Astronomical Data Analysis Software and Systems XVII **P3.1**
ASP Conference Series, Vol. 394, © 2008
R. W. Argyle, P. S. Bunclark, and J. R. Lewis, eds.

A Novel GUI Based Interactive Work Flow Application for Exploratory and Batch Processing of Light Curves

Evan Morikawa, Rahul Dave, Pavlos Protopapas

Initiative in Innovative Computing at Harvard, 60 Oxford St, Cambridge, MA 02138, USA

Abstract. Current methodologies for extracting desired data from light curves involves a complicated process that is often not portable to other projects or other disciplines. To create a more generic environment to easily filter, correct, and clean light-curve data from a myriad of sources; we have developed an XML based work-flow application which allows for a standard, yet scalable and flexible way to handle light curve processing. Wrapped in a Python powered interactive GUI, this application offers astronomers an environment to easily experiment with various light curve filtering techniques. Once a favorable filtering process is discovered, the sequence of filters can be stored, reused, and then applied in batch to large set of raw light curves.

1. Introduction

Harboring the vast quantities of data prevalent in modern astronomy yields numerous technical challenges from the collection source, through the distribution network, and finally to the astronomer responsible for interpreting this data. The Taiwanese American Occultation Survey (TAOS) is an example of a survey program that handles data processing on this scale. They operate four 50 cm telescopes, each equipped with a 2048×2048 pixel CCD camera. On operating nights, they monitor up to 3,000 stars at 5 Hz (Alcock et al. 2004).

In an attempt to abate the challenges of processing large data, this paper highlights a software package titled FilterFlow, which is being developed to expedite the analysis and processing of astronomical time-series data. While the software can technically be used to analyze any type of time-series data, the primary application is to investigate light curve data from large astronomical surveys such as TAOS. The FilterFlow software package allows astronomers to interactively apply various filters to raw signal data in an easy to use graphical user interface (GUI) driven application.

Filterflow grants the ability to instantly plot the results of filters and interact with them in real-time. In addition, FilterFlow uses a drag-and-drop paradigm to place objects onto a canvas. Data is manipulated by filter objects and is then passed to other filter objects along specified connections. This creates a dynamic and non-sequential method of processing data. This filter processing pileline provides FilterFlow with provenance and repeatability at each step.

Once a filter sequence pipeline has been created, a future iteration of FilterFlow will be able to intelligently apply the same sequence to an entire data-set and will be able to efficiently distribute the processes for parallelization.

Figure 1. Mockup of FilterFlow application.

2. Approaching a Solution

Various packages currently exist that are similar in nature to FilterFlow. Matlabs Simulink (Using Simulink 2004) and National Instrumentss LabView are two examples of common commercial software that are designed for canvas driven data processing. GridAssist, Pegasus, and GRID Superscaler are various software packages that have been developed for analyzing, managing, and scaling Grid-based workflows (Linden 2005; Badia 2003; Gil et al. 2004). Taverna is another application very similar to FilterFlow that is used in processing bioinformatics (Oinn et al. 2004). While FilterFlow draws many common features from the previously given applications, it is unique because of its astronomy-specific applications and integrated scalable backbone.

Figure 1 demonstrates a theoretical mockup of what the software can potentially grow into and is used here to demonstrate the core features of FilterFlow. The primary left column on Figure 1 demonstrates the concept of a filter library from which astronomers can pull pre-existing filters or load in their own custom code. The large space at the bottom of the figure demonstrates the canvas from which users can work off of. Each block in that space represents an individual filter. The arrows between them represent the data being piped from one filter to another. Filter parameters are adjustable, and additional filter modules may be inserted anywhere within the filter sequence. Any action taken to a preconnected sequence automatically updates the remainder of that sequence. The two spaces at the top-right of figure 1 represent an area to input parameters to a filter and a space to dynamically display the current state of a light curve as it is being processed throughout the sequence. The software is written in Python using the Tkinter GUI library and is structured around an XML backbone. The XML behind the interface is fundamental to the organization and execution of filter processing.

3. XML FilterFlow

Underneath the GUI of FilterFlow is a basic XML backbone that stores the state of the canvas. The base component of FilterFlow is a Filter Module as represented by figure 2. A filter module represents a function that is applied to time-series data. The filter takes data and input parameters, and then applies the filter to produce a combination of output parameters and time-series data.

Figure 2. (left) Representation of Filter Module. (right) Batching filter sequences.

Multiple filters can be grouped into one meta-module both for organizational purposes and for the consolidation of functions. This feature is represented in XML by tagging attributes of filter modules with the appropriate group. Figure 3 represents the concept of batching filter sequences in XML. Any uninterrupted deterministic sequence of filter modules creates a filter sequence. The same sequence of filters can be applied to multiple inputs. This abstractly expresses FilterFlows ability to take a workflow and batch it across data large dataset.

4. An Example of FilterFlow

To provide a practical example of the FilterFlow software, a scan-distribution algorithm is shown below acting on sample data from TAOS. The TOAS survey is designed to take wide field, high sample rate data to search for the presence of Kuiper Belt objects through stellar occultation. This means we are looking for temporally short dips in the apparent brightness of stellar objects.

Figures 4 and 5 represent two iterations of the filter sequences used to extract data from a TAOS light curve. The light curves are first smoothed using a moving window average. The magnitude of each point is ranked, and then a moving window passes along the ranks and looks for statistically significant dips. The statistics are compared to a randomized distribution of the same procedure to provide a mathematical representation of significant dips or spikes in the data (Protopapas 2006). Because of the abnormal and obviously erroneous spikes in the original data as seen in Figure 4, the solution lies far beyond the normal distribution curve. By inserting an extra 3-sigma cropping filter at the beginning of the workflow, as seen in Figure 5, we eliminate outliers and produce a cleaner signal to work with. As one can see in Figure 5, the solution is vastly improved from Figure 5 where it was obviously placed outside of the bounds of the experiment. This was accomplished simply and effectively in FilterFlow by inserting a filter at the beginning of a sequence. Assuming Figure 5 represents the final sequence; FilterFlow can divide the system up into filter sequences and intelligently batch the process across multiple light curves.

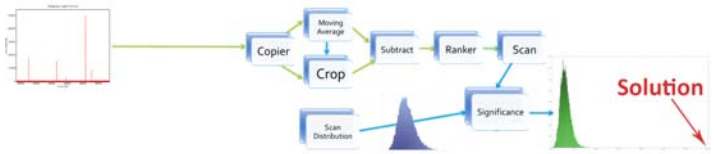

Figure 4. A sample filter Sequence without cropping erroneous data 'spikes'.

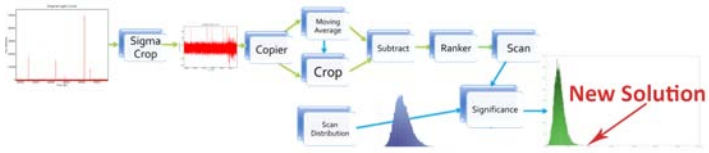

Figure 5. A sample sequence with an extra filter inserted before the entire process.

5. Current Implementation and Future Directions

FilterFlow is still in development and does not possess every feature that was initially intended. The software has the ability to create and group filters on a canvas, process data, write XML, and produce a final output. The larger goal is to create a more powerful orchestration engine that can use the XML backbone to seamlessly combine processes submitted through the GUI, an interactive web-form, or via an API. The engine will then intelligently batch and distribute the workload across multiple machines to significantly ease the process of analyzing extremely large datasets such as those experienced by TAOS.

References

Alcock, C. et al. "TAOS: The TaiwaneseAmerican Occultation Survey" Earth, Moon, and Planets 92.1-4 (2004): 459-464.

Badia, R. M. et al. "Programming Grid Applications with GRID Superscalar." Journal of Grid Computing 1.2 (2003): 151-170.

Gil, Y. et al. "Artificial Intelligence and Grids: Workflow Planning and Beyond." IEEE Intelligent Systems February 2004: 26-33.

ter Linden, M., de Wolf,H & Grim, R. "GridAssist, a User Friendly Grid-based Workflow Management Tool" Proceedings of the 2005 International Conference on Parallel Processing Workshops (2005).

Oinn, T. et al. "Taverna: a tool for the composition and enactment of bioinformatics workflows." Bioinformatics 20.17 (2004): 30453054.

Protopapas, P., Giammarco, J. M., Faccioli, L., Struble, M. F., Dave, R. & Alcock, C. "Finding outlier light-curves in catalogs of periodic variable stars." Monthly Notices of the Royal Astronomical Society (2006).

Using Simulink. Natick, MA: The MathWorks, 2004.