Astronomical Data Analysis Software and Systems: XXIV
ASP Conference Series, Vol. 495
A. R. Taylor and E. Rosolowsky, eds.
©2015 Astronomical Society of the Pacific

External Use of TOPCAT's Plotting Library

M. B. Taylor

H. H. Wills Physics Laboratory, University of Bristol, U.K.

Abstract. The table analysis application TOPCAT uses a custom Java plotting library for highly configurable high-performance interactive or exported visualisations in two and three dimensions. We present here a variety of ways for end users or application developers to make use of this library outside of the TOPCAT application: via the command-line suite STILTS or its Jython variant JyStilts, via a traditional Java API, or by programmatically assigning values to a set of parameters in java code or using some form of inter-process communication. The library has been built with large datasets in mind; interactive plots scale well up to several million points, and static output to standard graphics formats is possible for unlimited sized input data.

1. Introduction

TOPCAT[1] (Taylor 2005) is a desktop GUI application for analysis of tabular data, particularly source catalogues. Among other capabilities it provides high-performance interactive visualisation for large (and small) datasets. The plotting capabilities are focused on representations of point clouds in two or three dimensions, with special attention to large (many row) and high-dimensional (many column) datasets. Many configuration options are offered.

The visualisation is supported by a custom Java plotting library, written from scratch for TOPCAT v4.0 (Taylor 2014). The recent release (v3.0) of the STILTS[2] command-line suite exposes all of TOPCAT's visualisation capabilities in ways that can be harnessed from outside of the application itself.

2. Features

Each plot is composed of a *plot surface*, defining the geometry and decoration of the axes and zero or more *plot layers*. Different layers may use different data sets, allowing all kinds of overplotting. Currently defined plot surface types are 2d Cartesian, Celestial (offering a number of projections), 3d Cartesian, Spherical polar and 2d with a time axis. Many layer types are available including scatter plot, lines, contours, analytic functions, error bars, ellipses, pair links, text labels, vectors, sized markers, histogram, spectrogram, color coding by density or additional coordinates, and more. Each surface

[1] `http://www.starlink.ac.uk/topcat/`

[2] `http://www.starlink.ac.uk/stilts/`

and layer type offers a large number of configuration options. The structure of the package allows easy extensibility, so additional surface types, layer types and configuration options are expected in future releases or can be plugged in at runtime.

The library has been written with performance and scalability as a primary aim. Interactive navigation typically works well up to a few million points. When generating static output files, datasets of unlimited size can be used without large memory requirements. An all-sky density plot of 2 billion rows takes of the order of 30 minutes to produce. Moreover options such as hybrid scatter/density plots are available to provide visually meaningful graphics from very large datasets.

3. Implementation

The STILTS package (Taylor 2006) is a suite of command-line tools built on the same libraries as TOPCAT, aiming to provide access to the same capabilities. While providing alternative interfaces to the same data access, analysis, cross-matching, and I/O facilities is a straightforward enough goal, in fact a large proportion of the complexity of both the command-line and GUI tools is devoted to the user interface itself, so providing the same functions from both contexts is not trivial.

The problem is particularly challenging in the case of the new visualisation tasks. Although of the order of a hundred plotting parameters and configuration options are available to allow the construction of complex and finely tuned plots, a simple plot with default settings can be specified with a very small number of parameters (~ 4). Managing this complexity in both the implementation and the user interface requires careful design and a significant amount of logic. Some notable problems include defining command-line-friendly (i.e. string) representations for all parameter names and values, providing both interactive and reference documentation to the user for all the numerous options, and allowing easy extensibility (new or modified plot types and configuration options) without requiring changes to hand-crafted UI or graphical output code.

For this reason the plotting infrastructure is built on a pluggable and object oriented set of interfaces that allow the host application to construct a graphical or text user interface, as well as its user documentation, by interrogating the plotting objects themselves, with little detailed knowledge of the plotting capabilities.

4. Invocation

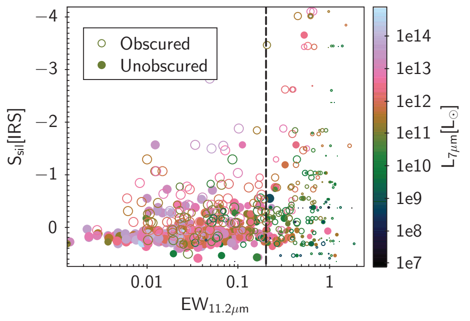
A number of options are available for invoking the plotting capabilities from outside the TOPCAT application.

First, the STILTS command-line interface allows generation of plots from a command line such as a unix shell, for instance:

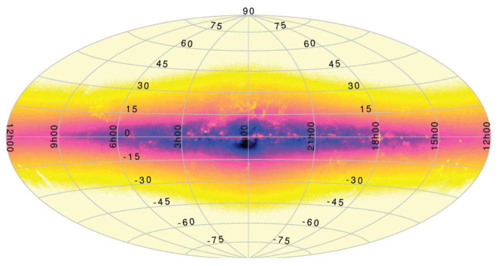
```
stilts plot2plane layer_1=mark in_1=data.fits x_1=BMAG-RMAG y_1=BMAG
```

will generate a simple scatter plot based on the given arithmetic expressions involving column names from the given input table (a FITS file in this case). The JyStilts package provides a Jython variant of this interface to allow use with python syntax and control structures.

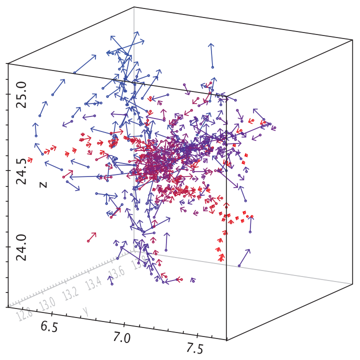
Java applications may make analogous invocations by specifying the parameter name/value pairs as entries in a supplied hash (a java `Map` object). In this case each *a)*



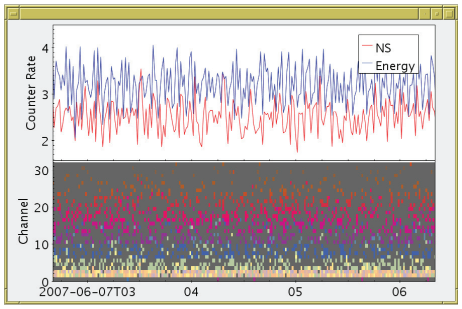
b)



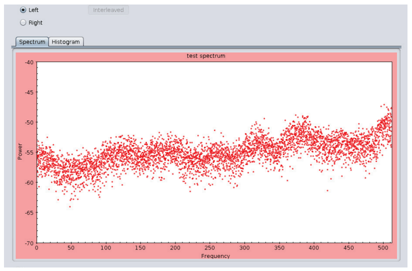
c)



d)



e)



f)

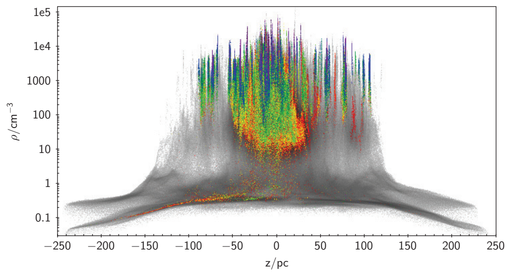


Figure 1. Example plots. (a) Points coded by marker shape, size and color (E. Hatziminaoglou, ESO). (b) Density map for the GUMS-10 Milky Way simulation dataset; 2 billion rows plotted in around 30 minutes. (c) Simulation data, with points and velocities represented. (d) Spectrogram and data samples plotted on an interactively scrollable time-axis. (e) Real-time display of an observed spectral data, 8 thousand points refreshed, easily, at 1 Hz (P. Harrison, Jodrell Bank). (f) SPH simulation data, 14 million points, plotted in 8 seconds (R. Smilgys, St. Andrews)

parameter value may be supplied either as a string, using the same rules as on the command-line, or as an object corresponding to the documented parameter type. For instance the value of a parameter supplying input table data may be either the name of a file containing a table in one of the supported formats (FITS, VOTable, CSV etc) or a programmatically constructed object implementing the `StarTable` interface, which could be a custom iterable over values in an existing memory-resident data structure.

Finally, java application code may use the low-level API documented in javadocs to specify all the details of the plot explicitly. Since this does not benefit from the extensive defaulting offered by the variants of the key/value interface it is quite verbose (50–100 lines to produce the example plot above) but it does provide detailed control and compile-time checking.

5. Graphics Output

Once a plot has been specified to the library it can be output in two basic ways.

First, it can generate a live interactive plot. From the command line, this appears in the form of a window popped up on the screen, and from the API as a Swing `JComponent` widget that can be incorporated into the host application's GUI. In either case this window can be resized at will and offers the user sophisticated interactive navigation capabilities (pan, zoom and for 3d plots rotation and recentering).

Alternatively, it can generate a static output file in one of several supported bitmap (PNG, GIF, JPEG) or vector (PDF or PostScript) graphics formats. Options for convenient and efficient production of animations are also provided.

6. Future Work

The STILTS plotting library is still somewhat experimental, and there may be adjustments to the API in future releases to accommodate more surface and layer types and configuration options. Full reference and some tutorial documentation is already available in the STILTS user document, but improved tutorial documentation and more code examples will be forthcoming. Efficiency improvements will be explored by use of multithreading. Integration with the TOPCAT application will be improved by showing the command-line settings corresponding to plots set up in the GUI.

We hope that the library presented here will be of use for scientists and application developers with requirements for plotting from java-friendly contexts.

References

Taylor, M. B. 2005, in Astronomical Data Analysis Software and Systems XIV, edited by P. Shopbell, M. Britton, & R. Ebert, vol. 347 of Astronomical Society of the Pacific Conference Series, 29

— 2006, in Astronomical Data Analysis Software and Systems XV, edited by C. Gabriel, C. Arviset, D. Ponz, & S. Enrique, vol. 351 of Astronomical Society of the Pacific Conference Series, 666

— 2014, in Astronomical Society of the Pacific Conference Series, edited by N. Manset, & P. Forshay, vol. 485 of Astronomical Society of the Pacific Conference Series, 257