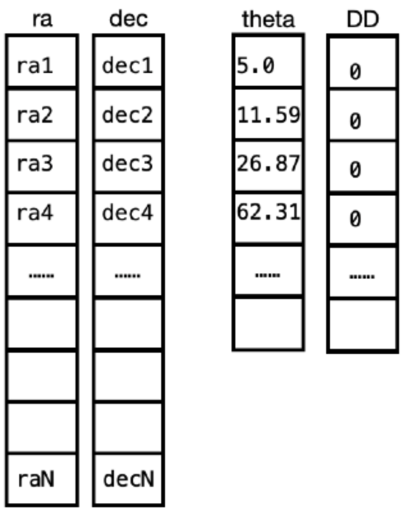
Workshop 2: Computing the clustering of a galaxy catalog (part 1)

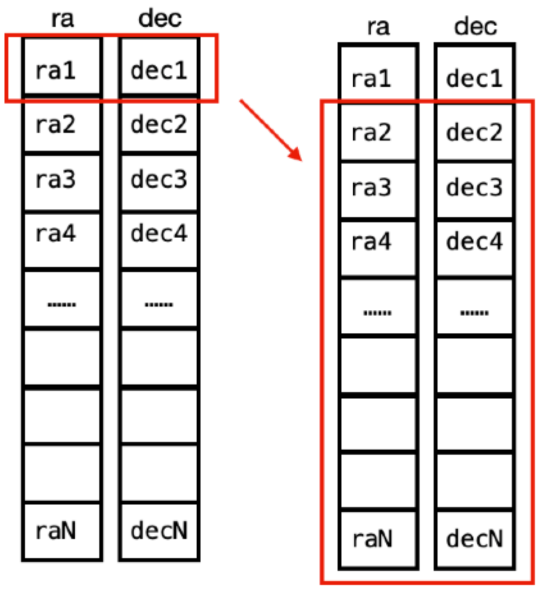
We will use the catalogs that you created yesterday to measure the angular clustering of the galaxies for the two samples. Since this require some amount of coding, we will work in the first part today and the second part tomorrow. If you were not successful creating the files with the samples of galaxies and the plots yesterday, you can download these from https://home.strw.leidenuniv.nl/~garcia/astrotwincolo/Workshop1/Results/

1. Make sure that you have the files low.dat, and high.dat in your working directory.
2. The first step we need to make is the creation of the random catalog. For that, first open the sky distribution plot that you created yesterday and check the minimum and maximum values of ra and dec for the galaxies in the samples.
3. Generate 10,000 random RA values and 10,000 random Dec values within the limits of the data. For that you can use a random generation in python (for example numpy.random.uniform()). Remember to use a "uniform" distribution. (Note: We normally should create a much larger random catalog to decrease Poisson error, but to save computational time, we will only create 10,000 sources for now). Save a file with the RA and Dec positions of your random catalog. (Call it random.dat).
4. Make a plot of the sky distribution of the random catalog. How does it look compared with the sky distribution of the data?
5. Define logarithmically spaced bins for which you would like to count galaxy pairs. For that you have to define a minimum value (for example 5 arcsecs), a maximum value (for example 200 arcsec), and a number of bins (for example 7). You can use the function in python np.logspace() to create the logarithmically spaced values. Store these values in an array, and call the array "theta_begin". Note that the three parameters used here (min, max and number of bins) are arbitrary, you can play with them and choose other values too. Print the values of theta_begin.
6. Compute the bin width and print their values.
7. Make a plot of theta_begin versus the bin width.
8. Now we will compute the array DD. This require a bit more of code then you can start this today, and if don't finish it we will continue tomorrow. Follow this steps:
   A. You need to create an array with the same length as the array "theta_begin". Call this array "DD"
   B. Choose one of your catalog of galaxies (high.dat or low.dat).
   C. Create a "for cycle" to go through each galaxy in the catalog.
   D. For each galaxy you need to compute the distance (in arcsec) with all the other galaxies. After to compute each distance, you need to check in which position of the array "theta" is that value located and increase "1" in that position of the array "DD" (see example below). Note that if you compute the distance between the galaxy 1 and 2, you don't have to compute the distance between the galaxy 2 and 1 anymore (you only have to count the pair once). Follow this example:

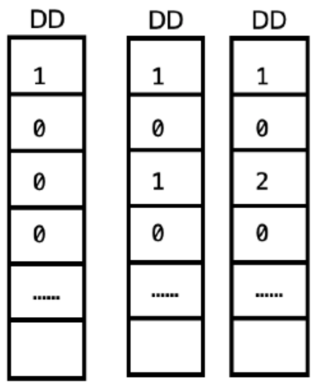
Asume you have the array “ra” and “dec” with the positions of the galaxies, the array “theta_begin” and the initial array “DD”:



You need to go through each position of the galaxies and compute the distance with all the rest. First pick the first galaxy and compute the distance between that one and all the rest:

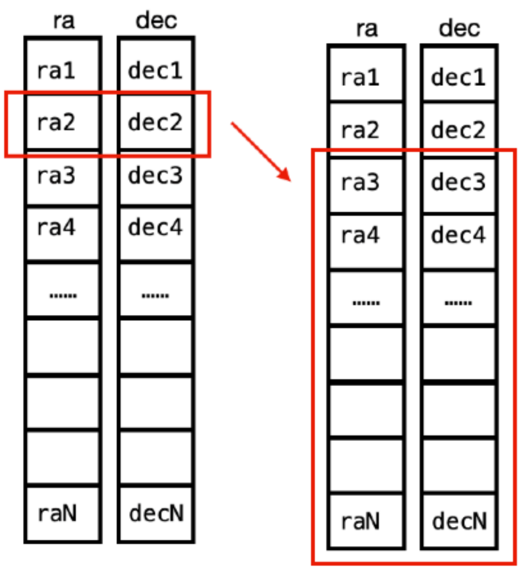


You will have N-1 distances. For each distance you need to check where is located in the “theta” array. For example if the first distance is 5.1 arcsec, then that correspond to the first position in the “theta_begin“ array, then you have to increment 1 in that position of the “DD” array. If the second distance is 60 arcsec you increment 1 in the third position of DD, if the third distance is 27 arcsec you increment 1 in the third position of DD, etc. After these 3 distances your DD array will look like:



You continue doing it for all the N-1 distances.

After that, you pick the second galaxy of the array and compute the distance with all the rest again, and repeat the same procedure:



Now you have N-2 new distances and you fill your DD array in the same way as before.

Note that in terms of code, you only need to create a for cycle with i ranging from galaxy 1 to N-1. Within this for cycle compute the distance between the galaxy “i” and the other galaxies (note that these are distances over an sphere, not in cartesians, then you can use a defined function in astropy for which you provide the array of positions and it return the array with the distances, for example SkyCoord.separation). If you are not familiar with those functions, just compute distances as in cartesian for now.

If you have the array of distances, you just need to check the corresponding bin in which the distance is located. The less efficient way to do this is to check with some “if the distance is between theta1 and theta2, then add 1 to the position 1”, etc. But there are more efficient ways to do this. For now, do whatever you think is better.

7. Make a log-log plot of theta_begin versus DD.