EUROPEAN
SPALLATION
SOURCE

SANS Data Reduction at the ESS

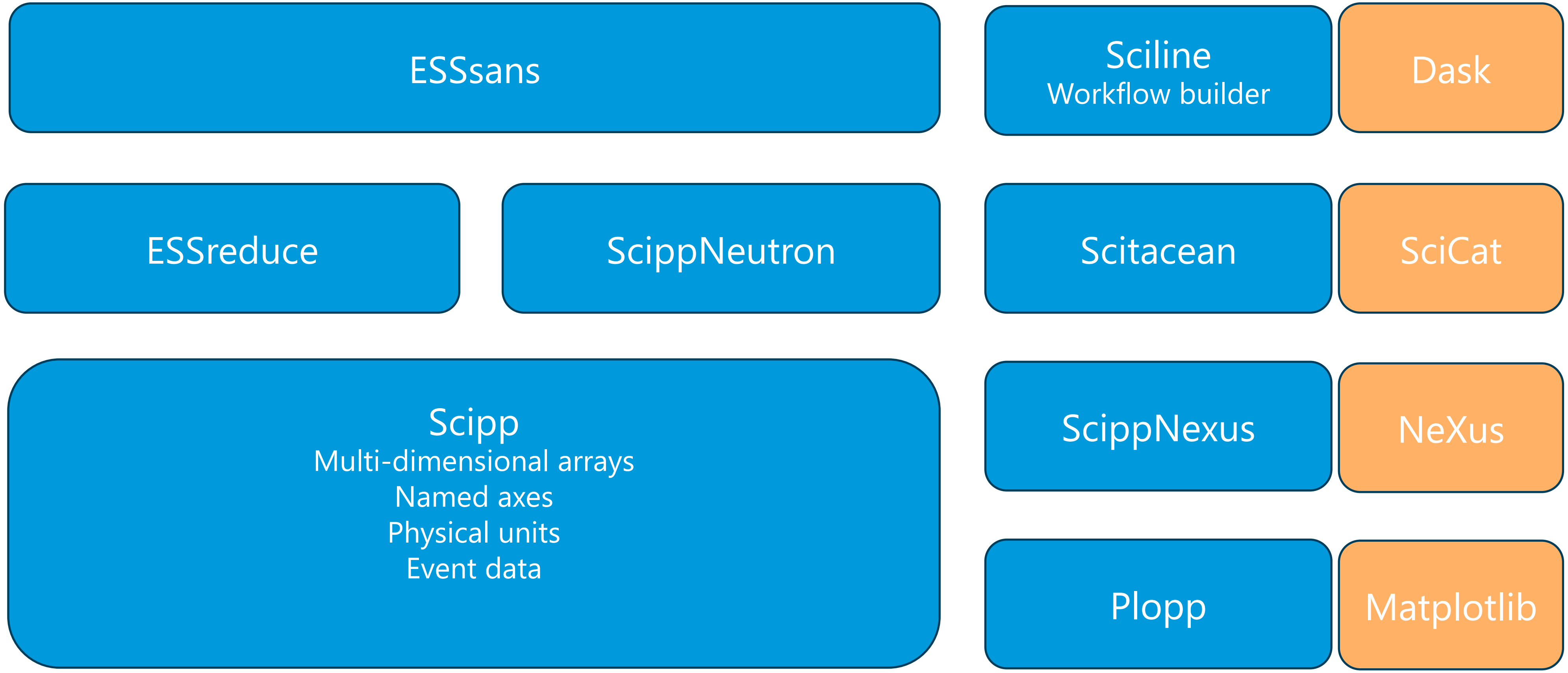
ESSsans and the Scipp software stack

PRESENTED BY SIMON HEYBROCK (ESS/DMSC)

2024-09-06

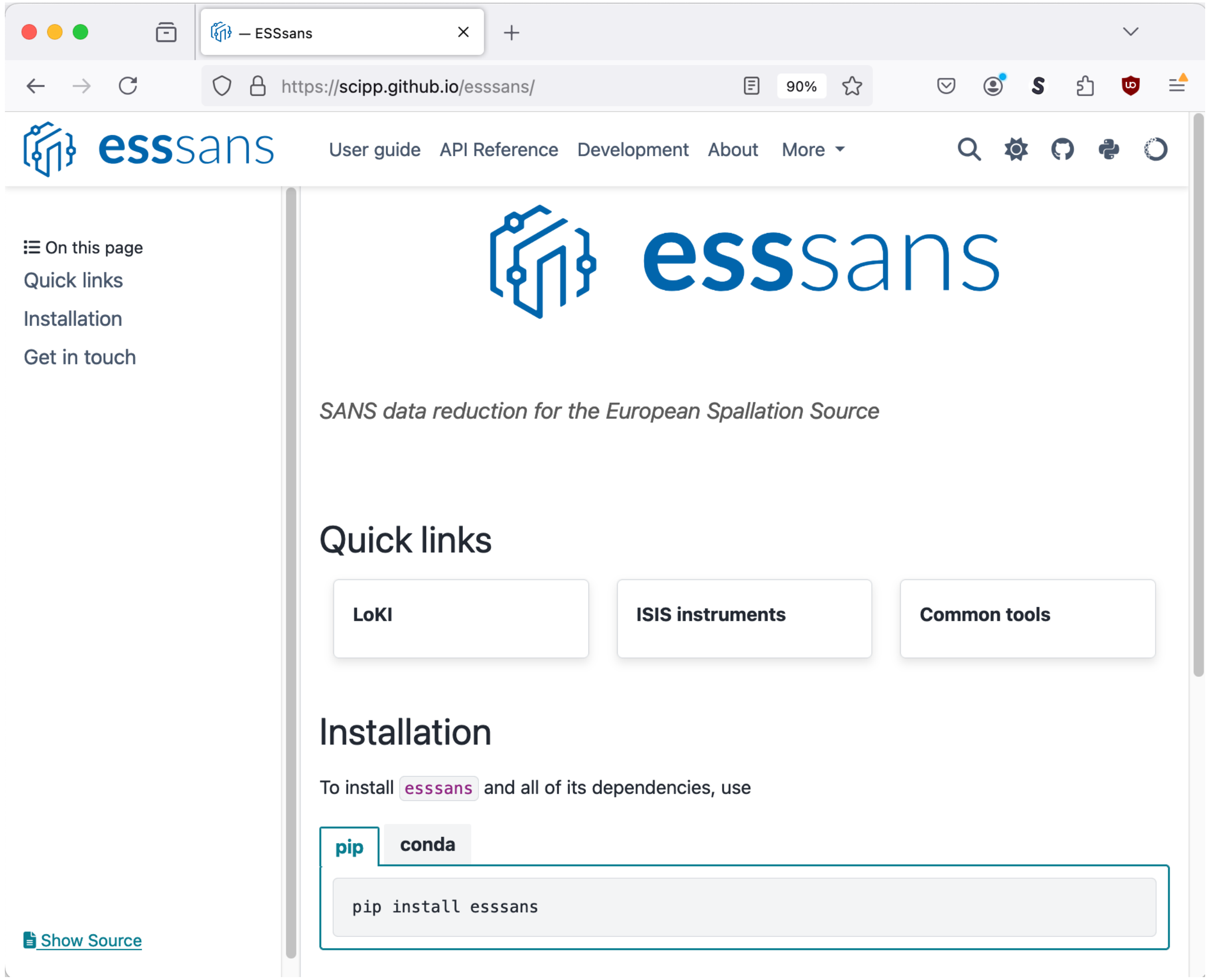
ESS software stack for SANS

Ecosystem around Scipp / Interfaces to the outside world



ESSsans

The ESS Python package for SANS data reduction



Main contributors:
- Neil Vaytet
- Simon Heybrock
- Wojciech Potrzebowski

Levels of interaction

Spectrum of accessibility vs flexibility

Flexibility

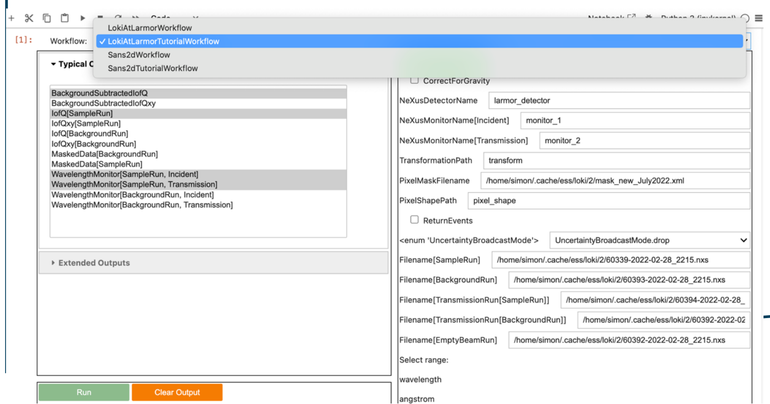
- Existing workflows
- Set parameters
- Pre-configured plots

- Run multiple workflows
- More advanced plots, result comparisons, …
- …

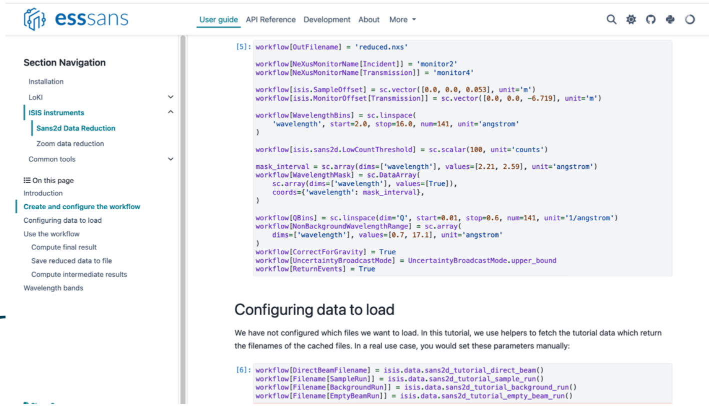
- Customize workflows
- Add workflow components
- Interact with Scipp data structures

Accessibility

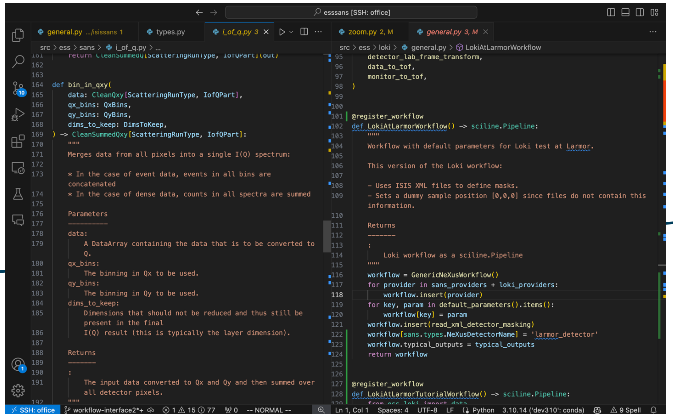
Graphical interface



Pre-written Jupyter Notebooks

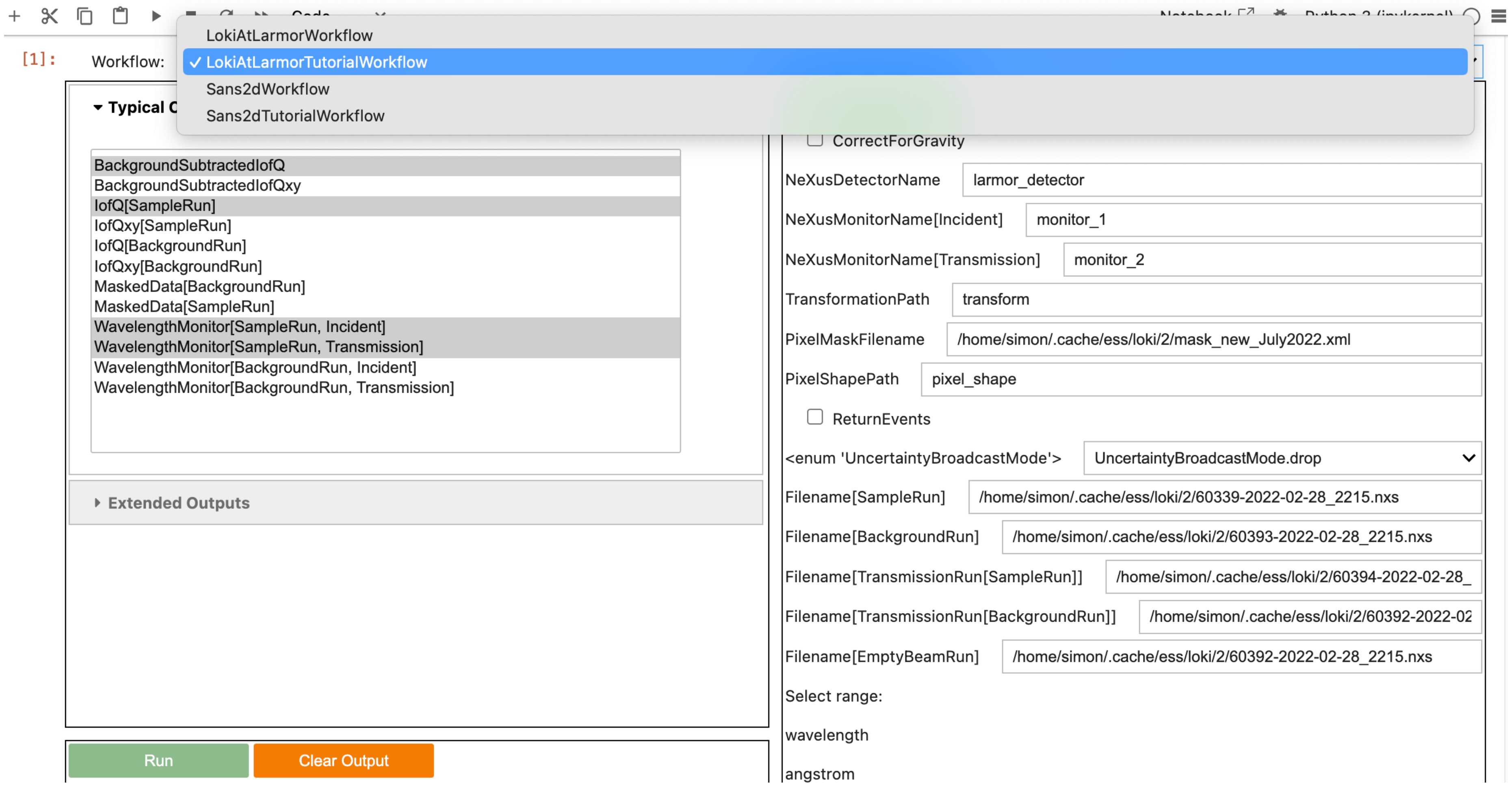


Customized notebooks
Custom Python code



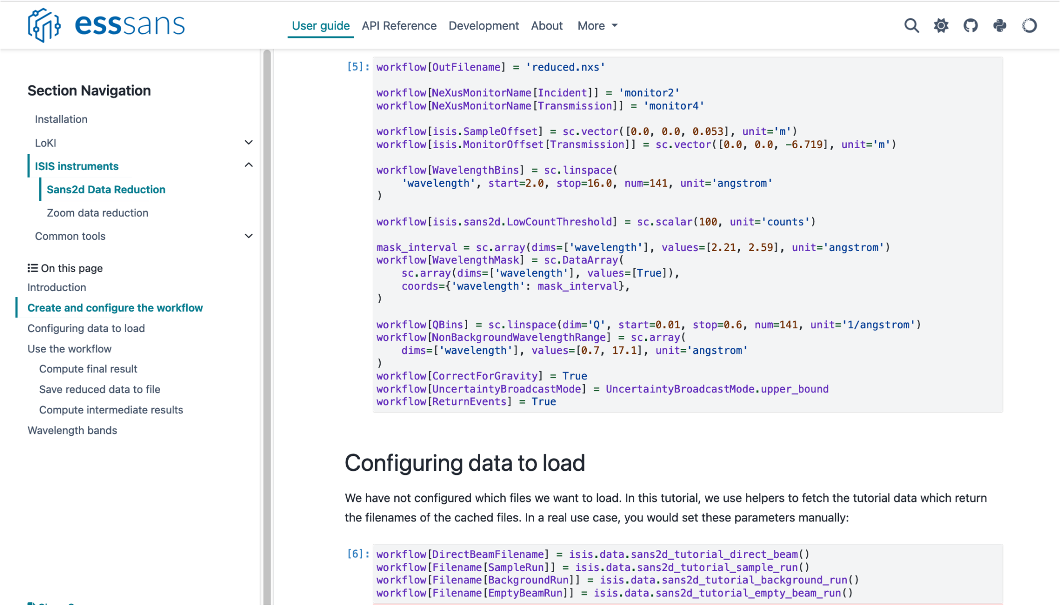
Graphical user interface (early state)

Configure & run pre-defined workflows



There are no plans to turn this into a full-fledged, feature-complete GUI application!

ESSsans notebook interface

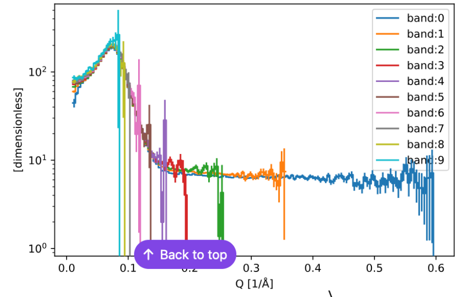



```
[5]: workflow[OutFilename] = 'reduced.nxs'

workflow[NeXusMonitorName[Incident]] = 'monitor2'
workflow[NeXusMonitorName[Transmission]] = 'monitor4'

workflow[isis.SampleOffset] = sc.vector([0.0, 0.0, 0.053], unit='m')
workflow[isis.MonitorOffset[Transmission]] = sc.vector([0.0, 0.0, -6.719], unit='m')

workflow[WavelengthBins] = sc.linspace(
    'wavelength', start=2.0, stop=16.0, num=141, unit='angstrom'
)

workflow[isis.sans2d.LowCountThreshold] = sc.scalar(100, unit='counts')

mask_interval = sc.array(dims=['wavelength'], values=[2.21, 2.59], unit='angstrom')
workflow[WavelengthMask] = sc.DataArray(
    sc.array(dims=['wavelength'], values=[True]),
    coords={'wavelength': mask_interval},
)

workflow[QBins] = sc.linspace(dim='Q', start=0.01, stop=0.6, num=141, unit='1/angstrom')
workflow[NonBackgroundWavelengthRange] = sc.array(
    dims=['wavelength'], values=[0.7, 17.1], unit='angstrom'
)
workflow[CorrectForGravity] = True
workflow[UncertaintyBroadcastMode] = UncertaintyBroadcastMode.upper_bound
workflow[ReturnEvents] = True
```

Configuring data to load

We have not configured which files we want to load. In this tutorial, we use helpers to fetch the tutorial data which return the filenames of the cached files. In a real use case, you would set these parameters manually:

```
[6]: workflow[DirectBeamFilename] = isis.data.sans2d_tutorial_direct_beam()
workflow[Filename[SampleRun]] = isis.data.sans2d_tutorial_sample_run()
workflow[Filename[BackgroundRun]] = isis.data.sans2d_tutorial_background_run()
workflow[Filename[EmptyBeamRun]] = isis.data.sans2d_tutorial_empty_beam_run()
```



```
[15]: result = workflow.compute(BackgroundSubtractedIofQ)
result
```

[15]: scipp.DataArray (235.49 MB)

Dimensions: (band: 10, Q: 140)

Coordinates:

L1	()	float64	m	19.334
Q	(Q [bin-edge])	float64	1/Å	0.01, 0.014, ..., 0.596, 0.6
gravity	()	vector3	m/s^2	[-0. -9.80665 -0.]
incident_beam	()	vector3	m	[0. 0. 19.334]
wavelength	(wavelength [bin-edge], band)	float64	Å	2.0, 3.4, ..., 14.6, 16.0

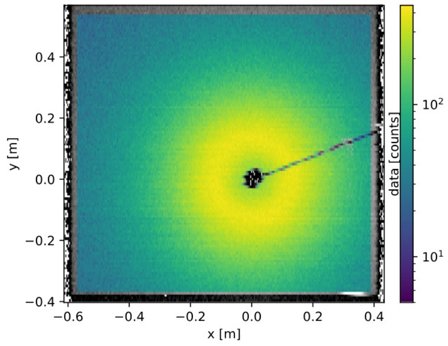
Data:

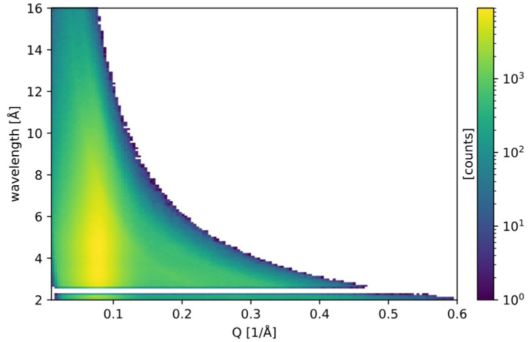
	(band, Q)	DataArrayView	binned data [len=236, len=2086, ..., le

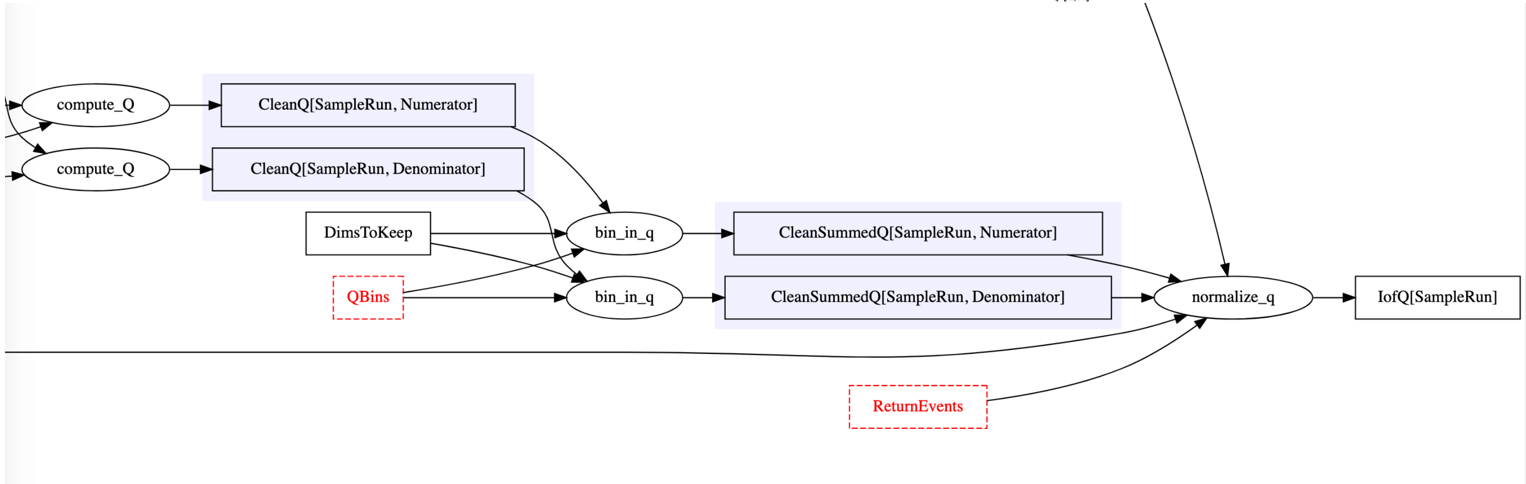
The result is two-dimensional and we over-plot all the bands onto the same axes:

```
[16]: pp.plot(sc.collapse(result.hist(), keep='Q'), norm='log')
```

[16]:

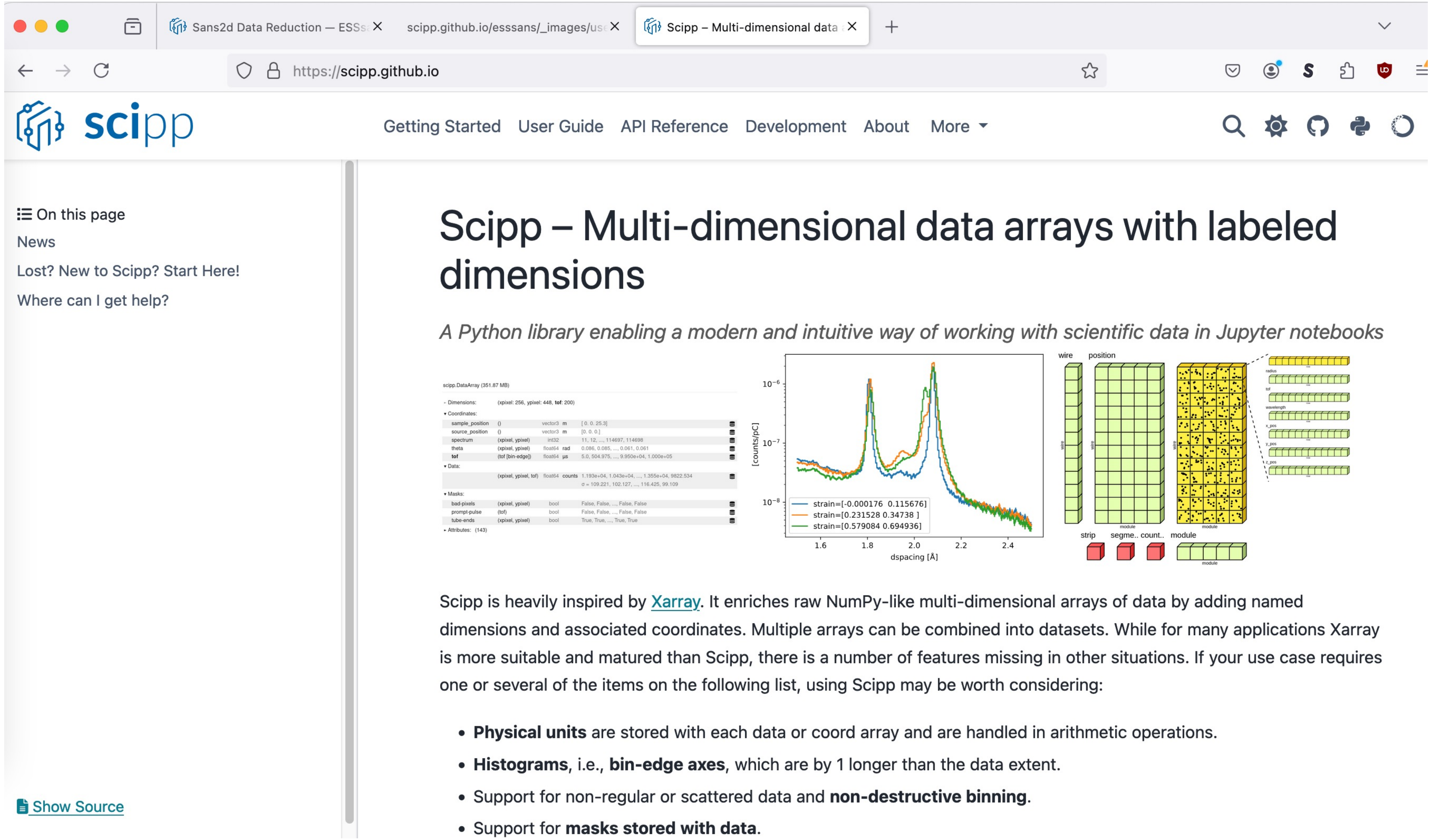






Scipp

Modern and intuitive way of working with scientific data in Jupyter notebooks



Sans2d Data Reduction — ESS
scipp.github.io/esssans/_images/us
Scipp – Multi-dimensional data

https://scipp.github.io

scipp

Getting Started | User Guide | API Reference | Development | About | More

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

Show Source

Scipp – Multi-dimensional data arrays with labeled dimensions

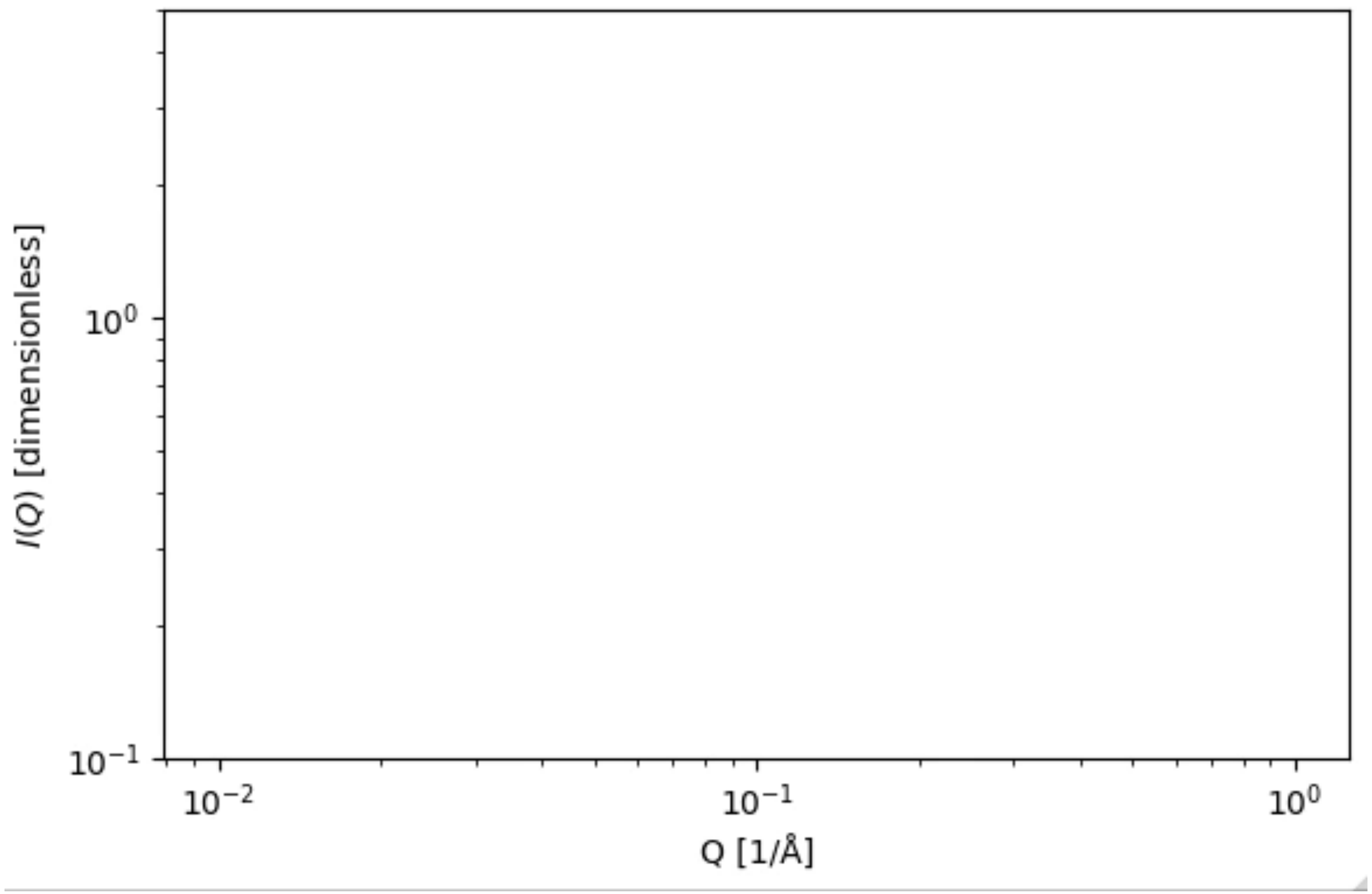
A Python library enabling a modern and intuitive way of working with scientific data in Jupyter notebooks

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**.

Streamed processing

Run full reduction workflow with streamed event data

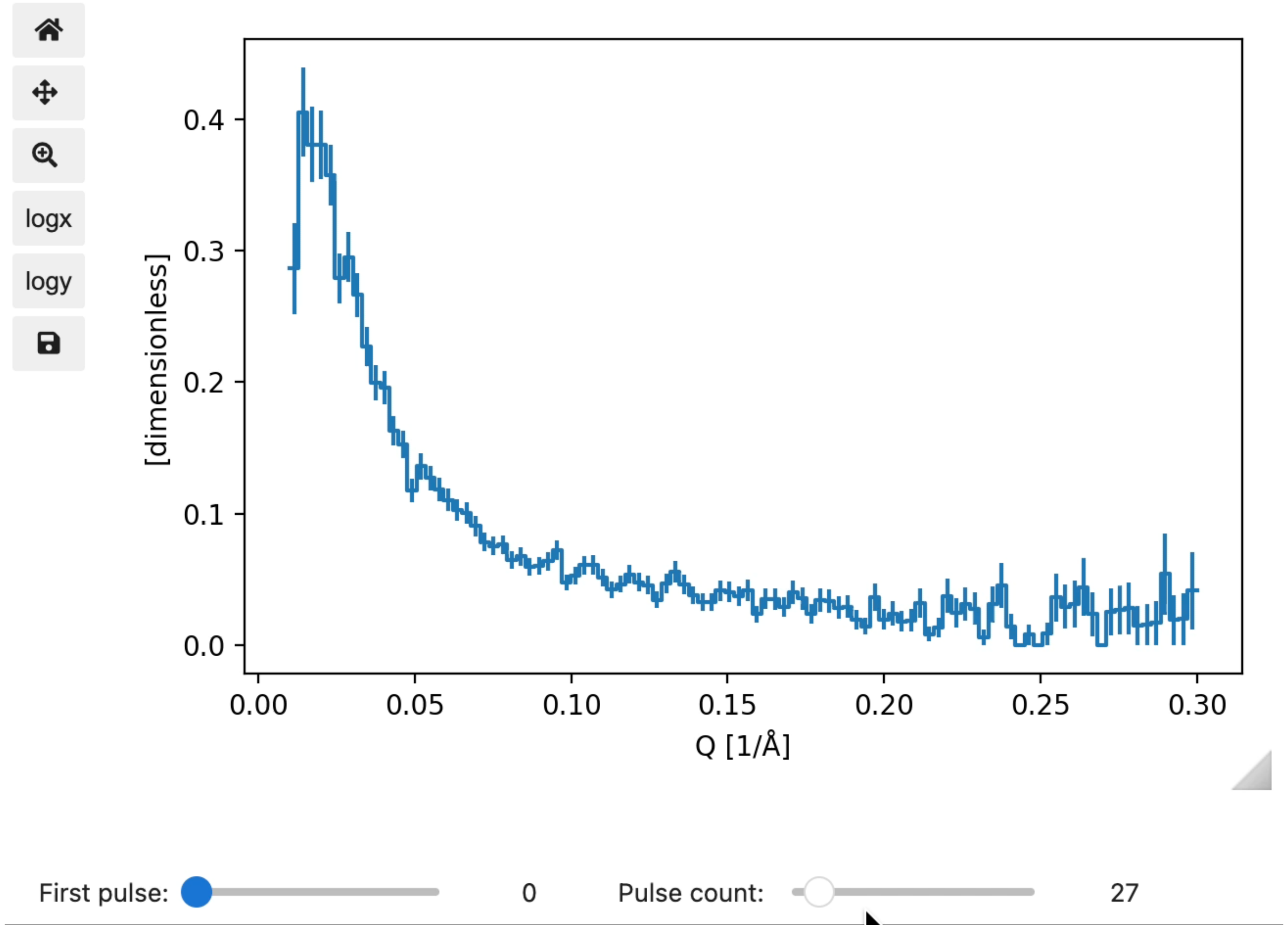


- Detector neutron events
- Monitor neutron events
- Recomputed scattering terms and normalization terms with every new chunk of events
- Different accumulation methods: rolling window, since beginning

Note: This demo shows the streaming operation mode of the SANS reduction workflow and does not connect to a Kafka stream.

Explore files interactively?

Interactive selection of pulse (time) range with quick result updates



Note: For now this is just an experiment to see what our workflows can do. This is not part of the interface we are currently planning to support!

Thank you for your attention! Questions?

Enable hot-commissioning and early science through:

- Modern and flexible Python toolbox
- Spectrum of accessibility and control instead of one-size-fits-all solution