**Instructions for XRF Analysis using PyMca and the Homebuilt Scripts**

**Help Document Version 0.1; Written by J. Hayes**

**Last Edit: November 29, 2016**

Introduction:

This document will guide you through the process of analyzing the XRF data collected from the VESPERS beamline at the Canadian Light Source. This is useful for mineral analysis and phenotypic studies. In theory, it could also be used to analyze XRF data from a wide variety of sources, though this has not been tested.

The current method for the program is that two full analyses must be carried out for a single set of data. The first analysis concerns determining the concentration of all the non-Ca elements of interest present in the sample. The second analysis concerns determining only the concentration of Ca in the samples. There are two main differences between these analyses: 1) the fit configuration file used in PyMca is different (see below) and 2) The calibration concentrations are usually different. **You should save these two analyses in different folders to avoid renaming/accidentally changing your data!**

The Homebuilt Scripts are written in Python 3.5, and a suitable interpreter should be downloaded. If Python is not already installed on your computer, I recommend downloading Anaconda, which includes a number of important scientific packages. An installer for Anaconda is available in the shared MolecCropQualCommon drive in the Resources folder. The Homebuilt Scripts are contained in a folder called XRF Analysis Script in the Resources folder.

This guide has been broken into two sections. The first is a brief, step-by-step guide /summary for how to perform the analysis. The second section of this guide will explicitly detail all the steps in the analysis process.

**Section 1: Summary of Analysis Steps**

1. Format plate sample data in a CSV file that can be read by the Analysis program
2. Put Io files and Spectrum files into two separate folders (one for the Io files, one for the spectra files)
3. Run the Transpose.py script in the folder containing the spectra files
4. In PyMCA, run a batch analysis using the BatchConfigSetting to analyze the non-calcium elements
5. In PyMCA, run a batch analysis using the BatchConfigSetting-Ca to analyze the Ca contents
6. Run the caller.py module in the XRF Analysis Script folder
7. In the CLI, follow the on-screen prompts, and input the location of the directory containing the spectra files, the directory containing the Io files, the CSV file containing the sample labels
8. Indicate whether you are performing a Ca analysis by entering Y (yes) or N (no) when indicated
9. When prompted, select Option 1 to perform a full analysis of the set of spectra
10. During the analysis, the program will ask for the concentrations of the calibration samples in ppm
11. You can exit the program and store the current file parameters or choose to exit the program and delete the file parameters

**Section 2: In Depth Discussion of Analysis Steps**

Step One: Format plate Sample data in CSV file

The program which analyzes the XRF data must be formatted in CSV file to allow the software to identify each sample and label them appropriately. These files can be made somewhat conveniently using Microsoft Excel. The format is as follows, with an example in Figure 1:

1. The first row of the spreadsheet should contain Plate in column A (to identify the plate name), and then Spot 01, Spot 02, Spot 03…Spot 84
2. The data for each plate is put in separate rows. The name put in the plate column **MUST MATCH THE PLATE NAME USED WHEN COLLECTING THE DATA** (including spaces, underscores, etc)
3. Calibration data must be labelled as Cal 1, Cal 2, Cal 3…Cal N exactly. (**Capitalizing the C in Cal is required**). There is no specific order for which concentration must be Cal 1, Cal 2, etc., but it must be consistent across all data points
4. All plate spots where there is no sample should be labelled as “Empty” or “empty”

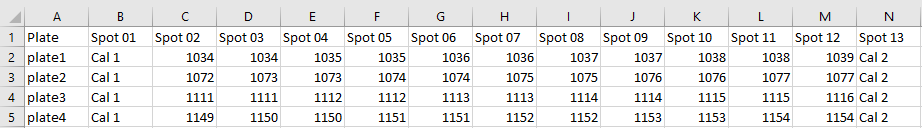


Figure 1: Example of formatted plate data

Once the data has been input into the spreadsheet, the data can be saved as a CSV file. To do this, go to File --> Save as…, and in the drop down box showing formats, choose “CSV (Comma delimited)(\*.csv), as shown below in Figure 2.

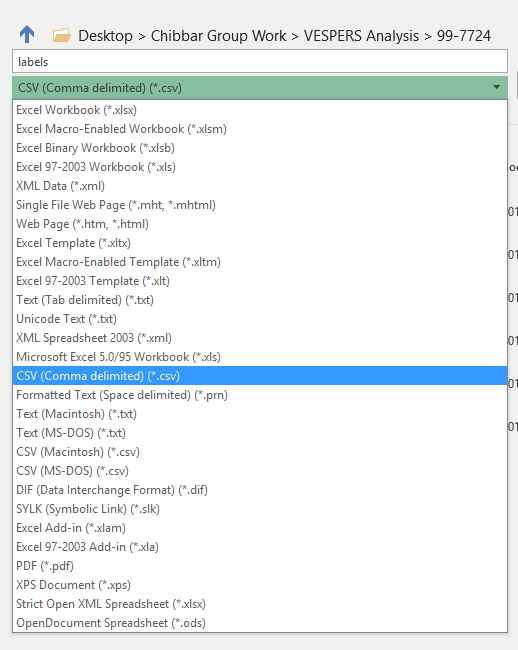


Figure 2: Save the label data as a CSV file

Step Two: Formatting and Organizing Data for Analysis

The data collected from the beamline is not formatted in a way that can be read by PyMca, so the first step is to format and organize the data for analysis. The first step is to put all of the files containing the Io information into a separate directory (folder). Io files will have the following format:

*Plate\_Name*\_1.dat

Where *Plate\_Name* is the name of the plate being analyzed (set during the collection process). The data collecting the spectra should also be saved to a separate directory. This data will generally have the following naming format:

Plate\_Name\_Ge13El\_1.dat

It is important to note that these file names **DO NOT contain Vline or Hline** in the file name.

Once the Io and spectra files have been separated into different directories, we can run the transpose.py program to format the spectra data. To do this, open up a Command Prompt window (Figure 3) and use the cd command to navigate to the folder containing the transpose.py file. (Note that in Windows, the contents of a directory can be found using the “dir” command, while when using Linux or Mac, the contents of a directory can be listed using the “ls” command).

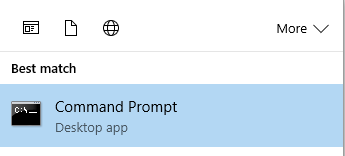


Figure 3: The command prompt window can be opened by clicking the windows icon and searching for “cmd” or “Command Prompt

Once you have navigated to the folder containing the transpose.py script, run the script by typing:

python transpose.py

As shown in Figure 4. The program will then ask for the directory containing the spectra files. Input the full directory address, using backslashes and press return (an example is shown in Figure 4).

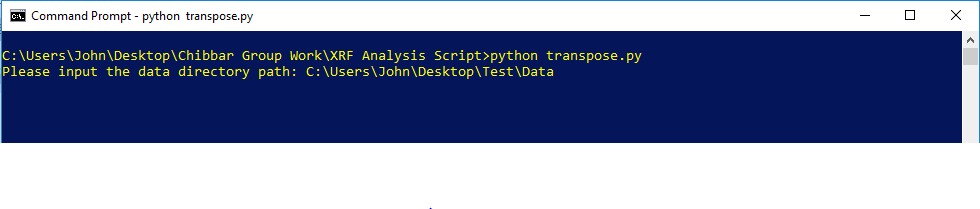


Figure 4: Running the transpose program – the inputs required

The transpose program will save the output files in a folder named “Transpose” in the directory containing the data. The program output/results of running the Tranpose.py script are shown in Figure 5.

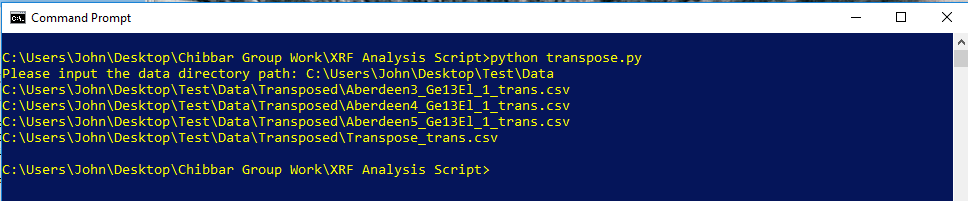


Figure 5: The results of running transpose.py

Step Three: Fitting the Background and Determining Peak Area using PyMca

Now that the data has been formatted, it’s time to do the analysis. To do this, open up the program PyMCA. Depending on what version you are using, the program might look slightly different, but the steps remain the same. **Unfortunately, at this time the analysis for the Ca peak and the analysis for the non-Ca elements must be done separately.**

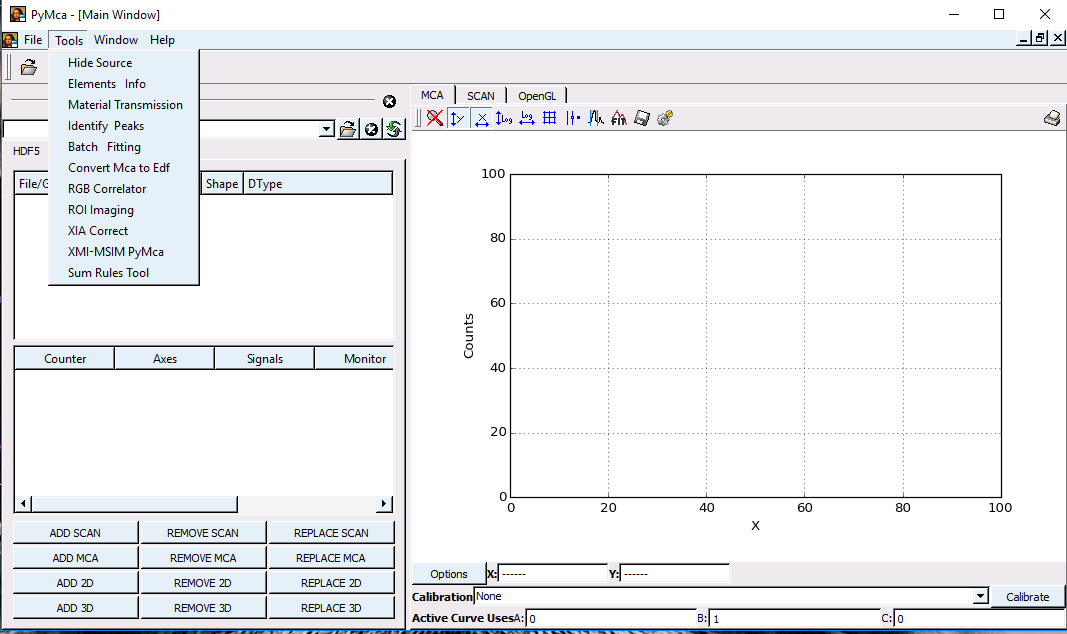


Figure 6: To analyze the data, select the batch fitting option from the

In the PyMCA window, click on the tools menus and select the “Batch Fitting Option (Figure 6). The window shown in Figure 8 will pop up, which we will now fill out.

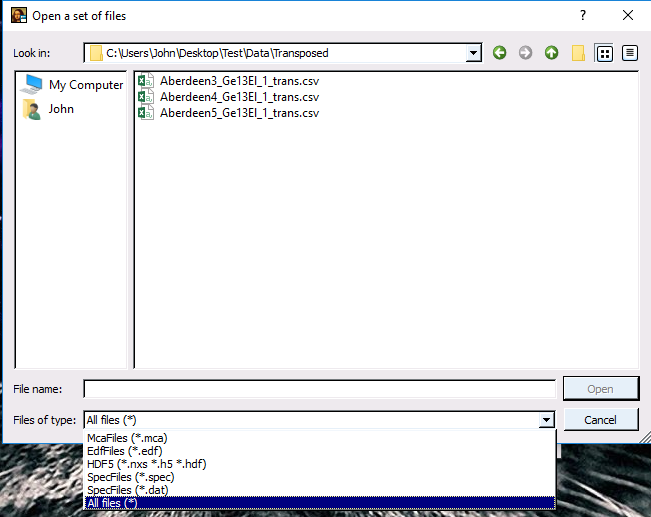
For the Input File List pane, select the browse button, and navigate to the folder containing the transposed data (from the previous step). Select all the data to be analyzed and click OK. Note that you will have to select the All Files option in the Files of Type menu to see the spectra files output by the Tranpose Program (Figure 7). 

Figure 7: If you don’t see your spectra files, ensure that you select the All files option from the Files of Type list.

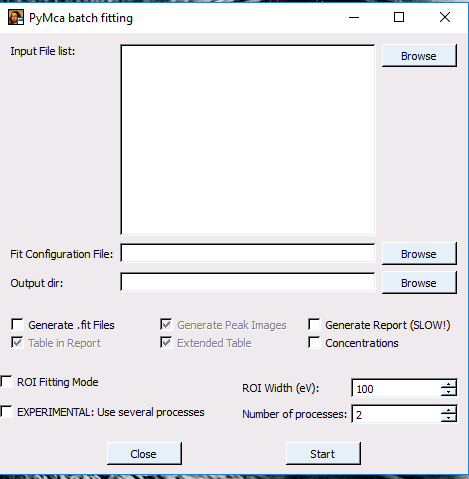


Figure 8: The batch fitting dialogue window

The Fit Configuration File to be used depends on whether or not Ca is being analyzed. The Fit Configuration Files are contained in a folder in the same directory as the python scripts. If you are analyzing **the non-Ca element peaks**, select the Fit Configuration called: **BatchConfigSetting.cfg**

If you are performing an analysis of the **Ca fluorescence peak**, select the Fit Configuration file entitled **BatchConfSetting-Ca-fitenergy.cfg**.

The output directory can be set to anything you like, though I recommend putting it in a separate directory called “PyMCA\_Files” or something similar, as a number of extra files not needed for our analysis are generated by the fitting process. **Ensure that the Generate .fit Files option is checked.** An example of a completed window is presented in Figure 9.

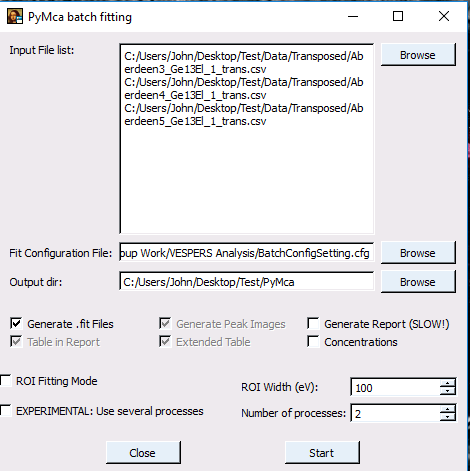


Figure 9: Completed batch fitting window. Ensure that the Generate .fit Files is clicked

Once you press start, the program will begin processing the images, and a window similar to the one in Figure 10 should pop up. Allow the process to finish, and close all the windows that appear when the fit is finished. We are now done using the PyMca program.

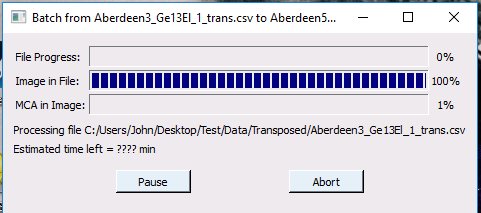


Figure 10: The batch processing status window.

Step Four: Analyze the Fit Results Using the Homebuilt Script

Now that the spectra have been fit, and the peak areas determined, it is time to turn this data into something useful. To do this, we will run the Homebuilt python scripts.

To start, open up a command prompt window (see Step 1), navigate to the folder containing the python scripts, and type the following:

python caller.py

Depending on how the software was last run, it may or may not ask you if you’d like to use the previously input parameters. If it does, you can say no by entering “n” or “N” to indicate that we would like to input our own parameters.

The program will ask for 5 parameters:

1. Folder containing the PyMca Fit files
2. The path to the csv file containing the header info (i.e., sample names, see Step 1 above)
3. The folder containing the Io files
4. The folder where you would like to save the results
5. If you are doing a Ca analysis or not (enter either Y or N)

After inputting the parameters, the program double check the parameters to ensure correctness. You can type Y to keep the parameters, or N to re-enter them if you made a mistake. A completed form is shown in Figure 11.

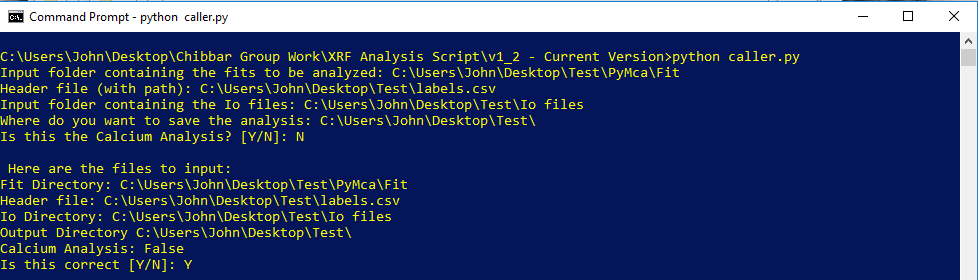


Figure 11: Entering the parameters for the program to run the files

Once you have entered the parameters and confirmed your entries, the program will then list 8 options for you to select from. You can choose an option by entering the number of the option you would like to select. An example screen is printed in Figure 12.

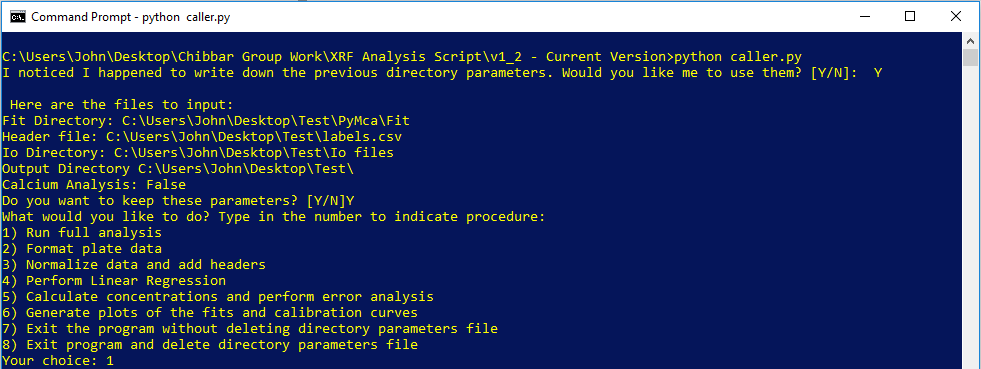


Figure 12: The “Home” screen of the program, where you can decide what action to perform

The possible options, with descriptions are as follows:

1. Run full analysis

This option combines running options 2 – 5 in one nice click. **Please note that you will still have to enter the calibration concentrations (or confirm using the previous calibration concentrations)**. This is the easiest option, and recommended for general use.

1. Format Plate Data

This module collects the peak area data generated by PyMca and combines the results into a single CSV file for each plate. The spectra and their fits are saved in csv files in a folder called “spectra” (found in the previously identified save directory) and the peak areas are tabulated in csv files saved in a folder called “peakfits”.

1. Normalize data and add headers

This module takes the peakfit files generated by the previous Format Plate Data module as inputs. It replaces the sample labels with those from the header file, adds the Io data from the Io files, and then normalizes the peak areas by dividing them by Io. The results of this analysis are saved as csv files (one per plate) in the folder called “Norm” in the save directory.

1. Perform Linear Regression

The Linear Regression module reads the data saved in the Norm file, and generates calibration curves for each element analyzed. While running this module the program will either ask for the concentrations for each calibration sample, or will recognize that you have previously entered calibration data.

The program will read all the calibration samples from each plate in the collection (i.e., from each plate in the Norm folder), and create a single calibration curve using this collected calibration data.

**Important: Remember that you cannot use the same concentrations for a Ca analysis as you do for the other elemental analyses!!!**

Figures 13 and 14 show possible options for entering calibration data. The concentrations here may not match what you used, so ensure you use the proper numbers. All concentrations are listed in PPM, though the program will work as long as your units are consistent.



Figure 13: Entering concentrations for different calibration samples

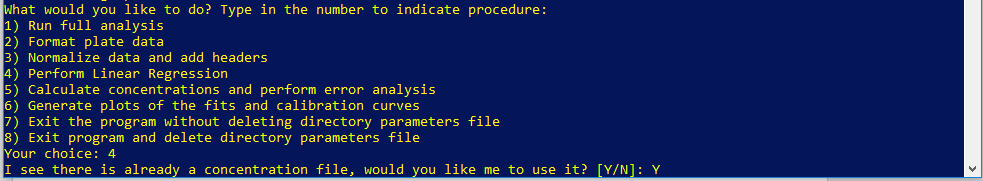


Figure 14: If you have already run the program, the program will remember the previously used calibration concentrations.

The program outputs two files, saved in the Calibration\_Curves folder. CalibrationCurvesOut.csv contains the linear regression information (i.e., slope, intercept, R2-value, etc) for each element, while CombinedCals.csv contains all of the peak areas from the calibration samples in one file. Both files are in the CSV format. The sample concentrations are also saved in the concentrations.npy file, in the numpy array format.

Also note that the program can remove up to one value from each calibration curve, if the point is greater than 5σ from the mean for that calibration concentration. There is currently no option to turn this feature off, but it will likely be implemented in future versions of the Homebuilt software.

1. Calculate concentrations and perform error analysis

This module does the final analysis and returns the crucial numbers: the final elemental concentrations with error bar. The module reads the info from the Linear Regression output, and as such, **the linear regression module must be run before the calculate concentrations module!** (In a new version of the program this will hopefully be eliminated, but for now you must run option 4 before running option 5).

The module will average the results of all materials having the same sample name on a given plate. **It is therefore necessary to ensure that all replicates of a given sample are contained on the same plate!**

The module returns the following files:

Analysis Results.csv, located in the specified save directory; contains the concentrations. If a value is below the Limit of Detection (LOD) or limit of quanitification (LOQ), it will be listed as such in this file.

Regression\_Stats.txt, located in the specified save directory; contains interesting statistics from the linear regression; used in the error analysis portion of the module.

1. Generate plots of the fits and calibration curves

Newly added! The plot generator will automatically plot the calibration curves and each spectra and fit as an individual file. To use, simply enter 6 as your choice (Figure 15). All of the spectra files are saved a folder called Spectra\_Fits and the calibration curve plots will be saved in the Calibration\_Curves folder. Please note that these are *not generally publication-quality plots*, and are meant mostly to be an easy way to visualize the data and fits. **Separate plots for publication should be made** if needed.

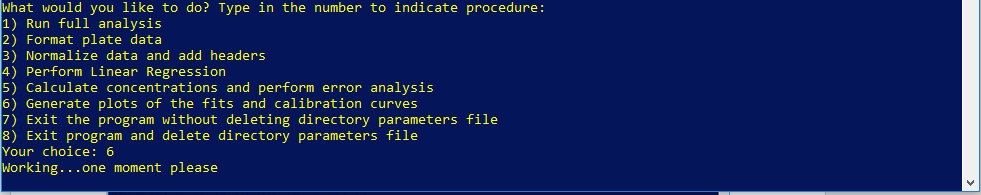


Figure 15: You can run the graphing program by entering 6. There will be a short delay between when the program is started and when the program starts saving files.

1. Exit the program without deleting directory parameters file

Use this option to exit the program while saving the directory parameters to a file for future use (i.e., fit location, header file location, etc.).

1. Exit program and delete directory parameters file

Use this option to exit the program without saving the directory parameters file.

Other quirks

The program can currently only deal with up to 10 replications of the same sample in a given plate, any more than that and it will crash/blow up. Sorry about that.