EXAFS. To view those we need to backtrack a little bit and redo the average of the mca channels. Click one of the scans in panel A, hit the "Replace" button in panel C, and then click the other two scans to replot all of the mca data. Redo the average, then go to 1D Plugins > XAS > EXAFS * k^3 to generate the k^3 -weighted EXAFS (Figure 2-3).

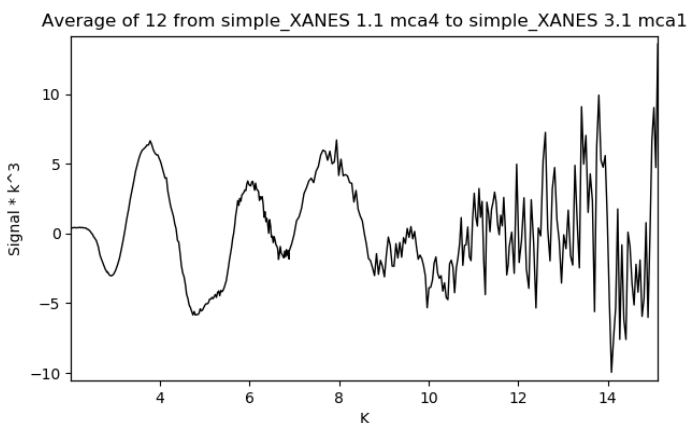


Figure 2-3: A plot of the k^3 -weighted EXAFS is shown.

Question 2: How do the plots of k^2 - and k^3 -weighted EXAFS differ?

- 4) Finally, let's look at the Fourier transform of the data to see something that can more readily be qualitatively interpreted. Again, redo the average of the mca data, then choose 1D Plugins > XAS > FT. You should get the result shown in Figure 2-4.

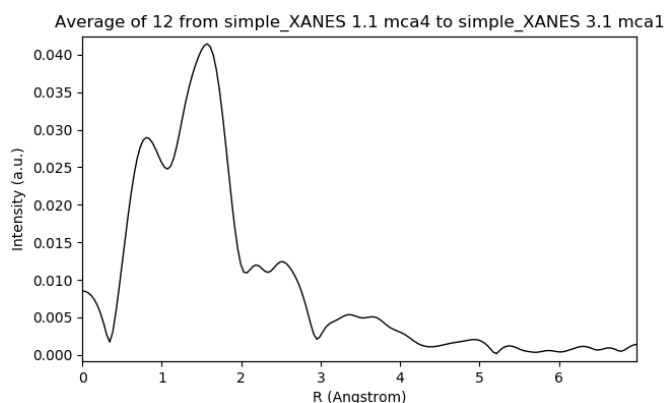


Figure 2-4: The Fourier transform of the EXAFS data.

- 5) Yikes! That's a pretty ugly Fourier transform! There's a lot of intensity below $R = 1$, which is not physically reasonable since no Co-ligand bonds will ever be $< 1 \text{ \AA}$, so we know the spline function needs work. The good news is that the intensity between $R = 5 - 6$ is low, so the noise isn't too bad. Thankfully the spline function can be modified in PyMCA so we can get an idea of what the data are telling us. To do this, again replot the average of the mca data one more time and then navigate to 1D Plugins > XAS > Configure to pull up the window shown in Figure 2-5.

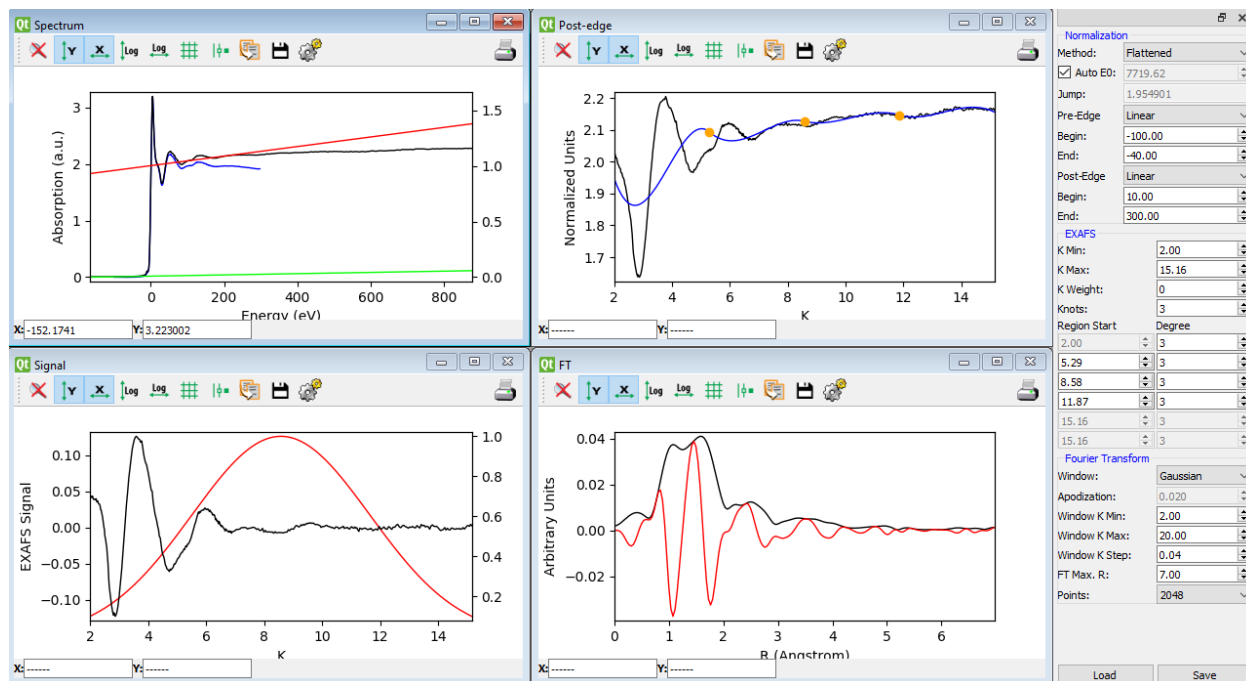


Figure 2-5: This window allows manual control over background and spline removal.

- 6) This is a lot of windows and a lot of information, but try not to feel overwhelmed, we'll go through what everything does one step at a time. The top left panel shows the pre-edge (green) and post-edge (red) background removal, the top right shows the EXAFS spline removal, the bottom left shows the Fourier transform window, and the bottom right shows the Fourier transformed data in black. The panel on the far right allows for control over parameters that control the various windows.
- 7) The first thing we will do is adjust the pre-edge and post-edge background removal from the XAS data. The goal is to get a flat post-edge with no slope and a reasonably flat pre-edge background. In the far right panel, adjust the pre-edge "Begin" value to be as low as possible (-164.79); this will make the pre-edge background begin fitting at the very beginning of the spectrum which is what we want here. Next, adjust the pre-edge "End" value to around -110 or so, which will make the pre-edge background fit the region from -164.79 eV to -110 eV below the edge (a zoom in shown in Figure 2-6). You'll notice that the green line representing the background is linear and nearly flat, which is a common observation for fluorescence data. Sometimes a sloping or even quadratic fit is needed, particularly for lower concentration samples, but that's not the case here.

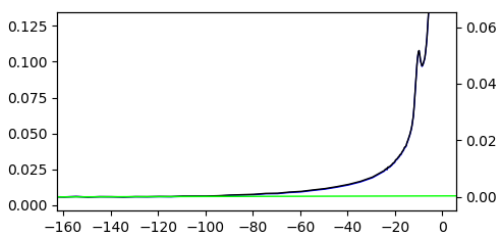


Figure 2-6: The fit to the pre-edge background is shown here.

- 8) With the pre-edge background removed, we'll now move on to removing the post-edge background. There is a lot of variability in post-edge backgrounds and it's common to need a quadratic fit to adequately remove them. Before we get to that, we should change the "End" value of the post-edge subtraction to be the end of the scan, in this case 875.43 eV; you'll note that doing this extends the blue subtracted plot to the end of the scan. If you zoom in on the post edge region (using the plot in the upper left window), you'll notice that at this point the subtraction causes the data to have a slightly negative slope (Figure 2-7). We want the slope to be as close to zero as possible, so the parameters still need some tweaking.

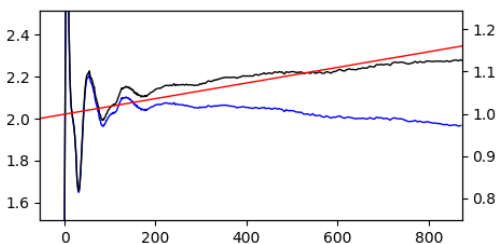


Figure 2-7: With this subtraction the post-edge still has a bit of negative slope.

- 9) Adjust the "Start" value of the post-edge subtraction to be 1.00 eV and the slope will improve significantly. If you look carefully though, you'll notice that there is still a bit of a "bulge" in the blue curve (the background-subtracted data), where the middle of the post-edge curves up

slightly from either end (it's ok if you can't see this, it takes some practice to notice such small effects :-). Feel free to play around with the start and end values to try and remove this bulge.

- 10) You'll likely notice that the bulge in the post-edge is always present, even when the overall slope is still close to zero. This is indicative of the need for a quadratic function for the background subtraction. In the "Post-Edge" box, change "Linear" to "Parabolic" and then adjust the start and end points to try and improve the slope of the post-edge region. With some tweaking I managed to get mine to look like Figure 2-8.

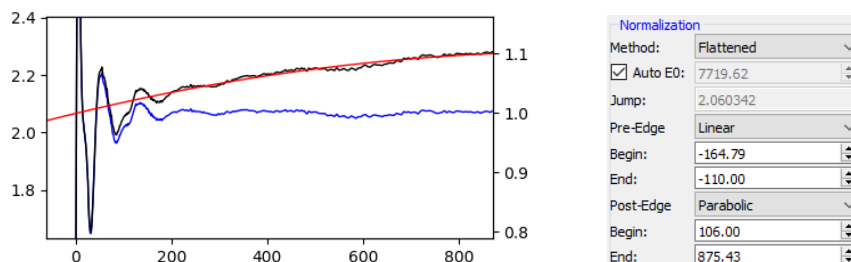


Figure 2-8: A parabolic background subtraction does a good job of flattening the post-edge region.

- 11) For an easy step, we'll set up how the Fourier transform is carried out. Under the "EXAFS" heading in the far right panel, change the "K Weight" value from 0 to 3. Next, change the "Window" function from "Gaussian" to "Hanning." This will cause the lower left panel to look like Figure 2-9.

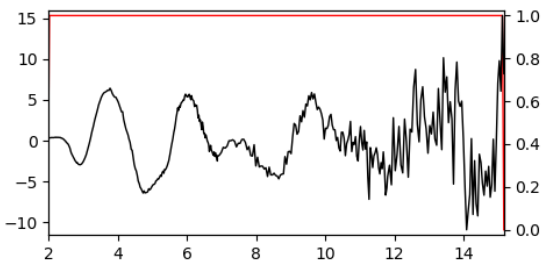


Figure 2-9: A Hanning function is used as the window function for the Fourier transform of k^3 -weighted EXAFS.

- 12) We are now finally ready to adjust the spline. The first thing we want to do is adjust the region over which we spline the data. By looking in the lower left panel, you can see that the EXAFS signal at $k > 12$ is very noisy and will be difficult to fit, so we should exclude that data before it has a chance to mess things up. In the "EXAFS" section in the right hand panel, decrease the "K Max" value to 12 and decrease "K Min" to 1; the plot of the EXAFS should update in real time to show the low quality data being excluded (Figure 2-10).

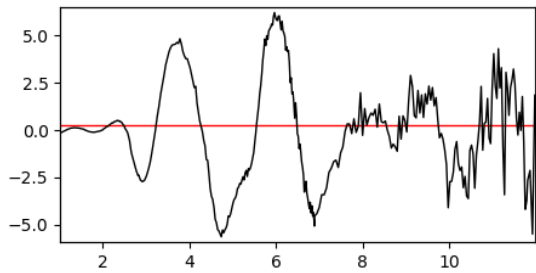


Figure 2-10: The low quality EXAFS data at high k (>12) have been excluded from the plot.

Question 3: The current EXAFS data are displayed from $k = 1 - 12$; over this k range, what is the expected resolution, in Å, of the Fourier transform?

- 13) From the lectures, you'll remember that the spline is a series of low order polynomials used to remove low frequency oscillations resulting from an atom scattering off its own electrons. With this in mind, there are a few things we want to accomplish with a spline: 1) we want to effectively remove low frequency oscillations from the EXAFS, which we can assess by looking for intensity below $R = 1$ in the FT, and 2) we don't want the spline to remove actual EXAFS wiggles, either by having too many segments or by using too high order polynomials. As a starting point, set the "Knots" value to 2, meaning that three polynomials are used (Figure 2-11). These settings will use three third order polynomials that are spliced (knotted, hence two knots) together to remove the spline from the EXAFS. The result should look like Figure 2-12.

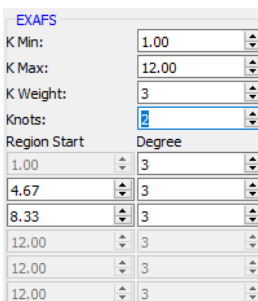


Figure 2-11: Here are the spline parameters we are starting with.

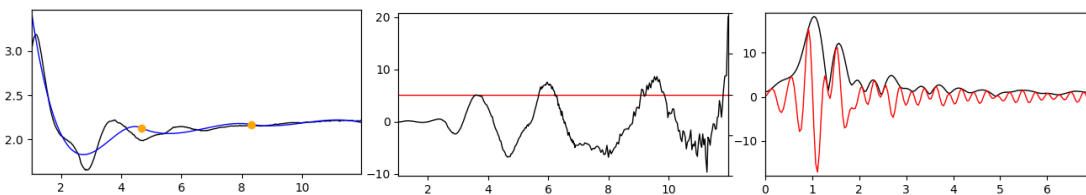


Figure 2-12: The EXAFS resulting from the subtraction of this spline are shown here.

- 14) From the FT you can see that the current spline does not do a good job of removing the intensity at low R . To fix that we're going to move the spline points around to try and improve the look of things. Go ahead and change around the "Region Start" values for the two knots to get things to look better, the only guideline here is to try and keep the knots equally spaced out in k space.

Have fun, and be prepared to see some really wacky FTs along the way. Eventually you'll hopefully end up with something like Figure 2-13 (don't cheat and look ahead!).

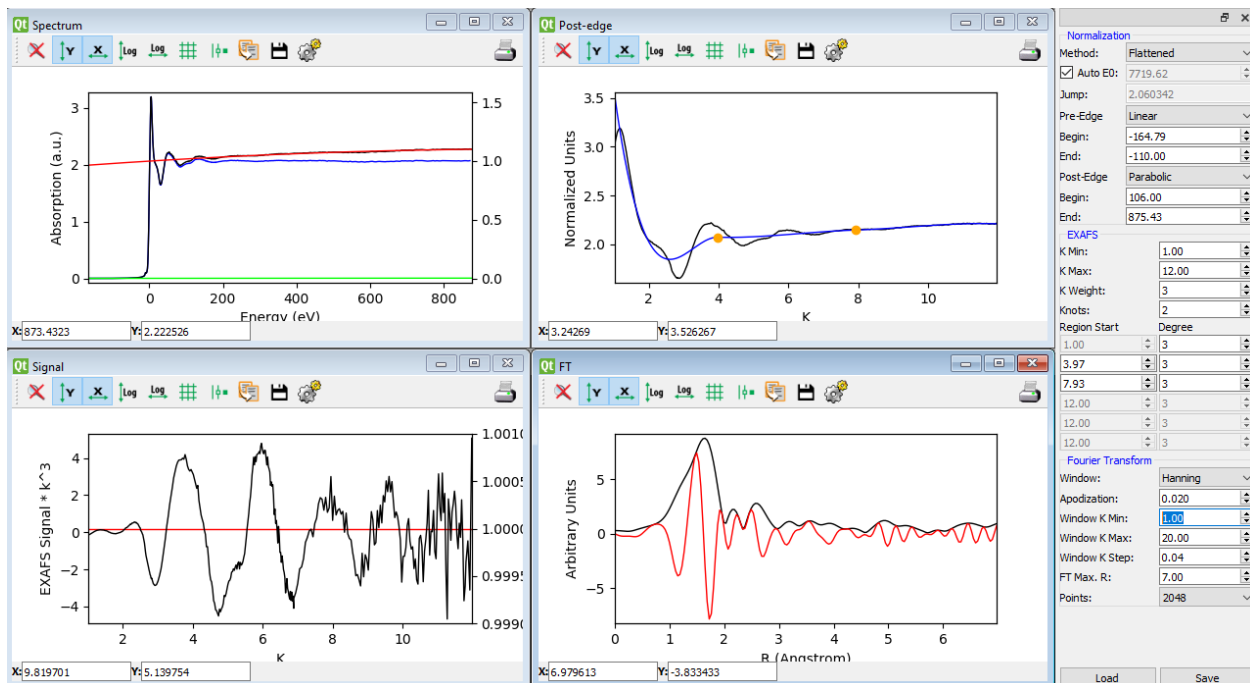


Figure 2-13: These parameters give a decent spline with most of the intensity below $R = 1$ removed.

- 15) Wasn't that fun? At least now we know a good spline is possible. When you're collecting data at the beamline, this is the procedure you'll want to go through periodically to assess the quality of the data you're collecting. Unfortunately, robust EXAFS analysis isn't possible within PyMCA, so programs such as Athena / Artemis are needed for actual analysis. Don't worry though, what we've done so far is certainly enough to be able to assess data at the beamline!

Part 3: Processing and Assessing XES Data

- 1) With XAS out of the way, we're ready to move on to looking at some XES data. Thankfully, the process for working with XES data is very similar to working with XANES data, so much of this should look familiar from earlier parts of the tutorial.
- 2) As a reminder, XES spectra are collected using a setup like the one shown in Figure 3-1, where an incident x-ray beam excites a sample and the resulting fluorescence is reflected onto a detector using crystal analyzers. At CHESS, we measure the incident intensity at the sample using an ion chamber (Ion1) and collect the XES fluorescence using a Pilatus 2D detector. In order to scan the emitted energy, the analyzers and detector move along the indicated circles (so-called "Rowland circles"), so the data we collect are a function of analyzer angle rather than emitted energy. Calibration from angle to energy is straightforward, but outside the scope of this tutorial.

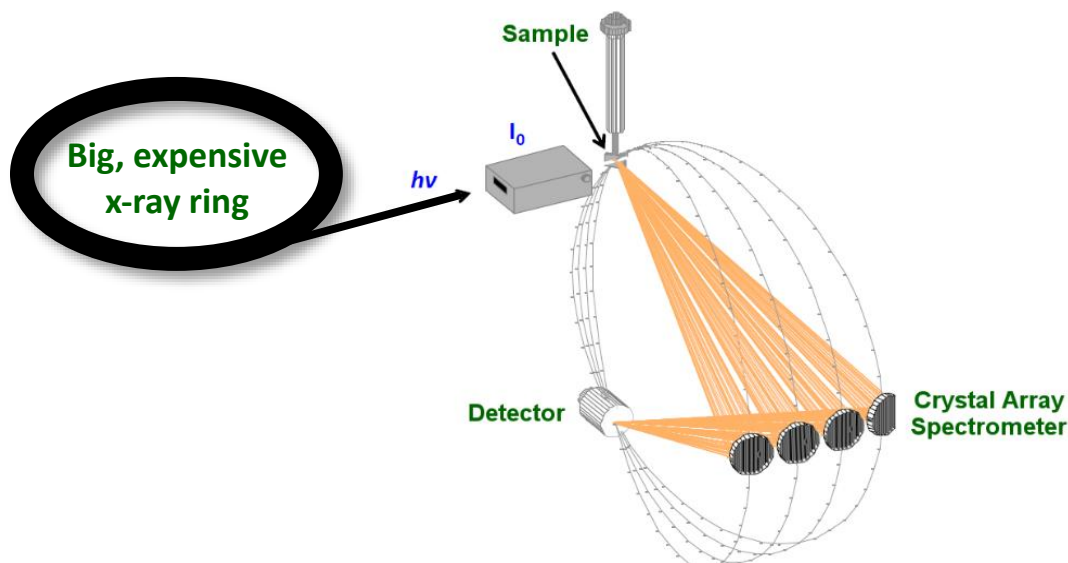


Figure 3-1: A cartoon of the experimental setup for XES.

- 3) Moving on to the data, go ahead and click on scan number 4 to plot it, making sure that the “X” column is chosen to be “xes_dn_ana”, “Y” is “PIL1roi1”, and “mon” is “lon1”. Once that’s done, click on scans 5 and 6 to plot them as well. Once plotted, the PyMCA window should look like Figure 3-2.

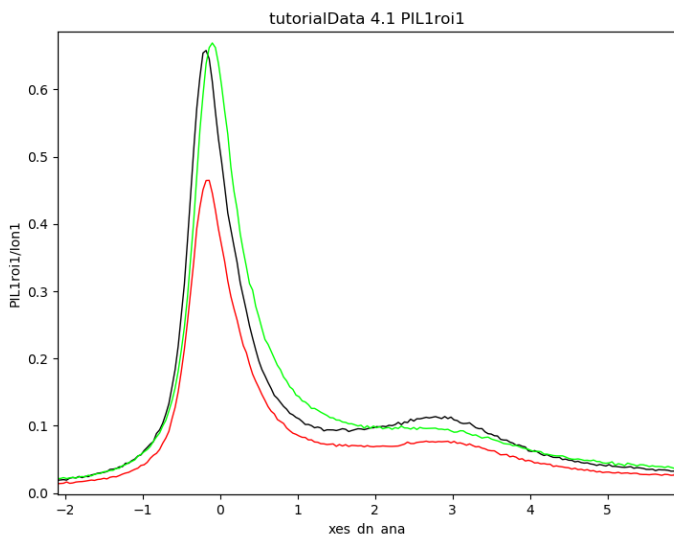


Figure 3-2: The three Kβ mainline spectra are plotted

- 4) These are Kβ mainline spectra from three different high spin manganese compounds, and we can use these spectra to assess differences in the oxidation states between these compounds. As you can see, though, it’s hard to compare the spectra because of their different intensities. Because the Kβ mainline spectra all correspond to 3p → 1s transitions, it’s reasonable to assume that all mainline spectra for a given element should have the same integrated intensity, regardless of

metal oxidation state. What that means is that we can use PyMCA to set the areas of the three spectra to be equal, which will make comparison much easier. To do that, click on the “Call / load 1D Plugins” button in panel D and then go to Normalization > $(y - \min(y)) / \sum(\max(y) - \min(y))$ (the second option on the list) to normalize by area. The result should look like Figure 3-3.

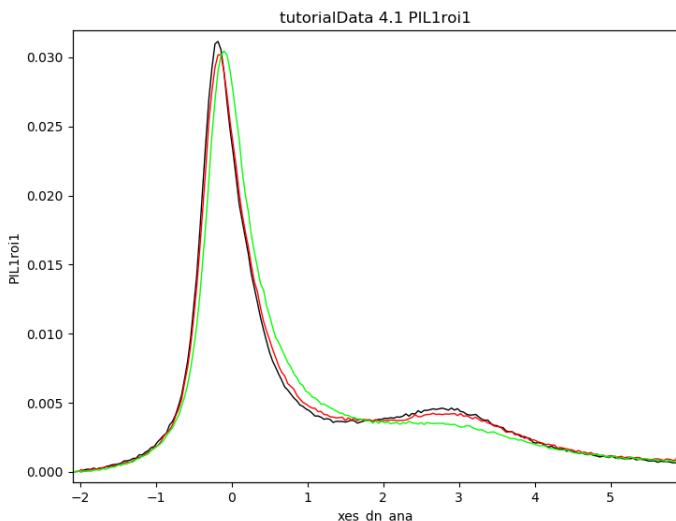


Figure 3-3: The three mainline spectra have been normalized by area.

Question 1: If the compound measured in scan 4 was Mn(II), what can you say about the oxidation states of the compounds measured in scans 5 and 6? Hint: Think about how the spin state of Mn changes as the oxidation state increases / decreases.

- 5) The benefit of K β mainline spectra is that they tend to be quite intense; even these single scans have excellent signal-to-noise ratios. This makes mainline spectra useful when measuring dilute samples, as even low concentration samples will yield nice spectra after only a few scans.
- 6) Valence-to-core (VtC) spectra, on the other hand, tend to be a factor of 50 – 100x lower in intensity as compared to K β mainlines. For an illustration of this, click on scan 4 and then hit the “Replace” button in panel C to plot only a single K β mainline. Next, click on scan 7 to also plot a VtC spectrum. As shown in Figure 3-4, the VtC spectrum is **much** weaker than the corresponding mainline!

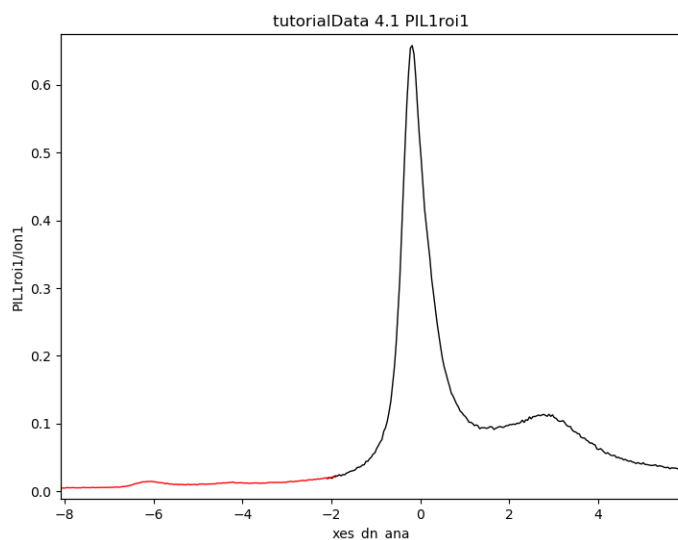


Figure 3-4: A plot showing both a $K\beta$ mainline and VtC spectrum.

- 7) As you can tell, at this scale it's impossible to see anything useful in the VtC region, so hit the "Replace" button in panel C to plot only the VtC spectrum (Figure 3-5). One of the most noticeable attributes of the spectrum is how much noisier it is than the mainline, as might be expected given how much lower the overall intensity is. For this reason, VtC spectra typically require many scans to get good statistics, even for concentrated samples; for dilute samples, good VtC data can take 12 – 24+ hours of scanning!

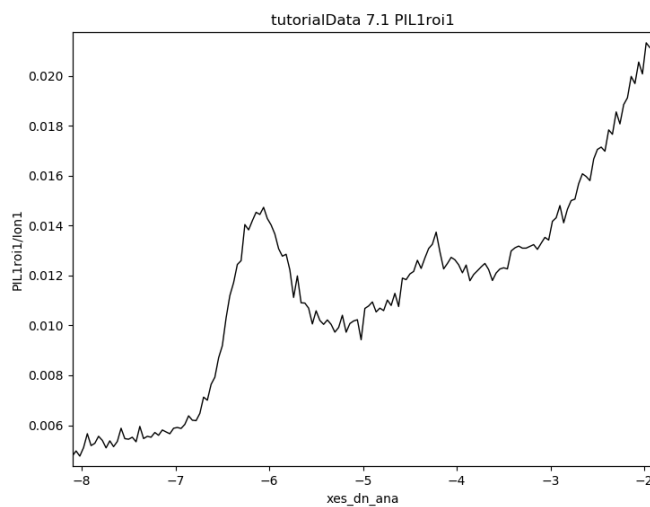


Figure 3-6: A single VtC spectrum is plotted.

- 8) Scans 7, 8, and 9 are all VtC spectra, so go ahead and plot all three of them together and then take the average. The result should look like Figure 3-7, where the improved signal-to-noise relative to one scan is clear. At least two peaks are apparent in the data—the $K\beta$ at -4.25° and the $K\beta_{2,5}$ at -6.1° —which can be assigned as originating from ligand 2s and 2p orbitals, respectively. Rigorous calibration and fitting of the data would be needed to assign the ligand identity using these data, but this type of quick analysis is useful for assessing the quality of the data at the

beamline; in the current case, it's likely that we'd want at least another 3 – 6 scans in order to improve the signal-to-noise.

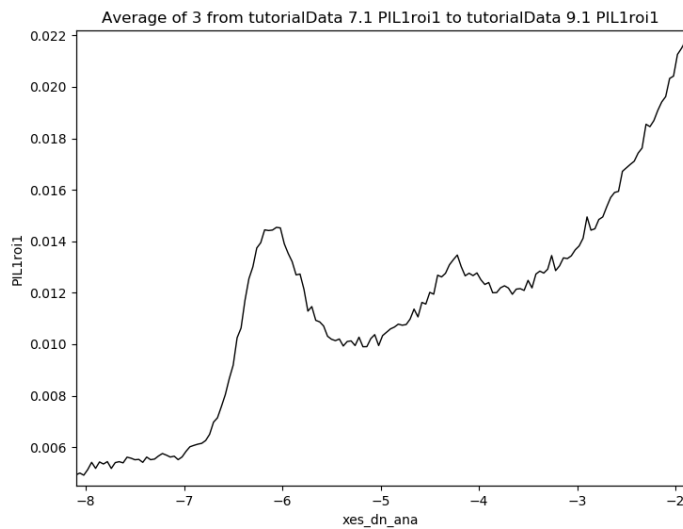


Figure 3-7: The averaged VtC data is shown.

Congratulations! At this point you've processed data for three different spectroscopic techniques and learned the basics of using PyMCA. Hopefully you've found this tutorial useful, and with any luck you'll get to use the skills you've learn when you're collecting data at PIPOXS in the future 😊