Dataflow

Web-based flexible data reduction for neutron reflecometry

Brendan Rowan (SHIP, NCNR) – Poolesville High School
Mentor: Dr. Brian Maranville

Background and Importance

Neutron reflectometry refers to the act of firing a beam of neutrons at a flat surface while measuring the intensity of the reflected neutrons. The data collected when using a neutron reflectometer gives the user valuable information about the structure of any thin films on the surface being measured. However, it is difficult to confidently state conclusions about the structure of these films with only raw data, as they are recorded in instrument-specific coordinates. These data are virtually useless unless they are converted into simplified, universal coordinates. Converting instrument-specific data into a more usable form is commonly known as data reduction, and is the main basis for the problem at hand.

The current alternative to performing data reduction by hand is to develop a new program for each instrument. The user of such a program is able to perform reduction techniques on data files, which are then ready for analysis. Although these programs produce correct results, the user must learn a new interface for every instrument he or she uses. In addition, it is difficult to change the transformations that must be applied, which currently requires a change in user interface, a shortcoming in the existing method.

The proposed program, Dataflow, allows the user to make unique reduction recipes, regardless of which instrument is used. By only requiring the user to learn one interface, Dataflow makes it easy for the user to learn new reduction routines for many different types of instruments and experiments. In addition, because the program resides in the user's web browser, a number of advantages are present such as platform and browser independence, version unity, and ease of access.

Caching Intermediate Results

As many data reductions take a large amount of time to compute, it makes sense to cache intermediate results. Therefore, when a user clicks on a wire to view a plot of the data at that point in the reduction, the program can short-circuit and retrieve the desired result if it has already been calculated.

The program that was used to stash results was Redis, a key-value datastore that has mechanisms for storing integers, strings, lists, and maps. Because it is not possible to store references in memory, the operable datatype in the reduction chain must be converted to a string, which is known as serialization. As Python, the programming language used for the backend of Dataflow, has modules for serialization, these modules are only useful for built-in datatypes such as lists and dictionaries. Because the Python serialization modules are worthless in regards to most user-defined classes, methods were written that converted Dataflow-specific datatypes to and from strings.

```
getting result
retrieving cached value: Fingerprint:9935f99ed9d065d05d1cab925ef77b93b72ff295:output
Starting new converter
```

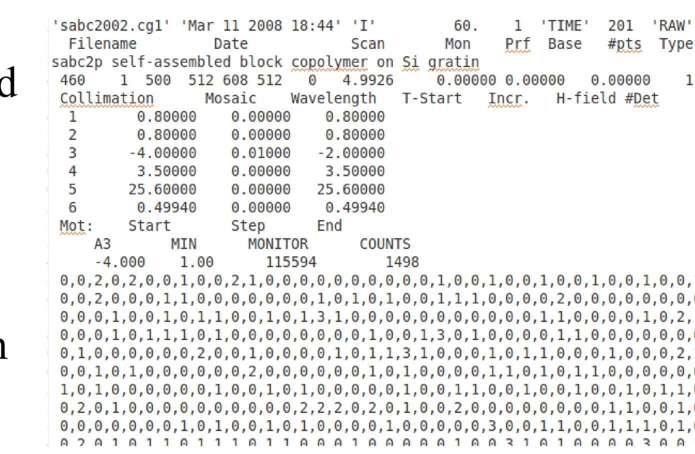
Besides serialization of objects, the program must be able to decide whether a certain calculation has been performed or not. To make this decision, a fingerprint can be created for a certain terminal of a certain node in the reduction diagram. However, certain factors have to be accounted for which include:

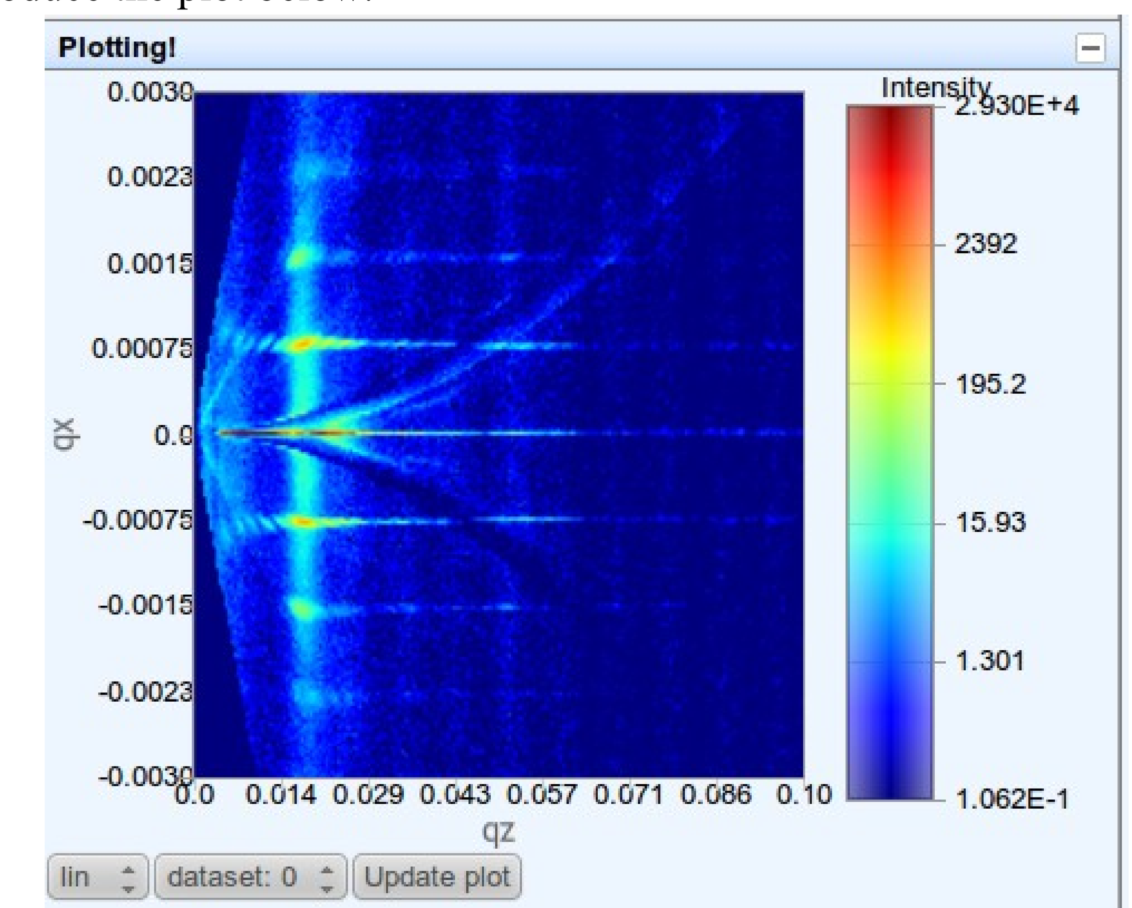
- Fingerprints of ancestors
- All arguments passed to the module
- The output terminal and node number

With this information, a SHA-1 hash, which creates a 40 character string of hex digits, can be created at each step in reduction for maximum efficiency.

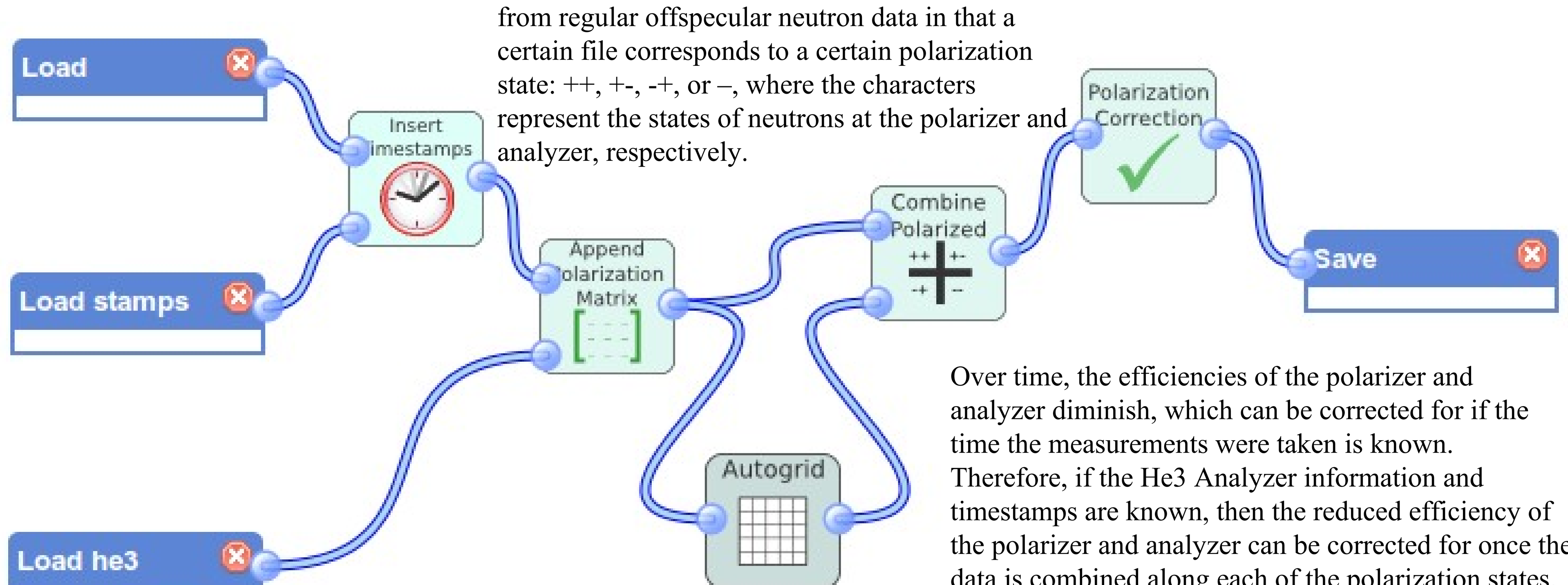
Example Output

On the right, you can see a very small portion of raw data taken from the Advanced Neutron Diffractometer/Refractometer (AND/R) at NIST. Below is a screen shot from Dataflow, which shows the plot of the final step in most offspecular reduction routines: converting to Q-space. The data on the right, along with ten other data files was used to produce the plot below.





Polarized Reduction Template Offspecular Neutron Reflectometry



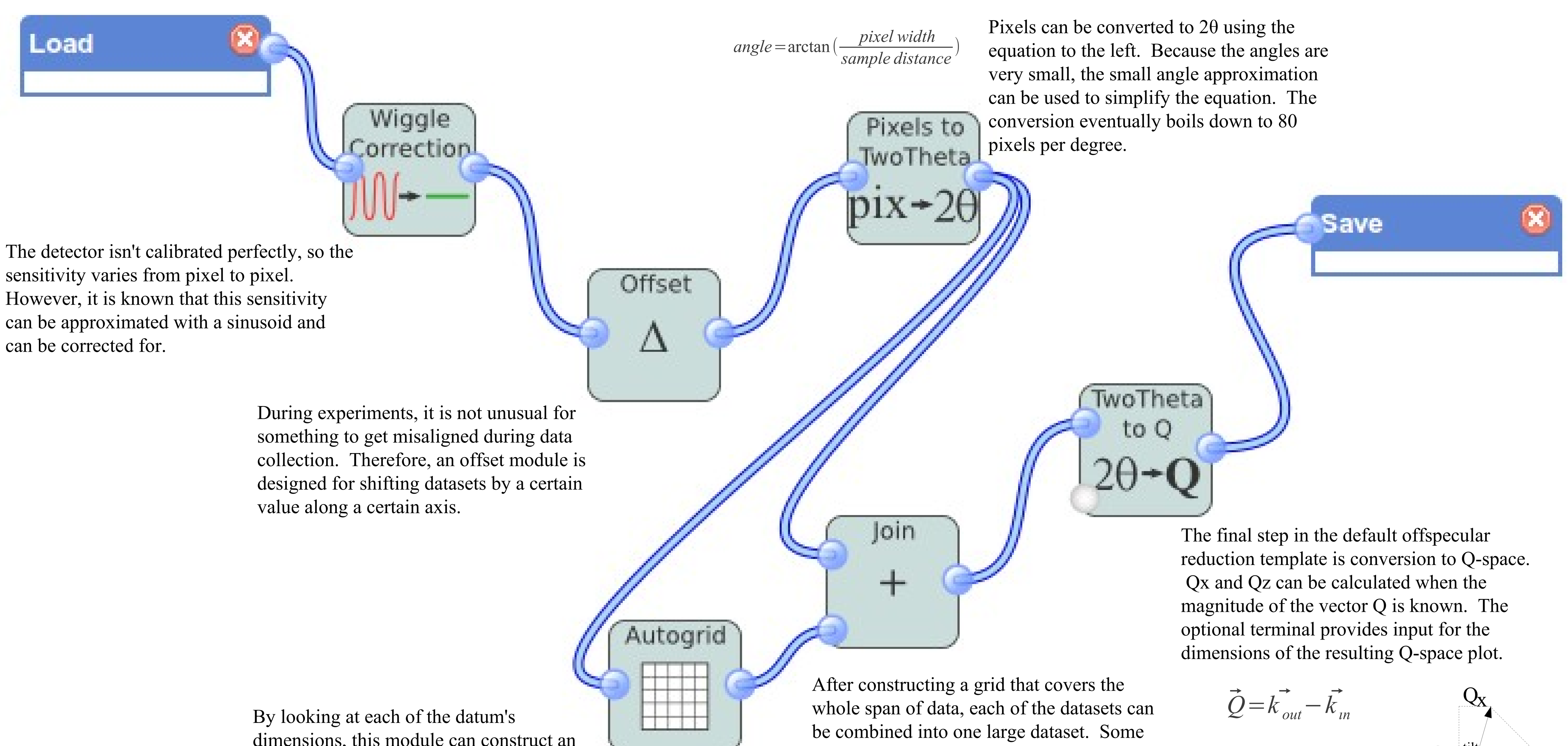
Polarized offspecular neutron data is different from regular offspecular neutron data in that a certain file corresponds to a certain polarization state: ++, +-, -+, or --, where the characters represent the states of neutrons at the polarizer and analyzer, respectively.

Over time, the efficiencies of the polarizer and analyzer diminish, which can be corrected for if the time the measurements were taken is known. Therefore, if the He3 Analyzer information and timestamps are known, then the reduced efficiency of the polarizer and analyzer can be corrected for once the data is combined along each of the polarization states.

Above and to the right are the default templates for polarized and non-polarized offspecular neutron data. As shown, these "templates" are flow diagrams that the user can create at will which link together "modules" that perform a certain reduction. The user is allowed to drag and drop individual modules onto the editor and link them together to perform reductions, although these default templates are provided. Then, the user specifies which data files are needed for certain loaders. Finally, the user can click on a wire to view his or her data at that point in the reduction.

Above is the default template for reducing polarized offspecular neutron data. After the conversions above, the data can be treated like normal offspecular neutron data and be converted to Q-space. Certain reductions, such as polarization correction, which uses complex linear algebra techniques, can take quite a long time due to the calculations that must be performed. Therefore, these reductions really show how useful caching intermediate results is. Without storing the results in a key-value datastore, the user would be unsatisfied with the amount of time needed to view his or her data.

General Reduction Template Offspecular Neutron Reflectometry



$$angle = \arctan\left(\frac{pixel\ width}{sample\ distance}\right)$$

Pixels can be converted to 2θ using the equation to the left. Because the angles are very small, the small angle approximation can be used to simplify the equation. The conversion eventually boils down to 80 pixels per degree.

The detector isn't calibrated perfectly, so the sensitivity varies from pixel to pixel. However, it is known that this sensitivity can be approximated with a sinusoid and can be corrected for.

During experiments, it is not unusual for something to get misaligned during data collection. Therefore, an offset module is designed for shifting datasets by a certain value along a certain axis.

The final step in the default offspecular reduction template is conversion to Q-space. Qx and Qz can be calculated when the magnitude of the vector Q is known. The optional terminal provides input for the dimensions of the resulting Q-space plot.

After constructing a grid that covers the whole span of data, each of the datasets can be combined into one large dataset. Some datasets might overlap, which needs to be handled accordingly.

By looking at each of the datum's dimensions, this module can construct an empty grid that covers the whole span of data supplied. Autogrid is implicitly called by a Join module, so does not necessarily need to be provided by the user.

$$\vec{Q} = \vec{k}_{out} - \vec{k}_{in}$$

$$\vec{Q}_x = |\vec{Q}| * \sin(tilt)$$

$$\vec{Q}_z = |\vec{Q}| * \cos(tilt)$$