Scipp / mantid position paper for ESS

Thomas H. Rod, GL Data Reduction, Analysis, and Modelling, ESS
Jonathan Taylor, Head of DMSC, ESS

(October, 2020)

Introduction

ESS is currently under construction with 8 phase 1 instruments to be followed by 7 phase 2 instruments. It is mandatory for these instruments to become successful that they have adequate, functioning, and well-tested data reduction software and workflows in place when these instruments start operation.

History

ESS is a member of the Mantid collaboration that develops the Mantid software for data reduction. The collaboration consists of ILL, ISIS, SNS, and ESS. Recently CSNS also joined this collaboration.

An evaluation found that due to expected higher data rates (10ˆ7-10ˆ9 events /s) and complex detector geometries[1] of ESS instruments the performance of the Mantid framework would be a limiting factor for performing data reduction. To increase the performance of Mantid an initial series of activities were undertaken in 2015 - 2016 adding some significant enhancements to the core Mantid framework.

The rate of these developments was quite slow in part due to accumulated technical debt and size / complexity of the Mantid framework. Changes at the core level for histogram type safety and geometry performance touched many parts of the framework necessitating considerable effort to refactor interfaces at the algorithm level. There were concerns that the resources that could be devoted to data reduction at ESS and the magnitude of required development were incommensurate.

At the 2016 or 2017 developer meeting a discussion was held regarding a plan to refactor the many workspace types in mantid to a single multipurpose workspace. Further discussions and prototypes were made and this effort became the scipp development.

The initial prototypes quickly showed that any change to the workspace types in mantid would require significant refactoring. Whilst this may have been the prefered route, the resources available to the mantid collaboration would not make such a plan feasible. Thus scipp became a separate but interoperable project.

[1] A high proportion of ESS instruments utilise voxel detectors which present a significant challenge for data processing due to the high number of individual channels >500k voxels. These novel detectors cannot be treated as 2D position sensitive arrays. Data processing is therefore a far greater challenge than at present facilities.

scipp

The requirements to the scipp development are based on the 10+ years of domain understanding acquired from Mantid development for neutron data reduction. The requirements are to develop a high-performance, flexible, sustainable and interoperable library for processing neutron scattering data. The library should be able to handle physical units, propagate uncertainties, support histograms as well as event data and handle masks stored with data. Moreover, the library should be written in C++ and with a coherent Python scripting interface with good interoperability with numpy. The choice of C++ in combination with Python / numpy is a commonly used model for scientific software and supports sustainability and interoperability of the framework. Writing the core of the library in C++, rather than Python, makes the library more performant from the outset but also easier to further optimize performance. Moreover, it allows for interfacing to existing C++ applications like Mantid and DIALS at a more 'basic' level than Python.

Status

The scipp library is currently a fully functional and well-documented library (https://github.com/scipp/scipp, https://scipp.github.io). It has been presented at ICANS XIII and has been published as a conference proceeding in Journal of Neutron Research (Heybrock et al, https://arxiv.org/pdf/2010.00257.pdf). scipp is developed following modern software development practises and is currently at version 0.4.0. The ISIS powder diffractometer WISH has been used to drive the development by developing a full data reduction workflow in scipp, but a workflow for LoKI has also been developed and scipp is being used by the LoKI team for analysing their detector data. Time of flight imaging data taken from the ESS Test Beamline has also been treated with scipp. This includes wavelength frame multiplication processing.

The Python interface is interoperable with numpy and does for instance allow for slicing of data structures in a pythonic way. The Python interface has also made it possible to leverage matplotlib, which is also used by Mantid, and Jupyter with its rich environment for combining documentation, graphical elements, and Python scripting in an executable and editable notebook. An example of this is provided with the documentation at https://scipp.github.io, which is developed in Jupyter. An extract from this documentation is shown in figure 1 below.

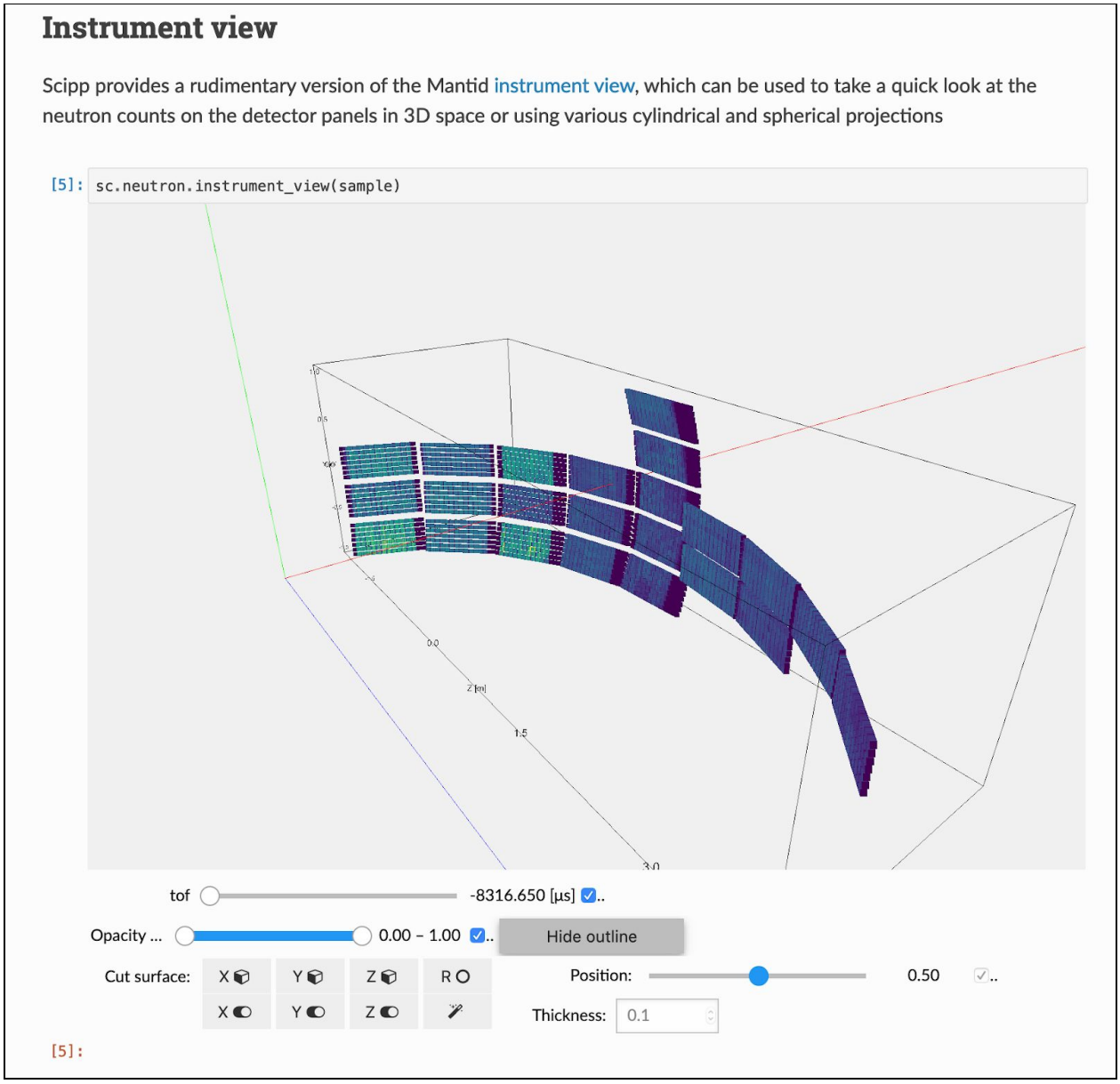

Figure 1. *Example of rich text, executable Python code, and plotting functionality obtainable with running scipp in a Jupyter notebook. The example is taken from the scipp documentation at https://scipp.github.io.*

The performance of scipp was benchmarked against Mantid for the major steps in a data reduction workflow for powder diffraction. The result is shown in Figure 2. It is noticeable that the non-threaded version of scipp is comparable or faster than (threaded) Mantid for all algorithms except for the focus step and that the threaded scipp version is significantly faster for all the steps. The figure also highlights that scipp relies on the Mantid data loader for loading data, hence there is no comparison for data loading. The benchmark indicates that scipp will be

sufficiently performant for handling ESS data rates without relying on MPI, which will reduce the development burden and increase sustainability significantly.

To ensure interoperability between scipp and Mantid, it is possible to convert between Mantid workspaces and scipp data structures (i.e., Dataset or DataArray). Moreover, scipp is currently relying on Mantid for some functionalities. It is essential for fitting and loading. Aside from using Mantid internally, for OSX we have created unofficial forked releases of MantidWorkbench that bundle scipp and is used by the LoKI team.

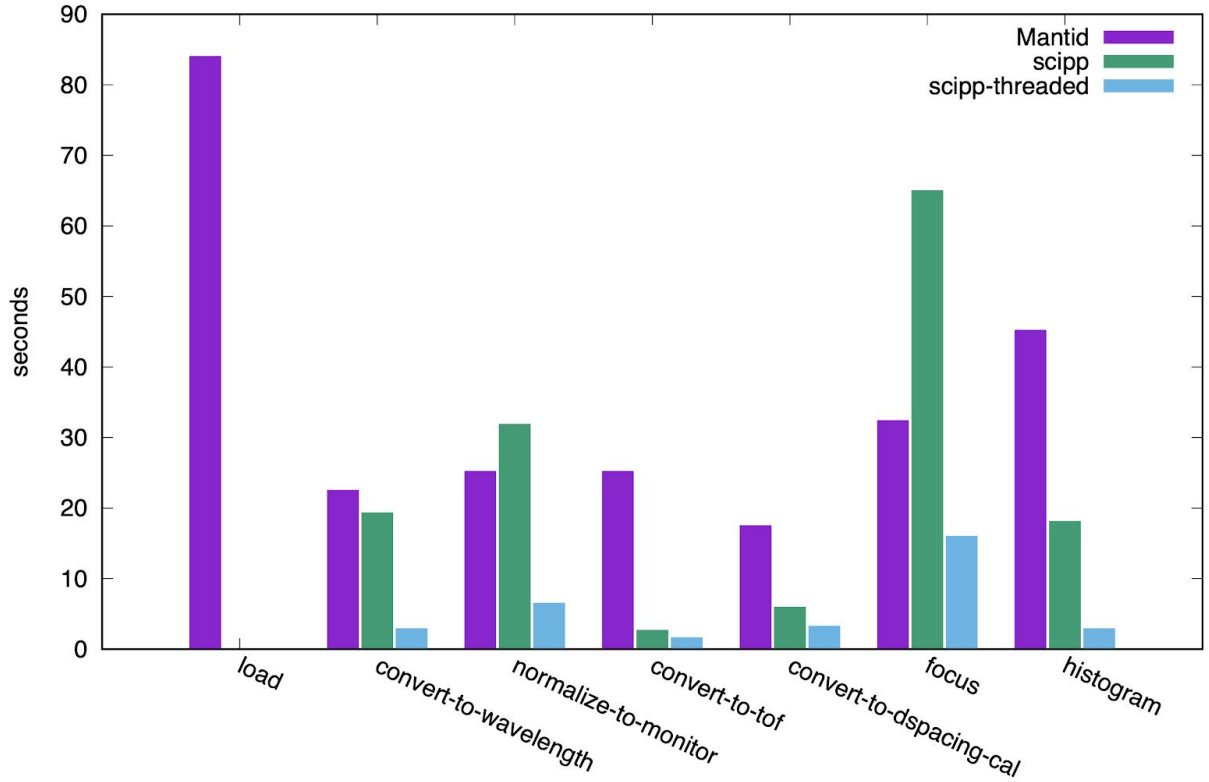

Figure 2. *Timing of major steps in data reduction workflow for powder diffraction with 3×10^9 events.*

Current plans.

Core development

ESS will continue to develop scipp as the core ESS framework for data processing in such a way that it is interoperable with Mantid. For some instruments it may be beneficial to also rely on Mantid or other frameworks. An example in case is NMX, where we may save resources for development and maintenance by hooking into the DIALS collaboration or simplify the user experience for users by relying on a framework they are familiar with from synchrotrons.

Interoperability is facilitated by making it possible to upload a scipp Dataset or DataArray as a workspace in Mantid, convert between Mantid workspace and scipp Dataset or DataArray, and import the scipp library in the Mantid Workbench scripting interface, and vice versa the Mantid API in Jupyter.

ESS will continue to be a member of the Mantid project and support the collaboration to ensure that ESS users have interoperability between facilities. ESS has developers who can contribute to the Mantid development if needed. It is currently being discussed to invest joint effort into more general libraries that could be used by both scipp and Mantid, such as Nexus Loaders or other low-level components.

ESS is in favor of using Jupyter as the user interface for scipp due to the rich environment it provides, particularly for Python applications, and because it is also supported by the projects PaNOSC (https://panosc.eu) and ExPaNDS (https://expands.eu) implementing the European Open Science Cloud (EOSC) for the European Photon and Neutron Sources. The reasons for that are that Jupyter provides a path for providing FAIR (Findable, Accessible, Inter-operable, Reusable) and reproducible data processing, it enables remote data processing through a web interface, and it supports Python which currently is the de facto scripting language for scientific applications. Like scipp, data analysis packages supported by ESS (and NICOS) can be run through a Python scripting interface (e.g., in Jupyter), and it is therefore fairly easy for instrument (data) scientists and users to develop bespoke documented work flows for the full data processing pipeline, which can be stored, reused, and made citable. By scripting we are able to react quicker to requests from the instrument teams during hot commissioning and beyond.

Our plan is to keep scipp focused on data reduction, and hence we will not implement functionality specific for data analysis. However, due to the interoperability of scipp, it can easily be interfaced to existing fitting applications in C++ (e.g. GNU Scientific Library) or in Python (e.g. lmfit) or used in data analysis packages if somebody wishes to do so for their specific need.

Since Python plays a pivotal role for data processing at ESS, we have requested that all instrument teams are able to use Python and we are hence also providing Python training. However, we also realize that not all users will be comfortable with using a scripting language and that the usage of an executable notebook may increase the risk of making errors. We are therefore in parallel with developing scipp investigating avenues for providing a more GUI like user interface and the current plan is to have a strategy ready for implementing a solution for non-savvy Python users by the end of 2021.

It is likely that for ESS standalone GUI interfaces development will follow a similar trajectory to that seen at other similar facilities.

- Multi purpose workbench type applications with a look and behaviour similar to matlab, mantid workbench, similarly to what we have with Jupyter for scipp.

- Generic interfaces that allow specific plugin modules for technique specific solutions such as DAWN / EasyScience.
- Specific interfaces for techniques that couple both data reduction / correction and analysis, such as MantidPlot, Dave, Grasp.

Each class has its place within the domain and can be deployed over remote desktop analysis interfaces ensuring functionality and stability.
Irrespective of specific future developments in operation ESS will host users familiar with software found at existing facilities. For european neutron scattering facilities this is Mantid and many other applications. The familiarity and extensive features of the mantid UI provides a compelling reason to provision Mantid at ESS for some use cases and specifically for pre-processed data. Considering both the volume of ESS data (either file size, or number of files) the only sensible approach for the community is to provision this type of application over remote desktop style data analysis services. This type of centralisation is resource intensive for DMSC and best practice provisioning and packaging strategies are essential.

Instrument specific development

ESS will focus on developing workflows for specific instruments according to their prioritization and we therefore currently focus on the needs of LoKI, DREAM, and ODIN. Whilst we test the framework by developing workflows for existing instruments (e.g. WISH at ISIS) and virtual ESS instruments through McStas simulations and also plan to document the expected workflows, we do not plan to have a fully functioning workflow in place at start of hot commissioning. We will rather finetune the workflow during hot commissioning, because, by experience, it is impossible to know the exact required workflow before the instrument is up running (e.g., time dependent target/moderator behaviour, background, etc.) The instrument data scientists will be pivotal in this process.