NP04 PDS operation meeting
29/08/2024

DEEP UNDERGROUND
NEUTRINO EXPERIMENT

WAFFLES
Waveform Analysis Framework For Light Emission Studies

interactive drawing tools

A. Cervera
(IFIC-Valencia)

Introduction

- Waffles is the NP04 PDS analysis framework, initially developed by Julio, and now with contributions from Manuel, Laura, Renan and Sam
- Git repository

  https://github.com/DUNE/waffles

- We have been working in a user friendly interface for quickly getting into plots. For the moment in a branch:
  - anselmo_plotting
- Three main files:

`waffles/src/waffles/plotting/drawing_tools.py`	The plotting tools
`waffles/docs/examples/drawing_tools_example.py`	Example
`waffles/src/waffles/np04_analysis/light_yield_vs_beam_energy/beam.py`	Beam selection example

Introduction

- This is very preliminary (just started last week)
- The idea of this talk is to show the general strategy behind that and make sure it is seen as a useful tool
- A proper presentation with precise instruction will be given once the tools are more advanced and integrated into the main branch
- Nevertheless, people can start using those tools. Feedback will be welcomed !!!!

Import the waffles interactive drawing tools

- Enter a python interactive session
- Importing the waffles interactive tools

```
import sys

# Change this folder to yours
waffles_dir = '/Users/acervera/HEP/DUNE/ProtoDUNE-HD/PDS/data_taking/waffles'
sys.path.append(waffles_dir+'/src')

# import the waffles drawing tools
import waffles.plotting.drawing_tools as draw
```

- You can add this code to a python script (e.g. draw.py) and call it like this:

```
python -i draw.py
```

help

- In a interactive python session you can type draw.help()

```
>>> draw.help()
List of commands. Type draw.help(draw.X) to see the arguments of command X
plot                             plot waveforms for a single waveform, list of waveforms or WaveformSet
plot_hm                          plot heat map for a WaveformSet
plot_charge                      plot charge histogram for a WaveformSet
plot_charge_peaks                plot charge histogram peaks given a charge histogram
plot_avg                         plot average waveform for a WaveformSet
plot_to                          plot time offset (timestamp-daq_timestamp) for a WaveformSet
get_wfs_with_variable_in_range   get all waveforms with a given variable in a given range
get_wfs_in_channel               get all waveforms in a given endpoint and channel
zoom                             makes a zoom of the current figure
```

- Usage for individual commands

```
>>> draw.help(draw.plot)
plot            plot waveforms for a single waveform, list of waveforms or WaveformSet
(object, ep: int = -1, ch: int = -1, nwfs: int = -1, xmin: int = None, xmax: int = None, offset: bool = False, op: str = None, show: bool = True)
```

```
>>> draw.help(draw.plot_charge)
plot_charge     plot charge histogram for a WaveformSet
(wset: waffles.data_classes.WaveformSet.WaveformSet, ep: int = -1, ch: int = -1, int_ll: int = 135, int_ul: int = 165, nb: int = 200,
hl: int = -5000, hu: int = 50000, b_ll: int = 0, b_ul: int = 100, nwfs: int = -1, variable: str = 'integral', op: str = None)
```

Drawing tools example

- Go inside `waffles/docs/examples`
- And run the example `python -I drawing_tools_example.py`
- This example contain the code shown in the next slides
- IMPORTANT !!! Plotting mode. Two options:

1. .png file, to be used for example in visual studio remotely

```
#open a png plot
draw.plotting_mode = 'png'
draw.png_file_path = waffles_dir+'/temp_plot.png'
```

4. html file

```
#open a html plot
draw.plotting_mode = 'html'
draw.html_file_path = waffles_dir+'/temp_plot.html'
```

Reading the data

- The WaveformSet object is a collection of waveforms with the same structure (same number of time ticks: self trigger or full streaming)

```
# read the root file
wset=draw.read(waffles_dir+"/../DATA/run26687.root",0,1)
```

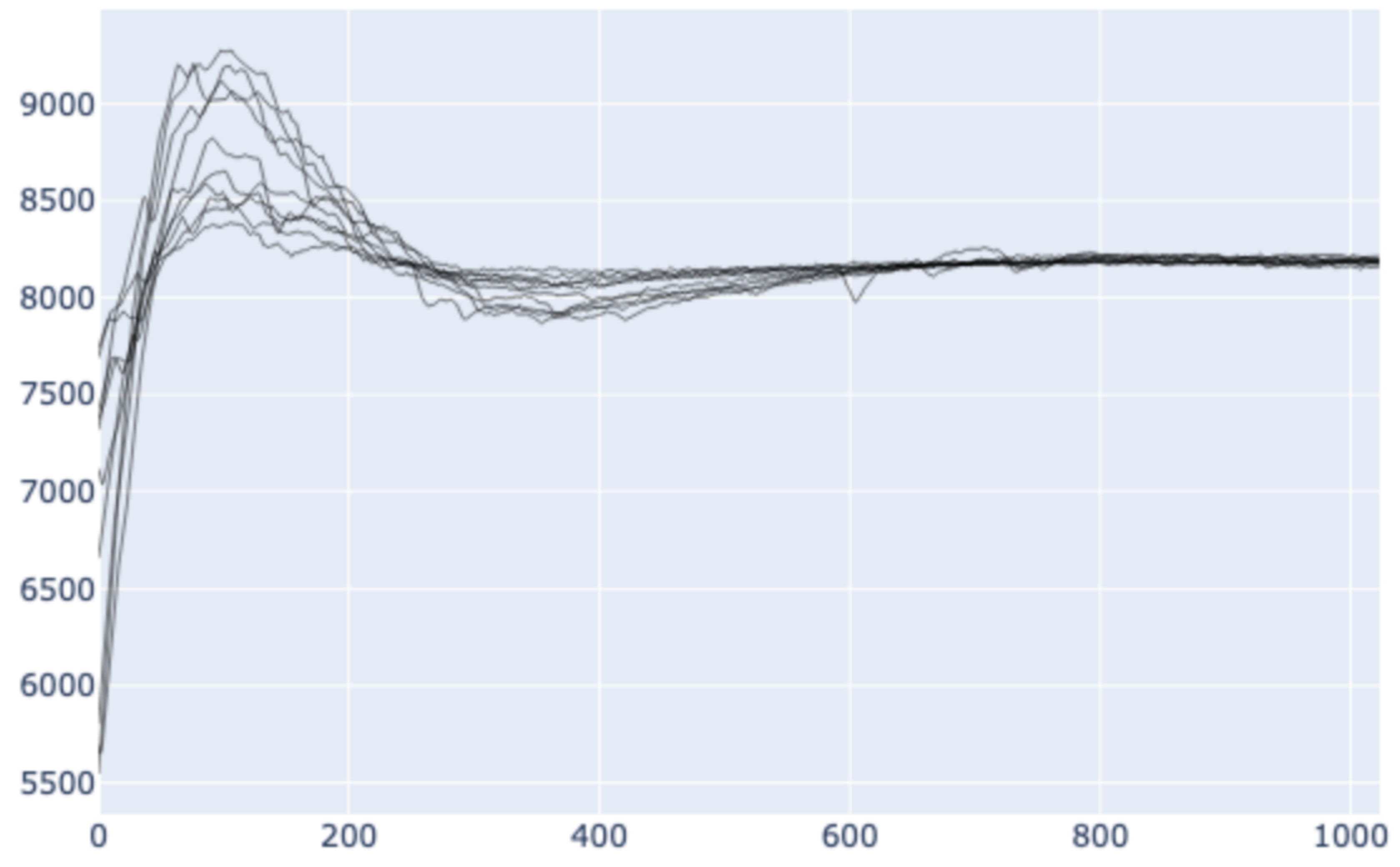
start fraction stop fraction

- This returns a WaveformSet object
- Root files for many runs can be found in

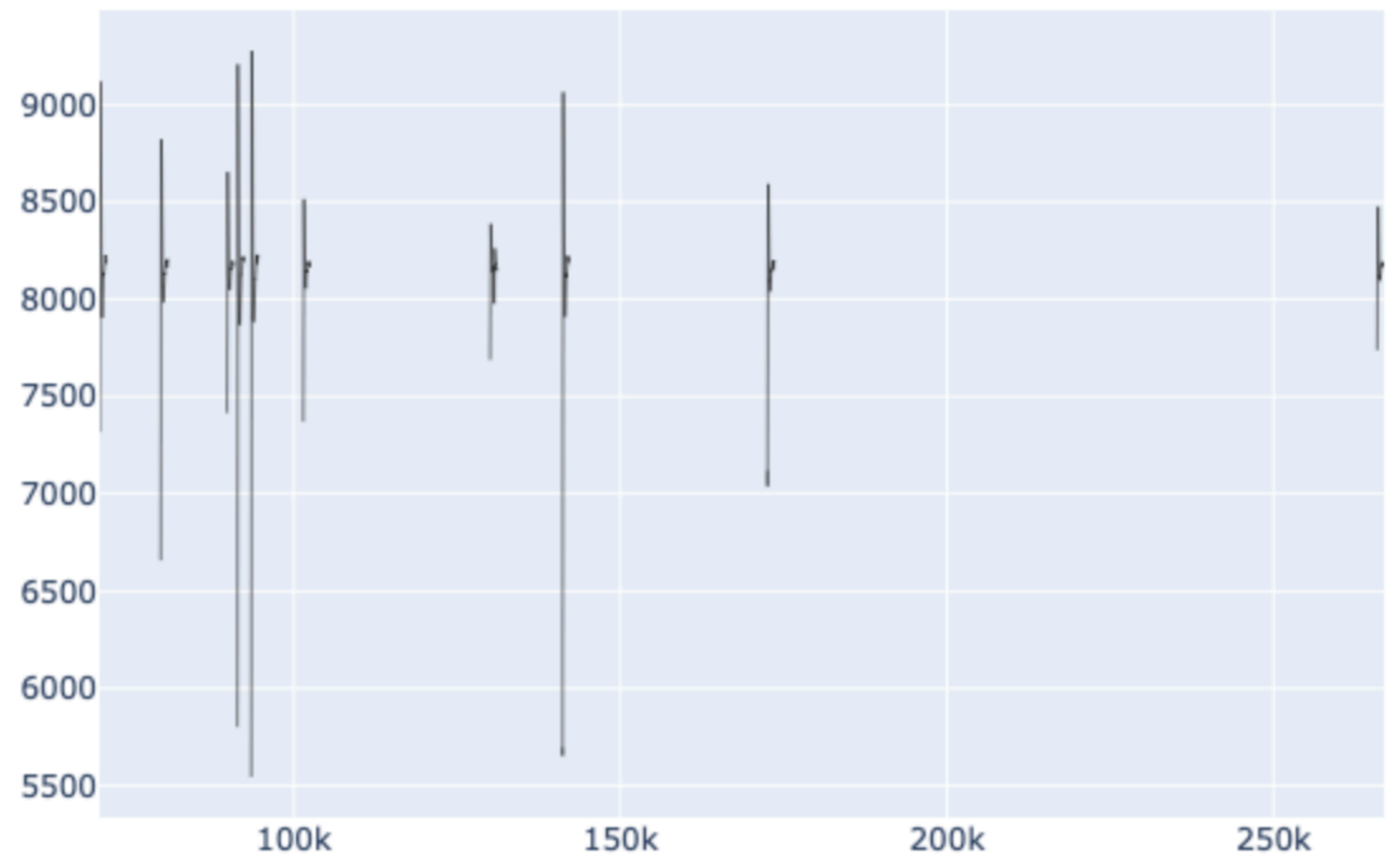
```
/eos/experiment/neutplatform/protodune/experiments/ProtoDUNE-II/PDS_Commissioning/waffles/2_daq_root/
```

Plotting single waveforms

```
# plot 10 wfs for endpoint 111 and channel 45
draw.plot(wset,111,45,10)
```



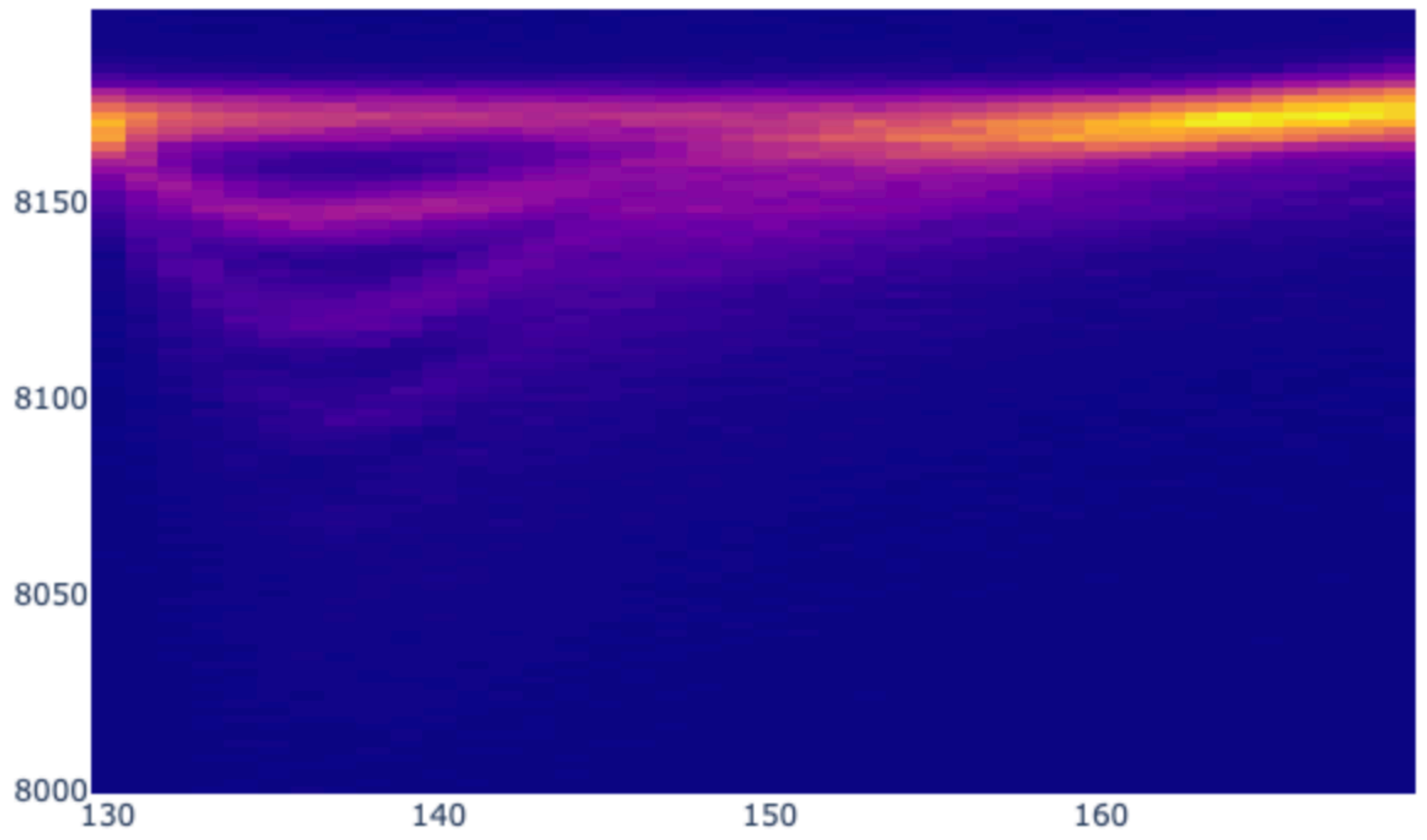
```
# Same plot but now with time with respect to daq window
draw.plot(wset,111,45,10,offset=True)
```



Heat map plot (persistency)

```
# plot the heat map for that channel.
draw.plot_hm(wset,111,45,40,130,170,100,8000,8200)
```

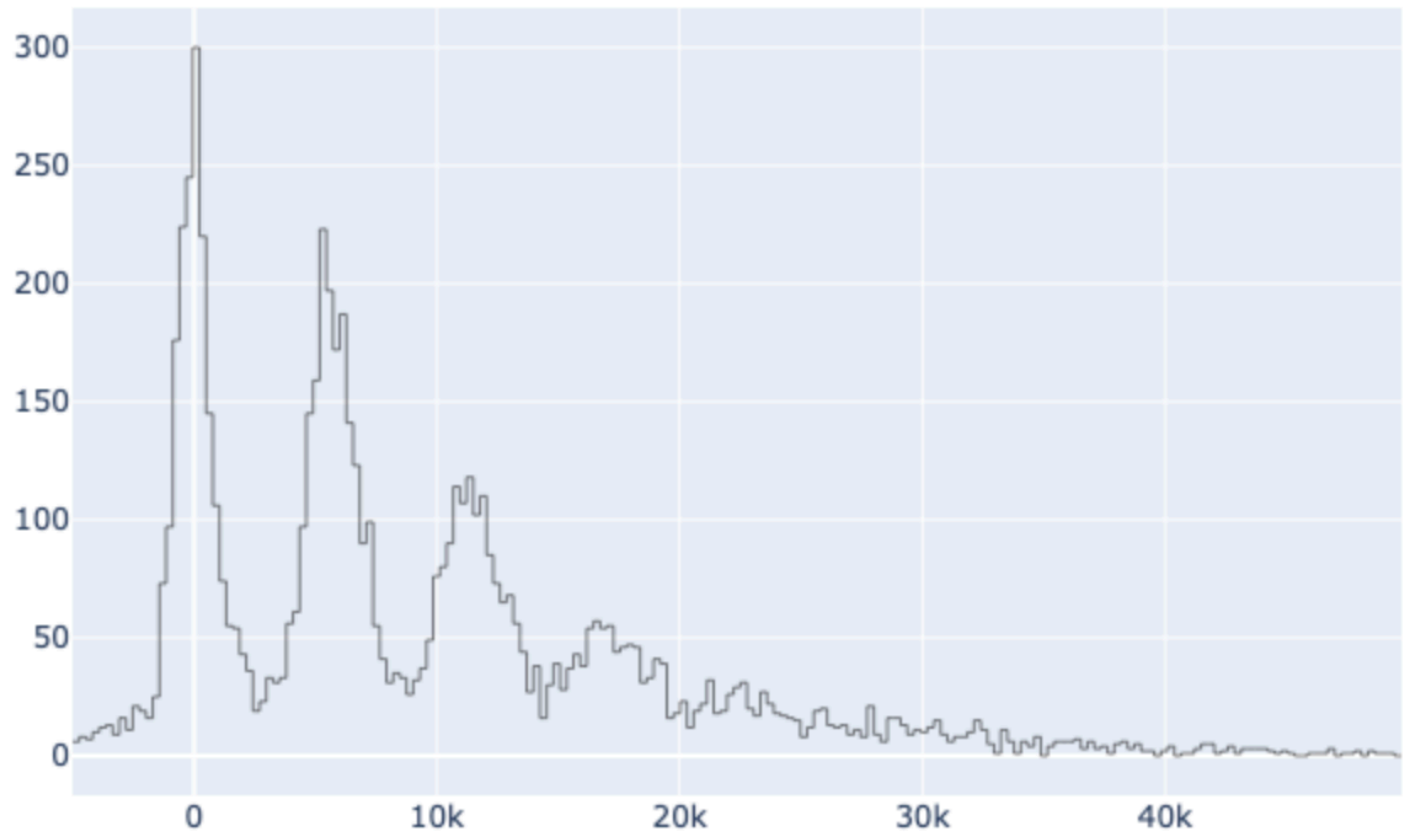
$\underbrace{\texttt{111,45}}_{\text{ep-ch}}$ $\underbrace{\texttt{40,130,170}}_{\text{x-binning}}$ $\underbrace{\texttt{100,8000,8200}}_{\text{y-binning}}$



Charge histogram

```
# plot the charge histogram with integration limits 135,165
draw.plot_charge(wset,111,45,135,165)
```

ep-ch Integ limits

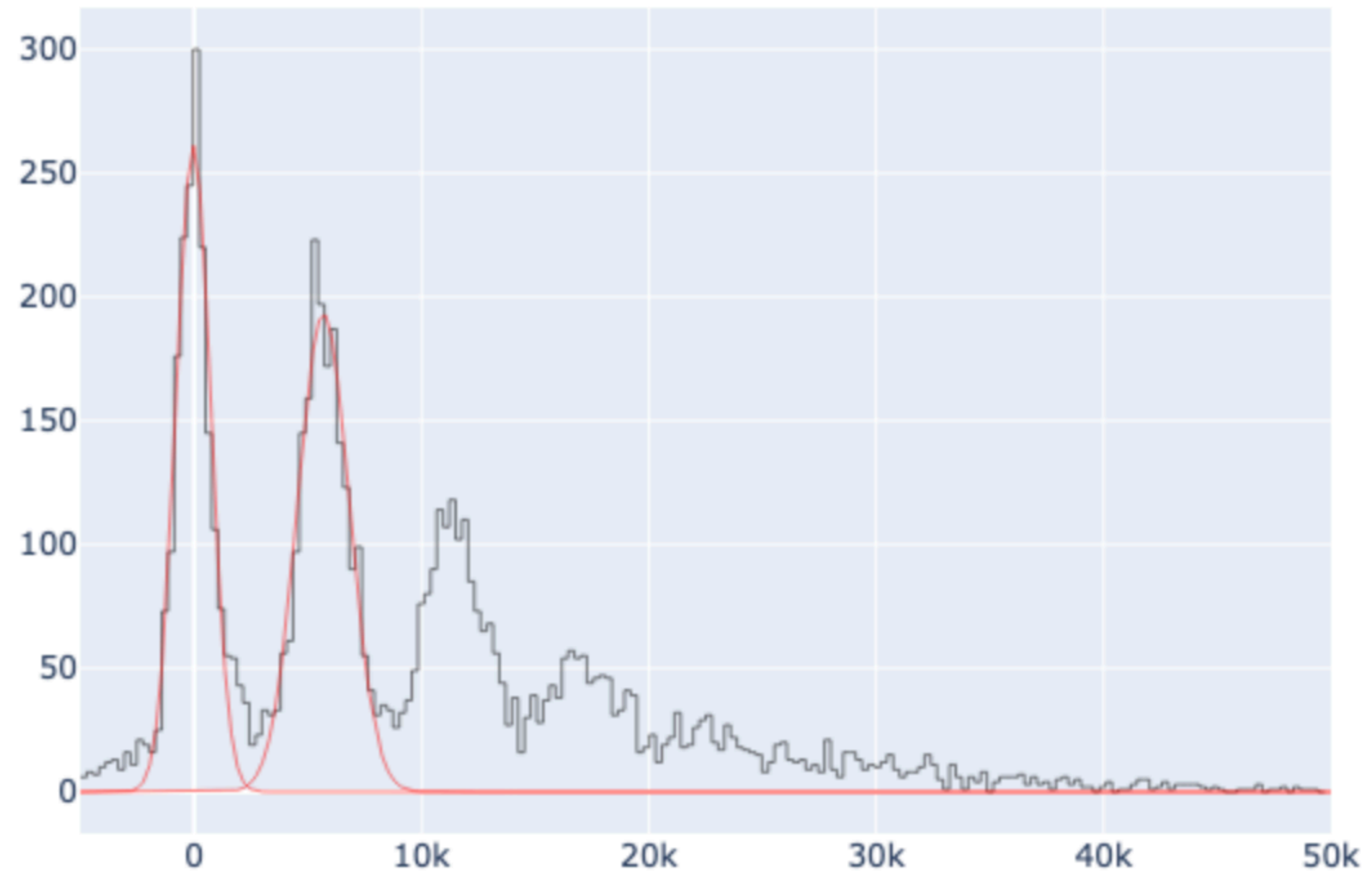


- This method returns a CalibrationHistogram object, so it could be called like this:

```
# plot the charge histogram with integration limits 135,165
charge_histo = draw.plot_charge(wset,111,45,135,165)
```

Fit peaks

```
# plot the charge histogram and show the peaks
draw.plot_charge(wset,111,45,135,165,op="peaks")
```



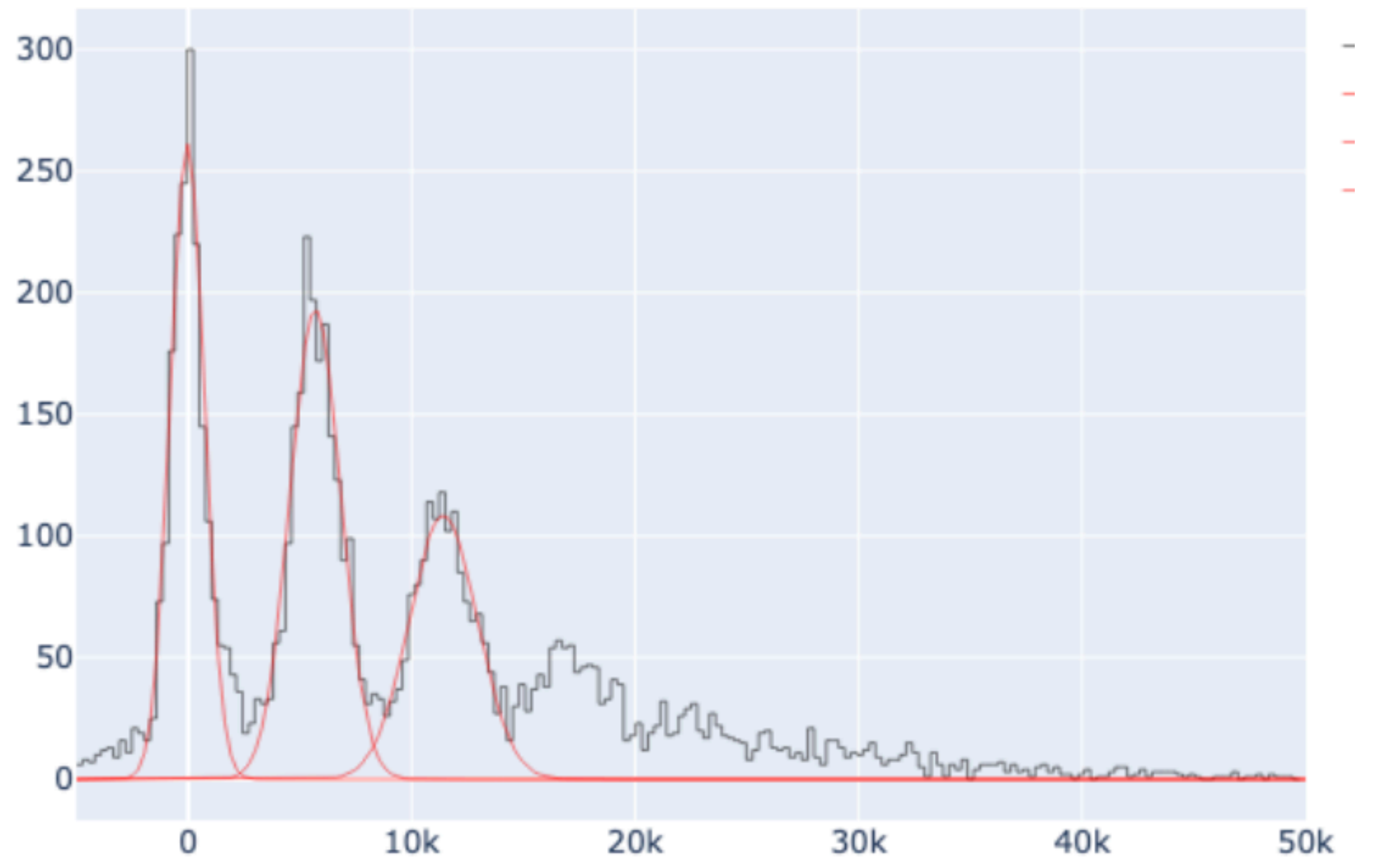
- Fits two peaks by default
- See next slide for changing parameters
- It also produces this output on the screen

```
S/N =    5.920916442575934
gain =  5723.0753896306105
s.p.e. mean charge =  5686.134710791045  +-  50.37054651124145
```

Fit peaks: change parameters

```
# plot the charge histogram with integration limits 135,165
charge_histo = draw.plot_charge(wset,111,45,135,165)
```

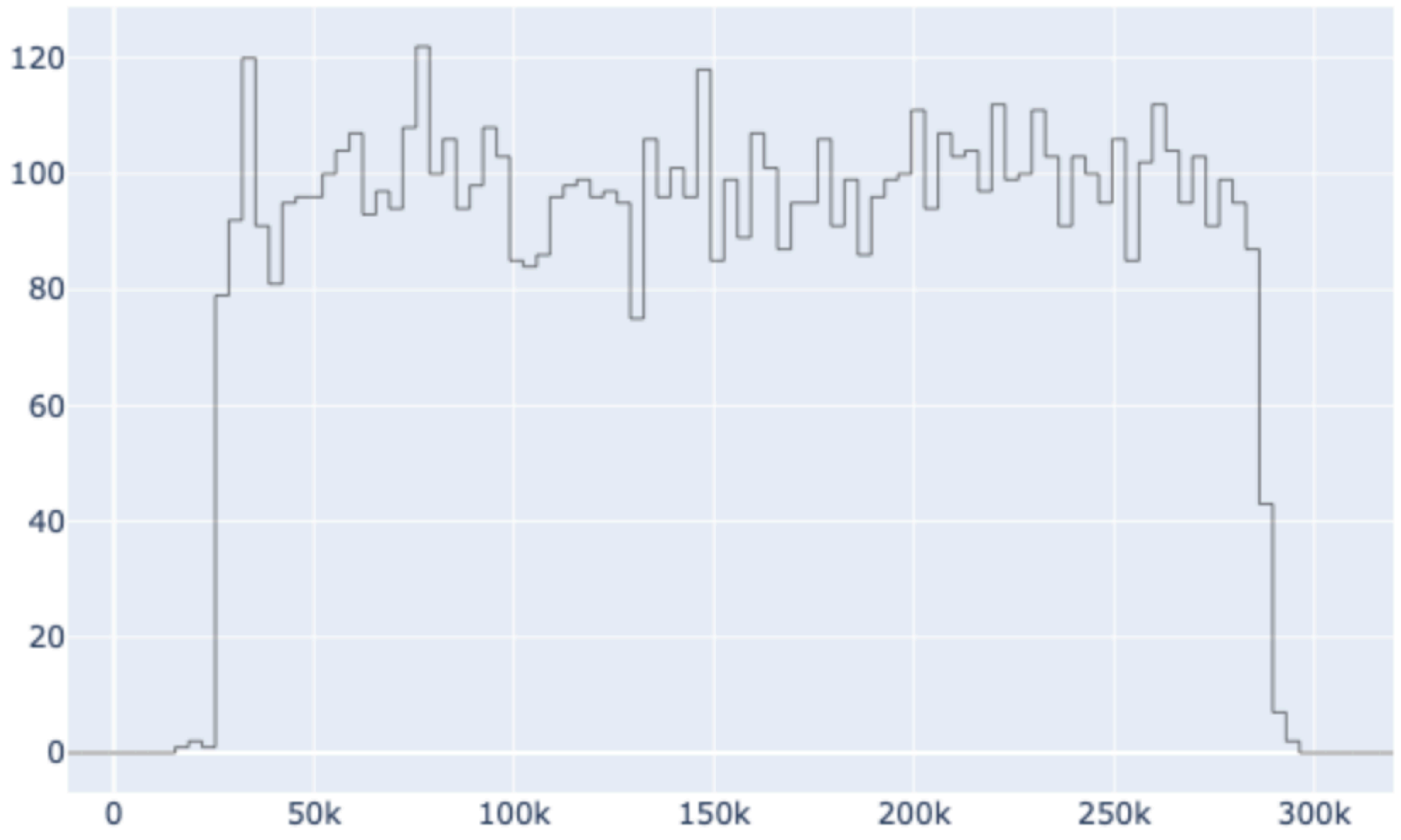
```
# if we want to change peaks parameters use this method, which takes as argument
# the charge histogram produced above

# plot the charge histogram with 3 peaks
draw.plot_charge_peaks(charge_histo,3)
```



```
>>> draw.help(draw.plot_charge_peaks)
(calibh: waffles.data_classes.CalibrationHistogram.CalibrationHistogram,
npeaks: int = 2, prominence: float = 0.2, half_points_to_fit: int = 10,
op: str = None)
```

Time offset

```
# plot time offset for all waveforms in channel 111 - 45
draw.plot_to(wset, 111, 45)
```



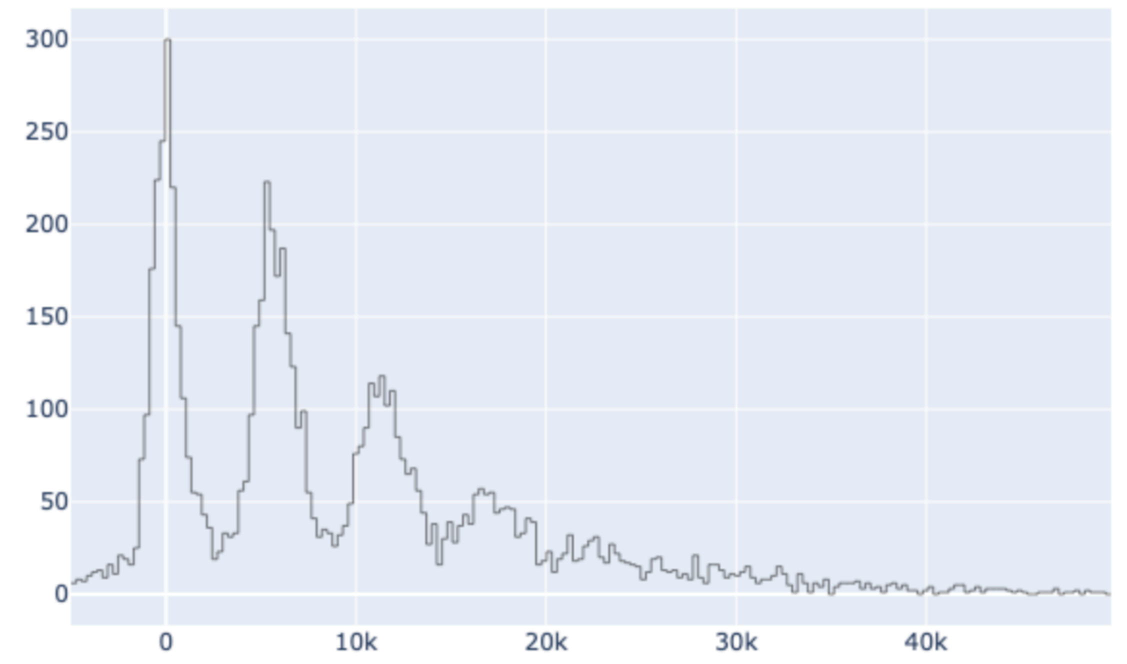
- This is the offset w.r.t. to the beginning of the ~4 ms DAQ window

Waveform selection

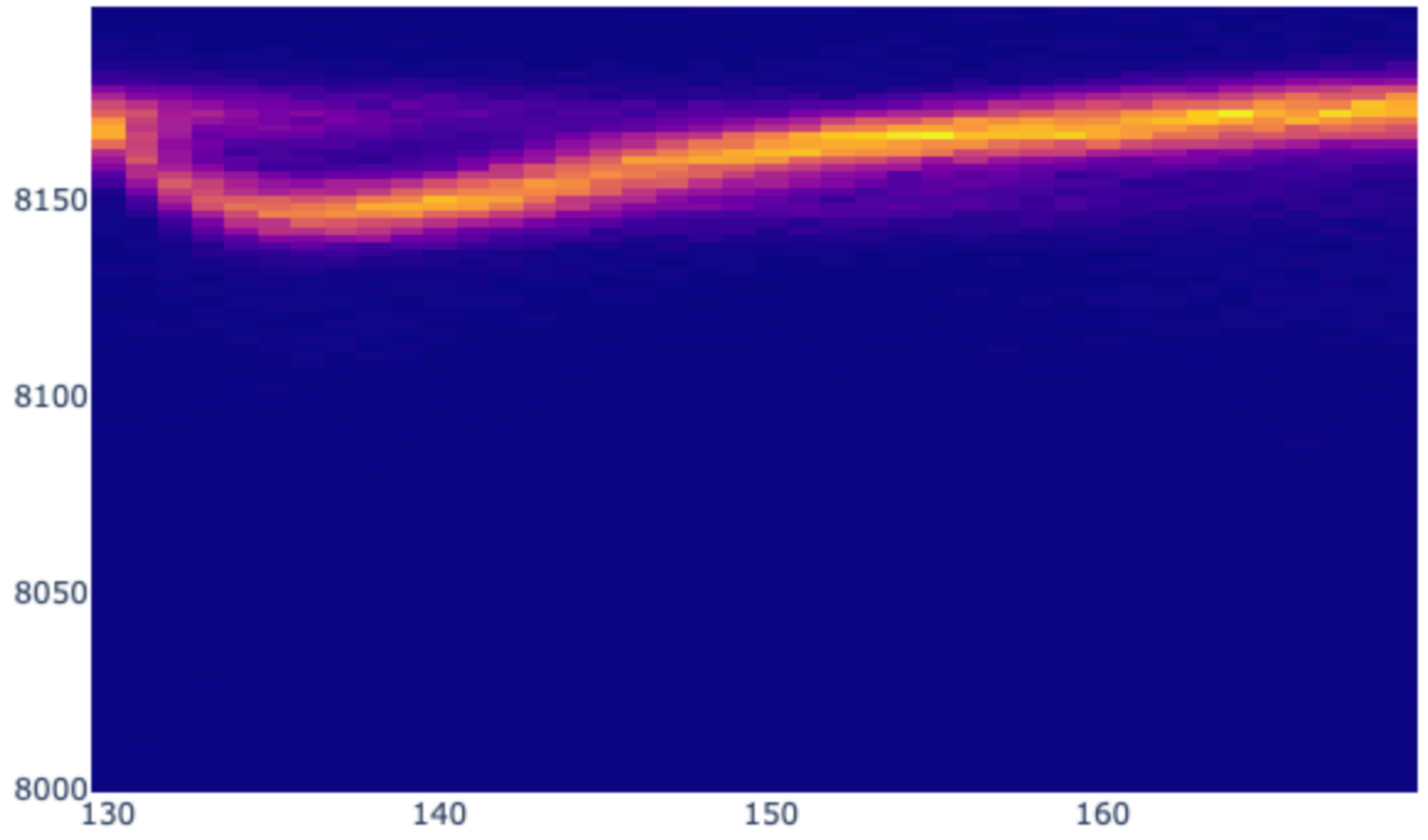
```
# get a WaveformSet with only wfs in ep 111 and ch 45
wset_11145 = draw.get_wfs_in_channel(wset,111,45)
```

```
# get all wfs in that channel with integral in the 1 p.e. peak [3500,7500]
wset_11145_1pe = draw.get_wfs_with_integral_in_range(wset_11145,3500,7500)
```

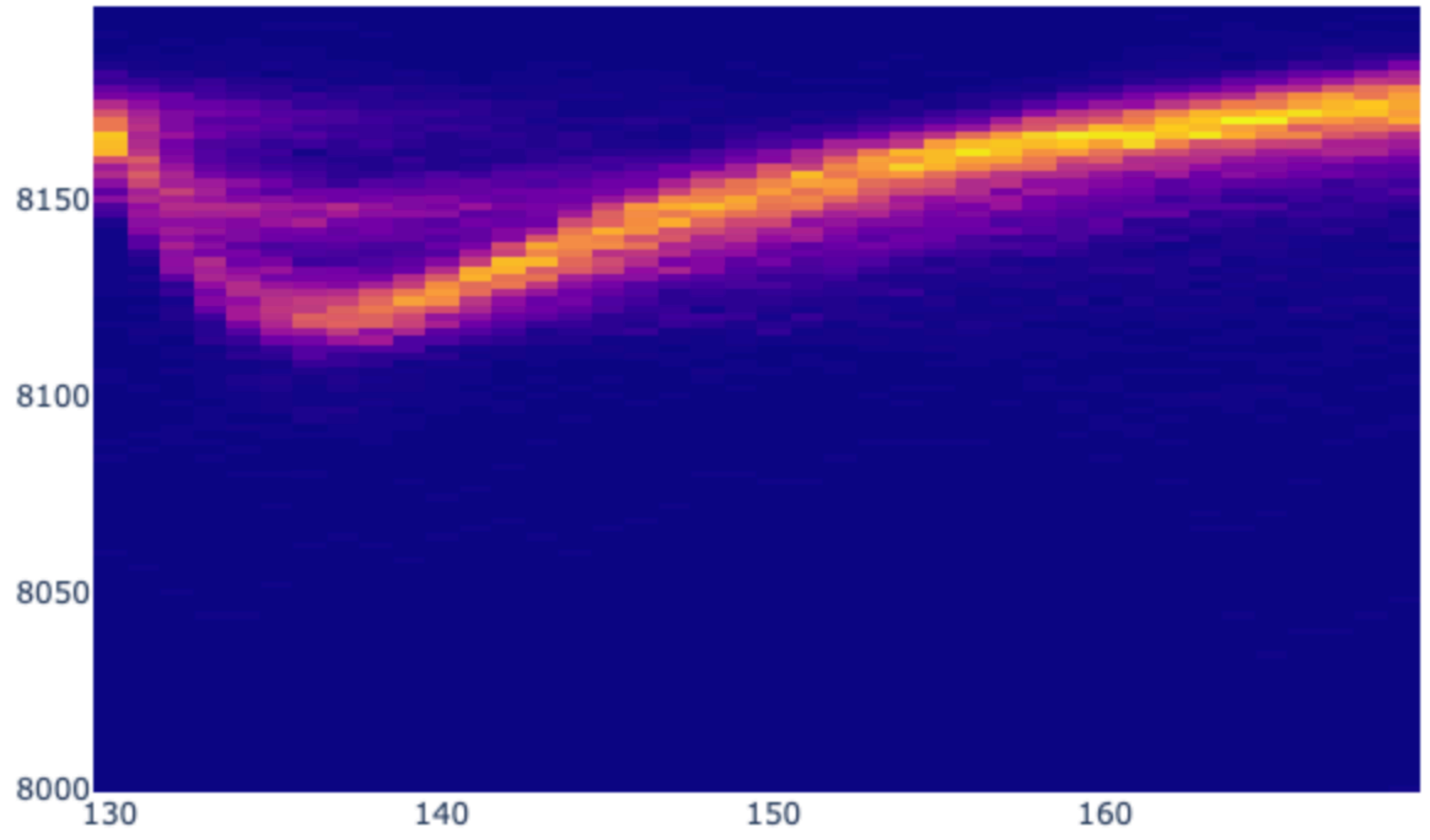
```
# get all wfs in that channel with integral in the 2 p.e. peak [10000,14000]
wset_11145_2pe = draw.get_wfs_with_integral_in_range(wset_11145,10000,14000)
```



```
# plot the heat map for that waveform subsample (1 pe waveforms)
draw.plot_hm(wset_11145_1pe,111,45,40,130,170,100,8000,8200)
```



```
# plot the heat map for that waveform subsample (2 pe waveforms)
draw.plot_hm(wset_11145_2pe,111,45,40,130,170,100,8000,8200)
```



General selection method

```python
from waffles.data_classes.Waveform import Waveform

# example of general filtering method
def filter_example(waveform: Waveform, allowed_channels) -> bool:
    # This condition could be whatever (use all Waveforms data members)
    if waveform.Endpoint == 111 and waveform.Channel in allowed_channels:
        return True
    else:
        return False

# collect all waveforms in chs 40 and 45
wset_40_45 = wset.from_filtered_WaveformSet(wset, filter_example,[40,45])


# collect all waveforms in chs 40,42,45
wset_40_42_45 = wset.from_filtered_WaveformSet(wset, filter_example,[40,42,45])
```

Selecting beam events

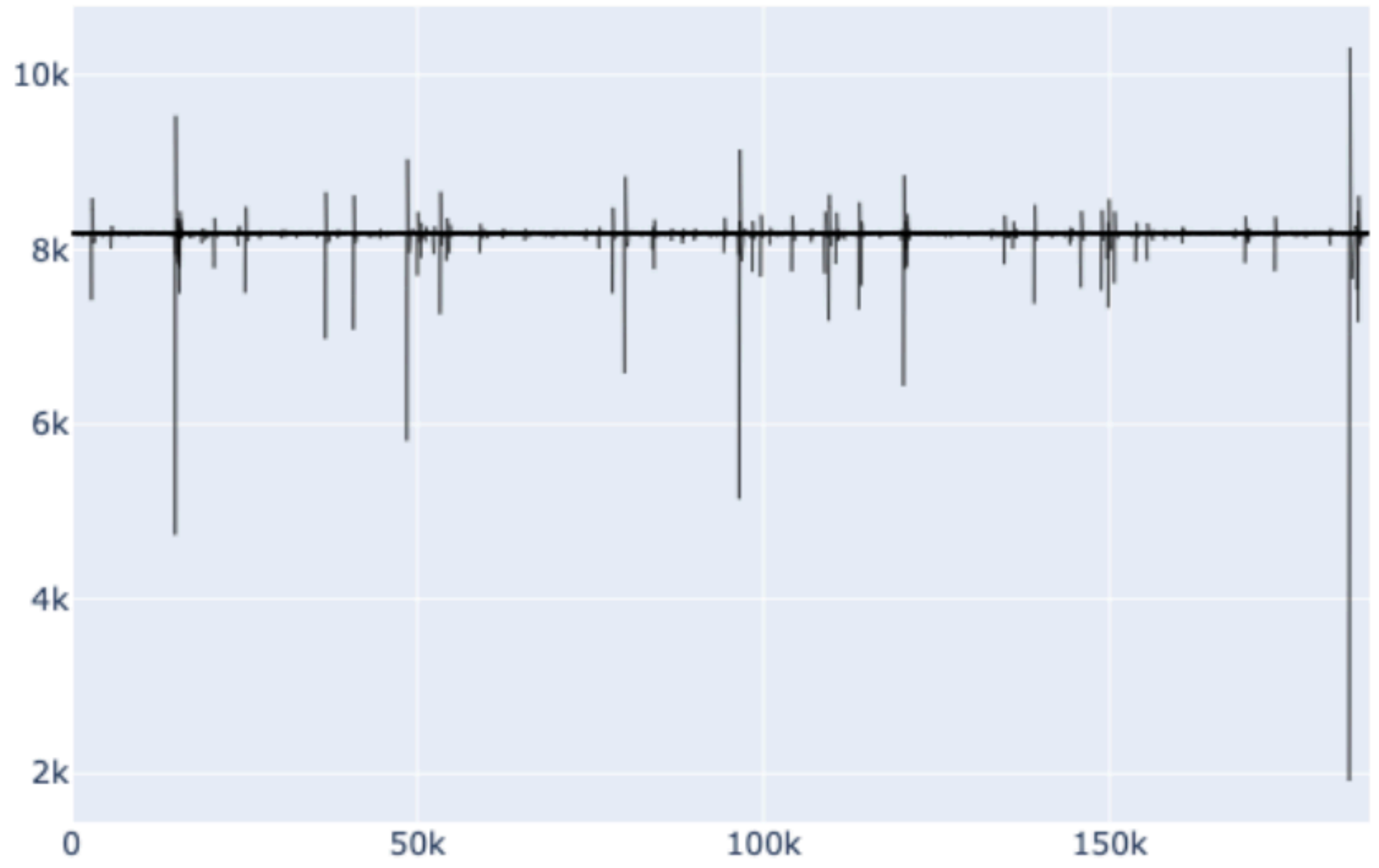
- Example of beam selection for self-trigger and full-streaming in:

`waffles/src/waffles/np04_analysis/light_yield_vs_beam_energy/beam.py`

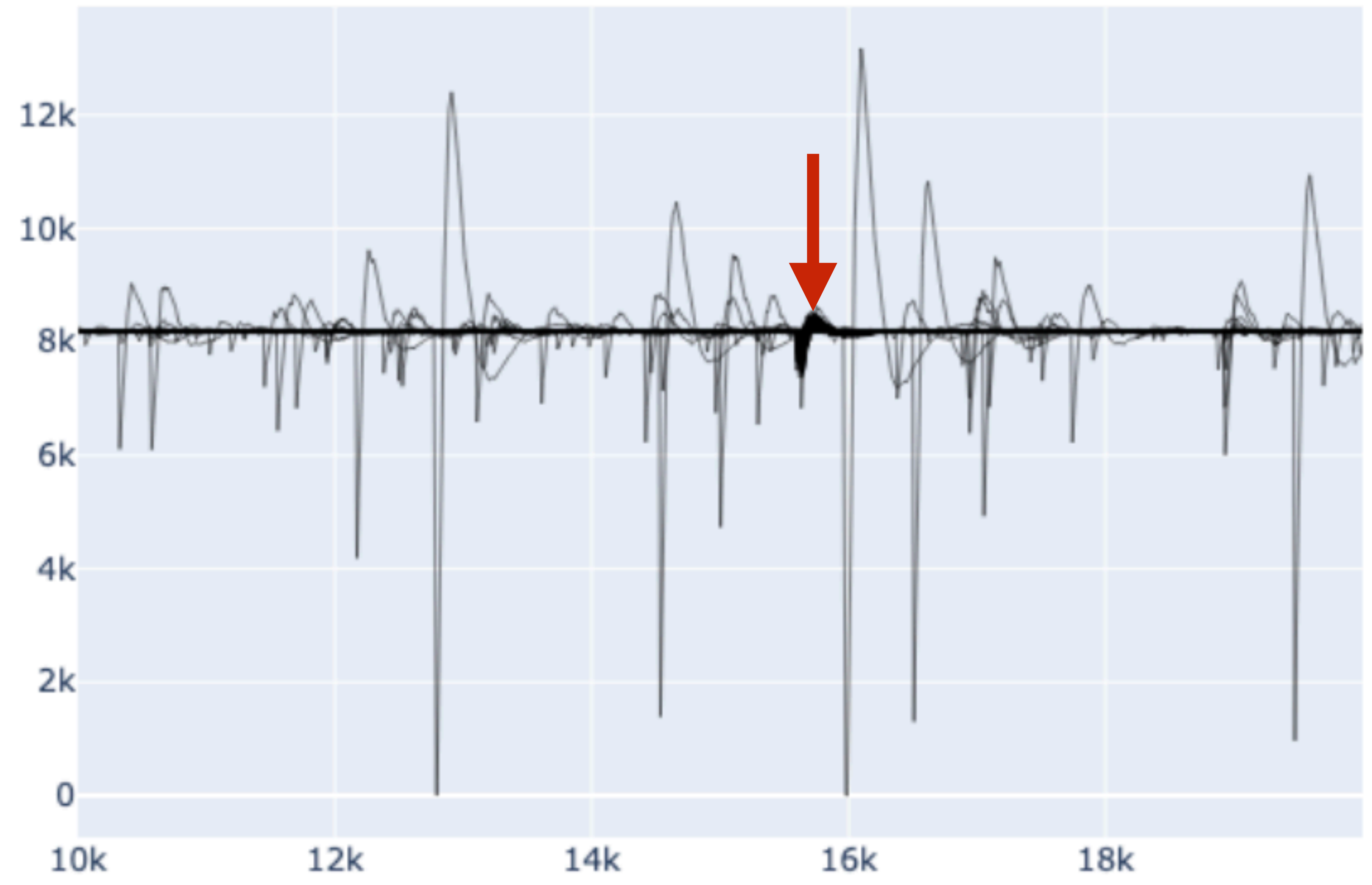
```
wset1=draw.read(waffles_dir+"/../DATA/run027338.root",0,1,True,True)
wset7=draw.read(waffles_dir+"/../DATA/run027374.root",0,1,True,True)
```

Needed for full streaming

```
# Plot 5 wfs in ep 104 ch 15 (APA1)
draw.plot(wset1,104,15,5)
```



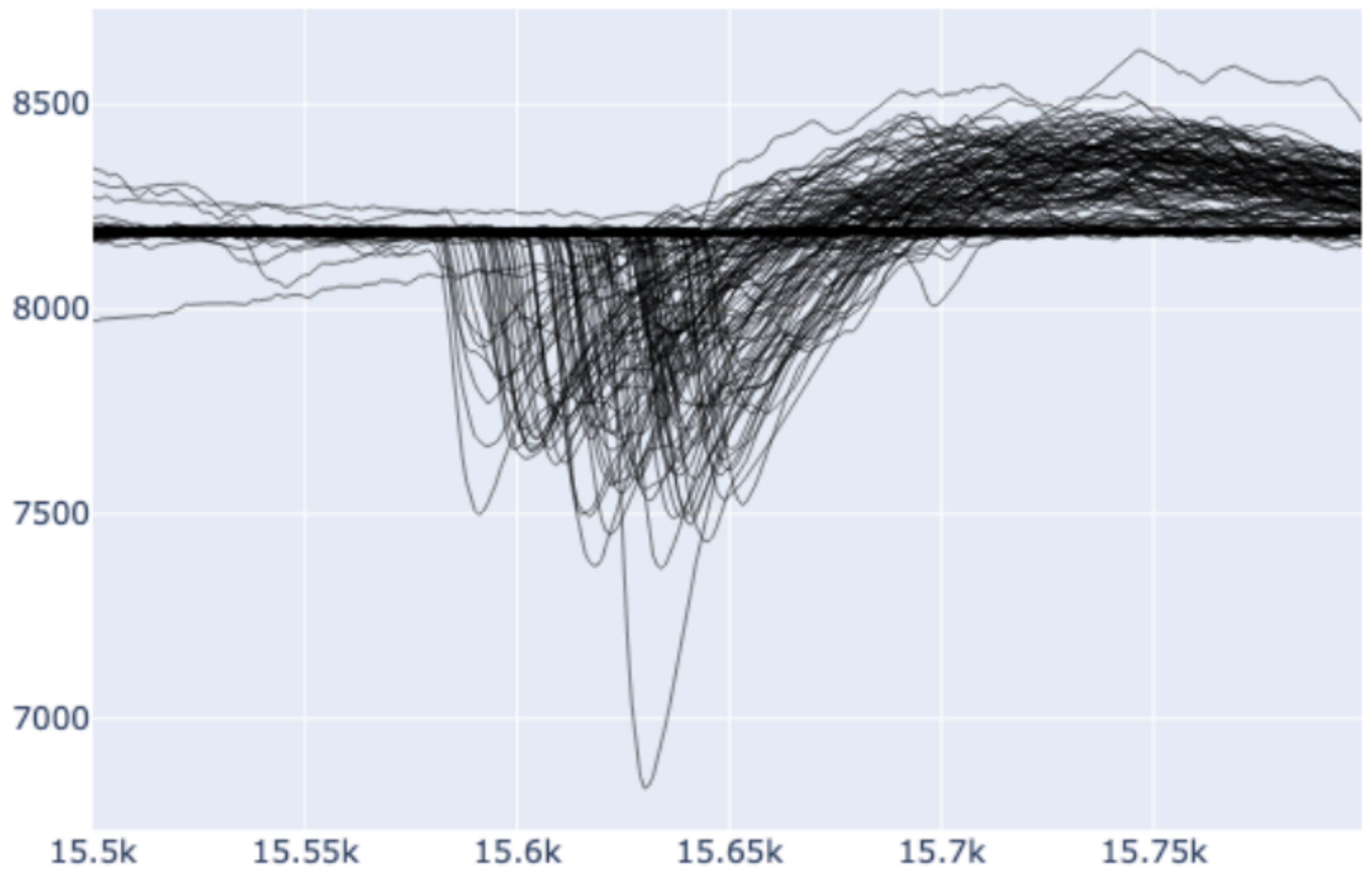
```
# zoom on the beam peak
draw.plot(wset1,104,15,200,10000,20000)
```

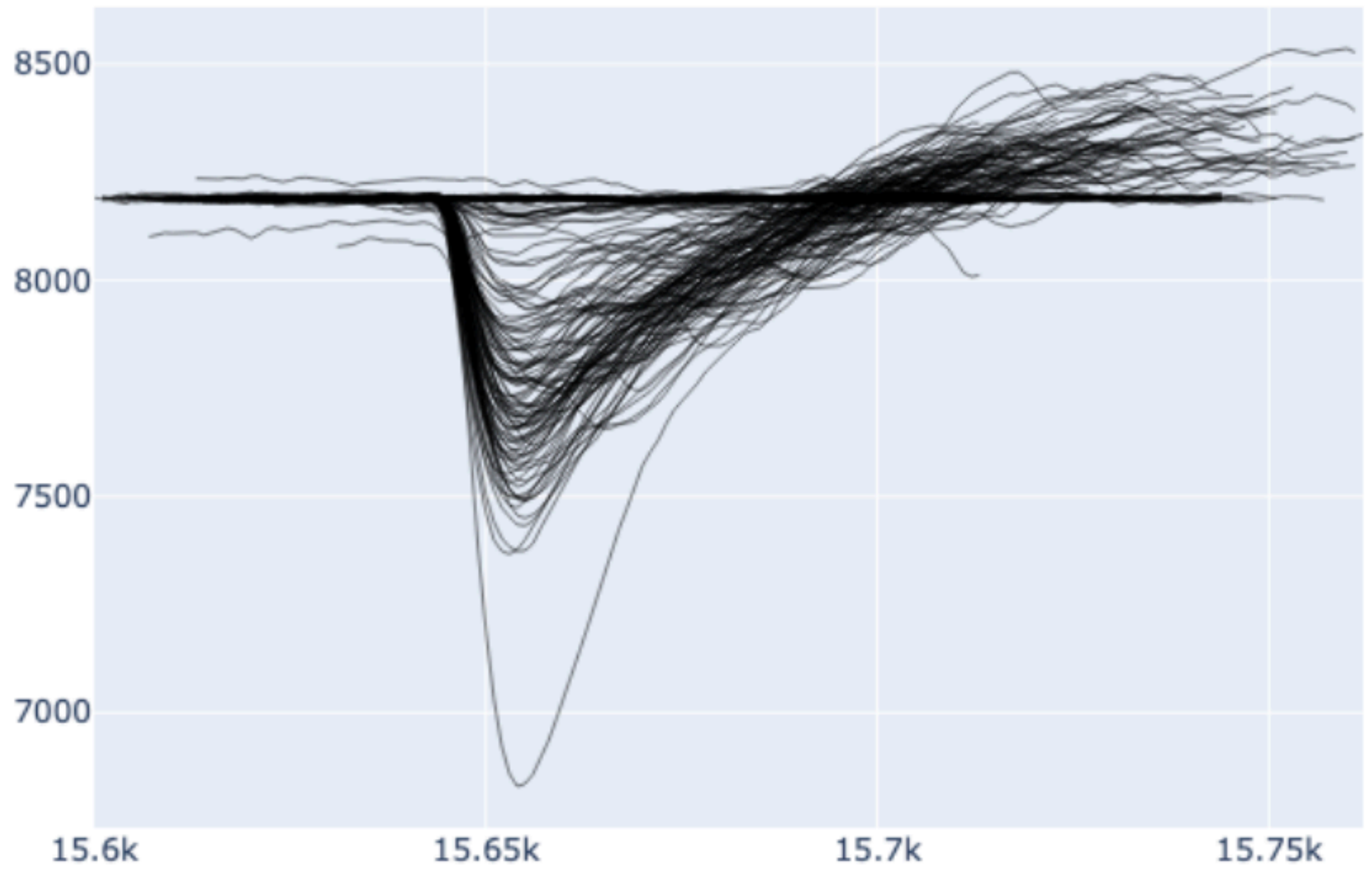


Time offsets

```
# Align all waveforms in time applying the offset (timestamp-daq_timestamp)
draw.plot(wset1,104,15,200,15600,15700,offset=True)
```

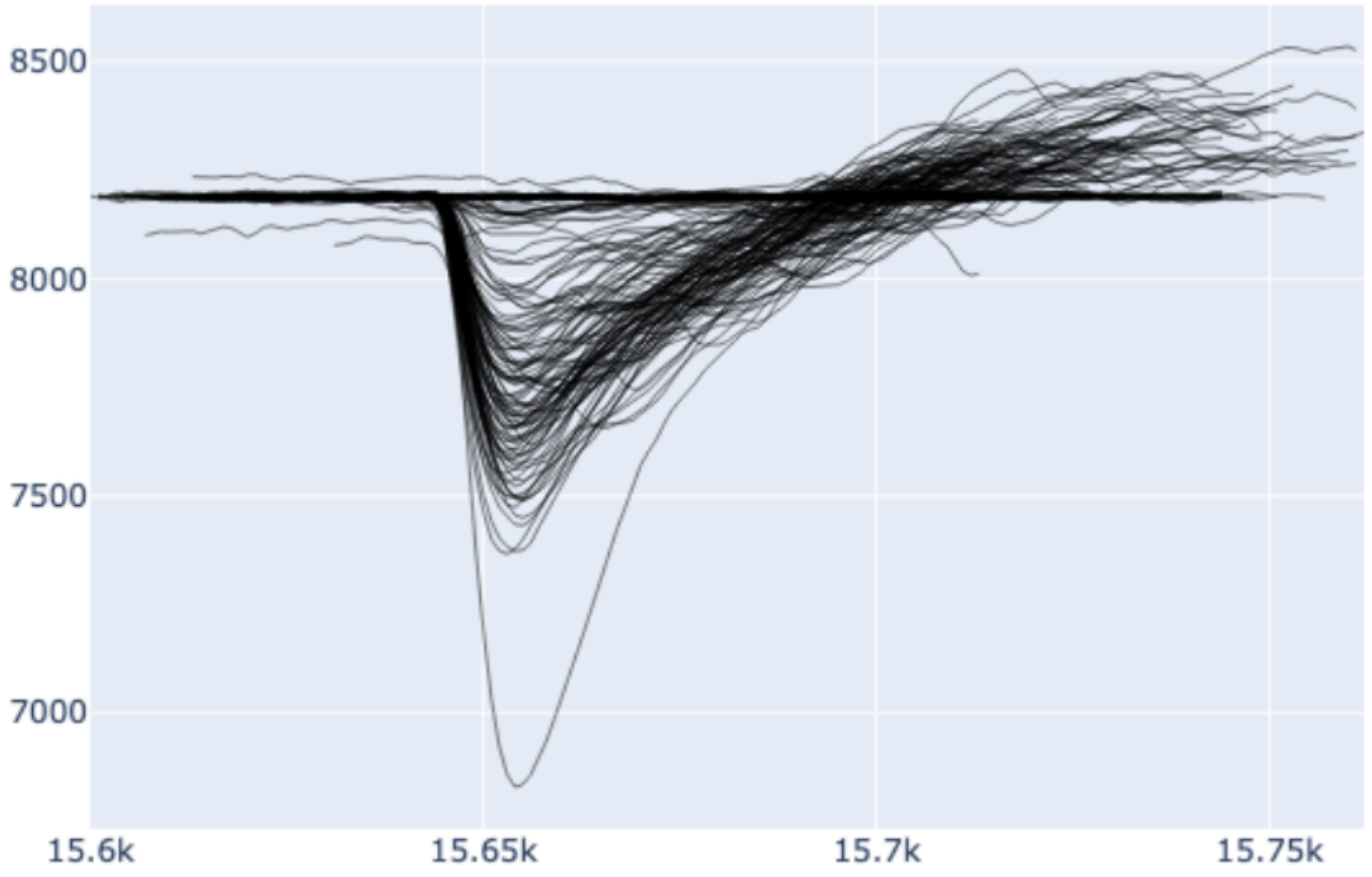
```
# zoom on the beam peak even more
draw.plot(wset1,104,15,200,15500,15800)
```



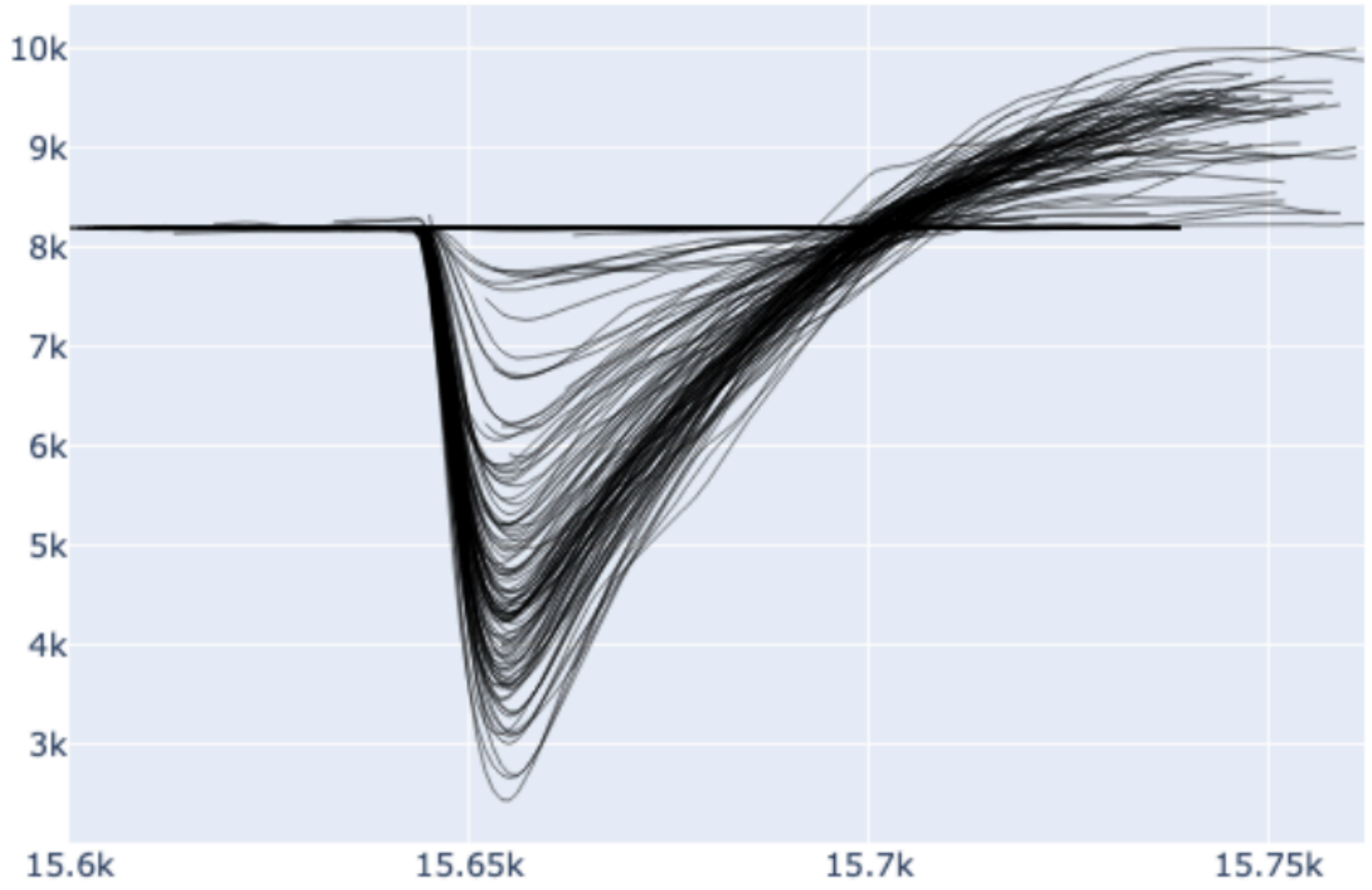


Various energies

```
# Align all waveforms in time applying the offset (timestamp-daq_timestamp)
draw.plot(wset1,104,15,200,15600,15700,offset=True)
```



```
# Do the same for 7 GeV
    draw.plot(wset7,104,15,200,15600,15700,offset=True)
```



Backup