Mantid - Data Analysis and Visualization 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], S. J. Cottrell[c], M. Doucet[a], N. Draper[b], R. Fowler[c], M. A. Gigg[b], M. E. Hagen[a], A. Hillier[c], V. E. Lynch[a], P. Manuel[c], A. Markvardsen[c], R. L. McGreevy[c], D. J. Mikkelson[e,a], R. L. Mikkelson[e,a], R. Miller[f], K. Palmen[c], P. Parker[c], G. Passos[c], T. G. Perring[c], P. F. Peterson[a], F. Pratt[c], Th. Profen[a], P. G. Radielli[h], S. Ren[a], M. A. Reuter[a], A. T. Savici[a], J. W. Taylor[c], R. J. Taylor[g,a], 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] *Computing and Computational Science Directorate, Oak Ridge National Laboratory, Oak Ridge, TN, USA*
[g] *Tessella Inc., Newton, MA, USA*
[h] *Department of Physics, University of Oxford, New Parks Road Oxford, UK*

Abstract

The Mantid framework is a software solution developed for analysis and visualization of neutron scattering and muon spin measurements. The framework is jointly developed by a large team of software engineers and scientists at the ISIS Neutron and Muon 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 Spallation Neutron Source (SNS) and High Flux Isotope Reactor (HFIR) at Oak Ridge National Laboratory (ORNL) and ISIS at Rutherford Appleton Laboratory (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 the Science and Technology Facilities Council (STFC) (UK) and the Department of Energy (DOE) (US) to co-develop a high performance computing framework for analysis of: powder and single crystal neutron diffraction data, inelastic and quasi-elastic neutron scattering data, polarised neutron diffraction data, neutron reflectometry 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 visualization toolkits such as ParaView[1]. This modular design allows users to easily extend the capability of the framework to almost any application. The framework is provided under the GNU General Public Licence version 3[2], 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[3, 4, 5, 6], 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 refactoring of existing code as the facility attempts to get back control of its mission critical software.

In this article we describe the Mantid framework and its novel features. Mantid has been developed with the overall objective of giving facilities and their users access to state of the art bespoke software that is professionally developed and maintained, with a clear science led strategic development and maintenance plan. 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 data types from all participating facilities. This idea leads to a framework that is, in principle, easier to use and maintain.

The Mantid project[7] 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 visualization for neutron scattering and μSR experiments. The main objectives for the project are:

- To provide a technique independent, neutron and muon specific framework to reduce, visualise and perform scientific analysis of data
- To ensure quality by following professional software development practices
- To actively support multiple platforms (Linux, OS X, Windows)
- The software, source and documentation will be freely distributable
- 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. It 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, 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 are 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 to the physically meaningful $S(q, \omega)$.

At a modern TOF neutron source it is common for instruments to have $10^5 n\ cm^{-1} s^{-1}$ and millions of pixels, generating 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 up to 1 TB. Recently pulsed sources have started to collect data in what is called event mode. 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 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. Muon Spin Relaxation/Rotation/Resonance (μSR)

Muons provide a local probe to investigate the properties of a wide range of materials. μSR has wide applicability and provides useful dynamic information for a broad range of science from soft matter to quantum magnetism, which is often complementary to that from neutron scattering. The technique is similar to that of nuclear magnetic resonance, in which the polarisation of the target nuclei, in this case the muon, is tracked as a function of time. In the case of muons, spin polarised muons are implanted into the material under investigation and these muons decay into positrons which are emitted preferentially along the final spin direction of the muon. By time stamping the detected positrons the muon polarisation is inferred. As muons are produced by the decay product of pions, which in turn are produced by high energy protons (~800 MeV at ISIS), experiments are conducted at proton accelerators and are often situated next to spallation neutron sources, e.g. ISIS, PSI and J-PARC. This means that the users of neutron instruments often use muons as well and having a familiar software framework for analysis is clearly beneficial. The Mantid framework fulfils this requirement, comprising a wide range of methods with which to analysis the muon depolarisation spectrum: integrated asymmetry, Fourier transform, maximum entropy and time domain analysis among others. Moreover, simulations of muon data using Density Functional Theory or electronic calculations can yield further insights into the material under investigation. The ability to link these simulations with the data analysis with a simple interface yields a very powerful tool for the analysis of muon experiments. Again, the Mantid framework offers this functionality to the instrument user.

4. Development Practices

One of the key aspects of Mantid is the manner in which it is developed. 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.

Another tool vital for organising work is the use of a version control system. Mantid uses git [8] repositoriesfor 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 feature. 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 feature branch has been completely addressed and tested that the code changes are merged onto the 'master' branch from which release builds and new features are based.

In order to ensure quality, the Mantid project uses continuous integration. 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. Once a day, a series of over 150 integration 'system tests' are run with the most recent locally installed version. Builds that pass all system tests are immediately available for download and, in some cases, automatically deployed to computers. Formal releases of Mantid occur approximately every three months, and undergo additional manual testing. These releases are accompanied by detailed release notes and 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 *a priori* knowledge of the data or the experiment that generated it. In principle, this ideology makes the framework cleaner and easier to use. In many instances, scientists are not experts in neutron scattering, μSR, or the associated data analysis that is required. Successful software application written for scientists 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. Workspaces can be loaded from various file formats, live data streams, or created by different algorithms. The workspaces can be manipulated by the many algorithms in Mantid, and saved in a variety of formats. By default, Mantid uses the NeXus format[9] for saving intermediate and processed data, but various other output formats are also supported.

To ensure high performance for data analysis, but also allow flexibility in how the data is processed, the project is written in C++ with Python bindings. For parallel processing, Mantid uses OpenMP[10], Posix threads and MPI[11]. The interaction with the Mantid Framework occurs through the application programming interface (API).

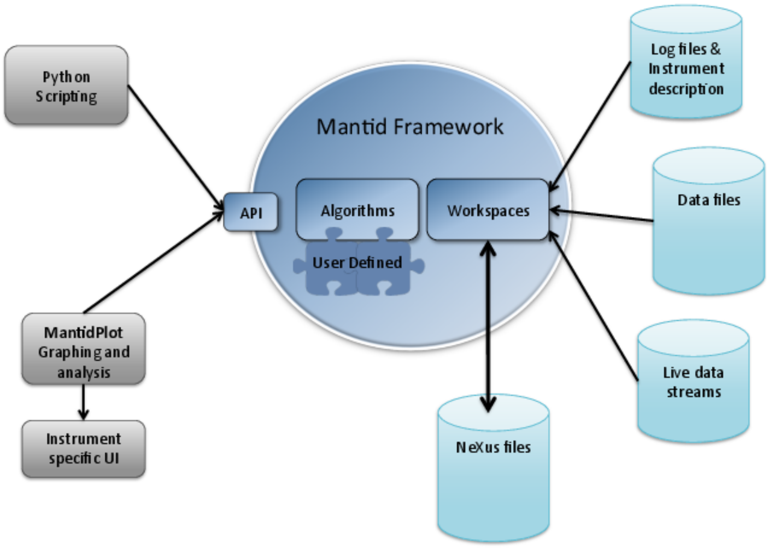

Figure 1: Mantid Framework design

Currently the main interactions occur through either Python or through the graphical interface.

5.1. Instrument Geometry

A full description of the instrument is used within the Mantid framework. One way to specify the geometry is the instrument definition file (IDF). The IDF is a XML description of all pertinent instrument components.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 flight paths and detector angles. A full description, based on constructive solid geometry, allows complex visualization of the instrument and its detectors, along with the possibility to perform Monte Carlo simulations. To account for moving instrument components, the instrument geometry is updated using log values.

5.2. Data Sources

The Mantid framework is capable of reading from a variety of data sources. 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 ASCII data. 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 processing.

5.3. Workspaces

Workspaces are the data containers in Mantid. In addition to the data, workspaces can hold other types of information, such as instrument geometry, sample environment logs, lattice parameters and orientation. Each workspace also holds a history of the algorithms that were used to create it. That way each workspace can show its 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.

MatrixWorkspaces contain data for multiple spectra, as independent variable (e.g. time of flight, energy transfer), signal, and uncertainty. This is a common way to store histograms.

The data acquisition system at several facilities now allow recording of each detected neutron, labelling it with time-of-flight and wall-clock-time. In Mantid, this is stored as EventWorkspaces [12]. EventWorkspaces also provide a histogram representation as well, which is calculated on the fly. This allows EventWorkspaces to be viewed as MatrixWorkspaces by the rest of the Framework. The result is that algorithms and plotting work without the need to know the details of how data is stored. There are various uses for EventWorkspaces. 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).

Another workspace type is the multi-dimensional workspace, or MDWorkspace. While for MatrixWorkspace there are two dimensions describing a data point (spectrum number and independent variable), for MDWorkspaces we have between 1 and 9 dimensions. Higher number of dimensions are required to accommodate labelling of data with extended parameter dependencies (e.g. sample environment variables).

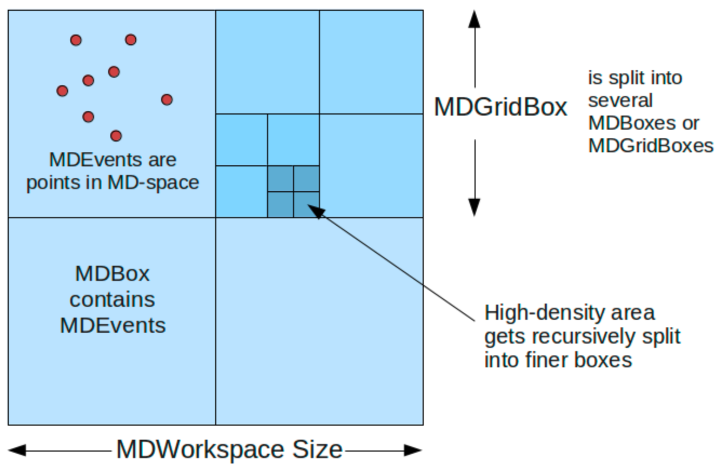

Figure 2: Schematic representation of the principle of adaptive rebinning used in the MDEventWorkspace type.

For MDEventWorkspaces, each MDEvent contains coordinates, a weight and an uncertainty. It might also contain information about which detector and which run it comes 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 (see Fig. 2). MDHistoWorkspaces consist of signal and error arrays on a regular grid.

For data formats that contain different field types, Mantid provides TableWorkspaces. A TableWorkspace is organised in columns with each column having a name and type. Examples of TableWorkspaces are the parameters from model fitting, and a representation of information about Bragg peaks.

5.4. Algorithms

The algorithm layer is a key aspect of the Mantid Framework. Mantid algorithms are procedures to manipulate workspaces. They can be predefined, or written by users, in either C++ or Python. The organization and development of algorithms is key to maintaining the ethos of the project. This presents a number of challenges for development as the Framework can access multiple data types, from a variety of instruments. At the present, there are over 500 algorithms covering data handling (loading/saving workspaces from/to files), arithmetic operations (plus, minus, multiply, divide), unit conversions, and many technique specific algorithms (powder diffraction, single crystal diffraction, SANS, reflectometry, direct and indirect spectrometry, and μSR).

The case of event mode data is interesting as it presents an efficient way of processing sparse data. It is often more efficient to keep the data as events through a chain of operations. This requirement has resulted in the development of a number of specialized event data handling operations. The end result is that for many reduction chains the data is events type until the final presentation.

Core algorithms can be grouped together to form 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.

In some cases a single "workflow 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 3 shows the generic workflow to process and view data as it is collected.

5.5. Python API and scripting

The Python API provides an exceptionally powerful interface to Mantid. Many classes within the Framework are open to Python control. The algorithms are added to the API at runtime, allowing new plugin algorithms to be available without further configuration. The Python API can be used to simply interact with existing functionality. Furthermore, Python can also be used to extend the capabilities of the Mantid Framework adding further algorithms or fit functions without needing to recompile or even restart the program.

The API has been written to give an intuitive Python feel, allowing a simple powerful syntax with minimal specific understanding of Mantid. More advanced usage is possible within Python scripts allowing popular packages such as NumPy, SciPy, matplotlib [13, 14, 15] to be mixed with Mantid Algorithms to process data.

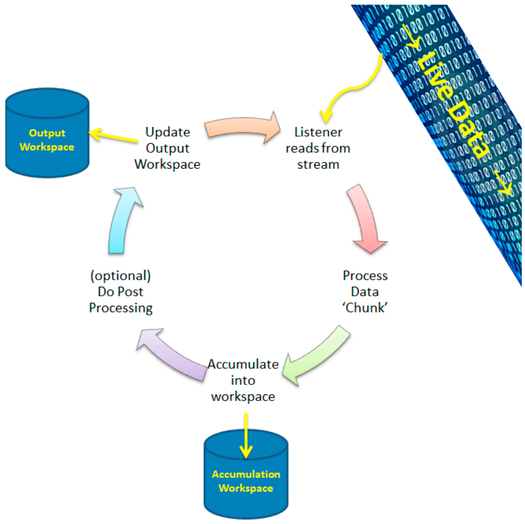

Figure 3: Flow diagram representing data flow when using live data processing within Mantid

6. User Interface

6.1. MantidPlot

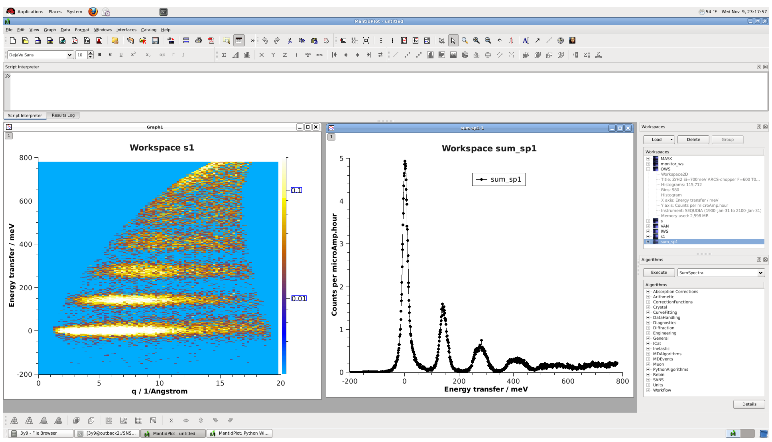

Figure 4: MantidPlot interface, showing 1D, and 2D plots. Lists of workspaces and algorithms are available on the right side

The main interaction with Mantid occurs through the MantidPlot interface (Figure 4), based on QtiPlot[16]. It allows visualising and processing the data, Python scripting, and a generic fitting system. A list of all algorithms, organized in categories, is also present. Clicking on an algorithm will open an automatically generated dialog box, with entries for each of the input parameters. A validation occurs when information is filled and any invalid input is flagged with an error message for the user. For each algorithm dialog box, a button allows for invoking the built-in help. A log window, where users can see the results of running different algorithms, is available. For several scientific techniques, custom interfaces are accessible from the MantidPlot menu.

The Workspaces toolbox shows a list of all the workspaces currently available. Expanding the workspace entries show information about their type and content. A context sensitive menu allows plotting, instrument view, inspection of the sample environment logs, or the history of the workspace.

6.2. Custom interfaces

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 meaningful units. As such, reduction for several scientific techniques can be complicated. 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 Mantid's objectives is to provide scientists 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. Science areas and instruments specifically supported by the Mantid Framework can be seen in Table 1.

Science area	Instruments
Powder neutron diffraction	GEM HRPD WISH POLARIS POWGEN NOMAD VULCAN
Single crystal neutron diffraction	WISH SXD TOPAZ MANDI
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 OFF-SPEC REF_L
μSR	MUSR HIFI EMU

Table 1: Current science areas and instruments supported by the Mantid framework

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 multidimensional datasets. Fitting peak functions and simple user derived functions to line data i.e. data that is in 1D x,y,e format can be also performed using simple user friendly interface. This user interface has been developed to allow complex function to be created from simple mathematical building blocks using GUI. Any function or dataset can also 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 Table Workspace dataset which can then be further manipulated and analysed to estimate statistical characteristics of fitting.

Figure 5 shows the GUI used for fitting mathematical functions to 1D data.

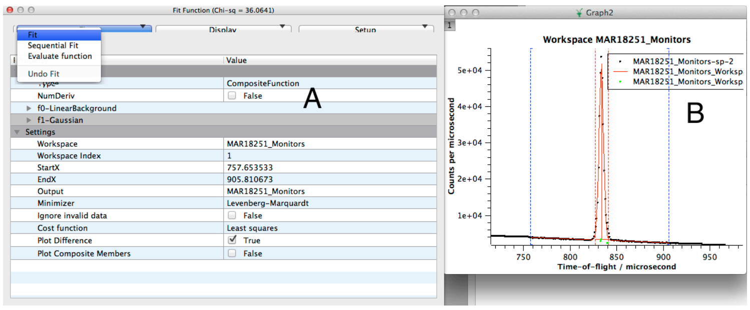

Figure 5: The simple fitting GUI interface in the MantidPlot application. Peak selection is performed using mouse selection on the displayed data. Panel A displays the required model and fit controls panel B displays the data, fitted model and difference the vertical dotted lines indicate extents in x for the model.

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 currently can link directly to $SASView$ for further analysis of experimental data.

6.4. Simulation and Analysis

Fitting over multidimensional datasets is used in more complex situations e.g. in the analysis of the results of the inelastic scattering experiments. Fitting a single resolution broadened model of scattering $S(q, \omega)$ to a n dimensional $S(q, \omega)$ dataset is a standard data analysis procedure in this area used to account for substantial changes in the results of the experiment due to the instrument resolution effects. Mantid contains a set of procedures for calculating an instrument resolution function and convoluting this resolution with chosen scattering model to obtain simulated resolution broadened scattering model. It then can use the multidimensional fitting framework to compare simulated model scattering with experimental scattering and fit the parameters of the scattering model to the results of the experiments. These capabilities are similar to the capabilities available in the past program like *Toby*fit [17].At the moment *Toby*fit's Monte-carlo-based instrument resolution model is implemented in the Mantid and the framework allows to define and deploy other instrument resolution models. Mantid also already contains range of *Toby*fit's scattering models used in the analysis of the inelastic neutron scattering data.

7. Visualization

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 samples such instruments become large Laue geometry cameras. For SANS experiments area detectors of 1x1m are commonplace and place another requirement on visualization. For visualising the instrument data in real space the InstrumentView can be used (figure 6), 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 sliceViewer7 described later or the entire volume can be rendered and manipulated using $paraView$ (PV). A simple instance of PV has been developed ($visulisationtoolkitVATES$) specifically for neutron specific methods and can be instantiated directly from within MantidPlot, from the MDEventworkspace context menu. The MDevent datatype and associated visualization 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 efficient.

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
- Bragg peaks can be selected and saved as peak workspaces
- For instruments that use position sensitive detectors the counts in each pixel can be plotted as a function of position along the tube.

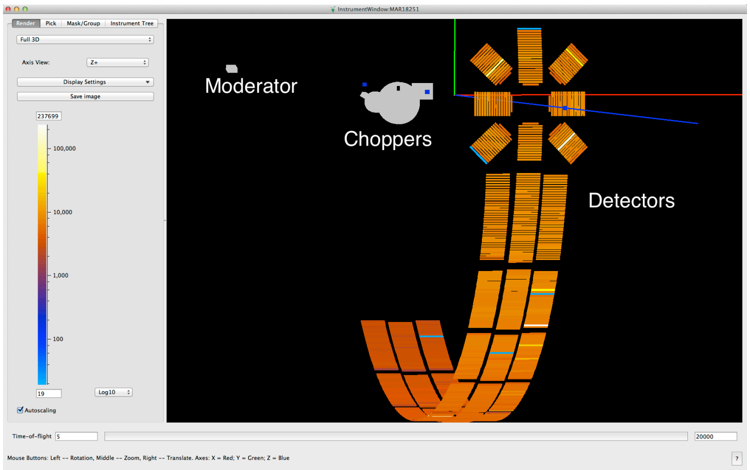

Figure 6: The instrument view GUI in MantidPlot showing a 3D representation of the MARI spectrometer. Various components are annotated.

7.2. Slice viewer

One tool for visualising multi-dimensional (MD) data is the SliceViewer (Figure 7). 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.

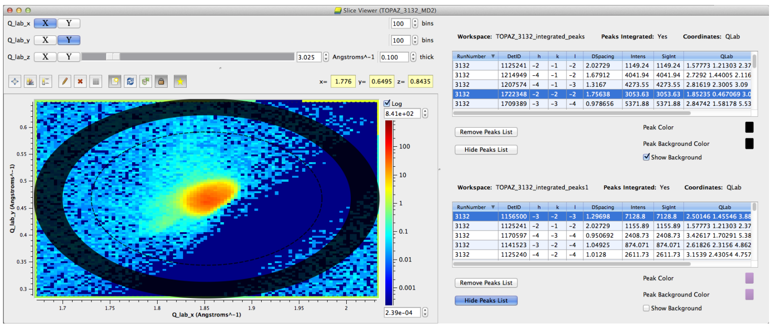

Figure 7: SliceViewer showing a single crystal peak and related information.

7.3. VATES

A major objective of this project has been the ability to represent multidimensional data [3, 18, 19]. 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.

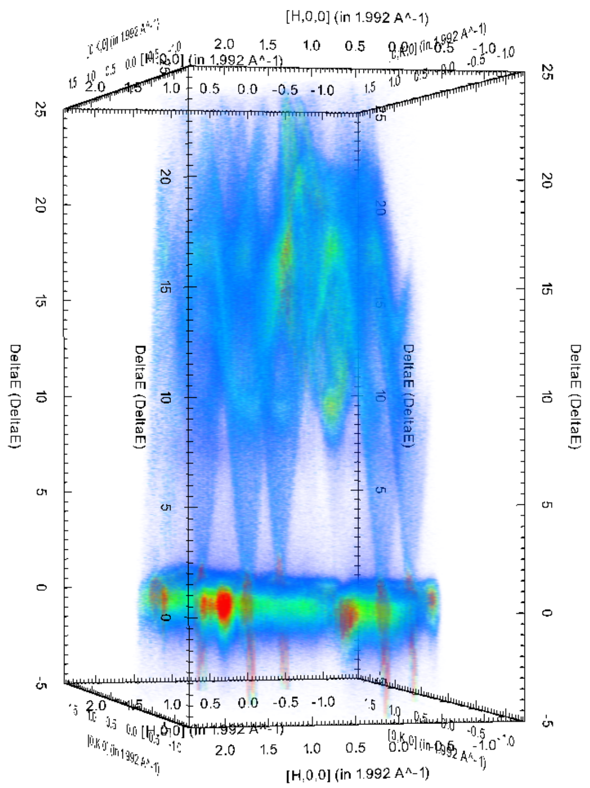

Figure 8: Volume rendering of Gd excitations, in ParaView. Data was measured on SEQUOIA spectrometer at SNS.

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[1] visualization 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[20] 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 9) 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-

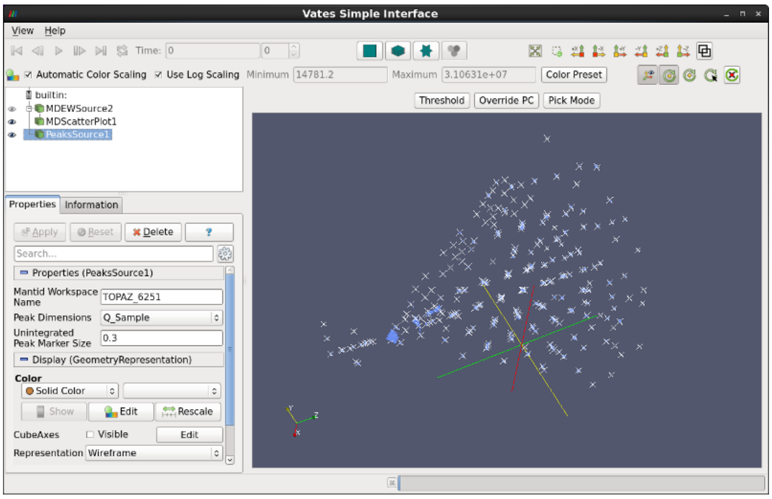

Figure 9: VSI in splatter plot mode with single crystal data from TOPAZ difractometer.

orthogonal axes such as the data in Figure 8. This capability was implemented by Kitware[21] via the SNS in support of the Mantid project.

Ongoing work on the VATES interface aims to provide full scriptable control over all the visualization 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 applications. User generated Python applications can be submitted to the Mantid script repository. The script repository allows users to contribute and share scripts with the rest of the Mantid community and MantidPlot allows upload and downloading as well as marking scripts to automatically updated with new versions from the repository. The flexible nature of the framework can be used to analyse most types of experimental data, and is often used in new and interesting ways by the community that were not originally envisaged by the development team. With the many algorithms supported by Mantid extensive documentation is required, This is provided at several levels, from helpful validation,"intellisense" style help within the scripting environment, offline help provided with the installation and online help including examples and tutorials. Finally the MantidPlot application allows users to submit bug reports, requests for assistance or just a suggestions for future development directly to the development team.

9. Facility integration

A very important step in Mantid development and deployment is facility integration with systems for locating files within the facilities data archive and accessing the metadata catalog for experiments. To assist in this Mantid interfaces with Information CATalog (ICAT) [22], and each facilities data archive through a plug in interface.

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. Each facility uses a different approach to storing their archived data files, Mantid allows a small archive search adapter to be written to a provided interface 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 in use on certain instruments. As soon as files are created and catalogued, a 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.

In addition at SNS and ISIS, the development team has developed 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. This approach captures either events as they stream off the data acquisition or point in time histograms, and allows for near real time processing of the results into scientifically useful results. This level of near real time data analysis allows for much better planning and use of valuable experiment time.

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[23].

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] A. Henderson, ParaView Guide, A Parallel Visualization Application, Kitware Inc., 2005.
[2] `https://www.gnu.org/licenses/gpl.html`, accessed: 2013-11-07.
[3] R. Azuah, L. Kneller, Y. Qiu, P. Tregenna-Piggott, C. Brown, J. Copley, R. Dimeo, J. Res. Natl. Inst. Stand. Technol. 114 (2009) 341.
[4] S. I. Campbell, F. A. Akeroyd, C. M. Moreton-Smith, Open GENIE - Analysis and Control, eprint arXiv:cond-mat/0210442.
[5] D. Richard, M. Ferrand, G. Kearley, J. Neutron Research 4 (1996) 33–39.
[6] T. Worlton, A. Chatterjee, J. Hammonds, C. Bouzek, D. Mikkelson, R. Mikkelson, M. Miller, B. Serum, P. Peterson, Scientific review: New software for neutron scattering data visualization, Neutron News 15 (3) (2004) 14–15.
[7] `http://dx.doi.org/10.5286/software/mantid`, accessed: 2013-11-07.

[8] http://git-scm.com/, accessed: 2013-11-07.
[9] http://nexusformat.org, accessed: 2013-11-07.
[10] http://openmp.org, accessed: 2013-11-07.
[11] http://openmp.org, accessed: 2013-11-07.
[12] J. Zikovsky, P. F. Peterson, S. I. Campbell, M. A. Reuter, R. J. Taylor, Event-based processing of neutron scattering data (in prep.).
[13] T. E. Oliphant, Python for scientific computing, Computing in Science & Engineering 9 (3) (2007) 10–20.
[14] E. Jones, T. Oliphant, P. Peterson, et al., SciPy: Open source scientific tools for Python (2001–). URL http://www.scipy.org/
[15] J. D. Hunter, Matplotlib: A 2d graphics environment, Computing In Science & Engineering 9 (3) (2007) 90–95.
[16] http://soft.proindependent.com/qtiplot.html, accessed: 2013-11-07.
[17] T. G. Perring, Tobyfit, http://tobyfit.isis.rl.ac.uk, accessed: 2013-11-07.
[18] R. Coldea, Mslice, http://mslice.isis.rl.ac.uk, accessed: 2013-11-07.
[19] T. G. Perring, Horace, http://horace.isis.rl.ac.uk, accessed: 2013-11-07.
[20] W. Schroeder, K. Martin, B. Lorensen, The visualization toolkit: an object-oriented approach to 3d graphics [visualize data in 3d - medical, engineering or scientific; build your own applications with c++, tcl, java or python; includes source code for vtk (supports unix, windows and mac)].
[21] http://www.kitware.com, accessed: 2013-11-07.
[22] http://www.icatproject.org/, accessed: 2013-11-07.
[23] http://www.mantidproject.org/, accessed: 2013-11-07.