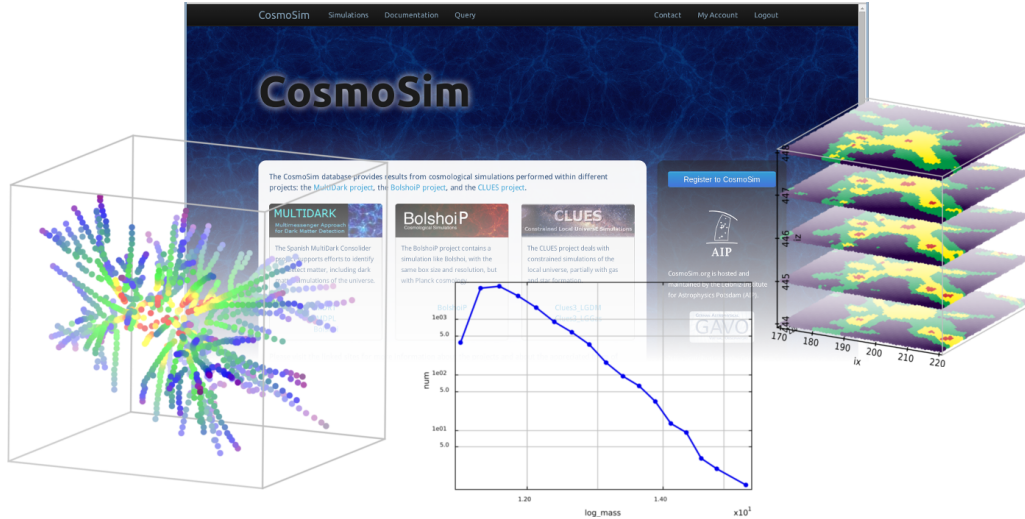


Introduction to simulation databases using CosmoSim



Kristin Riebe*, GAVO

June 27, 2016

Introduction

Simulation databases like the Millennium Database or CosmoSim contain data sets from cosmological simulations. Each user has access to catalogues of dark matter halos and their history (merger trees), supplemented by (mock) galaxy catalogues and information about the density field or tables with the simulation particles themselves and other data sets. Access to these data is possible via a web interface directly with the browser; for larger data it is more convenient to access the data via [TOPCAT](#), `wget` or using their TAP/UWS interface, if available. This tutorial gives a quick overview on the CosmoSim web interface and shows an example use case for extracting the mass function of a simulation, particles for a halo and a halo's merger tree. More demos, examples and tutorials can be found at the [CosmoSim](#) webpage.

Software: [CosmoSim](#) web interface, [TOPCAT](#)

1 The web interface

1. Open <https://www.cosmosim.org> in a web browser. This webpage provides access to simulations from different projects, with data sets like dark matter halo catalogues and merger trees. The available simulations and data are described in the **Simulations** section of the documentation. There are also many documentation pages on the database and table structures, example queries and a demo video¹. You don't need to look at this now, just be aware that there is extensive documentation (and it may even be more up-to-date than this tutorial).

*Leibniz-Institute for Astrophysics Potsdam (AIP); email: kriebe@aip.de

¹<https://www.cosmosim.org/cms/documentation/demos-and-tutorials/first-steps-with-cosmosim/>

2. Registration:

- For gaining access to all simulations and full functionality of the web application, it is required to register at <https://www.cosmosim.org/auth/registration/register> beforehand. The registration is free and open to everyone. (Some data is public for guests as well, but functionality is limited.)
- During tutorials and hands-on sessions a demo user account may be enabled for testing purposes. Please ask your tutors for the current demo user's credentials.

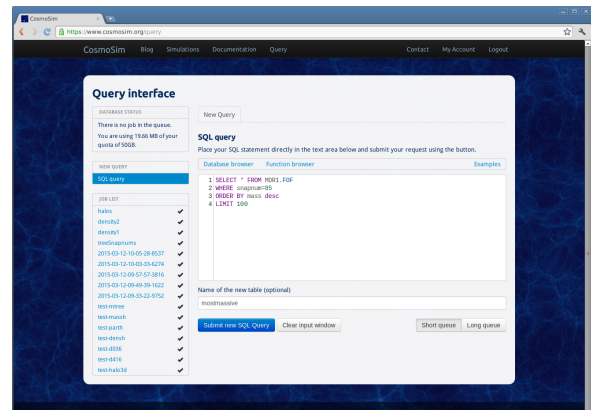
3. Query interface:

- Login with your username and password (or as the demo user/guest) and navigate to the **Query interface** (<https://www.cosmosim.org/query>).

- SQL queries are written directly into the main text area. The dialect used here is MySQL. Write the following text into the form:

```
SELECT * FROM MDR1.FOF
WHERE snapnum=85
ORDER BY mass DESC LIMIT 100
```

This query will retrieve the 100 most massive clusters (ordered by mass) for redshift 0 (= snapshot number 85) from the MDR1-simulation.



- Enter a new **name** for the result table below the SQL area. This name must be unique, if you leave the field empty, the web application will generate a unique name for you.
- Select **Short queue** and click on **Submit new SQL query**.
- Each query returns results in a (private) result table, the names of these tables are listed in the **Jobs list** on the left side. Click on your job to see its details.
- Once your job is finished (check-mark), click the job name again, then choose the tab **Results Table**. Browse the table to view the properties of the most massive halos.
- Above the form on the right side is the **Examples** link. Click on it to view the available example queries that you may also wish to try for yourself. Just double-click on one of them and the corresponding SQL query will be inserted into the SQL form.
- Browse the available simulations and tables by clicking at **Database Browser** on the left side above the query form.
- If you have long running queries, be sure to select **Long queue** before submitting, since the short queue times out after 30 seconds. It is also wise to restrict your query to use only the columns you are really interested in. This saves query time and disk space.
- In principle, a **basic SQL query** consists of the following parts:

SELECT	followed by a list of columns, e.g. x,Rvir,npart or * for all
FROM	followed by a list of tables (possible join)
Bolshoi.BDMV	database and table name
[WHERE]	followed by a list of restrictions (filter), e.g. snapnum=85
[ORDER]	followed by a (list of) columns, e.g. BY MASS DESC
[LIMIT]	followed by a number to restrict the number of returned rows

4. Consult the **documentation** (Database Structure → Tables) or the database browser above the form to get to know which table provides which data.

2 Mass function for FOF groups

In this chapter we plot the mass function for dark matter clusters of a simulation and learn about the **SAMP** interface to plot the results with **Topcat**.

1. Pick a simulation and redshift

- Open <https://www.cosmosim.org> in a web browser. Login and go to the **Query** form.
- Choose a simulation from the **Database Browser** above the query form, e.g. the **MDR1** simulation.
- Halo tables contain the column `snapnum`, which can be mapped to redshift via the `Redshifts`-table of the corresponding simulation database. For getting the `snapnum` for redshift $z = 0$, put the following query into the query form:

```
SELECT DISTINCT zred, snapnum FROM MDR1.Redshifts
WHERE zred<0.01
ORDER BY zred
```

- Give the result table a **name**, e.g. `mdr1-redshifts`, and **Submit** your query.
- Once the query is finished (check-mark symbol in the job list), click on the result name in the **job list** and choose the **Result Table** tab. The snapshot number (`snapnum`) for $z = 0$ is 85 for this simulation. This value is needed in the next step.

2. Get the mass function

- A mass function gives the number of halos for a selected mass bin. To get this, click the **SQL query** link above the job list to switch back to the interface and enter this new query:

```
SELECT 0.5+FLOOR(LOG10(mass)) AS logmass,
COUNT(*) AS num
FROM MDR1.FOF
WHERE snapnum=85
GROUP BY FLOOR(LOG10(mass))
ORDER BY logmass
```

This uses the FOF table of the MDR1 simulation, which contains dark matter clusters found with a Friends-of-Friends algorithm. The clusters are sorted into logarithmic mass bins using the `FLOOR`-function. The combination of `COUNT` and `GROUP BY` allows to count the number of clusters per bin.




- **Submit** the query and view the **Results Table** after it finished.
- Refine the bins by using steps of 0.25:

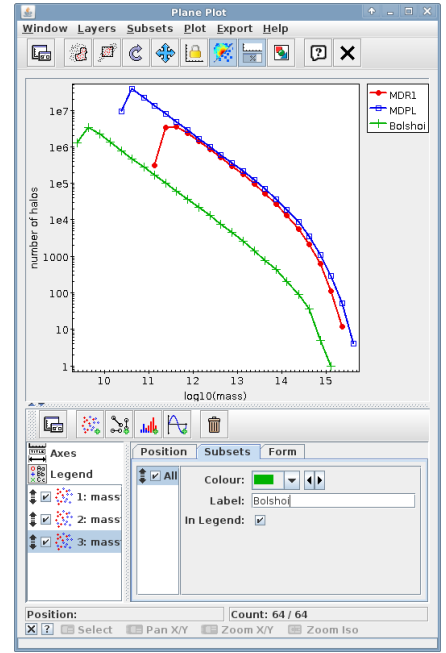
```
SELECT 0.25*(0.5+FLOOR(LOG10(mass)/0.25)) AS logmass,
COUNT(*) AS num
FROM MDR1.FOF
WHERE snapnum=85
GROUP BY FLOOR(LOG10(mass)/0.25)
ORDER BY logmass
```

Submit again and check the **Results Table**: There are more rows now (factor 4).

3. View the result

- **Start** [TOPCAT](#), e.g. using `java -jar topcat-*.jar` from the command line in Linux.
- Go back to the browser window, choose the **SAMP** tab of your latest result and click **Register with a local SAMP Hub**. You may need to add a security exception in your browser first (see instructions on web page). Authorize the SAMP connection with "Yes" in the pop-up window. If you cannot see the SAMP-link, be sure that you are logged in. Guest users cannot use the SAMP interface. If you don't have an account or SAMP does not work, you can still download the table via the [Download](#) tab (choose e.g. ASCII VO-table) and open the file in Topcat manually.

- Click **Send table to topcat** in the Topcat-row. This sends the result table to Topcat; the table name should appear there at **Tables**.
 - In **Topcat**, choose the **Plane Plot** . This opens a new window. In the **Position** tab, choose following axes: **X: logmass**, **Y: num**.
 - Select the **Form** tab next to **Position** and **Subsets** and click the button **Add new Line form** . This connects the points.
 - Choose **Axes** in the left side to get access to the axes-properties. At tab **Coords** (that's the default) tick **Y Log**. This makes the y-axis logarithmic. This is your mass function of FOF dark matter halos for the MDR1 simulation!
4. Experiment with mass functions for other simulations: How does the MDR1-mass function compare to MDPL or Bolshoi at the same redshift? How do mass functions change with the halo catalogue (use BDMV instead of FOF with Mvir for the mass) and redshift? Here are some tips:
- Take care to use the correct snapnum for each simulation to get the same redshift for each of them (e.g. from Redshifts-table: MDR1: 85, Bolshoi: 416, MDPL2: 125 for $z_{\text{red}}=0$).
 - In order to plot two tables in one **Plane Plot** window in Topcat, click the button **Add new positional plot control** . Then select a table and enter x- and y-axis.
 - For adjusting the main color and label for each table, select a table at the left side, then choose the **Subsets**-tab. Here you can enter your label text at **Label** and choose the **Color**.



3 Particles of a dark matter halo


For a few simulations, CosmoSim also provides the snapshots of selected timesteps. Thus it is possible to extract particles for a given position or a halo. In this section, the particles of a halo from the FOF-catalogue of the MDR1-simulation will be extracted and plotted with Topcat. Please note, that we use here the term *halo*, though actually, the FOF catalogue returns only groups of connected particles and makes no statement if they are truly gravitationally bound or not. Thus it would be more correct to speak of *FOF groups* instead, but let's keep it simple.

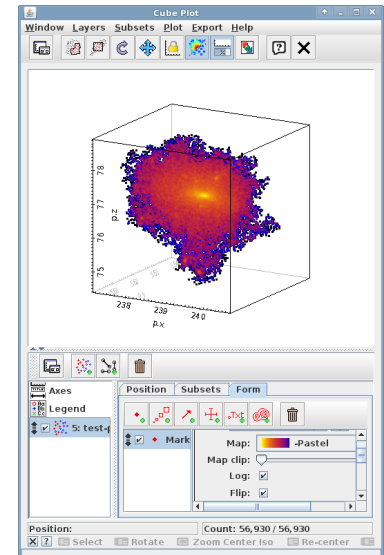
3.1 Extracting the particles

- Open <https://www.cosmosim.org> in a web browser. Login and go to the **Query** interface.
- Enter following query:

```
SELECT p.x,p.y,p.z
FROM
  MDR1.Particles85 AS p,
  (SELECT particleId from MDR1.FOFParticles
   WHERE fofId = 85000001014) AS fp
WHERE fp.particleId = p.particleId
```

This query first selects the ids of all particles for the FOF-halo 85000001014 and then uses them to extract their positions from the snapshot stored in table MDR1.Particles85.

- **Submit** the query.
- When the query is finished, **click on the job** in the **Job list** and go to the **Plot-tab**. Choose **p.x** and **p.y** as X and Y axis and click **Create Plot**. This plots the particle positions in x-y-plane and is a nice simple feature for getting a first impression of the results.
- Make sure that **Topcat** is still running.
- In the browser, switch to the **Results Table-tab**. Scroll down below the table. You should still be registered with SAMP – if not, then click again [Register with a local SAMP Hub](#). Click **Send table to Topcat**.
- In Topcat, select the new table, then choose **Graphics** → **Cube Plot** . This opens a new window for **3D plotting**.
- Within the **Cube Plot** window of Topcat, choose following axes: **X: p.x**, **Y: p.y** and **Z: p.z**
There are so many particles, that the true shape is somewhat obscured. Fortunately, since Topcat version 4, the dots can be colored by projected density: choose the **Form-tab** (next to **Positions** and **Subsets**) and select at **Shading** the **Mode: density**. Select a colormap to your liking as well.
- You can navigate round the 3D plot using the mouse: drag to rotate, use the mouse wheel to zoom in and out, and right-click (or CTRL-click) to re-center. More help on navigation is available by clicking the small question mark at the bottom left of the window.
- If you look close enough, you can even detect small lumps of higher density that could represent small subhalos. If you are further interested in them, have a look at the next section.




3.2 Searching for substructures [extra]

Curious about substructures? Let's take the example halo from the previous section and search for subgroups.



- Go again to CosmoSim's **Query** interface and enter following query:

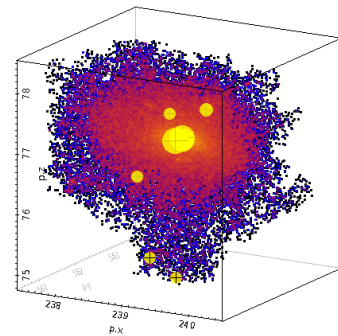
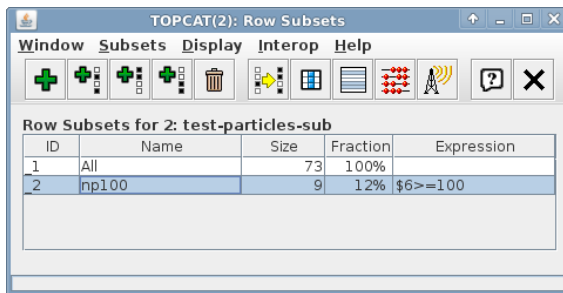
```
SELECT s.fofId, s.x,s.y,s.z, s.np
FROM MDR1.FOFSub AS s,
(SELECT fofSubId, lastSubId from MDR1.FOFSub WHERE fofId = 85000001014) AS si
WHERE s.fofSubId between si.fofSubId and si.lastSubId
```

This query uses the FOFSub-table which contains the subhalo relations of the FOF groups, i.e. which group is contained in another one. We first need the `fofSubId` of our halo and its `lastSubId`, which are extracted with the subquery. See the documentation on substructures at the CosmoSim website² to learn more about these identifiers. They are constructed such that the query above retrieves all the subhalos (actually: subgroups) of the given halo.


- **Submit** the query. After the job is finished, **send the table to Topcat via SAMP** (at **SAMP**-tab).
- In **Topcat's Cube Plot**-window of the previous section, add the subgroup-table by clicking the **Add new positional plot control**-icon .
- Select the new table and choose following axes: **X: p.x**, **Y: p.y** and **Z: p.z**.

This plots the centers of all retrieved subgroups. Note that these are only groups of particles which are close together (closer than a given linking length), there was no bounding-criterion applied. This means that these subgroups are not necessarily true *subhalos*.


- There are much more subgroup centers marked than were detectable by eye in the density plot, so let's get rid of the very small ones. We could rerun the query with an additional filter like `WHERE np > 100`, but we can also **filter results in Topcat**. So let's do this:
 - Select Topcat's main window, choose the subgroups-table and open the **Row Subsets**-window (**Views** → **Row Subsets**, ). Click on the button with the green plus sign  to define a new row subset. Enter a **Subset Name**, e.g. `np100` and an algebraic **Expression**: `$6<=100`. This will filter column \$6, which happens to be the number of particles `s.np` and put those rows with obey `$6<=100` into a new subset.



- In the **Cube Plot** window, choose the subgroups-table, then switch to the **Subsets** tab. Check `np100` instead of `All`. There are much less subgroup centers marked now.
- Let's use one more new feature of Topcat: **scale the subgroup-marks!**

Select the subgroups-table and switch to the **Form**-tab. Add a new size form  and choose for **Coordinates**, **Size: s.np** or **ln(\$6)**. This scales the marker size with the number of particles.

²<http://www.cosmosim.org/cms/documentation/database-structure/substructures/>

- If you want to get more information on a particular subgroup, then first go to Topcat's main window, choose the subgroup-table and open the table browser . This opens the **Table Browser** window which shows the complete result table.

Now find your preferred subgroup in the 3D plot, hide the particle distribution there (just uncheck the particle-table below the image) and click on the subgroup of interest. This highlights the corresponding row in the **Table Browser**. You can now use the `fofId` given there to extract all information about this subgroup from the corresponding FOF-catalogue, e.g. with a query like this in CosmoSim's query interface:

```
SELECT * FROM MDR1.FOF2
WHERE fofId=85200391320
```

In this example, we used `fofId=85200391320`, the digit right after the snapnum is 2 in our case, so we need to look into the FOF2-table. Just replace it with your own choice and explore the properties.

3.3 Extract particles of a BDM halo [extra]

There are no links between BDM halos and their particles stored in the database. But one can search for particles within a given distance from the halo's center. Let's start with finding the BDM halo that matches to the FOF group of the previous sections.

1. Find BDM halo for given FOF group:

- Go to the CosmoSim **query interface** and enter following query:

```
SELECT b.bdmId, b.hostFlag, b.x,b.y,b.z, b.Mvir, b.Rvir, b.np
FROM MDR1.BDMV b,
    (SELECT fofId, x,y,z, mass, size, np FROM MDR1.FOF
     WHERE fofId = 85000001014) AS f
WHERE SQRT( POWER(f.x-b.x,2)+POWER(f.y-b.y,2)+POWER(f.z-b.z,2) ) < 0.2
      AND snapnum=85
ORDER BY b.np
```

This query first selects the position of our FOF group and then joins this with the BDM-table based on matching positions. We choose $< 0.2 \text{ Mpc}/h$ here.

- **Submit** the query and after your job has finished, view the **Results Table**. It contains two BDM halos: the first one is only a subhalo, but the second one is a distinct halo with the appropriate mass range. Let's take this halo then and remember its id (`bdmId = 8501268794`).

2. Get particles for the BDM halo:

- Enter following query into the **query form**:

```
SELECT p.x AS x, p.y AS y, p.z AS z, p.particleId AS particleId
FROM MDR1.Particles85 AS p,
    (SELECT x,y,z, Rvir FROM MDR1.BDMV WHERE bdmId = 8501268794) AS b
WHERE SQRT( POWER(b.x-p.x,2)+POWER(b.y-p.y,2)+POWER(b.z-p.z,2) ) < 1.*b.Rvir
```

This extracts all particles with a distance smaller than the virial radius to the center of the BDM halo. For the MDR1-simulation, positions and radius have the same unit, for other simulations please first check the units via the online documentation and adjust the factor before `b.Rvir` where appropriate.

- **Submit** the query and after your job has finished, view the **Results Table**. You can view the particle distribution in Topcat by again switching to the **SAMP**-tab, clicking **Send table to topcat** and plotting the data in the same way as in the previous section. You will see now a spherical particle cut-out of the same region as before.


4 Merger tree of a halo

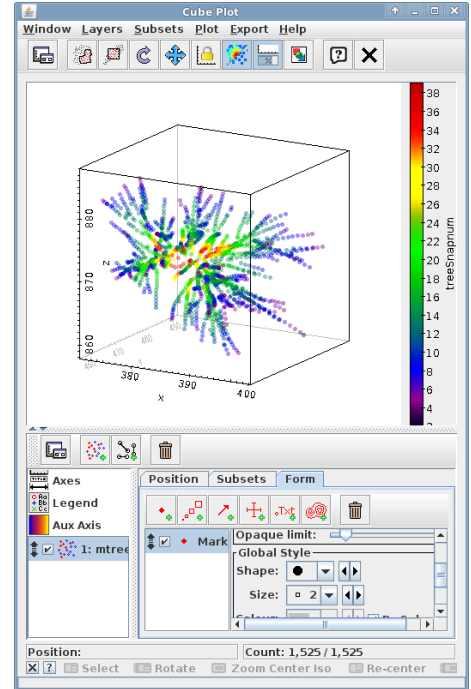
The merging history of halos is stored using the same database structure as was used for the Millennium database. This includes assigning unique identifiers to each halo of the merger tree, sorting them in a depth-first order and by the most massive progenitor. In this way, it is quite straightforward to retrieve all progenitors of a halo or just its main branch, as shown below. More details are given in the documentation of the CosmoSim website³.

- Open <https://www.cosmosim.org> in a web browser. Login and go to the **Query** interface.
- Enter following query:

```
SELECT x,y,z,treeSnapnum FROM MDR1.FOFMtree
WHERE fofTreeId BETWEEN 100000000 AND
(SELECT lastProgId FROM MDR1.FOFMtree
WHERE fofTreeId = 100000000)
AND np>200
```

This query first selects the `lastProgId` for the given halo in order to extract the positions of all the progenitors which consist of more than 200 particles.

- Enter a result table name (e.g. `mtree`) and **Submit** the query.
- When the query is finished, **click on the job** in the **Jobs** list and go to the **SAMP-tab**. Scroll down below the table. If you are not still registered with SAMP, make sure that Tocat is open, then click again [Register with a local SAMP Hub](#). Otherwise you can directly click **Send table to topcat**.
- In Topcat, go to **Graphics** → **Cube Plot** .
- Within the **Cube Plot** window, select following axes: **X**: `x`, **Y**: `y` and **Z**: `z`.
- Let's apply more color! Switch to the **Form**-tab, at **Shading**, choose **Mode**: `aux` and enter for **Aux**: `treeSnapnum`. The `treeSnapnum` is a snapshot number, i.e. it corresponds to a time axis. Halos from the beginning of the simulation are colored black-purple, more recent progenitors have yellow-red colors. Adjust the **Opaque limit** and **Shape/Size** to your liking, your plot should look similar to the figure here. This nicely shows how the progenitor halos have merged with each other at earlier times to form the final halo (red) at the center.



³<https://www.cosmosim.org/cms/documentation/database-structure/merger-trees/>

5 The Cosmic Web with TOPCAT

5.1 Load a slice (x-y plane) of the cosmic web for the Bolshoi simulation

- Before actually downloading some data, let's first check where the **most massive halo** is located, using the **BDMV halo** catalogue: Enter following query into the **Query Form** at CosmoSim⁴:

```
SELECT ix,iy,iz,x,y,z,Mvir FROM Bolshoi.BDMV
WHERE snapnum=416 ORDER BY Mvir DESC LIMIT 1
```

This selects some properties of the most massive halo for snapshot number 416 (corresponds to $z = 0$ for this simulation).



- **Submit** the query.
- When the query is finished, **click on the job** in the **Jobs** list and switch to the **Results Table** tab. We can see now the position and mass Mvir of the halo as well as the indices of a 1024^3 grid cell, in which the halo sits (ix,iy,iz). The size of the grid is given in the documentation.
- One can **convert these numbers** into the corresponding indices of a coarser grid by dividing them by **factors of 2**. E.g. a grid cell with ix=200 in a 1024^3 grid corresponds to $ix' = \text{floor}(ix/2.) = 100$ in a 512^3 grid, and $ix'' = 50$ for a 256^3 grid.
- Switch back to the **query interface** and enter a query to extract a slice of the cosmic web at the location of the most massive BDM halo:

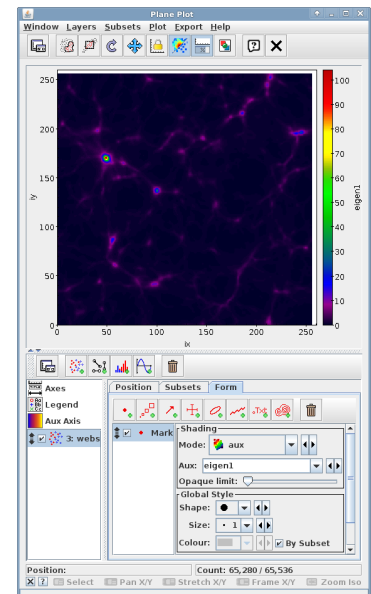
```
SELECT ix,iy,iz,phkey,eigen1,eigen2,eigen3 FROM Bolshoi.Tweb256 WHERE iz = floor
(377/4.)
```

We use here the table Tweb256, which contains a 256^3 grid, and for each grid cell the eigenvalues and eigenvectors for the gravitational tidal tensor. To match with the z -position of the most massive halo we need the conversion factor due to the different grid sizes, as explained above. Since we only fix the iz-value, we will get a 256×256 slice in x-y plane returned.

- **Submit** to submit the query and after it finished switch to the **SAMP** tab. Make sure that **Topcat** is still open before clicking **Send table to topcat**.

5.2 View the cosmic web

- Have a look at the data in Topcat with the **Table Browser**  which you can access from [TOPCAT](#)'s main window. It displays your data as a table. The column names have also been imported.
- **View your data** by clicking on the **Plane Plot** icon . Choose: X: ix, Y: iy
- Add some **color** by choosing the **Form** tab and at **Shading** select **Mode: aux**. Set **Aux** to one of the eigenvalues, e.g. **eigen1**, for coloring the dots. Adjust the point size at **Global Style**, **Size** to be at least 1.
- Step through **different columns for Aux** to get an impression of the other eigenvalues as well. Also look at the phkey-column: These are Peano-Hilbert keys, describing a space-filling curve. They are very useful for spatial queries.





⁴<http://www.cosmosim.org/query>

- Just visualising one of the eigenvalues does not yet give much information. Instead, we need to count the number of eigenvalues above a certain threshold. This number will allow us to assign each cell with following classification:

- 0 = cosmic void
- 1 = sheet-like structure
- 2 = filament
- 3 = knot (= location of galaxy clusters)

- We can get this number by adding a **computed column** to our data in Topcat:

- In Topcat’s main window, click on **Display Column Metadata** 
- Click on the green plus  to add a new column to your table.
- In the new dialogue, enter a **name** for your new column, e.g. **num_th0.4**
- Enter the **expression** to compute the column:

```
(eigen1$>=$0.4 ? 1:0) + (eigen2$>=$0.4 ? 1:0) + (eigen3$>=$0.4 ? 1:0)
```

This expression checks, if an eigenvalue is greater than or equal to the threshold value (e.g. 0.4) and adds 1, if this is the case, otherwise 0.

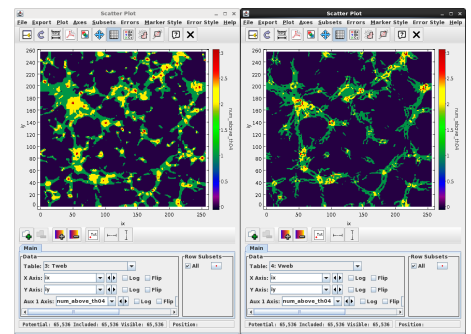
- Click OK. You can still edit the name and expression of your column by double clicking on it in the **Column Metadata** dialogue.
- Switch back to the **Scatter Plot Window** and choose the new computed column as **Aux** axis.
 - Now you should see some clustered structures (red knots) and filaments (green). Note that this is just a cut through the z -direction, so perpendicular sheets appear as filaments and filaments may appear as knots.
 - By the way: instead of calculating the additional column with Topcat, you could already do it within the SQL query itself:

```
SELECT ix,iy,iz,phkey,eigen1,eigen2,eigen3,
(CASE WHEN eigen1 >= 0.4 THEN 1 ELSE 0 END)
+ (CASE WHEN eigen2 >= 0.4 THEN 1 ELSE 0 END)
+ (CASE WHEN eigen3 >= 0.4 THEN 1 ELSE 0 END)
AS num
FROM Bolshoi.Tweb256
WHERE iz = floor(377/4.)
```

This method only has the disadvantage that you would need to query the database again for each new threshold value that you are testing.

5.3 Explore more data [extra]

1. Repeat the **same steps** for loading a slice of the same region from the **Vweb256** table. **Compare the differences:** the Vweb should give a much better resolution of structures than the Tweb. (Look in the documentation at the webpage and the linked papers to learn about the physical reasons behind this.)
2. Experiment with **different threshold values** by adding more computed columns.



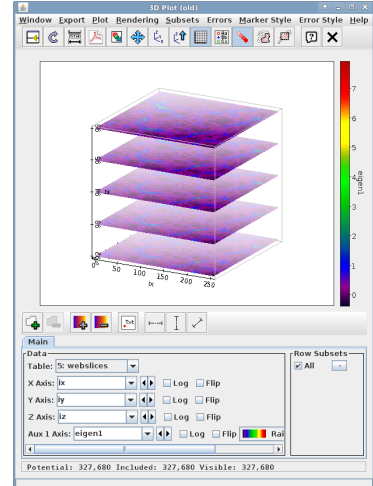
- Experiment with **slices at different positions**. You can also download several z -slices and plot them in 3D.


For example, do following query:

```
SELECT ix,iy,iz,phkey,eigen1,eigen2,eigen3
FROM Bolshoi.Vweb256
WHERE iz BETWEEN 92 AND 96
```

and send the result to Topcat. We will use the old 3D plot feature of Topcat now, because for this case it gives a nicer visual impression.

- Choose **Graphics** → **3D Plot (old)** and adjust the columns: **X: ix, Y: iy, Z: iz**.



- Add a new auxiliary axis by clicking the icon . Set **Aux 1** to one of the eigenvalues or add again the sum of eigenvalues above a given threshold value to color by this value.

- Load a **finer grid** by using the **Vweb512** table and compare with the coarser version. Note, that you need to **convert the iz-value properly** if you want to get matching slices:

```
SELECT ix,iy,iz,phkey,eigen1,eigen2,eigen3
FROM Bolshoi.Vweb512
WHERE iz = 188
```


If your job times out, select **Long queue** below the query form before submission.

5.4 Find halos in cells

- In Topcat, open the **Table browser** and a **Plane plot window** for the same data table (e.g. for Vweb256, from our first cosmic web query).
- Click on one of the knots** in your plot and check to which cell it belongs: the corresponding **row** in the table browser is **highlighted** now.
- Go back to CosmoSim's query interface in the web browser and enter following query to retrieve all halos inside a given cell, sorted by mass, e.g.:

```
SELECT bdmId, x,y,z, Mvir, spin FROM Bolshoi.BDMV
WHERE snapnum=416
AND FLOOR(0.25*ix) = 101
AND FLOOR(0.25*iy) = 138
AND FLOOR(0.25*iz) = 94
ORDER BY Mvir DESC
```

Replace **101**, **138** and **94** by the values for your cell.

- Submit** the query and send the results to Topcat.
- In Topcat, select the previous **Scatter plot window** and click on the icon for adding a new data set . Choose the halos-table you just downloaded and set the axes: **X: x, Y: y**.
- Adjust the point size and color in the **Form** or **Subsets** tab, if needed. You can also adjust the size of points depending on the halo's mass.
- Zoom** into the cell where the halos are located using the mouse-wheel. Check which is the most massive halo in that region and if it lies at the center of the knot (as expected) or farther away.

6 Command-line access via UWS

The web application behind CosmoSim also provides job handling via UWS, which allows users to script the submission and deletion of jobs. This is a very useful feature for long running jobs (e.g. when downloading particles for many halos) which need to be split up into smaller jobs due to the time limit per query.

In order to talk to the web application, one needs to send POST/GET request to a specified URL. This can be done with different tools. We're going to show three different tools and how to do the most basic operations: create/submit a job, get the job's status/info, get the job list, retrieve results and delete a job. Since it is all directed from the command line, this makes it easy to integrate results from the CosmoSim database into your work environment.

Please note that UWS only works for registered users, if you didn't register yet, please get an account (for free!) at the CosmoSim webpage.

The basic URL for CosmoSim's UWS service is: <https://www.cosmosim.org/uws/query>.

6.1 UWS with httpie

If you don't have `httpie` installed, download it from <https://github.com/jkbr/httpie> and install it. Please replace `<username>` and `<password>` in the examples below with your own credentials.

1. Create and submit a job

Write following lines in the terminal to **create a job** for retrieving the particles of a given FOF group, like in section 2. We will limit the result here to 20 rows for now.

```
http --auth <username>:<password> --form --follow POST \
http://www.cosmosim.org/uws/query \
query="SELECT p.x,p.y,p.z FROM MDR1.Particles85 AS p, (SELECT particleId \
from MDR1.FOFParticles WHERE fofId = 85000001014) AS fp WHERE \
fp.particleId = p.particleId LIMIT 20" \
table="demoparticles" queue="short"
```

The tablename must be unique, you can also omit this parameter and a unique tablename will be created automatically. When you send this, you will get a response in xml containing the job description. The job is now in **pending** phase and still waiting for you to really start it. Search for the **jobid** in the output. This jobid is needed to refer to this job in the next steps. Be sure to replace the example jobid below with your own value.

Submit the job by sending `phase=run` for this job identified by its jobid from the previous step.

```
http --auth <username>:<password> --form --follow POST \
http://www.cosmosim.org/uws/query/360912133673751 phase=run
```

2. Get the job's details

Check the **job phase**:

```
\begin{verbatim}
http --auth <username>:<password> --print b GET \
http://www.cosmosim.org/uws/query/360912133673751/phase
\end{verbatim}
```

This should return the phase **QUEUED**, **EXECUTING** or **COMPLETED**, if nothing went wrong.

Get the complete **job info**:

```
http --auth <username>:<password> --print b GET \
http://www.cosmosim.org/uws/query/360912133673751
```

When the job is finished (COMPLETED phase), you can get information about its `result url` like this:

```
http --auth <username>:<password> --print b GET \
http://www.cosmosim.org/uws/query/360912133673751/results
```

Actually, this outputs only that part of the complete job info from above that contains information about the result. Look here for the tag `<uws:result ...>`. The url at `xlink:href` given here (it contains the tablename, if you provided one) can be used to retrieve the results in the next step.

3. Retrieve the results

Download the results directly to the terminal:

```
\begin{verbatim}
http --auth <username>:<password> --print b GET \
http://www.cosmosim.org/query/download/stream/table/demoparticles/format/vodump-csv
\end{verbatim}
```

Or download your results into a file `</path/file.csv>`:

```
\begin{verbatim}
http --auth <username>:<password> --download --output </path/file.csv> GET \
http://www.cosmosim.org/query/download/stream/table/demoparticles/format/vodump-csv
\end{verbatim}
```

If you prefer votables, that's also supported. Just change the format into votable:

```
http --auth <username>:<password> --download --output </path/file.csv> GET \
http://www.cosmosim.org/query/download/stream/table/demoparticles/format/votable
```

4. List all jobs

```
http --auth <username>:<password> --print b GET http://www.cosmosim.org/uws/query
```

This prints all your jobs to the terminal.

5. Delete a job

```
http --auth <username>:<password> --follow DELETE \
http://www.cosmosim.org/uws/query/360912133673751
```

This deletes the job identified by its jobid. The details of the job may still be archived on the server, but the results will definitely be deleted.

6.2 Using the Python UWS client

Download the UWS-client from GitHub: <https://github.com/adrpar/uws-client> and install it. We will use here the script `uws-client/bin/uws` and short-cut it as `uws` in the commands below.

1. Create and submit a job

Write following lines in the terminal to **create a job** for retrieving the particles of a given FOF group, like in section 2. We will limit the result here to 20 rows for now.

```
uws --host www.cosmosim.org/uws/query --user <username> --password <password> \
job new \
query="SELECT p.x,p.y,p.z FROM MDR1.Particles85 AS p, (SELECT particleId \
from MDR1.FOFParticles WHERE fofId = 85000001014) AS fp WHERE \
fp.particleId = p.particleId LIMIT 20" \
table="demoparticles2" queue="short"
```

Again the tablename must be unique, you can also omit this parameter and a unique tablename will be created automatically. After sending this, you'll get some job information back, including the job id to refer to this job in the next steps. Be sure to replace the example jobid below with your own value.

Submit the job identified by its jobid from the previous step.

```
uws --host www.cosmosim.org/uws/query --user <username> --password <password> \  
job run 360912697344392
```

2. Get the job's details

```
uws --host www.cosmosim.org/uws/query --user <username> --password <password> \  
job show 360912697344392
```

The phase of your job should be **QUEUED**, **EXECUTING** or **COMPLETED**, if nothing went wrong.

The **result-url** is given in the last row. You can use this directly (e.g. with `wget` or in a browser) or follow the next step.

3. Retrieve the results

Download the results in a file directly into the directory from which you call `uws`:

```
\begin{verbatim}  
uws --host www.cosmosim.org/uws/query --user <username> --password <password> \  
job results 360912697344392  
\end{verbatim}
```

4. List all jobs

```
uws --host www.cosmosim.org/uws/query --user <username> --password <password> \  
list
```

This prints all your jobs to the terminal in a nicely formatted list. You can even filter by phase by adding one of the following optional arguments: `--pending`, `--queued`, `--executing`, `--completed`, `--error`, `--aborted`. E.g. for listing only pending jobs, use:

```
uws --host www.cosmosim.org/uws/query --user <username> --password <password> \  
list --pending
```

5. Delete a job

```
uws --host www.cosmosim.org/uws/query --user <username> --password <password> \  
job delete 360912697344392
```

This deletes the job identified by its jobid. The details of the job may still be archived on the server, but the results will definitely be deleted.

6.3 Querying with `astroquery.cosmosim`

Download `astroquery` from <https://github.com/astropy/astroquery> or install it via `pip install astroquery`. Since the `cosmosim` package is still under heavy development, we refer to the online documentation at <http://astroquery.readthedocs.org/en/latest/cosmosim/cosmosim.html> for the moment. You can also checkout CosmoSim's documentation on UWS access at <http://www.cosmosim.org/cms/documentation/data-access/access-via-uws/>.