



**EUROPEAN  
SPALLATION  
SOURCE**



# SANS Data Reduction at the ESS

ESSsans and the Scipp software stack

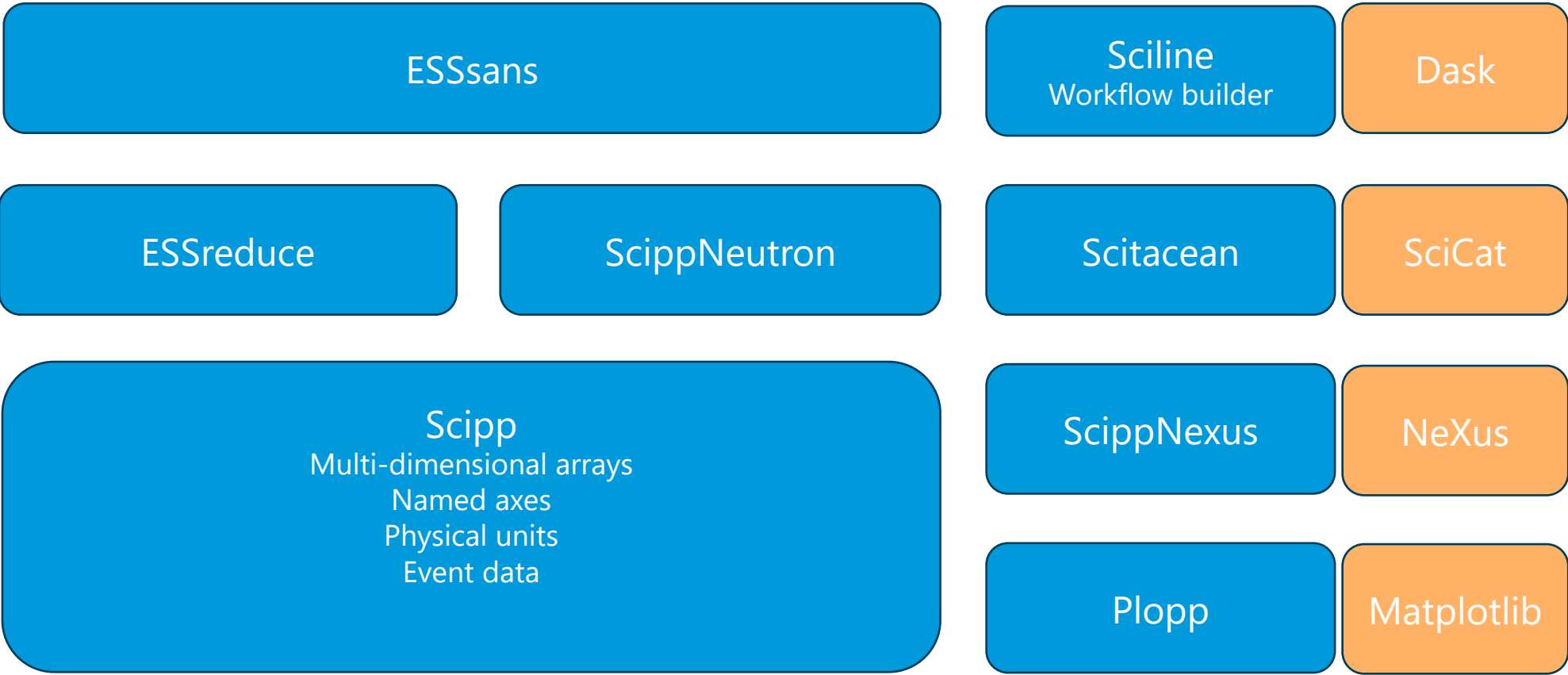
PRESENTED BY SIMON HEYBROCK (ESS/DMSC)

2024-09-06



# ESS software stack for SANS

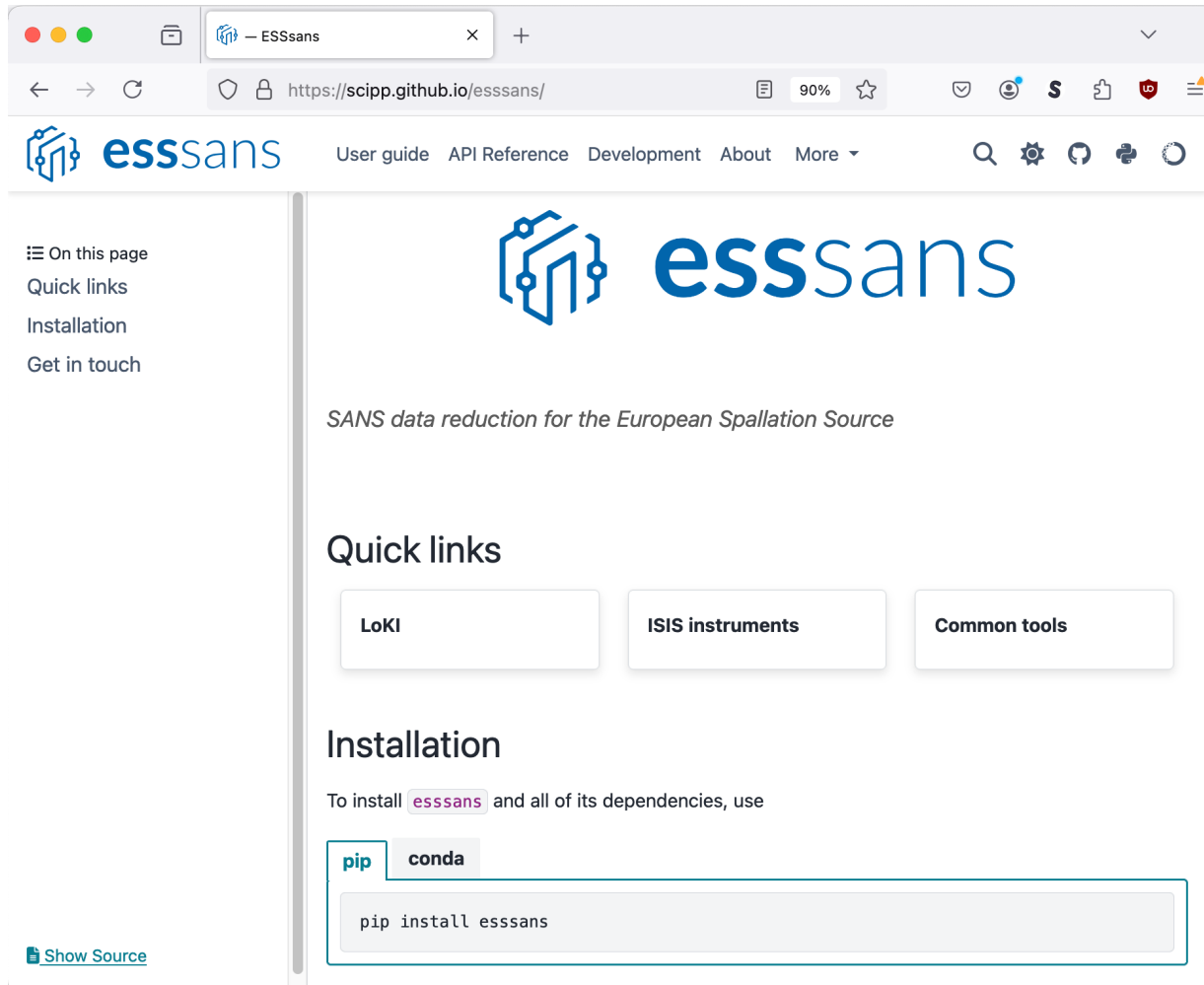
Ecosystem around Scipp / Interfaces to the outside world



# ESSsans



## The ESS Python package for SANS data reduction

A screenshot of a web browser displaying the ESSsans website. The browser's address bar shows the URL 'https://scipp.github.io/esssans/'. The website header includes the 'esssans' logo and navigation links for 'User guide', 'API Reference', 'Development', 'About', and 'More'. A sidebar on the left lists 'On this page', 'Quick links', 'Installation', and 'Get in touch'. The main content area features the 'esssans' logo and the tagline 'SANS data reduction for the European Spallation Source'. Below this, there is a 'Quick links' section with three buttons: 'LoKI', 'ISIS instruments', and 'Common tools'. The 'Installation' section follows, with the text 'To install `esssans` and all of its dependencies, use' and two tabs for 'pip' and 'conda'. A code block shows the command 'pip install esssans'. A 'Show Source' link is located at the bottom left of the page.

Main contributors:

- Neil Vaytet
- Simon Heybrock
- Wojciech Potrzebowski

# Levels of interaction

## Spectrum of accessibility vs flexibility

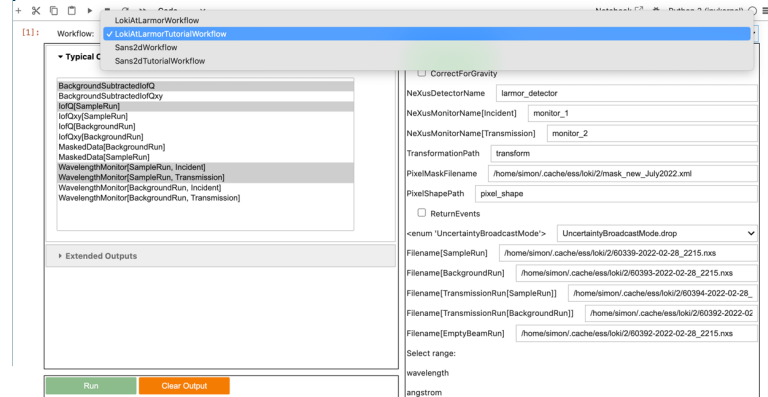
Flexibility

- Existing workflows
- Set parameters
- Pre-configured plots
- Run multiple workflows
- More advanced plots, result comparisons, ...
- ...

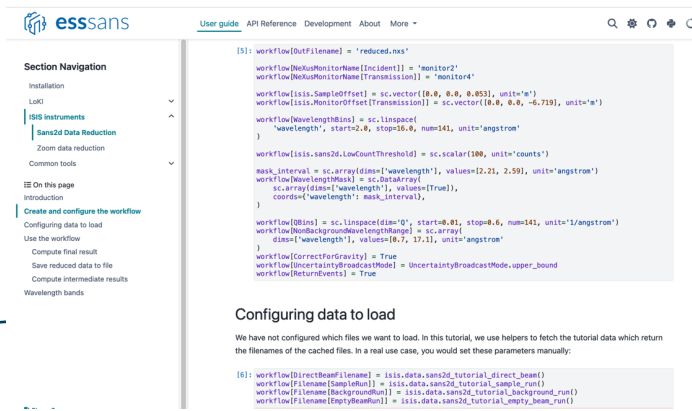
- Customize workflows
- Add workflow components
- Interact with Scipp data structures

Accessibility

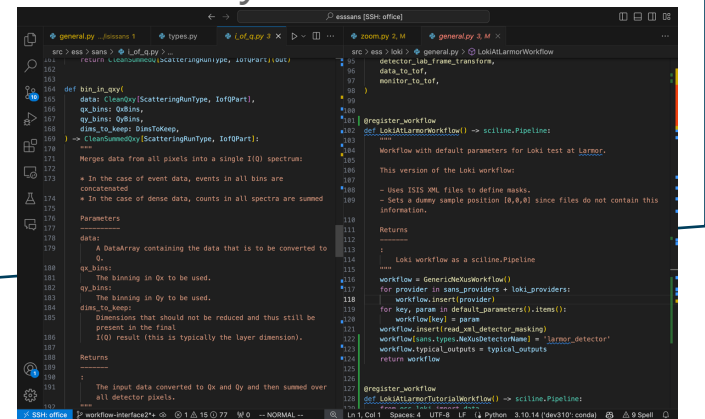
### Graphical interface



### Pre-written Jupyter Notebooks

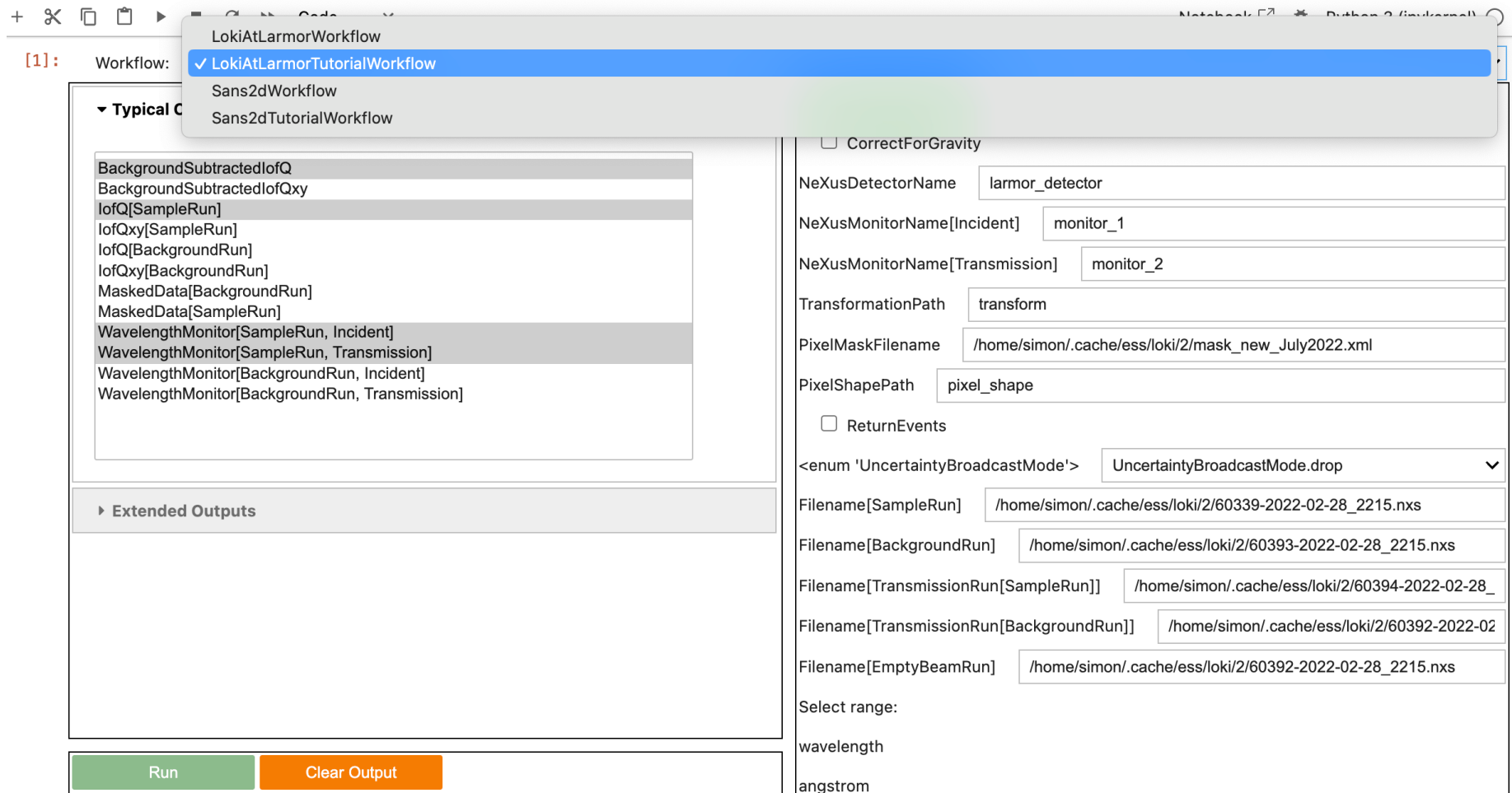


### Customized notebooks Custom Python code



# Graphical user interface (early state)

## Configure & run pre-defined workflows



The screenshot displays a graphical user interface for configuring and running workflows. At the top, a dropdown menu shows the selected workflow: **LokiAtLarmorTutorialWorkflow**. Below this, a list of typical workflows is visible, including **BackgroundSubtractedIofQ**, **IofQ[SampleRun]**, and **WavelengthMonitor[SampleRun, Incident]**. The interface also features a section for **Extended Outputs** and a configuration panel with various input fields and checkboxes.

Configuration parameters shown in the right panel:

- CorrectForGravity
- NeXusDetectorName: larmor\_detector
- NeXusMonitorName[Incident]: monitor\_1
- NeXusMonitorName[Transmission]: monitor\_2
- TransformationPath: transform
- PixelMaskFilename: /home/simon/.cache/ess/loki/2/mask\_new\_July2022.xml
- PixelShapePath: pixel\_shape
- ReturnEvents
- <enum 'UncertaintyBroadcastMode'>: UncertaintyBroadcastMode.drop
- Filename[SampleRun]: /home/simon/.cache/ess/loki/2/60339-2022-02-28\_2215.nxs
- Filename[BackgroundRun]: /home/simon/.cache/ess/loki/2/60393-2022-02-28\_2215.nxs
- Filename[TransmissionRun[SampleRun]]: /home/simon/.cache/ess/loki/2/60394-2022-02-28\_
- Filename[TransmissionRun[BackgroundRun]]: /home/simon/.cache/ess/loki/2/60392-2022-02-
- Filename[EmptyBeamRun]: /home/simon/.cache/ess/loki/2/60392-2022-02-28\_2215.nxs
- Select range:  
wavelength  
angstrom

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

# ESSsans notebook interface



User guide API Reference Development About More



## Section Navigation

- Installation
- LoKI
- ISIS instruments
  - Sans2d Data Reduction
  - Zoom data reduction
- Common tools
- On this page
  - Introduction
  - Create and configure the workflow
    - Configuring data to load
    - Use the workflow
      - Compute final result
      - Save reduced data to file
      - Compute intermediate results
    - Wavelength bands

```
[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)

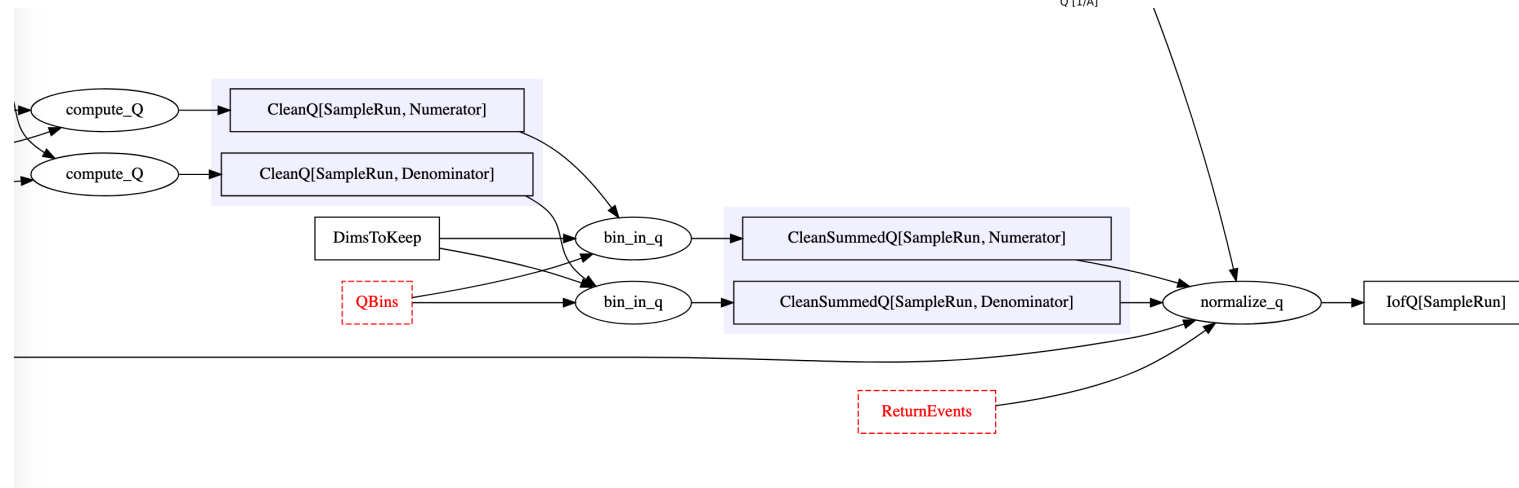
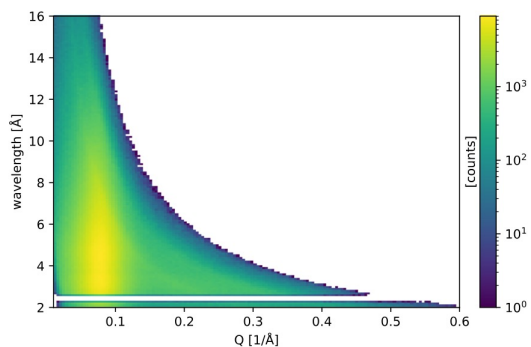
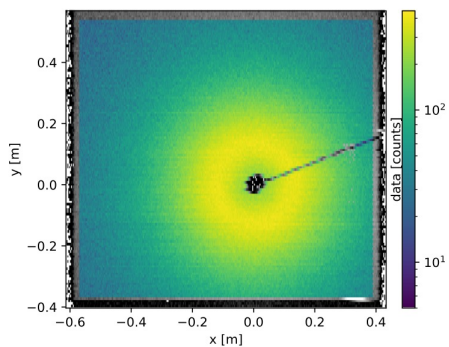
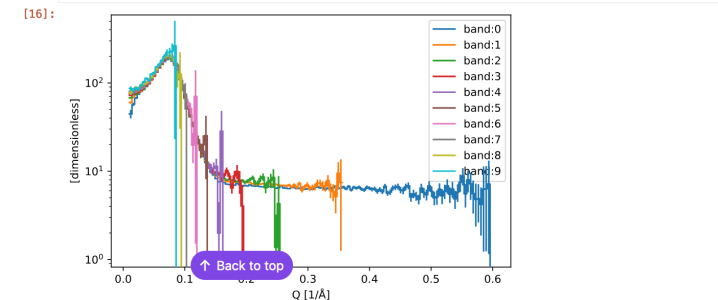
Dimensions: (band: 10, Q: 140)

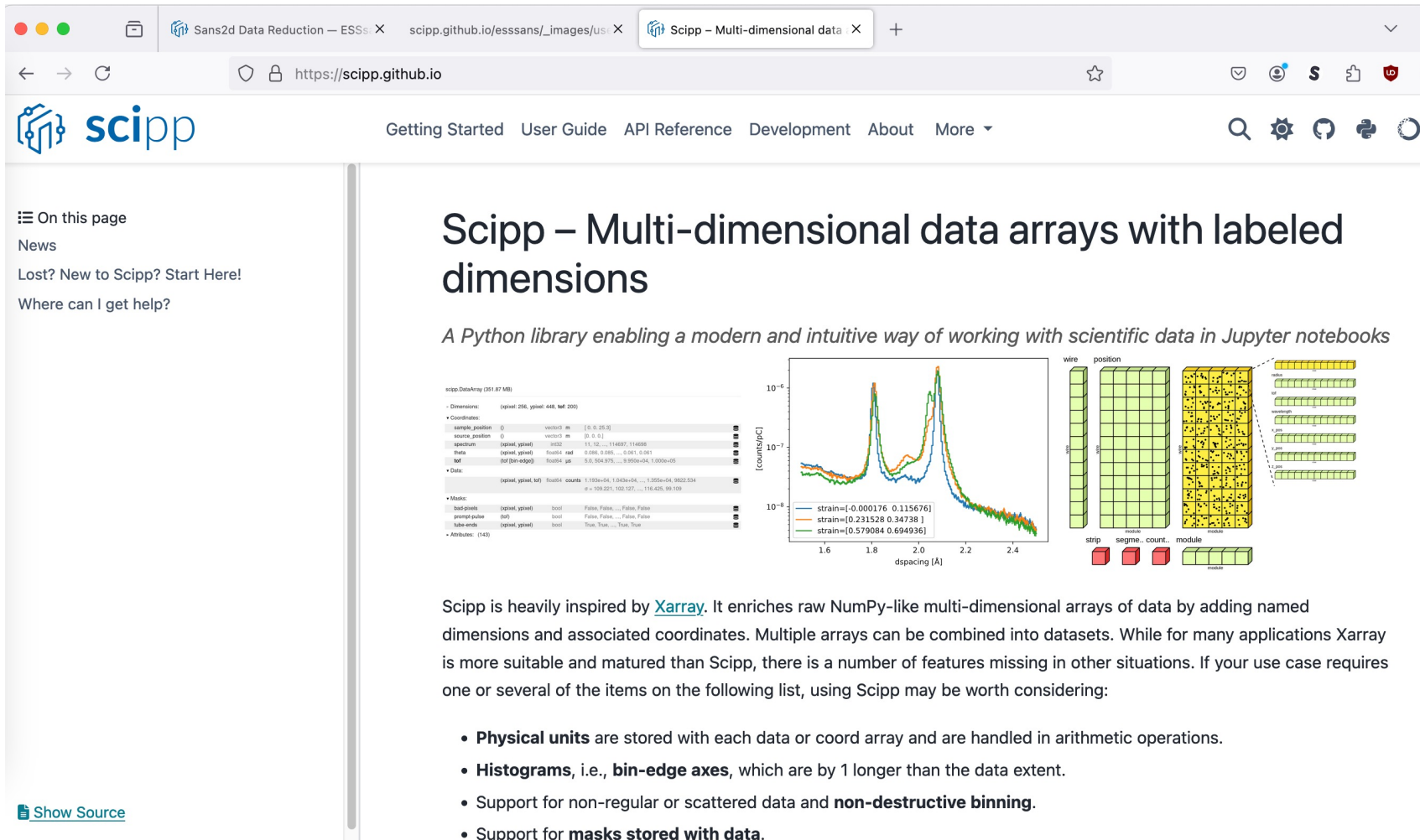
Coordinates:
L1          ()          float64 m      19.334
Q           (Q [bin-edge]) float64 1/Å     0.01, 0.014, ..., 0.596, 0.6
gravity     ()          vector3 m/s^2  [-0. -9.80665 -0.]
incident_beam ()        vector3 m      [ 0. 0. 19.334]
wavelength (wavelength [bin-edge], band) float64 Å     2.0,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

Data:
          (band, Q)          DataArrayView  binned data [len=236, len=2086, ..., 1e
```

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')
```





scipp.DataArray (511.87 MB)

• Dimensions: (dtype: object)

- sample\_position: (dtype: object) vector3 m [0.0, 0.25, 0]
- source\_position: (dtype: object) vector3 m [0.0, 0.0]
- spectrum: (dtype: object) H2O 17.12 ... 1.14897 1.14898
- time: (dtype: object) scalar real 0.000, 0.000 ... 0.001, 0.001
- tof: (dtype: object) scalar pos 5.0, 504.975 ... 9.950e+04, 1.000e+05

• Data: (dtype: object) counts 1.102e+04, 1.042e+04 ... 1.355e+04, 9822.534  
n = 100,221, 100,127 ... 116,435, 98,109

• Masks: (dtype: object)

- strip-ends: (dtype: object) bool False, False, ... False, False
- prompt-pulse: (dtype: object) bool False, False, ... False, False
- tube-ends: (dtype: object) bool True, True, ... True, True

• Attributes: (142)

scipp.DataArray (511.87 MB)

Counts vs dspacing [Å]

— strain=[-0.000176 0.115676]  
— strain=[0.231528 0.34738 1]  
— strain=[0.579084 0.694936]

wire position  
strip segme.. count.. module  
module

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:

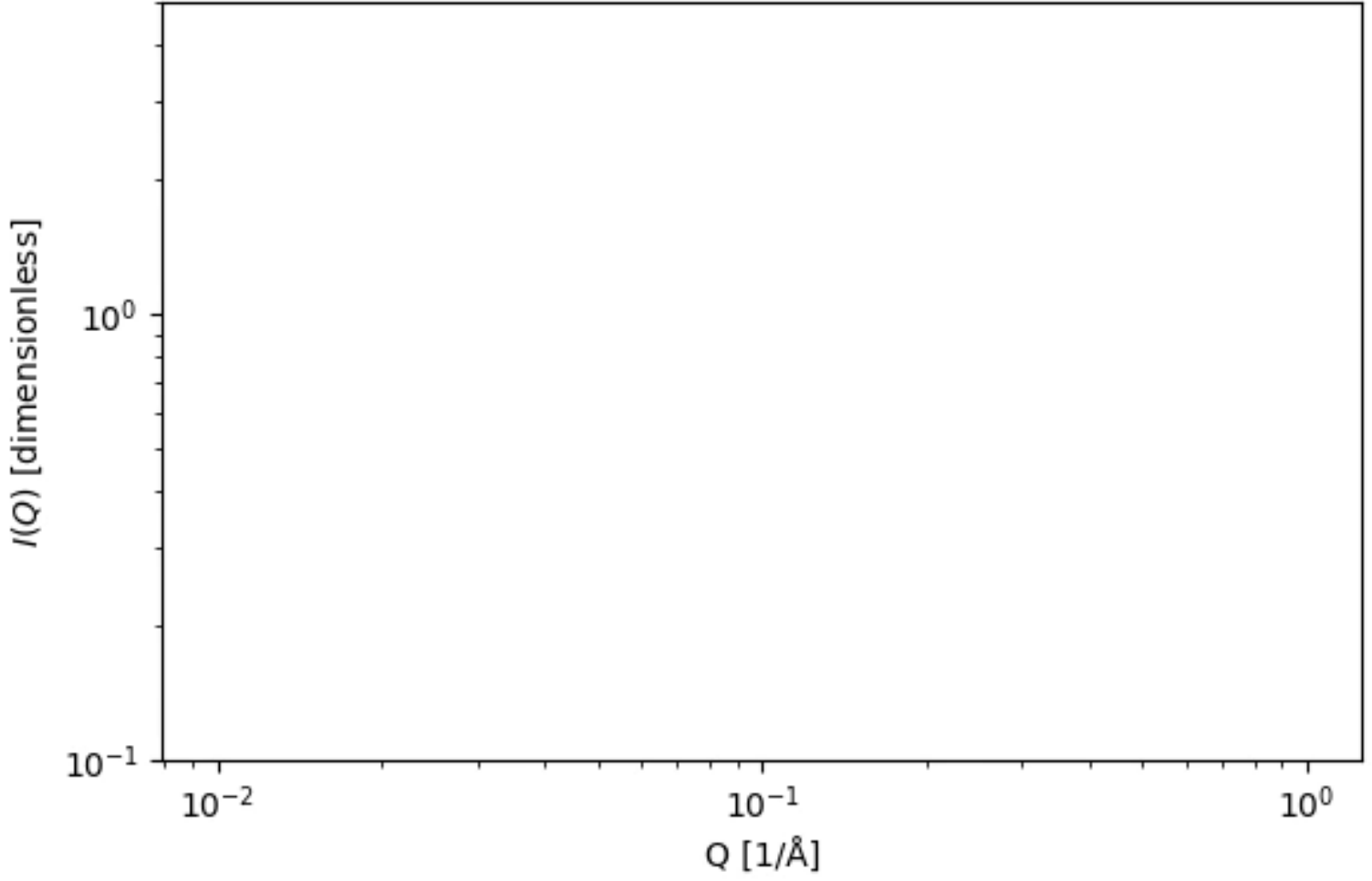
- **Physical units** are stored with each data or coord array and are handled in arithmetic operations.
- **Histograms**, i.e., **bin-edge axes**, which are by 1 longer than the data extent.
- Support for non-regular or scattered data and **non-destructive binning**.
- Support for **masks stored with data**.





# Streamed processing

Run full reduction workflow with streamed event data



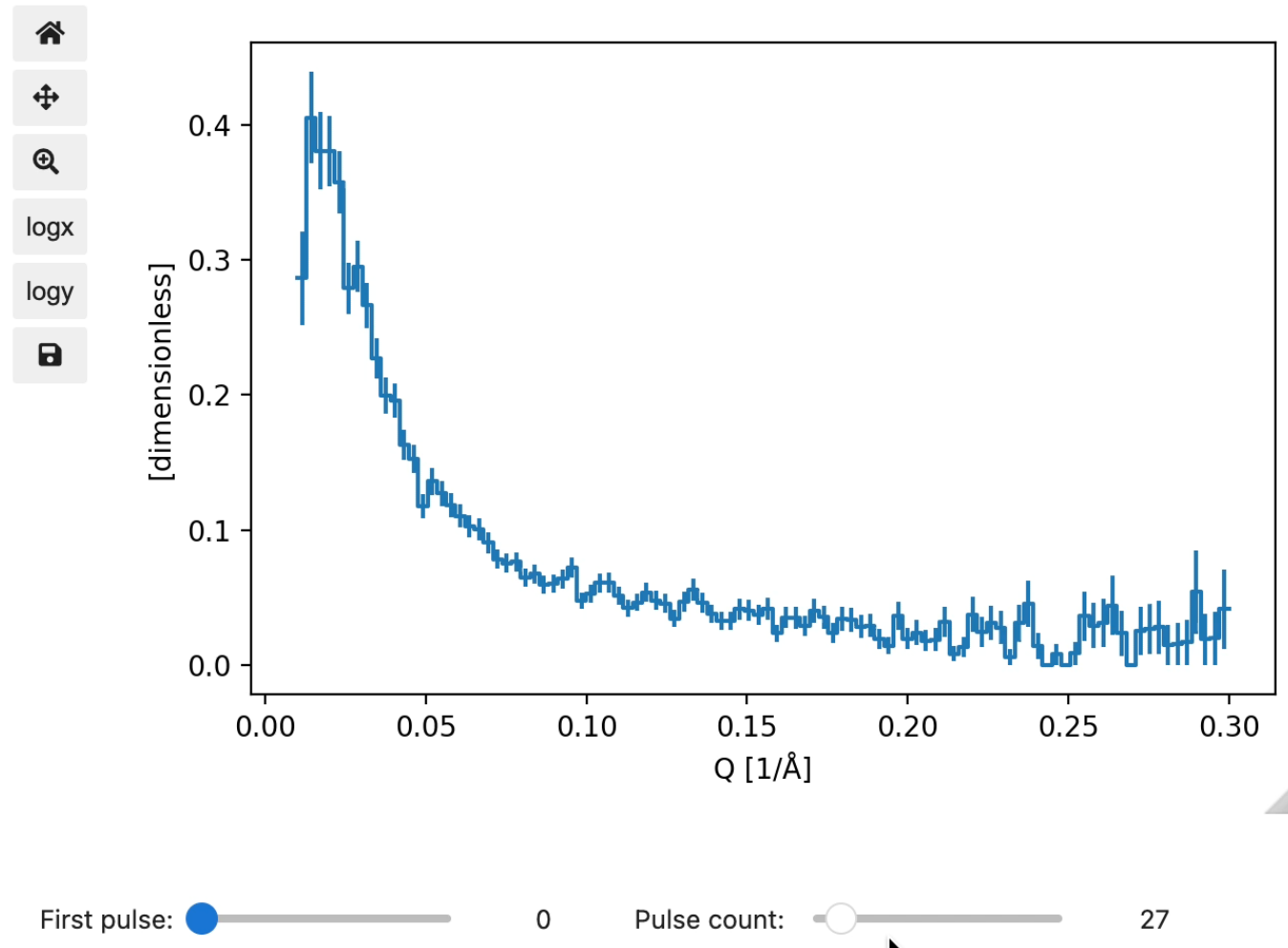
- Detector neutron events
- Monitor neutron events
- Recomputed scattering terms and normalization terms with every new chunk of events
- Different accumulation methods: rolling window, since beginning

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