CHOA

Cosmology with Higher Order Astrostatistics

2021/3/5

Cristiano Sabiu
Se Yeon Hwang Su Mi Kim

University of Seoul

OutLine

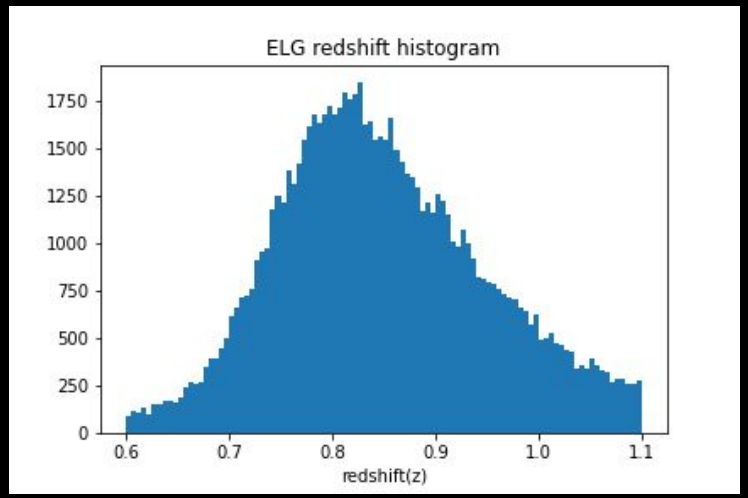
1. DATA(SDSS/simulation)
2. METHOD
3. GOAL

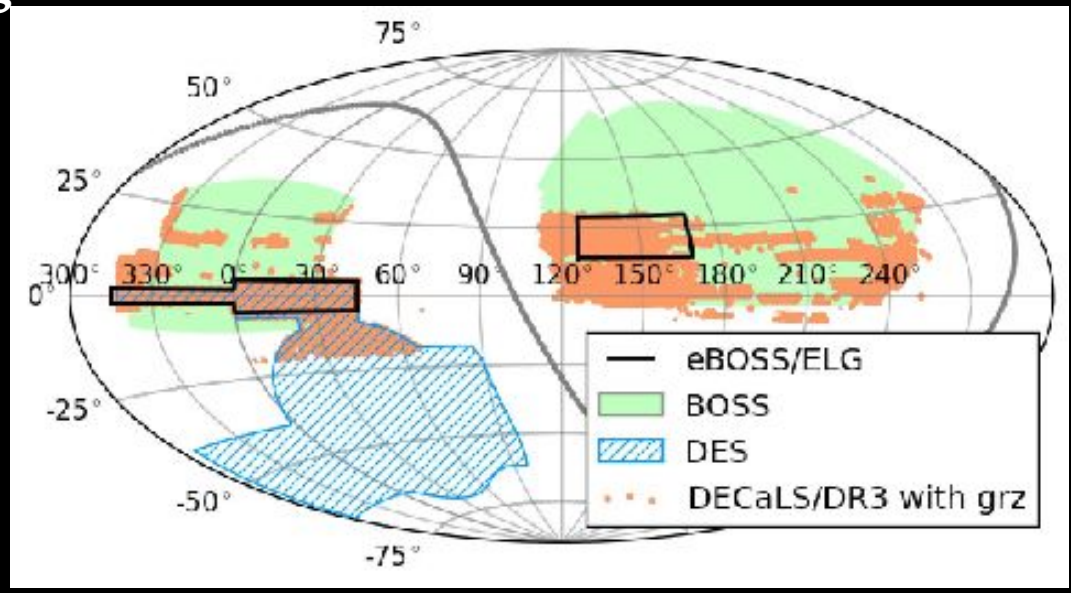
Real Data

The Extended Baryon Oscillation Spectroscopic Survey (eBOSS)

- **When combined with previous phases of SDSS, eBOSS precisely measures the expansion history of the Universe throughout eighty percent of cosmic history, back to when the Universe was less than three billion years old.**

- **Especially we use ELG data from eBOSS:**
  ELGs = emission line galaxies over the redshift range 0.6<z<1.1
  - Number of galaxy = 83,769
  - first we use NGC (Northern Galactic Cap) but later we also use SGC
  - the sky area of the NGC sample of ELGs is 200 deg^2

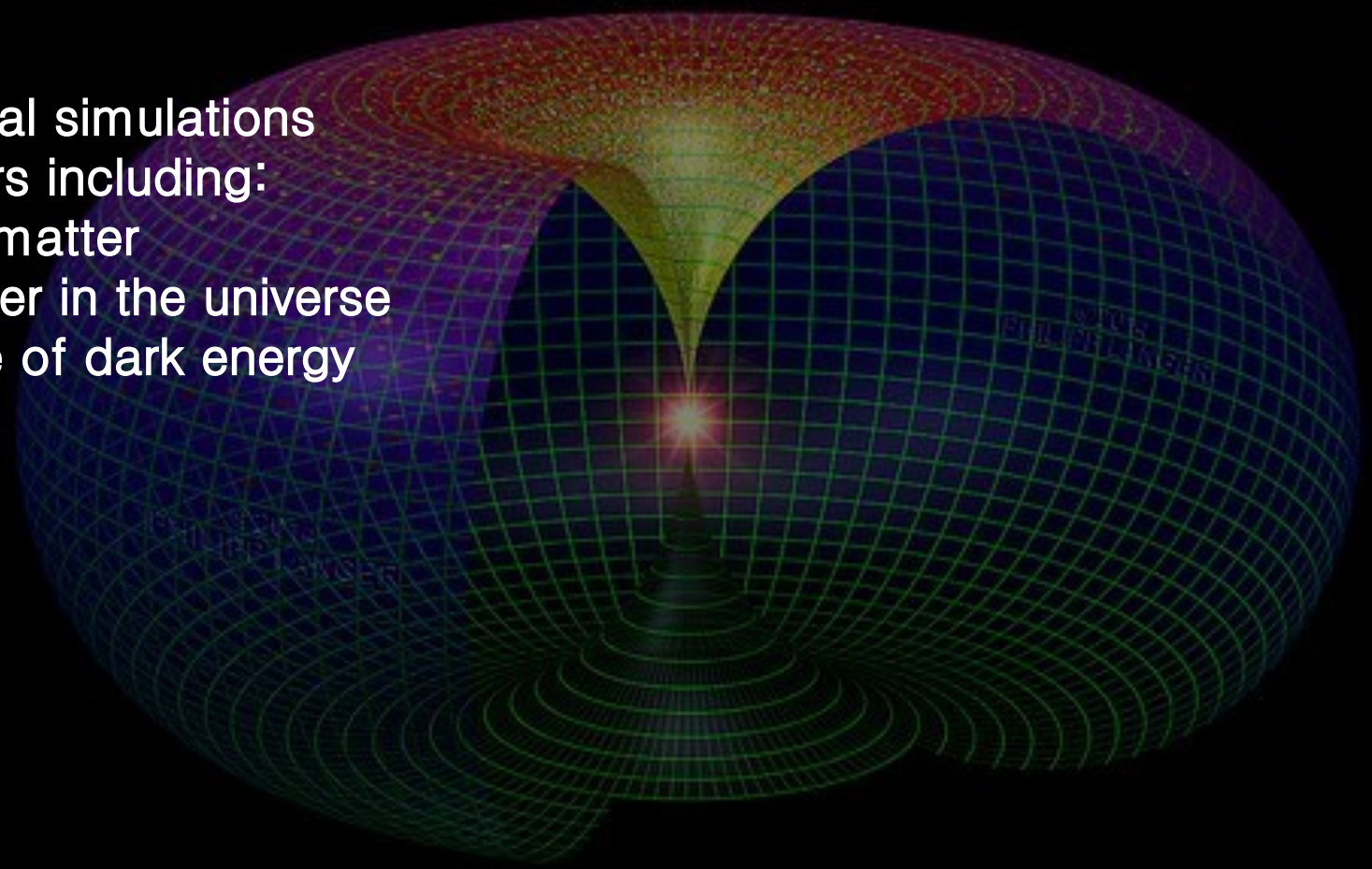




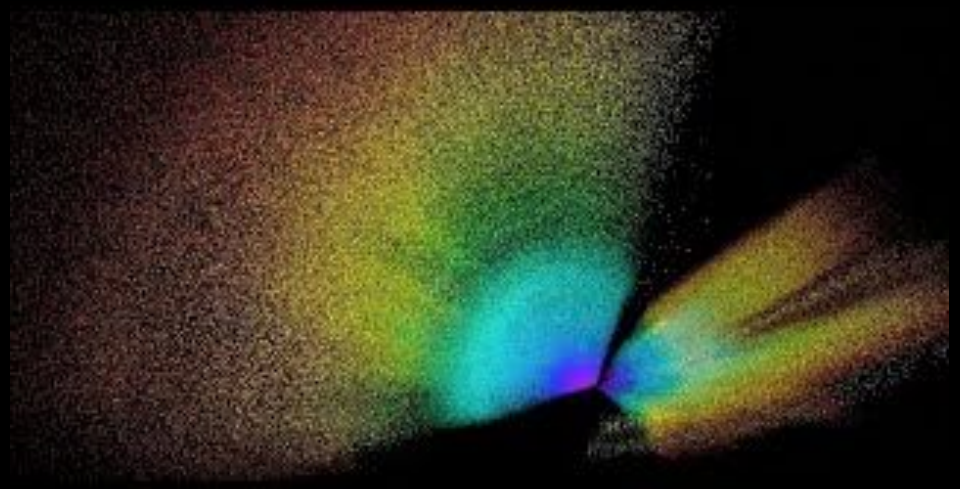
Simulation Data by Cris

We will use 500 large cosmological simulations
that sample a range of parameters including:

- Ω_b : the amount of baryonic matter
- Ω_{CDM} : the fraction of dark matter in the universe
- w (=p/ρ) : the equation of state of dark energy



Method

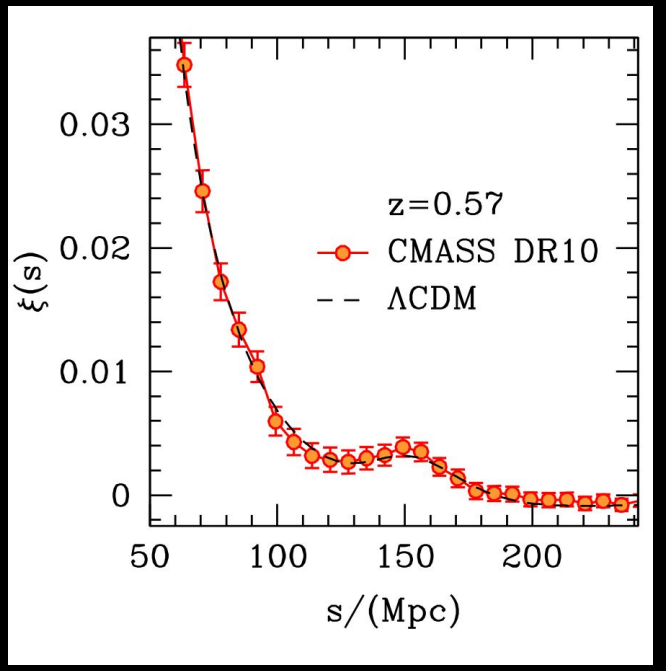


eBOS
S

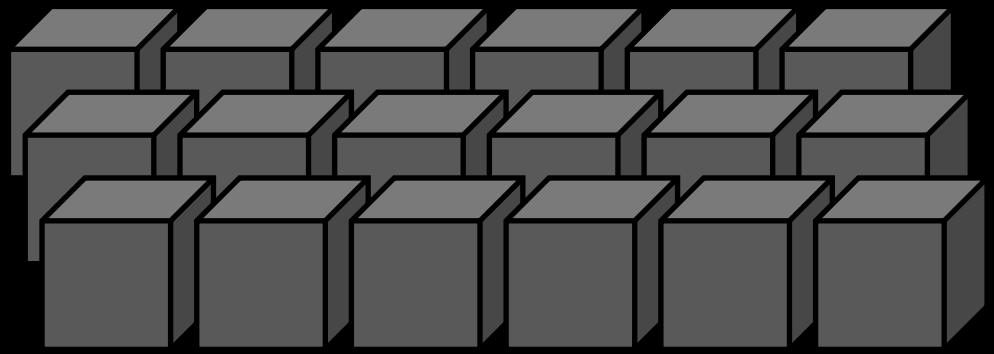
Astrostatistics

$$\widehat{\xi}_{\rm N} = \frac{DD}{RR} - 1$$

$$\widehat{\xi}_{\rm LS} = \frac{DD - 2DR + RR}{RR}$$

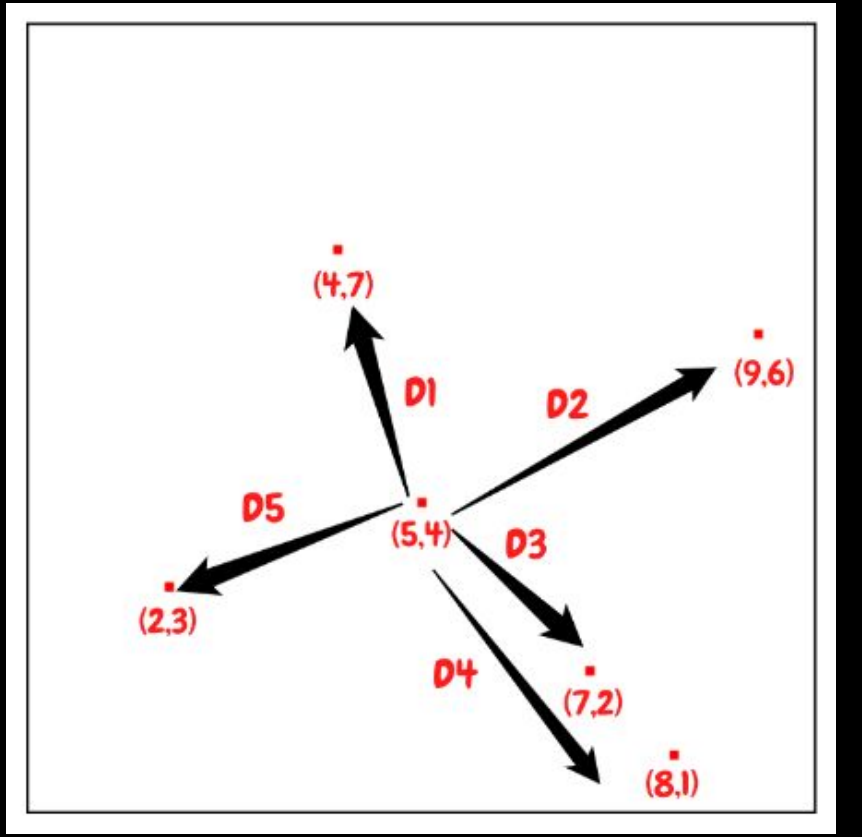


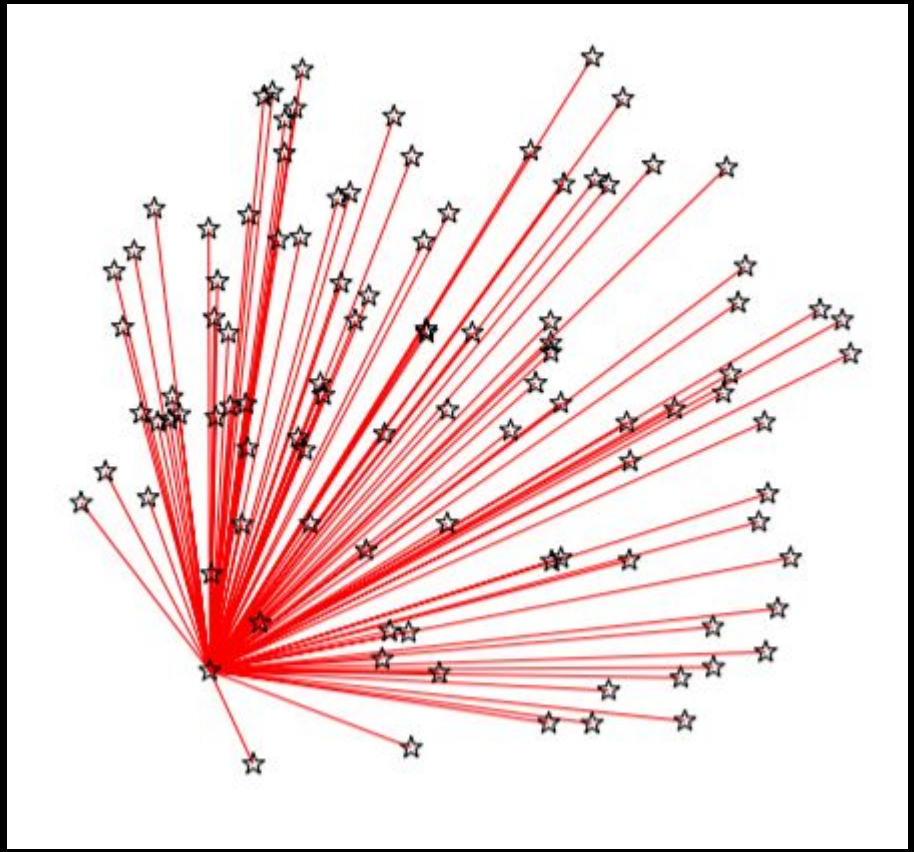
The two-point correlation function of the BOSS DR10 CMASS sample. The orange data points are the measurements for the observed galaxies, the dashed line denotes the expectation from the currently accepted cosmological model. Credit: Ariel G. Sánchez and SDSS collaboration



Simulation data

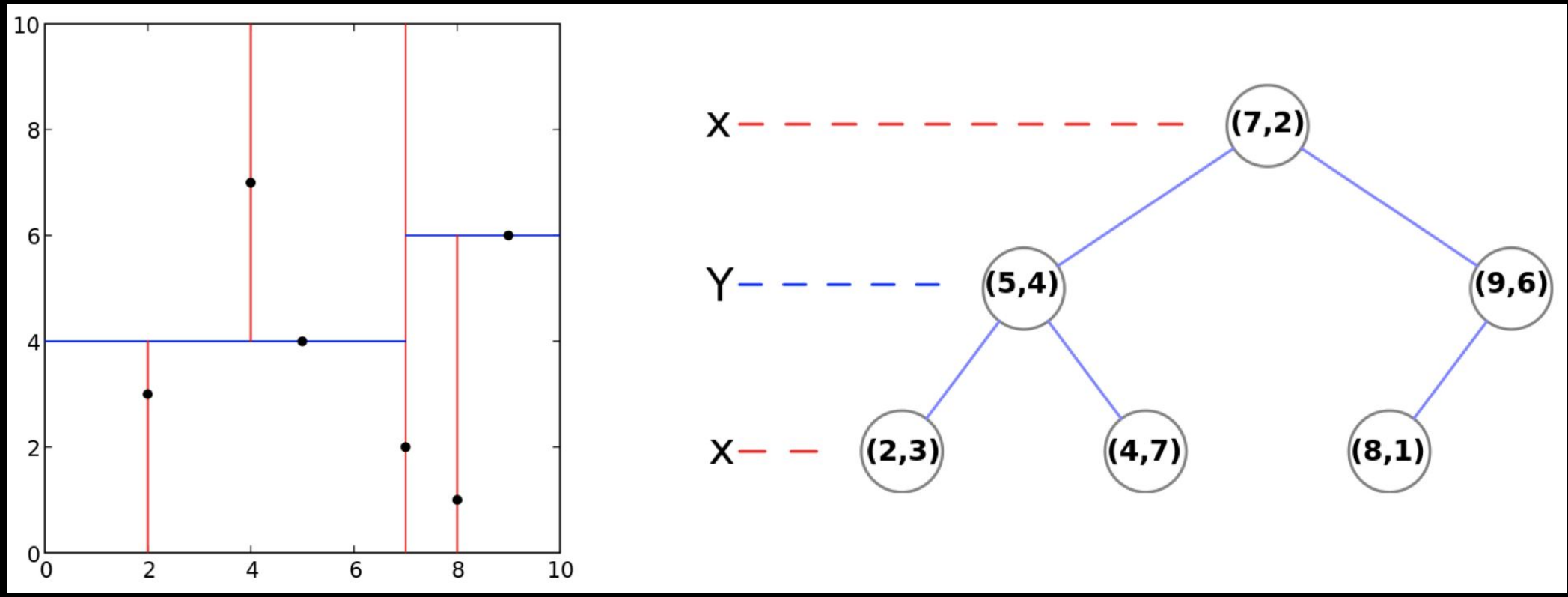
Counting Neighbors
(1) Brute Force





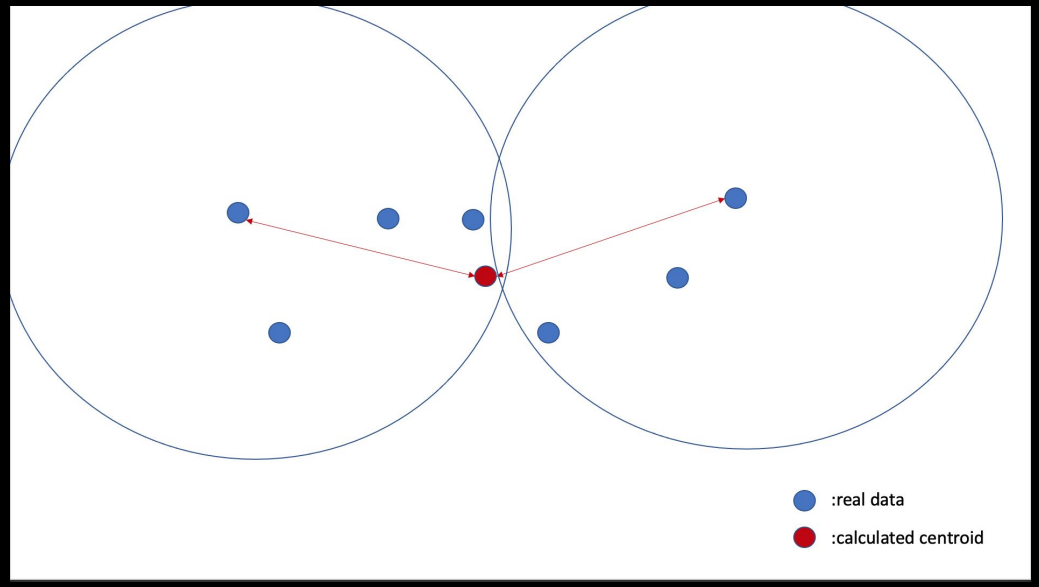
The most common neighbor search implementation involves the brute-force computation of distances between all pairs of points in the dataset: for **N** samples in **D** dimensions, this approach scales as **O[DN^2].** Efficient brute-force neighbors searches can be very competitive for small data samples. However, as the number of samples N grows, the brute-force approach quickly becomes infeasible.

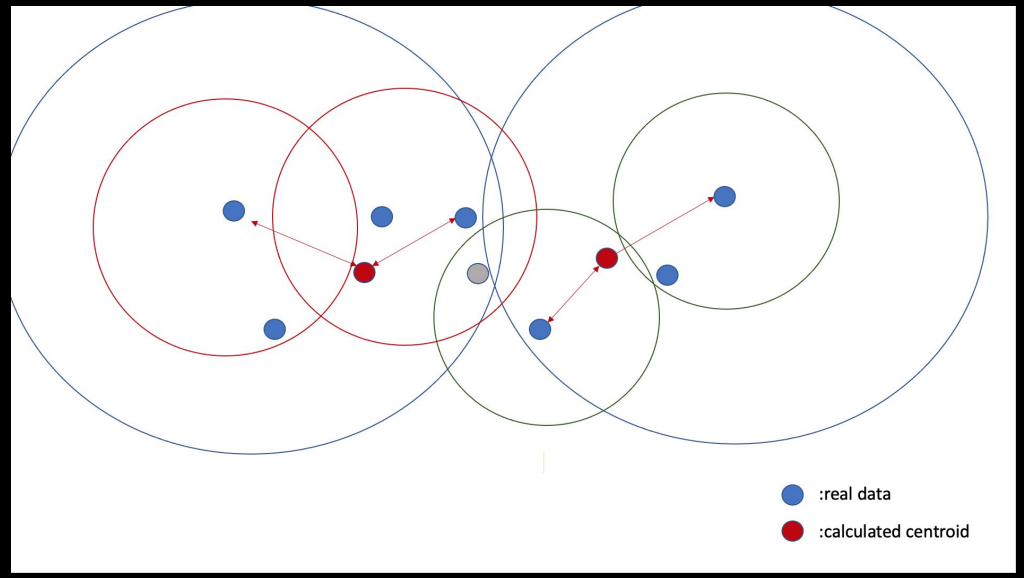
Counting Neighbors
(2) KD tree



when we choose a median of the x-values of the data, We divide the values which are greater or less than that median. and apply same method on y-values and again doing that with x and y

Counting Neighbors
(3) Ball Tree





In ball tree, we calculate the centroid of the data points and we choose 2 furthest data from that centroid.
The two data point which are chosen become a center of circle and we can draw each two circle to include all data points.
Again, we calculate the centroid of the data points for two circles and again find out which data is furthest from the centroid.
and then that data become the new center of circle.
The sphere lines can intersect each other, but the data points must be clearly assigned to one cluster.
The circle can be unbalanced (like radius of circle). But that is basically the concept behind the Ball Tree algorithm

Difference Between Methods

Difference between brute force, kdtree, balltree(10Mpc~20Mpc)
Data : ELG(number of galaxy = 83,769)

```
import time
start=time.time()

tree = KDTree(xyz, leaf_size=2)
r20=tree.query_radius(xyz, r=20, count_only=True)
r10=tree.query_radius(xyz, r=10, count_only=True)
print("r20-r10 = ",np.sum(r20-r10))

print("time = ",time.time()-start)
```

```
r20-r10 =  201454
time =  1.892350196838379
```

KDTree

```
import time
start=time.time()

def Distc(i,xyz):
    arrDist = np.sqrt((xyz[:,0]-xyz[i,0])**2
                      + (xyz[:,1]-xyz[i,1])**2 + (xyz[:,2]-xyz[i,2])**2)
    return(len(arrDist[(arrDist>10)&(arrDist<20)]))

r2010=np.sum(np.array([Distc(indx,xyz) for indx in range(len(xyz))]))
print("r20-r10 = ", r2010)

print("time = ",time.time()-start)
```

```
r20-r10 =  201454
time =  65.23844885826111
```

Brute Force

```
start=time.time()

tree = BallTree(xyz, leaf_size=2)
r20=tree.query_radius(xyz, r=20, count_only=True)
r10=tree.query_radius(xyz, r=10, count_only=True)
print("r20-r10 = ",np.sum(r20-r10))

print("time = ",time.time()-start)
```

```
r20-r10 =  201454
time =  0.8568165302276611
```

BallTree

KDTree : 1.89s

BallTree : 0.86s

Brute Force : 65.2s

CHOA

Short goal

We compute the 2– and 3– and 4–point clustering statistics of a sample of ELGs and also study the sensitivity of these statistics to various cosmological models.

Long goal

After we study with ELG data, we compare this with simulation data and study about various cosmological models. And In doing that, we can also include Machine Learning eg CNN