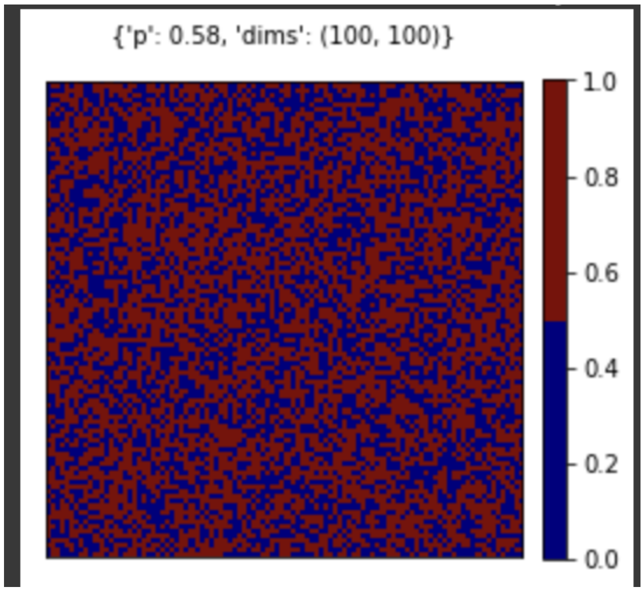
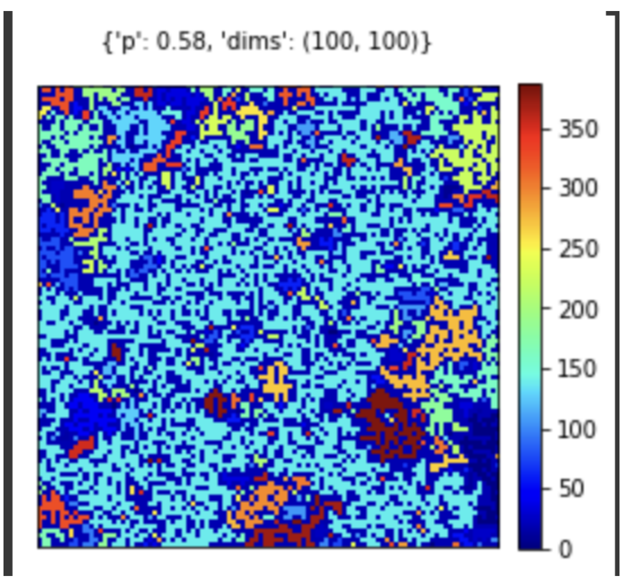
NetSci Homework#2 Report **Muhammed Orhun Gale - 26754**

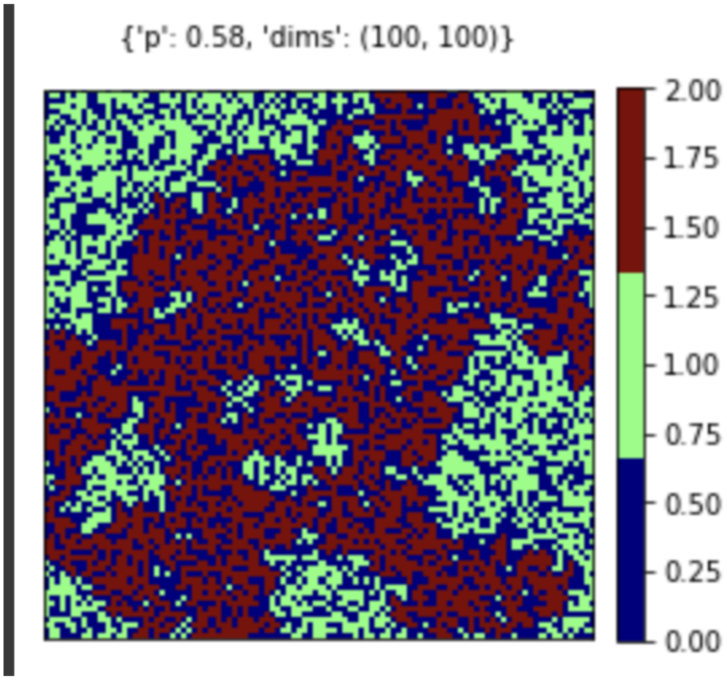
As the first step to simulate percolations, a lattice is filled with 0’s for unoccupied cells and 1’s for occupied cells in accordance with the probability of a cell being occupied or not.



In order to visualize clusters, I used the scipy.ndimage library which is mainly used for image processing. I imported measurements.label function in order to label Connected Components (CCL) which are clusters in this case. Besides visualizing the clusters, this also give me the clusters with their labels which I used to detect percolation. In order get and evaluate each and every cluster, I used np.where function and



After I got clusters and their label information, in order to detect percolation I used a very basic technique. For all clusters, I checked if there exist two elements which one of them starts with X index 0 and the other one ends with N (depending on the X axis size of the matrix). For the Y axis, I do the same thing. This technique works correctly since if there exist two such points in a cluster, it means that the cluster lies between those two points since the clusters are connected components.



Red area is the percolated cluster.

After I completed the above steps which are required to process lattices, I ran simulations with random inputs and created a JSON file which includes each lattice, clusters in them, percolation status and other various information. I used two different simulations over lattices which are 100x100. For the first simulation which is named as the pExplore, I ran 100 simulations for probabilities ranging from 0 to 1 with 0.1 increment. For the pExpand simulation, I ran 50 simulations for probabilities ranging from 0.5 to 0.65 with 0.003 increment. With the pExplore simulation, I aimed to find the probability interval where the Pc is located. Using this simulation's results, I decided that the Pc is located around 0.5 and 0.65 and in order to converge the exact Pc value I used the pExpand simulation.

```
#pExpand

NRANDOM = 50
DIMS = (50,50)
pExpand = np.linspace(0.5,0.65,52)[1:]

expResults = list()
for n in range(NRANDOM):
```

```
#pExplore

NRANDOM = 100
DIMS = (50,50)

pExplore = np.linspace(0,1,101)[1:-1]

expResults = list()
for n in range(NRANDOM):

    # You can run simulation for pExplore
```

```
exploreDf
```

	p	idx	dim	isPercolated	clusters
0	0.01	0	[100, 100]	False	[[[7, 18]], [[77, 9]], [[94, 29]], [[94, 35]],...
1	0.02	0	[100, 100]	False	[[[30, 11]], [[69, 51]], [[66, 62]], [[63, 7]]...
2	0.03	0	[100, 100]	False	[[[14, 77]], [[39, 24]], [[13, 31]], [[65, 16]...
3	0.04	0	[100, 100]	False	[[[96, 57]], [[54, 72]], [[60, 89]], [[11, 67]...
4	0.05	0	[100, 100]	False	[[[94, 56]], [[92, 40]], [[4, 72]], [[59, 76]]...
...	...	...	...	...	...
9895	0.95	99	[100, 100]	True	[[[0, 0], [0, 1], [0, 3], [0, 4], [0, 5], [0, ...
9896	0.96	99	[100, 100]	True	[[[0, 0], [0, 1], [0, 2], [0, 3], [0, 4], [0, ...
9897	0.97	99	[100, 100]	True	[[[0, 0], [0, 1], [0, 2], [0, 3], [0, 4], [0, ...
9898	0.98	99	[100, 100]	True	[[[0, 0], [0, 1], [0, 2], [0, 3], [0, 4], [0, ...
9899	0.99	99	[100, 100]	True	[[[0, 0], [0, 1], [0, 2], [0, 3], [0, 4], [0, ...

9900 rows × 5 columns

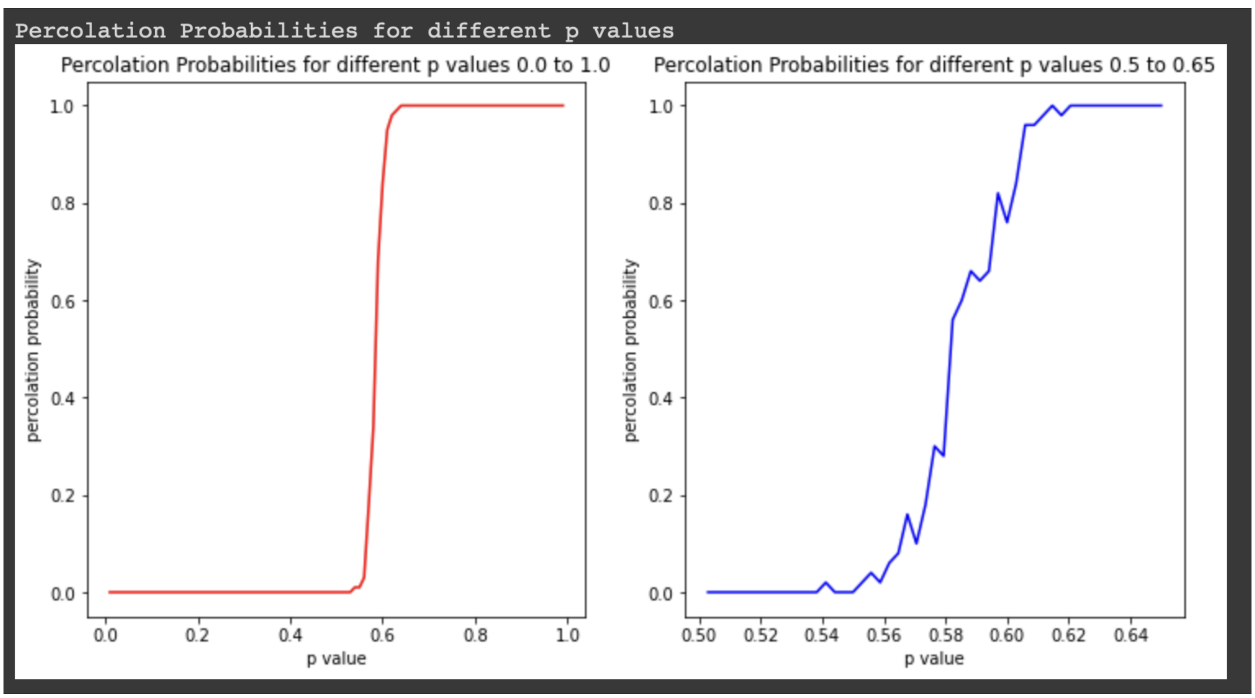
```
expandDf
```

	p	idx	dim	isPercolated	clusters
0	0.502941	0	[100, 100]	False	[[[36, 0]], [[27, 18]], [[75, 95]], [[35, 83]]...
1	0.505882	0	[100, 100]	False	[[[29, 28]], [[97, 4]], [[93, 88]], [[87, 35]]...
2	0.508824	0	[100, 100]	False	[[[54, 35]], [[19, 75]], [[27, 75]], [[27, 77]...
3	0.511765	0	[100, 100]	False	[[[1, 63]], [[89, 41]], [[94, 27]], [[54, 71]]...
4	0.514706	0	[100, 100]	False	[[[11, 97]], [[88, 48]], [[56, 72]], [[22, 98]...
...	...	...	...	...	...
2545	0.638235	49	[100, 100]	True	[[[31, 73]], [[90, 77]], [[39, 72]], [[52, 0]]...
2546	0.641176	49	[100, 100]	True	[[[88, 15]], [[24, 94]], [[37, 42]], [[65, 35]...
2547	0.644118	49	[100, 100]	True	[[[3, 49]], [[95, 37]], [[83, 79]], [[20, 36]]...
2548	0.647059	49	[100, 100]	True	[[[13, 81]], [[11, 40]], [[19, 61]], [[66, 10]...
2549	0.650000	49	[100, 100]	True	[[[58, 79]], [[56, 24]], [[44, 66]], [[20, 16]]...

2550 rows × 5 columns

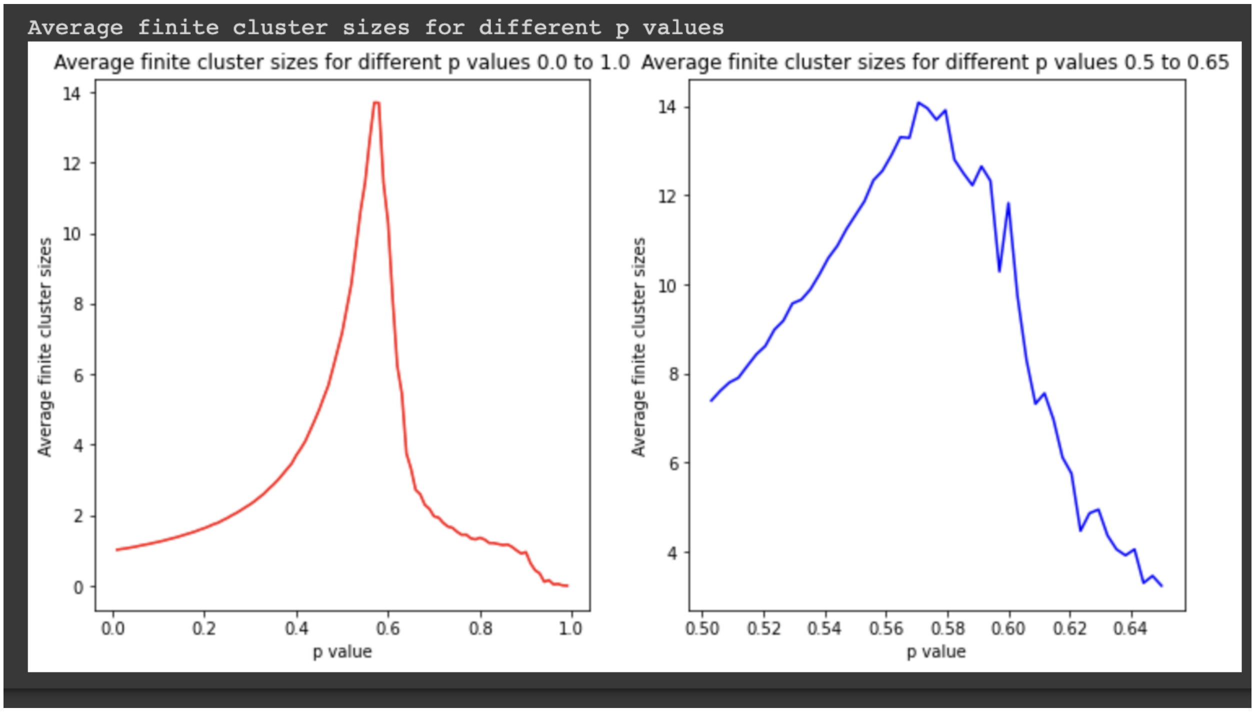
In order to analyze the simulation results and decide a Pc value, I used various strategies that are shown in the class.

First I used “Percolation Probabilities for different p values” in order to determine the Pc value.

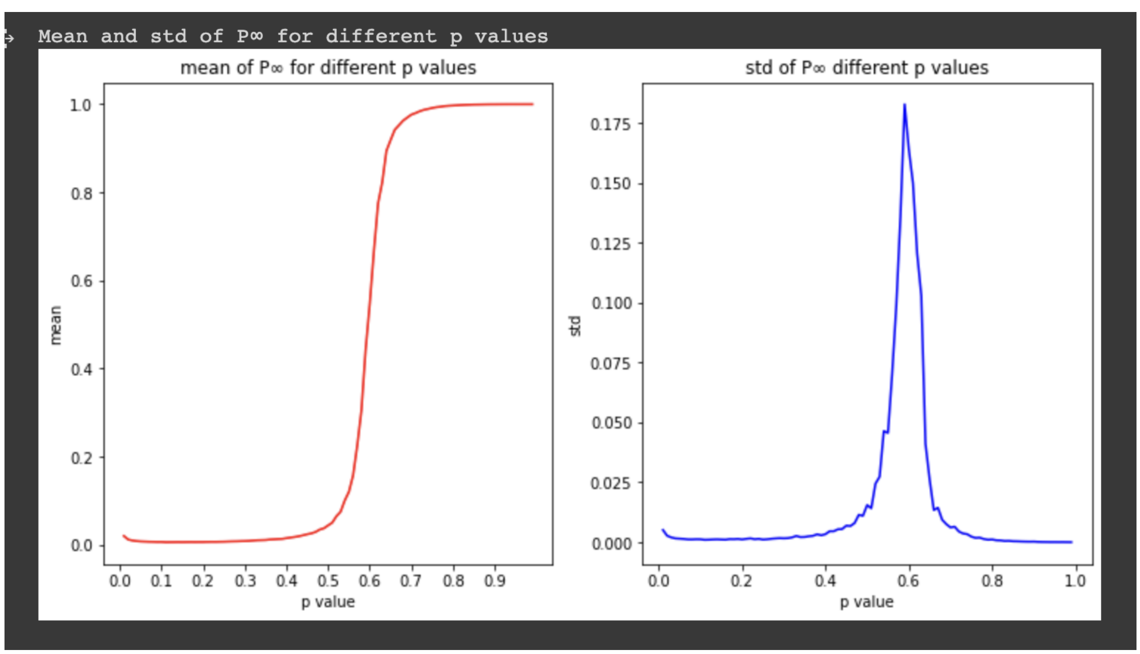


According to the first simulation it can be said that the Pc value is located around the p value 0.57 - 0.62 and according to the second simulation which expands this interval it can be said that Pc value’s exact location would be around 0.59.

Secondly, I used the “Average finite cluster sizes for different p values” strategy in order to detect a sharp increase in the finite cluster size and I found that this phenomenon occurs around p values 0.58. When the first and second strategies' results are combined it can be said that Pc value is located between 0.58 and 0.59.



Lastly, I tried to analyze the P_infinity value via analyzing its mean and standard deviation and I found that again the P_infinity value gives the intuition that the Pc value would be around 0.58-0.59. This information shows that around these values, for a cell, probability of being a member of the biggest cluster drastically increases and std shows that this change occurs mostly around these points.



Therefore, I can conclude that the Pc value is around 0.58-0.59.