

# TOPCAT/STILTS Advanced Tutorial

---

Mark Taylor  
December 2015

---

## Contents

- [1 Introduction](#)
  - [2 Data Acquisition](#)
    - [2.1 File Formats](#)
    - [2.2 Getting data from the VO](#)
  - [3 Linked Views](#)
    - [3.1 Linked views in TOPCAT](#)
    - [3.2 Linking with other tools using SAMP](#)
  - [4 TAP](#)
    - [4.1 Locate a TAP service](#)
    - [4.2 Explore Metadata](#)
    - [4.3 Write/Run a query](#)
  - [5 Crossmatching](#)
    - [5.1 Crossmatch Window\(s\)](#)
    - [5.2 CDS X-Match window](#)
    - [5.3 TAP Join](#)
    - [5.4 Multi-Cone](#)
    - [5.5 Crossmatching Tips](#)
  - [6 Visualisation](#)
    - [6.1 Get RAVE data](#)
    - [6.2 Sky plot](#)
    - [6.3 Plane Plot](#)
    - [6.4 3D Plot](#)
  - [7 Activation Actions](#)
  - [8 STILTS/JyStilts](#)
    - [8.1 JyStilts](#)
- 

## 1 Introduction

These are notes for the Advanced TOPCAT tutorial at the ASTERICS VO school at ESAC, December 2015. It covers use of the desktop GUI tool [TOPCAT](#) and its command-line evil twin [STILTS](#).

These tools are designed to do things with tabular data - typically source catalogues. They don't do science for you, but they let you do the mechanical manipulation of tables that you need to do to understand their science content in detail.

TOPCAT and STILTS can do basically the same things, but are used in different ways. TOPCAT is easier to learn, and good for interactive use, especially exploring data to get a feel for what's there. For production work, it is sometimes better to move on to STILTS, which has a steeper learning curve but can be scripted for repeated or reproducible work. STILTS can also, for some purposes, be used for larger datasets.

Most links in the document are to:

- [SUN/253](#), the TOPCAT user document
- [SUN/256](#), the STILTS user document

These manuals contain much more detailed reference documentation, so should be consulted for more information.

Some of the data files used below are:

- [tycho-pleiades.fits](#)
- [2mass-pleiades.fits](#) (10Mb)
- [rave-dr4.fits](#) (25Mb)
- [millennium-g2.fits](#)
- [sn-hst-spectra.vot](#)
- [messier.xml](#)

## 2 Data Acquisition

### 2.1 File Formats

TOPCAT/STILTS like FITS files. They can work with some other formats (CSV, VOTable, IPAC, a few others) but they are slower. More details [here](#). If you're going to work with a table (especially a large one) a lot it's a good idea to convert it to FITS first.

#### What file format?

- Small table (<1000 rows): doesn't matter.
- Medium-sized (fits into physical memory; (rows\*cols) < 20million?): **FITS**:
  - Binary format, loads fast
  - Topcat/stilts usually use *FITS-plus*, a variant that stores extra metadata. You can mostly forget this fact.
- Big (millions of rows, especially with lots of columns): **colfits**:
  - Stores each column contiguously (normal FITS stores each row contiguously); works better for very wide tables when you only need a few columns

#### Already got the data locally?

- In the right format (see above)?
  - **Load it**
- In TOPCAT-friendly-format (CSV, VOTable, IPAC)?
  - Convert to FITS/colfits
    - *either* load into TOPCAT and save as FITS/colfits
    - *or* convert using STILTS:
      - `stilts tpipe in=xxx.csv ifmt=csv out=xxx.fits`
- In some other format (HDF5, your favourite ASCII variant, ...)?
  - Use some other way to convert it; either to FITS directly, or to e.g. CSV so you can convert it as above. AstroPy supports a wide range of formats.

#### What about VOTable?



VOTable is mostly intended for data transfer. It's not particularly efficient for data access (especially random access). VO services use it to exchange data and metadata, but topcat prefers FITS locally.





VOTable is good for storing *metadata* - column descriptions, units, UCIDs, history information etc. FITS isn't. TOPCAT uses a home-made FITS variant it calls *FITS-plus* that has the best of both worlds. You don't need to worry about the details, but if you use topcat/stilts to convert from VOTable to FITS, the metadata isn't lost.

### 2.2 Getting data from the VO

There are various ways to get data from the VO. Some are more straightforward than others.

#### Data acquisition tips


- If it's *small enough*, grab the whole thing
  - [VizieR](#)  with **All Rows**
  - [TAP](#)  `SELECT * FROM table`
  - Google? Visit survey home page? (*not very VO*)
- If not, you have to think about what parts you need:
  - Select by sky region

- [VizieR](#)  with **Cone Selection**
- [Cone Search](#) 
- [TAP](#)  `SELECT ... FROM table WHERE 1=CONTAINS(POINT(..),CIRCLE(..))`
- Select by some other criterion
  - [TAP](#)  `SELECT ... FROM table WHERE ...`
- Select by crossmatching with another dataset (see [Section 5](#))

What counts as "small enough"? As far as TOPCAT's concerned, that's around a million rows. However, if you're on a slow internet link you may have to adjust this downwards.

### VizieR dialogue example

This gets a list of Tycho detections in the region of the Pleiades. We'll use this later for crossmatching examples.







1. Open the [VizieR Load window](#) from the  **VO|VizieR CatalogueService** menu
2. Ensure **Cone Selection** radio button (not All Rows) is selected
3. Enter "pleiades" in the **Object Name** field
4. Hit the **Resolve** button to fill in the RA and Dec fields
5. Enter "2" (degrees) in the **Radius** field
6. In the **Catalogue Selection** panel at the bottom, select the **Surveys** tab
7. Scroll down to **Tycho-2** (they're alphabetical) and click on it
8. Hit **OK**.
9. This loads two new tables into topcat; one for the main table, and one for the supplement. Ignore the supplement (you can **File|Discard Table** if you like), but (for convenience) rename the other one (`I_259_tyc2`) as "tycho" by typing in the **Label** field of the main control window with that table selected.

## 3 Linked Views

Here we plot data in one parameter space, identify the points in a subregion, and see where those fall in a different parameter space. First we do this using topcat windows, then, using SAMP, between topcat and another application.

### 3.1 Linked views in TOPCAT


Linked views in TOPCAT:

1. Use the Tycho-2 data (cone around pleiades location) downloaded from VizieR earlier.
2. Plot a colour-magnitude diagram:  $X=VTmag - BTmag$ ,  $Y=VTmag$ . Use the **Axes** control  control to flip the **Y** axis.
3. Open another Plane Plot  window, and plot proper motions:  $X=pmRA$ ,  $Y=pmDE$ .
4. Click on a point in one plot; the same object is highlighted in both plots.
5. Open the **Data View** window  clicking on points highlights the corresponding row and vice versa.
6. In the proper motion plot, see that there is a cluster of co-moving objects. Draw a blob around it to create a new subset (click , drag out the cluster region, and then click the same button  again). The **New Subset** window will pop up; choose a name (e.g. "comoving") and hit **Add Subset**.
7. The selected points are plotted in a different colour on the colour-magnitude diagram. If you look at the proper motion plot, you can see they are also identified in a different colour on that one - you can see the main sequence traced out by the comoving stars that form a single population.
8. Look at the **Subset** tab in the plot windows. You can use that to control the colour of different subsets and which ones are plotted.
9. Look at the **Subsets Window**  from the main control window toolbar. You can see the new subset. You can use this for other things.

### 3.2 Linking with other tools using SAMP

Many VO tools can communicate with each other using Simple Application Messaging Protocol (SAMP). A SAMP Hub process needs to be running on the desktop. Usually, this just works, since topcat and other tools start such a hub when they start up.

Here is an example of it in operation:

1. Start up Aladin.
2. Select some all-sky imagery (e.g. **DSS** from the list across the top of the screen).
3. Enter "pleiades" in the **Location** field, and zoom in.
4. Select the Tycho table in topcat, and use the **Interop|Send Table To...|Aladin** menu item. Positions of all the table rows are plotted.
5. From the **Subsets** window , select the **comoving** subset, and use the **Interop|Send Subset To...|Aladin** menu item. Positions of the comoving subset are highlighted in Aladin.
6. If you hover the mouse over positions in Aladin, they are highlighted in topcat plots. Check the **Broadcast Row** checkbox in topcat's main control window, and the same thing happens the other way round (clicking on a row in topcat highlights the same one in Aladin).

## 4 TAP

TAP is the **Table Access Protocol**. It lets you perform queries against a remote relational database. In TOPCAT, you write queries in ADQL (Astronomical Data Query Language). This is essentially a dialect of SQL.

TOPCAT gives you the [TAP Load Window](#). Open it using the  button on the main toolbar; it's also available from the **VO** menu or the **Load** window.

It's a complicated window, because it does a lot of stuff. It's organised in several tabs, and some of the tabbed parts have tabs of their own.

The most important jobs it lets you do are:

- Find/choose a TAP service to talk to
- See information about the tables available in that service
- Construct and submit ADQL queries, retrieving the results as tables loaded into TOPCAT

(it also lets you monitor running jobs and return to previously started ones).

### 4.1 Locate a TAP service

Use the [Select Service](#) tab (this is visible by default when you open the TAP window). The job of this tab is to let you choose a service to talk to. (If you try to open the **Use Service** now it won't let you, because you haven't chosen one).

You should see a list of all known TAP services (this is all the *registered* TAP services; some others for non-public use may exist as well). If you don't see that after a short delay ... there's probably a network or registry error. The services are listed in order of the number of tables they contain (this number is in brackets after the name).

If you know what service you want to use, you can browse the list and just click its name.

More often, you want to search for a given data set. To search for (e.g.) the RAVE survey:

- Type "rave" (it's not case-sensitive) into the **Keywords** field
- Click the **Find Services** button (or hit return)
- After a short delay, you should see a shorter list (3 items?) of services. These have tables with names or descriptions containing the term "rave". E.g. the line "GAVO DC TAP (3/142)" means the service "GAVO DC" has 142 tables altogether, and 3 of them (maybe) have something to do with RAVE.
- Click on the "handle" to the left of the GAVO service to open it up. This will list the three tables. If one of them looks like what you're after (*rave.main*) then click on that, or the service name, to tell topcat this service is the one you want to use.
- This fills in the **TAP URL** field at the bottom of the window.
- Now you can hit the **Use Service** button next to it, or alternatively the **Use Service** tab at the top.

*Note: not all TAP services are equal. This GAVO one (and some others using the same software) generally works very well, and supports many optional features like table upload and examples. Some other services may work less smoothly. Don't be too surprised if some things break. Hopefully, this will improve in the future...*

### 4.2 Explore Metadata

When you open the [Use Service](#) tab, after a short delay, you should see on the left of the window a list of tables in the service's database. The

top level of this tree shows you *Schemas*; these are just an organisational level that (sometimes) groups tables into related sets. Click on the handles to expand schemas and see the tables inside. The tabs to the right contain information depending what schema/table you have selected in the tree.

### Find a table you're interested in:

#### Browse the tree

If the service doesn't have too many tables, and they have sensible names, you might find the one(s) you're looking for by scrolling and clicking.

#### Search by metadata

If the list is very long, or the tables have unintuitive names, use the **Find** field. As you type in one or more search terms, the list below will be narrowed down to include only the relevant items. You can control whether the terms are combined using And/Or, and whether they match just table/schema name or description as well.

- Type in "rave" - you'll see three tables
- Type in "radial" - with just the **Name** checkbox checked you see none. If you check **Description** as well, it will show you more tables with "radial" in their table/schema description.

### Use the different tabs to show you different information:

- Click on the **Service** tab: This contains general information about the current TAP service. Near the bottom is (maybe) a list of *User-Defined Functions* - non-standard functions offered by the service, that may be useful.
- Click on the **Schema** tab. As you select different schemas in the tree, it will give you descriptions.
- Click on the **Table** tab. As you select different tables in the tree, it will give you descriptions.
- Click on the **Columns** tab. When you select a table in the tree, this will show you all the columns in that table. **This is the main useful information** you will need to construct a query and understand the results. You can click on the column-headers of this display to sort the table columns (i.e. displayed rows in this panel). Sorting by **Unit** can be useful to group table columns with similar values.
- Click on the **Hints** tab. **Use this to give you a clue how to write ADQL!**. Follow its advice about Examples at the top.

## 4.3 Write/Run a query

Now you can type an ADQL query into the text field at the bottom of the window. If you don't know how to do that, **don't worry**; there is help at hand.

#### Use the Hints tab

This gives you a very basic cheat sheet as a reminder of how to put together an ADQL query. At the bottom, it links to some other resources with a bit more information; clicking on blue links should open them in your browser. It also tells you to *use the examples*.







#### Use the Examples menu

At the bottom left of the window is an **Examples** menu. When you select an item, it will enter text into the entry box. In most cases, it relates to the table you are currently looking at in the metadata browser. Try some out; when the text appears, hit the **Run Query** button. Edit the text; if you make a syntax error it will be highlighted in red.

#### Service-Provided examples

**Service-Provided** examples are written by the hosting TAP service, so may give you some good ideas about what will work with these specific tables. Follow the instructions in the **Hints** tab to see them.

When you've got the idea, try your own queries, or edit the examples to taste. Useful hints:

- **Syntax errors** are highlighted for you with a red background.
- You can have **multiple queries** on the go at once in different tabs:  opens a new blank tab, and  copies the current text into a new tab.
- Text editing has full **Undo/Redo** using the / buttons or **Ctrl-Z/Ctrl-Shift-Z**.
- Insert a list of column names into the query by highlighting them in the column browser (ctrl-click) and then hitting the  button. The  button works the same way for the currently selected table name.

## 5 Crossmatching

There are various different ways to crossmatch tables against each other. Which is best depends on the details of what you want to do, how big the tables you want to match are, and what data services are available that provide the (large) tables of interest.

There are four main options available from TOPCAT (and STILTS), described below. The **examples** here identify objects in the region of the Pleiades with detections in both Tycho-2 and 2MASS, though note some of these methods are more appropriate than others for this job. These examples use the **Tycho** catalogue from the previous section. Alternatively, just download it from <http://andromeda.star.bris.ac.uk/data/tycho-pleiades.fits>.

## 5.1 Crossmatch Window(s)




The [Crossmatch Window](#)  is generally the easiest way to do it, very flexible, and usually quite fast.

### Notes

For this, you need both catalogues loaded into TOPCAT, so they can't be too big (<million rows?).

- For large tables, the crossmatch can run out of memory (symptoms: memory error popup window *or* it just grinds to a halt. Keep an eye on the Control Window [Memory Monitor](#)). Increasing heap memory (run with `-Xmx1000M`) may help.
- Think about the output options. Especially in crowded fields, the default **Best Match, Symmetric** can give surprising results. If you want **Best Match for each Table x Row**, select that.
- There are lots of different match types (**Algorithm** selector), not just Sky.
- You can do single-table or 3-, 4-, 5-table matches too - see the **Joins** menu.

### Example

1. You need to start with all the rows from the 2MASS Point Source Catalogue in the region of interest to do it this way (in some cases that's not feasible). Get them from the VizieR window (**2MASS-PSC**) like for the Tycho data above. You may need to increase the **Maximum Row Count** (plot the result on the sky  when loaded to check that the region looks like a circle, not a circle with some parts missing). You can relabel it `2mass`.
2. Open the **Pair Match window**  using the button in the main toolbar (or the **Join** menu)
3. Check the **Match Criteria** panel at the top; the defaults of **Sky** match with 1 arcsec error are reasonable.
4. Fill in the **Table** selectors with `tycho` and `2mass` tables.
5. Check that the two **RA** and **Dec** selectors are filled in with suitable columns
6. Hit **Go**.
7. It should do some calculations and pop up a window telling you how many matches have been found (about 500?). Click the  **Plot Result** button.


### Stilts equivalents

[tskymatch2](#), [tmatch2](#), [tmatch1](#), [tmatchn](#). Note these are still vulnerable to running out of memory.

The above example in STILTS looks like:

```
stilts tskymatch2 \
  in1=tycho-pleiades.fits ra1=_RAJ2000 dec1=_DEJ2000 \
  in2=2mass-pleiades.fits ra2=_RAJ2000 dec2=_DEJ2000 \
  join=1and2 find=best error=1 \
  out=tycho-2mass.fits \
```

## 5.2 CDS X-Match window



The [CDS X-Match window](#)  is based on the X-Match service provided by CDS. If you have one table loaded in topcat, and want to match against another one that's in VizieR (or the SIMBAD database), it works well, and is very fast, even for large tables.

### Notes

- It's not that straightforward to find out the VizieR ID for the remote table, if it's not one of the popular named ones in the selector box. You need the *table id* not *catalogue id*. The table searching facility at [TAPVizieR](#) web page is a good place to look.
- Only a restricted set of columns is available (not all from VizieR). If the columns you get back don't include the ones you want, you'll have to use a different method.

- You can only specify simple positional criteria.
- Increasing the **Block Size** may make it go faster. Or fail.

### Example

1. Open the **CDS Upload X-Match window**  using the button in the main toolbar (or **VO** menu).
2. Choose **2MASS** in the **VizieR Table ID/Alias** selector. If the table you want isn't there, try searching at <http://tapvizier.u-strasbg.fr/adql/> and use the table identifier (e.g. **II/246/out**) in the right hand column.
3. Select **tycho** as the **Input Table**
4. Check the **RA** and **Dec** columns are filled in correctly.
5. Check the **Match Parameters**. The defaults should be OK.
6. Hit **Go**
7. It should (quickly) tell you that a new table has been created.
8. You can overplot the results it on the same plot as before to compare the results. Add a new plot layer with the **Add Position Control**  button and select the new table. Vary the marker **Shape** and **Size** (in the **Form** tab) and **Colour** (in the **Subsets** tab) so that all the different markers are visible at the same time.


### STILTS equivalent

[cdsskymatch](#). This will work with a local table of unlimited size.

The above example in STILTS looks like:

```
stilts cdsskymatch cdstable=2MASS \
      in=tycho-pleiades.fits \
      ra=_RAJ2000 dec=_DEJ2000 radius=1 \
      find=best out=tycho-2mass.fits
```


## 5.3 TAP Join

TAP joins use the **TAP window** . If there are two huge tables in a remote TAP database, use TAP to join them. If you want to join a local table with a huge table in a remote TAP database, use a *TAP Upload* query (if available).

### Notes

- TAP is very powerful..
- ... but it's not so simple to use. Look at the **Hints** tab and examples!
- Not all TAP services support uploads.
- TAP service implementation is patchy. Not all services have the same capabilities. Some are a bit broken. This may get better in the future.

### Example

1. Open the **TAP Window**  using the button in the main toolbar (or **VO** menu).
2. Find a TAP service containing 2mass data. Type **2MASS** in the **Keywords** field and hit the **Find Services** button. It seems there are lots of services providing 2MASS or 2MASS-related data. Let's use **GAVO DC TAP** (near the top, probably). Select that row, and hit the **Use Service** button at the bottom.
3. When it's finished loading the metadata, type **2mass** in the **Find** box. No results. Try clicking the **Description** checkbox, which will check for text matches in the table description as well as name. Open the **twomass** entry in the tree and select table **twomass.data**.
4. Do the upload match the easy way: use the **Examples**. Make sure the **tycho** table is selected in the main Control Window, and **twomass.data** is selected in the TAP window. Then click the **Examples** button at the bottom of the screen, and select **Upload|Upload Join**. It puts some text (an upload join query) into the ADQL text box.
5. Take a look at the query. Delete "TOP 1000".
6. Hit the **Run Query** button.
7. It should load a new table in topcat. You can overplot this one too.

### STILTS equivalents

[tapquery](#), [tapskymatch](#). **tapskymatch** only does positional sky matches: but it works on unlimited sized tables, and is more



straightforward to use.


The above example in STILTS looks like:

```
stilts tpskymatch \
  tapurl=http://dc.g-vo.org/tap \
  tatable=twomass.data taplon=raj2000 taplat=dej2000 \
  in=tycho-pleiades.fits inlon=_RAJ2000 inlat=_DEJ2000 \
  sr=1./3600. find=best sync=true \
  out=tycho-2mass.fits
```



or

```
stilts tapquery tapurl=http://dc.g-vo.org/tap \
  nupload=1 upload1=tycho-pleiades.fits \
  ucmd1='colmeta -name ra _RAJ2000' ucmd1='colmeta -name dec _DEJ2000' \
  adql="SELECT * FROM twomass.data AS twom \
  JOIN TAP_UPLOAD.up1 AS tycho \
  ON 1=CONTAINS(POINT('ICRS', twom.raj2000, twom.dej2000), \
  CIRCLE('ICRS', tycho.ra, tycho.dec, 1./3600.))" \
  sync=true out=tycho-2mass.fits
```


## 5.4 Multi-Cone

The [Multi-Cone window](#)  lets you make one cone search for each row of a local table, effectively joining it with the remote one that is exposed via a Cone Search service.

### Notes

- These are *v.e.e.r.y s.l.o.w* and not usually a good idea, but if the remote data you need is not available in another form, it may be necessary.
- [Multi-SIA](#)  and [Multi-SSA](#)  do the same thing for Simple Image and Spectral Access services.
- ObsTAP services (TAP services with the `ivoa.obscore` table) may be another way to get image/spectral observation data).

### Example

1. Open the **Multi-Cone window**  using the **VO|Multicone** menu item
2. Search for a Cone Search service providing the 2MASS Point Source Catalogue. Type in "2mass" to the **Keywords** field and hit **Find Services**. It gives you too many results ... try "2mass point". Pick one that looks OK - (*hint: the Vizier one ("II/246") will probably work best*). When selected this will fill in the **Cone Search URL** field.
3. Enter `tycho` as the **Input Table**
4. Check that the **RA column** and **Dec Column** fields are filled in correctly.
5. Adjust the **Search Radius column** field - arsec not degrees!
6. Hit **Go**
7. It should, after a while (see the progress bar at the bottom), load a new table with the result. You can overplot it on the others.

### STILTS equivalent [coneskymatch](#)

The above example in STILTS looks like:

```
stilts coneskymatch \
  serviceurl='http://vizier.u-strasbg.fr/viz-bin/votable/-A?-out.all&-source=II%2F246%2Fout&' \
  in=tycho-pleiades.fits icmd=progress \
  sr=0.0002777 find=best \
  out=tycho-2mass.fits
```

## 5.5 Crossmatching Tips

To summarise, the best choice mainly depends on whether your tables are Small (<thousand row), Medium (<million row), or Huge.




T1	T2	Crossmatch	CDS X-Match	TAP	Multi-Cone
Small/Medium	Small/Medium	YES	yes	yes	no
Small	Huge	no	YES	yes	yes
Medium	Huge	no	YES	yes	no
Huge	Huge	no	(web iface)	yes	no

Some miscellaneous tips:

- When you've done a crossmatch, **plot the results** to see if they look sensible
- Crossmatching works on the [Apparent Table](#) - Current Subset and currently visible rows only.
- Before you do a crossmatch, you might like to restrict the columns (or rows?) in the local table(s) so the result isn't too unwieldy.




## 6 Visualisation

### 6.1 Get RAVE data



1. Acquire the RAVE DR4 dataset. There are various ways to do it. Try one of:
  - VizieR all-sky
  - TAP
  - <http://andromeda.star.bris.ac.uk/data/rave-dr4.fits>
2. Acquire the list of Messier objects (<http://andromeda.star.bris.ac.uk/data/messier.xml>, or some other way).
3. Look at the columns ( button on main toolbar). Make sure you have (2MASS) J, H and K magnitudes, heliocentric radial velocity (*rv*), parallax (*P<sub>l</sub>x*) and calibrated metallicity (*met*). If they don't have those names, you can rename the columns by double-clicking on the names in the columns display and typing the new name.

### 6.2 Sky plot

View the RAVE data on the sky.

1. Open a Sky Plot window .
2. You might want to split the display and control panels with the **Float Controls**  button, especially if your screen is small/short.
3. Ensure the RAVE table is selected in the **Table** selector.
4. The **Lon** and **Lat** fields should be filled in automatically; if not, select RA and Dec values
5. Note that darker coloured regions are denser. You can see slight inhomogeneities in the observation pattern.
6. Navigate on the plot using the mouse. See line at the bottom of the window with the little mouse icons to see what you can do. There's more detail (including what to do if you don't have 3 mouse buttons) if you click the little question mark button bottom left (it takes you [here](#)).
7. Change the view by opening the **Axes** control  and adjusting the options:
  1. Select **View Sky System: galactic** to see positions in galactic (rather than equatorial) coordinates
  2. Select **Projection: aitoff** to see an all-sky projection
  3. You can fiddle around with the way the coordinate grid is drawn and labelled in the **Grid** and **Font** tabs

For fun, you can overplot the Messier objects on the same plot.













1. Add a new layer by clicking the **Add Position Layer**  button
2. Select the Messier table in the **Table** selector.
3. The **Lon** and **Lat** fields should be filled in automatically; if not, select RA and Dec values
4. See the new points plotted.
5. Make them more obvious by adjusting options in the **Form** tab
6. Change the size and shape with the **Size** and **Shape** selectors.
7. Change the colour using the **Subsets** tab.
8. Add a text label layer with the  button
9. Select the **Name** or **ID** column in the **Label** selector
10. Change the font size etc to taste.

This is how you do that in STILTS:

```
stilts plot2sky \
  viewsys=galactic \
  in1=rave-dr4.fits lon1=raj2000 lat1=dej2000 \
  layer1=mark \
  in2=messier.xml lon2=RA lat2=DEC color2=cyan \
  layer2a=mark size2a=2 \
  layer2b=label label2b=name
```

## 6.3 Plane Plot

Do a colour-colour plot of RAVE data:

1. Open the Plane Plot window 
2. You might want to split the display and control panels with the **Float Controls**  button, especially if your screen is small/short.
3. Select the RAVE table in the **Table** selector.
4. Enter the colour quantities as coordinates: **X** as  $h-j$  and **Y** as  $h-k$ . Note these are using topcat's [expression language](#); in this case the expressions are pretty simple.
5. Navigate so that most of the plotted points fill the screen (navigation hints at the bottom of the window; details if you click the little question mark or [here](#)).
6. Investigate the density profile of the plot.
  - o Experiment with different settings of the **Shading** selector in the **Form** tab:
    -  **Auto** (default): you can get a fairly good idea that the density is bimodal, but not all the details
    -  **Flat**: hopeless for this kind of high-density plot
    -  **Translucent**: adjust the **Transparency Level** slider. Can see bimodality, but hard to see all levels of detail at once.
    -  **Transparent**: like Translucent, but behaves differently if you zoom in/out
    -  **Density**: more options to explore density profiles. Fiddle with colour maps, scaling, clips, quantisation, ...
  - o You can also try other *Forms*:
    - **Density** 
    - **Contour**  (probably need to increase **Smoothing**)
7. Investigate how metallicity (the column may be called  $[M/H]$  or something else, depending where you got the data) varies over the colour-colour space. You can do this by using the metallicity value to colour-code plotted points with the remaining **Shading Modes**. In both these cases, a colour ramp is displayed on the right, and you can control the map using the **Aux Axis Control**  control that appears in the control stack on the left. Play around with the colour map etc in the **Map** tab and the data range in the **Range** tab for best results. The **Aux** and **Weighted** shading modes can both do this:
  - o  **Aux**: select the metallicity column as the **Aux** coordinate. You can see some low-metallicity hotspots, but they're a bit noisy because points are plotted over each other. Adjusting transparency, zooming in, or making points smaller may help a bit.
  - o  **Weighted**: select the metallicity column as the **Weight** coordinate. This averages the values at each pixel and makes it easier to see what's going on. The combination method is **Mean** by default; you can experiment with the others. **Median** may give you a more robust view, but *note* it's slower.


There are three regions of the colour-colour space with low metallicity. What's going on here? (*Hint: I don't know!*)

This is how you do that in STILTS:



```
stilts plot2plane \
  in=rave-dr4.fits \
  icmd='colmeta -name met c[M/H]K' \
  x=h-j y=h-k \
  xmin=-1.3 xmax=0.2 ymin=-0.2 ymax=0.7 \
  auxmap=cubehelix auxmin=-3 auxmax=0 auxfunc=square \
  layer1=mark shading1=weighted weight1=met
```





## 6.4 3D Plot

Get some theoretical data from the Millennium simulation in one of the following ways:


- From the Millennium Query load dialogue:
  1. Use the  **VO|GAVO Millennium Run Query** menu item
  2. Select the example menu item **GalaxySamples|G2**
  3. Hit **OK**
- If that doesn't work, just download it from <http://andromeda.star.bris.ac.uk/data/millennium-g2.fits>.

Plot X,Y,Z positions:


1. Open the **Cube Plot** window 
2. You might want to split the display and control panels with the **Float Controls**  button, especially if your screen is small/short.
3. Ensure the simulation table is selected in the **Table** selector.
4. Select columns *x*, *y* and *z* as the **X**, **Y** and **Z** coordinates.

Note the default shading mode in 3D is **Flat** . There's no **Auto** because it plays badly with multi-dataset plots. The transparency options ,  are OK. But if you have a single-dataset plot you can use **Density**  mode.




Navigate in 3D:

- Most useful actions:
  - Drag to rotate around the center of the cube
  - To look at a particular point or group of points, right-click (or CTRL-click) to re-center around that point
  - Use the mouse wheel (or your equivalent - 2-fingered drag on mouse pad?) to zoom in/out around the center
- Use the  button to rescale if you get lost.
- The navigation hints are at the bottom of the window as usual, and the help link points [here](#).

Plot velocities:

- In the **Form** tab, add a new **XYZVector** layer by clicking .
- Fill in the **X/Y/Z Delta** fields with the velocity columns, *velX/Y/Z*. See the little arrows.
- Uncheck the **Mark** control, so only the arrows and not the spots are visible
- Try different shapes in the **Arrow** selector.
- The arrows are automatically scaled to be a sensible size. You can make them all bigger or smaller by sliding the **Scale** slider. If you want to see the absolute sizes represented by the DeltaX/Y/Z coordinates, you can uncheck the **Auto Scale** control.

Plot separate subsets:

- This table has a column *type* which identifies galaxies as central/FOF (0), central/subhalo (1) or satellite. It's useful to plot these as separate subsets.
- Open the **Subsets** window 
- You could add three new subsets using the **Add New Subset**  button using the expressions *type==0*, *type==1* and *type==2*.
- But to save you the trouble, you can use the **Classify By Column** window  (in the **Subsets** menu, or on the toolbar)
- Enter *type* in the **Classification Value** selector, and hit **Classify** button
- Review the subsets it's identified, and if you like them hit **Add Subsets**.
- It will add the new subsets, and they will be reflected in the plot. In the plot window you can use the **Subsets** tab to control which ones are plotted and what colours are used.


This is how you do one of those plots in STILTS:

```
stilts plot2cube \
  in=millennium-g2.fits \
  x=x y=y z=z \
  layer1=mark \
  xdelta2=velx ydelta2=vely zdelta2=velz \
  shading2=transparent opaque2=3 \
  layer2a=xyzvector icmd2a='select type==1' color2a=cyan \
  layer2b=xyzvector icmd2b='select type==2' color2b=magenta \
  layer2c=xyzvector icmd2c='select type==0' color2c=606060 \
  seq=2a,2b,2c
```

## 7 Activation Actions

Activation Actions are things you can cause to happen when you "activate" a row. Activation happens when you click on a row in the [Data Window](#)  or click on a point in a plot. Some of these use **SAMP**.

They are controlled from the [Activation Window](#), which you can reach using the **Activation Action** button in the main Control Window.

This example uses a list of HST spectral observations of positions in the Asiago supernova catalog (B/sn in VizieR). It was obtained by using the [Multiple SSA window](#)  from the **VO** menu with B/sn as the local table and `ivo://mast.stsci/ssap/hst` as the remote service.

Alternatively, like this (which took about 5 minutes):

```
stilts conesky match \
  in='http://vizier.u-strasbg.fr/viz-bin/votable?-source=B%2fsn' \
  icmd='replacecol -units deg RAJ2000 hmsToDegrees(RAJ2000)' \
  icmd='replacecol -units deg DEJ2000 dmsToDegrees(DEJ2000)' \
  icmd='select RAJ2000>0' \
  icmd=cache icmd=progress \
  servicetype=ssa \
  serviceurl='http://archive.stsci.edu/ssap/search2.php?id=HST&' \
  dataformat=fits \
  ra=RAJ2000 dec=DEJ2000 sr=5./3600. \
  usefoot=false find=best \
  parallel=5 emptyok=false \
  ocmd='colmeta -utype ssa:Access.Reference utype$ssa_access_reference' \
  ostream=true \
  out=sn-hst-spectra.vot
```

You can find the data file at <http://andromeda.star.bris.ac.uk/data/sn-hst-spectra.vot>.

We will try a few activation actions. Click the **Activation Action** button in the Control Window to open the **Activation Window** and select different options by selecting one of the radio buttons on the left, filling in the corresponding panel, and clicking **OK** at the bottom (which closes the window).

### Display Cutout Image

Check the RA/Dec columns are filled in, and pick an all-sky cutout service. Activating rows displays a postage stamp image in a basic image viewer.


### Transmit Coordinates

Start up Aladin, choose (e.g.) 2MASS All-Sky imagery from the menu along the top, and zoom in (e.g. mouse wheel). Check the RA/Dec columns are filled in. Activating rows moves the Aladin view to the position of that row.

### View URL as Spectrum

Start up a SAMP-capable spectrum viewer (e.g. [SPLAT-VO](#)). The **Spectrum Location Column** should be the `url` column (may get filled in automatically). Activating rows loads the corresponding spectrum into SPAT.

### Custom code

You can enter here functions that are executed on activation. Look at the **Activation Functions** section in the [Available Functions window](#)  for the options. One of the most useful are the `exec` functions listed under **System**. These let you execute a shell command.

If you enter `'exec("curl", url, "-o", sn+".fits")'`, then every row activation will result in a FITS file being written to the current directory, named after the nearby supernova (e.g. `1987A.fits`), containing the corresponding HST spectrum. For more control over what happens, you could write a custom shell script that takes command-line parameters supplied by such an `exec` invocation.

## 8 STILTS/JyStilts

STILTS is a set of command line tools that can do more or less the same things as TOPCAT. For full details see [the manual](#), but here is some explanation of the basic concepts.

### Invocation

The easiest thing is to copy the files [stilts.jar](#) and [stilts](#) files into the same directory, set `stilts` executable (`chmod u+x stilts`) and use that. Alternatively, do `java -jar stilts.jar`.

All commands are of the form

```
stilts <task-name> <param1>=<value1> <param2>=<value2> ...
```

## tpipe

The most basic command is [tpipe](#). This takes an input file, lets you apply a number of [filters](#) (zero or more `cmd=...` parameters) that modify the table data or metadata, and [output it in some way](#) (`omode=...` parameter). It works like a Unix pipeline, but processing table data and metadata not lines of an input stream. If there are no `cmd` parameters, no changes are made. If there is no `omode` parameter, the result is just written to file (format default or as specified)

### Format conversion

```
stilts tpipe in=messier.xml out=messier.fits
```

### Row selection

```
stilts tpipe in=rave-dr4.fits cmd="select abs(hrv)>100" omode=count
```

## Filters

Many filters are listed in [the manual](#). In `tpipe` they are given as `cmd=` arguments. For some other commands which distinguish between different table streams they can be e.g. `icmd/ocmd` (for input/output) or `cmd1/cmd1` (for two different input tables) etc. Here are some examples of using filters:

### Add a new calculated column

```
cmd='addcol B_R BMAG-RMAG'
```

### Select only rows in a given region

```
cmd='select skyDistanceDegrees(RA,DEC,78.63,-8.20)<0.001'
```

### Select only ten reddest objects

```
cmd='sort RMAG-BMAG' cmd='head 10'
```

## Expression language

- Mostly "common-sense" programming language syntax (*it's actually java syntax*)
- Column names as variables, expressions are evaluated for each row
- Avoid funny characters in column names (start with alphabetic; then alphanumeric/underscore)
- You can use "\$n" syntax as alias for column #n.
- Various maths/astro functions (`exp(x)`, `janskyToAb(flux)`, `substring(str,index),...`) - use `stilts funcs` to investigate.
- Logical expressions: `&&` (and), `||` (or), `==` (equals), `!` (not), ...
- Conditional expression: `<expr> ? <value-if-true> : <value-if-false>`

## Getting help

Use [the manual](#)!

- Identify the command you want from the [Commands by Category](#) section (if in doubt, try `tpipe`)
- Look at its **Usage** subsection to see what the parameters are
- Look at its **Examples** subsection to see examples of it in use

Help is available from the command line too:

- If you omit a required parameter you will be prompted
- You can type `stilts <task-name> -help` for help on the command
- You can type `stilts <task-name> help=<param-name>` (or `?` at the prompt) for help on a parameter.

Admittedly, the plotting commands can be quite complicated... They have their [chapter](#) in the manual to help.

## 8.1 JyStilts

**JyStilts** is a Jython (like python, but implemented in Java and with no C extensions) front-end to `stilts`.

### Invocation

```
jython -jar jystilts.jar .
```

### Documentation

[JyStilts chapter](#) in the manual.

### Comparison with command-line `stilts`

#### From the shell

```

stilts tskymatch2 in1=survey.fits \
                 icmd1='addskycoords fk4 fk5 RA1950 DEC1950 RA2000 DEC2000' \
                 in2=mycat.csv ifmt2=csv \
                 icmd2='select VMAG>18' \
                 ra1=ALPHA decl1=DELTA ra2=RA2000 dec2=DEC2000 \
                 error=10 join=2not1 \
                 out=matched.fits

```

### JyStilts

```

>>> import stilts
>>> t1 = stilts.tread('survey.fits')
>>> t1 = t1.cmd_addskycoords(t1, 'fk4', 'fk5', 'RA1950', 'DEC1950', 'RA2000', 'DEC2000')
>>> t2 = stilts.tread('mycat.csv', 'csv')
>>> t2 = t2.cmd_select('VMAG>18')
>>> tm = stilts.tskymatch2(in1=t1, in2=t2, ra1='ALPHA', decl1='DELTA',
...                       error=10, join='2not1')
>>> tm.write('matched.fits')

```

### Pythonic features

- Array-like row-counting: `len(t)`
- Array-like row slicing: `t[100:200]`
- Iteration over rows: `for row in t:`
- Index columns by name or number: `row['BMAG']` or `row[2]`
- Docstring help: `help(t.cmd_addcol)`
- ...

### JyStilts Advantages

- Good if you like python
- Syntax is less tortured (parameter quoting problems not so bad)
- Can be more efficient for multi-step procedures: intermediate results are kept as objects in memory, they don't have to be de/serialised to (e.g.) FITS files for every command.
- Pythonic access (to some extent) to table data/metadata.