Project 3 - Percolation
FYS4460

Fredrik E Pettersen

May 3, 2013

a

Starting out with a uniform random matrix which we transform to a porous matrix consisting of either filled (1) or empty (0) places we will now have clusters of connected, filled places. The ammount of filled and empty places depends on our chosen porosity for the random matrix. An example of a porous, labelled matrix is shown in figure 1.
We want to find the probability distribution for a site in the porous matrix to be in a spanning cluster. A spanning cluster is a cluster of connected sites which makes a path from one side of the matrix to the other. This can be done by finding the area of the spanning cluster, and divide it by the area of the entire matrix. We find the spanning cluster by labelling the connected clusters and checking if the same labels can be found on each side of the matrix. The resulting "distribution" is shown in figure 2. This method lets me calculate at least for any rectangular matrix boundary, though I do not know how periodic bounday conditions will affect the calculations.

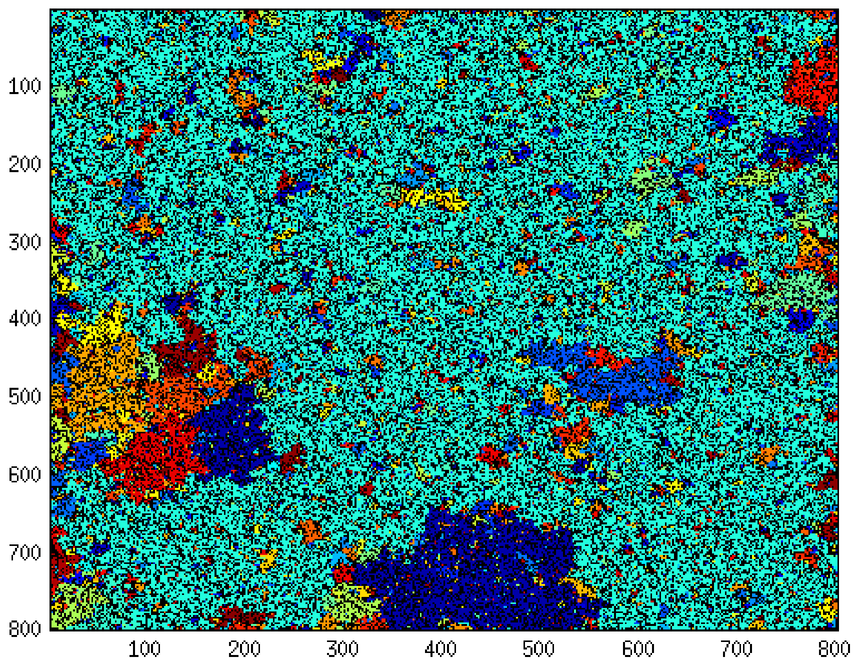


Figure 1: An example of a porous 800×800 matrix with porosity $p = 0.6$.

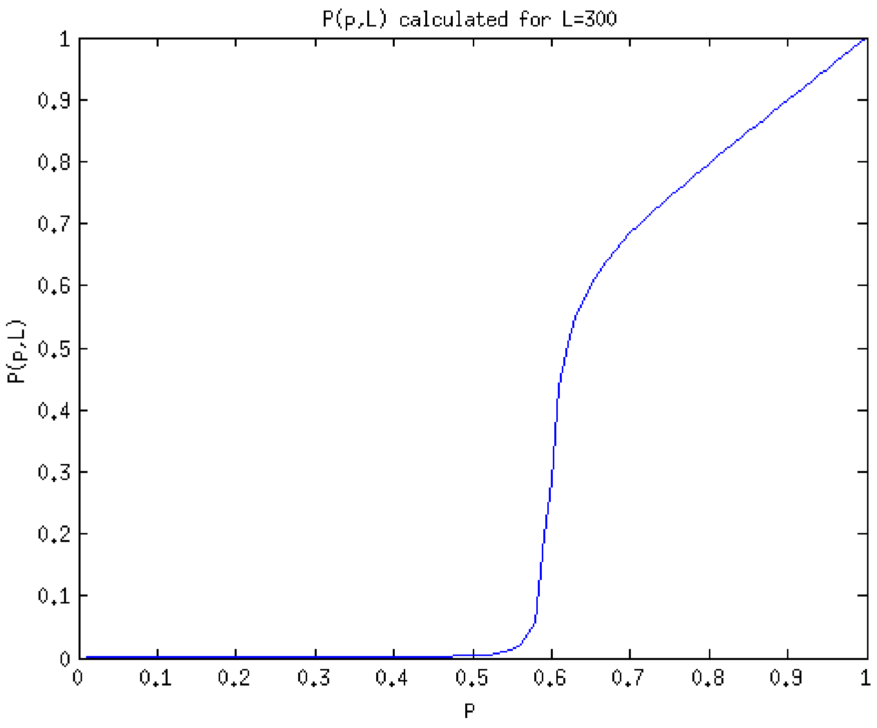


Figure 2: Probability for a randomly selected site to be a part of the spanningcluster

b

For $p > p_c$ we have that $P(p, L) \propto (p - p_c)^\beta$ $(p_c = 0.59275)$. We can find β by doing a double-logarithmic plot of our measured $P(p, L)$ versus $(p - p_c)$.

c

We are now gives a collection of random numbers on the form

```
z = rand(1e6,1).^(-2);
```

and area asked to find the distribution function $f_z(z)$ using that $f_z(z) = \frac{dP(Z>z)}{dz}$ and that we know f is on the form $f_z(z) \propto z^\alpha$. We can plot the cumulative distribution function $P(Z > z)$ (see figure 4) and use this to calculate the distribution function. Since

$$z^\alpha \propto \frac{dP(Z > z)}{dz} \implies \int z^\alpha dz \propto \int dP(Z > z)$$

$$\frac{1}{\alpha + 1} z^{\alpha+1} \propto P(Z > z) \implies (\alpha + 1)\log(z) \propto \log(P(Z > z))$$

So we see that we are actually measuring $\alpha + 1$ if we do a double logarithmic plot of z and the cumulative distribution function. From this plot (figure 3) we find $\alpha = -3/2$. In figure 4 we have plotted the measured cumulative distribution function and the modelled $f_z(z) = z^\alpha$. We see that there is a good fit between the two graphs.

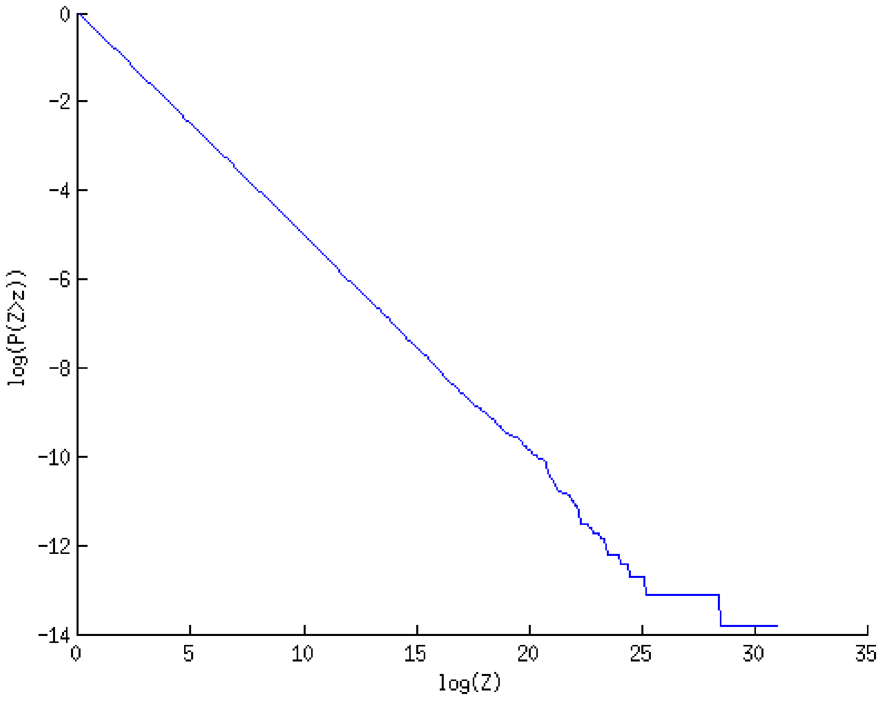


Figure 3: Double logarithmic plot of the cumulative distribution function $P(Z > z)$

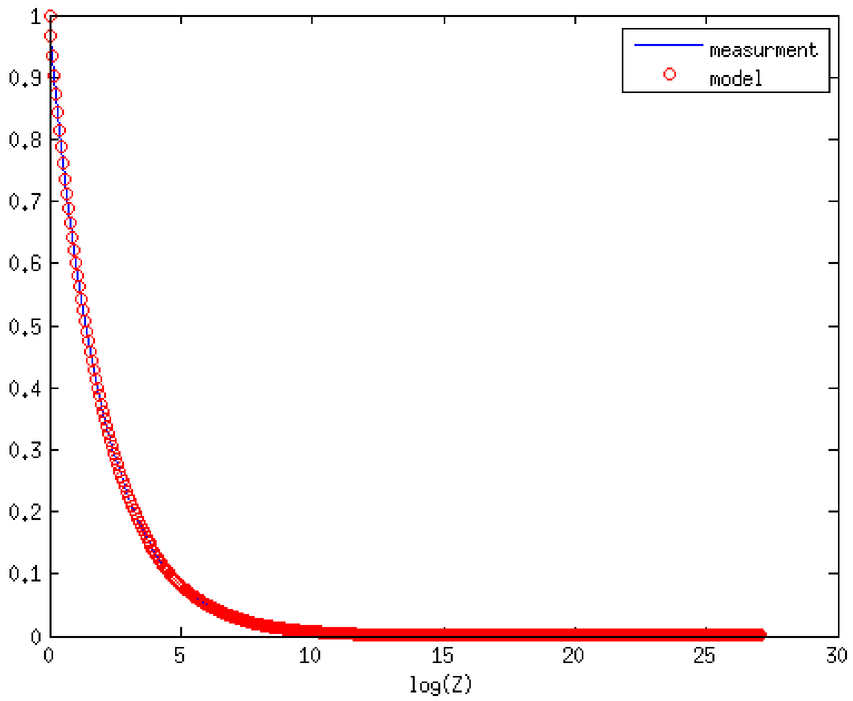


Figure 4: The measured cumulative distribution function (blue line) and the calculated model $f(z) = z^{\alpha}$ (red circles).

d

There are quite a few ways to do logartihmic binning, some more elaborate than others. In the lecturenotes we can find one quite elaborate method (which is probably very good). Another very simple method is to use the following command i MatLab

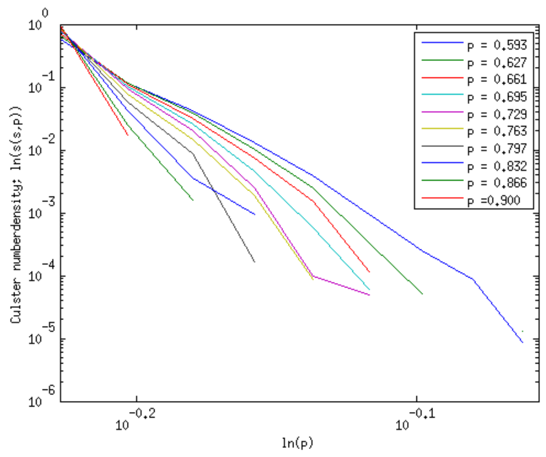
```
z = floor((1.8)^(0:N));
```

The method I have settled on is also a quite simple one

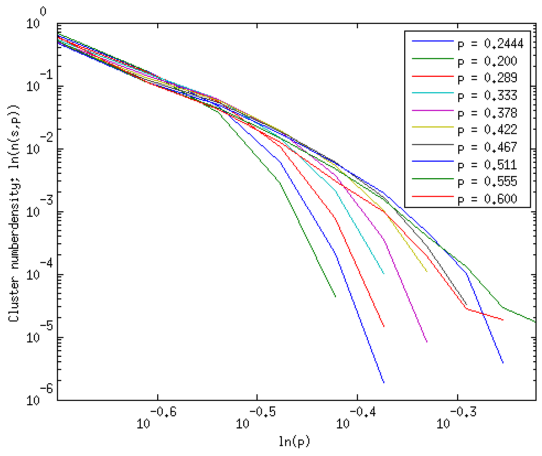
```
z = floor(exp(linspace(1,log(max(s.area)),N)));
```

e

We can find the cluster numberdensity $n(s, p)$ as the number of clusters with a certain area relative to the total number of clusters in the matrix. This is done for different porosities on matrices of the same size. As we can see from figure 5 the cluster numberdensity decreases as a power-law with increasing clustersize.



(a) $p > p_c$



(b) $p < p_c$

Figure 5: Cluster numberdensity for various porosities.

f

Doing the same thing for differnt system sizes at the critical porosity will let us estimate the exponent τ from the relation $n(s, p; L) = s^{\tau}$. Doing the usual loglog-plot will give us a straight line where τ is the inclination of the line. Matlab's polyfit function gives us $\tau \approx -1.84$, which is not a perfect fit to the tabulated value of $\tau = 2.05$, but still not to bad. The graph used is shown in figure 6

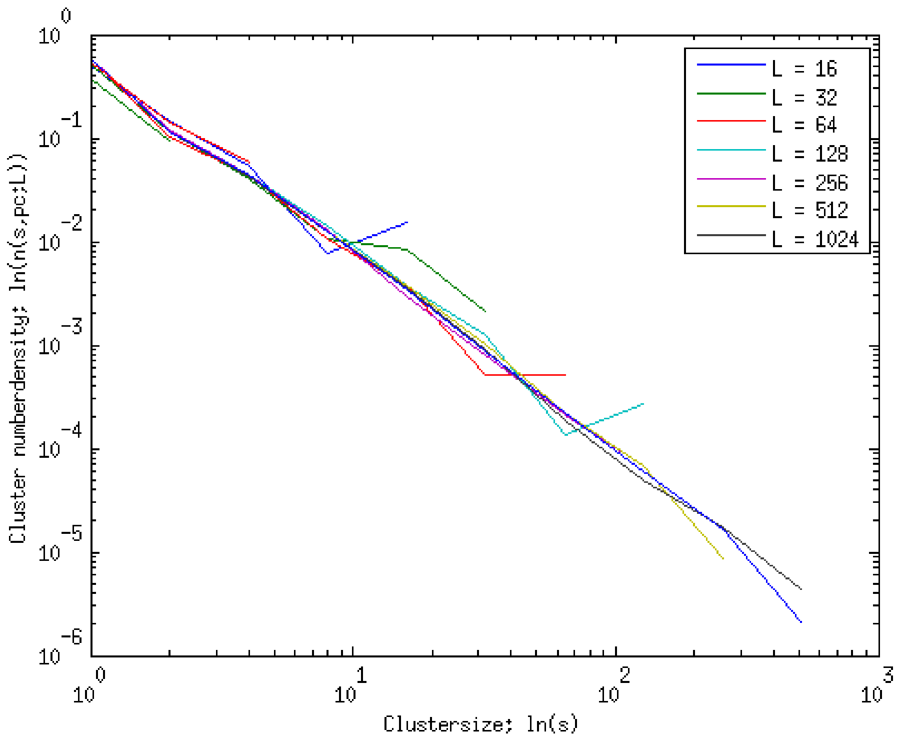


Figure 6: An estimate of τ. The plot does not look exactly like it should, probably a result of a to large binsize in the logarithmic binning used for the clustersizes.

g

Using the dataset from the two previous sections we could estimate the typical clustersize s_ξ, however there is a lot of noise in this data, and our estimate would be a poor one. A better way of estimating s_ξ is to make a data-collapse plot. We know that $s_\xi \sim |p - p_c|^{-1/\sigma}$. If we plot the cluster numberdensity for increasing p-values (keeping L constant) and multiply with s^τ which we found in section we should get a data-collapse plot where the curves "fall onto eachother" for a good value of σ. The tabulated value is $\sigma = 0.395$, and I found $\sigma \approx 0.4$. Figure 7 shows the data- collapse.

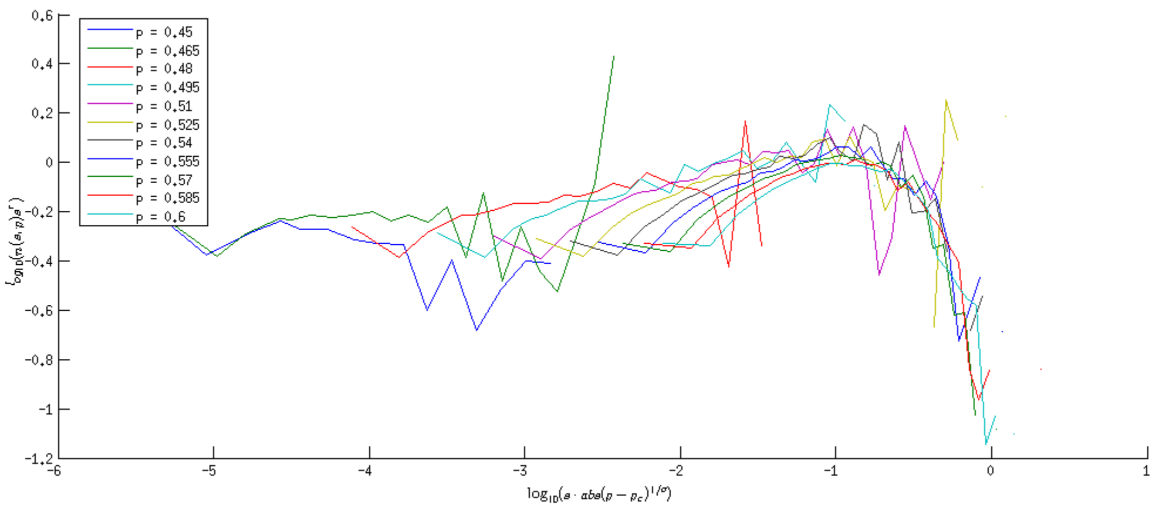


Figure 7: Datacollapse

h

We will now find the “mass” of the percolating (spanning) cluster at $p = p_c$ as a function of the system size L. In 3D the mass of an object is proportional to its volume, it stands to reason then that a 2D object will have a mass proportional to it’s area. We will also assume (more specifically) that the mass of the percolating cluster will satisfy the following

$$M(p, L) = L^D P(p, L).$$

Doing a double logarithmic plot of M as a function of L lets us determind the exponent D. The plot is shown in figure 8, and using matlabs polyfit function we find $D \approx 1.89$ which is the tabulated value.

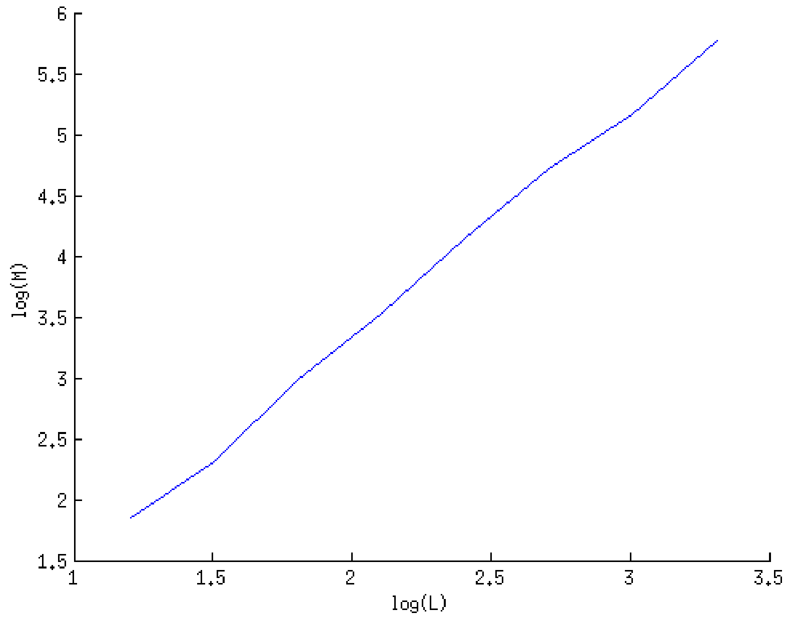


Figure 8: Mass of the spanningcluster for increasing systemsize L.

i

Defining $\Pi(p, L)$ which is the probability of having a percolating cluster, this should go to a Heaviside step-function in $p = p_c$ as the system size goes to infinity. We can of course check this (see figure 9), but what is more interesting is to try and estimate another exponent using finite size scaling. Finding the probability for Π to be equal to some porbability (say $p_{\Pi=x}$) which is simply read of figure 9 we have the relation

$$p_{\Pi=x_1} - p_{\Pi=x_2} = (C_{x_1} - C_{x_2})L^{-1/\nu} \tag{1}$$

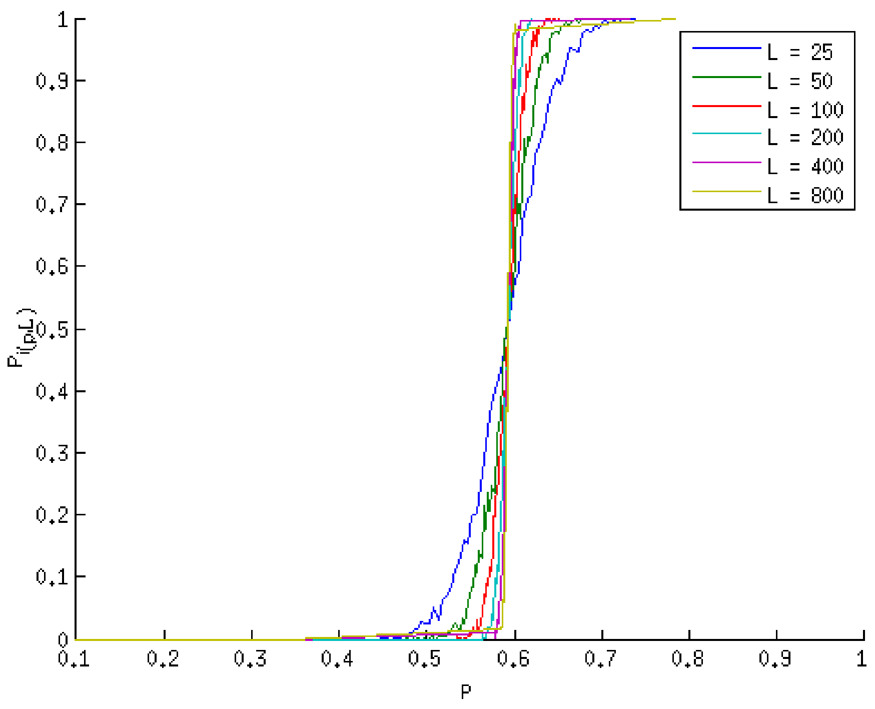


Figure 9: Probability for there to be a percolating cluster on our matrix, plotted for different systemsizes.

j

It should come as no surprise then that we can use equation 1 to estimate ν by doing a double logarithmic plot of $p_{\Pi=x_1} - p_{\Pi=x_2}$ vs L. This plot is shown in figure 10. Using MatLab's polyfit function I got $\nu \approx 1.262626$ compared to the analytical value of $\nu = \frac{4}{3}$. I have only made a rough estimate using some 10 samples when estimating this.

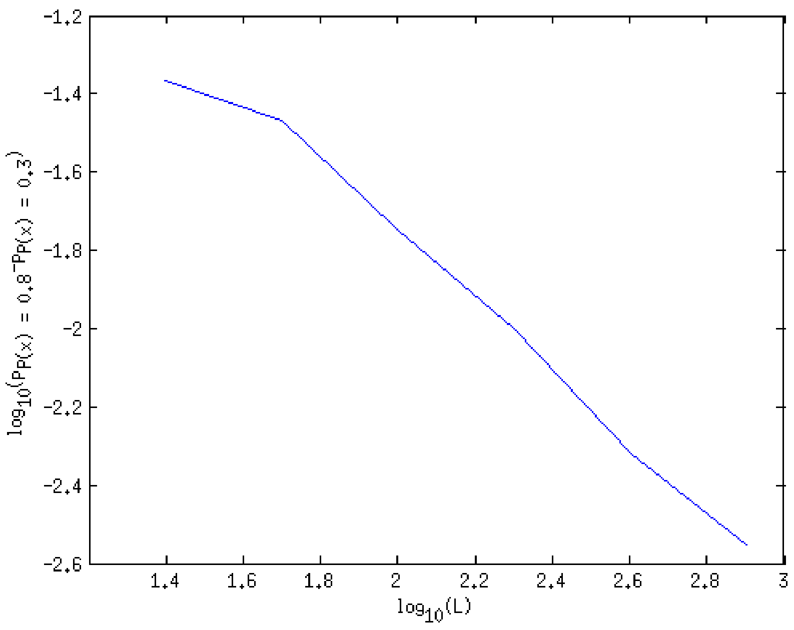


Figure 10: Estimate of ν for $x_1 = 0.8$ and $x_2 = 0.3$

k

Using the data we now have gathered we can make an estimate of the percolating thershold p_c. Plotting $p_{\Pi=x}$ vs $L^{-1/\nu}$ theese lines will intersect the y-axis in p_c (see figure 11). This figure gives the estimate of p_c as $p_c \in [0.5923, 0.594]$, which is not bad considering the small ammount of samples I have used.

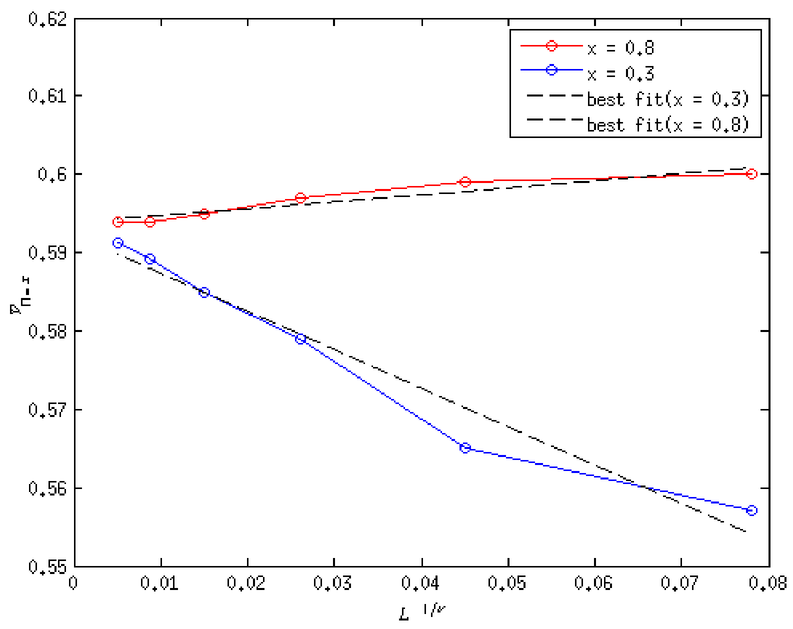


Figure 11: Estimate of p_c for $x_1 = 0.8$ and $x_2 = 0.3$

l

We are given the script “exwalk” which uses the function “walk” to find the singly connected bonds of a percolating cluster. The script finds the percolating cluster of a matrix and starts a walker on this cluster. The walker mooves by trying to turn left or right as much as possible. It’s goal is to make it to the other side of the matrix. Two walkers try walking across the cluster from top to bottom, and from right to left. A singly connected bond is a cluster of places on the matrix where both walkers have visited. The singly connected bonds of a randomly chosen matrix is visaluzed in figure 12. The script finds theese singly connected bonds by storing the path of each walker, that is it’s position on the matrix, and how long it stayed at each position in the form of which direction it moved further in. Theese matrices will be rather sparse, and so multiplying them will result in a matrix containing the places both walkers have visited. Some of theese positions will have values larger than 1, and theese positions are the places where at least one of the walkers have lingered.

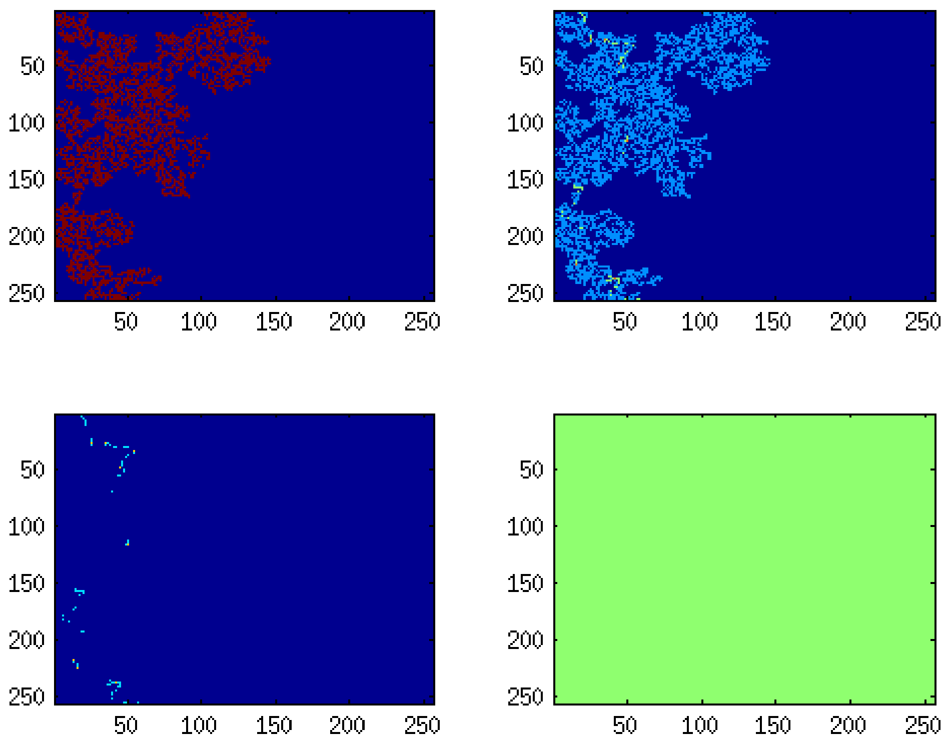


Figure 12: Various output from the script “exwalk.m“. The bottom left picture shows the singly connected bonds of the percolating cluster. The top left picture is the percolating cluster, and the top right is the percolating cluster with the singly connected bonds on top.

m

We can also find the mass of the singly connected bonds as a function of system size from the relation $M_{SC} \propto L^{D_{SC}}$. Figure 13 shows this relation averaged over 500 samples. MatLab’s polyfit function gives us $D_{SC} \approx 0.7745$.

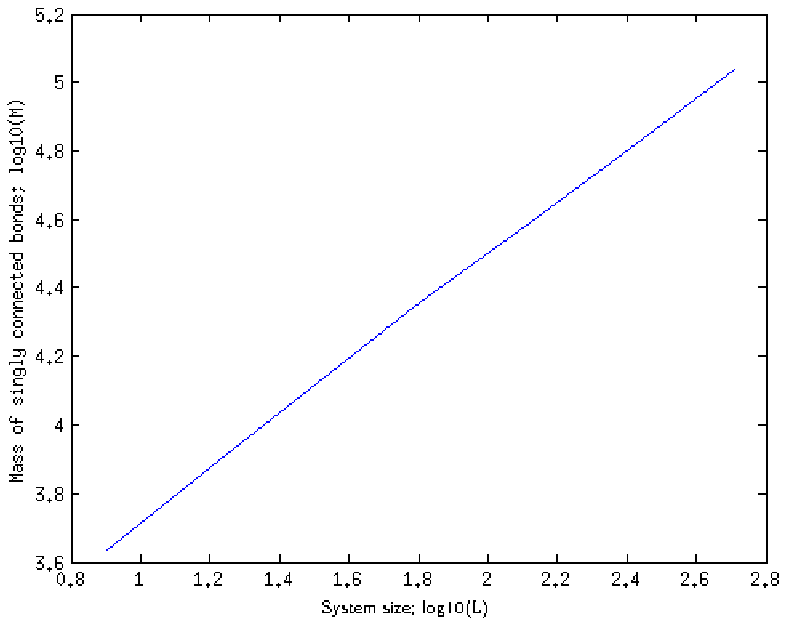


Figure 13: Estimate of M_{SC} as a function of L for $L = 2.^{(3:9)}$. 500 samples used per L.

q

We will now study random walks on the percolating cluster. Again, we are given the important parts of the code (one C program which can be used in matlab through mex, and one matlab script which uses the program). The C program returns two vectors containing the distance moved in x and y direction. We can find the distance $\langle R^2 \rangle$ moved by the walker by a simple conversion $\langle R^2 \rangle = (x(end) - x(1))^2 + (y(end) - y(1))^2$. And to get more accurate results we will average this over 5000 samples. We are interested in how $\langle R^2 \rangle$ scales with the number of steps the walker carries out for different porosities. The results can be found in figure 14 .
Producing a data collapse is not possible...

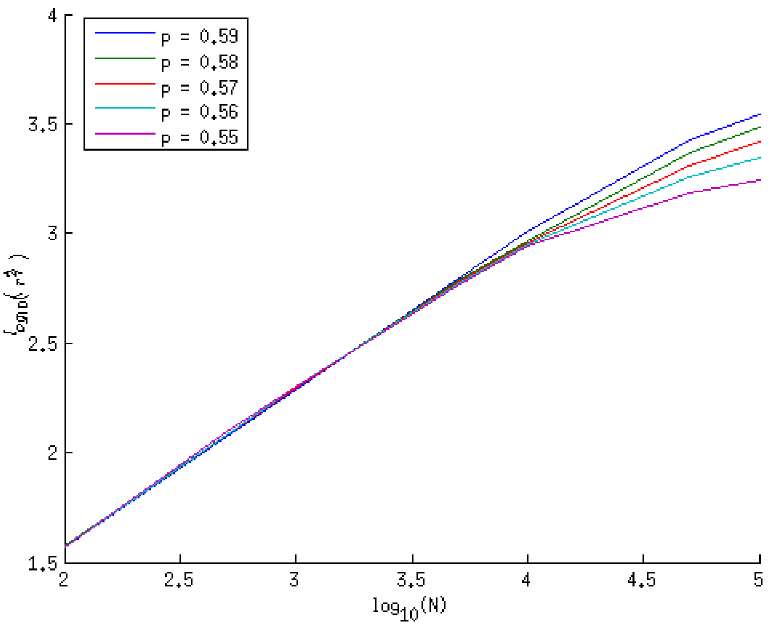


Figure 14: $\langle R^2 \rangle$ as a function of the number of steps done by the walker. Double logarithmic axes.

r

From figure 14 and figure 15 we can estimate the correlation length ξ. Though it might not be very clear from fgure 15, $\langle r^2 \rangle$ will tend to a constant as N tends to infinity because the walker is walking on a cluster of finite size (that is, not a spanning cluster). The correlation length squared, ξ^2 is found where the graphs of $\langle r^2 \rangle$ start to fall off. We read this off figures 14 and 15 as happening at roughly $N = 10000$, making $\xi^2 \approx 1000$ and $\xi \approx 32$. From figure 14 we can also finf the dimensionality of the walker by using the relation $\langle r^2 \rangle \propto N^{2/d_w}$ meaning that the incline of the curves in figure 14 is $\frac{2}{d_w}$. We find d_w to be somewhere between 2.766 and 2.986. This means that the walker spanns more than 2 dimensions.

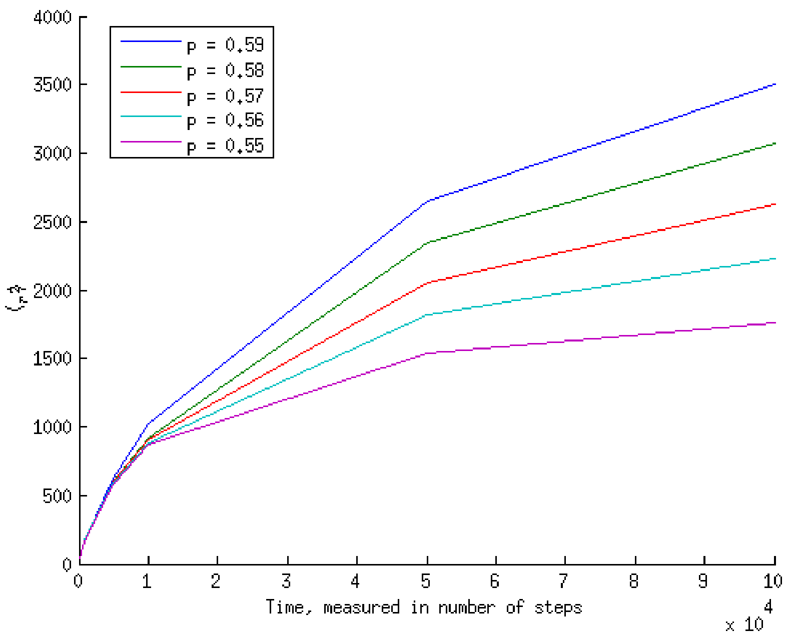


Figure 15: $\langle R^2 \rangle$ as a function of the number of steps done by the walker.