Mainly focused on Barabasi algorithm

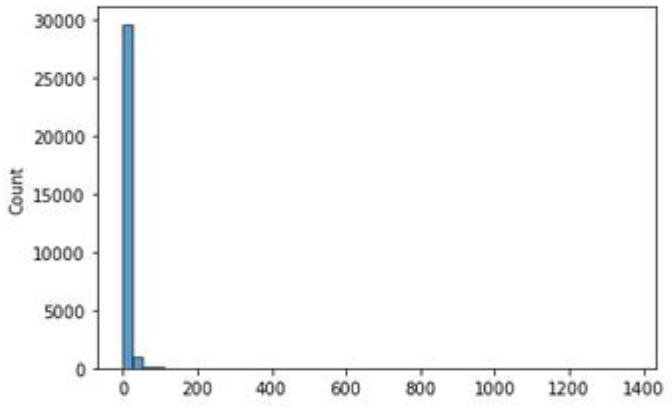
- Looking at the Barabasi algorithm
  - Making it faster
  - Making it work so it is scale-free and not exponential
  - Looking at the exponential distribution - normal distribution behaviour
  - Dunbar number

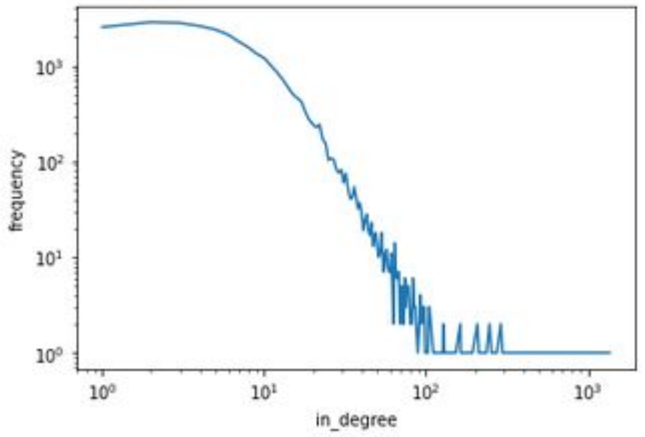
Scale-free network

- The way I implemented the Barabasi algorithm does not show the scale-free property
- So I changed it to the following:
    - Start by choosing a random destination node from the destination group
    - Put this node inside a destination bin
    - Take a random node from the source group
    - Connect random node from source group with a random node from the destination bin (which has in the beginning only 1 node inside)
    - Add the chosen node from the bin again in the Bin (so we increase its probability of being chosen)
    - Add a random other node from the destination group to the destination bin

Checking if sale-free property is met

- Example between the same group ('[0,20), Native, Man, 1')
- Looks scale free

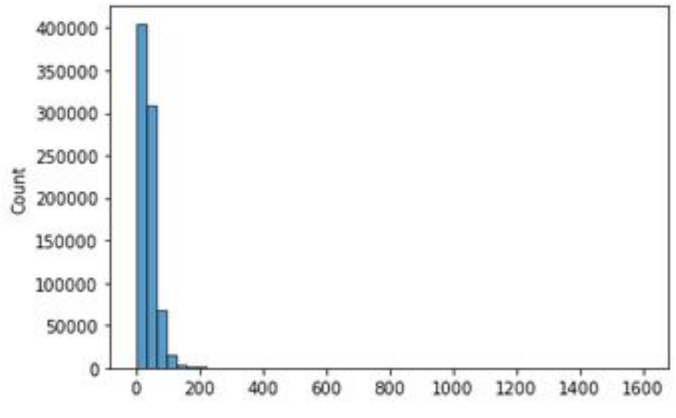


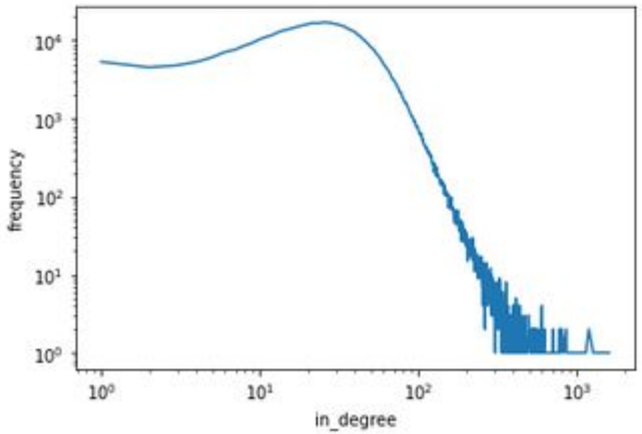


Checking if sale-free property is met

- Using the package power_law to make sure
  - Jeff Alstott, Ed Bullmore, Dietmar Plenz. (2014). powerlaw: a Python package for analysis of heavy-tailed distributions
- Comparing fit between power law and exponential
  - power law significant
- Power law - truncated power law
  - No significant difference
- Power law/truncated power law - lognormal
  - No significant difference
  - But not big problem as Alslott et al., show that even well known powerlaw data fit both
  - Also the paper of Broido and Clauset show that lognormal are even more common and almost all powerlaws also fit lognormal
    - Broido, A.D., Clauset, A. Scale-free networks are rare. *Nat Commun* **10,** 1017 (2019).

Looking at the whole network

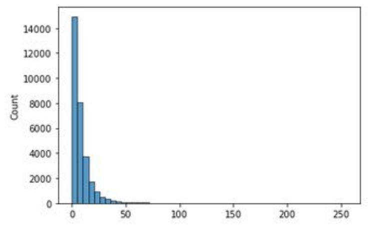


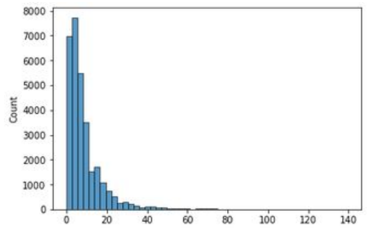


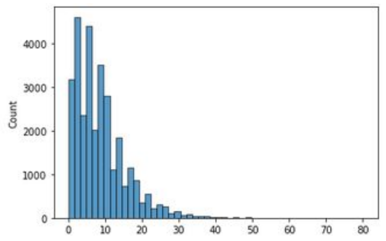
- Statistics are the same as with the ('[0,20), Native, Man, 1') group

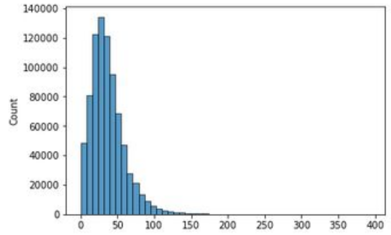
Exponential distribution

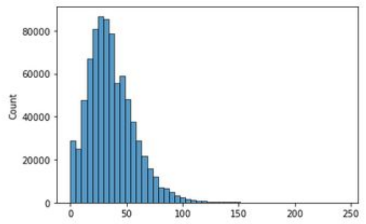
- When we increase the amount of initial nodes in the destination bin, we see that the distribution between groups start following an exponential distribution and the total network will look more like a normal distribution
- E. of workschool data on groups ('[0,20), Native, Man, 1') with 1,5 and 50%

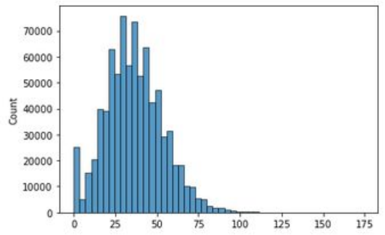












Speeding up the network initialization

- Because I changed the weighted random.choice function with a normal random.choice function the process speeded up a lot.
- Both Barabasi and random network can now be runned on the local computer in 3-4 minutes

Dunbar number in the scale-free network (work/school)

- It has been proposed to lie between 100 and 250, with a commonly used value of 150.
- With a destination bin of 1% only 66 persons have a higher in degree value than 250, the largest has a in degree of 394 making it more plausible 1% initial nodes more plausible than only 1 node which results in
- highest in-degree of 1987 links and 963 nodes above 250 in-degree

- Interesting is that with random network the maximum in_degrees lays on 152

To do

Sharing

- I will upload my code to github so everyone can see it
- Also I will put all the presentations together so we have a overview

Initial network

- Still looking at reciprocity
- Making multilayered network
- Looking at spatial data
- Age distribution

Dynamical network

- Also thinking about possible dynamics
- Maybe brainstorm on the possibilities

Notes

- Instead one put all the nodes based on the random parameter
- High percentile to dunbar
- Algorithms that produce scale-free properties when you have two groups
    - Check if there is a paper already on this topic

- Only Age the population in static network
    - age distribution in Amsterdam in group
    - Age population into the future ⇒ remove nodes when dead
    - Change links when getting older or stay the same
    - **Show network dynamics move one edge to the other**
        - **Check how new age group compares to old age group**