Instruction for PyAbel GUIs

Jan.16th, 2025
Haohui Che
Loomis Group

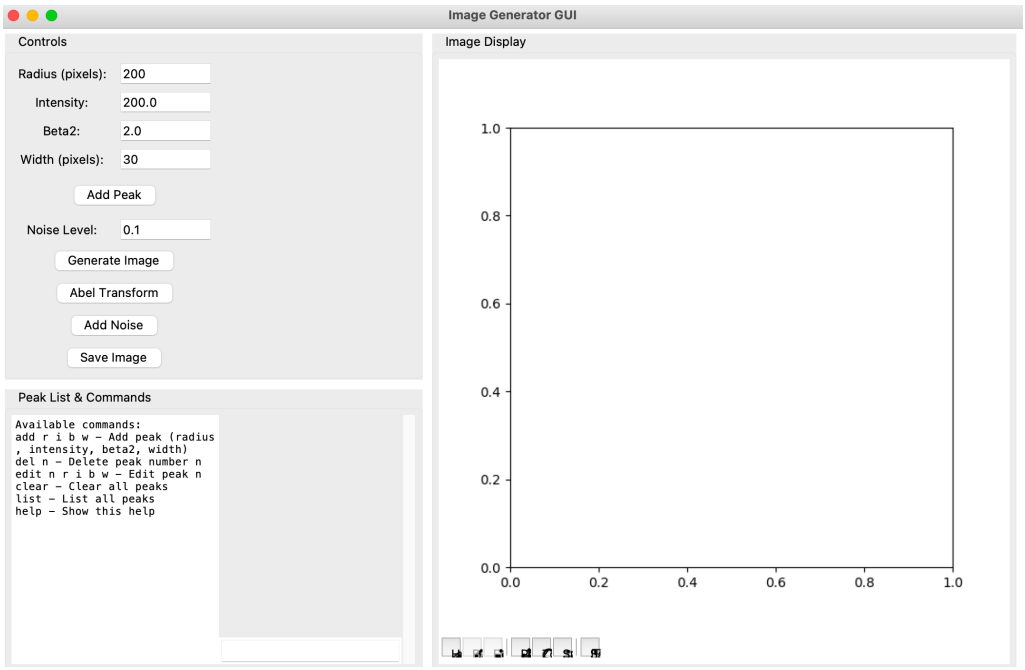
To set up the program:

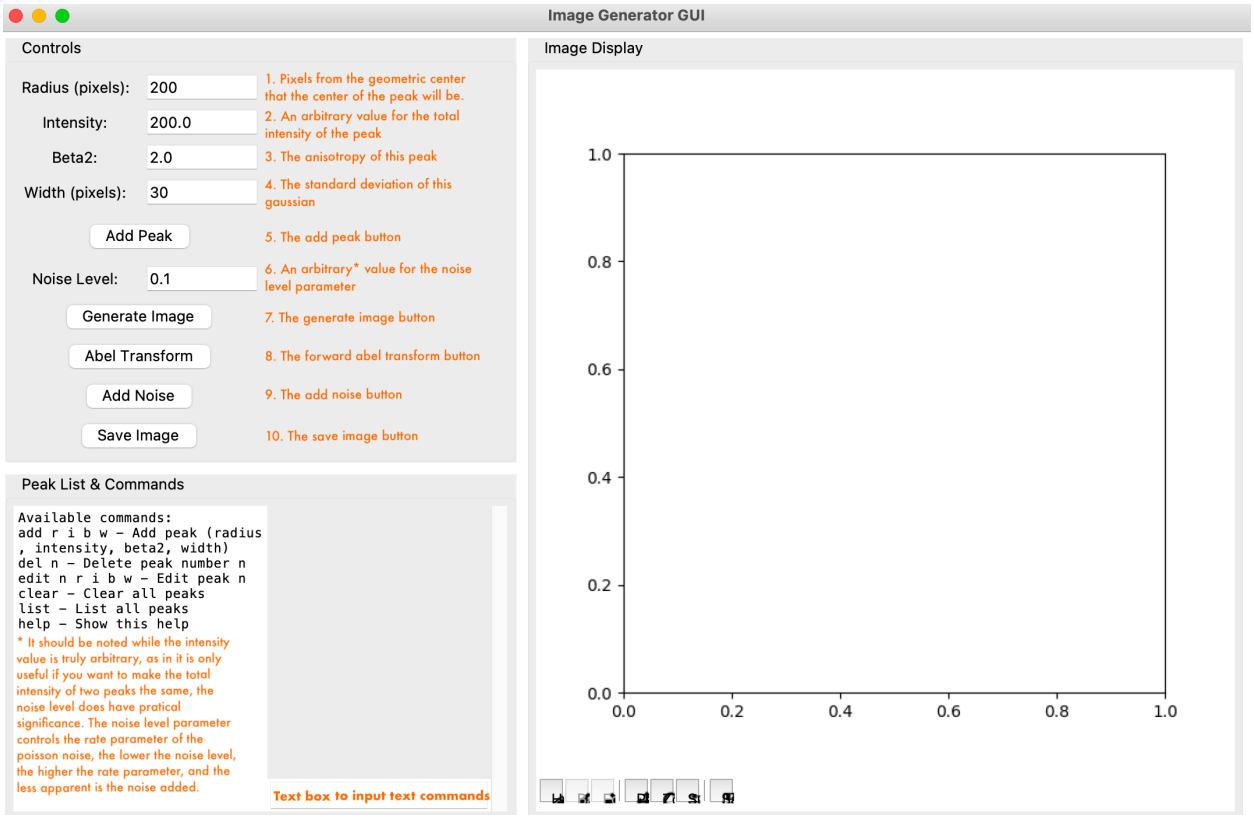
1. Use whatever compiler you wish to use and open the PyAbel folder (not just the .py script)
2. From that, open
   Supp_ImageGen_GUI.py
   or
   Mast_GUI_with_Anora_with_calib_fact.py
3. Open the terminal tab in your compiler, enter
   `pip install matplotlib`
   then press the return key, wait for pip to do its thing, and when it shows install successful, enter
   `pip install numpy`
   then press the return key, wait for pip to do its thing, and when it shows install successful, enter
   `pip install scipy`
   then press the return key, wait for pip to do its thing, and when it shows install successful, enter
   `pip install abel`
4. The click run, wait a few moments, and a window should pop up

To use it:

A. For the Image Simulation Program:

1. Open Supp_ImageGen_GUI.py and click run, a window should pop up





2. For a normal operation, each button should be clicked from top to bottom. That is, 1. program all the peaks you wish to add; 2. generate the clean image and check if it is what you want; if not, go back to step 1; 3. perform a forward abel operation to make the testable image; 4. program and add noise to the transformed image; 5. save the image(s)
3. For easier peak manipulation, the text box at the bottom can be used to program the peaks.
    a. To add peak: `add r i b w`
    For example: to add a peak **150** pixels away from the center, with an intensity of **500**, an anisotropy of **-2**, and a width of **15**, the command should be:
    `add 150 500 -2 15`
    b. To delete a peak: `del n`
    For example: here are four peaks
    Peak 1:
    R: 150.0
    I: 200.0
    B2: -1.0
    W: 10.0

    Peak 2:
    R: 200.0
    I: 200.0
    ß2: 1.0
    W: 10.0

    Peak 3:

R: 250.0
I: 200.0
B2: 2.0
W: 15.0

Peak 4:
R: 300.0
I: 200.0
B2: 0.0
W: 5.0
And you wish to delete peak 3, then enter:

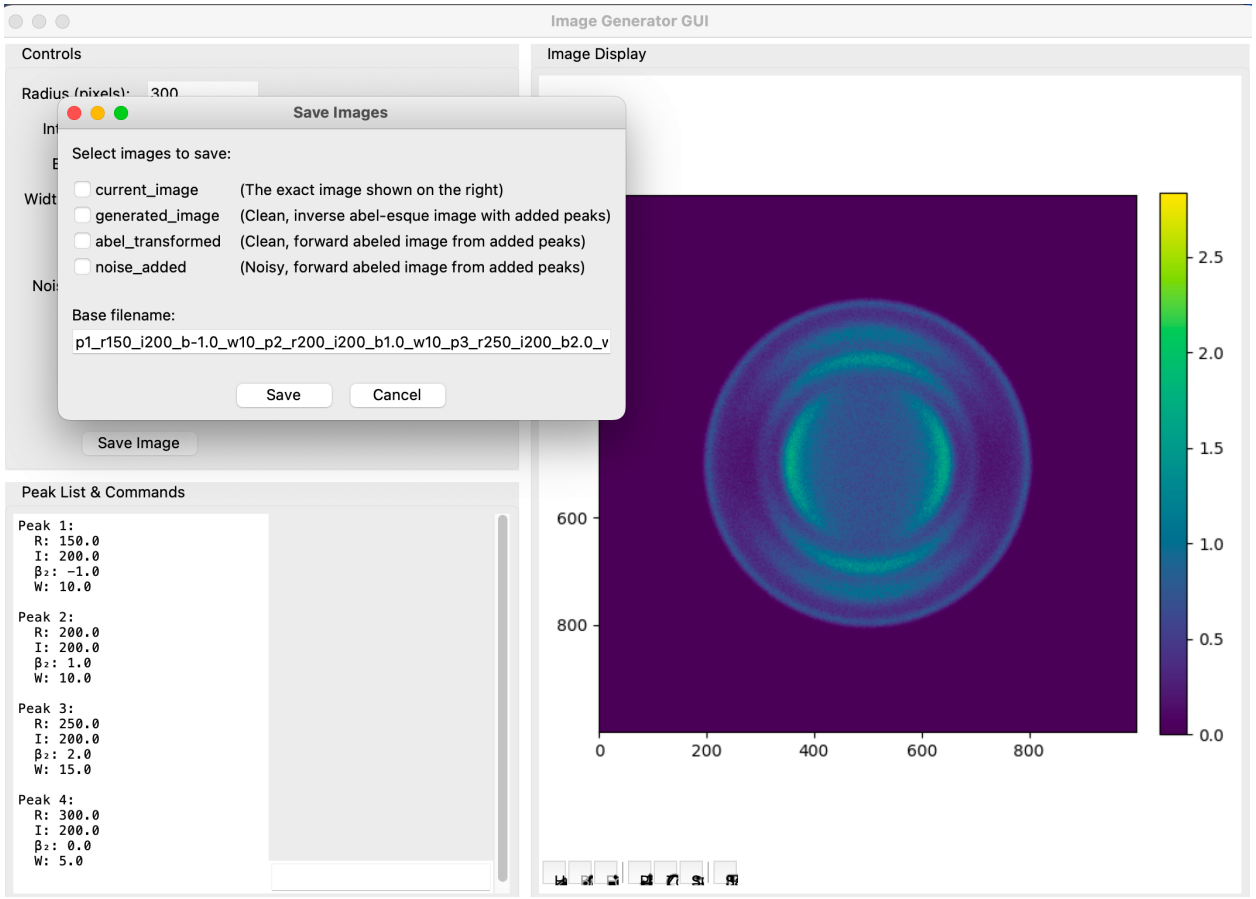
```
del 3
```

c. To edit a peak: `edit n r i b w`
For example: still the four-peak case, if you wish to edit peak **2** from **200**, **200**, **1**,**10** to **50**, **150**, **-0.5**, **20**, then enter:

```
Edit 2 50 150 -0.5 20
```

d. To clear all peaks: `clear`
e. To list all peaks: `list`
f. To show the help page: `help`

4. To save, click the save button, and there will be a pop-up menu

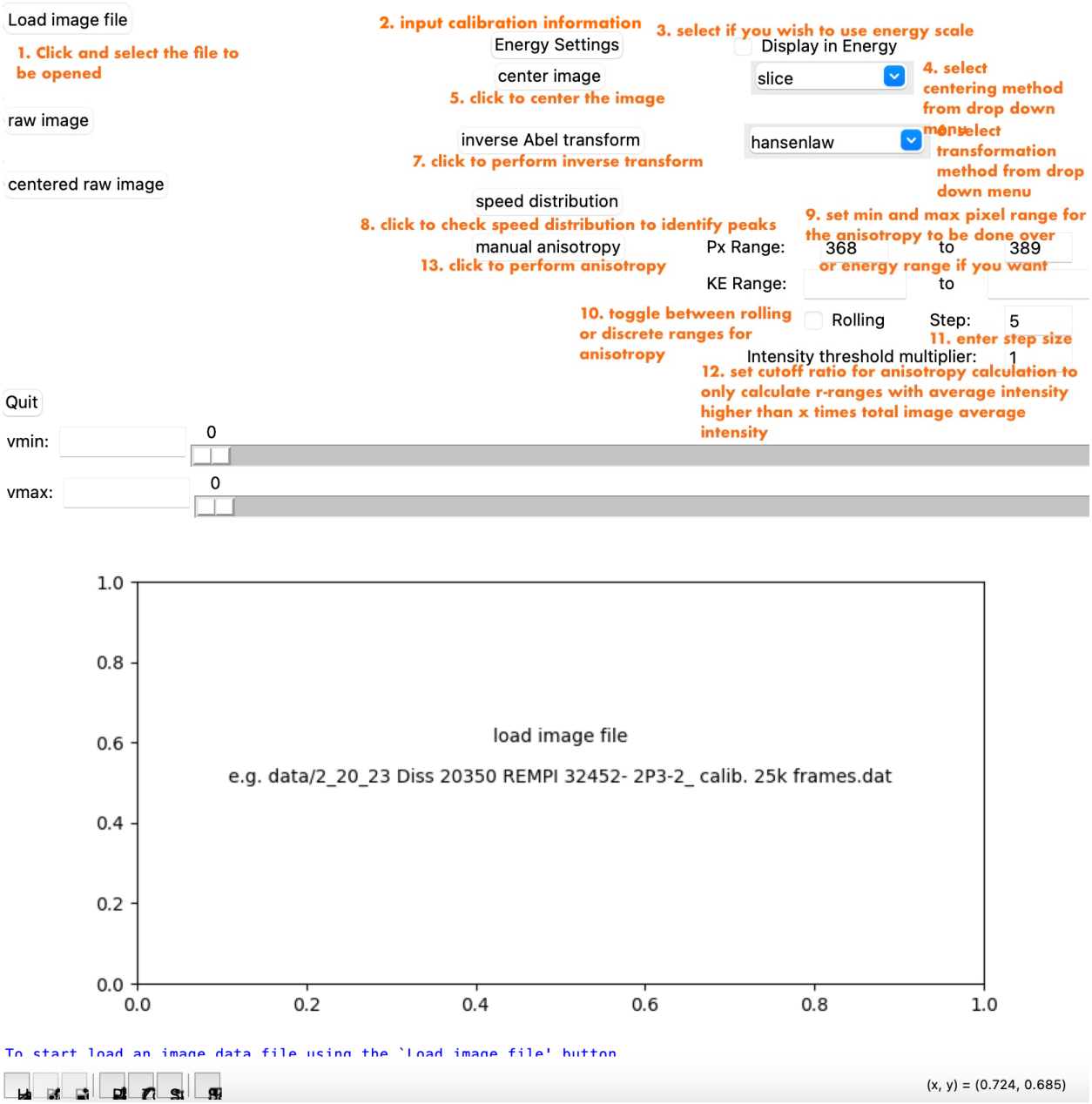


Each toggle represent different images to be saved, select the images you wish to save, and change the base file name at your will (it will be the file name of the current_image if the toggle

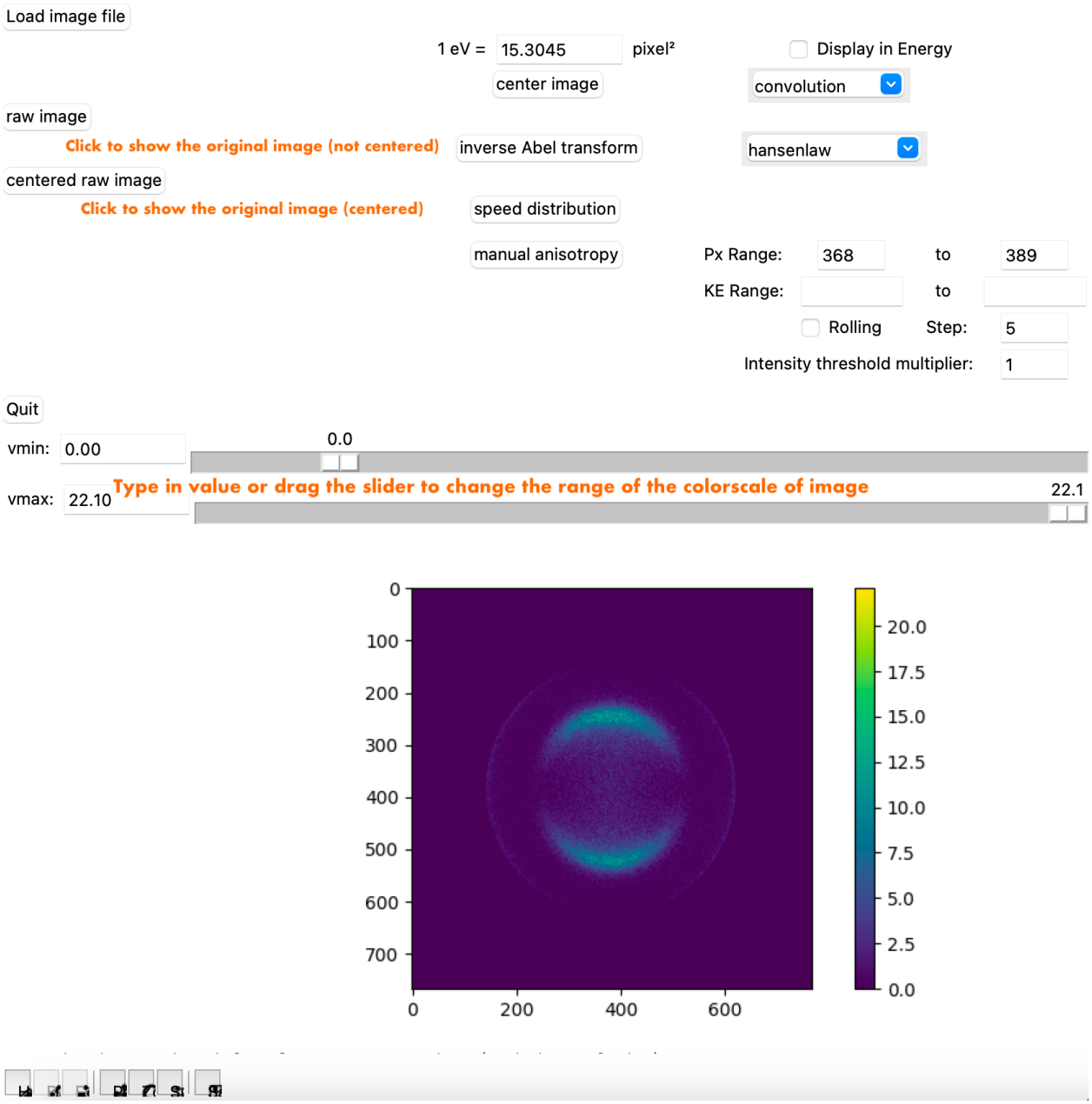
is selected) the naming convention of the default name is “peak#+radius#+intensity value+anisotropy_beta+standard_deviation”, repeated for each peak.

B. For Master GUI

1. Open Mast_GUI_with_Anora_with_calib_fact.py and click run, a window should pop up



2. Above is the basic workflow of using the program. Follow the steps from 1 to 13, and there should be a popup window asking if you wish to save the anisotropy data. In the save file, there will be information on image manipulation methods used, anisotropy at each range, and their average intensity.
3. To input calibration information, prepare two values: calculated Px to momentum (in kg*m/s) coefficient, and the mass of the ion fragment (in amu).
   Click Energy Settings button, and input the values.
4. For better visualization of files, two buttons and two sliders are included to facilitate the display of images while not altering the actual file. When you load an image, the window should look something like this:



Parameters to change for accommodating different environments

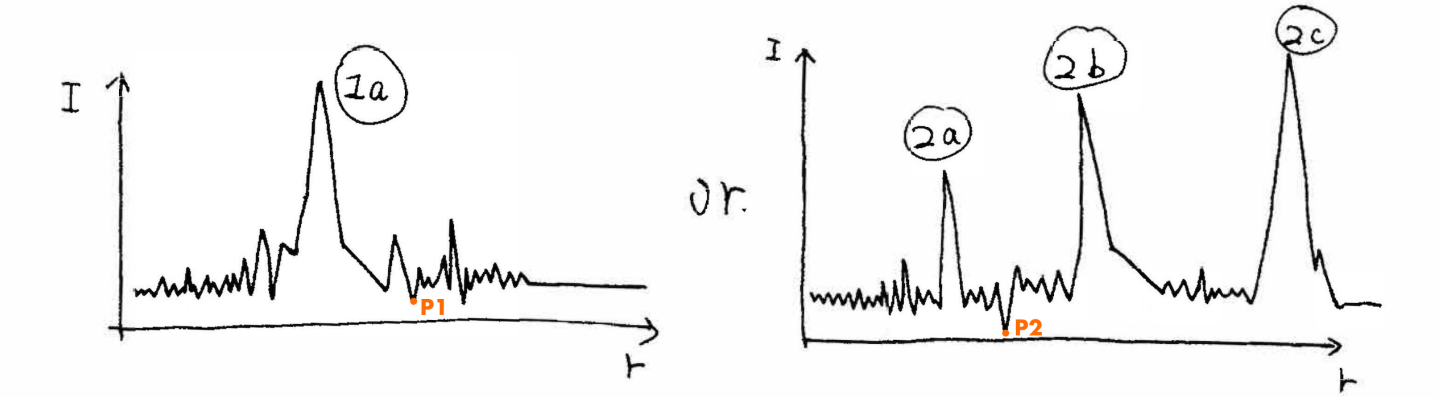
1. **Peak finding parameters:**
   In the speed code block of the Master GUI, the main peak finding command is this:

```
peak_ranges = find_peak_ranges(radial, speed_distribution,threshold=1,
                               rel_height=0.5, min_width=5, prominence_min=0.1)
```

   `radial`: the x-axis coordinates over which to find peaks.
   `speed_distribution`: the y-axis coordinates where the peaks are manifested.

   `threshold`: the minimum height the peak has to be for it to be detected (normalized to the highest value in the speed distribution).

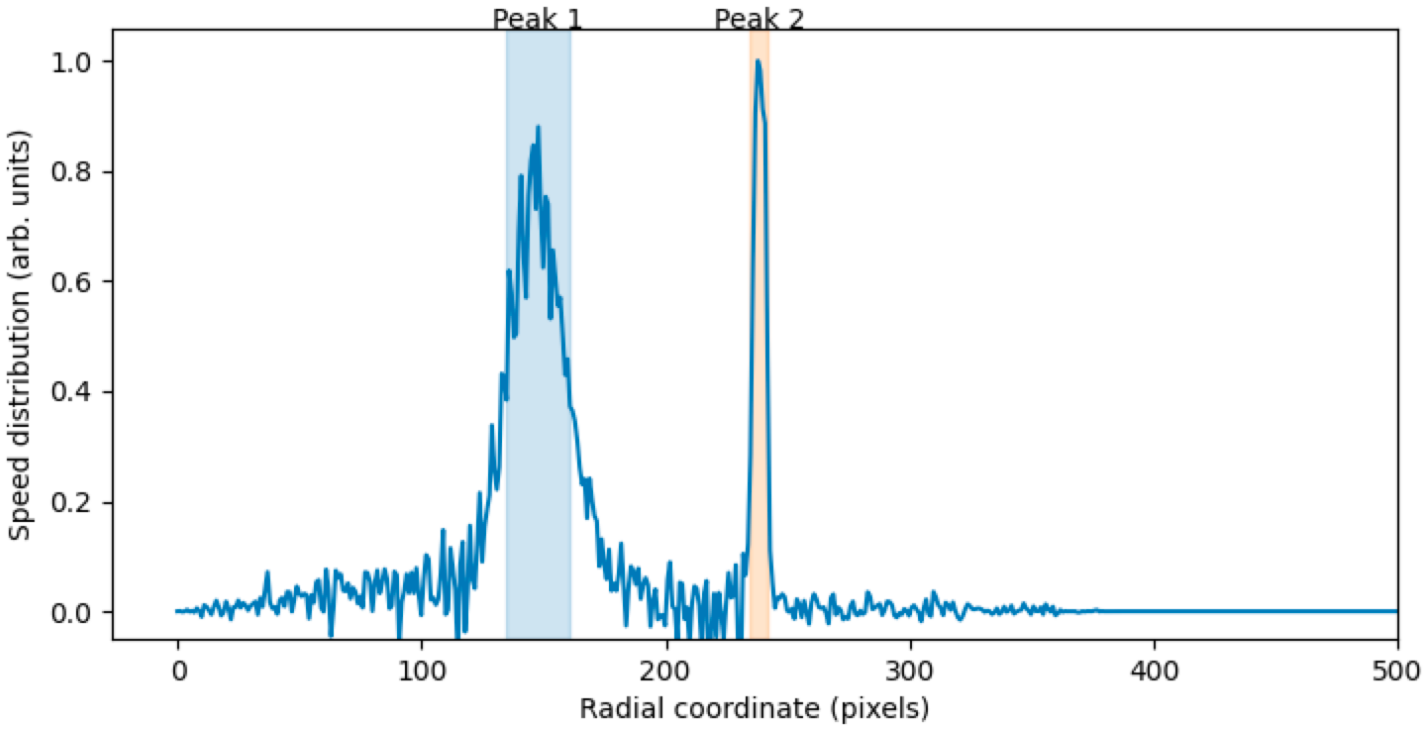
`rel_height:` a parameter that works in conjunction with the prominence of the peak (not `prominence_min`) to find the left and right bounds of the peak. ie. In the distribution there is a peak with peak value of 0.8 (normalized to the highest peak), and we find this peak has a prominence of 0.6 (normalized to the highest peak). To find the midpoint of the peak, we will need to define at which intensity do we set its left and right boundaries to be. With rel_height parameter, we can modify this threshold by multiplying the prominence with this multiplier. This is useful in situations where the peak looks like this:



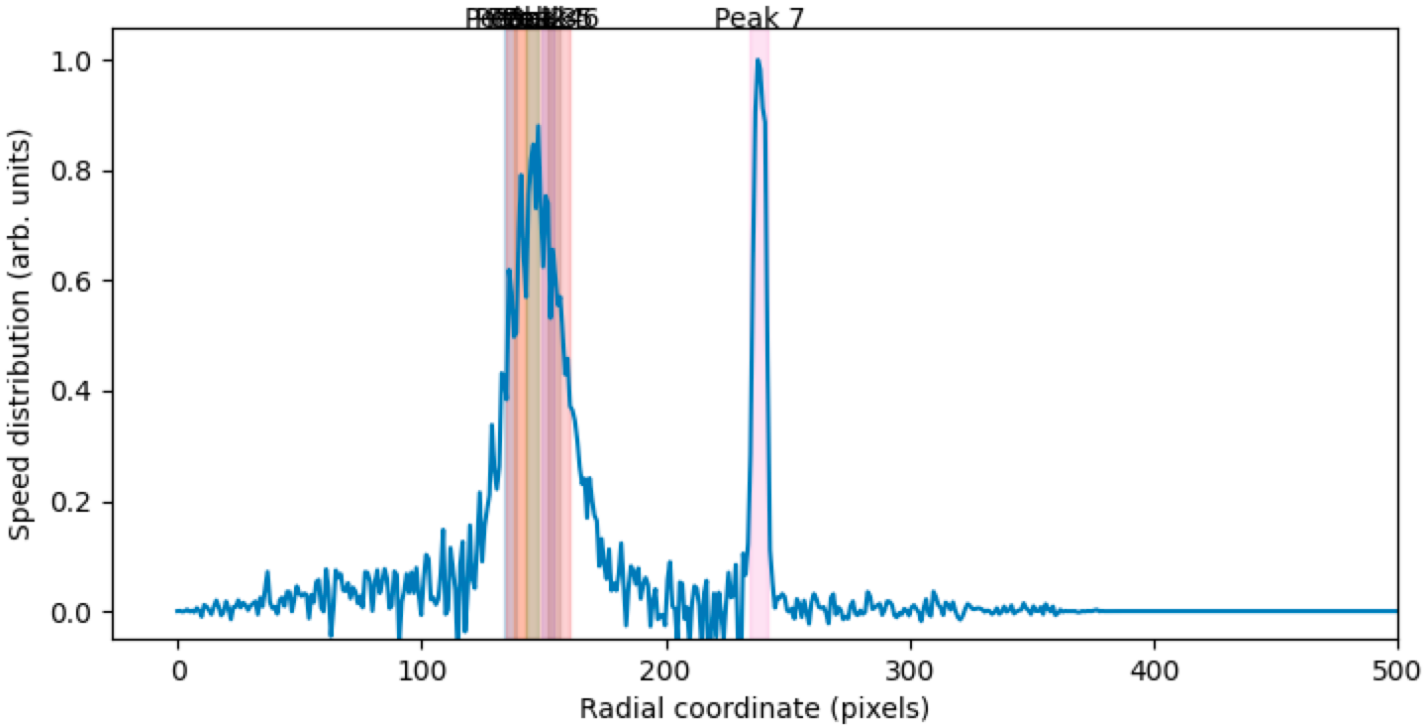
When peaks have a very asymmetrical base (peak 1a and 2b) or when the prominence of the peak is outside of the peak (peak 2a). In these two cases, modify the rel_height parameter to make the left and right boundaries to be at the correct y value.

`min_width:` The minimum width (distance between the left and right boundary) of the peaks.

`prominence_min:` The minimum prominence of peaks to be selected. Prominence measures how low the y-value has to go down from the peak before it runs into another peak higher than the said peak. Using the previous example, the prominence of the 1a peak would be from the max value on the peak to P1 point, and the prominence for 2a would be from its max to P2. This setting is particularly useful in situations where there are rough distributions with a major peak surrounded with many smaller peaks going down, such as this:



In this picture, we are using a minimum prominence of 0.5, meaning a peak must be the tallest peak in its surrounding such that one has to descend at least 0.5 units before encountering another higher peak. If we chose a minimum prominence of 0.1, this would happen:



We can see there are many peaks being identified at the broad region, which is definitely not what we want.

2. **Anisotropy parameters:**

   For anisotropy, important parameters are separated into two code files, one in Mast_GUI_with_Anora_with_calib_fact.py, others in anni.py (in the called_functions folder).

   a. **Mast_GUI parameters**

      In this category of parameters, we do not need to change the source code, rather, the GUI provide buttons and textboxes for parameter input. In a normal anisotropy calculation, the most important item to specify is the r ranges over which the program should calculate the beta 2 parameter. Therefore, on the GUI, this block can be seen:

      Px Range: The ENTIRE r range over which calculations should be done. With embedded range list generators, we do not need to specify every range over which the calculation should be done, instead, input the smallest and the largest pixel distance we wish the calculation should take place is enough. Also, the range does not need to be precise, as the intensity threshold multiplier would further filter useful range lists for calculations. (This range in interconnected with KE Range)

      KE Range: The ENTIRE Energy range over which calculations should be done. Basically the same as Px Range, for people who like to do image manipulation in energy space for some reason. (This range in interconnected with Px Range)

      Rolling: This is a toggle between rolling and non-rolling r ranges. For rolling, the r ranges overlap with each other, ie. [0,6] [1,7] [2,8] [3,9] [4,10] [5,11], so we can use larger step size to account for noisy data while still getting enough data points. For non-rolling, the ranges are not overlapping, ie. [0,3] [3,6] [6,9] [9,12], suitable for thin and clear distributions.

      Step: The step size for generating r ranges. Step size CAN be zero.

      Intensity threshold multiplier (ITM): This parameter governs the final validity check of r ranges before anisotropy calculations. In order to decrease unproductive ranges when doing the calculation, the program will calculate the entire image mean intensity (AIMI), and the r range mean intensity (RMI), then it will compare RMI to the product of AIMI * ITM, proceeding to anisotropy calculations only when RMI is larger than AIMI*ITM. Therefore, by changing ITM, we can finetune how aggressive do we want to filter out the “empty” r ranges.

   b. **anni.py parameters**

      The only parameter that can be changed, and possibly the most important for our purpose, is the bounds for the β2 parameter in this line:

```
# Bounds for the parameters: A > 0, beta2 between -1 and 2
bounds = ([0.0, -1.0], [np.inf, 2.0])
```

To change the bounds for β2, change the orange number for the lower bound, and green number for the upper bound.