



**EUROPEAN
SPALLATION
SOURCE**



DMSC STAP

Updates from DRAM

TORBEN NIELSEN



Agenda



1. DRAM
2. Staff
3. Updates since last STAP
4. Comments to STAP report
5. Summary

DRAM

Data Reduction, Analysis and Modelling



Data reduction

- **scipp** will be used for all instruments
- Possibly in combination with other software for NMX & Imaging
- Are looking for partners

Data analysis

- **easyScience** for powder, sxtal & reflectometry
 - possibly also QENS and TOF imaging
 - But always in combination with other libraries (backengines)
- **SasView** for SANS
- **PACE** for spectroscopy
- **MuhRec** for Tomography.

Data modelling

- **McStas** for instrument simulations
- Now also with Python API **McStasScript**
- and optimized for GPU
- **NCrystal**

DRAM

Data Reduction, Analysis and Modelling - Staff



Simon Heybrock



Neil Vaytet



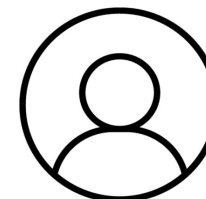
Jan-Lukas Wynen



Sunyoung Yoo



Johannes Kasimir



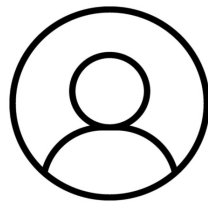
MS



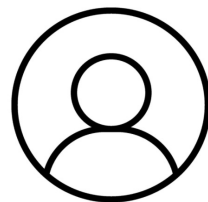
Piotr Rozyczko



Andrew Sazonov



Developer 3



Developer 4



Developer 4.5



Peter Willendrup



Mads Bertelsen



Thomas Kittlemann*

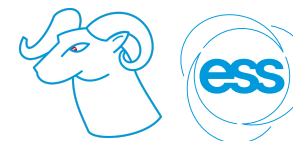
➤ 3 teams (14+ persons)

1. Data Reduction (scipp)
2. Data Analysis (SasView, SpinW, EasyScience, external collaborations)
3. Modelling (McStas++, pan-learning.org, NMX IDS, Detector Group)

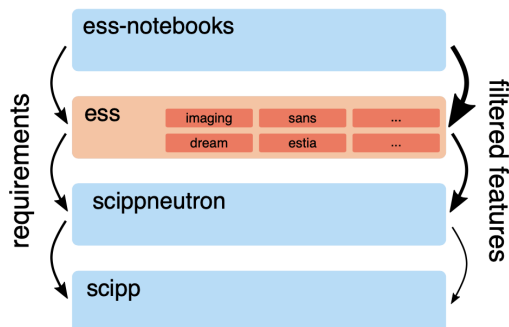
Scope

The DRAM group is responsible for providing the data reduction, analysis and modelling soft-ware for all instruments at ESS.

Scipp – Updates



1. Requirements from IDS
2. **ess**
3. **ess_sans**
4. **ess_reflectometry**
5. **DMSC Summer School**
6. **Sciline**
7. Beamline
8. Scitation
9. Tof
10. New staff
11. Copier



esssans
SANS data reduction for the European Spallation Source

Examples API Reference Developer documentation About Esssans More

Search

On this page
Overview
Table of contents

Overview
This documentation is under construction. See the [Sans2d data reduction](#) example for a quick start.

Table of contents

- Examples
 - [Sans2d data reduction](#)
- API Reference
 - [Classes](#)
 - [Top-level functions and attributes](#)
 - [Submodules](#)

ANACONDA.ORG Search Anaconda.org About Anaconda Help Download Anaconda Sign In

scipp / packages

Packages Files Install Instructions

Filters
Type: conda Access: all Label: main

Package Name	Access	Summary	Updated
esssans	public	SANS data reduction for the European Spallation Source	2023-10-20
scippneutron	public	Neutron scattering tools for Data Reduction	2023-10-19
scipp	public	Multi-dimensional data arrays with labeled dimensions	2023-10-19
plopp	public	A plotting library for Scipp	2023-10-04
ess	public	Neutron scattering tools for the European Spallation Source (ESS)	2023-10-03
sciline	public	Build scientific pipelines for your data	2023-09-15
scippnexus	public	An h5py-like utility for NeXus files based with seamless scipp integration	2023-08-14
tof	public	A simple tool to create time-of-flight chopper cascade diagrams	2023-08-10
scippuncertainty	public	Advanced uncertainty propagation with Scipp	2023-06-18
scippbuildtools	public	Tools for building C++ libraries, python packages and documentation	2023-06-16

Scipp – Updates 1

Requirements from the IDS



- We have previously reported efforts on gathering concrete and actionable requirements from the IDS
- While the effort has led to outcomes for some techniques, others have not been able to provide any feedback.
- We will now try to kick off workflow development in remaining techniques **by a more agile approach**: Our new idea is **that the team will take the lead in implementing a rudimentary workflow** for each technique (unless it exists already).
- **This will serve as a starting point to gather feedback from the IDS**, and to identify and prioritize missing features. Even for this first step, we will, however, require IDS involvement, in particular we need to obtain sample data.

Scipp – Updates 2

IDS Requirements - Template on Confluence


 Data Analysis and Modelling

 ☆
 

- Goal
- Context (Background Knowledge)
- References
- Environment
- Assumptions
- Use-cases
- Preconditions
- Interfaces
- Input/Output
- Input
- Output
- Data Processing Procedure
- Test Cases
- Performance metric (non-functional requirements)
- Acceptance Criteria
- Potential Effects
- Followup Work



SANS direct beam iterations

Created by Wojciech Potrzebowski, last modified by Neil Vaytet on Jun 08, 2023



Goal

We need a notebook (or interactive script) that allows for choosing which tubes, straws or layers will be used in iterations that allow for determining direct beam function. It may be useful to provide various presets to the script e.g. have predefined list of tubes and layers to be used in analysis

Context (Background Knowledge)

D() is the "direct-beam function", which allows us to cross-normalise the incident spectrum to that of the empty beam (without sample) seen on the main detector. The direct-beam function also accounts for the adsorption by any windows in the beamline between the monitor and the detectors, such as Bragg dips at short wavelengths caused by the vacuum window in front of the detector. For more context see: [Direct Beam notes](#) and [SANS Data reduction documents](#)

The version of direct beam iteration in scipp was developed already in 2019: <https://github.com/scipp/less-legacy/blob/master/sans/direct-beam.ipynb>. In the first approximation it may be enough to modernize the notebook so it works with latest version of scipp and ess. There is also a script in Mantid, which has been developed by Richard Heenan and Judith. The script is available from ocncloud: <https://project.ess.dk/owncloud/index.php/?/16903437>. (This is long script but not all there is needed for achieving minimum required functionality). These two scripts should be used as reference for implementation.

Environment

Assumptions

- Flood source simulation data available from Geant4 simulation
- The NeXus input file can be combined from multiple runs, however for the first pass is ok to use a single run

Use-cases

Instrument (Data) Scientists will use Jupyter notebook or script to input which straw, tubes or layers can be used that run the script to investigate intermediate results and then store resulting curves to the file.

Interfaces

Interactive Jupyter notebook

Input/Output

Input

LoKi detector test data, Geant4 simulations (for starting points)

IO and Rg for different standard samples

Output

NeXus file with direct beam function defined per pixel id and the number of comparison plots.

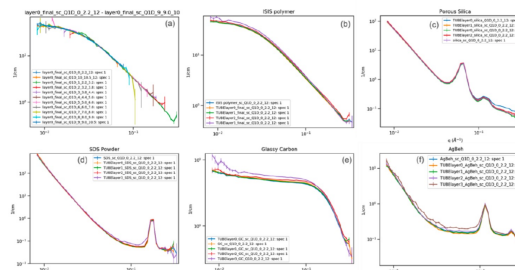


Figure 4 Reduced SANS data from the LoKi detector test. (a) Example wavelength overlap plot of the ISIS polymer standard for one layer of straws (out of 28). (b-f) Reduced data using the DB correction function. The detector data in each has been split into the four tubes layers – to illustrate (i) the current methodology works well for layer 0-2, layer 3 is clearly out in each case, however this is not surprising, given we need to adjust return now the masks have been finally sorted. (b) ISIS standard polymer of partially-deuterated solid polystyrene, (c) Porous silica, (d) deuterated SDS powder, (e) glassy carbon and (f) AgBh.

- Files:
 - Direct beam correction file (written to NeXus)
 - Text files with values from iterations e.g. IO, Rg - it maybe not necessary if it is written in notebooks

Change History

Version	Published	Char
CURRENT (v. 27)	Jun 08, 2023 12:58	
v. 26	Jun 08, 2023 10:49	
v. 25	Jun 08, 2023 09:20	
v. 24	Jun 08, 2023 09:07	
v. 23	Jun 08, 2023 08:58	
v. 22	May 12, 2023 12:48	
v. 21	May 12, 2023 12:42	
v. 20	Apr 18, 2023 09:56	
v. 19	Apr 18, 2023 09:52	
v. 18	Apr 18, 2023 09:43	

Scipp – Updates 3

Breaking apart the `ess` package



- **The `ess` package has long served as a collection of all ESS-specific data reduction code.** It contains submodules for instruments, techniques, and generic helpers. We have, however, struggled for a while with keeping it up to date with developments in Scipp and other libraries, as it proved **challenging to update all techniques “in sync”**, in particular in light of ongoing developments.
- As an example, this approach prevents releasing new features and **bugfixes for instrument A if instrument B is not yet ready for release.** While this has a relatively low impact currently, we believe it would turn into a major problem when entering hot-commissioning.
- A second issue is that dependencies for all techniques are tied together. For example, some techniques **may depend on Mantid which is not available via `pip`.** This slows down continuous-integration and releases as `conda` has to be used. Furthermore, **Mantid requires a fixed Python version**, and would thus prevent techniques that do not even depend on Mantid from using a more recent Python version.

Scipp – Updates 4

esssans



With the release of Sciline, we are now able to move existing workflows to use the new library. As SANS currently is the most developed and most actively used workflow, we have focused on this technique.

The old (pre-Sciline) [workflow](#) is available in the `ess` package, in the `ess.sans` module. We have been intending to break the `ess` package into smaller packages for more than a year (see below), and this has hereby been started: The rewrite of the `ess.sans` module can be found in the new [esssans package](#). While the bulk of the rewrite is complete, it is not done yet, and still has to be rolled out to users.

`esssans` is spearheading the move to Sciline, serving multiple purposes:

- Identify remaining problems and usability issues in Sciline.
- Serve as a full and nontrivial example for how Sciline can be used to write other data-reduction workflows.
- Enable developments that have deliberately been avoided in the old workflow due to growing complexity, e.g., combining data from multiple input files.

Scipp – Updates 5

Sciline



- After many discussions over the past year, as well as a number of small prototypes, we have converged on a solution for writing data-reduction workflows. This has resulted in our [new Python library Sciline](#), which will be used to build such pipelines.
- Sciline builds **tasks graphs** which can be used to compute desired workflow results or intermediate results.
- It is worth noting that Sciline is independent of Scipp, i.e., it can be used with other data such as NumPy arrays of Pandas DataFrames. This might make this solution more attractive to outside users.

Sciline
Build scientific pipelines for your data

Why Sciline?

Writing, testing, and maintaining complex data analysis workflows is hard. Boiler-plate code may hide the actual analysis, making it hard to understand. The code may be hard to test, leading to bugs.

Systems like [Snakemake](#) can help with this, by using a set of *rules* declaring their inputs and outputs. A rule might run, e.g., a Python script. Snakemake then automatically assembles a task graph, and runs the tasks in the correct order to compute the desired outputs. But how do you write the Python script? It in itself can be thought of as a workflow. It may have a significant number of inputs and outputs, and may be complex with many internal computation steps. If intermediate results are large, splitting the rule into multiple rules may be prohibitive due to the overhead of repeatedly writing to and reading from disk.

Motivation

Data analysis workflows are often complex and involve many steps. For example, we may want to:

1. Define or import functions to use for processing.
2. Define parameters for processing.
3. Load the data and apply functions to it.

There are a couple of problems with this:

- In complex workflows, we are forced to write a lot of boilerplate code to load data, apply functions, and save results. This is tedious and error-prone, e.g., since the order of function calls may be wrong or the wrong data and parameters may be passed to a function. This makes it hard to focus on the actual analysis.
- In Jupyter notebooks, the order of cell execution is not clear. This frequently leads to errors that are hard to track down in retrospect and analysis results that are hard to reproduce.

In traditional software development some of these problems are addressed by writing unit tests or integration tests. However, in our experience this is challenging to do properly for data analysis workflows:

- Workflows are often interactive and under active development. Very frequently part of or all of the workflow is written in a Jupyter notebook.
- It is very time consuming to setup good test data for a workflow with good *fidelity*, i.e., test data that will actually allow you to catch errors in your workflow.

Scipp – Updates 6

DMSC Summer School 2023



The team has contributed a major session to the DMSC Summer School. As a highly valuable side effect, the work on the teaching material served as an integration test of almost the entire data pipeline. We will be looking into how we could turn this into a regular integration test, e.g., by running it as part of our continuous integration.

<https://ess-dmsc-dram.github.io/dmsc-school/intro.html>

The purpose of this online book is as a companion to the workshop materials from the ESS Data Management and Software Centre Summer School. This material can be treated in a self-guided fashion.

Welcome

The aim of the summer school is to cover *in-silico* some the ESS data pipeline.

McStas

- Simulation
- SANS exercise
- QENS exercise

Reduction with scipp

- Introduction to Scipp
- Coordinate transformations
- SANS data reduction
- QENS data reduction

Python

- Introduction to JupyterLab
- Python basics
- Using external libraries

Scicat

- Dataset
- Data Curation
- FAIR Data
- Data and Metadata
- Data Catalog
- SciCat
- Python Libraries
- Example
- Data Curation Exercise

Data Analysis – Updates 2

EasyDiffractionApp



Main effort on redesigning the EasyScience based applications to provide performance comparable to most widely used software in the field

Minimization (fitting) performance has been made more robust and significantly improved (3-30 times)

Improved **GUI design** and significantly improved visualization performance of charts and structures

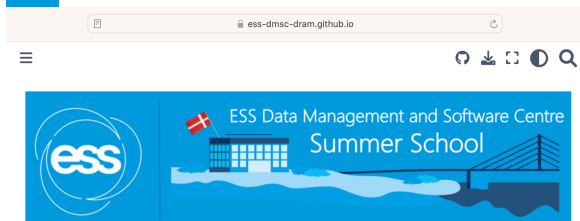
New crystal structure visualiser which will be extended to support magnetic structures
Currently on [8th alpha release](#) (Oct 2023)



Data Analysis – Updates 3

DMSC Summer School 2023

The team has contributed a major session to the DMSC Summer School. Showing how easyCore can be used for fitting.



The purpose of this online book is as a companion to the workshop materials from the ESS Data Management and Software Centre Summer School. This material can be treated in a self-guided fashion.

Welcome

The aim of the summer school is to cover *in-silico* some of the ESS data pipeline.



The EasyScience framework

EasyScience is a framework of software tools that can be used to build experimental data analysis packages. For example, it has been used in the development of [EasyDiffraction](#) and [EasyReflectometry](#). The framework consists of both front- and back-end elements, known as easyApp and easyCore, respectively. The front-end provides a shared library of graphical interface elements that can be used to build a graphical user interface. The back-end offers a toolset to perform model-dependent analysis, including the ability to plug-in existing calculation engines.

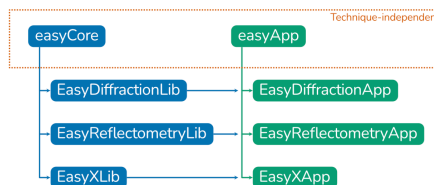


Fig. 5 A normal distribution (blue line), centred on 10.4 with a standard deviation of 1.6 with the maximum likelihood value (red circle). #

The focus in this school is on the Python library, [easyCore](#), which can be used to perform complex model-dependent analysis. The use of [easyCore](#) to perform Bayesian sampling, using external libraries, will also be introduced.



Fitting data with [easyCore](#)

The [easyCore](#) library is designed to enable model-dependent analysis, using a pure Python interface, and give access to a range of optimization algorithms. It is possible to analyse any data for which there is a closed-form mathematical description (i.e., a mathematical model) with parameters to be refined.

This short demonstration will show how [easyCore](#) can be used to analyse the toy problem of data following a quadratic relationship. We manufacture some quadratic data to work with below.

```
from easyCore import np
np.random.seed(123)

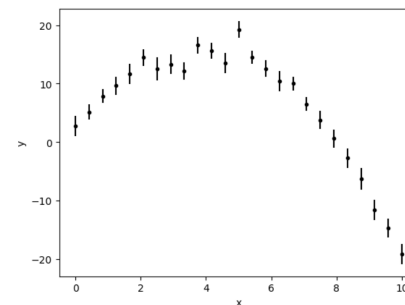
a_true = -0.9594
b_true = 7.294
c_true = 3.102

N = 25
x = np.linspace(0, 10, N)
yerr = 1 + 1 * np.random.rand(N)
y = a_true * x ** 2 + b_true * x + c_true
y += np.abs(y) * 0.1 * np.random.rand(N)
```

The data created above is shown as a standard error bar plot below.

```
import matplotlib.pyplot as plt

plt.errorbar(x, y, yerr, marker='.', ls='', color='k')
plt.xlabel('x')
plt.ylabel('y')
plt.show()
```



Instrument modelling – Update 1

McStas etc. updates



- McStas 3.3, March 31st, 2023 (Presented at ECNS in Garching)
 - Built-in NeXus support on all platforms
 - mcgui run dialogue now allows to directly specify --format=NeXus and --format=NeXus -IDF
- McStas 3.4, September 19th, 2023
 - End of McStas 2.x releases
 - End of support for Perl tools
 - Build process for both macOS and Windows overhauled, embedded conda environment.
 - MPI support is now included with the installation on Windows
- Collaboration with SOLEIL for McXtrace school May 2023, Talk and demo at HighNESS scattering kernel school May 2023, DMSC Summer School September 2023
 - <https://e-learning.pan-training.eu/course/view.php?id=135> school entry point (Federated login)
 - <https://panlearningjhub.esss.dk/dmscsummerschool2023> “Fat” docker with McStas-McStasScript-NeXus-SCIPP developed and deployed

Instrument modelling – Update 2

McStas etc. updates



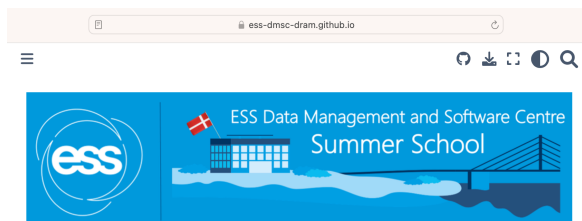
- McStasScript continues, the package can now load McStas generated NeXus data.
- McStas now ships with McStasScript and a JupyterLab
- The McStas package now also includes a translation tool that can write a McStasScript python file from a classic instrument file
- The current development efforts on McStasScript concerns creating a widget interface to explore event data generated by McStas.
- This widget uses scipp and plopp to group the data as individual rays using the neutron id and to build interactive widgets that display the data.
- It will for example be possible to highlight a peak on the detector and see the history of the neutron rays that make up the peak.

Instrument modelling – Update 3

DMSC Summer School 2023

The team has contributed a major session to the DMSC Summer School. Showing how McStasScript can be used for instrument simulations.

<https://ess-dmcs-dram.github.io/dmcs-school/intro.html>



The purpose of this online book is as a companion to the workshop materials from the ESS Data Management and Software Centre Summer School. This material can be treated in a self-guided fashion.

Welcome

The aim of the summer school is to cover *in-silico* some of the ESS data pipeline.



Simulation

At the ESS we have used simulations extensively to design and optimize the facility.

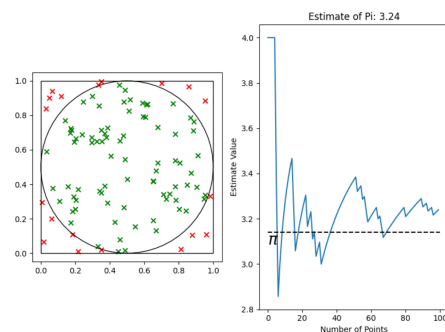
Monte Carlo Ray-tracing

For simulation of neutron scattering the most popular technique is Monte Carlo ray-tracing. Lets look at the two separately and then merge them.

Monte Carlo

The Monte Carlo technique relies on using random numbers to estimate values that would otherwise be difficult to calculate. It is a computationally expensive technique, but is very versatile and often work when other more efficient techniques fail. For example one could estimate the value of π by using random sampling to find the ratio of the areas of a circle and a square as shown below. The ratio of their area's can simply be estimated by the number of random points falling outside and inside the circle, from which an estimate of π can be calculated. The more random points we use, the better an estimate of π .

```
import monte_carlo
monte_carlo.example()
```



```
import mcstas as ms
import make_QENS_instrument
import quizlib

quiz = quizlib.QENS_Quiz()
```

QENS exercise

This notebook contains code and questions for a McStas simulation of a simplified backscattering instrument that can investigate quasi-elastic scattering from samples.

Quasi-elastic scattering is inelastic scattering with small transfers and typically views a broadening of the elastic signal. At ESS the backscattering instrument under construction is called MIRACLES and uses an inverse time of flight technique, here neutrons are scattered off the sample and some hit an analyzer afterwards. This analyzer is angled such that the neutron is scattered almost backwards, and due to Bragg's law this will happen with a given energy.

It turns out the precision of that energy is highest when the neutron is scattered back in the direction it came from, but most instruments choose a slightly lower angle to avoid hitting the sample a second time. The detector is then placed slightly above or below the sample.

Since the analyzer chooses a specific energy, the final energy of the neutrons being recorded in the detector is known, this can be used to propagate the time of the neutron to the sample position. Then the time at that moment and the known pulse time can be used to calculate the time-of-flight, which with the known distance gives the speed and thus energy before scattering in the sample. The difference between the known initial and final energy provides the energy transfer, which for backscattering can be down to μeV , where most other inelastic techniques look at meV .

In this notebook you will get this simplified backscattering instrument and answer a few questions about the results. You will also get to improve it and run experiments with a small range of known and unknown samples.

Get the instrument object

First we need the McStas instrument object. Here it is retrieved from a local python function that generates it.

```
instrument = make_QENS_instrument.make(input_path="run_folder")
```

SANS exercise

In this notebook you will get this simplified SANS instrument and answer a few questions about the results. You will also get to improve it and run experiments both with and without a sample.

SANS is an abbreviation for Small Angle Neutron Scattering, and as the name suggests is concerned with the scattered signal at very small angles. Here we will look at a sample in solution composed of some simple geometry, which will cause an interesting scattering pattern on the detector.

Get the instrument object

First we need the McStas instrument object. Here it is retrieved from a local python function that generates it.

```
instrument = make_SANS_instrument.make(input_path="run_folder")
```

Investigate instrument

First investigate the instrument object (`instrument`) using some of the available methods. All the methods that help do that start with the word `show`. In particular, look at what parameters are available and take a look at the instrument geometry.

```
instrument.show_parameters()
```

- <https://e-learning.pan-training.eu/course/view.php?id=135>
- Docker with McStas-McStasScript-NeXus-SCIPP developed and deployed

Instrument modelling – Update 4

The conclusion of the 3 year EU project HighNESS



- Simulated their performance (three conceptual instruments) on the different proposed moderators put forth in HighNESS
- The concepts were two SANS instruments, one conventional and one with focusing optics, and a simple imaging instrument.
- In the HighNESS project it was found that using a solid deuterium moderator rather than liquid deuterium could provide unprecedented intensity at wavelengths above 40 Å.
- The results were described in a thorough deliverable report and in the main HighNESS deliverable, the conceptual design report (CDR) <https://arxiv.org/pdf/2309.17333.pdf>

McStas timeline at a glance

When did what functionality arrive



v1.1-1.6x, 1999-2002
mcrun, mcplot, mcgui,
Single_crystal, Source_gen,

v1.9-1.11 2005-2007
PowderN & Isotropic_Sqw,
mcstas.org domain
Polarisation, macOS,
Debian pkg, MPI support

v2.0-v2.1 2012-2014
/Initial Mantid support,
comp std., first
Python tools

v2.5-2.7 2018-2020
NCrystal
GPU efforts
Lots of new instruments

v3.2, 2022
Official GPU support
arrives
v2.7.2 is last 2.x release

v1.0, October 15, 1998

v1.7-1.8, 2003-2004
Windows supp.
GPL license,
new tools

v1.12.x "era" 2008-2011
McXtrace project start,
ESS-oriented simulation
work, workshop efforts
take off
v1.12c is last 1.x release

v2.3-2.4.x 2015-2017
ESS_butterfly,
MCPL
Union subsystem, Python
tools fully default

v3.0-3.1, 2020-2021
Official GPU support
arrives

v3.3-3.4, 2023
Embedded NeXus,
mcstas-pygen,
McStasscript embedded

Steady in-flow of "smaller" developments, bugfixes, user contributions...

Addressing STAP comments & suggestions



From STAP report – and reply from DRAM

1) Workflows

- Regarding workflows, many of the IDS are having challenges in working with scientists on the instrument teams to get the calibration and reduction/analysis steps defined.
- A good starting point would be to look at the existing workflows in Mantid as an outline to at least help start a discussion.
- **A good suggestion we will keep in mind. Aligns with our new approach for handling IDS's requirements.**

2) Repository

- Having a separate repository for workflows make sense.
- ESS-Spectroscopy, ESS-Diffraction, ESS-Imaging/Engineering, ESS-SANS, ESS- Reflectometry, ESS-General?
- **We have arrived at the same conclusion. This has already been implemented a few months ago.**

3) GitHub repository tools

- Having the GitHub repository provides the ESS with a unique opportunity that other facilities did not have in the build up to operations.
- The time and process of developing specific workflow tasks can be monitored and then the process refined in response to expedite future developments.
- **A good suggestion we will keep in mind. It was mentioned during the April STAP meeting; and point was noted.**

4) Mcstas – Geant4 coupling

- The current workflow of merging McStas and Geant for Dream is not sustainable beyond instrument design
- **This was never the design goal. The goal is to mimic all of Geant4 features in a one-stop McStas-shop for the untrained McStas user.**

Addressing STAP comments & suggestions

From STAP report – and reply from DRAM



5) Ownership

- Something that needs to be decided is who owns the simulations
- Will they be maintained long term by the instrument teams, the IDSs, centrally by the DRAM group, or by some other means
- **Already on our radar. We have more or less consensus in the team. Need to check with stakeholders.**



Finish presentation

Questions ?

