

Mantid - Data Analysis and Visualization Package for Neutron Scattering and μSR Experiments

O. Arnold^{a,b}, J. C. Bilheux^c, J. M. Borreguero^c, A. Buts^a, S. I. Campbell^c,
L. Chapon^{a,d}, M. Doucet^c, N. Draper^{a,b}, R. Ferraz Leal^d, M. A. Gigg^{a,b}, V. E. Lynch^c,
A. Markvardsen^a, D. J. Mikkelsen^{e,c}, R. L. Mikkelsen^{e,c}, R. Miller^f, K. Palmen^a,
P. Parker^a, G. Passos^a, T. G. Perring^a, P. F. Peterson^c, S. Ren^c, M. A. Reuter^c,
A. T. Savici^c, J. W. Taylor^a, R. J. Taylor^{c,g}, R. Tolchenov^{a,b}, W. Zhou^c, J. Zikovsky^c

^aISIS Facility, Rutherford Appleton Laboratory, Chilton, Didcot, Oxfordshire, UK

^bTessella Ltd., Abingdon, Oxfordshire, UK

^cNeutron Data Analysis and Visualization, Oak Ridge National Laboratory, Oak Ridge, TN, USA

^dInstitut Laue-Langevin, Grenoble, France

^eUniversity of Wisconsin-Stout, Menomonie, WI, USA

^fComputing and Computational Science Directorate, Oak Ridge National Laboratory,
Oak Ridge, TN, USA

^gTessella Inc., Newton, MA, USA

Abstract

The Mantid framework is a software solution developed for the 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 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 fields 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, and ISIS at Rutherford Appleton Laboratory, have ever increasing user demand and produce ever increasing volumes of data. One of the 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

For general Mantid correspondence use mantid-help@mantidproject.org

Email address: saviciat@ornl.gov (A. T. Savici)

Preprint submitted to Elsevier

March 3, 2014

Data[1] (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[2]. 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[3], 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[4, 5, 6, 7]. 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 programs 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[1] 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 $S(\mathbf{q}, \omega)$ in a single pulse. In a neutron experiment one must relate measured counts to the physically meaningful $S(\mathbf{q}, \omega)$.

At a modern TOF neutron source it is common for instruments to have $10^5 n\text{ cm}^{-1}\text{s}^{-1}$ 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, and 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 analyse 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.

114 4. Development Practices

115 One of the key aspects of Mantid is the manner in which it is developed. In order
116 to achieve the stated goals, a large team of scientists and scientific software engineers in
117 Europe and United States are collaborating on this project. For an effective collaboration,
118 we use several software development tools and practices designed to support distributed
119 development teams. New feature requests or defect reports are entered into an issue
120 tracking system.

121 Another tool vital for organising work is the use of a version control system. Man-
122 tid uses git [8] repositories for the source code, configuration files, and much of the
123 documentation. To allow multiple developers to work in similar areas without interfer-
124 ence, developers work on separate branches for each feature. To verify that there are
125 no cross-platform compatibility issues, each feature branch is merged onto a ‘develop’
126 branch whenever new code is ready. It is only after a feature branch has been completely
127 addressed and tested that the code changes are merged onto the ‘master’ branch from
128 which release builds and new features are based.

129 In order to ensure quality, the Mantid project uses continuous integration. When-
130 ever new code is committed to the ‘develop’ branch, builds for each supported operating
131 system are started, and are tested against a suite of over 6000 automated unit tests. A
132 build is marked successful only if all of these unit tests pass. Once a day, a series of
133 over 150 integration ‘system tests’ are run with the most recent locally installed version.
134 Builds that pass all system tests are immediately available for download and, in some
135 cases, automatically deployed to computers. Formal releases of Mantid occur approxi-
136 mately every three months, and undergo additional manual testing. These releases are
137 accompanied by detailed release notes and training.

138 5. Mantid Design

139 One of the main design consideration for this project was the separation of data and
140 algorithms. The ethos of the development is that algorithms should (where possible)
141 operate on all data types without *a priori* knowledge of the data or the experiment that
142 generated it. In principle, this ideology makes the framework cleaner and easier to use. In
143 many instances, scientists are not experts in neutron scattering, μSR , or the associated
144 data analysis that is required. Successful software application written for scientists must
145 take this into account at the design stage.

146 Data containers (called workspaces) and algorithms, which manipulate workspaces,
147 compose the central element of the Mantid framework (Figure 1). Workspaces and
148 algorithms are aware of the geometry of each individual instrument. Workspaces can be
149 loaded from various file formats, live data streams, or created by different algorithms.
150 The workspaces can be manipulated by the many algorithms in Mantid, and saved in a
151 variety of formats. By default, Mantid uses the NeXus format[9] for saving intermediate
152 and processed data, but various other output formats are also supported.

153 To ensure high performance for data analysis, but also allow flexibility in how the data
154 is processed, the project is written in C++ with Python bindings. For parallel processing,
155 Mantid uses OpenMP[10], Posix threads and MPI[11]. The interaction with the Mantid
156 framework occurs through the application programming interface (API). Currently the
157 main interactions occur through either Python or through the graphical interface.

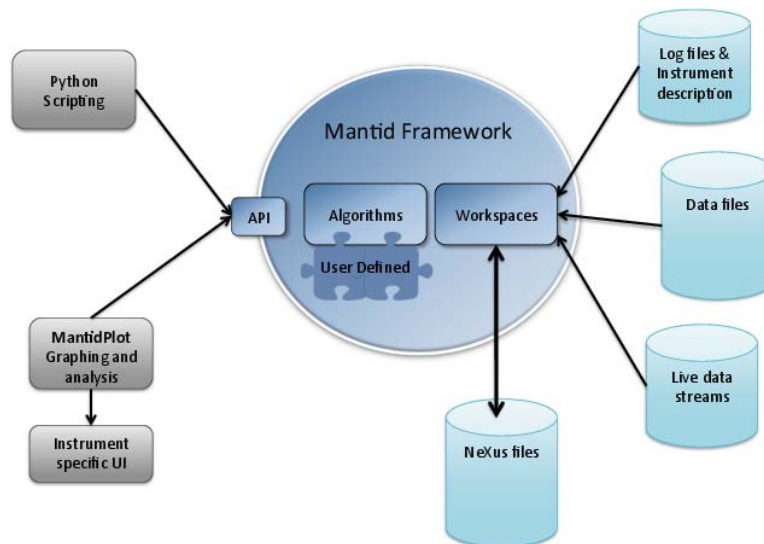


Figure 1: Mantid framework design

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 in Figure 2, temperature scans in Figure 3), and pump probe experiments (pulse magnets, high frequency deformations of materials, and so on).

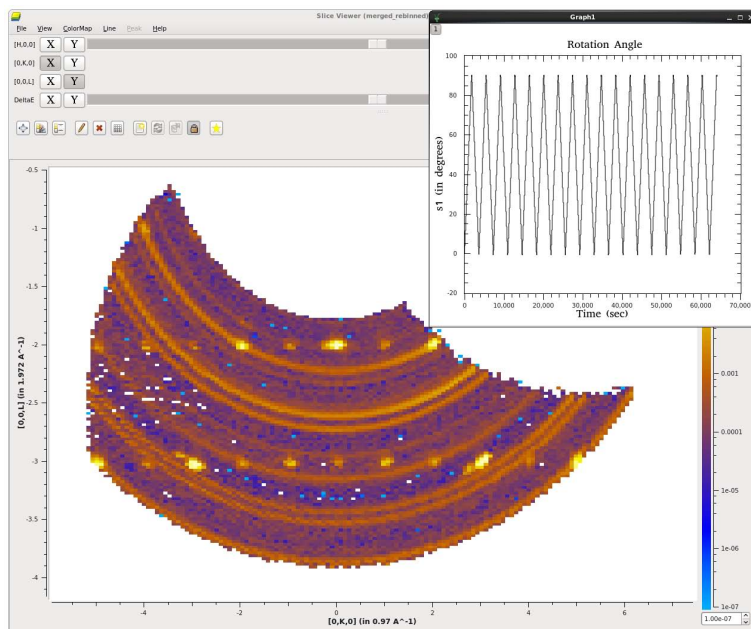


Figure 2: Continuous angle rotation example. Event data taken on HYSPEC spectrometer at SNS is filtered by angle, then converted to HKL momentum transfer components.

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).

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

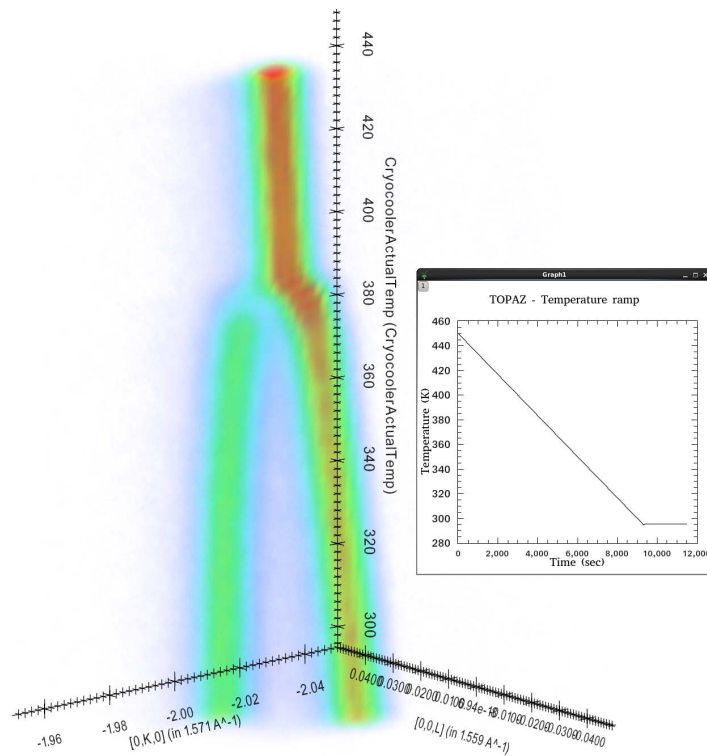


Figure 3: Example of event workspace usage for temperature ramp. Data taken on TOPAZ single crystal diffractometer shows a phase transition symmetry splitting on one of the Bragg peaks.

an efficient searching and binning, and allows plotting on an adaptive mesh (see Fig. 4). 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

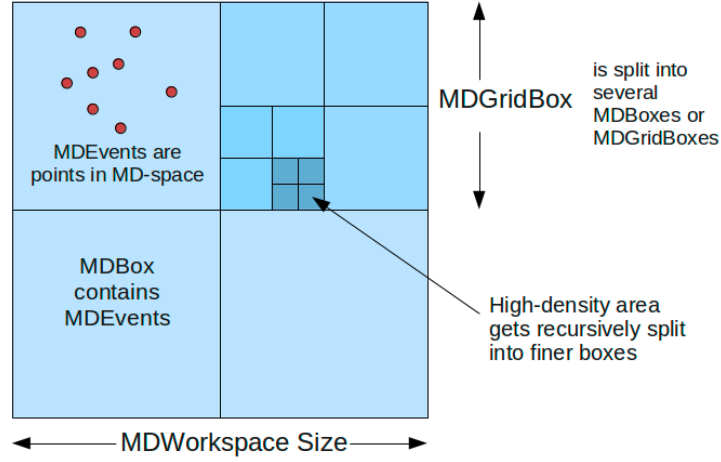


Figure 4: Schematic representation of the principle of adaptive rebinning used in the MEventWorkspace type.

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, such as for live event process. The application can access the live data streams of event mode instruments at SNS and ISIS and can directly read histogram data from the detector electronics of ISIS instruments.

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 by 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.

246 6. User Interface

247 6.1. MantidPlot

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

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

261 6.2. Custom Interfaces

262 Data reduction and basic analysis for individual instruments or science areas is gener-
263 ally a sequential chain of operations starting from data loading and resulting in a dataset
264 that has meaningful units. As such, reduction for several scientific techniques can be
265 complicated. More often than not, development of new features in this area must take
266 into account legacy usage requirements and be well validated against existing "known"
267 good results. In all cases, development of data reduction chains are tightly controlled
268 and validated.

269 One of Mantid's objectives is to provide scientists with a simple and efficient interface
270 to allow them to analyse their data. To achieve this for multiple science areas and
271 instruments, a number of custom interfaces have been implemented. Science areas and
272 instruments specifically supported by the Mantid framework can be seen in Table 1.

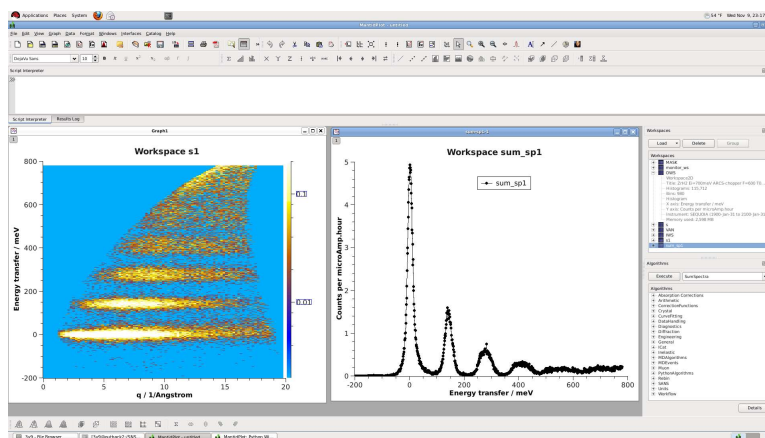


Figure 5: MantidPlot interface, showing 1D, and 2D plots. Lists of workspaces and algorithms are available on the right side

| 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 reflectometry | CRISP SURF POLREF INTER OFFSPEC REF_L |
| μ SR | MUSR HIFI EMU |

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

6.3. Fitting

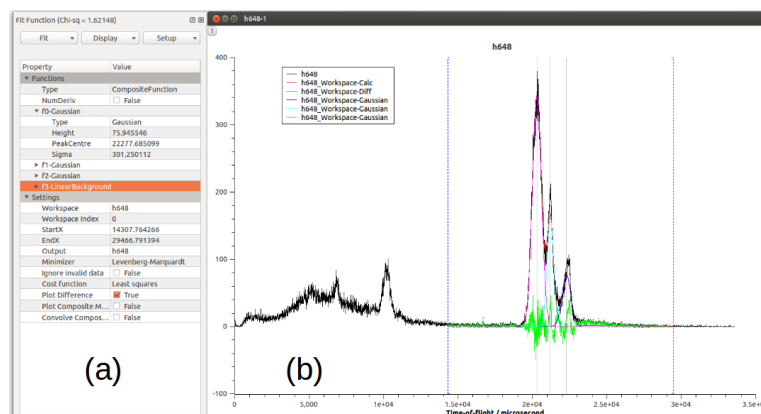


Figure 6: The simple fitting GUI interface in the MantidPlot application. Peak selection is performed using mouse selection on the displayed data. Panel (a) displays the required model and fit controls. Panel (b) displays the data, fitted model and difference the vertical dotted lines indicate extents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

Fitting mathematical functions and models to experimental data is a key requirement of any scientific computing application. The Mantid framework has implemented a powerful engine for fitting multidimensional datasets. This is not intended to replace domain specific fitting software (e.g. GSAS, SASView). Fitting peak functions and simple user derived functions to line data, i.e. data that is in 1D x,y,e format, can also be performed using a simple user interface (Figure 6). All of the fitting can be done within the Python scripting 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. Fitting results are displayed as TableWorkspaces which can then be further manipulated and analysed.

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

288 broadened model of scattering $S(\mathbf{q}, \omega)$ to a n dimensional $S(\mathbf{q}, \omega)$ dataset is a standard
289 data analysis procedure in this area used to account for substantial changes in the results
290 of the experiment due to the instrument resolution effects.

291 Mantid contains a set of procedures for calculating an instrument resolution func-
292 tion and convoluting this resolution with chosen scattering model to obtain simulated
293 resolution broadened scattering model. It then can use the multidimensional fitting
294 framework to compare simulated model scattering with experimental scattering and fit
295 the parameters of the scattering model to the results of the experiments. These ca-
296 pabilities are similar to the capabilities available in legacy programs (e.g. TobyFit [17],
297 DAVE [4]). A Monte Carlo based instrument resolution model is implemented in Mantid.
298 The framework allows to define and deploy other instrument resolution models. Mantid
299 also contains a range of scattering models used in the analysis of the inelastic neutron
300 scattering data.

301 7. Visualization

302 Modern instruments survey broad regions of reciprocal space, and therefore generate
303 large data sets which cannot be easily visualised in 1D or 2D projections. Mantid provides
304 a variety of tools for visualising higher dimensional data.

305 7.1. Instrument View

306 The Instrument View (Figure 7) is a 3D representation of the whole instrument, with
307 component positions calculated from the IDF. Non-detector components (e.g. choppers,
308 guides) can be toggled on or off. The colour of the detectors is representative of the total
309 integrated counts. In addition to the 3D rendering, various 2D projections (e.g. spherical
310 along x , cylindrical along y) of the detectors are available. The Instrument View allows
311 for quick access to information about detectors, and provides a simple graphical interface
312 for masking, grouping, and viewing spectra.

313 7.2. Slice Viewer

314 One tool for visualising multidimensional (MD) data is the Slice Viewer (Figure 8).
315 The Slice Viewer provides an interactive 2D projection of multiple data types. Advanced
316 features provide interactive line integration and overplotting PeaksWorkspaces. A list of
317 overplotted peaks is available in this view.

318 7.3. VATES Simple Interface

319 A major objective of Mantid has been the ability to represent multidimensional data
320 [4, 18, 19]. Originally the Visualization and Analysis Toolkit Extensions (VATES) project
321 was an add-on to Mantid that is now fully integrated into the project. The VATES
322 Simple Interface (VSI), offers a limited set of data views and access to a subset of
323 Mantid algorithms. It is based on application widgets and rendering libraries from the
324 ParaView[2] visualization program. The VSI takes advantage of the ParaView plugin
325 architecture to provide functionality from within Mantid and from ParaView outside of
326 Mantid.

327 The data to be visualised passes through an API layer which translates the inter-
328 nal Mantid data structure to a VTK[20] data structure, that can be rendered in the

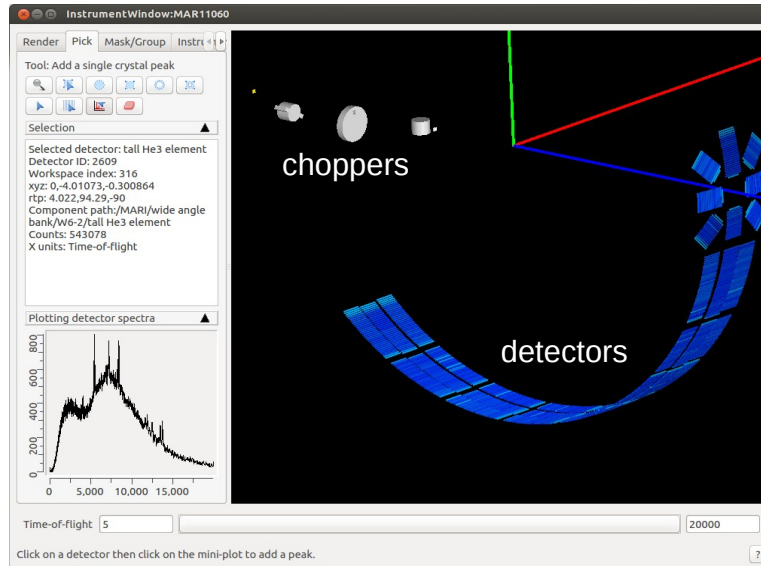


Figure 7: The Instrument View showing a 3D representation of the MARI spectrometer. Various components are annotated.

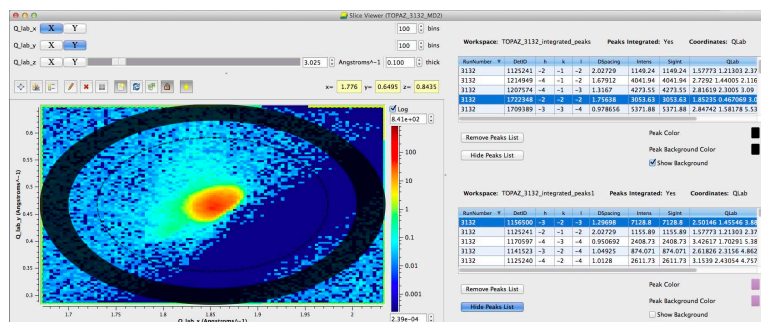


Figure 8: Slice Viewer showing a single crystal peak and related information.

329 *VSI*. Those same data structures can be saved to a file and visualised in the ParaView
 330 application. Indeed, it is possible to drive some aspects of multidimensional analysis di-
 331 rectly from ParaView (Figure 9 and Supplementary material. Data is from reference[21]).
 332 The API layer provides the desired decoupling of the data structures and provides good
 333 flexibility to handle the various needs of the Mantid data structures and algorithms.

334 The *VSI* has a view called Multi Slice which allows placing multiple orthogonal slices
 335 on the data. Those slices can then alternately be viewed in Slice Viewer for further
 336 exploration. The Splatter Plot (Figure 10) view is oriented towards visualising peaks
 337 in single crystal diffraction data. In this view the user can interact with the data to
 338 retrieve information about a selected peak. The Three Slice view shows three orthogonal
 339 planes through the data with the capability exploring via moving a crosshair in one of
 340 the planes with a coordinate readout in each plane to show the location. The *VSI* has

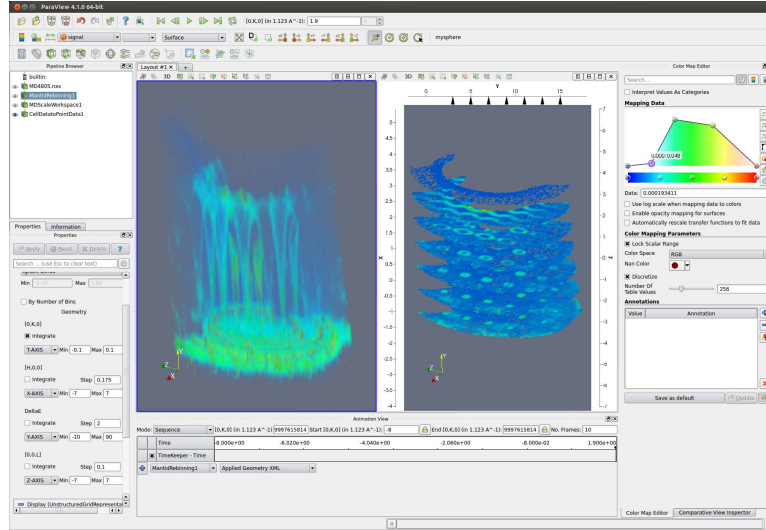


Figure 9: Paraview showing single crystal data[21] on YFeO_3 from the SEQUOIA spectrometer at the SNS.

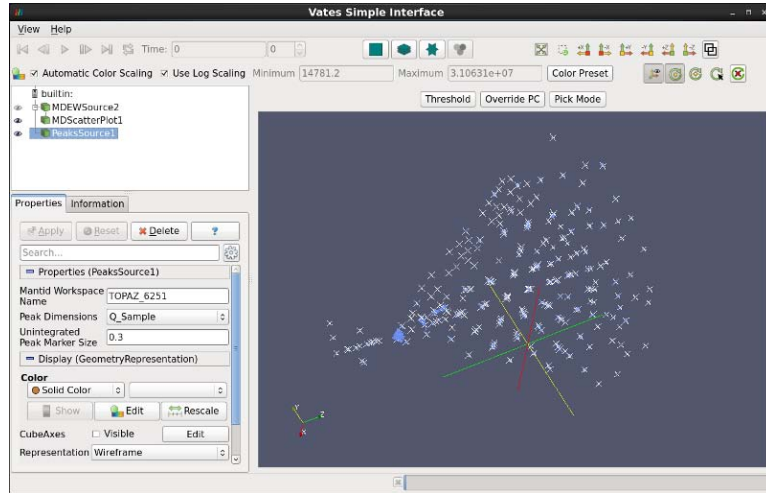


Figure 10: VSI in Splatter Plot mode with single crystal data from the TOPAZ diffractometer at the SNS.

341 the ability to show the data with non-orthogonal axes such as the diffraction pattern for
 342 triclinic materials[22] in Figure 11. This capability was implemented by Kitware[23] via
 343 the SNS in support of the Mantid project.

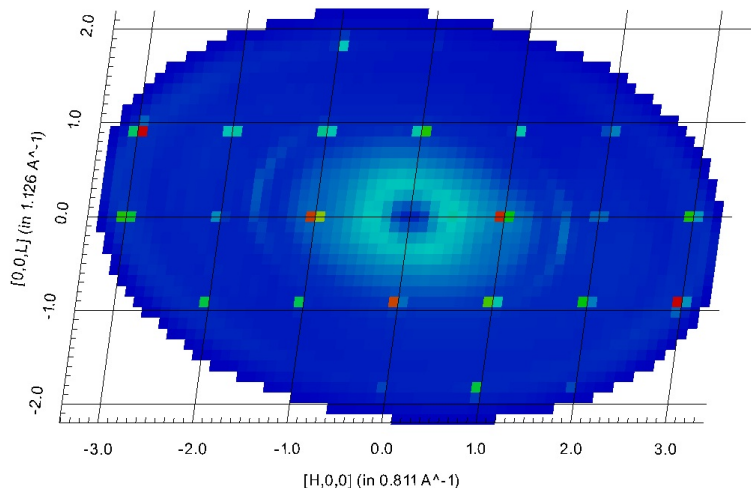


Figure 11: Diffraction pattern from a triclinic lattice, showing non-orthogonal axes. Data was measured on the CNCS spectrometer at the SNS.

344 8. Community Involvement and Expandability

345 The Mantid framework provides facility users with a very powerful data analysis
 346 tool. The Python API gives the user the ability to expand functionality for many differ-
 347 ent applications. User generated Python applications can be submitted to the Mantid
 348 script repository. The script repository allows users to contribute and share scripts with
 349 the rest of the Mantid community and MantidPlot allows upload and downloading as
 350 well as marking scripts to automatically updated with new versions from the repository.
 351 The flexible nature of the framework can be used to analyse most types of experimental
 352 data, and is often used in new and interesting ways by the community that were not
 353 originally envisaged by the development team. With the many algorithms supported by
 354 Mantid extensive documentation is required, This is provided at several levels, from help-
 355 ful validation, intelligent code completion within the scripting environment, offline help
 356 provided with the installation and online help including examples and tutorials. Finally
 357 the MantidPlot application allows users to submit bug reports, requests for assistance or
 358 just a suggestions for future development directly to the development team.

359 9. Facility Integration

360 A very important step in Mantid development and deployment is facility integration.
 361 To assist in this, Mantid interfaces with Information CATalog (ICAT) [24]. It is in use
 362 at both ISIS and SNS. Each facility uses a different approach to storing their archived
 363 data files. Mantid allows a small archive search adapter to be written to a provided
 364 interface to locate raw or processed files in data archives at each of the facilities.

365 One important use of the ICAT interface is the autoreduction process on certain
 366 instruments. As soon as files are created and catalogued, a reduction script is automati-
 367 cally invoked. This script uses metadata in the file and/or the ICAT catalogue to reduce
 368 the raw data to a form that users are interested in.

In addition at SNS and ISIS, the development team has implemented an interface between Mantid and the data acquisition systems, in order to allow users to look and analyse their data in near real time. This approach allows for processing of the live data into scientifically useful results. This level of near real time data analysis allows for much more efficient 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 for widespread adoption at several other neutron facilities. Up to date information and usage tutorials can be found on the Mantid web page[25].

11. Acknowledgements

The development team would like to thank all instrument scientists and students and ISIS and SNS for their feedback and contributions, P. Radaelli, and R. McGreevy for championing the project at ISIS in the initial stages, and R. McGreevy, I. Anderson, and M. Hagen for forging the collaboration between ORNL and STFC. We acknowledge A. Hillier for contributions to this manuscript. 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] <http://dx.doi.org/10.5286/software/mantid>, accessed: 2013-11-07.
 - [2] A. Henderson, ParaView Guide, A Parallel Visualization Application, Kitware Inc., 2005.
 - [3] <https://www.gnu.org/licenses/gpl.html>, accessed: 2013-11-07.
 - [4] R. Azuah, L. Kneller, Y. Qiu, P. Tregenna-Piggott, C. Brown, J. Copley, R. Dimeo, Dave: A comprehensive software suite for the reduction, visualization, and analysis of low energy neutron spectroscopic data, J. Res. Natl. Inst. Stand. Technol. 114 (2009) 341.
 - [5] S. I. Campbell, F. A. Akeroyd, C. M. Moreton-Smith, Open GENIE - Analysis and Control, eprint arXiv:cond-mat/0210442.
 - [6] D. Richard, M. Ferrand, G. Kearley, Analysis and visualisation of neutron-scattering data, J. Neutron Research 4 (1996) 33–39.
 - [7] T. Worlton, A. Chatterjee, J. Hammonds, C. Bouzek, D. Mikkelsen, R. Mikkelsen, M. Miller, B. Serum, P. Peterson, Scientific review: New software for neutron scattering data visualization, Neutron News 15 (3) (2004) 14–15.
 - [8] <http://git-scm.com/>, accessed: 2013-11-07.
 - [9] M. Könncke, The state of the nexus data format, Physica B: Condensed Matter 385 - 386, Part 2 (2006) 1343 – 1345.
 - [10] <http://openmp.org>, accessed: 2013-11-07.
 - [11] <http://www.mpi-forum.org/>, accessed: 2013-12-07.
 - [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/>

- 414 [15] J. D. Hunter, Matplotlib: A 2d graphics environment, Computing In Science & Engineering 9 (3)
415 (2007) 90–95.
- 416 [16] <http://soft.proindependent.com/qtiplot.html>, accessed: 2013-11-07.
- 417 [17] T. G. Perring, Tobyfit, <http://tobyfit.isis.rl.ac.uk>, accessed: 2013-11-07.
- 418 [18] R. Coldea, Mslice, <http://mslice.isis.rl.ac.uk>, accessed: 2013-11-07.
- 419 [19] T. G. Perring, Horace, <http://horace.isis.rl.ac.uk>, accessed: 2013-11-07.
- 420 [20] W. Schroeder, K. Martin, B. Lorensen, The visualization toolkit: an object-oriented approach to
421 3D graphics, Kitware, 2006.
- 422 [21] S. E. Hahn, A. A. Podlesnyak, G. Ehlers, G. E. Granroth, R. S. Fishman, A. I. Kolesnikov, E. Pom-
423 jakushina, K. Conder, Inelastic neutron scattering studies of $YFeO_3$, Phys. Rev. B 89 (2014)
424 014420.
- 425 [22] D. H vonen, S. Zhao, G. Ehlers, M. M nsson, S. N. Gvasaliya, A. Zheludev, Excitations in a
426 quantum spin liquid with random bonds, Phys. Rev. B 86 (2012) 214408.
- 427 [23] <http://www.kitware.com>, accessed: 2013-11-07.
- 428 [24] <http://www.icatproject.org/>, accessed: 2013-11-07.
- 429 [25] <http://www.mantidproject.org/>, accessed: 2013-11-07.