Mantid - data analysis and visualisation package for neutron scattering and μSR experiments

O. Arnold[b], J. C. Bilheux[a], J. M. Borreguero[a], A. Buts[c], S. I. Campbell[a], L. Chapon[d], M. Doucet[a], N. Draper[b], R. Fowler[c], M. Gigg[b], M. Hagen[a], V. E. Lynch[a], P. Manuel[c], A. Markvardsen[c], R. McGreevy[c], D. J. Mikkelson[e,a], R. L. Mikkelson[e,a], R. Miller[a], K. Palmen[c], P. Parker[c], G. Passos[c], T. G. Perring[c], P. F. Peterson[a], T. Profen[a], P. Radielli[g], S. Ren[a], M. A. Reuter[a], A. T. Savici[a], J. Taylor[c], R. Taylor[f], R. Tolchenov[b], R. Whitley[c], W. Zhou[a], J. Zikovsky[a]

[a] *Neutron Data Analysis and Visualization, Oak Ridge National Laboratory, Oak Ridge, TN, USA*
[b] *Tessella Ltd., Abingdon, Oxfordshire, UK*
[c] *ISIS Facility, Rutherford Appleton Laboratory, Chilton, Didcot, Oxfordshire, UK*
[d] *Institut Laue-Langevin, Grenoble, France*
[e] *University of Wisconsin-Stout, Menomonie, WI, USA*
[f] *Tessella Inc., Newton, MA, USA*
[g] *Department of Physics, University of Oxford, New Parks Road Oxford, UK*

Abstract

The MANTID framework is a software solution developed for analysis and visualisation of Neutron scattering and Muon Spin relaxation measurements. The framework is jointly developed by a large team of software engineers and scientists at the ISIS neutron facility and the Oak Ridge national laboratory. The objective of the development is to improve software quality, both in terms of performance and ease of use for the the user community of large scale facilities. The functionality and novel design aspects of the framework are described.

Keywords: Data analysis, Data visualization, Computer interfaces
PACS: 07.05.Kf, 07.05.Rm, 07.05.Wr

1. Introduction

The use of large scale facilities by researchers in the field of condensed matter, soft matter and the life sciences is becoming ever more prevalent in the modern research landscape. Facilities such as SNS and HiFR at ORNL and ISIS at RAL have ever increasing user demand and produce ever increasing volumes of data. One of the single most important barriers between experiment and publication is the complex and time consuming effort that individual researchers apply to data reduction and analysis.

The objective of the Manipulation and Analysis Toolkit for Instrument Data or MANTID framework is to bridge this gap with a common interface for data reduction and analysis that is seamless between the user experience at the time of the experiment and at their home institute when performing the final analysis and fitting of data.

The MANTID project is a large international collaboration between STFC (UK) and DOE to co-develop a high performance computing framework for analysis of: powder

and single crystal neutron diffraction data, inelastic and quasi inelastic neutron scattering data, polarised neutron diffraction data, neutron relectometery data, small angle neutron scattering data and μSR data .

The MANTID framework consists of a highly modular C++/Python architecture which supports user built plug-in functions as well as access to powerful visualisation toolkits such as ParaView. This modular design allows users to easily extend the capability of the framework to almost any application. The framework is provided under the GNU open source licence and is built for all commonly used operating systems.

In the past, each instrument (or instruments groups at a given facility) would develop individual bespoke software routines for their own science areas, over the life of a facility >40 years this leads to a vast unmanageable library of mission critical software routines. Such a model is prone to single point failures as individual authors of software leave a facility they take with them the key knowledge of the software they developed. This often leads to much refactoring of existing code as the facility attempts to get back control of its mission critical code.

In this article we describe the MANTID framework and its novel features. MANTID has been developed with the overall objective of giving facilities and the users of facilities access to state of the art bespoke software that is professionally developed and maintained, with a clear science led strategic development and maintenance plan. The manner in which the MANTID framework has been developed is a paradigm shift in software development at large scale facilities. The previous single developer model has been superseded at ISIS and SNS by a well resourced development team. This methodology allows instrument scientists time to determine key software requirements for their user programmes rather than having to develop and maintain software packages, in so doing both the user community and the facility benefit.

The overall ethos of the project is that of abstraction, that is to say code developed within the project should at all times operate on all datatypes from all participating facilities. This idea leads to a framework that is in principle easier to use and maintain.

The Manipulation and Analysis Toolkit for Instrument Data (MANTID) project [22], was started in 2007 at ISIS, and joined by SNS and HFIR in 2010, with the goal of implementing a new framework for data analysis and visualisation for neutron scattering and μSR experiments.

The main objectives for the project are:

- To provide a technique independent, neutron specific framework to reduce, visualise and perform scientific analysis of data
- The framework is developed by a core group of developers using professional software development practices
- Core framework aspects are developed using change control project management practices
- To actively support multiple platforms (Linux, Windows, MacOS)
- The software and documentation will be freely distributable, and open source
- The framework must be easily extensible by instrument scientists and users
- Provision of comprehensive, well maintained documentation

2. Neutron scattering

Neutron scattering is an established technique for determining the structure and dynamics of materials. Using the power of the neutron as a probe of structure and dynamics has generated a large user community with research interests from life sciences to quantum magnetism. To meet the current and future demands in these areas of materials science a there have been a number of new large scale facilities built, or in the process of being built in the last 10 years. These new facilities all pulsed spallation neutron sources rather than reactors. Pulsed spallation sources by definition have a time structure to the neutron production and as a result of this all instruments operate in a detection mode known as time of flight (TOF). TOF neutron instruments have the advantage of being able to collect data over a wide range in $S(q, \omega)$ in a single pulse. In a neutron experiment one must relate measured counts $I\theta, t)$ to a the physically meaningful $S(q, \omega)$. At a modern TOF neutron source it is common for instruments to have $10^5 ncm^{-1}s^{-1}$ and 60k detectors generating large GB size data files. In many experiments It is possible for several files to be combined together to create a large $n$dimensional dataset or volume with a size of .1 to 1 TB Recently pulsed sources have started to collect data in what is called event mode collection. This method simply lists to a file every detected neutron with a time of collection (and other metadata), from the event list one may filter based on time or metadata to create data subsets. This method has several advantages, it is effective for storing sparse data, it allows novel time resolved experiments to be performed. Large data volumes, $n$dimensional data and event mode format add several layers of complexity to the data reduction chain, for the instruments to be fully exploited high performance software is a necessity.

3. *mu*SR

This section is being written by A. Hillier

4. DevelopmentPractices

One of the key aspects of MANTID is the manner in which it is developed. Compared to other similar projects MANTID has a large number of core developers >20. The development team make use many modern software development practices. Code development is science led with a large steering committee.

To ensure high performance for data analysis, but also allow flexibility in how the data is processed, most of the project it is written in C++, with Python bindings.

In order to achieve the stated goals, a large team of scientists and scientific software engineers in Europe and United States are collaborating on this project. For an effective collaboration, we use several software development tools and practices designed to support distributed development teams. New feature requests or defect reports are entered into an issue tracking system. Mantid uses *Trac* [9] for this purpose.

Another tool vital for organising work is the use of a version control system. Mantid uses git [10] repositories hosted at GitHub [11] for the source code, configuration files, and much of the documentation. To allow multiple developers to work in similar areas without interference, developers work on separate branches for each ticket. To verify that there are no cross-platform compatibility issues, each feature branch is merged onto a

'develop' branch whenever new code is ready. It is only after a ticket has been completely addressed and tested that the code changes on the feature branch are merged onto the 'master' branch from which release builds and new features are based.

In order to ensure quality, the Mantid project uses a continuous integration environment built around the Jenkins continuous integration server[12]. Whenever new code is committed to the 'develop' branch, builds for each supported operating system are started, and are tested against a suite of over 6000 automated unit tests. A build is marked successful only if all of these unit tests pass. Every night, a nightly build of the 'master' branch is done once a day. For successful nightly builds, a series of over 150 integration 'system tests' are also run against a locally installed version. Successful builds that pass all system tests are immediately available for download, and in some cases automatically deployed to computers. Stable releases of Mantid software occur approximately every three months, and undergo additional rigorous manual testing by the development team. Stable releases are accompanied by detailed release notes and user training.

5. Mantid Design

One of the main design consideration for this project was the separation of data and algorithms. The ethos of the development is that algorithms should (where possible) operate on all data types without *aprirori* knowledge of the data or the experiment that generated it. In principle this ideology makes the framework cleaner and easier for the scientist to use. It it important to note that the framework is developed for the user community of the facilities, in many instances external visiting scientists are not experts in neutron scattering or the associated data analysis that is required. Any software application written at facilities and funded by governments must take this into account at the design stage. Data containers (called workspaces) and algorithms, which manipulate workspaces, compose the central element of the Mantid Framework (Figure 1). Workspaces and algorithms are aware of the geometry of each individual instrument. Instrument geometries are xml files containing the real space position of all instrument components. Workspaces can be loaded from various file formats, from live data streams, or created by different algorithms. They can be manipulated by algorithms, and saved to disk. By default Mantid uses the NeXus format for saving intermediate and processed data, but various other output formats are also supported.

The interaction with the Mantid Frameworks occurs through the application programming interface (API). While initially a Matlab API was envisioned, currently the main interactions occur through either the Python API, or through MantidPlot graphical interface.

The key features of the framework are:

- Powerful data container workspaces handling multiple data types, experiment logs and experiment metadata
- Abstracted algorithms
- Inbuilt instrument geometry and solid geometry handling
- Handling of multiple data types including event mode data

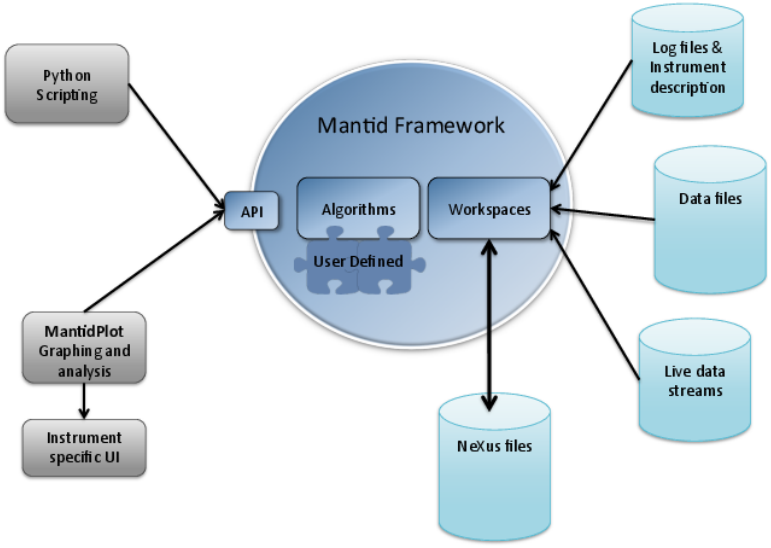

Figure 1: Mantid Framework design

- Python bindings for all aspects of the core framework
- Platform independent code
- Multiple platform builds along with platform specific installers for MANTID, Python, Ipython, SciPy and NumPy
- The framework is openMP aware and designed to maximise performance on multicore machines

5.1. Instrument Geometry

A full description of the instrument is used within the MANTID framework. The instrument definition file is an xml styled description of of all pertinent instrument components see Table1. The IDF component description can be expanded upon to increase the information level accessible to the MANTID framework. Previous applications for neutron scattering data analysis have generally only described instruments by their primary and secondary flighpaths and detector angles ($\theta and \phi$). A full description allows complex visualisation of the instrument and its detectors, along with the possibility to perform monetcaro simulations of the incident flux and resolution using the neutron ray tracing package McStas [**?**] The IDF can be updated from values in log files to account for moving instrument components. The solid geometry engine within the MANTID framework can be used to described complex sample shapes to allow absorption correction to be made.

5.2. Classes

is this required too

Component type	Position information	Supplemental information
Beam Monitor	flightpath position	Monitor type Gas / scint
Detector	X,Y,Z r,t,p	detector type gas/scint gas pressure wall thickness
Neutron guide	dimensions /length	m factor
Beam Chopper	slit dimensions	speed
Beam defining slits	dimension	
Source type	dimension	type /temperature

Table 1: Component types currently stored in the instrument definition in MANTID together with possible position information and metadata

5.3. Data Types

The MANTID framework is capable of reading a variety of data types. The most commonly used are data files written in the nexus standard. However the framework can read legacy files from ISIS as well as generic x,y,e ascii data. Generic and histogram data and read info the workspace2D type of data container. instrument data are populated alongside the IDF description of the instrument. Event mode nexus files are read into event workspaces and can be filtered on TOF at load. Alongside the standard loading of a pre-existing datafile MANTID can also access the instrument data directly to provide real time display of detector counts and live 'on the fly' data reduction.

5.4. Workspaces

Workspaces are the data containers in Mantid. In addition to the data, workspaces can hold other types of information, such as instrument geometry, lattice parameters and orientation, or sample environment logs. Each workspace also holds its history, a list of algorithms that were used to create that workspace. That way each workspace can prove it's provenance, and also regenerate the commands used to make it. Depending on the organization of the data, there are various types and subtypes of workspaces. Table 2

Workpsacetype	dataFormat	mode	metaData	uses
Workspace2D	X,Signal, Error	Histogram / point	sample logs, Spectrum Number detectorID	raw data
EventWorkpsace	eventlist	adaptive mesh rebin	correlated sample log spectrum number detector ID	raw event data
Multidimensional workspace	n eventlist n histogram	adaptive mesh rebin	$n$dim axis units	processed volume data sets
Tableworkspace	column - row format	—	—	fitting results

Table 2: Workspace types in MANTID together with type of data stored, meta data and current uses in the framework for each type of workspace

Workspace2Ds contain data for multiple spectra, in the X, Signal, Error format, where X is a coordinate such as time of flight or energy transfer.

The data acquisition system at several facilities now allow separate recording of each detected neutron, and labelling it with time-of-flight, and wall-clock-time stamps. For each spectrum, the EventWorkspace contains a list of events [21].

EventWorkspaces can also provide a histogram representation as well,which is calculated on request. This allows event workspaces to support the MatrixWorkspace interface,

also being supported by Workspace2Ds. The result is that algorithms and plotting work on the two types of workspaces interchangeably, without the need to know the details of how their data is stored. There are various uses for event workspaces. One can filter out unwanted events, such as events recorded during temperature spikes. The other big use for events is allowing novel techniques, such as asynchronous parameter scans (continuous angle scans, temperature scans), and pump probe experiments (pulse magnets, high frequency deformations of materials, and so on).

For data formats that contain different field types, Mantid provides various TableWorkspaces. A table workspace is organised in columns. Each column has a name and a type - the type of the data in that column. Examples of table workspaces are the outputs from the Fit algorithm, and PeaksWorkspaces, a representation of information about Bragg peaks, that is used in crystallography experiments.

The last major workspace type is the multi-dimensional workspace, or MDWorkspace. While for matrix workspace there are two dimensions describing a data point (spectrum number and X coordinate), for MDWorkspaces we have between 1 and 9 dimensions. Higher number of dimensions are required to accommodate labelling of data with extended parameter dependencies, *i.e.* sample environment variables. MDEventWorkspaces are adaptivley rebbined onto a mesh that is dependent on the number density of events Fig2.

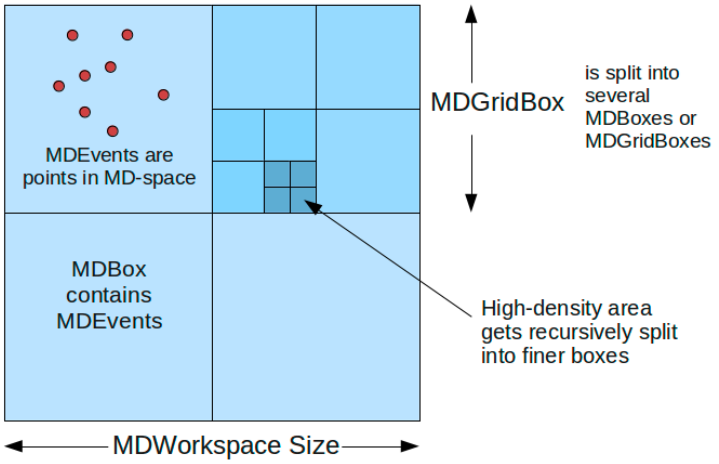

Figure 2: Schematic representation of the principle of adaptive rebinning used in the MdEventWorkspace type.

For MDEventWorkspaces, each MDEvent contains coordinates, a weight and an error. It might contain also information about which detector and which run it come from. All MDEvents are contained in MDBoxes. Above a certain threshold, the MDBox becomes an MDGridBox, by splitting into several equal size MDBoxes. This allows for an efficient searching and binning, and allows plotting on an adaptive mesh. MDHistoWorkspaces consist of signal and error arrays on a regular grid.

5.5. Algorithms

Mantid algorithms are procedures to manipulate workspaces. They can be predefined, or written by users, in either C++ or Python. The algorithm layer is a key aspect of the MATID framework. The organisation and development of algorithms is key to maintaining the ethos of the project; that is all algorithms should operate on all data for all instruments. This ethos therefore presents a number of challenges for development. The framework can access multiple data types, including event mode data. As a result of this the algorithm layer must be able to cope with multiple data type input and at the same time maximise performance for individual data types. The case of event mode data in interesting as it presents an efficient way of processing sparse data. It is therefore more efficient to keep the data as events through a chain of operations. This requirement has resulted in development of a number of event specific data handling operations. The end result is that for many reduction chains the data is event type until the final rebin for presentation. The key advantage of this for the end user is flexibility.

Core algorithms can be grouped together to form bespoke data reduction and analysis for individual instruments and science areas. These large algorithms can then be presented to the user at the python scripting layer, command line interface or as a custom reduction user interface.

At the present, there are over 500 algorithms covering data handling (loading/saving workspaces from/to files), arithmetic operations(plus, minus, multiply), unit conversions, and many technique specific algorithms (powder diffraction, single crystal diffraction, SANS, reflectometry, direct and indirect spectrometry, and μSR).

5.6. Python API and scripting

The python API for the MANTID framework provides an exceptionally powerful interface to the core C++ framework. All classes within the framework are open to python control. The python API can be used to simply interact with algorithms and workspaces or as a method to quickly expand the framework with new functionality.

The API has been written to give a intuitive command line interface feel.

something more is required here

6. User Interface

6.1. MantidPlot

The main interaction with Mantid occurs through the MantidPlot interface (Figure 3). It is a graphical user interface based on QtiPlot[14]. It allows 1D, and 2D plots of the data, and access to the VATES interface for MDWorkspaces (see section **??**). From the MantidPlot window, accessing the Python interface can be achieved through the script window (used to run entire scripts at once) the script interpreter or python shell.

A list of all algorithms is also present by default, organized both alphabetically, and by category. Clicking on an algorithm will open an automatically generated dialog box, with entries for each of the input parameters. A quick validation occurs when information is filled, and if any input is invalid it is flagged with a error message for the user. For each algorithm dialog box, a button allows for invoking the built-in help. A results log window is also available, where users can see the results of running different algorithms.

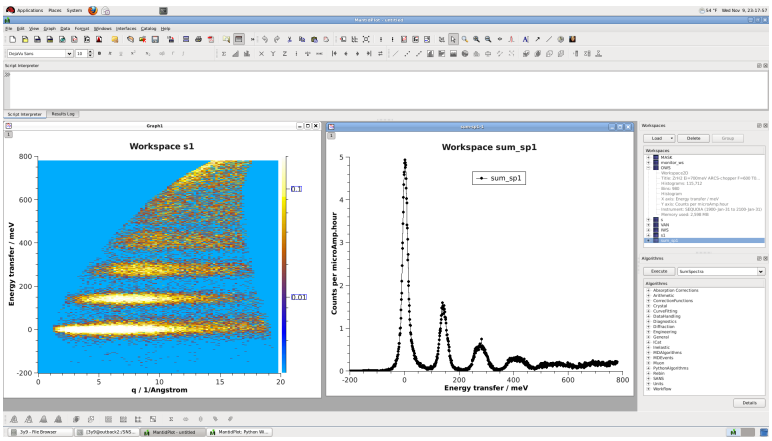

Figure 3: MantidPlot interface, showing 1D, and 2D plots. Lists of workspaces and algorithms are available on the right side

For several scientific techniques, custom interfaces are available from the MantidPlot menu.

A list of available workspace is present by default. Clicking on the workspaces show information about workspace type and content. A context sensitive menu allows simple plotting, instrument view, inspection of the sample environment logs, or a listing of the history of the workspace.

6.2. Data reduction and bespoke interfaces

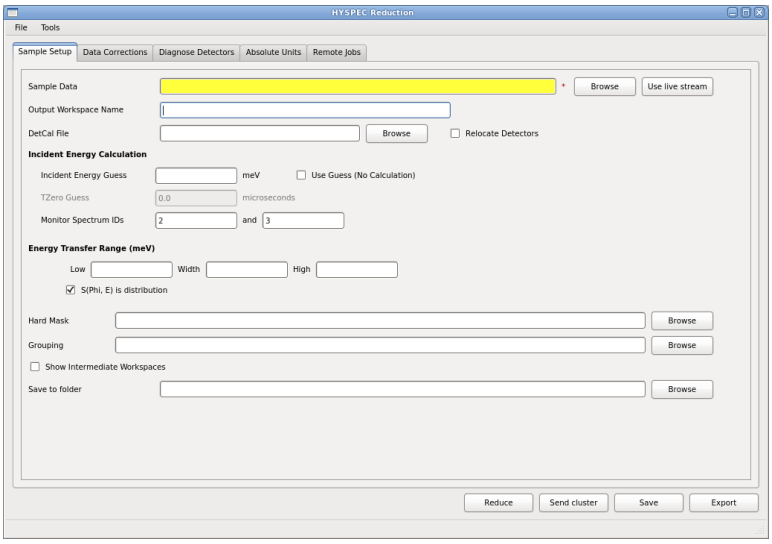

Figure 4: Custom interface for direct geometry reduction on HYSPEC instrument

Data reduction and basic analysis for individual instruments or science areas is generally a sequential chain of operations starting from data loading and resulting in a dataset that has physically intractable units. As such reduction scripts for several scientific techniques can be complicated, and depending on a large number of parameters. More often than not development of new features in this area must take into account legacy usage requirements and be well validated against existing "known" good results. In all cases development of data reduction chains are tightly controlled and validated.

One of the objectives of MANTID is to provide the facility user with a simple and efficient interface to allow them to analyse their data. To achieve this for multiple science areas and instruments a number of custom interfaces have been implemented to process data from specific experiments. These are presented to the user through a GUI layer and often a "super algorithm" A number of bespoke GUIs have been implemented to deal with utility functions useful for neutron science. Science areas and instruments specifically supported by the MANTID framework can be seen in Table3

Science area	Instruments
Powder neutron diffraction	GEM HRPD WISH POLARIS POWGEN
Single crystal neutron diffraction	WISH SXD TOPAZ
Inelastic neutron scattering (direct)	MERLIN MAPS MARI LET SEQUOIA ARCS HYSPEC CNCS IN4 IN5 IN6
Inelastic neutron scattering (indirect)	BASIS IRIS OSIRIS TOSCA VISION
Small angle neutron scattering	SANS2D LOQ EQ-SANS GP-SANS BIO-SANS D33
Neutron reflectometery	CRISP SURF POLREF INTER OFFSPEC
μSR	MUSR HIFI EMU

Table 3: Current science areas and instruments supported by the MANTID framework

In some cases a single 'super algorithm' is beneficial, one such case is for live event process. The application can access the live data steams of event mode instruments at SNS and ISIS and can directly read histogram data from the detector electronics of ISIS instruments. Fig 5 shows the generic workflow to process and view data as it is collected.

The custom user interfaces, available from MantidPlot, group together inputs from related reduction parameters, and spread independent steps onto different tabs. Figure 4, shows an example, the DGS Reduction interface for the HYSPEC instrument at SNS this is a generic reduction chain for inelastic neutron scattering instruments.

Paz may need to expand on this The framework has inbuilt methods to process single crystal diffraction data, along with python routines to calculate and refine the orientation matrix, index reflections integrate reflections and generate goiniometer translations for crystal alignment.

should we add more in this section about other types of reduction key features etc

6.3. Fitting

Fitting mathematical functions and models to experimental data is a key requirement of any scientific computing application. The MANTID framework has implemented a powerful fitting engine for fitting peak functions and simple user derived functions to line data i.e. data that is in 1D x,y,e format.

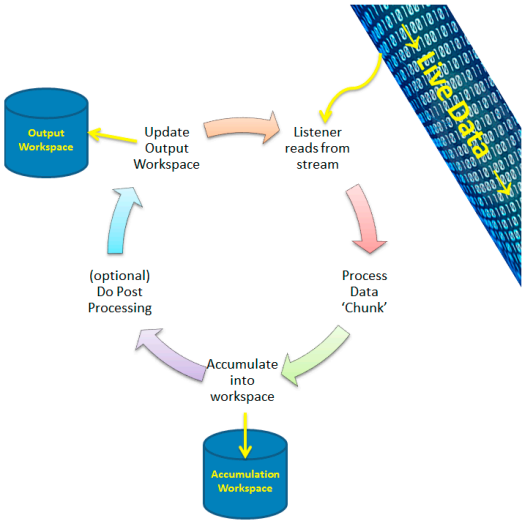

Figure 5: Flow diagram representing data flow when using live data processing within MANTID

The user interface has been deliberately developed to allow complex function to be created from simple mathematical building blocks in a GUI framework. Any function or dataset can be fitted within the scripting python interface.

Once the user has generated a model the subsequent fitting can be batch processed across many different datasets, with the option of plotting fit results against a log parameter. Output from the fit procedure is displayed as a tableWorkspace data set which can then be further manipulated.

Figure 6 shows the GUI used for fitting mathematical functions to 1D data.

There are a number of scientific techniques that always fit complex function to reduced datasets in order to extract meaningful physical parameters. For the case of MUSR and QENS the MANTID framework fitting module is sufficient. For SANS the data can be output to external model fitting packages for further analysis. MANTID curly can link directly to $SASView$ for further analysis of experimental data.

In the area of inelastic neutron scattering fitting a single resolution broadened model of $S(q, \omega)$ to a n dimensional $S(q, \omega)$ data set is a standard data analysis procedure. In the past programs like $Tobye$Fit [**?**] have been used to perform this type of analysis. The MANTID framework can replicate $Tobye$Fit functionality for inelastic neutron scattering data.

7. Visualisation

Modern multi detector instruments generate large data sets which cannot be easily visualised on a 1D graph or a 2D projection. Large position sensitive detector backs allow diffractometers and spectrometers to survey large areas of reciprocal space in a single experiment. A large multi detector instrument typically has 60-100k pixels visualising experimental data in real space of the entire instrument is often useful. For single crystal

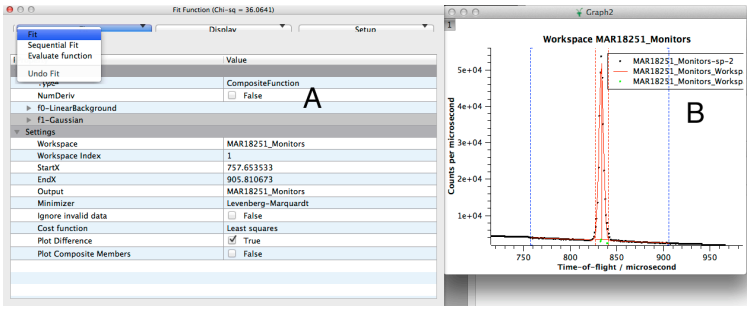

Figure 6: The simple fitting GUI interface in the MantidPlot application. Peak selection is performed using mouse selection on the displayed data. Panel A displays the required model and fit controls panel B displays the data, fitted model and difference the vertical dotted lines indicate extents in x for the model.

samples such instruments become large Laue geometry cameras. For SANS experiments area detectors of 1x1m are commonplace and place another requirement on visualisation. For visualising the instrument data in real space the InstrumentView can be used, and is described later. **??**

Often experiment perform crystal rotations to generate even grater maps of reciprocal space and for the case of inelastic neutron scattering that resolves both reciprocal space and energy transfer the data volume that is produced is in greater than 3 dimensions. Volumes of data are inherently large in memory and require special techniques to visualise efficiently. Volume data in MANTID is encapsulated in the MDevent workspace type which uses dynamic box rebining to maximise efficiency and minimise memory allocation. MDevent workspaces can be viewed using the sliceViewer**??** described later or the entire volume can be rendered and manipulated using *paraView* (PV). A simple instance of PV has been developed (*vsiulisationtoolkitVATES*) specifically for neutron specific methods and can be instantiated directly from within MantidPlot, from the MDEvent-workspace context menu. The MDevent datatype and associated visualisation tools is not confined to processed data in reciprocal space and can be used to visualise 3D volume data from neutron topography experiments. The application has been developed in this way to ensure that the scientific workflow is intuitive and efficiency.

7.1. Instrument View

The instrument view (IV) is a 3D representation of the whole instrument component positions are calculated from the IDF. Individual IDF components, i.e. choppers, guides etc can be toggled on or off. Detectors are shaded with a colour representative of the goal integrated counts. The IV allows for quick access to information about detectors

and spectra, and provide a simple graphical interface for masking and grouping. The IV has a number of useful features:

- A 2D projection of the detectors can be selected.
- Users can group mask or select shapes or ranges of detectors and save the output as a separate workspace.
- Groups of detectors can be flagged to be ignored by subsequent algorithms (masking)
- Users can quickly look at individual spectra in what ever x axis unit is currently selected
- brag peaks can be selected and saved a table workspaces
- For instruments that use position sensitive detectors the counts in each pixel can be plotted as a function of position along the tube.

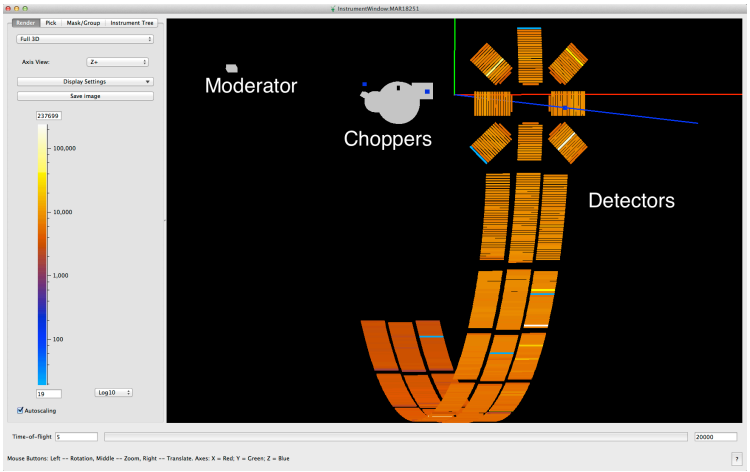

Figure 7: The instrument view GUI in MantidPlot showing a 3D representation of the MARI spectrometer. Various components are annotated.

7.2. Slice viewer

One tool for visualising multi-dimensional (MD) data is the SliceViewer (Figure 8). The SliceViewer provides an interactive 2D projection of multiple data types, both common 2D data and MDWorkspaces Advanced features provide interactive line integration or overplotting integrated or non-integrated single crystal peak locations and regions.

7.3. VATES

A major objective of this project has been the ability to represent multidimensional data [16, 4, 17]. Originally the Visualization and Analysis Toolkit Extensions (VATES) project was an add-on to Mantid that is now fully integrated into the project. For visualising 3 or more dimension datasets, Mantid provides several options.

The VATES Simple Interface (*VSI*), offers a stock set of data views and access to a subset of Mantid algorithms. It is based on application widgets and rendering libraries from the ParaView[18] visualisation program. The *VSI* takes advantage of the ParaView plugin architecture to provide functionality from within Mantid and from within ParaView standalone. The data in Mantid to be visualised passes through an API layer which translates the internal Mantid data structure to a VTK[19] data structure, that can be rendered in the *VSI*. Those same data structures can be saved to file and visualised in the ParaView program. Indeed, it is possible to drive some aspects of multidimensional analysis directly from ParaView. The API layer provides the desired decoupling of the data structures and provides good flexibility to handle the various needs of the Mantid data structures and algorithms.

The *VSI* has a view called MultiSlice which allows placing multiple orthogonal slices on the data. Those slices can then alternately be viewed in SliceViewer. The SplatterPlot (Figure 10) view is oriented towards visualising peaks in single crystal diffraction data. In that view the user can interact with the data to retrieve information about a selected peak. The ThreeSlice view shows three orthogonal planes through the data with the capability exploring via moving a crosshair in one of the planes with a coordinate readout in each plane to show the location. The *VSI* has the ability to show the data with non-orthogonal axes such as the data in Figure 9. This capability was implemented by Kitware[20] via the SNS in support of the Mantid project.

Ongoing work on the VATES interface aims to provide full scriptable control over all the visualisation tools described, as well as increased support for multidimensional processing in technique specific areas.

8. Community involvement and expandability

The MANTID framework provides facility users with a very powerful data analysis tool. The python API gives the user the ability to expand functionality for many different

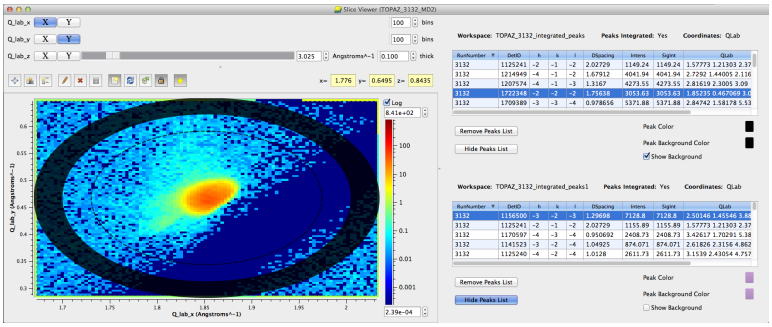

Figure 8: SliceViewer showing a single crystal peak and related information.

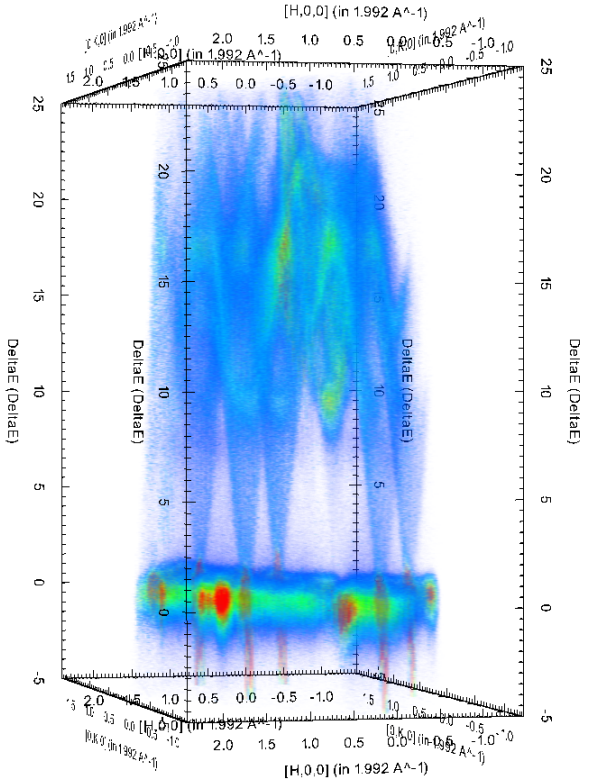

Figure 9: Volume rendering of Gd excitations, in ParaView. Data was measured on SEQUOIA spectrometer at SNS.

applications. User generated python applications can be submitted to the MANTID script repository. The script depository can be assessed from with MantidPlot allowing upload and download of user generated content. The application can be used to analyse most types of experimental data if required. The MantidPlot application can be used to submit bug reports, requests for assistance and viewing of the online and offline help files.

9. Facility integration

A very important step in Mantid development and deployment is facility integration. Both at SNS and ISIS, the development team is working on the interface between Mantid and the data acquisition systems, in order to allow users to look and analyse their data in real time, before a file is even written to the disk. To assist in this Mantid interfaces with Information CATalog (ICAT) [7].

The ICAT cataloguing software provides a well defined API to interface with data at large research facilities. It is in use at both ISIS and SNS, and provides a mechanism to link all aspects of the research chain, from proposal through publication. Mantid uses

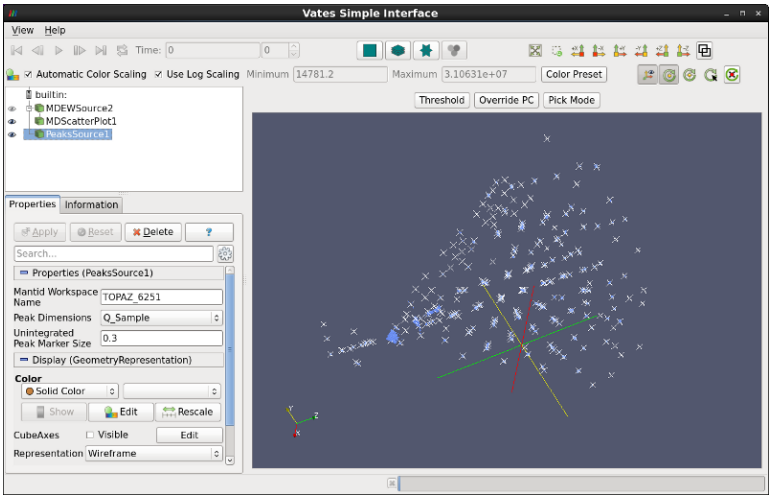

Figure 10: VSI in splatter plot mode with single crystal data from TOPAZ difractometer.

the provided interface for to locate raw or processed files in data archives at each of the facilities.

One important use of the ICAT interface and Mantid is the autoreduction process. As soon as files are created and catalogued, an reduction script is automatically invoked. This script uses metadata in the file and/or the ICAT catalogue to reduce the raw file to a format that contains the data that users are interested in. **NOT TRUE - THIS IS DONE BY THE WORKFLOW MANAGER**

10. Conclusion

The Mantid project offers an extensible framework for data manipulation, analysis and visualization, geared toward neutron scattering and μSR experiments. It is the main reduction software in use at SNS and ISIS, and partially in use or considered at several other scientific facilities. Up to date information, and usage tutorials can be found on the Mantid web page[13].

11. Acknowledgements

The development team would like to thank all instrument scientists and ISIS and SNS for their feedback and help. Work at ORNL was sponsored by the Scientific User Facilities Division, Office of Basic Energy Sciences, US Department of Energy. Work at the ISIS facility was funded by the Science and Technology Facilities council (STFC) UK. Development for ILL instruments was funded by NMI3 (WP6)

[1] Takeshi Egami and Simon J.L. Billinge, Editor(s), Pergamon Materials Series, Pergamon, 2003, Volume 7

[2] H. Nojiri *et al.*, Phys. Rev. Lett. 106, 237202 (2011)
[3] X.L. Wang *et al.*, Scientific Reports 2, 747 (2012)
[4] R.T. Azuah *et al.*, J. Res. Natl. Inst. Stan. Technol. 114, 341 (2009).
[5] S.I. Campbell *et al.*, arXiv:cond-mat/0210442
[6] D. Richard, M. Ferrand and G.J. Kearley, J. Neutron Research 4, 33-39, 1996
[7] http://www.icatproject.org/
[8] T. G. Worlton *et. al.*, Neutron News 15(3),14-15 (2004)
[9] http://trac.edgewall.org/
[10] http://git-scm.com/
[11] https://github.com/
[12] http://jenkins-ci.org/
[13] http://www.mantidproject.org
[14] http://soft.proindependent.com/qtiplot.html
[15] S. Cottrell *et. al.*, Physics Procedia 30, 20-25 (2012)
[16] R. Coldea, MSlice http://mslice.isis.rl.ac.uk
[17] T.G. Perring *et. al.*, http://horace.isis.rl.ac.uk
[18] A. Henderson, ParaView Guide, A Parallel Visualization Application, Kitware Inc.(2007)
[19] W. Schroeder *et. al.*, The Visualization Toolkit, Kitware Inc.(2006)
[20] http://www.kitware.com
[21] J. Zikovsky *et. al.*, Event-Based Processing of Neutron Scattering Data, in progress.
[22] J. Taylor *et. al.*, Bulletin of the American Physical Society 57 (2012)