ASTRONOMICAL DATA ANALYSIS SOFTWARE AND SYSTEMS XIV
ASP Conference Series, Vol. 347, 2005
P. L. Shopbell, M. C. Britton, and R. Ebert, eds.

TOPCAT & STIL: Starlink Table/VOTable Processing Software

Mark B. Taylor

H. H. Wills Physics Laboratory, Bristol University, Tyndall Avenue, Bristol, UK

Abstract. The Starlink Tables Infrastructure Library (STIL) is a pure-Java, open source library for I/O and manipulation of tabular data such as astronomical catalogs. It is designed to be high-performance and to cope with large tables. The core library is format-neutral, with the work of serialization and deserialization performed by pluggable format-specific I/O handlers. This means that the programmer seees a high-level abstraction of a table which is easy to work with, and also that support for new data formats can be added easily. Supplied handlers provide support for VOTables, FITS table extensions, relational databases via SQL and plain text tables, amongst others. The VOTable handler is believed to be the only existing library capable of reading or writing all the defined VOTable encoding formats (TABLEDATA, FITS, BINARY).

TOPCAT, based on STIL, is a user-friendly graphical program for viewing, analysis and editing of tables. It has facilities for plotting, cross matching, row selection, sorting and manipulation of data and metadata. Synthetic columns can be created and row selections made using a powerful and extensible algebraic expression language.

1. Introduction

Tables are common in astronomy, and are a prominent feature of the data produced, transmitted and consumed by human and software elements of the Virtual Observatory. The fact that the VOTable format is one of the first standards to become an IVOA recommendation[1] bears witness to this observation. A common example of astronomical tabular data is an object catalogue, but other examples, such as event lists, are important too. This paper describes STIL, a library for generic I/O and processing of tables, and TOPCAT, a graphical user application built on top of STIL. Both of these products are open source (released under the GNU Public License) and pure Java (J2SE1.4), which makes them highly portable and easy to deploy. They have been developed for the Starlink Project.

[1] `http://www.ivoa.net/Documents/latest/VOT.html`

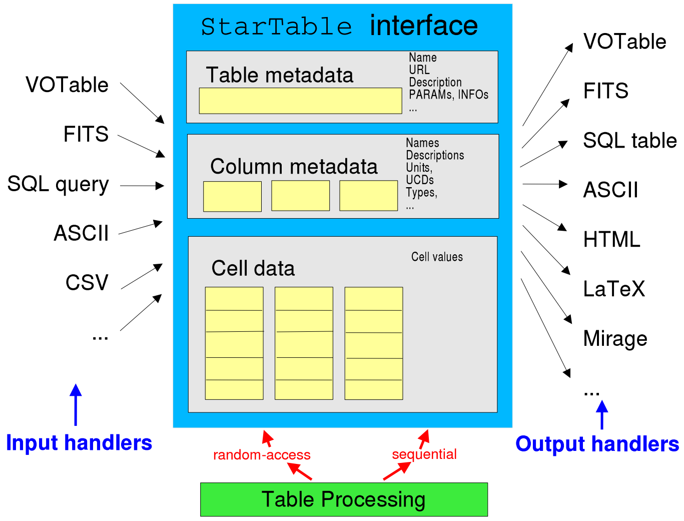

Figure 1. Schematic of STIL architecture.

2. STIL

The Starlink Tables Infrastructure Library (STIL)[2] is a generic I/O and processing library for tabular data. Central to STIL is a relatively simple model of what a table is, namely a data structure which has some per-table metadata, per-column metadata and the cell data themselves. The `StarTable` Java language interface embodies this data model, and it is `StarTable` objects that the programmer deals with when manipulating tables within STIL.

STIL has no native external data format, but a number of pluggable input and output handlers. Input handlers perform the job of deserializing tables from external storage to turn them into StarTables, while output handlers perform the opposite task of taking a StarTable and serializing it to external storage. Since these I/O handlers are separate from the core of the library, it is easy to change their implementation and add new ones without affecting application code. It is possible to install a new handler at run time by specifying its class name to an application. This design resembles the way that Java's JDBC database communication layer works. A schematic diagram of the architecture is given in Figure 1.

Among the I/O handlers supplied are ones which can read/write VOTable documents, FITS BINTABLE and TABLE extensions, relational databases (using SQL), and a number of text-based file formats including Comma-Separated Values. A feature of this design is that conversion between any of the supported formats is trivial, and STIL comes with a simple command-line utility `tablecopy` which performs this task.

As well as I/O, STIL provides a number of facilities for manipulation of tables including ways to add, remove and rearrange columns and rows, join and split

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

tables, and modify data and metadata. Flexible and efficient facilities for cross-matching are also distributed with the library, although these are currently not fully documented and somewhat experimental.

Care has been taken to make the I/O and processing facilities scalable throughout STIL; the aim in particular is that it should be possible to process a table with an unlimited number of rows in a limited amount of memory. The `StarTable` interface provides both sequential and random methods of data access for different processing requirements; in the former case limited memory use can be achieved by streaming the data a row at a time, and in the latter by caching the cell data in a temporary disk file, for which the library provides facilities.

STIL is fully documented; the public classes have comprehensive javadocs, and a user document (SUN/252), which contains examples and overview documentation, is available in HTML and PDF formats.

2.1. STIL for VOTables

As noted above, one of the formats supported by STIL is the XML-based VOTable format. Since this is of particular importance in the Virtual Observatory era, and because it has some notable features, there follow a few comments on this handler in particular.

The STIL VOTable parser is at time of writing, as far as we know, the only one which fully supports the VOTable standard. Although other available parsers can read the pure-XML TABLEDATA variant of the format, no others can read the FITS and BINARY variants. For reasons of bandwidth and CPU efficiency, TABLEDATA is inappropriate for encoding very large amounts of data, so this ability is of considerable significance. As well as reading, STIL also makes it easy to write VOTables in any of the three variants.

STIL can also provide to the programmer a hierarchical in-memory view (DOM) of the structure of a VOTable document. By use of custom stream-based processing (SAX) it is able to do this using modest amounts of memory even when the tables contained are large.

3. TOPCAT

TOPCAT[3] is a graphical user application for viewing, analysis and editing of tables. Being based on STIL, it can read and write tables in many formats, and it is extensible to new ones in the same way. It is not in practice able to cope with tables of unlimited size, but is happy to manipulate fairly large ones; on a normal desktop machine tables of order 10^6 rows × 10^2 columns can be processed easily. The program can be deployed in a number of ways, including as a WebStart application or from a single local jar file. Comprehensive user documentation (SUN/253) is available either in PDF or HTML form, or from within the program's context-sensitive help browser.

The program offers many ways to view and manipulate the data and metadata of tables. Some of the actions it permits are:

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

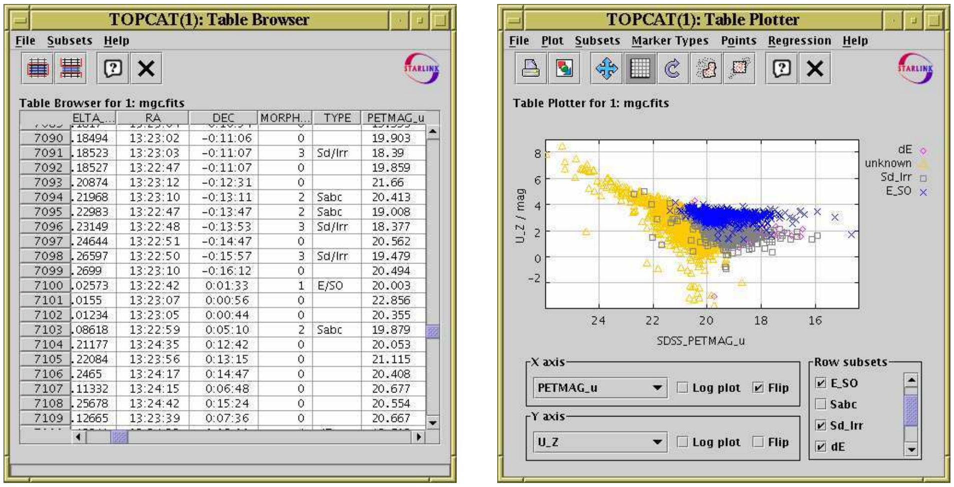

Table Browser for 1: mgc.fits

	ELTA_...	RA	DEC	MORPH...	TYPE	PETMAG_u
7090	.18494	13:23:02	-0:11:06	0		19.903
7091	.18523	13:23:03	-0:11:07	3	Sd/Irr	18.39
7092	.18527	13:22:47	-0:11:07	0		19.859
7093	.20874	13:23:12	-0:12:31	0		21.66
7094	.21968	13:23:10	-0:13:11	2	Sabc	20.413
7095	.22983	13:22:47	-0:13:47	2	Sabc	19.008
7096	.23149	13:22:48	-0:13:53	3	Sd/Irr	18.377
7097	.24644	13:22:51	-0:14:47	0		20.562
7098	.26597	13:22:50	-0:15:57	3	Sd/Irr	19.479
7099	.2699	13:23:10	-0:16:12	0		20.494
7100	.02573	13:22:42	0:01:33	1	E/SO	20.003
7101	.0155	13:23:07	0:00:56	0		22.856
7102	.01234	13:23:05	0:00:44	0		20.355
7103	.08618	13:22:59	0:05:10	2	Sabc	19.879
7104	.21177	13:24:35	0:12:42	0		20.053
7105	.22084	13:23:56	0:13:15	0		21.115
7106	.2465	13:24:17	0:14:47	0		20.408
7107	.11332	13:24:15	0:06:48	0		20.677
7108	.25678	13:24:42	0:15:24	0		20.554
7109	.12665	13:23:39	0:07:36	0		20.667

Table Plotter for 1: mgc.fits

X axis: PETMAG_u ☐ Log plot ☑ Flip

Y axis: U_Z ☐ Log plot ☐ Flip

Row subsets: ☑ E_SO ☐ Sabc ☑ Sd_Irr ☑ dE

Figure 2. Example windows from TOPCAT: the table browser window and the plot window

- View/edit table data in a scrollable browser
- View/edit table and column metadata
- Re-order, hide and unhide existing columns
- Insert algebraically-defined “synthetic” columns
- Sort the rows
- Define row subsets (selections of the rows) in various ways
- Plot columns against each other, distinguishing different row subsets
- Calculate statistics on each column for some or all rows
- Perform cross-matching between tables or within a table
- Create a new table by concatenating the rows of two existing ones

Two of TOPCAT’s many windows are shown by way of example in Figure 2. Some, though by no means all, of the program’s key capabilities are described in more detail in the remaining sections.

3.1. Plotting

The Plot Window displays a scatter-plot of values from one of a table’s columns against those from another. The columns can be selected interactively, and points may be plotted on logarithmic or reversed axes if desired. It is easy to zoom in and out to focus on regions of interest in the plot. If multiple row subsets have been defined, they will be plotted with different symbols, and the user can select which subsets are displayed and control the kinds of plotting symbols that are used. New subsets can be defined from the plot by indicating a rectangular region or by drawing an arbitrary shape or shapes with the mouse. The plot currently in view can be exported at any time to Postscript or GIF format.

3.2. Joining Tables

TOPCAT provides flexible and efficient facilities for cross-matching, either between two or more tables, or internally to a single one. The most common match criterion is a maximum angular separation between two (RA,Dec) points on the celestial sphere, but a range of other criteria are available including ones based on separation in isotropic or anisotropic Cartesian space of one, two or more dimensions, value equivalence, and combinations of these. In most cases, the speed of the match algorithm scales as $O(N \log N)$, where N is the total number of rows involved.

It is also possible to concatenate two tables "head-to-toe" by specifying the correspondence of columns between them.

3.3. Algebraic Expressions

A powerful feature of TOPCAT is the ability to create new columns or specify row subsets algebraically. In both cases the user enters a textual expression in which the names or identifiers of table columns serve as variable names; this expression can be evaluated for each row with each column identifier evaluating to that column's entry in the row. The expression thus defines a new "synthetic" column, and in the case of a boolean-valued expression it can be taken to define a row subset (if the expression evaluates true for a given row, that row is taken to be included in the subset).

The expression syntax is powerful and extensible; expressions are actually written in the Java language and compiled to bytecode prior to evaluation. This means that the full power of a programming language can be used to define operations. A number of functions such as arithmetic, trigonometric and string manipulations are initially available, but the user can augment these by supplying Java classes which define new functions and making them known to the program at run time. Despite this flexibility, it is in many cases easy for the non-specialist to specify algebraic expressions; for instance to create a new column which contains the average of columns named RMAG and BMAG, it is only necessary to enter the expression "`(RMAG+BMAG)*0.5`".

3.4. Activation Actions

Facilities exist in TOPCAT to focus on a particular row of a table, which can be useful for instance if it is an outlier in some sense. If you click on a row in the table browser, the corresponding point becomes highlighted in the plot window, and vice versa. It is also possible to cause other actions to take place when a row is "activated" in this way; one possibility is to display in SoG or SPLAT (Giaretta et al. 2005) an image or spectrum related to the selected row. There are pre-packaged facilities to display an image of the sky region surrounding a row obtained from certain cutout servers (DSS, 2MASS quick-look and SDSS), but activation can be configured to trigger almost any action, defined by user-supplied classes.

References

Giaretta, D, et al. 2005, in ASP Conf. Ser., Vol. 347, ADASS XIV, ed. P. L. Shopbell, M. C. Britton, & R. Ebert (San Francisco: ASP), 22