

XRDpy User Manual

Andrew Garcia

Updated: July 2020

Summary

What is XRDpy?	3
Compatibility	5
How do I start using it?	6
Clone the repository	6
Run a simple command in the Terminal	6
Make your database file	7
Set your common XRD file directories	7
Run XRDpy Example No. 1	8
Run XRDpy Example No. 2	10
XRD.py arguments	11
-h, -help	11
-p, -path_database_file	11
-p2, -path_files_folder	11
-d -see_database	11
-ka, -K_alpha_wavelength	11
-b, -background_sub	12
-o, -overlaid	12
-x, -overlaid_split	12
-s, -single	12
-u, -units	12
-r, -Scherrer_range	12
-K, -shape_factor_K	12
XRDsingle.py arguments	13
-h, -help	13
-p, -path	13
-s, -file_name	13
-ka, -K_alpha_wavelength	13
-se, -second_emission	13
-kb, -K_beta_wavelength	13

-b, -background_sub	13
-xl, -toexcel	14
-r, -Scherrer_range	14
-K, -shape_factor_K	14

What is XRDpy?

Github repository: [XRDpy](https://github.com/andrewrgarcia/XRDpy) (<https://github.com/andrewrgarcia/XRDpy>)

Needed: [Python](https://www.python.org/) (<https://www.python.org/>)

XRDpy is an XRD pattern plotting Python program which calculates crystallite size in an easy way. This program executes through the Terminal ('command line') and the execution is easy and straightforward. The program is currently divided into 2 main scripts, XRD.py and XRDsingle.py.

The first script, XRD.py, calculates the Scherrer widths for multiple XRD patterns and plots them in a single figure, where the order of the plots can be easily customized by the user through the Terminal.

The second script, XRDsingle.py, does the same but for a single XRD pattern, giving a single plot in a figure.

These scripts are currently separate of one another, but a single script integration has been entertained as it may facilitate ease of use. Nonetheless, would such be the case the functionality of the program is expected to remain the same or improve.

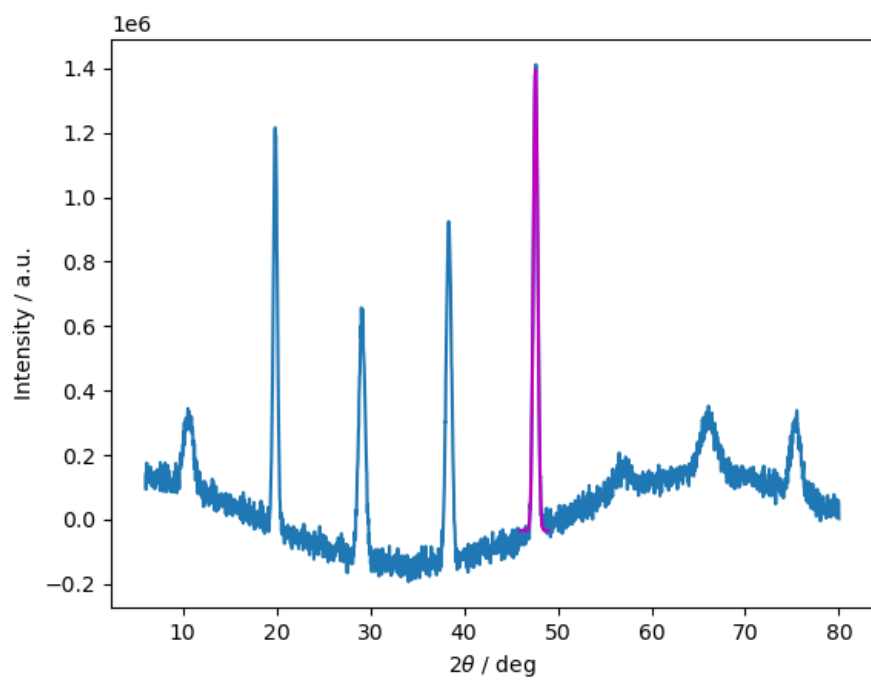


Figure 1: This is an output of the program (**python XRDsingle.py -s sample1.csv -b False -r 46 49** to be more exact).

Compatibility

XRDpy runs on Python, which technically makes it multiplatform. However, I have only tested it in **Windows** and **Linux (Ubuntu)** operating systems, and it runs wonderfully on both. One should not find many complications running it with **Apple** systems iOS either, though the path format (directories) to access your excel and csv files may need to be changed accordingly.

How do I start using it?

Taking a pragmatic approach you may find the use of XRDpy to be very straightforward. You're encouraged to watch the instructional video of XRDpy in YouTube by **clicking here** .

Clone the repository

Github repository: [XRDpy](https://github.com/andrewgarcia/XRDpy) (https://github.com/andrewgarcia/XRDpy)

Needed: [Git](https://git-scm.com) (git-scm.com)

Optional: [Github account](https://github.com) (github.com)

You may need to learn the basics of Git, which aren't complicated to do. Creating a Github account will allow you to fork my repository and share any changes you may make of your forked version(s).

In the main XRDpy Github repository, there should be a button called "Code" or clone. Click on it and copy the HTTP address

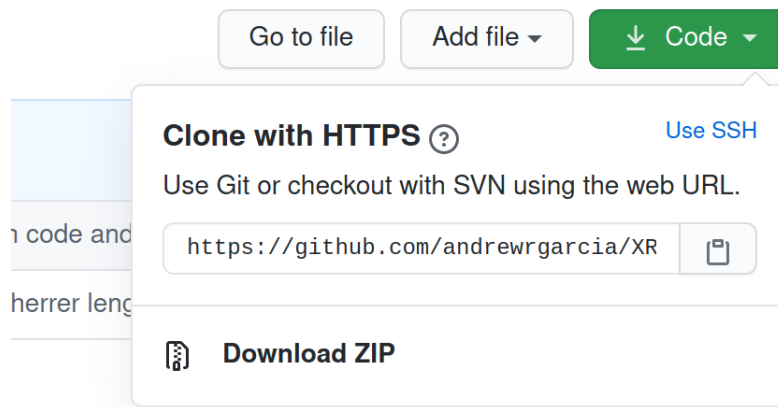


Figure 2: it looks like this

Open your Terminal and select the folder or path where you want to place the XRDpy program (i.e. use cd command).

In the Terminal, type: **git clone [pasted HTTP address]**

Run a simple command in the Terminal

Open your Terminal (or command line, or shell, whatever you call it)

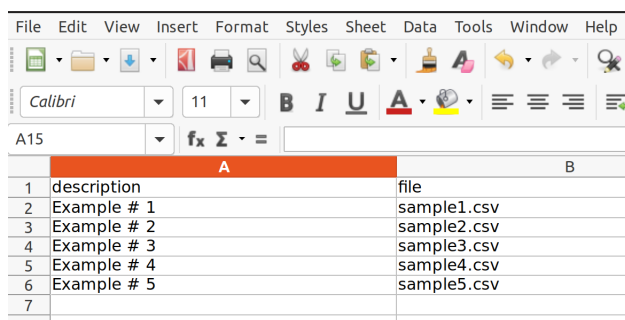
Change to the folder containing your cloned version of XRDpy (Use the cd command to get there)

While in that folder, type: **python XRD.py -h**. This should bring up the lists of all the arguments available to customize and make your plot(s).

```
~/scripts/XRDpy$ python XRD.py -h
```

Make your database file

Your database file should contain the names of all your files with their .csv extension (oh yes, they should be converted or be in csv format) in the right column, as well as a short nickname in the left column that will be used to call them by the XRDpy program. Please use the **database.template.xlsx** file provided in your cloned XRDpy folder. It makes things easier.



	A	B
1	description	file
2	Example # 1	sample1.csv
3	Example # 2	sample2.csv
4	Example # 3	sample3.csv
5	Example # 4	sample4.csv
6	Example # 5	sample5.csv
7		
8		

Set your common XRD file directories

```
21 ap.add_argument("-p", "--path_database_file", \
22                 default = '/home/andrew/XRD/database-template.xlsx', \
23                 type = str, help="Path and filename of Excel database which is used\
24                 to call all your files. Please update with your path and databse file name.")
25 ap.add_argument("-p2", "--path_files_folder", \
26                 default = '/home/andrew/scripts/XRDpy/XRD-patterns-fake/', \
27                 type = str, help="path to FOLDER containing XRD files listed in Excel database")
```

Open **XRD.py** and **XRDsingle.py** in a Python interpreter or text editor and change the paths where you are housing your database (see above) and your XRD files. Though you can call to switch the path in the Terminal by python XRD.py -p [**your-pathhere/.../...**] it is more efficient to have your permanent paths as defaults.

The argument IDs should be -p and -p2 for XRD.py and -p for XRD.single.py

For an interesting demonstration, leave the **database_template.xlsx** file unchanged and set the -p2 path to the folder named **XRD-patterns-fake** inside your XRDpy folder

Run XRDpy Example No. 1

Leaving the **database_template.xlsx** file unchanged and having set the -p2 path to the **XRD-patterns-fake** folder, type the following commands in your terminal (with the Terminal path set to the XRDpy folder):

python XRD.py

This will display your database in the Terminal; it is an alternative to **python XRD.py -d True**. You will use the indexes from that list display and not the descriptions (i.e. use the numbers 0 1 2 3 4 in this example)

```
?
if args["overlaid"] is -1:

    description
0 Example # 1
1 Example # 2
2 Example # 3
3 Example # 4
4 Example # 5

Traceback (most recent call last):
  File "XRD.py", line 314, in <
```

Figure 3: This is what you see. The list of the **database_template.xlsx** file is displayed along with some silly syntax error warnings which do no harm to the development and are not even necessary in my opinion, but that's just me.


```
python XRD.py -o 0 1 2 3 4 -x 3 2 -s 2
```

The above command will give you this plot:

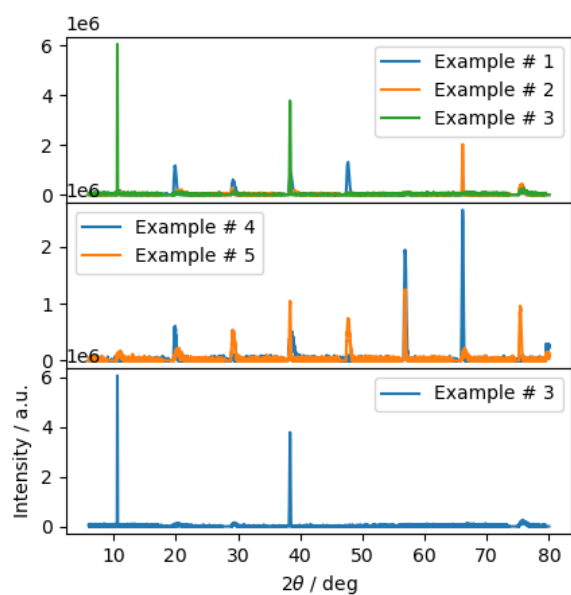


Figure 4: **XRDpy Example No. 1** -o stands for overlaid with -x being the split of the declared overlaid plots and -s being the single plot

Run XRDpy Example No. 2

Type this in your Terminal:

python XRDpy -o 0 2 4 3 -x 4 -s 0 1 4 -r 36 41 -b False

The command plots the XRD patterns and calculates the crystallite size (Scherrer Width) for the single plots (those coming after -s); coloring the 2- θ range chosen to calculate it (here -r 36 41) in **purple**.

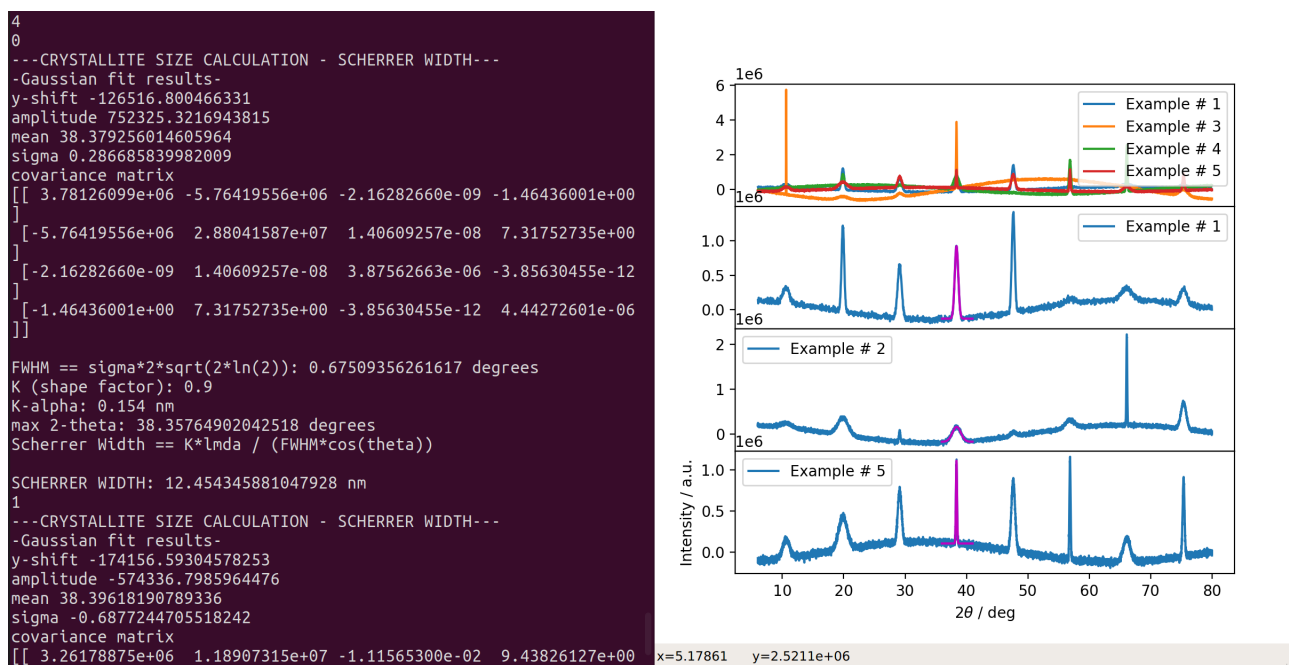


Figure 5: **XRDpy Example No. 2** -r calculates the crystallite size using the Scherrer equation at the range specified, in this case, in the range of 36 to 41 2- θ ; -b subtracts the background of all XRD patterns (it is set to True by default, so here I set it to False to show how the plots would look without the auto background subtraction)

XRD.py arguments

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam lobortis facilisis. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante...

-h, --help

Typing **python XRD.py -h** in your command prompt Terminal will give you the complete list of arguments you can run to process your XRD pattern [all the ones you see here below] .

-p, --path_database_file

python XRD.py -p r'C:\Users\...\database.xlsx' (Windows)
python XRD.py -p '/home/user-name/.../database.xlsx' (Ubuntu)

This specifies the address and file name of your excel database of XRD patterns, where the second column displays the file names of your XRD patterns (which should be changed to .csv extensions) and the first column can be a short name you use to easily call the .csv file (see database_template.xlsx) For easy execution, please update the default address to your address with your excel file on the Python script and save it (line 23 XRD.py)

-p2, --path_files_folder

python XRD.py -p2 r'C:\Users\...\XRD-files' (Windows)
python XRD.py -p2 '/home/user-name/.../XRD-files/' (Ubuntu)

This specifies the path of your folder housing all your XRD patterns which should be mentioned in your Excel database. For easy execution, please update the default address to your address with your excel file on the Python script and save it (line 27 XRD.py)

-d --see_database

Boolean which, when set to True, displays every common name of the XRD patterns written in your Excel database. It is defaulted to False; if you ever want to look at the contents of your database before running the plotting script, type the above underlined command.

-ka, --K_alpha_wavelength

This is the wavelength of K- α X-ray radiation, here written in units of nanometers (nm). It is defaulted to that of Copper K- α which corresponds to an X-ray

wavelength of 0.154 nm

-b, -background_sub

This is an option to perform a baseline correction on the XRD pattern. Default is set to True, so it does it automatically. Setting it to False gives the original, uncorrected XRD pattern.

-o, -overlaid

Here is where you list all the plots you want to overlay in your figure (see Examples 1 and 2)

-x, -overlaid_split

The overlaid "split". dictates how you want to partition the overlaid plots into different sets (see Examples 1 and 2)

-s, -single

List of plots you want to place separated in single charts in your figure (see Examples 1 and 2)

-u, -units

The units of your x-axis. Default is set to units of 2θ (**-u angle**). Units of interplanar spacing in nm can be set by the command **-u braggs**

-r, -Scherrer_range

Calculates crystallite size using the Scherrer equation with the specified 2θ range input after -r (e.g. **-r 20 40** would be range from 20 to 40 2θ)

-K, -shape_factor_K

This specifies the shape factor or "K" from the Scherrer equation. Default is set to $K = 0.9$

XRDsingle.py arguments

-h, -help

Typing **python XRD.py -h** in your command prompt Terminal will give you the complete list of arguments you can run to process your XRD pattern [all the ones you see here below] .

-p, -path

python XRD.py -p2 r'C:\Users\.../' (Windows)
python XRD.py -p2 '/home/user-name/.../' (Ubuntu)

For **XRDsingle.py**, -p specifies the path of your folder where your chosen XRD file is.

-s, -file_name

The actual name of your chosen XRD file with its .csv extension in the end (e.g. sample-file.csv)

-ka, -K_alpha_wavelength

This is the wavelength of K- α X-ray radiation, here written in units of nanometers (nm). It is defaulted to that of Copper K- α which corresponds to an X-ray wavelength of 0.154 nm

-se, -second_emission

In XRD machinery with no secondary emission filters, extra peaks coming from such may show up. Setting this option -se True subtracts these secondary emissions with wavelength specified by -kb (see next argument)

-kb, -K_beta_wavelength

Wavelength of secondary emission to be subtracted. Default set to wavelength of K- β radiation, 0.139 nm.

-b, -background_sub

This is an option to perform a baseline correction on the XRD pattern. Default is set to True, so it does it automatically. Setting it to False gives the original, uncorrected XRD pattern.

-xl, -toexcel

Currently discontinued (obsolete). Set to False.

-r, -Scherrer_range

Calculates crystallite size using the Scherrer equation with the specified $2-\theta$ range input after -r (e.g. **-r 20 40** would be range from 20 to 40 $2-\theta$)

-K, -shape_factor_K

This specifies the shape factor or "K" from the Scherrer equation. Default is set to $K = 0.9$