Guidelines and Standards + Mantid

Anders Markvardsen
STFC

April 2016, SINE2020, PSI

Mantid

Science & Technology Facilities Council
ISIS

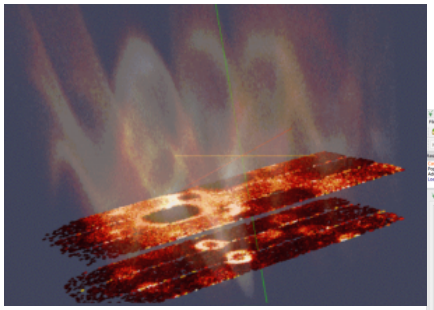
Outline

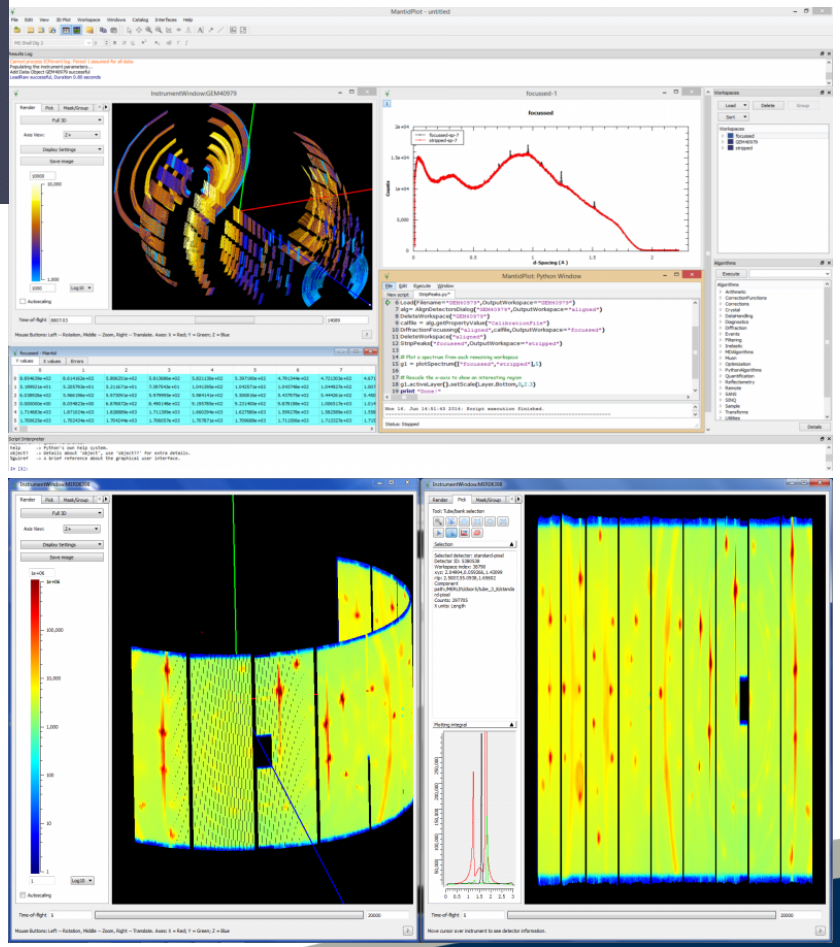
Task 10.2: Guidelines and Standards

More broadly tasks 10.2-4 and Mantid

- Visualisation
- Fitting
- Interoperability

In one slide Mantid





THE MANTID PROJECT

- **Aims**
  - Data reduction/analysis framework for neutron scattering and muon **experiments**
    - Cross-platform
    - Easily extensible
    - Freely redistributable
    - Open source
- **Adoption**
  - **3(4) Partner Facilities**
  - **9(8) Contributing Organisation**
  - **In use on 52 instruments worldwide**

OAK RIDGE National Laboratory
HIGH FLUX ISOTOPE REACTOR
SPALLATION NEUTRON SOURCE

EUROPEAN SPALLATION SOURCE

Tessella
Technology & Consulting

Science & Technology Facilities Council
ISIS

Guidelines and Standards

- Task 10.2 of this WP
- Due date 1st April 2017
- Coordinator: STFC
- Partners: ESS, FZJ, PSI
- Observers: ILL

Previous work to task 10.2

- 2014: NMI3-II Data Analysis Standards (WP6) Task 2: solutions for ***developing a common software infrastructure***

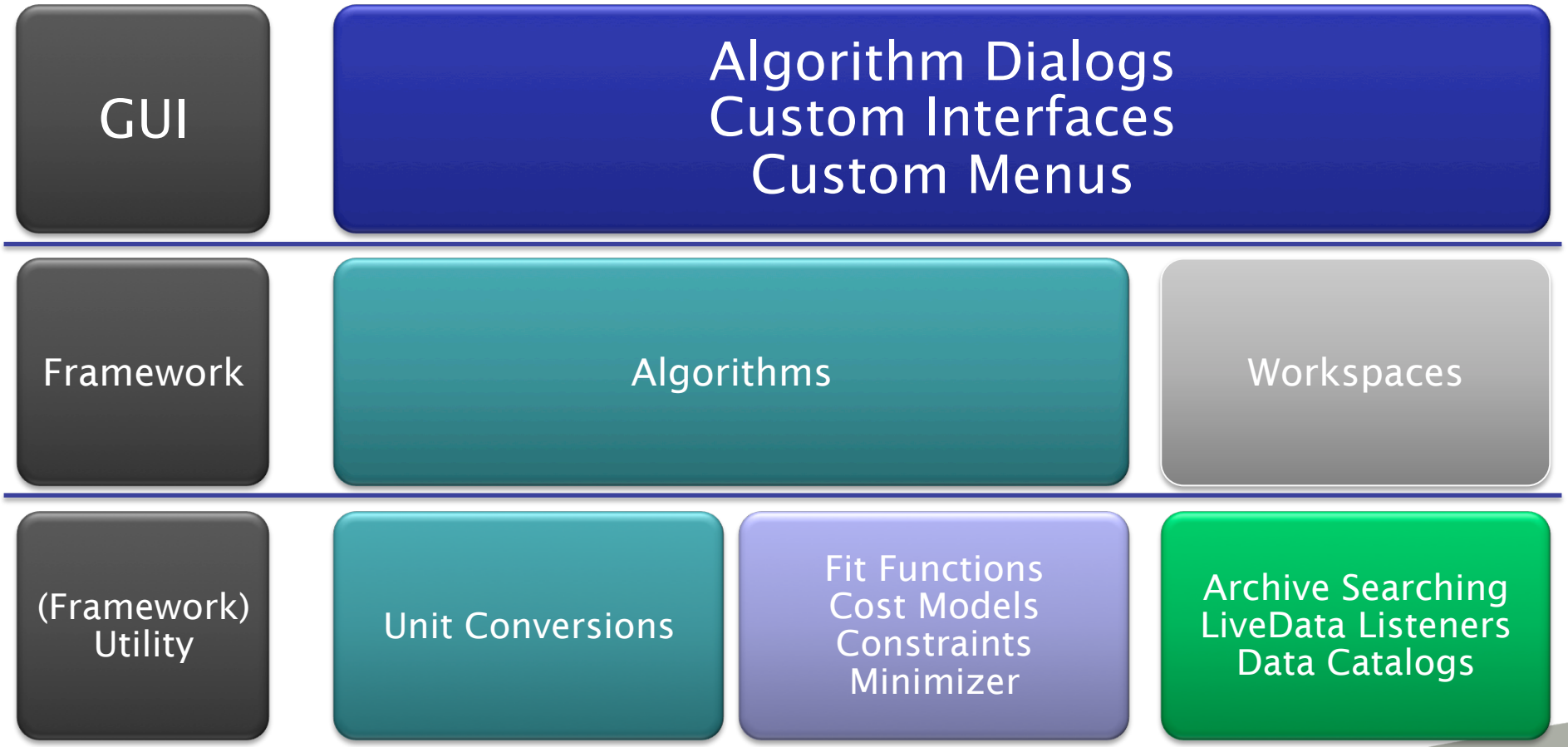
- 2011: PaN-data (D2.2) ***Common policy framework on analysis software***

Sentences from Task 10.2

- Guidelines and standards will be established that enable different software components to be **linked together in an interoperable (plug and play) manner**

- Particular emphasis will be put on **pluggable fitting functions**, including **Bayesian fitting**

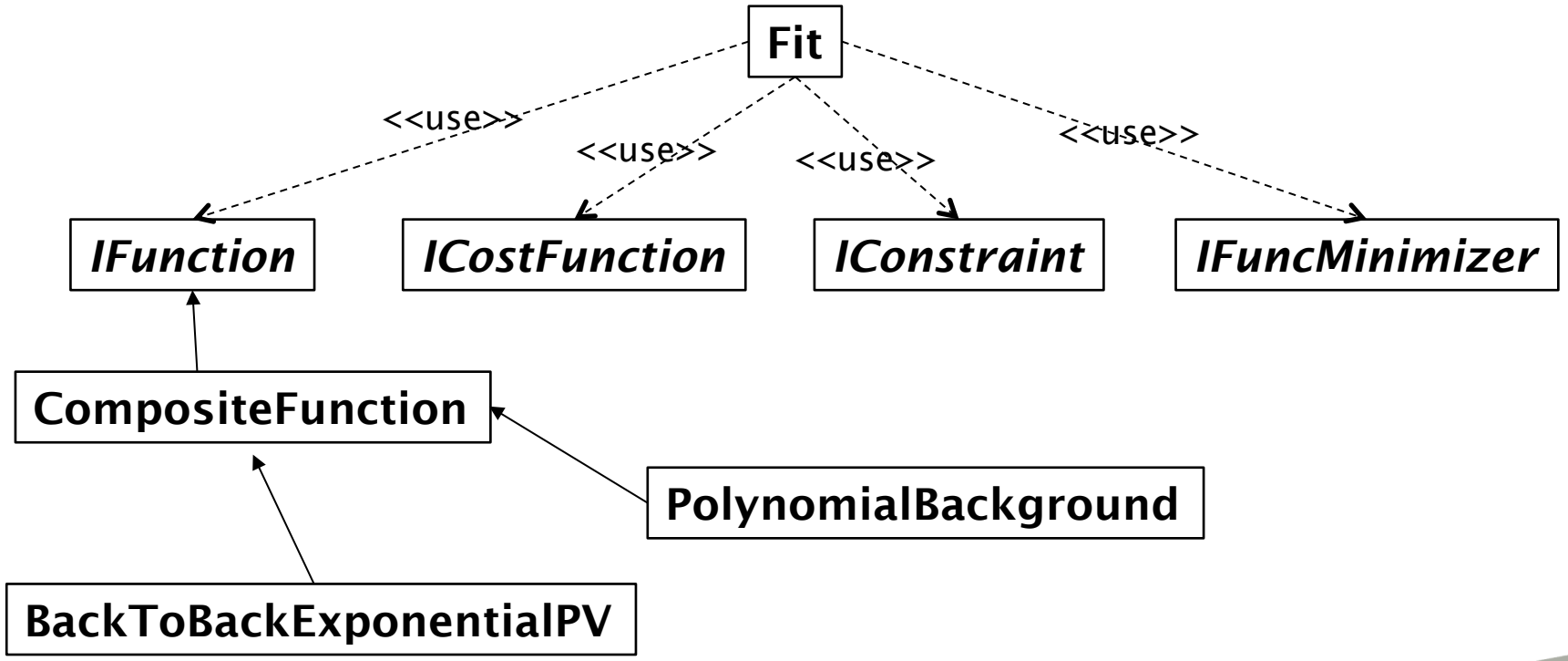
‘Low’ level interoperability



Pluggable fitting functions

- SasView
- BornAgain
- Mantid
- Other?

Pluggable fitting functions - Mantid



Pluggable fitting functions - Mantid

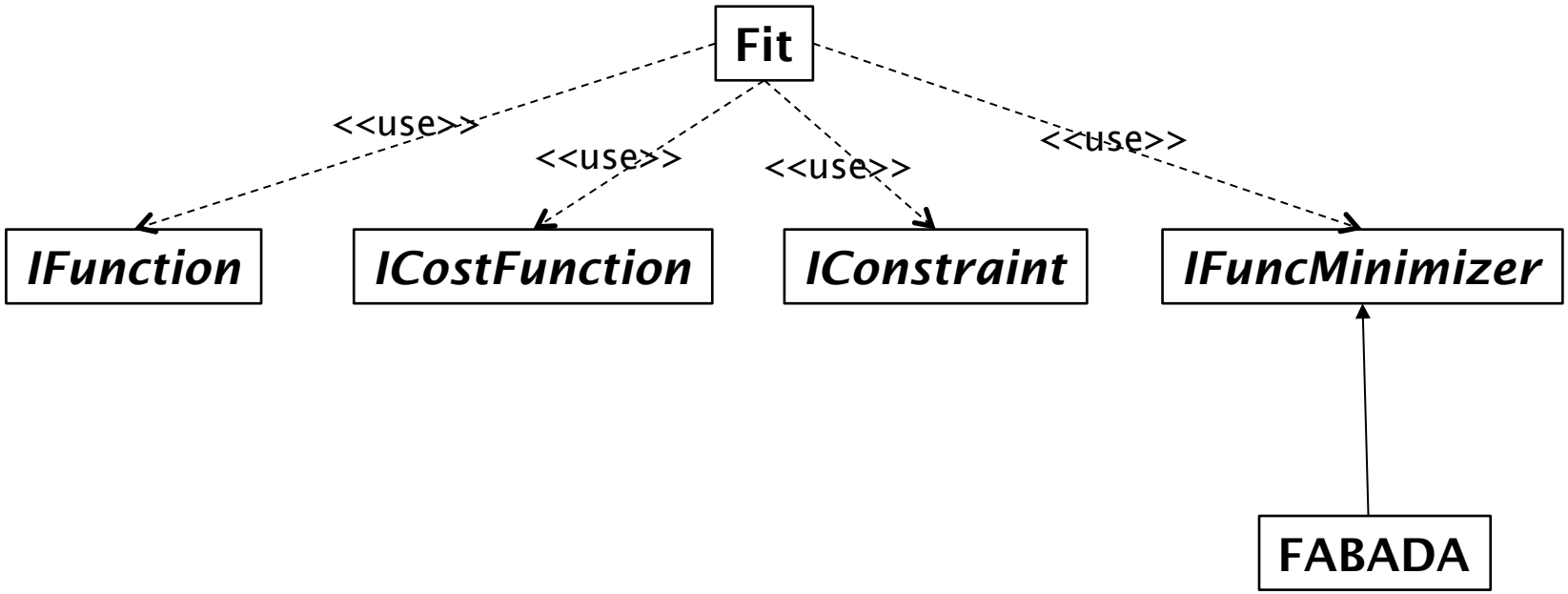
Fit Function in Python

```python
class Example1DFunction(IFunction1D):

    def init(self):
        self.declareParameter("A0", 0.0)
        self.declareParameter("A1", 0.0)

    def function1D(self, xvals):
        a0 = self.getParameterValue("A0")
        a1 = self.getParameterValue("A1")

        # Use numpy arithmetic to compute new array
        return a0 + a1*xvals

    def functionDeriv1D(self, xvals, jacobian):
        i = 0
        for x in xvals:
          jacobian.set(i,0,1) # paramter at index 0
          jacobian.set(i,1,x) # paramter at index 1
          i += 1

FunctionFactory.subscribe(Example1DFunction)
```

Same Fit Function in Python

```cpp
DECLARE_FUNCTION(LinearBackground)

void LinearBackground::init() {
  declareParameter("A0", 0.0, "coefficient for constant term");
  declareParameter("A1", 0.0, "coefficient for linear term");
}

void LinearBackground::function1D(double *out, const double *xValues,
                                  const size_t nData) const {
  const double a0 = getParameter("A0");
  const double a1 = getParameter("A1");

  for (size_t i = 0; i < nData; i++) {
    out[i] = a0 + a1 * xValues[i];
  }
}

void LinearBackground::functionDeriv1D(Jacobian *out, const double *xValues,
                                       const size_t nData) {
  for (size_t i = 0; i < nData; i++) {
    out->set(i, 0, 1);
    out->set(i, 1, xValues[i]);
  }
}
```

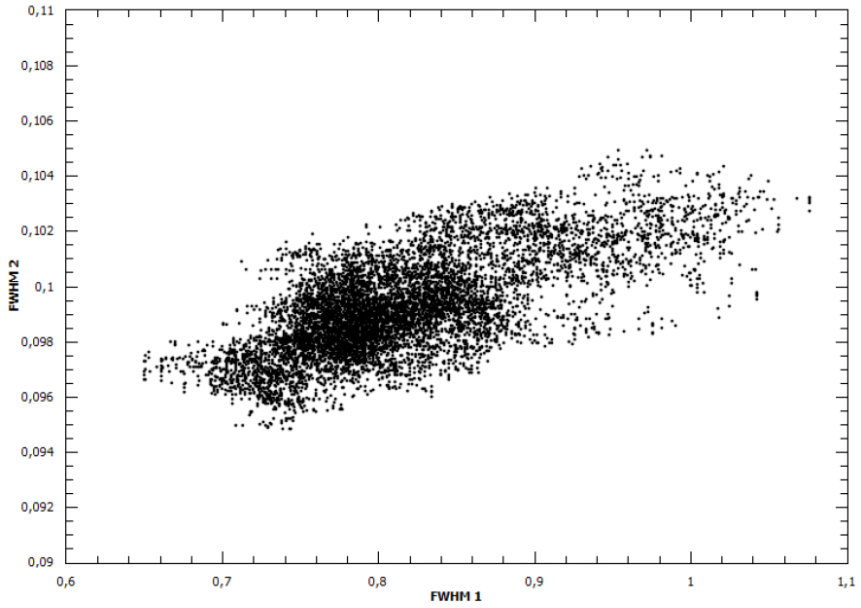
Bayesian fitting



Bayesian fitting: FABADA

- **F**itting **A**lgorithm for **B**ayesian **A**nalysis of **DA**ta

$$P(D_k \mid H_k) \propto \exp\left(-\frac{\chi^2}{2}\right)$$



Science & Technology Facilities Council
ISIS

Guidelines & standard for testing fit minimizers

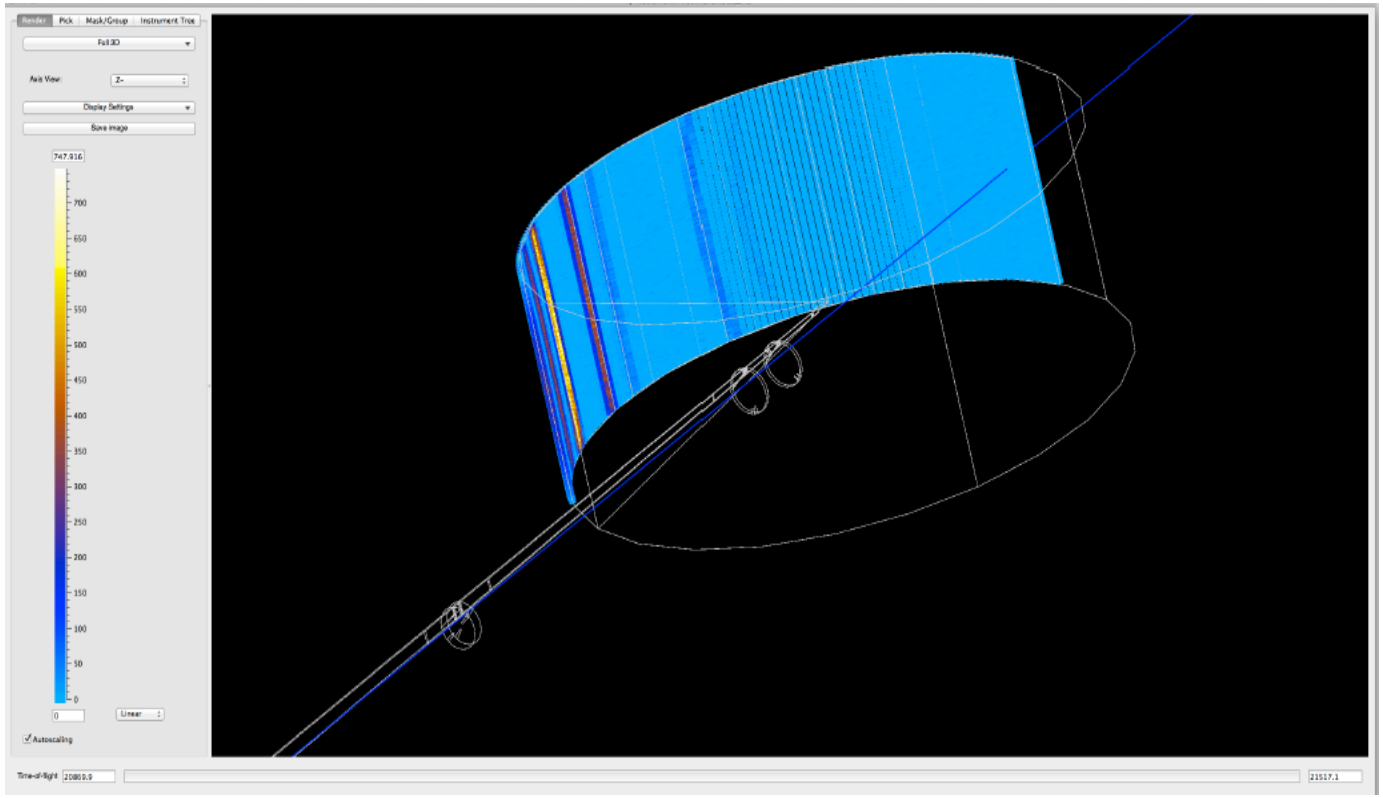
- How do you know if one minimizer is better than another for a certain class of neutron facility fitting problems?

- First demonstration of this is expected by June 2016 (Mantid 3.7)

‘High’ level interoperability

- E.g. SasView/GSAS/McStas/nMoldyn/ASE… linking with Mantid
- Send-to option

‘High’ level interoperability – McStas/Mantid

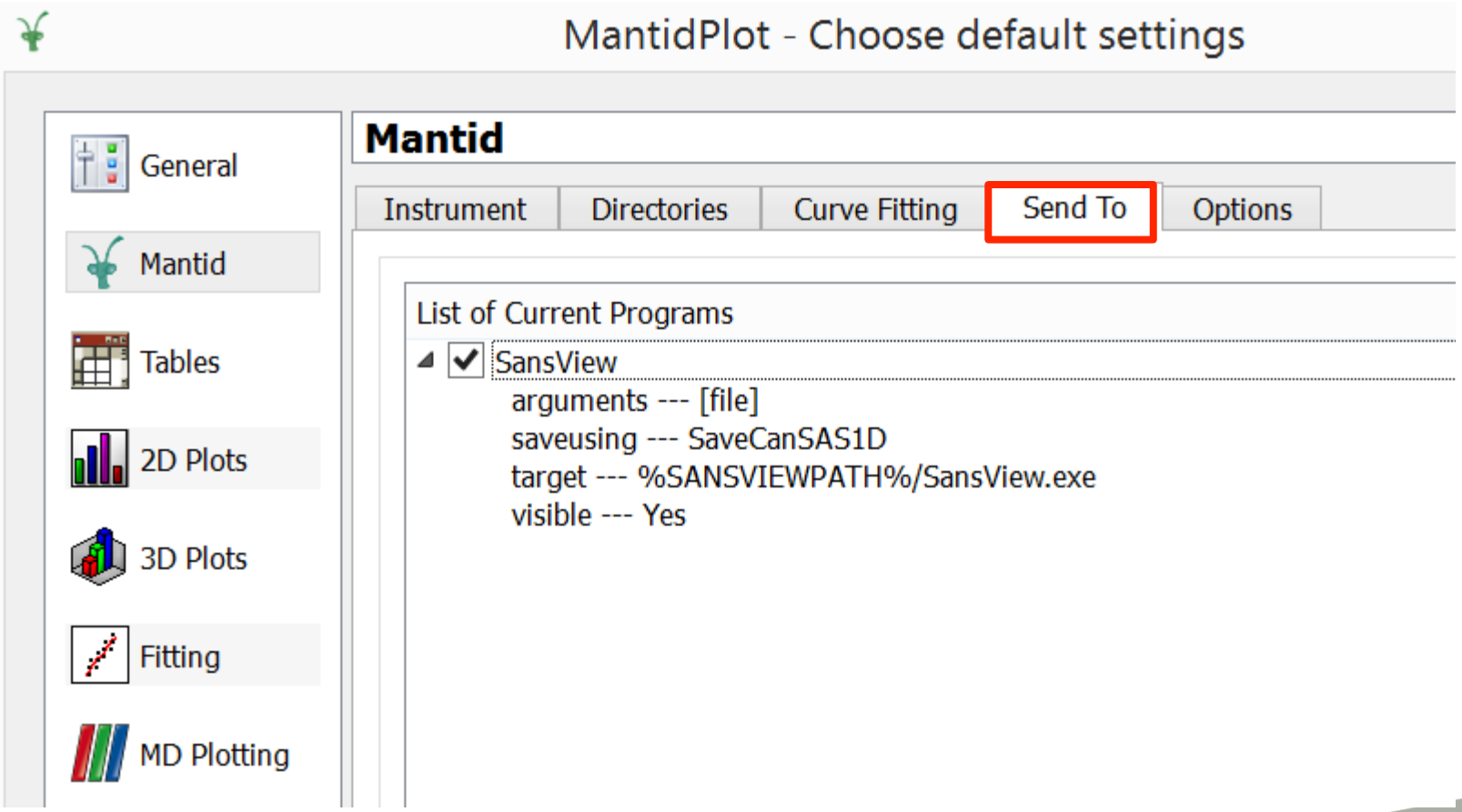


LoadMcStas algorithm

‘High’ level interoperability – GSAS/Mantid

- SaveANSTOAscii
- SaveAscii
- SaveCSV
- SaveCalFile
- SaveCanSAS1D
- SaveDaveGrp
- SaveDetectorsGrouping
- SaveDiffCal
- SaveDspacemap
- SaveFocusedXYE
- SaveFullprofResolution
- SaveGSASInstrumentFile
- SaveGSS
- SaveHKL
- SaveILLCosmosAscii
- SaveISISNexus
- SaveIsawDetCal
- SaveIsawPeaks
- SaveIsawQvector
- SaveIsawUB
- SaveLauenorm
- SaveMD
- SaveMDWorkspaceToVTK
- SaveMask
- SaveNISTDAT
- SaveNXSPE
- SaveNXTomo

‘High’ level interoperability – Send-to option



‘High-high’ level interoperability – ASE

Atomic **S**imulation **E**nvironment

http://www.mantidproject.org/ASE

Further sentences from Task 10.2

- Guidelines and Standards will be provided for application programming interfaces (**API**), command line interfaces (**CLI**), graphical user interfaces (**GUI**), **data formats** as well as **software development environments**

- Application Programming Interface (API)
- Command line interfaces (CLI) – Python
- Graphical user interfaces (GUI) – Qt, reusable widgets, PyQt
- Data formats - NeXus

Standard for documenting data loaders

Data loaded from Nexus File

Not all of the nexus file is loaded. This section tells you what is loaded and where it goes in the workspace.

The nexus file must have `raw_data_1` as its main group and contain a `/isis_vms_compat` group to be loaded.

The workspace data is loaded from `raw_data_1/Detector_1`. Instrument information is loaded `raw_data_1/Instrument`, if available there and not overriden. Also the `NSP1`, `UDET`, `SPEC`, `HDR`, `IRPB`, `RRPB`, `SPB` and `RSPB` sections of `raw_data_1/isis_vms_compat` are read. The contents of `isis_vms_compat` are a legacy from an older ISIS format.

Here are some tables that show it in more detail:

Description of Data	Found in Nexus file (within 'raw_data_1')	Placed in Workspace (Workspace2D)
Monitor Data	within groups of Class NXMonitor (one monitor per group)	Depending on property LoadMonitors, monitor histogram data
Detector Data	group `Detector_1` (all detectors in one group)	Histogram Data
Instrument	group `Instrument`	Workspace instrument if not overridden
Spectrum of each detector ID	`NSP1`, `UDET` and `SPEC` within `isis_vms_compat`	Spectra-Detector mapping
Run	various places as shown later on,	Run object
Sample	`SPB` and `RSPB` within `isis_vms_compat`	Sample Object

Software development environments

	NMI3-II 2014	Now (Mantid)
Repository	Git/SVN Facility based	Git Cloud based
Collaborative tools	E.g. Trac Redmine	Github
Test environments	Jenkins	Jenkins
Unit/Integration testing	Yes please	Yes please
Package builds	Cross-platform	Cross-platform (CPACK)

Software development - Mantid

- Continuous build and testing
  - 20+ build servers managed through Jenkins
  - Static code analysis
  - Over 8,500 automated unit tests
  - Almost 300 automated system tests
- Comprehensive documentation
  - Including 675 automated tests on included scripts
  - Both online and installed with Mantid

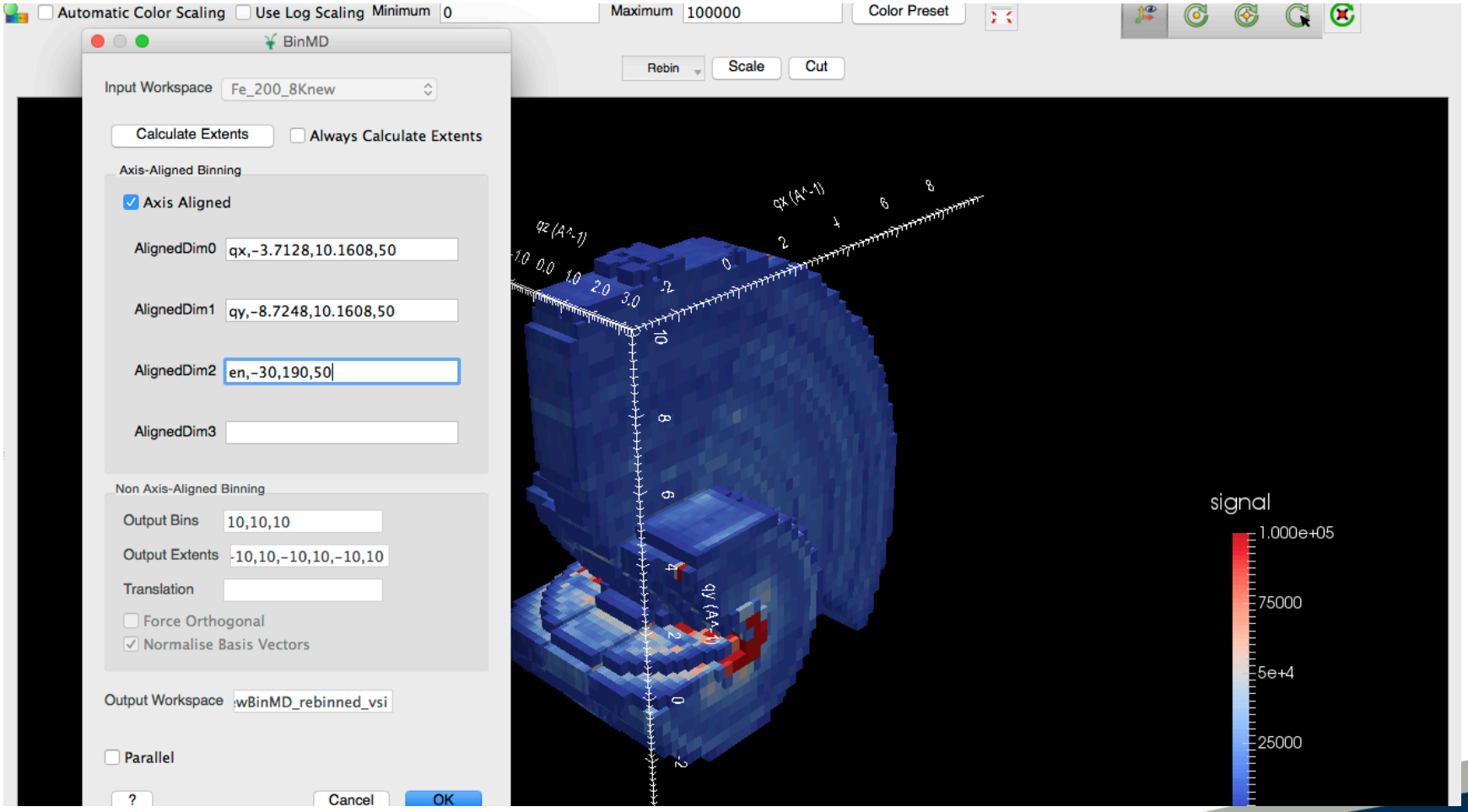
STFC contributing broader: Tasks 10.2-4

- Visualisation
- Fitting
- Interoperability of software

Challenges facing MD visualisation

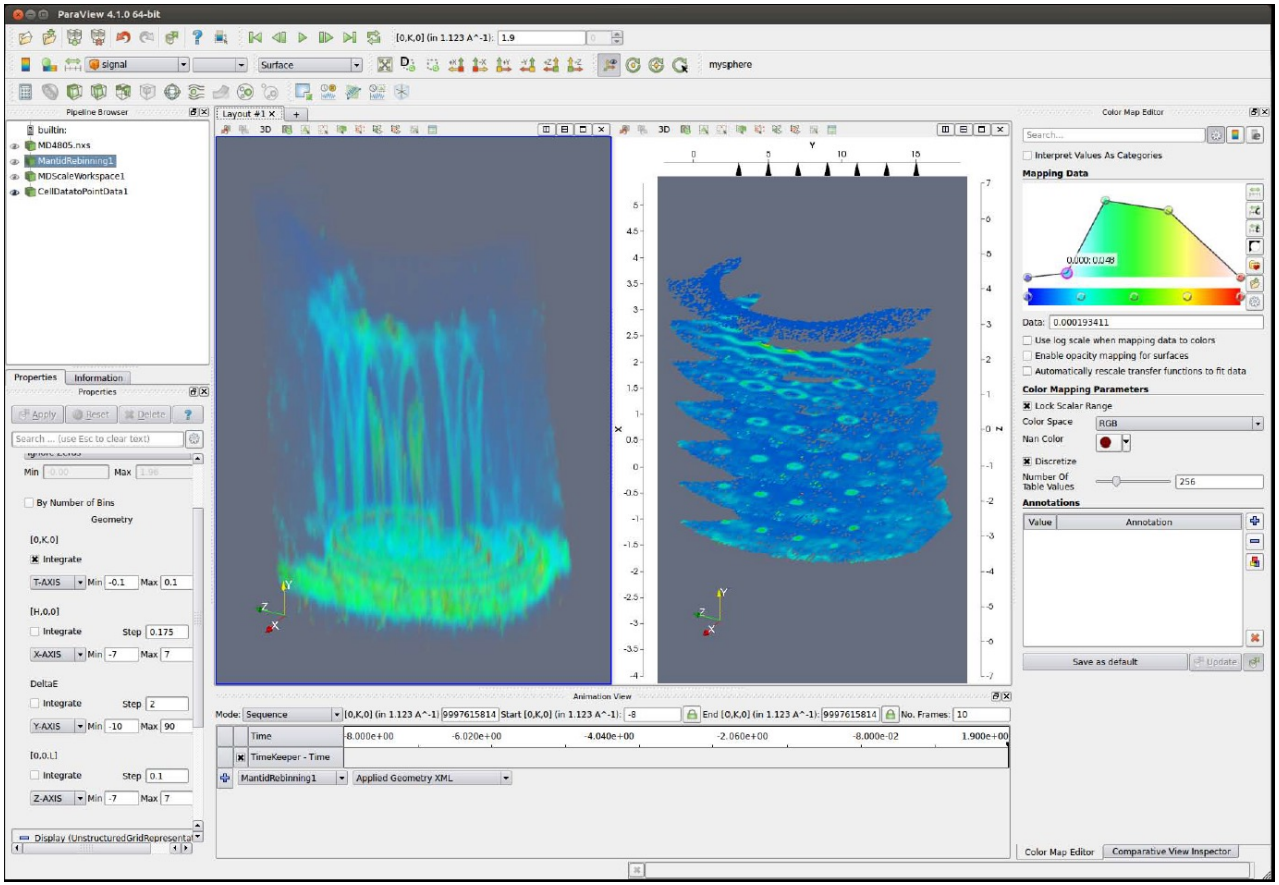
- Visualize massive data sets in n-dimensions
- Efficiently handle sparse as well as dense n-dimensional data
- Operate on such data very quickly
- Usability / customisability for different techniques
- Simulation and fitting

In-situ rebinning



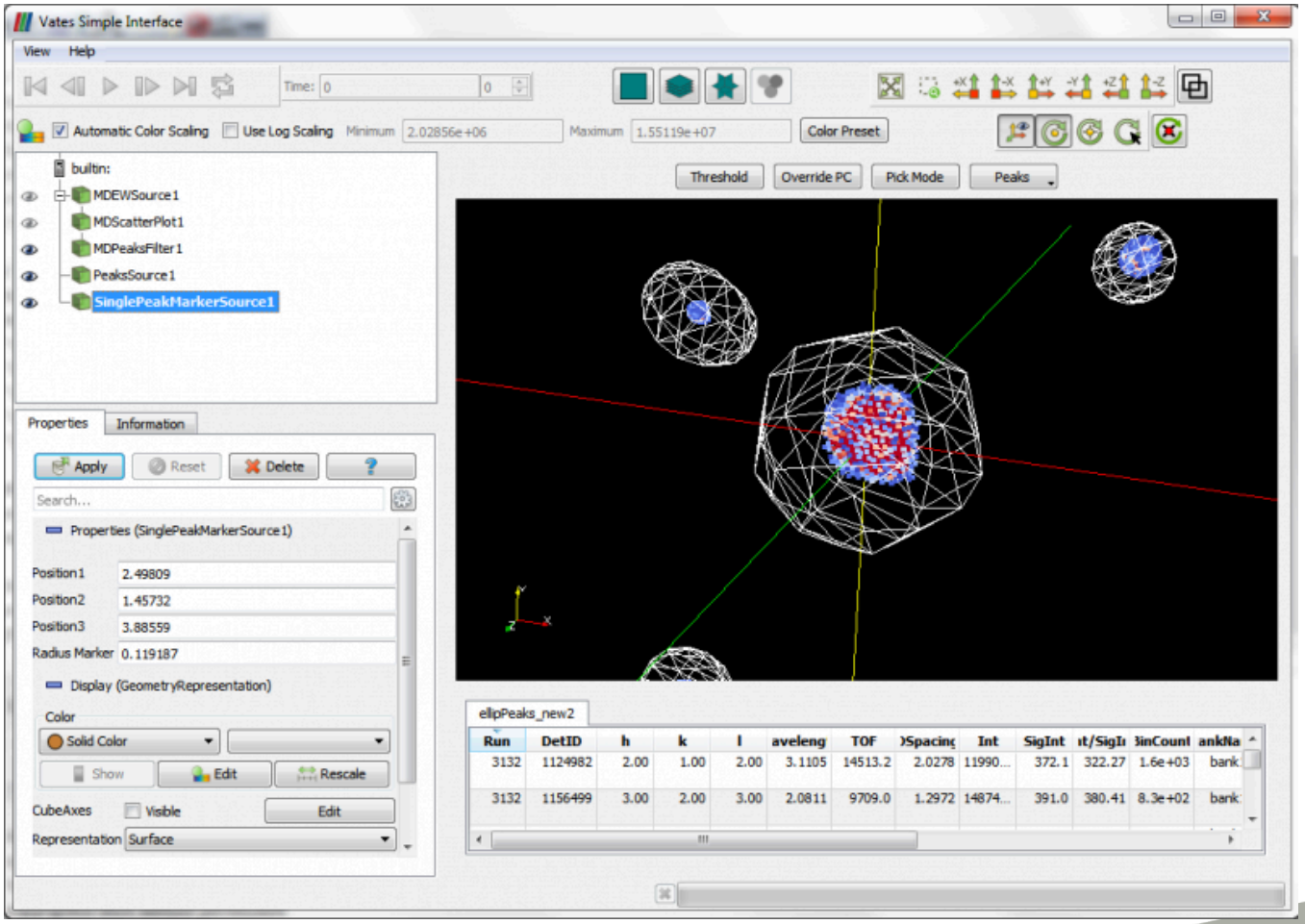
Science & Technology Facilities Council
ISIS

YFeO3 magnetic scattering



Science & Technology Facilities Council
ISIS

Inspecting peak integration

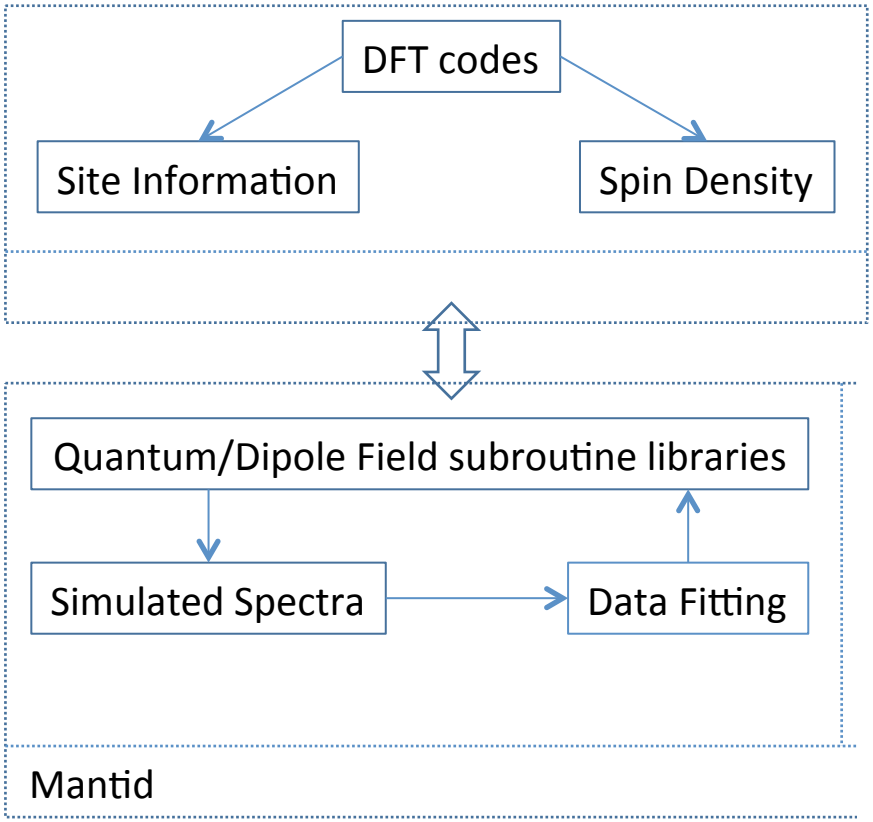


Science & Technology Facilities Council
ISIS

Challenges facing fitting

- Usability
- Some documentation aspects missing
- Fitting large number of datasets simultaneously
- Better minimizers

Interoperability project planned



Challenges with Mantid imaging

- September: first IMAT user
- Filters for pre and post processing
- Wavelength dependent reconstructions
- 3D visualisation

Future Mantid Tasks

- **Multiple Scattering Corrections**
- **Better integration with Third Party simulation codes**
  - VASP, Gaussian, CASTEP, McPhase, SpinW/Spinwavegenie etc
  - Common Language - ASE
- **Better use of facility / cloud resources**
  - Remote desktop – Performance and stability
  - Client server Mantid
  - Web based UI
  - Parallel visualisation rendering

Discussion items

- Standard for a fit function format?
- Bayesian fitting guidelines and/or interoperability library?
- Better send-to options
- Any comments to standard for documenting data loaders
- Visualisation
- Discuss other interoperability projects

- Any other?