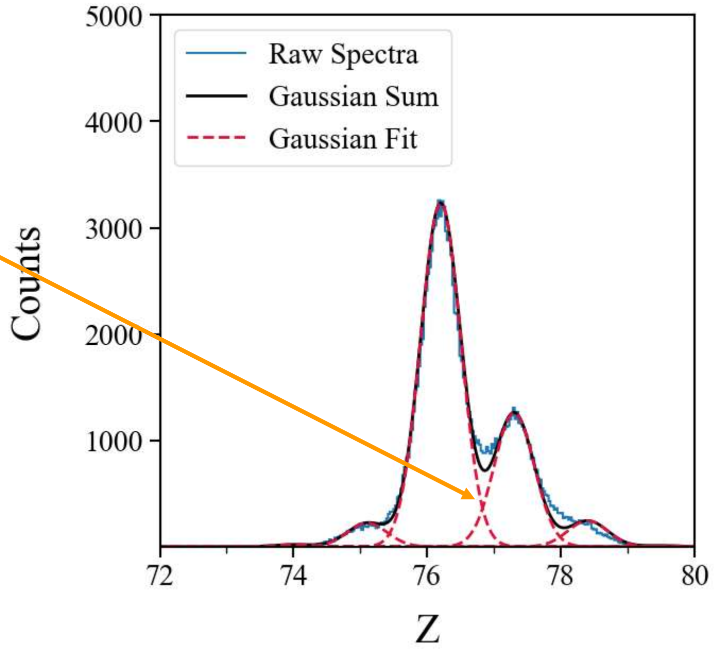
Peak Fitter

Kenny Haak

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Nuclear Physics and used resources of the Facility for Rare Isotope Beams (FRIB) Operations, which is a DOE Office of Science User Facility under Award Number DE-SC0023633, and by the US National Science Foundation under Grants No. PHY-20-12040 and 23-10078 “Windows on the Universe: Open Quantum Systems in Atomic Nuclei at FRIB”.

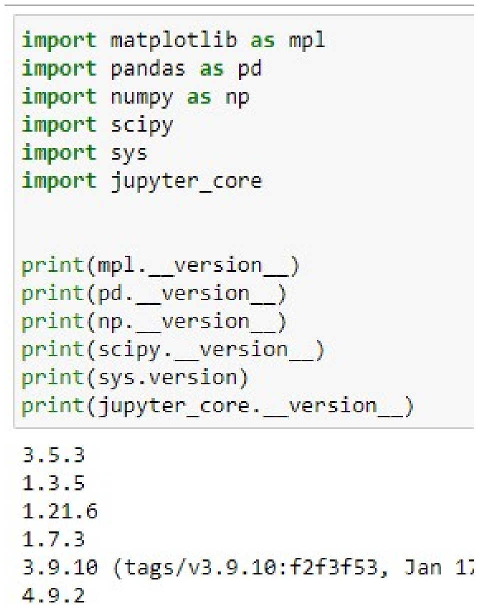
Foreword

- This code fits one-dimensional histograms with a series of Gaussian functions
- The value of this routine is to accurately events corresponding to a given PID (regularly drawn contours will fail to disentangle the overlapping tails of the distributions)
- What makes this fitting function special is not that it has unlimited flexibility to change the parameters of each gaussian, but in fact there are restraints that correspond to real physical factors
  - One fixed width – This means that there is a single width for all peaks (they all share the same underlying physical resolution)
  - Integer centers – Identified peaks correspond to isotopes which have a whole integer number of protons, neutron, and electrons
  - Constant (or Linear) spacing – Ideally your identified spectra should be well calibrated, but if it still needs a final shift and slope adjustment, this routine will calibrate the 1-D axis for you



Initialize

- Make sure you have the same versions

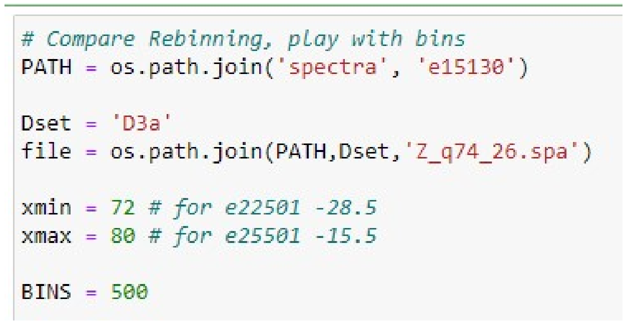


```
import matplotlib as mpl
import pandas as pd
import numpy as np
import scipy
import sys
import jupyter_core


print(mpl.__version__)
print(pd.__version__)
print(np.__version__)
print(scipy.__version__)
print(sys.version)
print(jupyter_core.__version__)
```

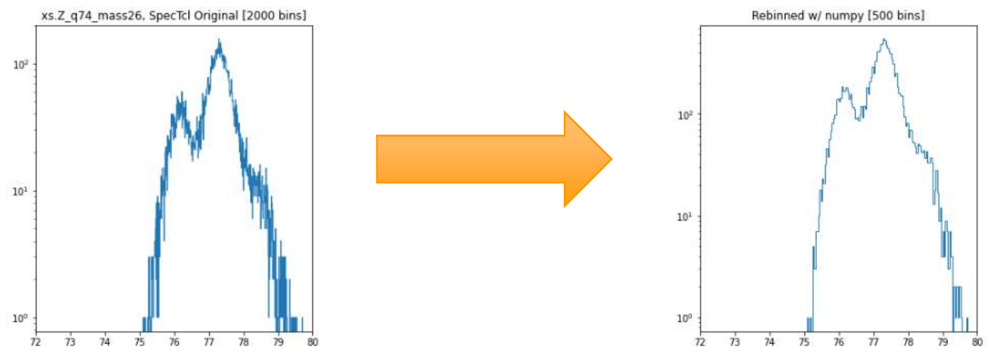
```
3.5.3
1.3.5
1.21.6
1.7.3
3.9.10 (tags/v3.9.10:f2f3f53, Jan 17
4.9.2
```

Optimal Binning

- The first actual cell allows you to play around with different bin widths
- For this to be useful, your original spectra should be exported from SpecTk with the finest possible resolution
- To run, simply set the file path and then change the BINS parameter
- You will notice there are two plots at the bottom, this code simply shows you the default binning in which you exported from SpecTk and your new binning
- There is an analytical function for optimal binning that depends on the counts and width of your peaks… Ask Isaiah

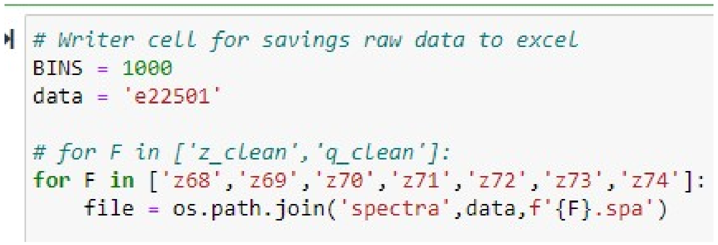


```
# Compare Rebinning, play with bins
PATH = os.path.join('spectra', 'e15130')

Dset = 'D3a'
file = os.path.join(PATH,Dset,'Z_q74_26.spa')

xmin = 72 # for e22501 -28.5
xmax = 80 # for e25501 -15.5

BINS = 500
```

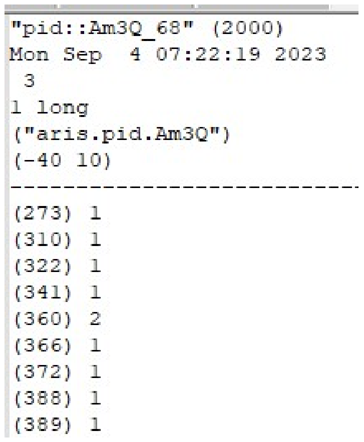


Binning 2

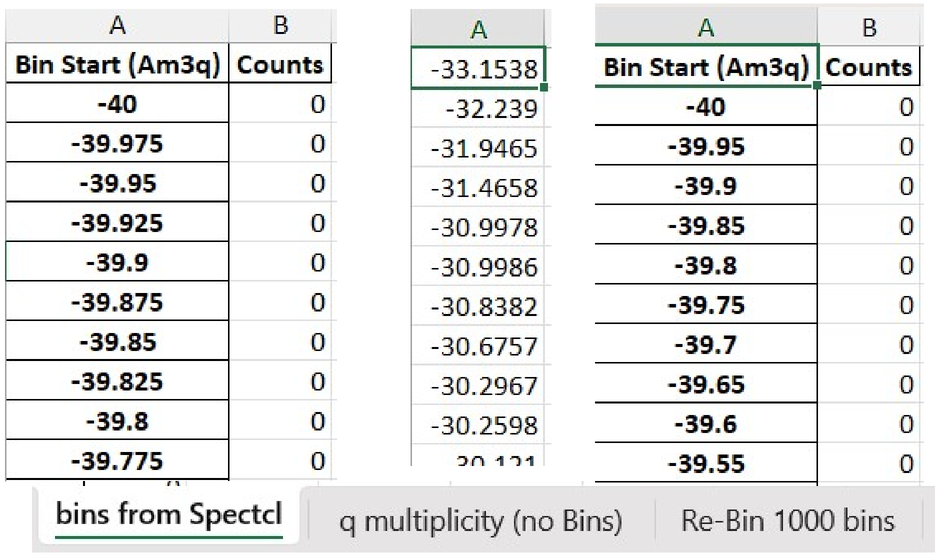
- The next cell allows you to change binning to a spa file and have it automatically resaved in xlsx format



```
# Writer cell for savings raw data to excel
BINS = 1000
data = 'e22501'

# for F in ['z_clean','q_clean']:
for F in ['z68','z69','z70','z71','z72','z73','z74']:
    file = os.path.join('spectra',data,f'{F}.spa')
```



```
"pid::Am3Q_68" (2000)
Mon Sep  4 07:22:19 2023
 3
1 long
("aris.pid.Am3Q")
(-40 10)
----------------------------
(273) 1
(310) 1
(322) 1
(341) 1
(360) 2
(366) 1
(372) 1
(388) 1
(389) 1
```



A	B
Bin Start (Am3q)	Counts
-40	0
-39.975	0
-39.95	0
-39.925	0
-39.9	0
-39.875	0
-39.85	0
-39.825	0
-39.8	0
-39.775	0

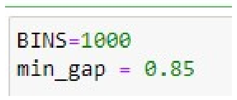
A
-33.1538
-32.239
-31.9465
-31.4658
-30.9978
-30.9986
-30.8382
-30.6757
-30.2967
-30.2598

A	B
Bin Start (Am3q)	Counts
-40	0
-39.95	0
-39.9	0
-39.85	0
-39.8	0
-39.75	0
-39.7	0
-39.65	0
-39.6	0
-39.55	0

bins from Spectcl | q multiplicity (no Bins) | Re-Bin 1000 bins

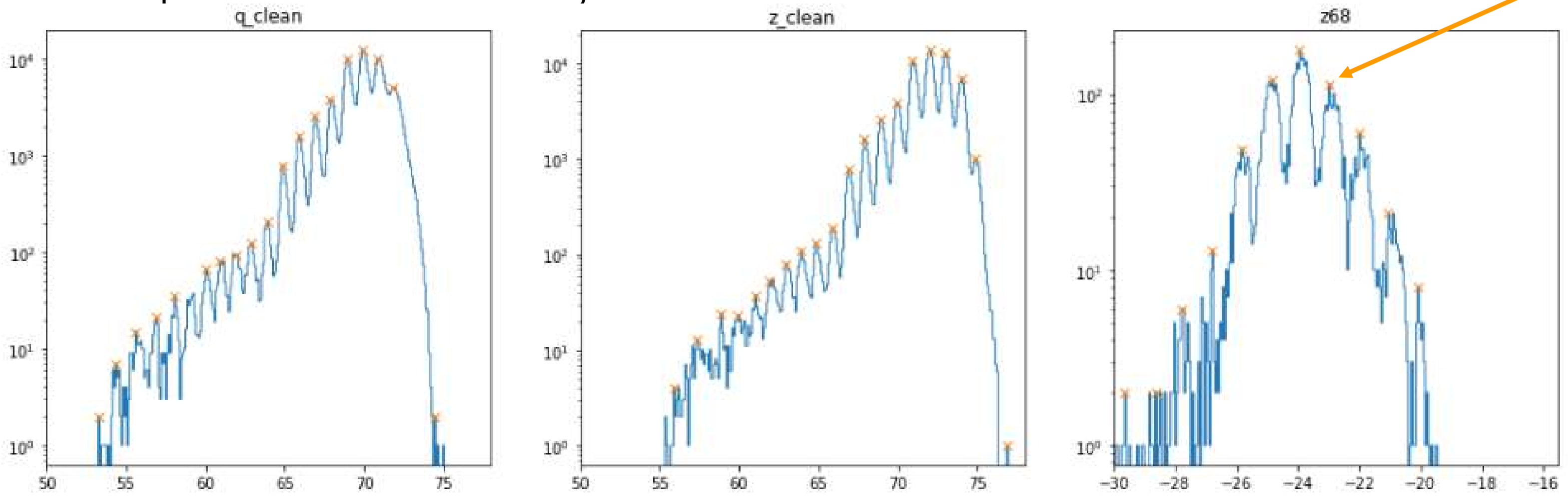
Finding Peaks

- The next cell will mark every peak found in a given spectrum
- First you should set the top two parameters
  - BINS – You can redefine here or keep same as before
  - min_gap – This is the minimum spacing in which you will allow the program to "find another peak". I find 0.85 is reasonable but this depends on the resolution of your data



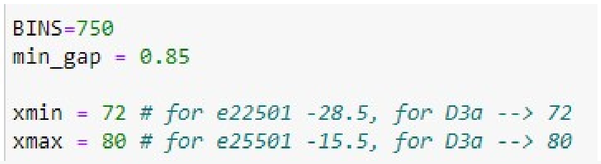
```
BINS=1000
min_gap = 0.85
```

Each peak found is indicated with an orange X

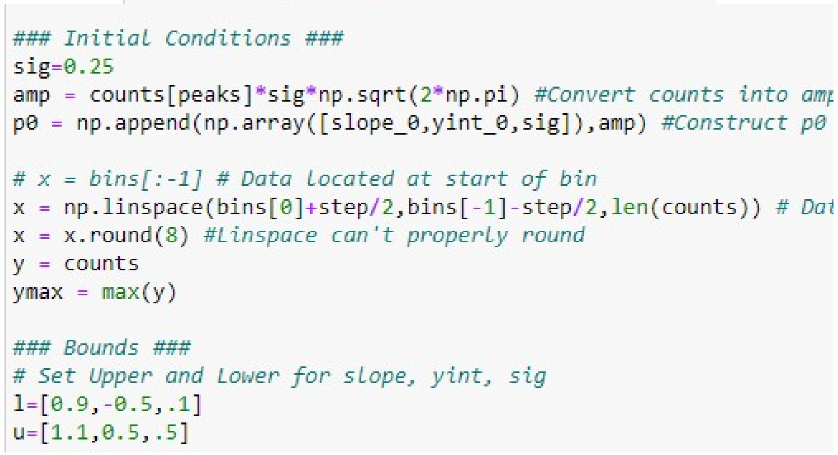


Fitting Peaks

- The next cell is setup to plot a fitted peak figure that is acceptable for use in a publication so there is a large matplotlib settings preamble
- After that you set all important parameters
  - BINS and min_gap as before
  - xmin / xmax – This is the range the figure will be plotted on; you need to set this manually to actually see the data
  - sig – This is the initial condition for that “one fixed width” mentioned in the first slide of this document
  - l / u – The lower and upper bounds for the fitting parameters
    - Slope – for linearly fitting the spectra to linearly spaced integer centers
    - Yint – the shifting part of the linear fit
    - Sig – One Fixed Width



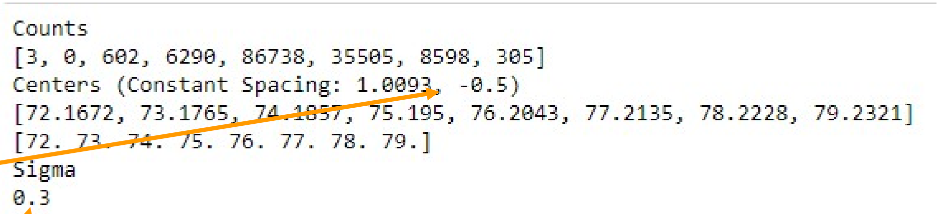
```
BINS=750
min_gap = 0.85

xmin = 72 # for e22501 -28.5, for D3a --> 72
xmax = 80 # for e25501 -15.5, for D3a --> 80
```



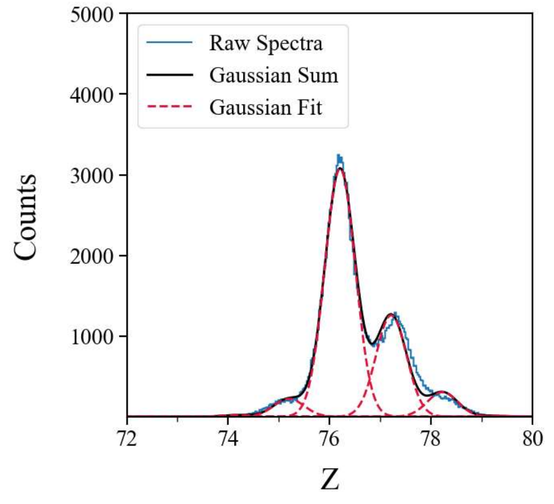
```
### Initial Conditions ###
sig=0.25
amp = counts[peaks]*sig*np.sqrt(2*np.pi) #Convert counts into amp
p0 = np.append(np.array([slope_0,yint_0,sig]),amp) #Construct p0

# x = bins[:-1] # Data Located at start of bin
x = np.linspace(bins[0]+step/2,bins[-1]-step/2,len(counts)) # Dat
x = x.round(8) #Linspace can't properly round
y = counts
ymax = max(y)

### Bounds ###
# Set Upper and Lower for slope, yint, sig
l=[0.9,-0.5,.1]
u=[1.1,0.5,.5]
```

Fitting Peaks 2

- The results of the fit will be printed below along with the final figure
- First you have the counts in each peak
  - This is the data you came here for! Now go put that into your cross section calculation!
- Next you can see how the one dimension of the spectra was linearly transformed with the slope and yint parameters
- The next two lines show you the actual location of each peak and where they SHOULD be (on integer values)
  - This functionality is unnecessary if you have your PID spectra properly calibrated, but we all know how picky the source code of SpecTcl can be when trying to calibrate
- And finally you get the single fixed width in Sigma

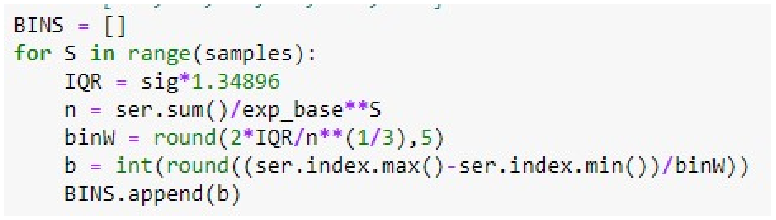


```
Counts
[3, 0, 602, 6290, 86738, 35505, 8598, 305]
Centers (Constant Spacing: 1.0093, -0.5)
[72.1672, 73.1765, 74.1857, 75.195, 76.2043, 77.2135, 78.2228, 79.2321]
[72. 73. 74. 75. 76. 77. 78. 79.]
Sigma
0.3
```



Binning 3

- You can build on this code
- One such example is writing a loop for multiple bin widths to compare a bunch of options all at once
- And would you look at that, there is the analytical function for optimal bin width here
  - IQR – Inter Quartile Range
  - n – number of counts in a given peak
- The issue with this expression is that it changes based on the number of counts in a peak
- Not all peaks have consistently the same statistics
- So while wider binning will look good for the low stat peaks, then the high stat peaks look funny
- Therefore a range of values are tried here

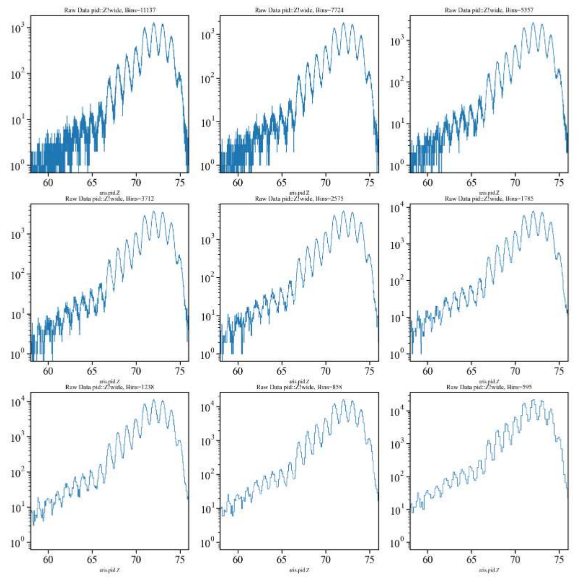


```
BINS = []
for S in range(samples):
    IQR = sig*1.34896
    n = ser.sum()/exp_base**S
    binW = round(2*IQR/n**(1/3),5)
    b = int(round((ser.index.max()-ser.index.min())/binW))
    BINS.append(b)
```

$$IQR = 1.34896\sigma$$

$$n = \frac{\Sigma C}{x^S}$$

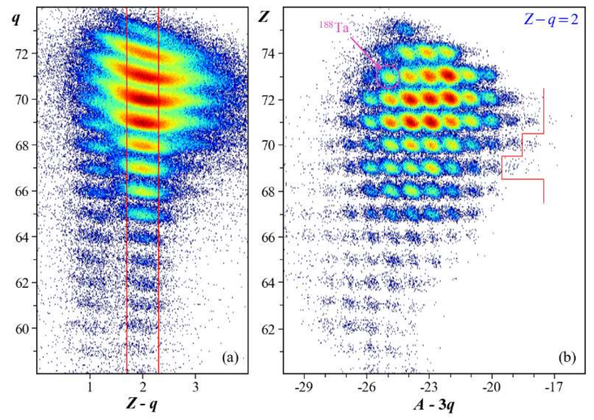
$$W = \frac{2IQR}{n^{\frac{1}{3}}}$$

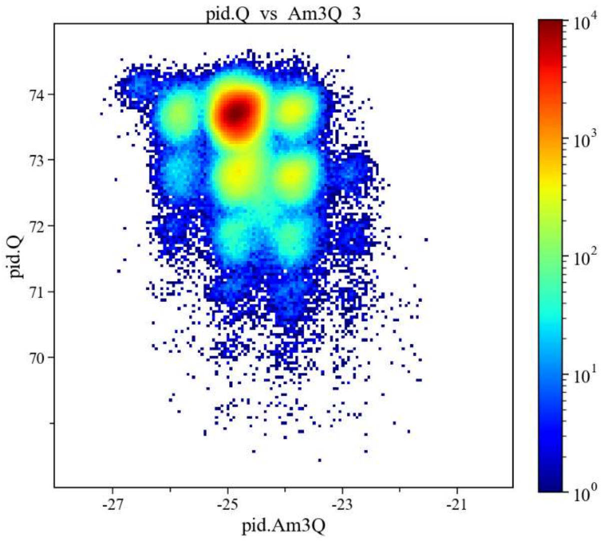


Appendix

- The last two cells don’t have anything to do with peak fitting
- But they CAN read in 2-D spectra files and plot them
- In fact the final cell is the code that generated the “famous” 5 new isotopes figure for
    - O.B. Tarasov et al., PRL 132 (2024) 072501

```
Image size: (320, 320), inherent AR: 1.0, desired AR: 1.732, applied AR: 1.732
Image size: (291, 483), inherent AR: 1.6597938144329898, desired AR: 1.212, applied AR: 2.0116701030927837
```





Someday it would be nice to fit 2-D plots with 2-D gaussian as opposed to 1-D plots…
It is possible, you need an AI clustering script and efficient coding of the fitting algorithm
Oleg made the BI program in LISE which does a decent job at finding and providing parameters for 2D peaks
But without being able to capture each peak, I was unable to make use of it in my analysis