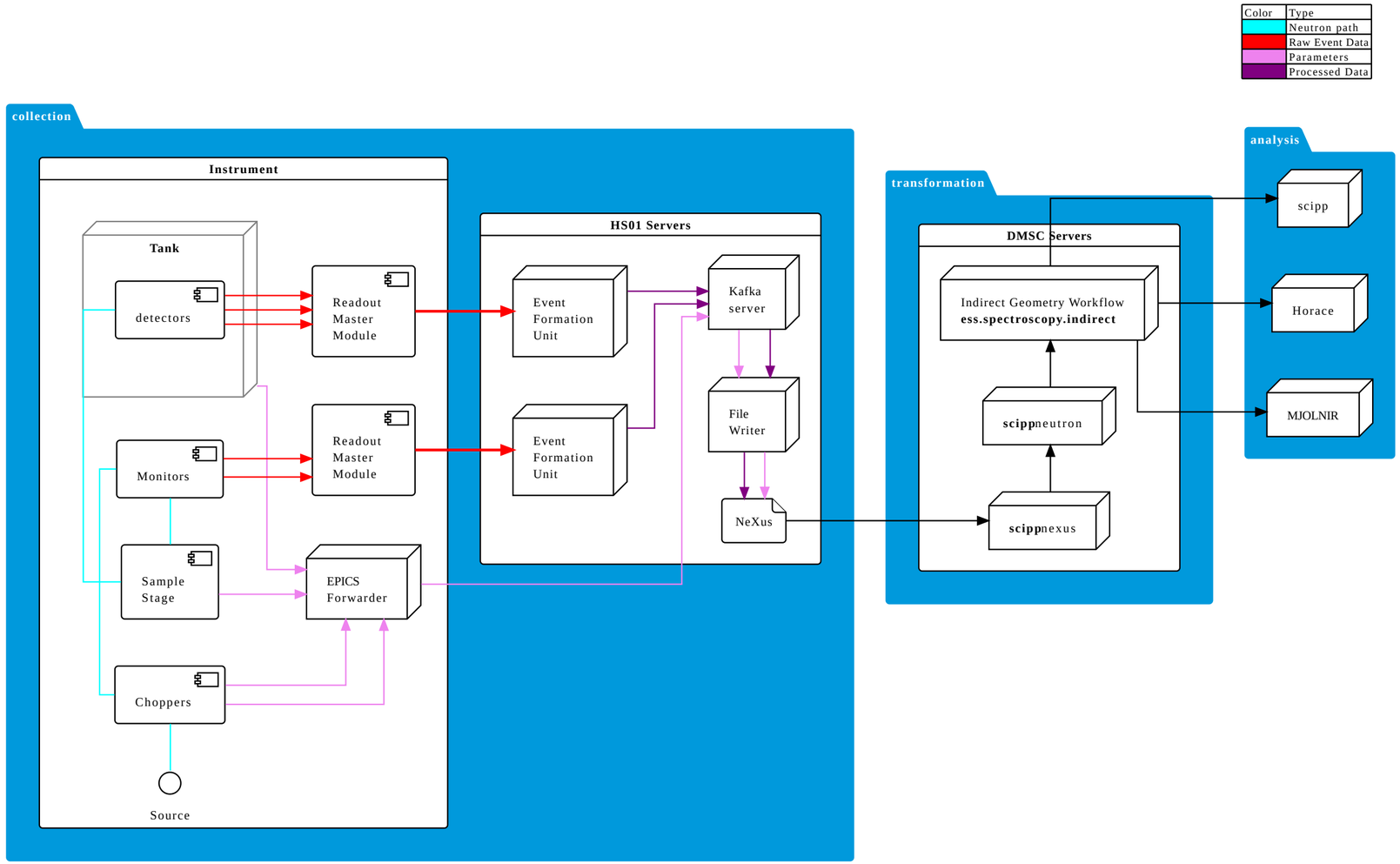
Simulations for Hot Commissioning

Gregory S. Tucker
DMSC, ESS

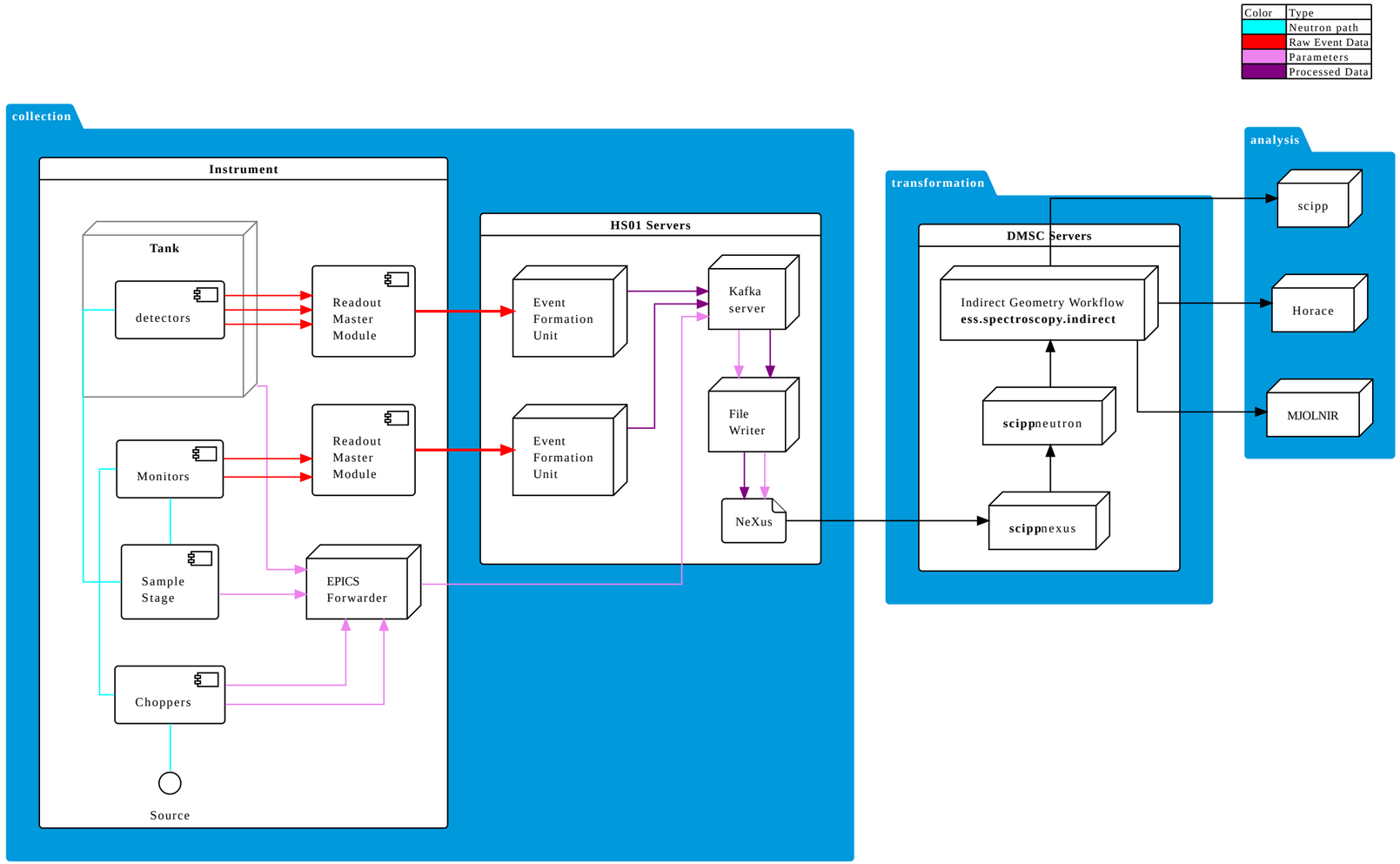
Overview

- Data Pipeline
- New Tools
  - Faithful Simulations
  - `fylgje`
- Simulated Hot Commissioning
  - Primary Flight Path
  - Detector Charge Division
  - Detector Scattering Angle

Data Pipeline



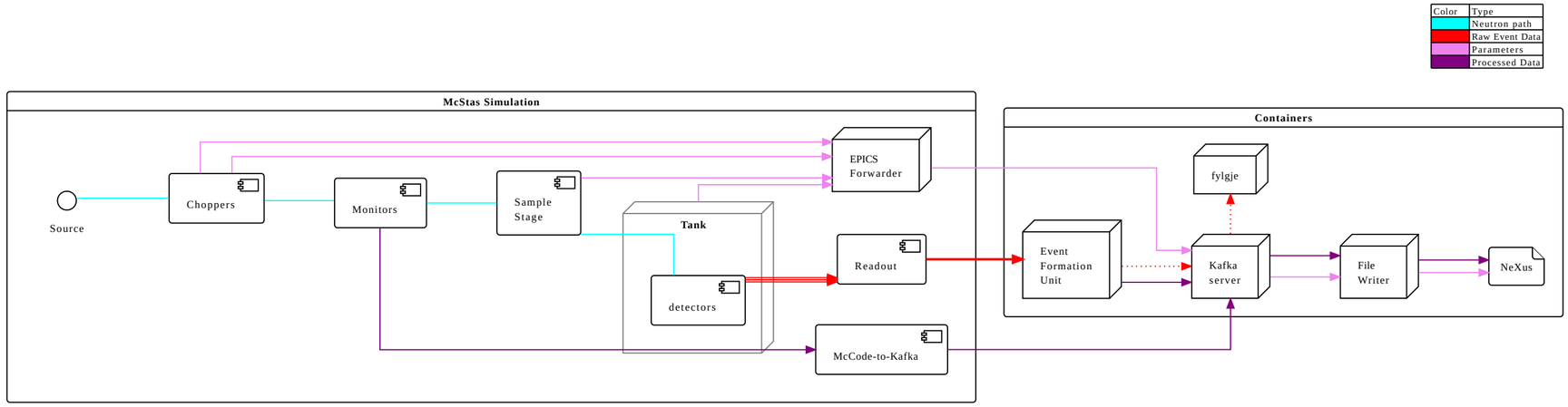
Data Pipeline



There are many hot commissioning (HC) tasks for BIFROST (see the previous presentation).

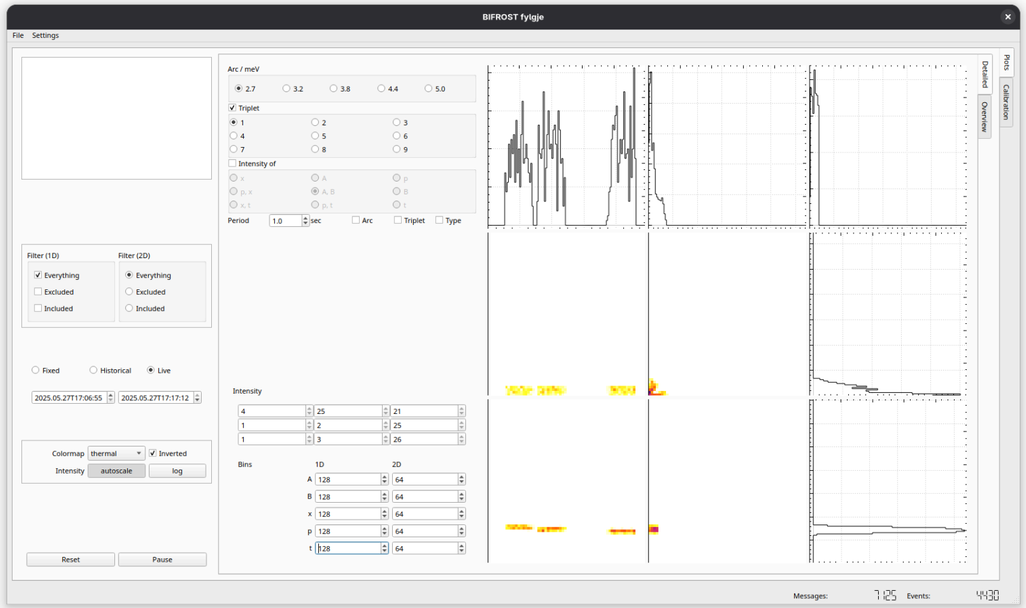
New tools

Faithful Simulations



- Mimic hardware and firmware in McStas
- Use same acqusition software stack
- Produce nearly indistinguishable output

fylgje

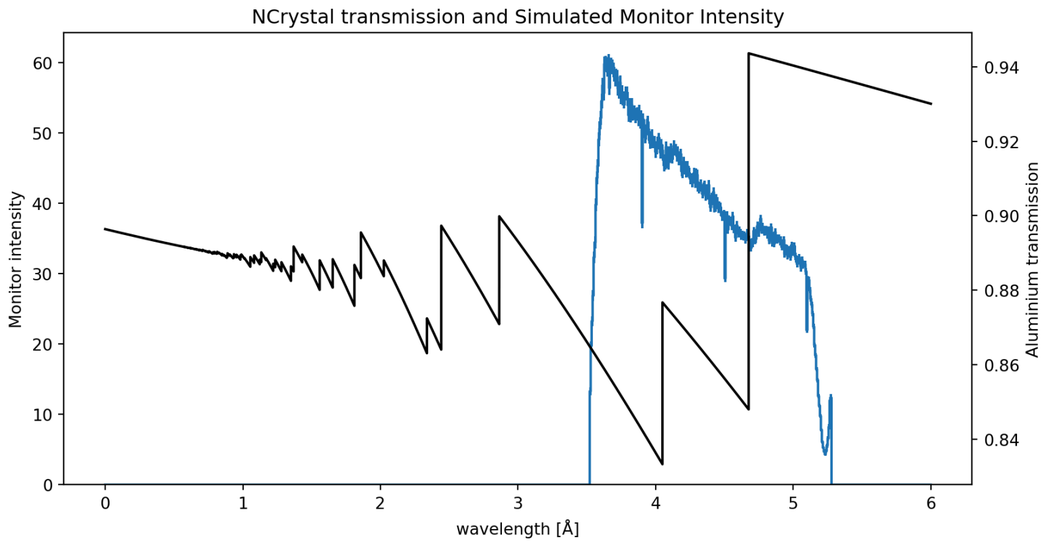


Incoherent elastic line simulation

- realtime detector follower
- recent historical data (limited to Kafka stream)
- various charge-division histograms
- output histograms and `AR51` messages to HDF5

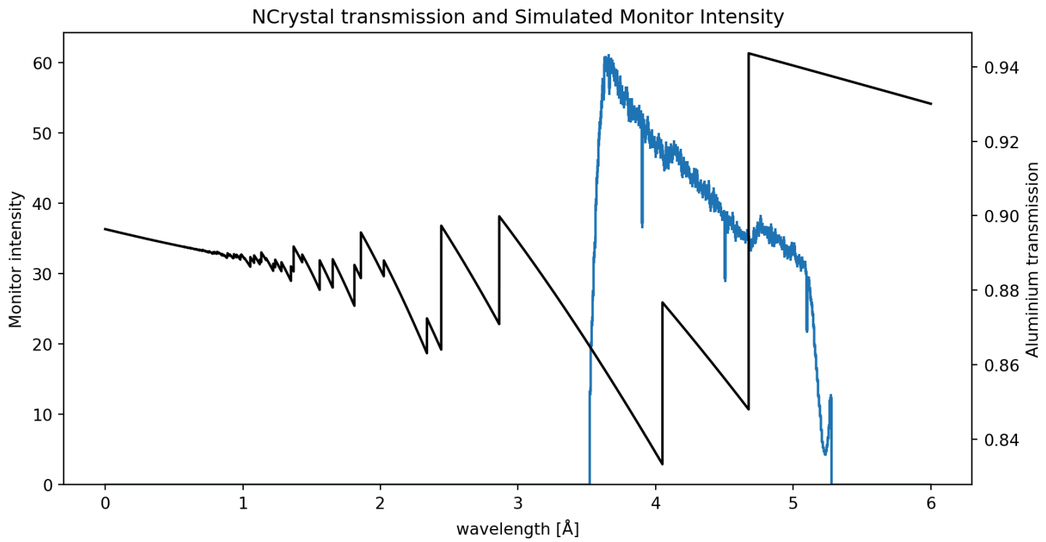
Primary Flight Path

Primary Flight Path Calibration



- Guide has 19 Al windows, 13 mm in total
- NCrystal for Al Bragg edge scattering
- Simulated 1.7 Å bandwidth

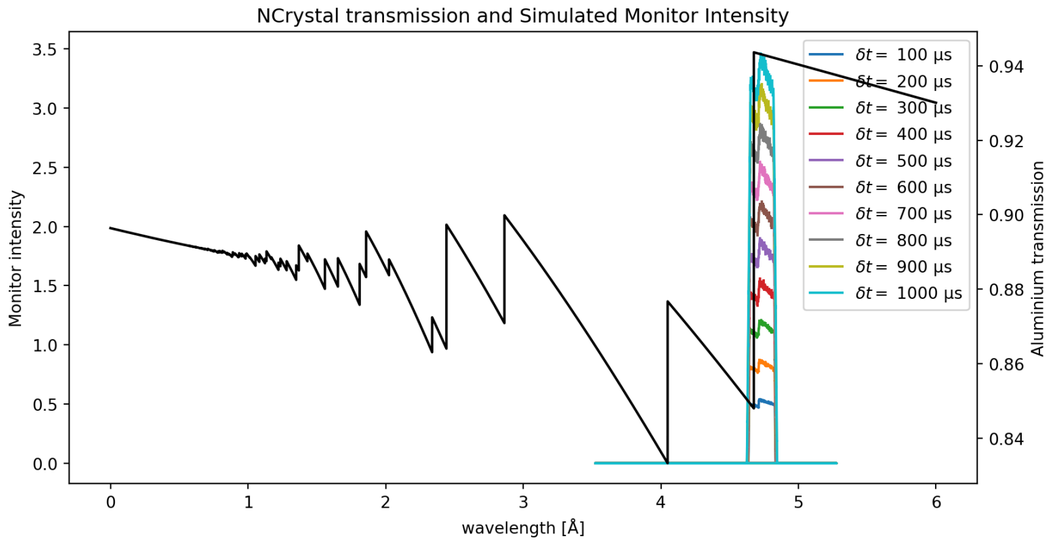
Primary Flight Path Calibration

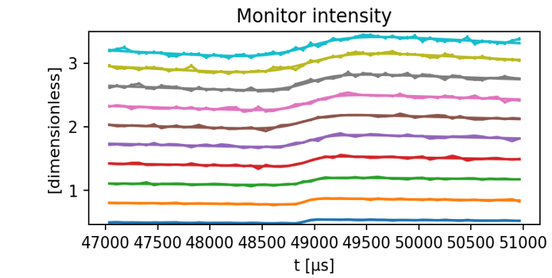


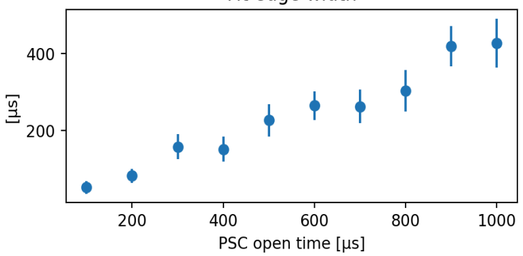
- Guide has 19 Al windows, 13 mm in total
- NCrystal for Al Bragg edge scattering
- Simulated 1.7 Å bandwidth

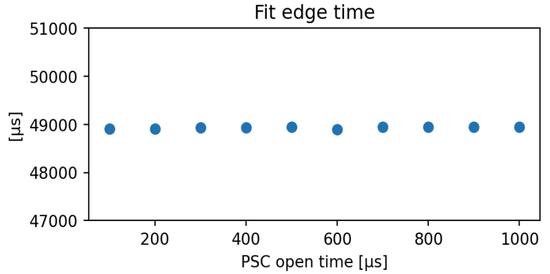
Focus bandwidth and vary PSC opening time

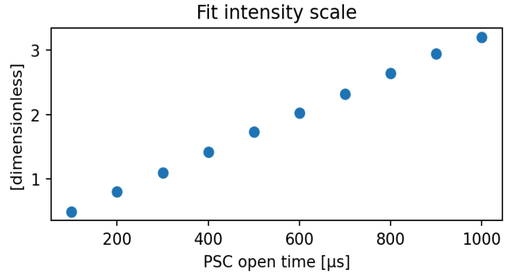
Primary Flight Path Simulations



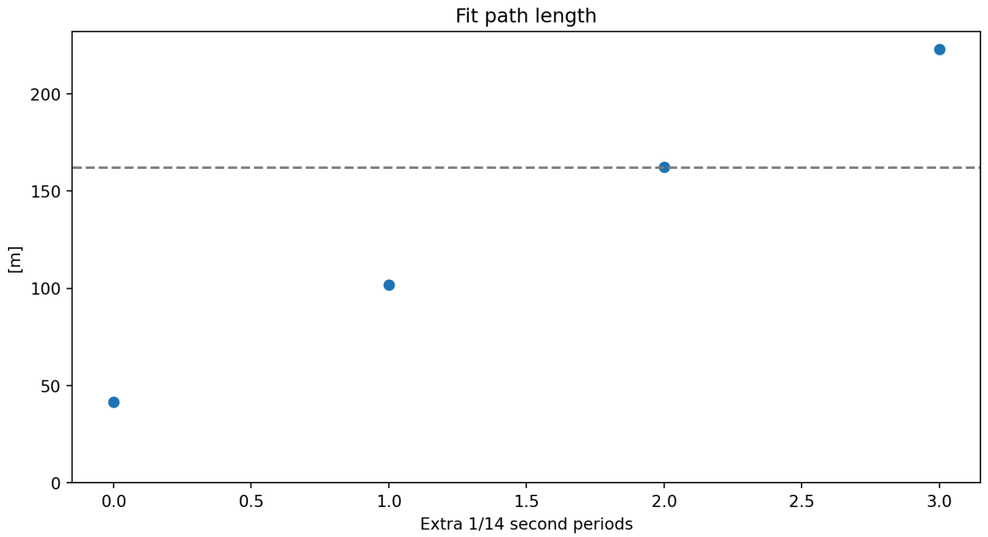








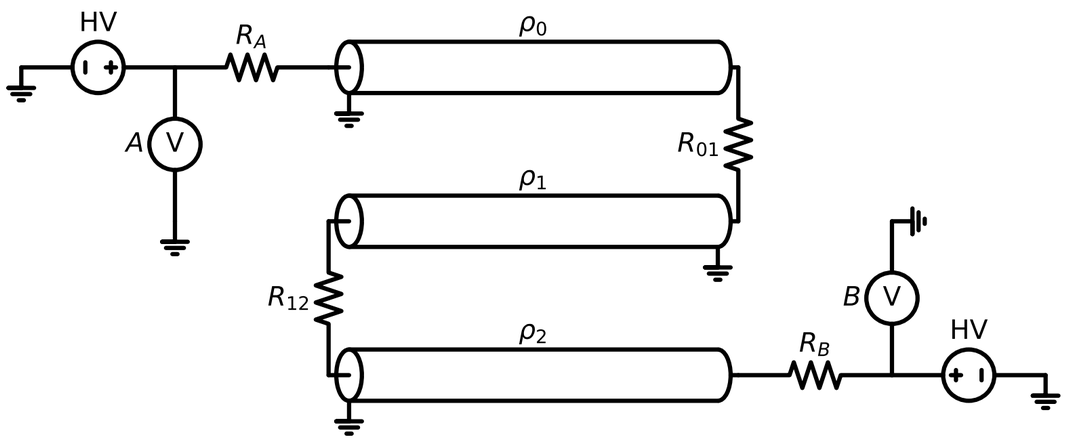
Primary Flight Path Fitting



- Use $\lambda \sim 4.676$ Å and edge-time ~ 49 ms to find path length
- $d = \frac{h}{m}\frac{t+N\tau}{\lambda}$
- $N = 2$ periods from generating pulse
- $\therefore d =$ 162.256(15) m

Detector Charge Division

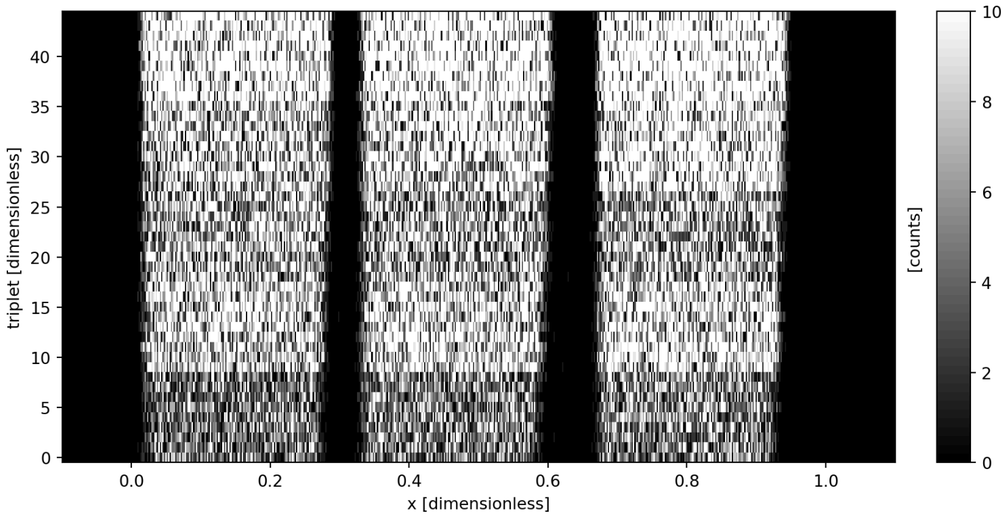
Detector Charge Division



BIFROST triplet

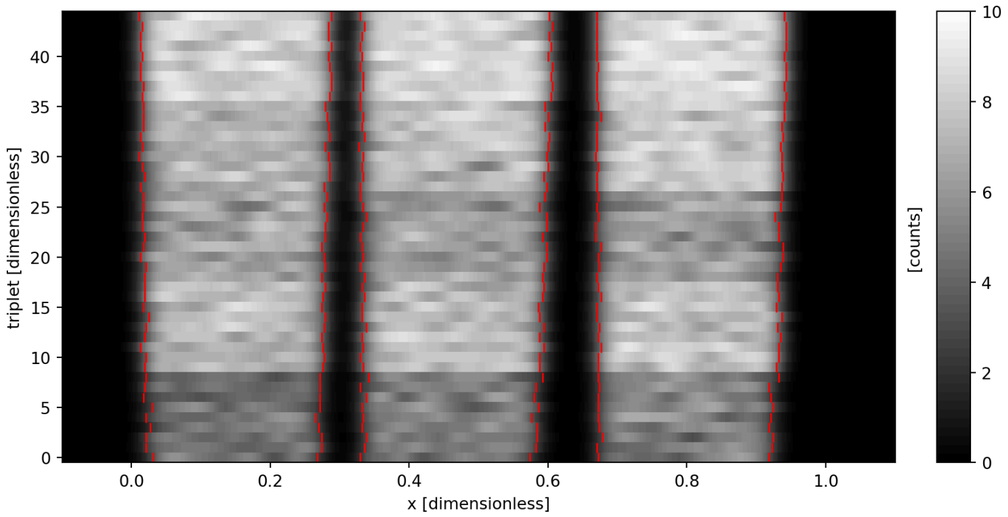
- 45 triplets
- A and B digitized pulse heights per event
- charge division $x = A/(A+B)$
- Event Formation Unit (EFU) needs 6 edges per triplet for pixelation

Detector Charge Division Boundaries



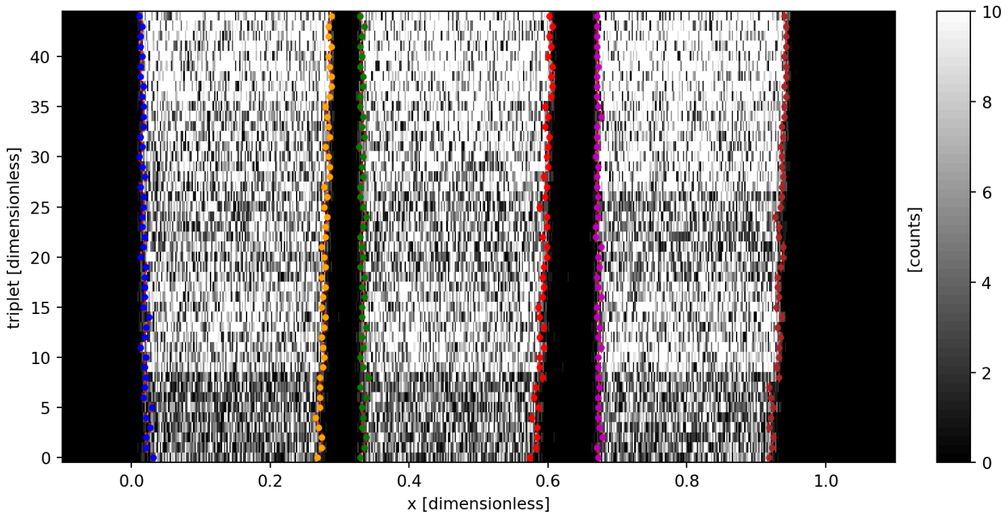
- Simulated incoherent elastic illumination
- `AR51` messages extracted with `fylgje`
- Need to find the 270 x boundary values

Detector Charge Division Smoothed



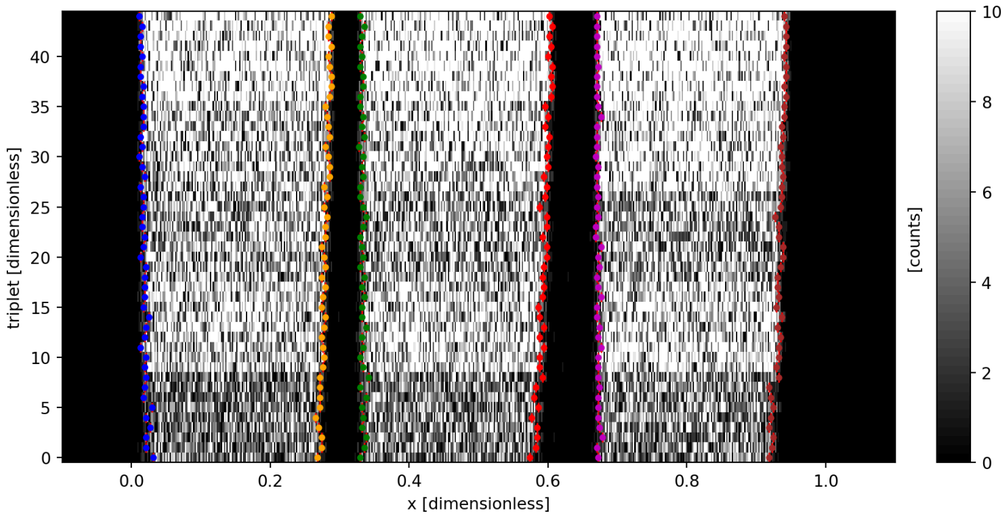
- `scipy` Gaussian filter, $\Delta x = 0.015$
- extract extrema in central derivative along x

Detector Charge Division Edge Utility



- basic `mpltoolbox` GUI
  - drag edges if derivative method failed
- output resulting positions to EFU calibration file

Detector Charge Division Edge Utility

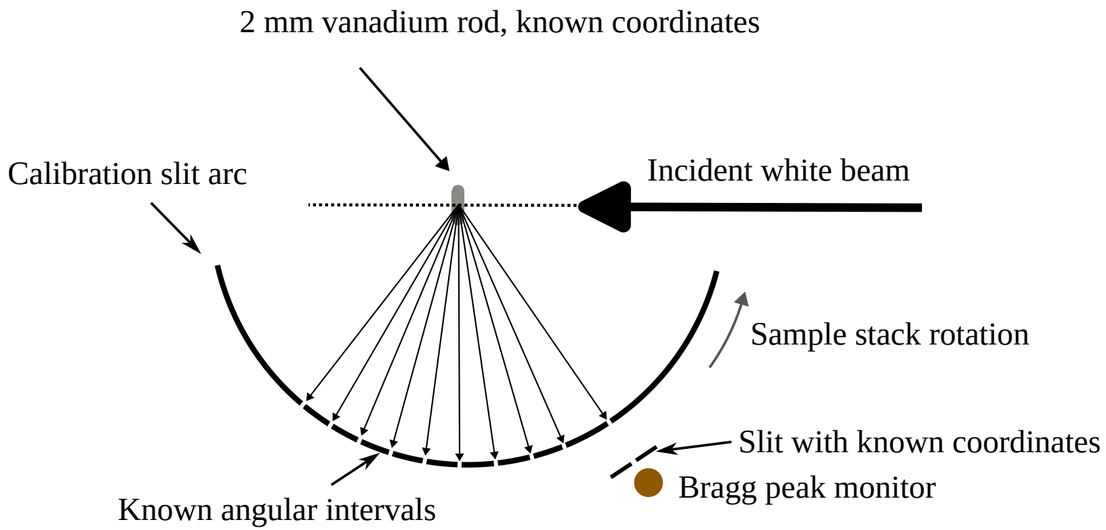


- basic `mpltoolbox` GUI
  - drag edges if derivative method failed
- output resulting positions to EFU calibration file


```
{
    "date": "2025-09-02T15:39:27.674693+00:00",
    "groups": 45,
    "info": "Calibration ranges extracted from smoothed data",
    "instrument": "bifrost",
    "groupsize": 3,
    "version": 0,
    "Parameters": [
        {
            "groupindex": 0,
            "intervals": [
                [
                    0.9186978297161938,
                    0.6722871452420702
                ],
                [
                    0.32971619365609356,
                    0.5741235392320536
                ],
                [
                    0.2676126878130218,
                    0.031218697829716216
                ]
            ],
            "polynomials": [
```

Detector Scattering Angle

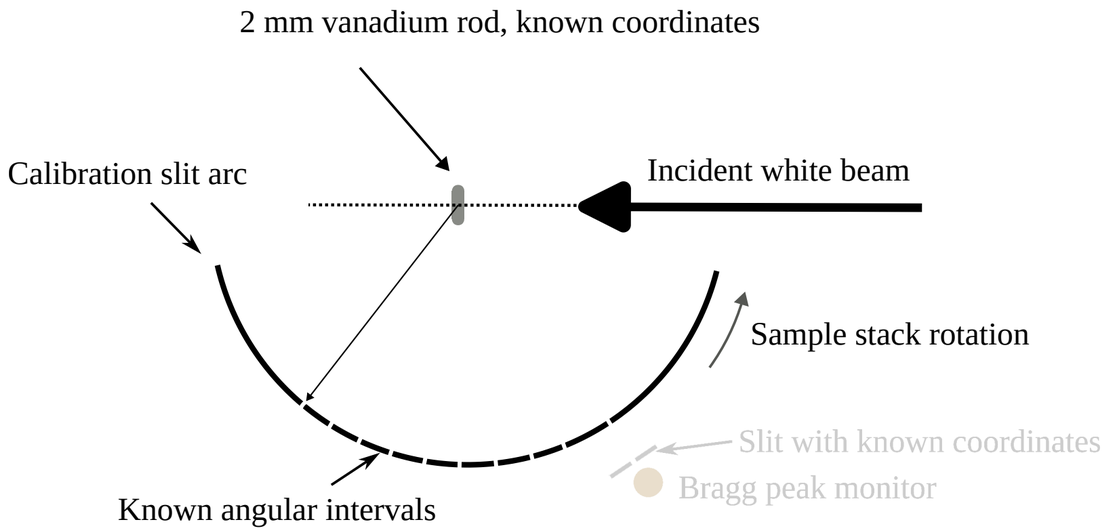
Detector Scattering Angle



courtesy Rasmus Toft-Petersen

- Incoherent elastic illumination
- Repeated slit between sample and analyzers
- Pixelation by EFU

Detector Scattering Angle Simulation



courtesy Rasmus Toft-Petersen

- *Simulated* incoherent elastic illumination
- Repeated slit between sample and analyzers
- Pixelation by EFU
- Scattering focused on one slit
- (No Bragg peak monitor)

Detector Scattering Angle Results

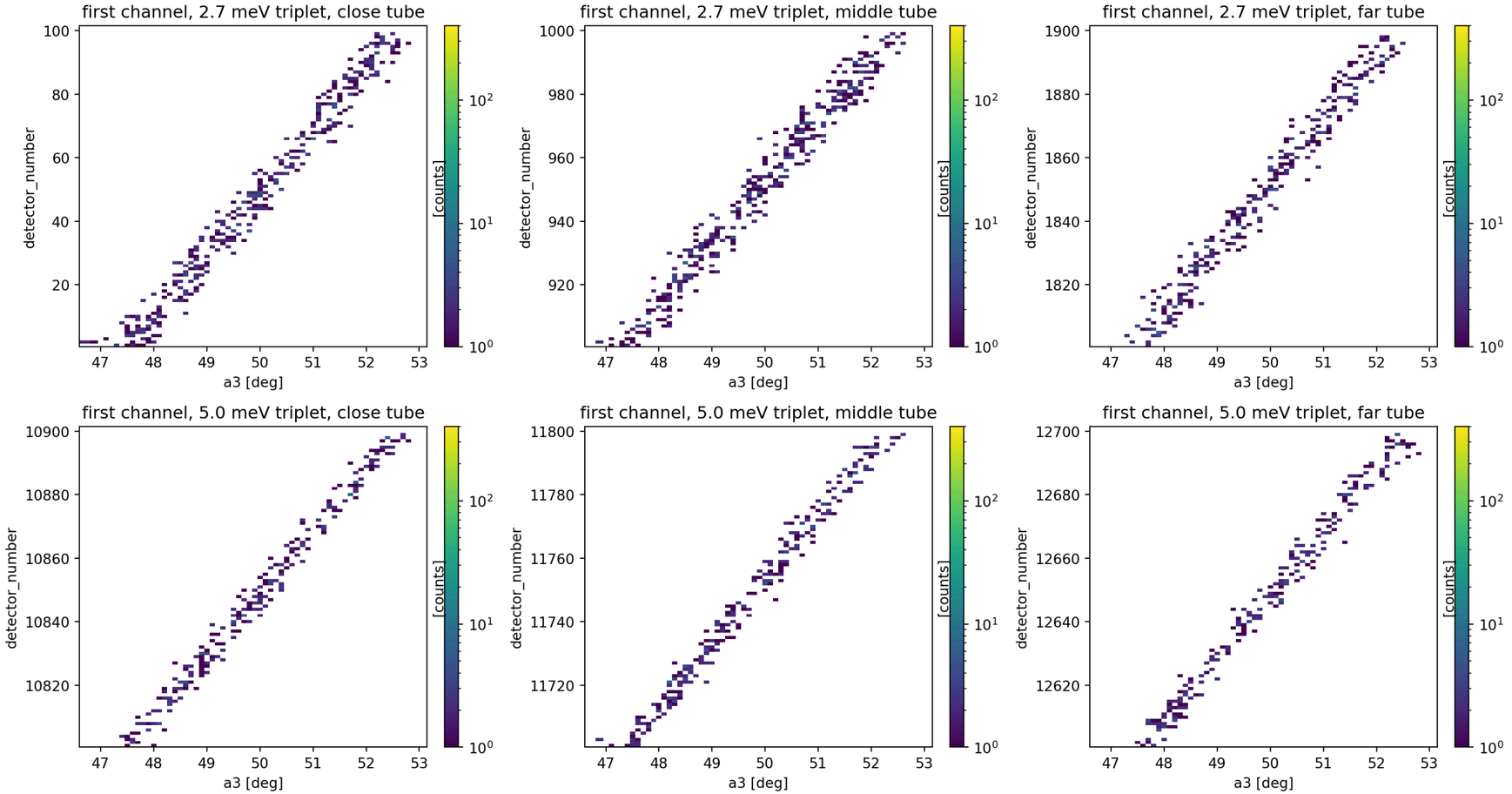
Data read from a `kafka-to-nexus` produced `NeXus` file

```
with scippnexus.File('bifrost_20250522T114526.h5') as file:
    dets = {k: v[...] for k, v in file['entry/instrument/'][snx.NXdetector].items()}
    a3 = file['entry/parameters/a3'][...]['value']
```

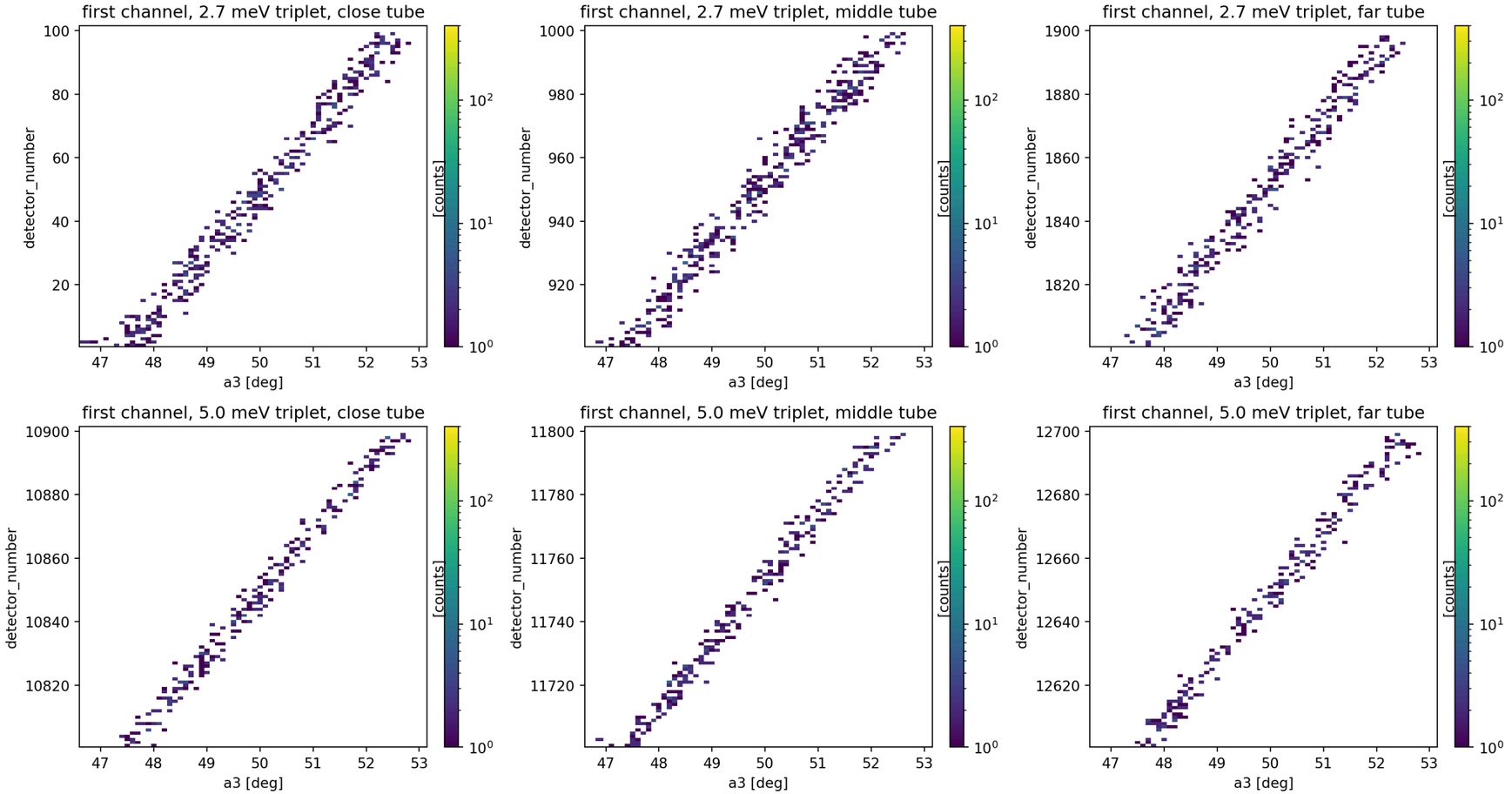
`a3` `NXlog` asigned to events

```
# Make the a3 look-up table
a3_lookup = scipp.lookup(a3, 'time')

# Combine the 45 detector groups' events
pixels = sc.sort(sc.concat([v['data'].group('detector_number') for v in dets.values()], 'detector_number'), 'detector_number'

def event_time(event_time_zero, event_time_offset):
    return event_time_zero + event_time_offset

def lookup_a3(time):
    return a3_lookup[time]

graph = {'a3': lookup_a3, 'time': event_time}

# Add the a3 coordinate to each event based on its detection time
pixels = pixels.transform_coords('a3', graph=graph).group('detector_number', 'a3')
```

Detector Scattering Angle Results



Detector Scattering Angle Results

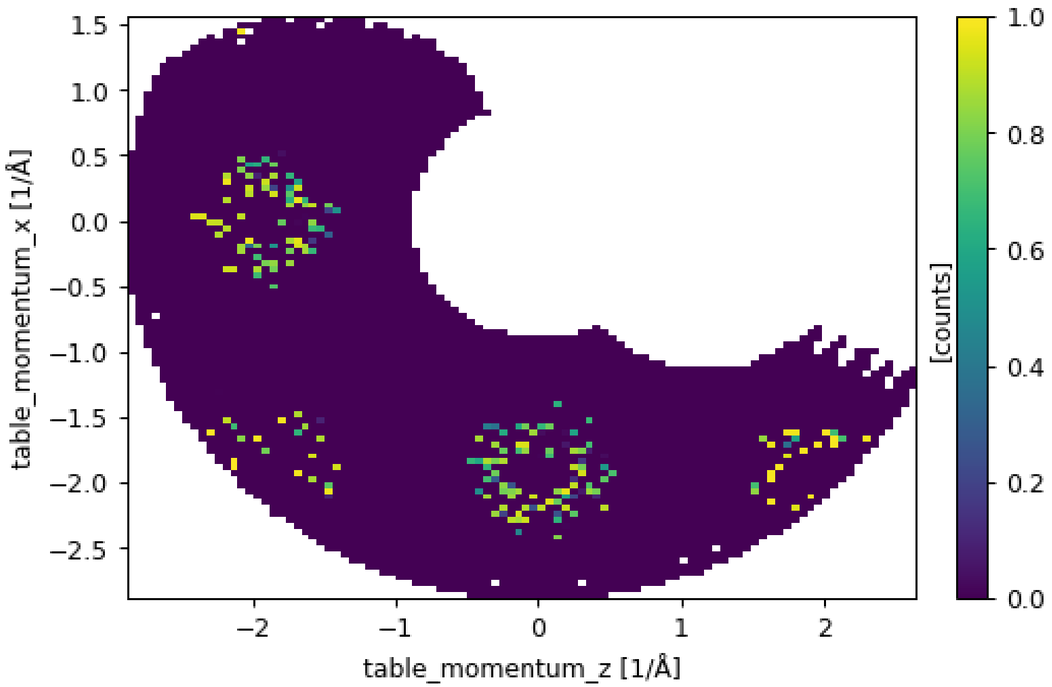


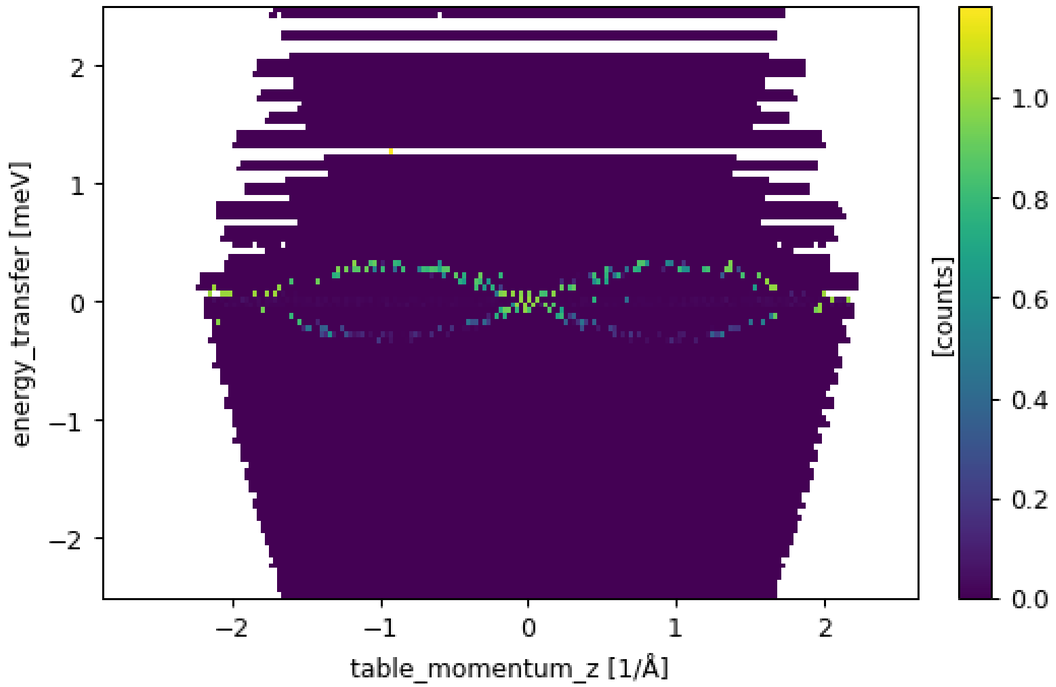
With more events, fit the per-pixel a3 dependence to extract its effective a4.

Then find any non-linear correction terms per tube for EFU calibration

Intensity Normalization

Events must be normalized by monitor intensity to be proportional to $\frac{\partial^2\sigma}{\partial E \partial \Omega}$.





Work in progress

Summary

- New tools enable simulating and retreiving needed HC data
- A handful of HC tasks have been simulated
  - No major problems analyzing HC data
- More can be done in preparation for HC and beyond
  - further HC task simulations
  - full simulated experiments
  - beyond the data pipeline tests
  - user-experience improvements