CTA200H Mini-Project: MW Analogues in cosmological simulations

Advisors: Josh Speagle & Alicia Savelli
Adapted from a project by Ted Mackereth

May 5, 2023

This introductory exercise is intended to take you through the initial steps to accessing data from the EAGLE simulations and begin isolating galaxies from the simulation that are similar to the Milky Way. To do this, we will use the database of galaxy properties which can be accessed via http://icc.dur.ac.uk/Eagle/database.php. For this you will need to sign up for a VirgoDB account here: http://virgodb.dur.ac.uk (the registration should take no longer than 24hr, but needs human intervention, so please do this ASAP).

Through the course of the exercise, we will learn how to make SQL queries, run them and examine their output. Furthermore, we will make some figures using `matplotlib` in python. Finally, we will find out how to select subsets of the data and make some further analysis of these subsets. As we go through these steps, we will try to figure out some details of the simulations and how galaxies are represented in them.

1 Retrieving EAGLE Galaxy properties: your first SQL query in python

1. First you need to install the python based tools for using the EAGLE SQL database. You can find those here: https://github.com/kyleaoman/eagleSqlTools. There are many ways to install python packages, but the simplest is usually to type '`pip install eaglesqltools`' at the command line (of course, other packages can be installed by changing the package name). Depending on where you are installing, you might need to add '`--user`' at the end of the command.

2. The next step is to either open up a jupyter notebook or start writing a python script that will run your first SQL query on the EAGLE database, and return some data for all of the galaxies in the final output of the largest volume simulation (100 Mpc on a side). To do this you will need to import the eagleSqlTools module, and enter your database username and password as a variable. Then you will initialise the connection to the database. This is achieved like this:

```
import eagleSqltools as sql

username = #add your username here
password = #add your password here

con = sql.connect(username, password=password)
```

now you have a connection, you can continue on to write a query. In python, we write the queries in an object called a docstring, which is essentially just a multi-line string:

```
query="""SELECT MassType_Star as stellar_mass,
                StarFormationRate as SFR
     FROM RefL0100N1504_SubHalo
     WHERE SnapNum = 28"""
```

Finally, we run the query (which can take a few seconds to a minute), and we store its output into a variable (the contents of which we will explore next):

```
data = sql.execute_query(con,query)
```

For this part of the exercise, it would be great if you not only copied and pasted the above code (either in your script or jupyter notebook), but also try to work out what the query above is actually doing – using the documentation and information available online at http://virgodb.dur.ac.uk:8080/Eagle/ and https://arxiv.org/pdf/1510.01320.pdf. Additional information about SQL queries and their syntax can be found here: https://www.w3schools.com/sql/. What information does the query retrieve from the simulation? What condition is placed on the query, and what does that mean for the eventual output? Could you have made the same constraint using a different column in the table?

3. Next, we'll inspect the data that was retrieved by making a simple plot of the information using `matplotlib`. You will have hopefully been introduced to `matplotlib`, but if not, their website (and google, as always) are great resources here. The columns that were retrieved are indexed using the names we gave in the query above, like:

   ```
   stellar_mass = data['stellar_mass']
   ```

   This is because the execute query function used above returns a numpy record array. Use this information to make a simple scatter plot of each of the retrieved columns against each other. You should notice that the data are very concentrated in the corner of the figure - you can alleviate this by switching the axes to logarithmic scales. You will also notice that the default data point size means that many points overlap - you can alleviate that by resizing them or changing their opacity - there are a number of ways to do this, so experiment a bit (you can also try a 2D histogram if you feel confident!).

4. Once you make the plot, you should notice some interesting features arising in the distribution. This figure is actually a fundamental diagnostic of galaxy formation which has had a huge number of observational studies devoted to it. Try to describe some of the features here and try to relate them to what you think this means for galaxies in the Universe (googling the term 'galaxy star formation main sequence' will help here! This 'astrobite' may also be useful!). There are also a number of features of this distribution that are very different to what we see in nature. What do you think these are? Why do you think they would arise in these simulations of galaxies? As a hint: the answer to this is one of the reasons we can run simulations of large volumes of the Universe on useful timescales. Can you see anything else in this figure that is unusual?

5. STRETCH GOAL: As a final (entirely optional) exercise in this part, we'll try and relate what we see in this figure back to an easily observable property of the galaxies - their colour (which is the difference between the brightness of the galaxy in two wavelength bands). To do that you will have to adjust the query above to cross-reference the main RefL0100N1504 SubHalo table with another one that contains model magnitudes for the galaxies: RefL0100N1504 Magnitudes. An example of such a match is given in 'Listing 2' on page 10 of this paper. Try to add the g and r band magnitude to the query above, then plot the Mass-SFR distribution again, this time colouring it by the $g-r$ colour (colour and magnitude are some unusual concepts at first, so please come to us if you need help). What do you notice about the colours of galaxies in different parts of the distribution? Can you write down one thing that you think that this might tell us about old vs young stars

2 Going Deeper into the Simulations: more properties and more snapshots

1. As you may have realised already, the simulation data in the database covers a number of different 'snapshots' of the simulation. These are a full dump of the current state of all the particles at any time. While we are looking at the overall properties of galaxies which are derived from these particles, these properties are computed at every snaphot as well. Above, we specified that we wanted to look at the final output (at a redshift of 0) by specifying `SnapNum = 28`. However, we can change this to look at a wider range of outputs. Try to change this so that we access all galaxies that are found in snapshots after a redshift of 0.5 (you will need the `Redshift` column in the database, and remember

that 'after' redshift 0.5 is a range of redshifts from $0 < z < 0.5$). Can you compare the number of galaxies between this output and that from Section 1? Write down 2 changes in the distributions when you remake the figure.

2. Selecting the galaxies in this way is more representative of what we would see in an observational galaxy survey looking out from the Milky Way. Can you explain why? What do you think determines the maximum redshift that a galaxy survey can see galaxies at? (this can be a very complex question but a simplistic answer here will do fine!)

3. STRETCH GOAL: Since these snapshots are just extracted from different time-steps of the simulation, by selecting multiple snapshots we will actually extract information from the same galaxy across different times. In the reference paper, they provide a figure (Fig. 2) that visually describes the ID numbers given to galaxies (in the `GalaxyID`, `DescendantID`, `LastProgID` and `TopLeafID` columns). This is what we refer to as a 'merger tree'. In our selection of galaxies from $0 < z < 0.1$, can you figure out a way to isolate the most massive galaxy at z = 0 and its direct (i.e. Main Branch) descendants? This is quite a tough one to get to grips with, so ask for help if you need it!

4. STRETCH GOAL: Finally here, for that same, most massive galaxy, can you isolate all of its descendants (in every branch)? By also taking the total mass (in stars, gas, black holes and dark matter) of each galaxy in this 'tree' - can you figure out a way to plot the total mass in galaxies in this tree at every time-step (redshift)? What about just the galaxies in the 'main branch'? What differences can you see between these two histories? If we defined a timescale which was the time at which these curves reached half the final mass – what are the differences?

3 Another Simulation: Querying data from IllustrisTNG

This exercise intends to take you through the steps of querying the database from a different set of simulations – the IllustrisTNG project. In order to access this data, you will need to sign up for the API, which you can do here: https://www.tng-project.org/users/register/ (unlike EAGLE, it should be fairly immediate!).

1. TNG doesn't use SQL like EAGLE; instead it uses an API (Application Programming Interface). To access the data, you will first need to set up a helper function, whose purpose is to make a HTTP GET request to a specified URL ("endpoint"), and verify that the response is successful.

```
import requests

baseUrl = 'http://www.tng-project.org/api/'
headers = {"api-key":"put_api_key_here"} #enter your own API key here

def get(path, params=None):
    # make HTTP GET request to path
    r = requests.get(path, params=params, headers=headers)

    # raise exception if response code is not HTTP SUCCESS (200)
    r.raise_for_status()

    if r.headers['content-type'] == 'application/json':
        return r.json() # parse json responses automatically
    return r
```

   The response will automatically be stored into a python dictionary. If dictionaries are new to you, this tutorial could be helpful (or just ask us!).

2. Now we want to look at all of the galaxies at redshift $z = 0$ in the TNG100-1 simulation. We have to issue a request to the API root using

```
z0url = 'http://www.tng-project.org/api/TNG100-1/snapshots/99/subhalos'
r = get(z0url)
```

(We can go over how to construct these urls by hand another time, but it is maybe slightly too complicated for this project). The response will be a dictionary object. To print out the keys, use

```
r.keys()
```

Explore the keys to find out how many galaxies there are at $z = 0$ in this simulation, and compare the number to the galaxies you pulled from EAGLE in Section 1. Store the number of galaxies under the variable name `n_gal`.

3. The initial data pull only grabs the first 100 galaxies. To get all of the galaxies, rerun the request using

```
subhalos = get(z0url, {'limit':n_gal})
```

4. Let's now make the same plot of the galaxy main sequence that we did in Sections 1 and 2. First inspect the available fields by querying the results of the first galaxy

```
get(subhalos['results'][0]['url'])
```

Then pull the fields of interest

```
# prepare dict to hold result arrays
fields = ['id', 'sfr', 'mass_stars']
r = {}
for field in fields:
    if field == 'id':
        r[field] = np.empty(n_gal, dtype = int)
    else:
        r[field] = np.empty(n_gal, dtype = float)

# fill in dictionary with results from each galaxy
for i in tqdm(range(n_gal)):
    sub_i = get(subhalos['results'][i]['url'])

    # save fields to dictionary
    for field in fields:
        r[field][i] = sub_i[field]
```

Use this data to reproduce the SFR vs SM plot. Do you see any differences between this plot and the same one made using the EAGLE data? What do you think is causing these differences? Specifically consider these plots in the context of the 'red and dead' galaxies you read about.

4 STRETCH GOAL: Isolating Milky Way-like galaxies in the galaxy color-magnitude diagram

This final part is very much a stretch – feel free to go as far as you would like, and definitely come to us with questions. It is *not* a requirement in the report.

1. For this final section, your task is to extract all central galaxies in the simulations that have a stellar mass $M_\star > 10^9$ $M_\odot$ at $z = 0$ (note that you will need to incorporate some more columns in the `WHERE` statement here). From this large set, you should isolate galaxies which have a stellar mass within at least three error-bars from a current estimate of the Milky Way stellar mass $M_\star = 6.08 \pm 1.14 \times 10^{10}$ $M_\odot$, and the star formation rate: $\dot{M}_\star = 1.65 \pm 0.19$ $M_\odot$ yr^{-1} (these values come from this paper). Finally, you should now plot a 'colour-magnitude diagram' of the large set of galaxies in g and $g-r$, overplotting the selection of galaxies with similar properties to the Milky Way in a different colour/size/marker-style (as a further stretch, you could try colouring these points by the 'distance' from the median MW value!). What do you notice based on this analysis? What do you think this tells us about the Milky Way as a galaxy?

2. Repeat Step 1 with data from TNG100-1 (you will have to edit the code to pull fields of interest to include photometric bands. How do the plots from the two different sims compare?