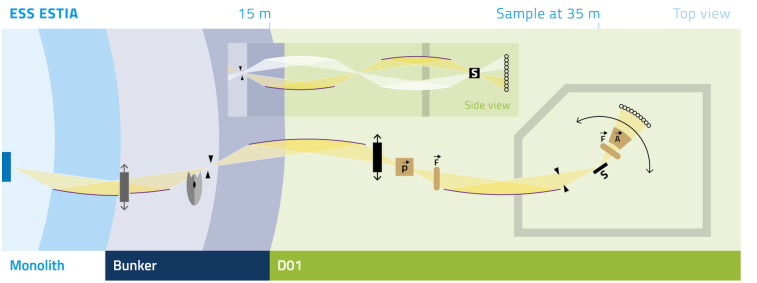
ESTIA data reduction

Speaker: Johannes Kasimir, data reduction group at DMSC

ESTIA characteristics

- Focusing reflectometry.
- Fixed footprint on sample.
- Polarization.



Source: https://ess.eu/instruments/estia#instrument-description.

ESTIA data reduction overview

Supermirror normalization

- The neutron beam intensity is inhomogeneous over wavelength and divergence angle.
- Detector pixel efficiencies can vary.
- → Supermirror sample with known reflectivity is used to normalize the measured intensity and correct for the inhomogeneity.

Virtual source slit correction

- The "virtual source slit" reduces the beam width with the beam incident angle.
  - This keeps the footprint on the sample fixed for the entire divergent beam.
- → Sample sees relatively fewer events from low incident angles.
  Requires correction in the data reduction.

Implementation: `scipp`, `scippnexus`, `scippneutron`

- `essreflectometry` is built on top of the `scipp` data array library, like most ESS data reduction workflows.
- Few surprises for users familiar with other data reduction tools at ESS.
- Events are kept throughout processing.

scipp.github.io/ess/reflectometry

scipp.github.io

scipp.github.io

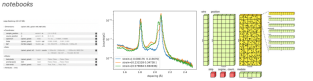
scipp

Getting Started User Guide API Reference Development About More

On this page
Lost? New to Scipp? Start Here!
Where can I get help?

Scipp – Multi-dimensional data arrays with labeled dimensions

A Python library enabling a modern and intuitive way of working with scientific data in Jupyter notebooks



Scipp is heavily inspired by Xarray. It enriches raw NumPy-like multi-dimensional arrays of data by adding named dimensions and associated coordinates. Multiple arrays can be combined into datasets. While for many applications Xarray is more suitable and matured than Scipp, there is a number of features missing in other situations. If your use case requires one or several of the items on the following list, using Scipp may be worth considering:

- **Physical units** are stored with each data or coord array and are handled in arithmetic operations.
- **Histograms**, i.e., **bin-edge axes**, which are by 1 longer than the data extent.
- Support for non-regular or scattered data and **non-destructive binning**.
- Support for **masks stored with data**.
- **Propagation of uncertainties**.
- Internals written in C++ for better performance (for certain applications), in combination with Python bindings.

Generic functionality of Scipp is provided in the **scipp** Python package. In addition, more specific functionality is made available in other packages. Examples for this are Plopp for data visualization, ScippNexus for loading NeXus files, and ScippNeutron for handling data from neutron-scattering experiments.

Lost? New to Scipp? Start Here!

The **Getting Started** section motivates Scipp in What is Scipp?, provides installation instructions, and gives a brief overview in Quick start.

The **User Guide** provides a high-level overview of the most important Scipp concepts and features. Read Data Structures, Slicing, and Computation (in this order) to develop an understanding of the core concepts of Scipp.

Further sections of the User Guide are optional and can be studied in arbitrary order. Depending on your area of application you may be most interested in Binned Data, Coordinate Transformations, GroupBy, or Masking. The combination of these features is what sets Scipp apart from other Python libraries. Make sure to also check out Representations and Tables and Plotting for an overview of Scipp's powerful visualization features.

The **Reference** documentation section provides a detailed listing of all functions and classes of Scipp, as well as some more technical documentation. If you are looking for something in particular, use the **Search the docs** function in the left navigation panel.

Where can I get help?

We strive to keep our documentation complete and up-to-date. However, we cannot cover all use-cases and questions

This Page

- Show Source

scipp.github.io

essreflectometry

User Guide API Reference Development About More

Section Navigation

- Installation
- Amor
- ESTIA
  - Reduction of ESTIA McStas data
  - ESTIA advanced data reduction
  - Reduction of ospi file data from McStas simulation
  - Create a wavelength lookup table for ESTIA
- FREIA
- Offspec
- Focused reflectometry data reduction

On this page
Create and configure the workflow
Use the reduction workflow
Saving to ORSO ORT file

Show Source

User Guide > ESTIA > Reduction of ESTIA McStas data

Reduction of ESTIA McStas data

- Audience: Instrument users, beginners
- Prerequisites: Basic knowledge of Scipp

This notebook demonstrates the basic reflectometry data reduction workflow for ESTIA with simulated data. A workflow for data recorded at ESS would be very similar but is not yet available. The workflow:

1. Converts the data to momentum transfer Q.
2. Normalizes by a reference measurement.
3. Packages the results to be saved to an ORSO ORT file.

The data is available through the ESSreflectometry package but accessing it requires the pooch package. If you get an error about a missing module pooch, you can install it with `!pip install pooch`.

```
[1]: import scipp as sc

     from ess.estia import data
     from ess.estia import EstiaMcStasWorkflow
     from ess.reflectometry.types import *
```

Create and configure the workflow

We begin by creating the ESTIA (McStas) workflow object which is a skeleton for reducing ESTIA data with pre-configured steps:

```
[2]: wf = EstiaMcStasWorkflow()
```

We then need to set the missing parameters which are specific to each experiment (the keys are types defined in ess.reflectometry.types).

```
[3]: # Specify input data files:
     # (The choice of '96cf7b-multilayer' file number 3 is arbitrary.)
     wf[Filename[SampleRun]] = data.estia_mcstas_sample_example('96cf7b-multilayer')[3]
     wf[Filename[ReferenceRun]] = data.estia_mcstas_reference_example()

     # Select a region of interest:
     wf[YIndexLimits] = sc.scalar(25), sc.scalar(64)
     wf[ZIndexLimits] = sc.scalar(0), sc.scalar(48 * 32)
     wf[BeamDivergenceLimits] = sc.scalar(-0.75, unit='deg'), sc.scalar(0.75, unit='deg')

     # Configure the binning of intermediate and final results:
     wf[WavelengthBins] = sc.geomspace('wavelength', 3.5, 12, 401, unit='angstrom')
     wf[QBins] = 400

     wf[LookupTableFilename] = data.estia_wavelength_lookup_table()

     # There is no proton charge data in the McStas files, here we just add some fake proton charge
     # data to make the workflow run.
     wf[ProtonChargeSampleRun] = sc.DataArray(
         sc.array(dims=['time',], values=[1], unit='uC'),
         coords={'time': sc.array(dims=['time',], values=[1, 2], unit='s')})
     wf[ProtonChargeReferenceRun] = sc.DataArray(
         sc.array(dims=['time',], values=[2], unit='uC'),
         coords={'time': sc.array(dims=['time',], values=[1, 2], unit='s')})
```

We can visualize the workflow as a graph. This can help us understand how the data will be reduced. (`ReflectivityOverQ` is the key of the output data.)

```
[4]: wf.visualize(ReflectivityOverQ, graph_attrs={'rankdir': "LR"})
```

Interface: `essreflectometry`

```
from ess.reflectometry.estia import EstiaWorkflow
wf = EstiaWorkflow()
```

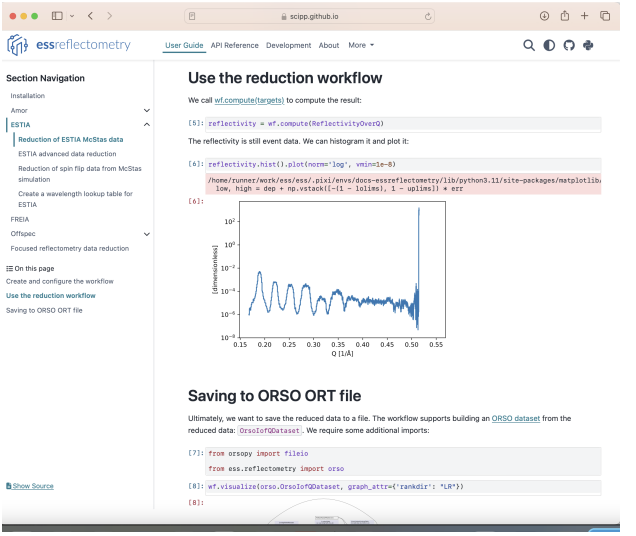
`EstiaWorkflow` is a "Sciline workflow"

A Sciline workflow contains "recipes" for computing quantities of interest:

- Reflectivity as a function of `Q`, `theta`, or `wavelength`.
- Raw detector event data.
- Histograms over time of flight or detector pixels.
- etc.

Example

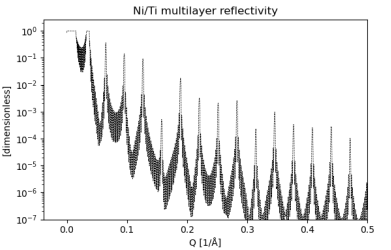
```
wf[Filename[SampleRun]] = 'estia_192424_00000001.hdf'
... # set more parameters, ROI, reference measurement filename, etc.
result = wf.compute(ReflectivityOverQ)
```


scipp.github.io
essreflectometry
User Guide
API Reference
Development
About
More
Section Navigation
Installation
Amor
ESTIA
Reduction of ESTIA McStas data
ESTIA advanced data reduction
Reduction of spin flip data from McStas simulation
Create a wavelength lookup table for ESTIA
FREIA
Offspec
Focused reflectometry data reduction
On this page
Create and configure the workflow
Use the reduction workflow
Saving to ORSO ORT file
Show Source
Use the reduction workflow
We call wf.compute(targets) to compute the result:
[5]: reflectivity = wf.compute(ReflectivityOverQ)
The reflectivity is still event data. We can histogram it and plot it:
[6]: reflectivity.hist().plot(norm='log', vmin=1e-8)
/home/runner/work/ess/ess/.pixi/envs/docs-essreflectometry/lib/python3.11/site-packages/matplotlib
low, high = dep + np.vstack((-(1 - lolims), 1 - uplims)) * err
[6]:
[dimensionless]
Q [1/A]
Saving to ORSO ORT file
Ultimately, we want to save the reduced data to a file. The workflow supports building an ORSO dataset from the reduced data: OrsoIofQDataset. We require some additional imports:
[7]: from orsopy import fileio
from ess.reflectometry import orso
[8]: wf.visualize(orso.OrsoIofQDataset, graph_attr={'rankdir': 'LR'})
[8]:

Demonstration

Experiment setup

- ESTIA experiment on Ni/Ti multilayer sample was simulated with McStas
  - One measurement on a supermirror (for reference).
  - 6 measurements on Ni/Ti sample at sample rotations from 1° to 7°.



Experiment setup (cont.)

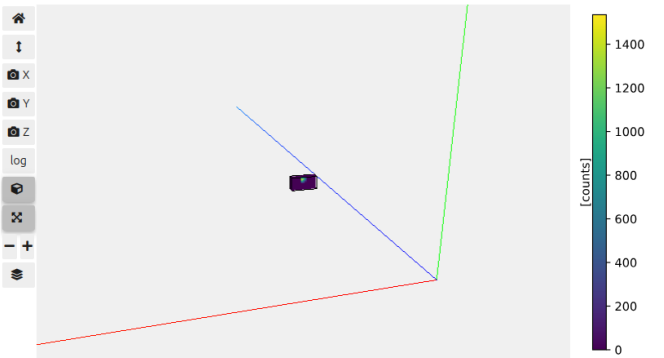
- McStas output was sampled and events were written to ESS NeXus files.
- → **Input file format identical to what is written by ESTIA during measurement.**
- Data in the NeXus files were reduced using `essreflectometry`.

Goals

1. Visualize the raw data in the detector.
2. Compute a reflectivity curve for each of the 6 measurements at different angles.
3. Combine the reflectivity curves into a single curve.
4. Compare the result to the known "ground truth" reflectivity curve.

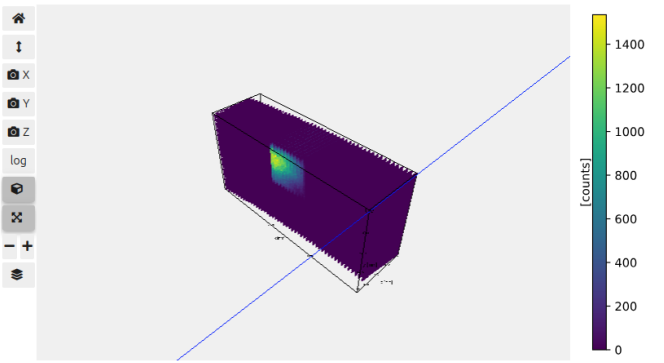
Interactive detector view

```
from scippneutron import instrument_view

instrument_view(wf.compute(RawDetector[ReferenceRun]).hist(), size=3)
```

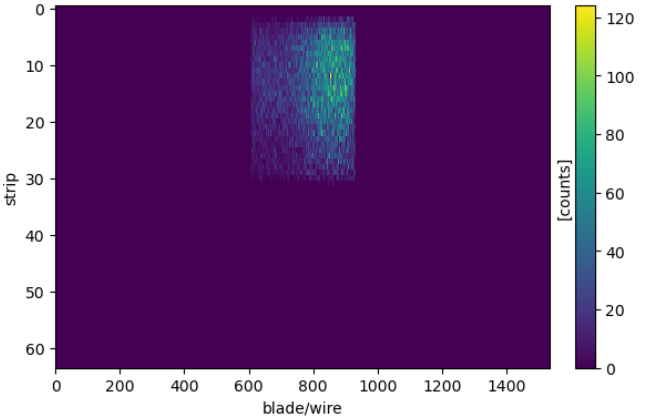


Interactive detector view

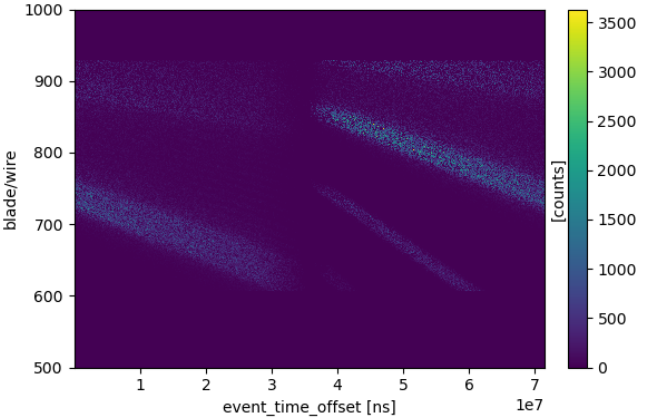
```
from scippneutron import instrument_view

instrument_view(wf.compute(RawDetector[ReferenceRun]).hist(), size=3)
```



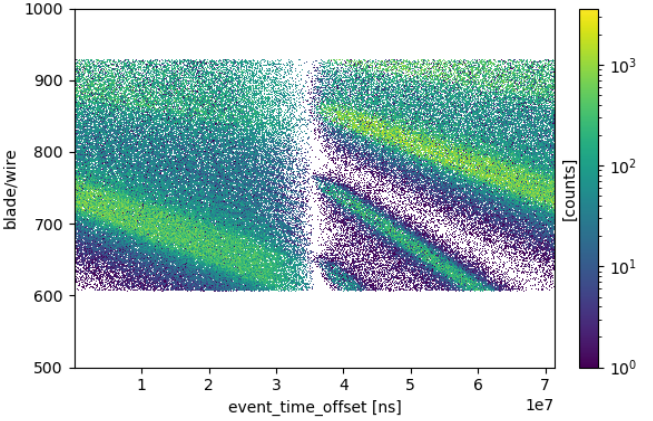
Raw detector data



Event-time detector view



Event-time detector view - log scale



Note: The theoretical reflectivity curve of sample does not account for instrument resolution.

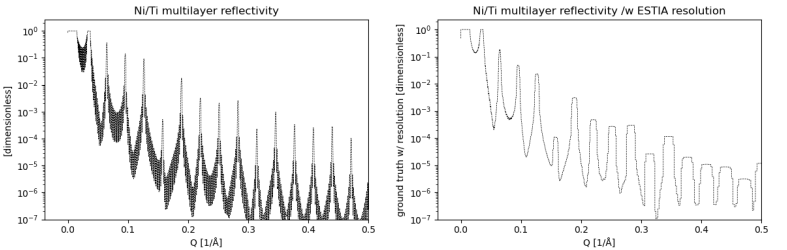
Resolution is taken into account by convolving the reflectivity with a resolution function.

For ESTIA these factors dominate the resolution:

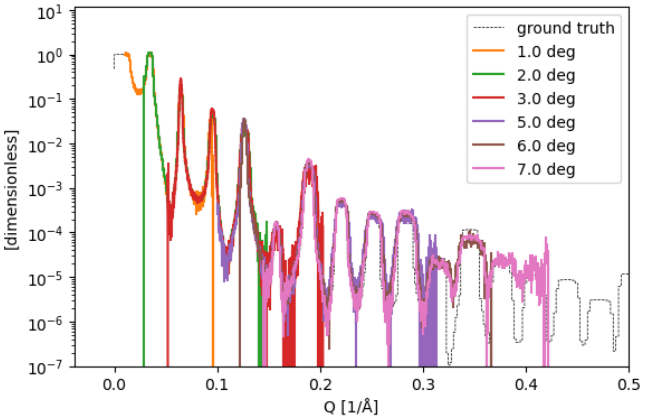
- Wavelength resolution.
- Detector spatial resolution.
- Footprint size on sample.

The resolution function associated with the wavelength uncertainty is close to a rectangular pulse.

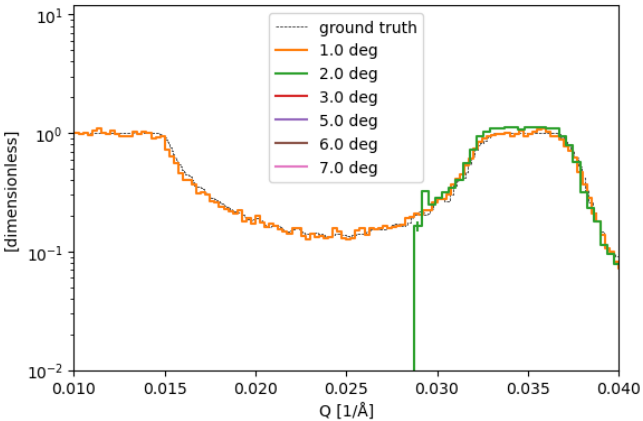
To get a quantity that is comparable to the computed reflectivity curve **a rectangular resolution function is applied to the theoretical reflectivity curve.**



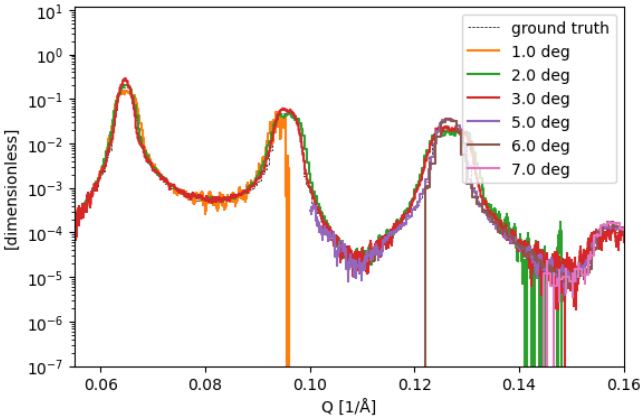
Reflectivity curves



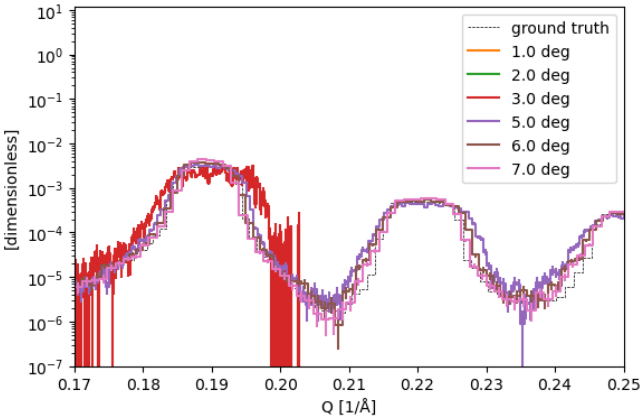
Critical edge



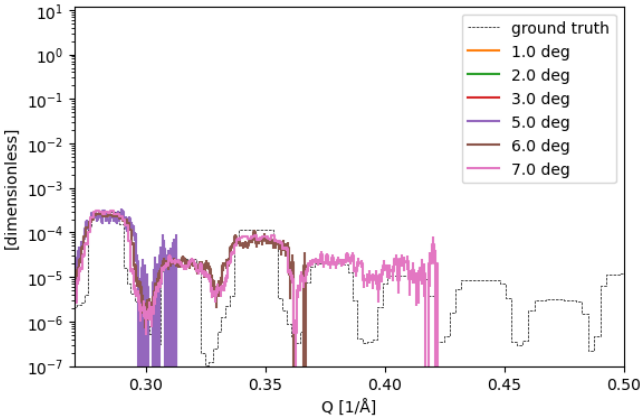
Low Q


ground truth
1.0 deg
2.0 deg
3.0 deg
5.0 deg
6.0 deg
7.0 deg
[dimensionless]
Q [1/Å]
10^{1}
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^{-6}
10^{-7}
0.06
0.08
0.10
0.12
0.14
0.16

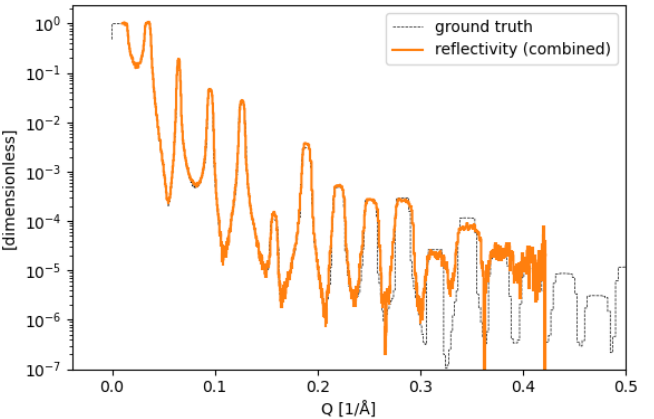
Medium Q


ground truth
1.0 deg
2.0 deg
3.0 deg
5.0 deg
6.0 deg
7.0 deg
[dimensionless]
Q [1/Å]
10^{1}
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^{-6}
10^{-7}
0.17
0.18
0.19
0.20
0.21
0.22
0.23
0.24
0.25

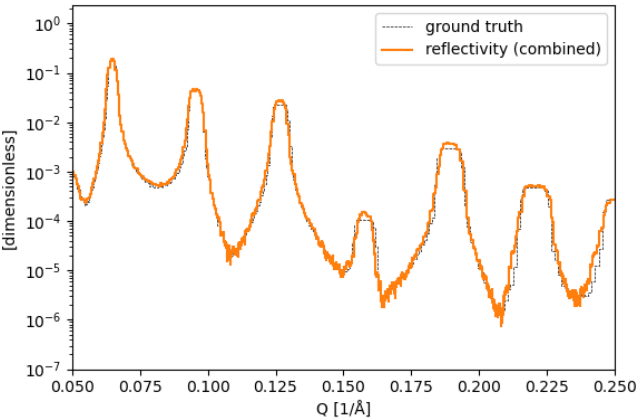
High Q


ground truth
1.0 deg
2.0 deg
3.0 deg
5.0 deg
6.0 deg
7.0 deg
[dimensionless]
Q [1/Å]
10^{1}
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^{-6}
10^{-7}
0.30
0.35
0.40
0.45
0.50

Combined reflectivity



Combined reflectivity, zoomed



Conclusion

The theoretical and computed reflectivity curves match well.

Current limitations

- No background subtraction.
- Off-specular data reduction prototypical.
- Data reduction tools mostly geared toward “expert users”.

Implemented but not shown here

- Polarized data reduction.
- Data reduction for time dependent reflectivity.

Polarization

- Wavelength-dependent correction factors
  - associated with polarizing supermirrors (polarizer and analyzer),
  - calibrated by measuring two calibration samples,
  - with each polarization setting.
  - Sample 1: Non-magnetic supermirror.
    Sample 2: Magnetic supermirror.
- Calibration has been implemented and tested on McStas data.
- Polarized data reduction has been implemented and tested on McStas data.

- Implementation uses generic procedures for polarized data reduction that will be used for most ESS instruments with polarization.

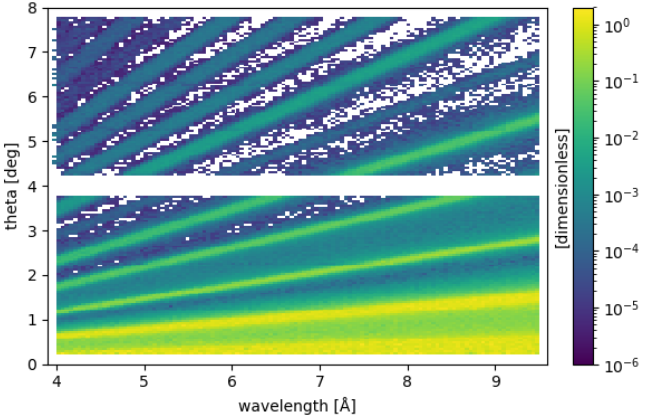
Visualizations for troubleshooting

Reflectivity is expected to depend only on Q.

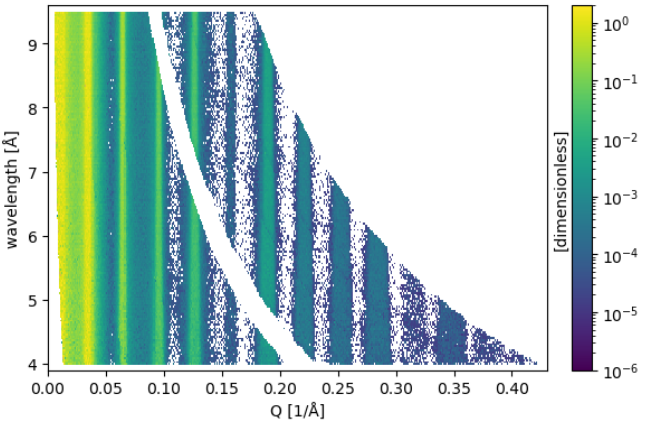
To verify that this is the case, it is useful to visualize the reflectivity as a function of:

- θ (angle of reflection) and λ
- Q and λ

Reflectivity over wavelength and angle of reflection



Reflectivity over Q and wavelength



Questions?