Computational Physics
Problem Set 3

Saleh Shamloo Ahmadi
Student Number: 98100872

October 17, 2021

1 Ballistic Deposition

Ballistic deposition models add particles randomly to a surface and in the case of non-random ballistic deposition models, apply specific rules after each "deposition".

In general, we can approximately describe the growth of each model with the constants α, β, and z (not independent); If there is any interaction between neighboring points, the roughness of the surface (defined as the standard deviation of heights in one-dimensional systems) will approach an upper limit w_s ($w(t)$ is the roughness) at *time of saturation* t_s. Then

$$w(t) \sim t^{\beta} (\text{before saturation}), \tag{1}$$

$$t_s \sim L^z, \quad w_s \sim L^{\alpha} \sim t_s^{\beta} \sim L^{z\beta}, \tag{2}$$

so we should have $\alpha = z\beta$. Note that depending on the method used to determine saturation time, β before saturation can be different form β obtained from saturation.

1.1 Ballistic Deposition with Relaxation

In this model, after each deposition, the particle is dropped onto a neighboring point if it is at a lower height (less particles have been deposited on that point). If more than one neighbor has a lower height, the particle is dropped onto the point with the lowest height.

The faint "bands" in each plot shows the standard deviation of each data point taken over all runs.

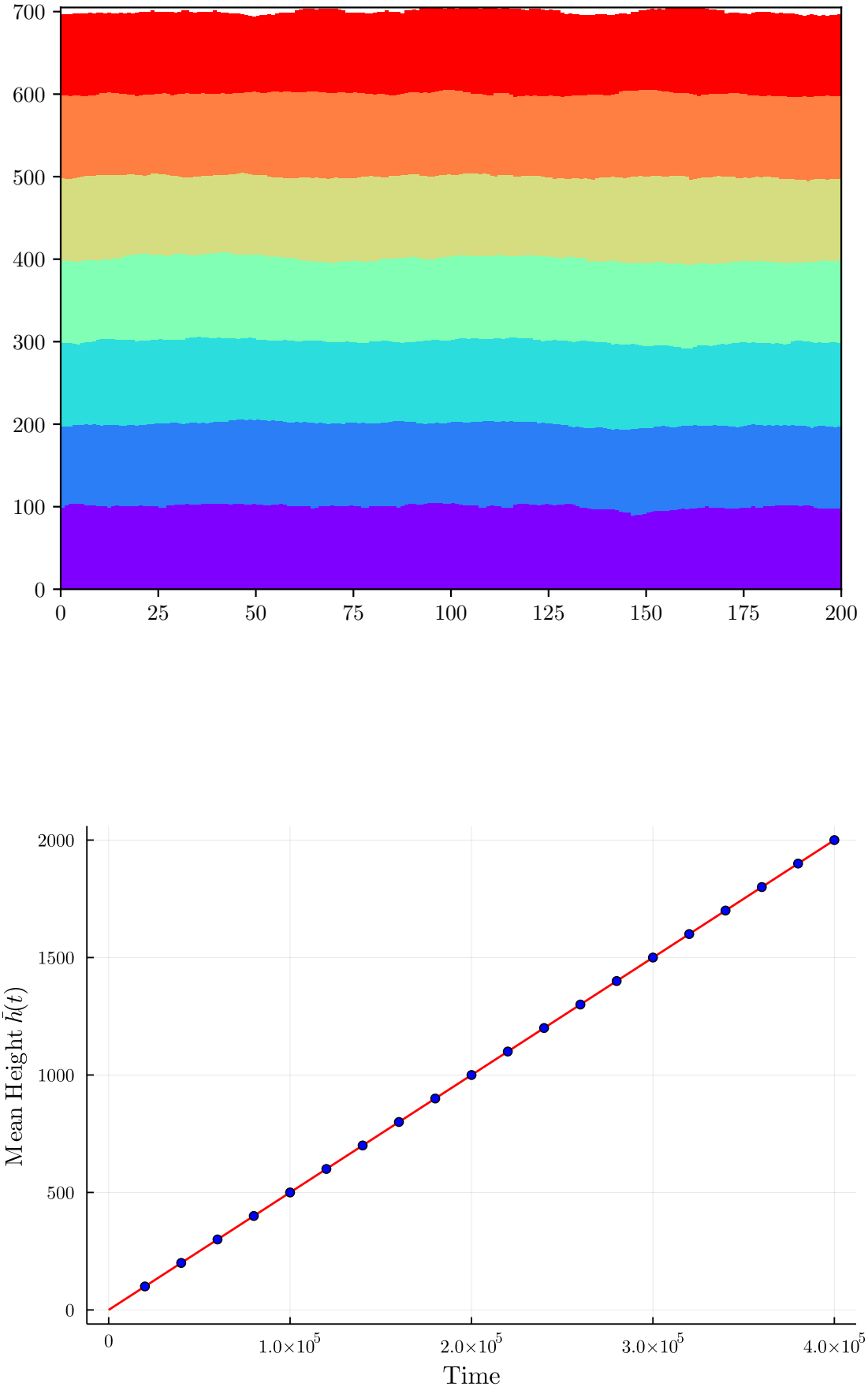


Figure 1: slope $= 0.005 \pm 6 \times 10^{-20}$

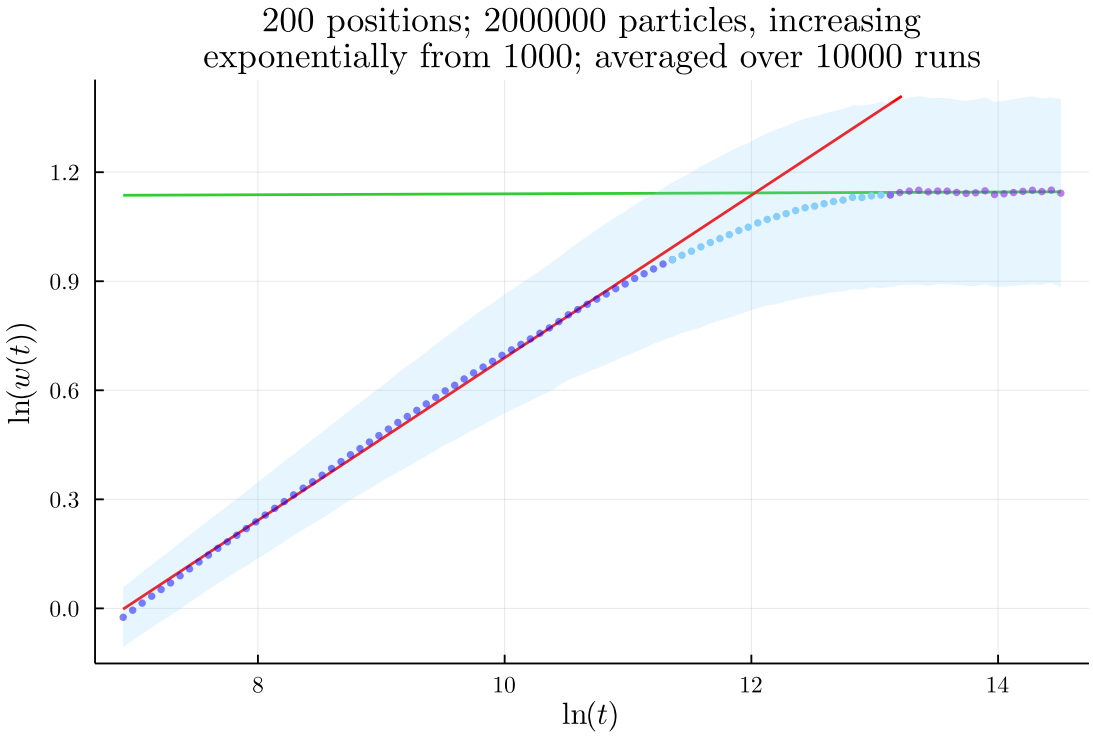
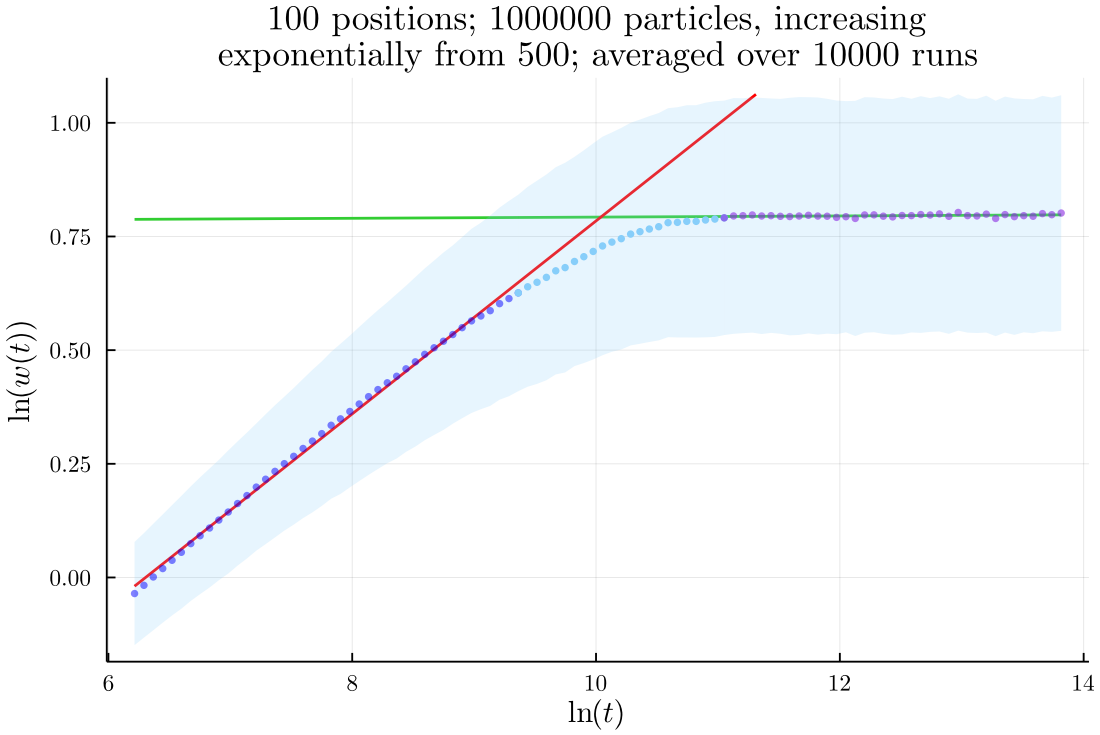
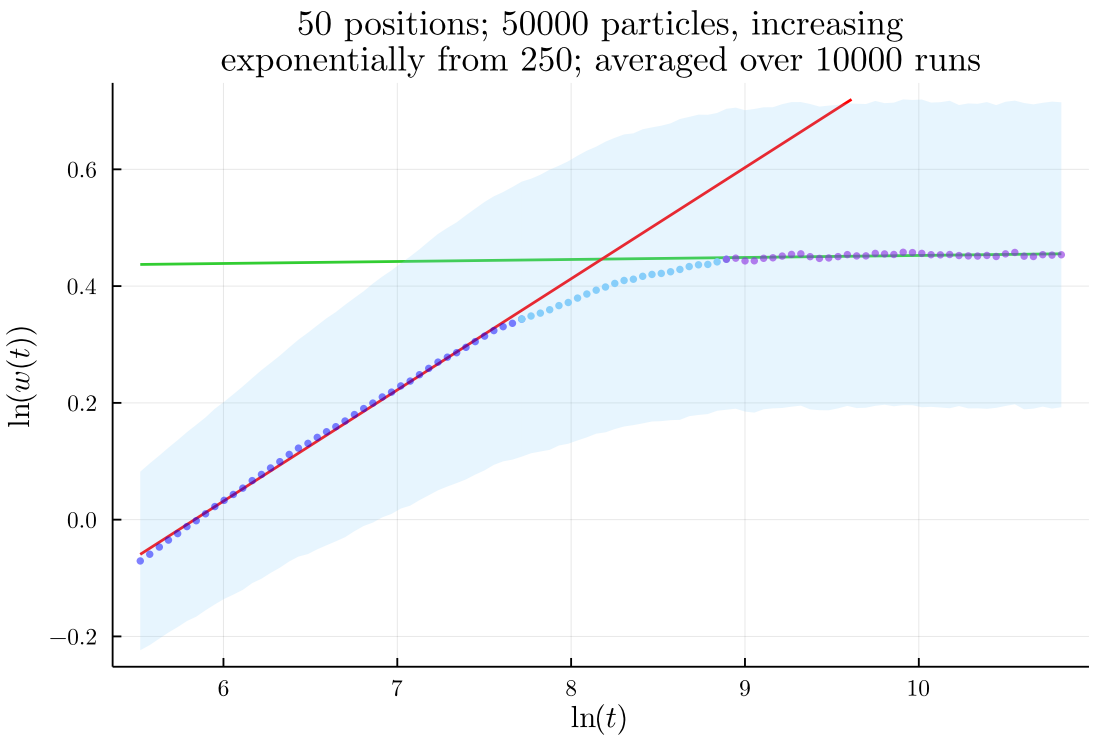
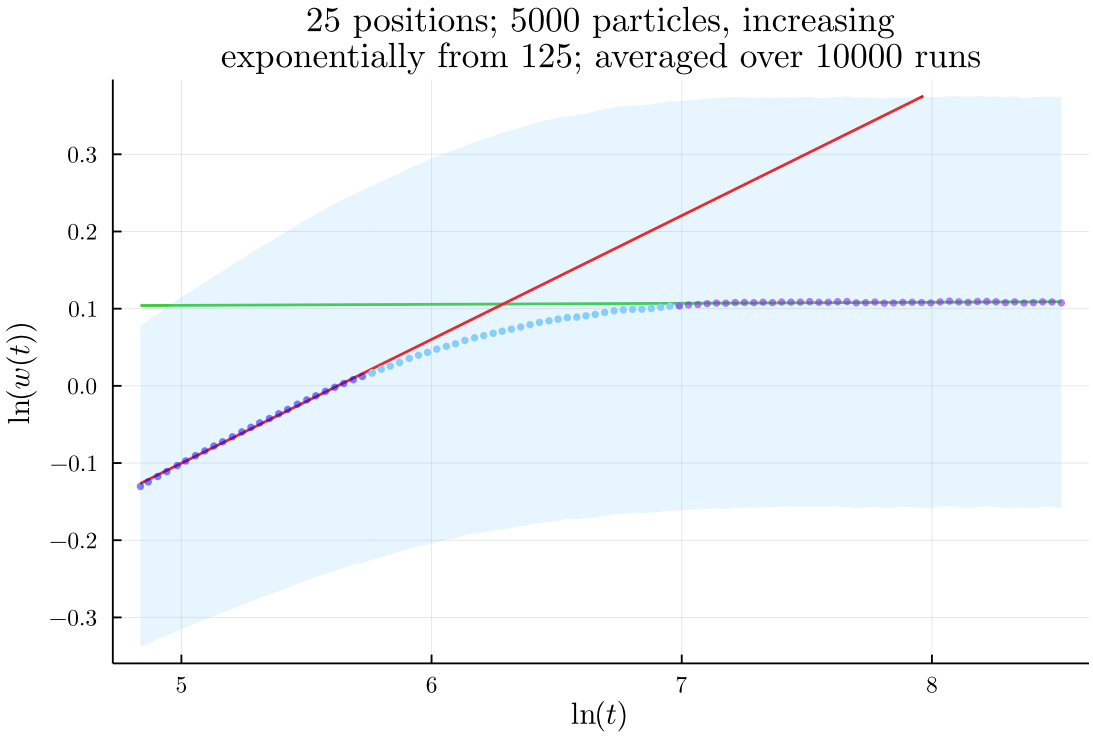
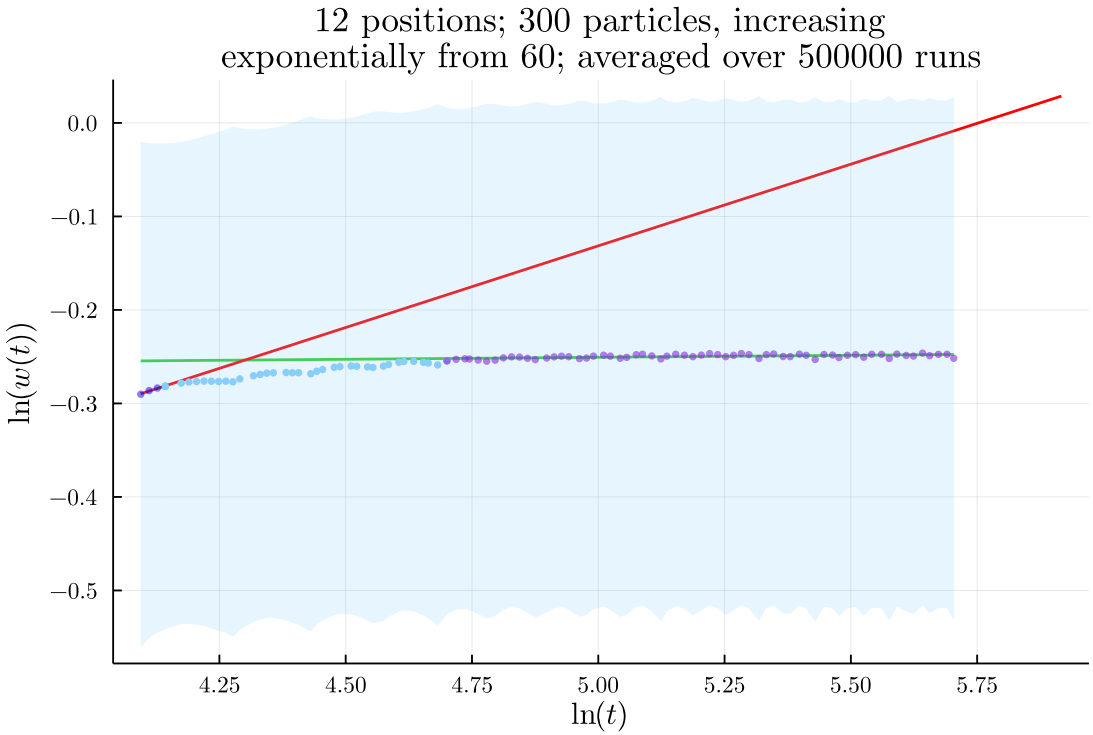


Figure 2: $\beta = 0.224 \pm 0.001$




50 positions; 50000 particles, increasing
exponentially from 250; averaged over 10000 runs
0.6
0.4
0.2
0.0
−0.2
ln(w(t))
6
7
8
9
10
ln(t)


25 positions; 5000 particles, increasing
exponentially from 125; averaged over 10000 runs
0.3
0.2
0.1
0.0
−0.1
−0.2
−0.3
ln(w(t))
5
6
7
8
ln(t)



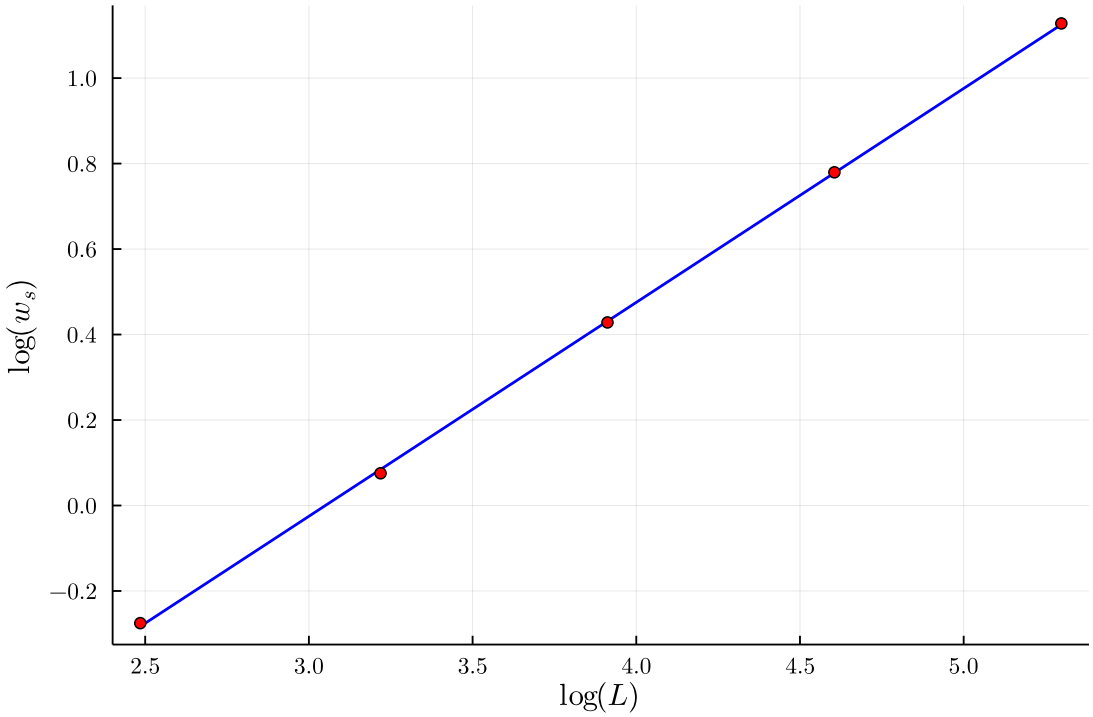


Figure 3: $\alpha = 0.500 \pm 0.003$

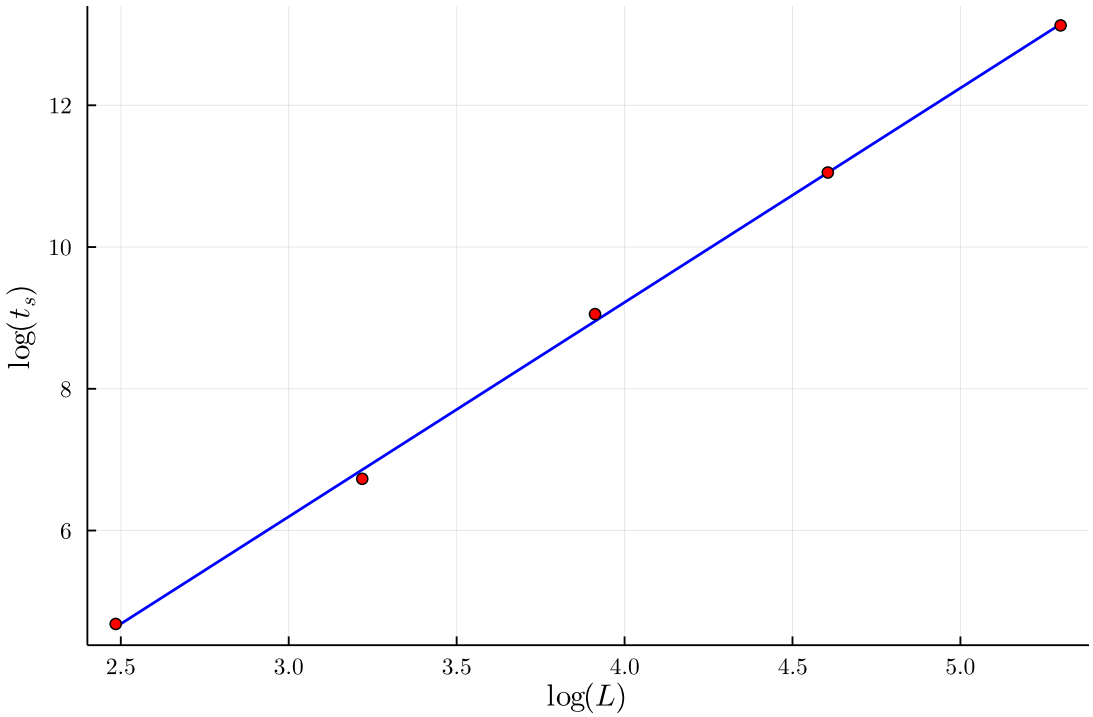


Figure 4: $z = 3.02 \pm 0.04$

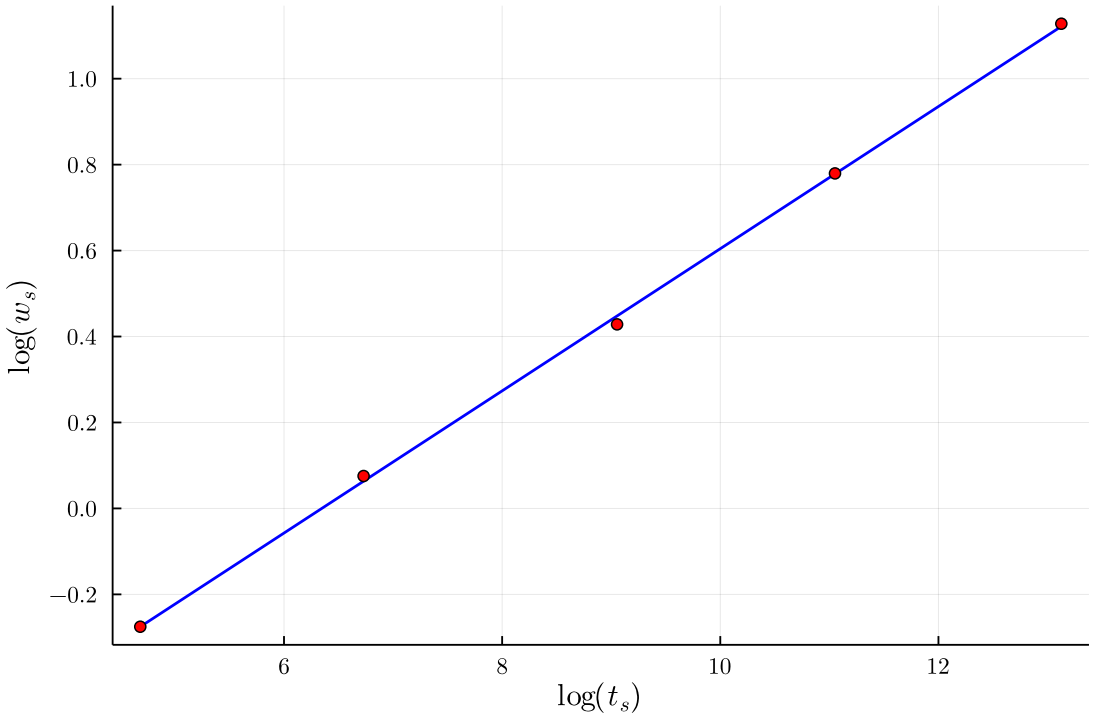


Figure 5: $\beta = 0.165 \pm 0.002$

Table 1: Ballistic Deposition with Relaxation

L	t_s	w_s
12	108	0.759364
25	837	1.07834
50	8550	1.53467
100	63042	2.18055
200	502161	3.08905

I calculated the time of saturation by fitting a line through the trailing points with constant $w(t)$. This method is not optimal. It would be better to find the intersection of the linear fit and constant fit lines. Due to a lack of time, I am unfortunately stuck with this (this was the first method I tried and there was no time left before the deadline to improve my methodology).

As you can see, this description of the model is not perfect, since β is not the same as L changes.

1.2 Ballistic Deposition (regular)

In this model, particles stick to the first point they come in contact with. This means each particle will be stuck at the maximum heigh in its dropping neighborhood ($\max\{h_{\text{left}}, h_{\text{drop}} + 1, h_{\text{right}}\}$).

Results have similar characteristics to the last model. Notably, β obtained from saturation is equal within the margin of error and β obtained from ftting to data is also very close. α and z are smaller, since correlation is stronger in this model (height of each point is directly related to the neighboring heights, instead of being indirectly affected by them through the distribution of heights).

Table 2: Ballistic Deposition (regular)

L	t_s	w_s
12	107	2.42859
25	406	2.84437
50	2715	4.20796
100	12386	4.94841
200	65003	6.75316

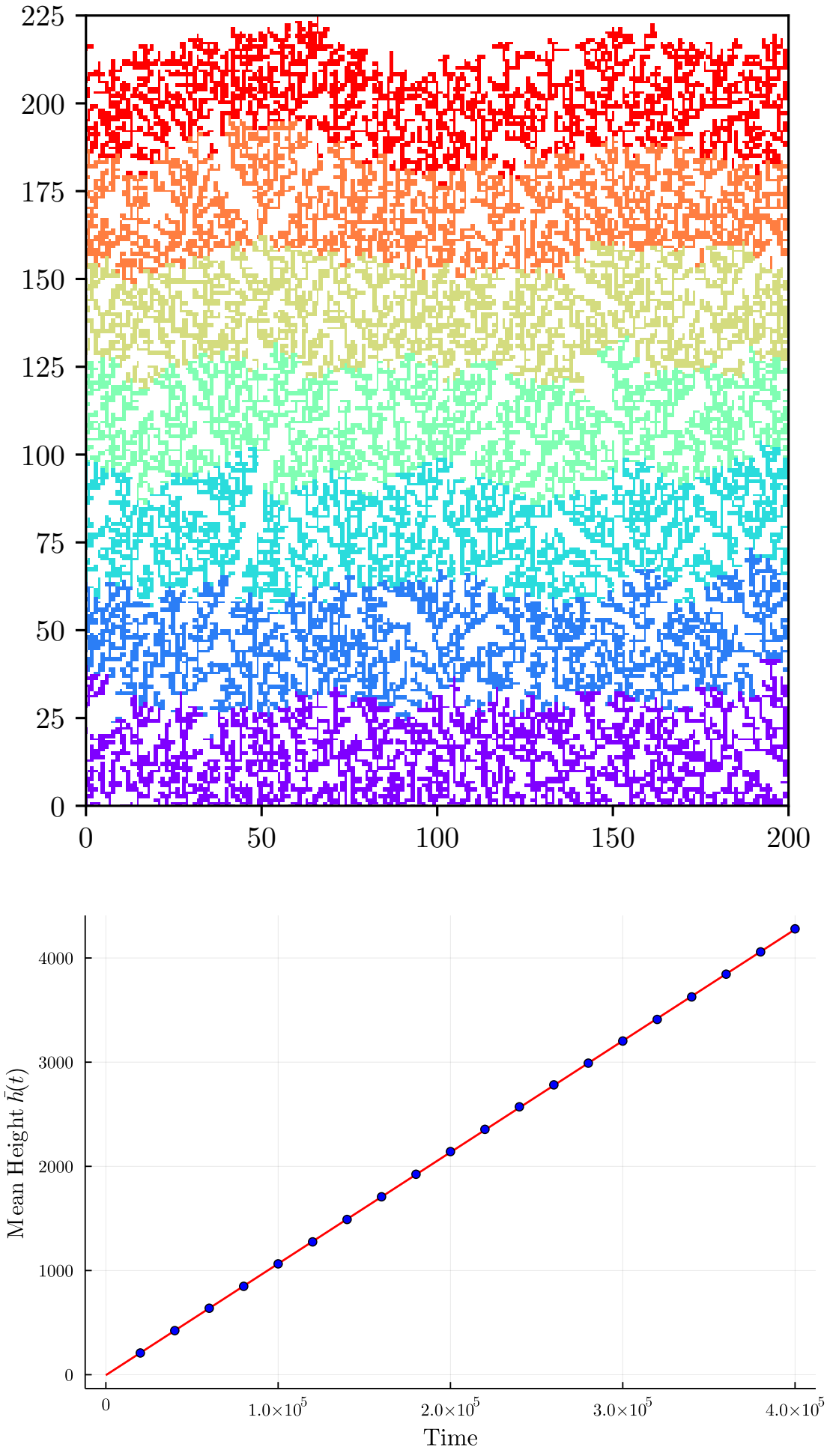
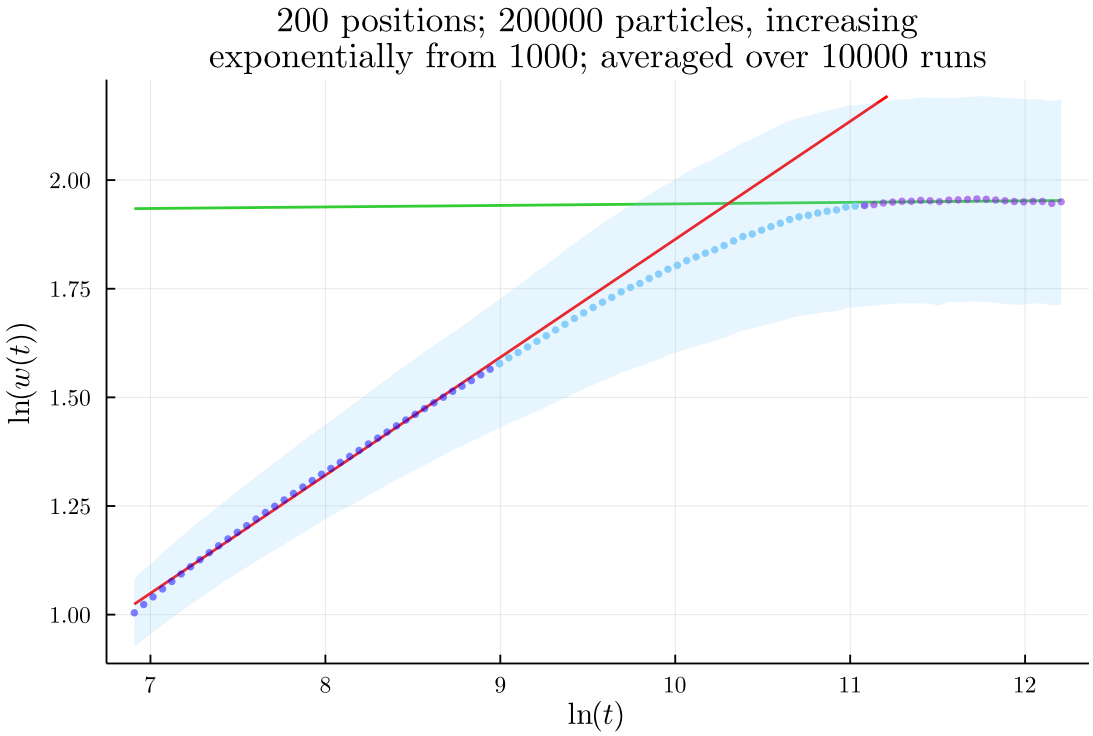


Figure 6: slope $= 1.0698 \times 10^{-2} \pm 9 \times 10^{-6}$



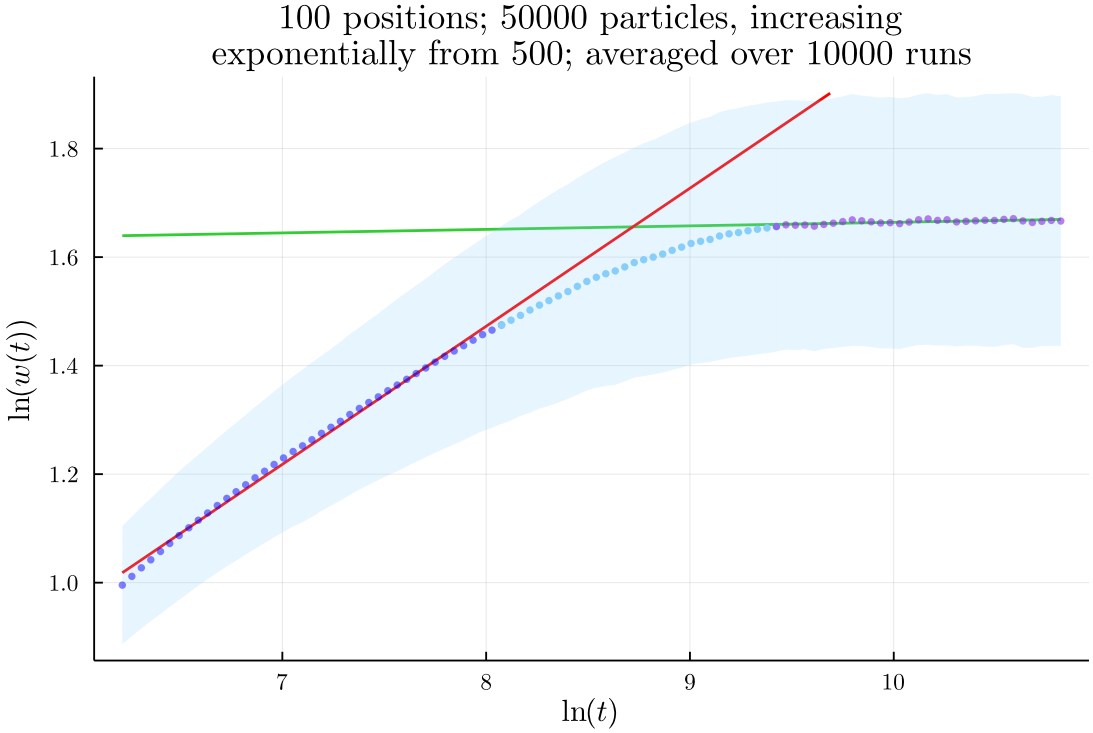
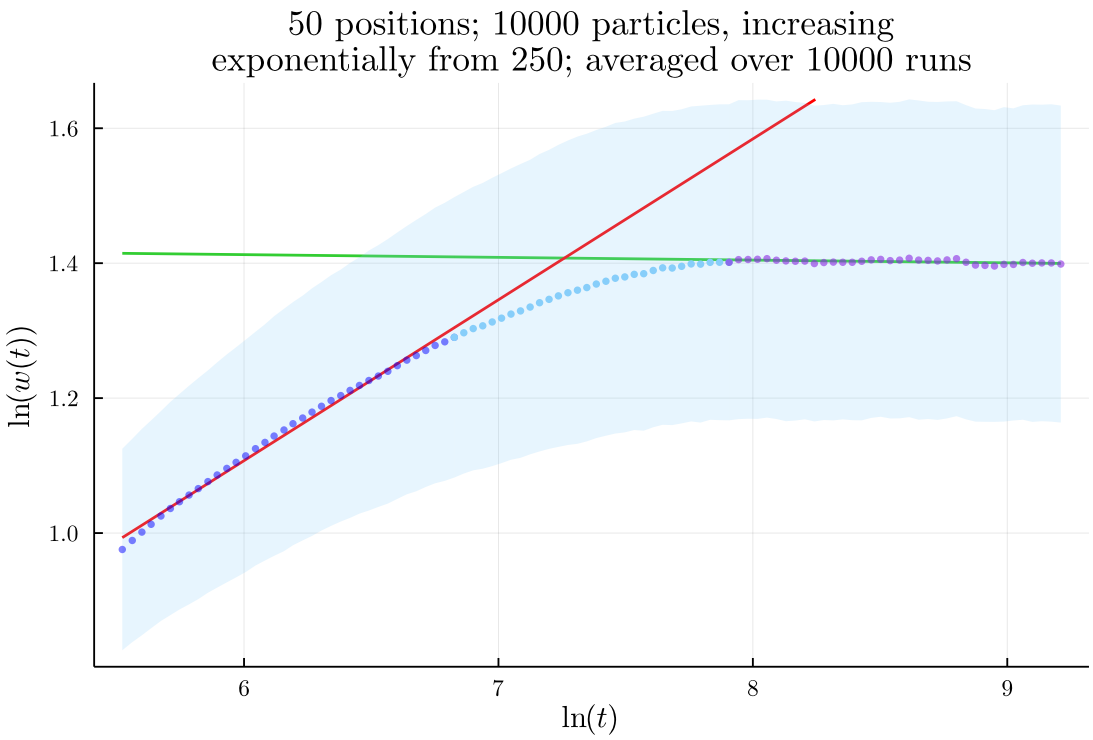
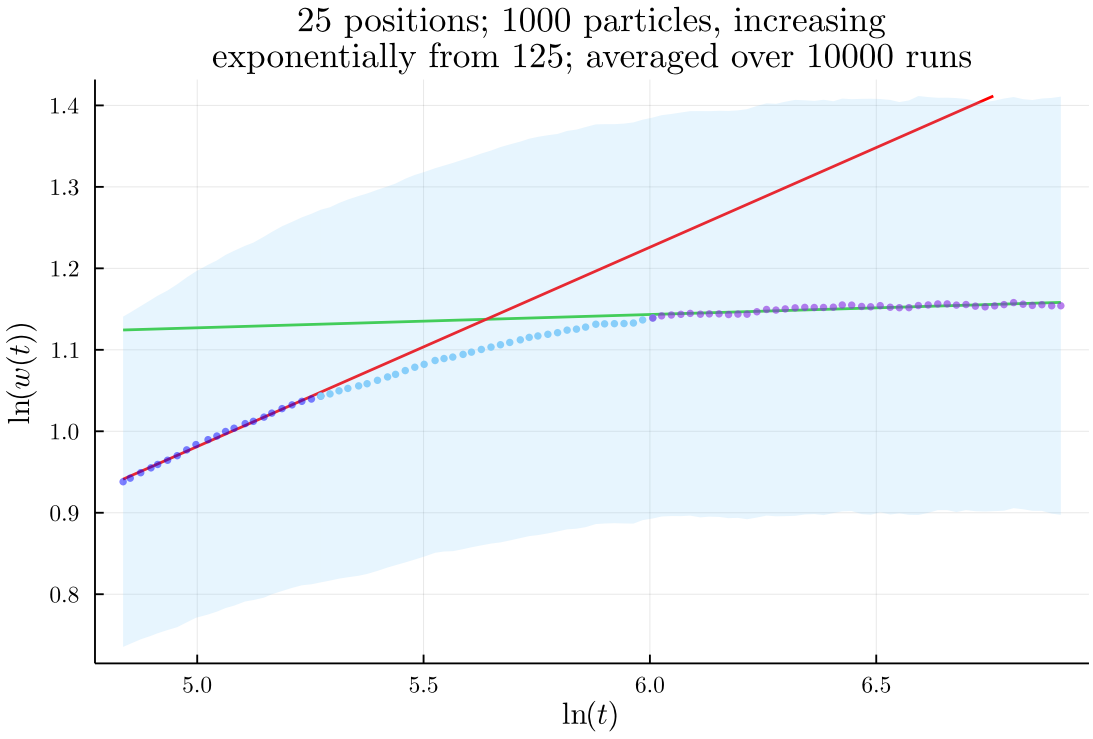
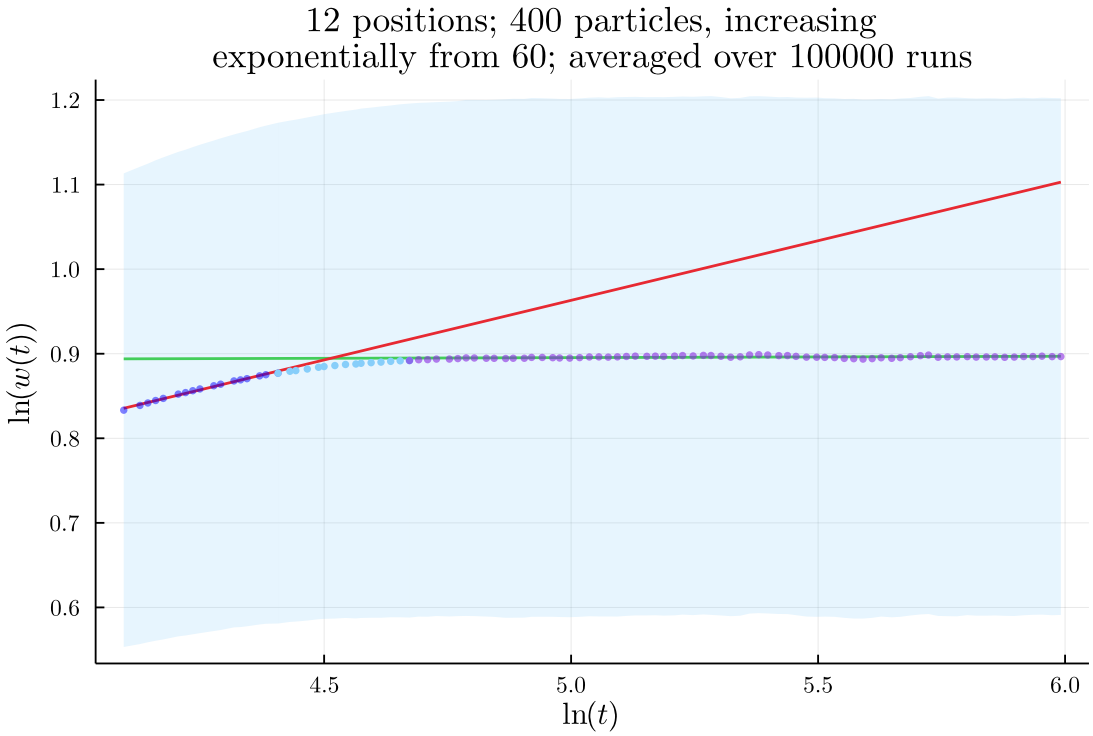


Figure 7: $\beta = 0.254 \pm 0.003$


50 positions; 10000 particles, increasing
exponentially from 250; averaged over 10000 runs
1.6
1.4
1.2
1.0
ln(w(t))
6
7
8
9
ln(t)


25 positions; 1000 particles, increasing
exponentially from 125; averaged over 10000 runs
1.4
1.3
1.2
1.1
1.0
0.9
0.8
ln(w(t))
5.0
5.5
6.0
6.5
ln(t)



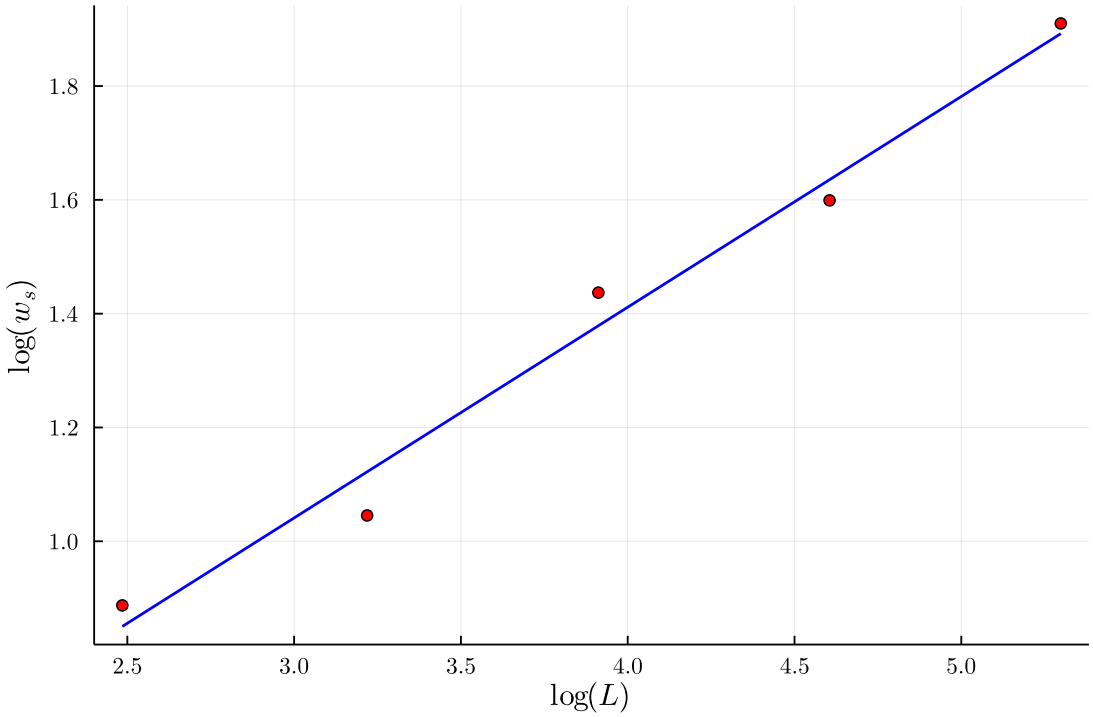


Figure 8: $\alpha = 0.37 \pm 0.03$

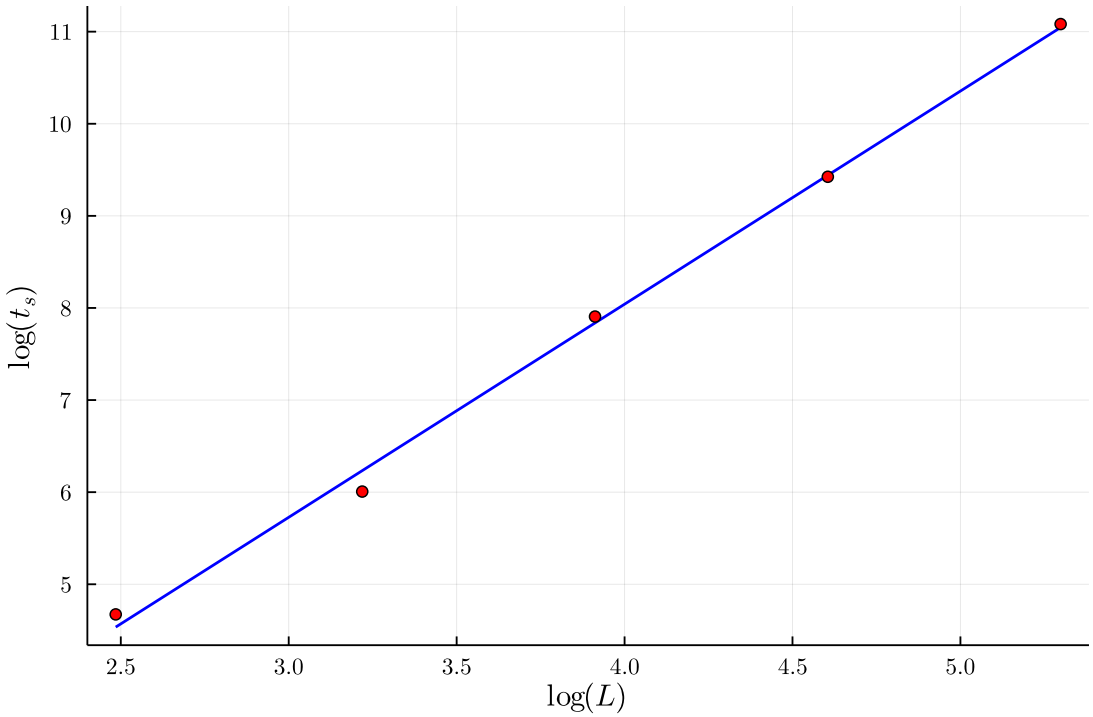


Figure 9: $z = 2.31 \pm 0.07$

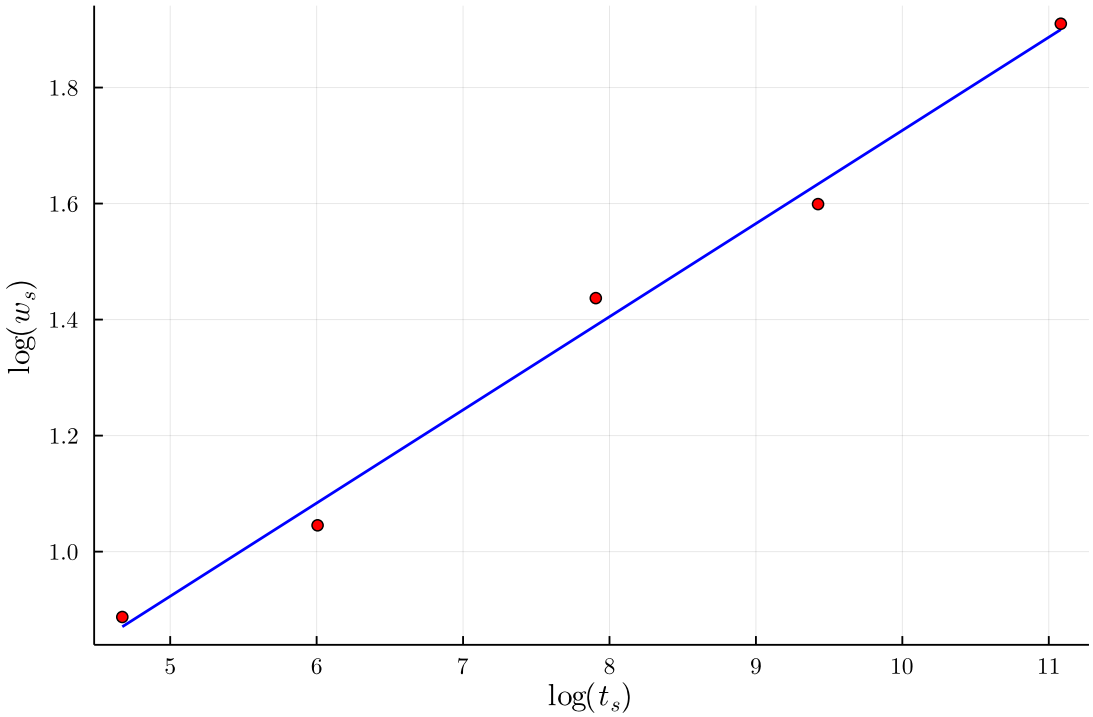
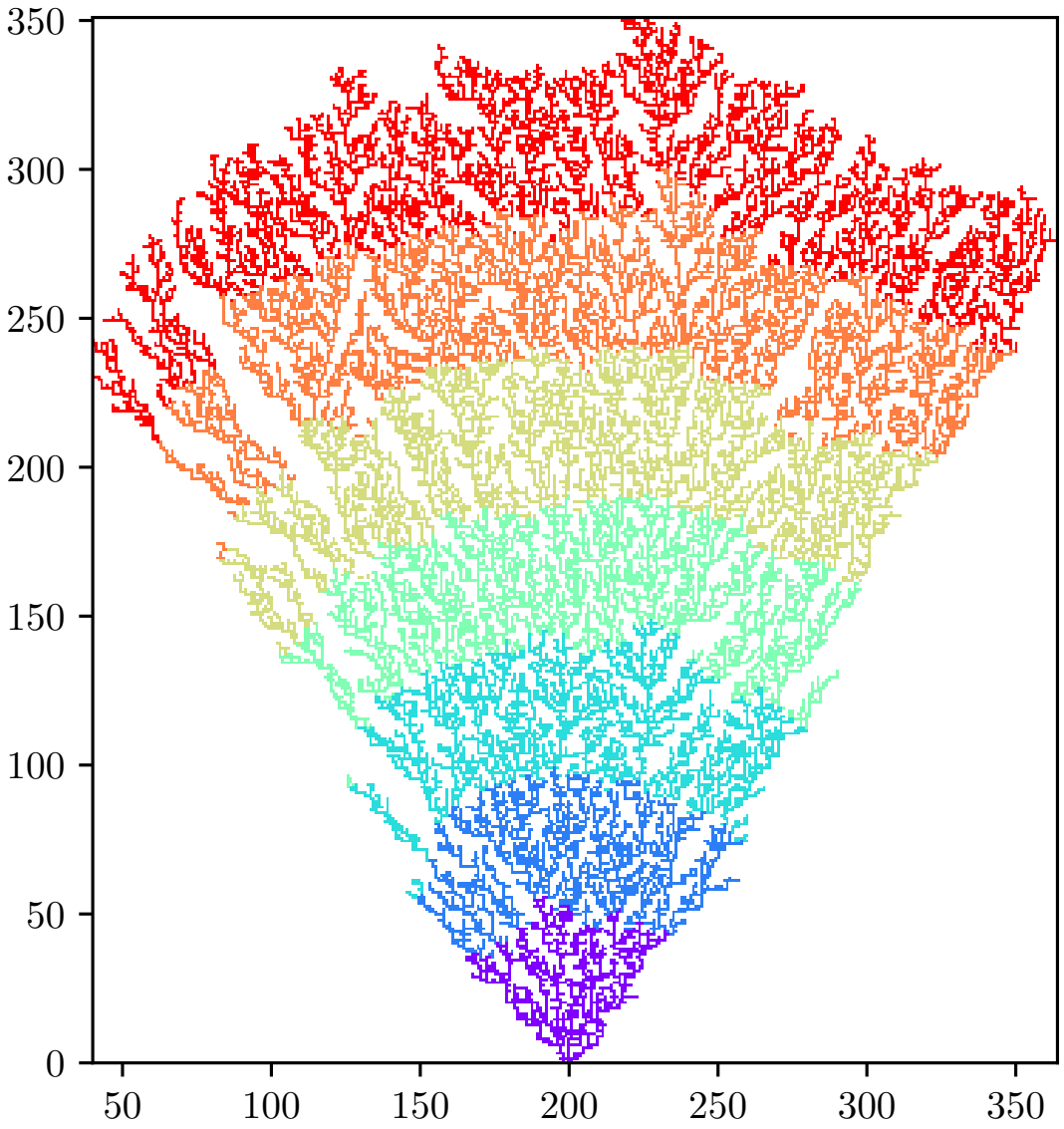


Figure 10: $\beta = 0.161 \pm 0.008$



1.2.1 Correlation Length

In the ballistic deposition model, each point can have long range effect on further points through clustering with its neighbors, which in turn cluster with their own neighbors, and so on. To find the range of interaction, or *the correlation length*, we can isolate the particles that stem from a single point (the "tree" attached to a "seed").

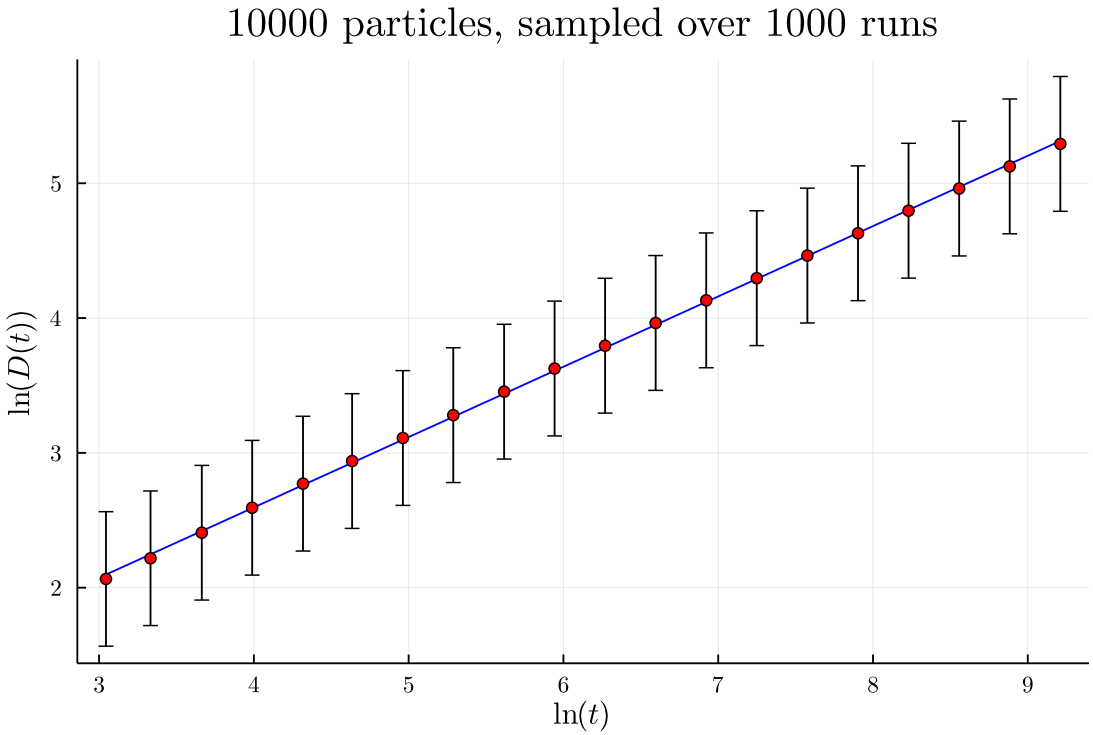


Figure 11: slope $= 0.522 \pm 0.002$.
The slope is the growth exponent for of the correlation length.

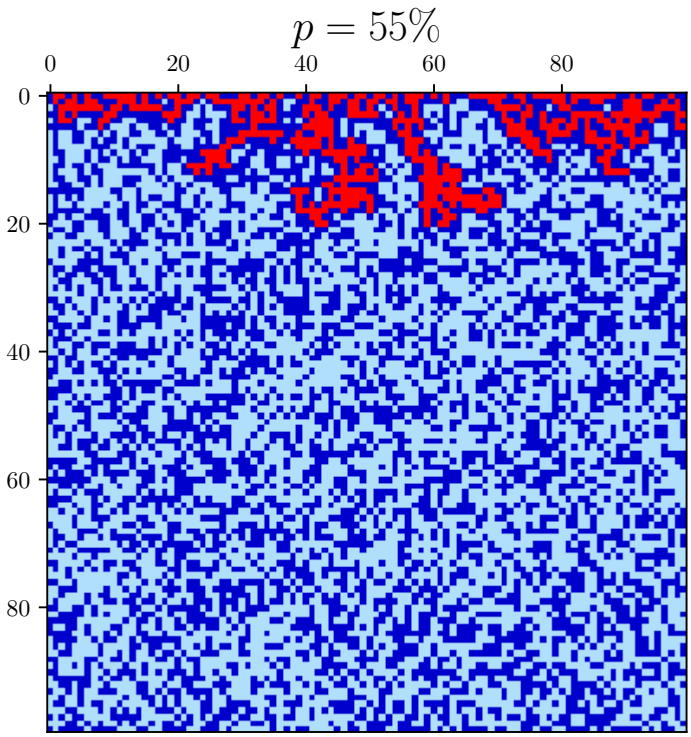
2 Percolation

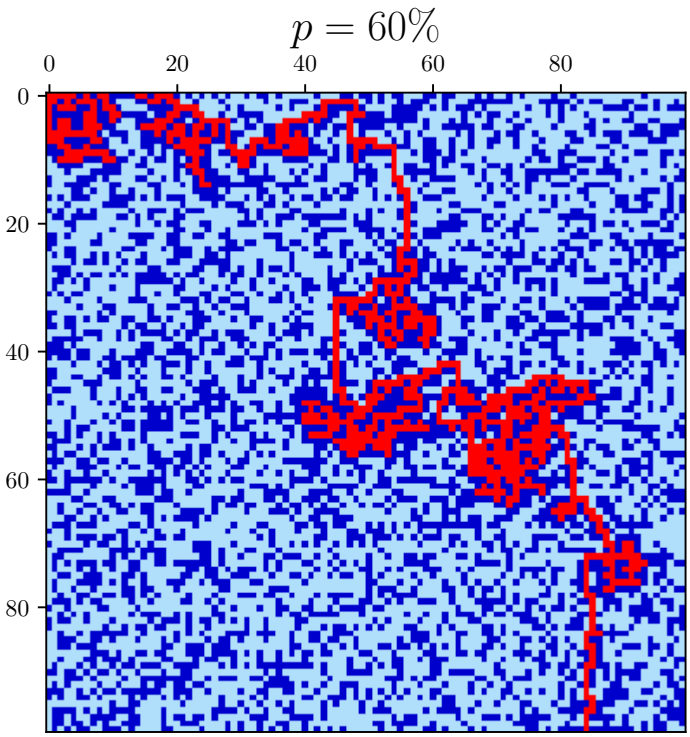
2.1 Depth-First Search

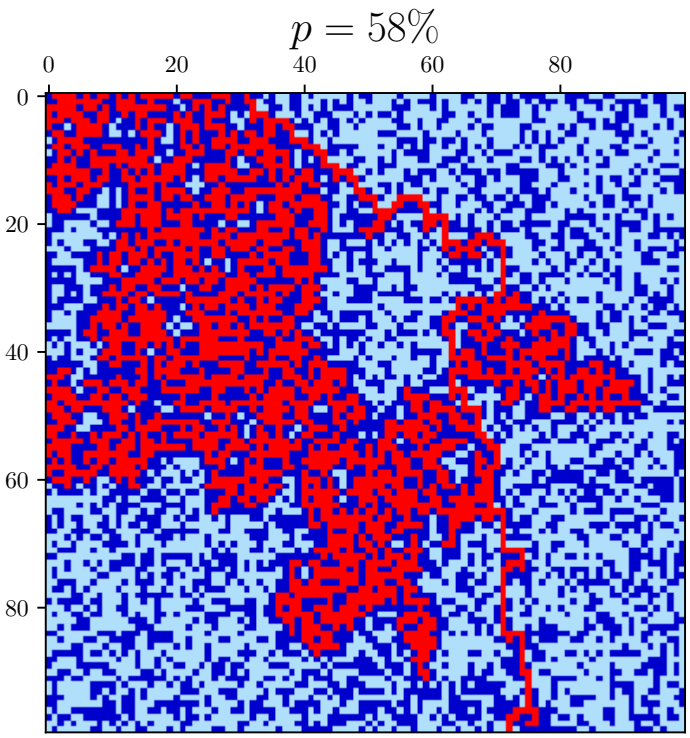
Using a modified version of the depth-first search algorithm (a.k.a. DFS) that is designed and optimized for grids, percolation problems can be solved efficiently (time complexity $\mathcal{O}(L^d)$, where L is the lenght of the grid and d is the number of dimentions). Algorithm 1 outlines the implementation.

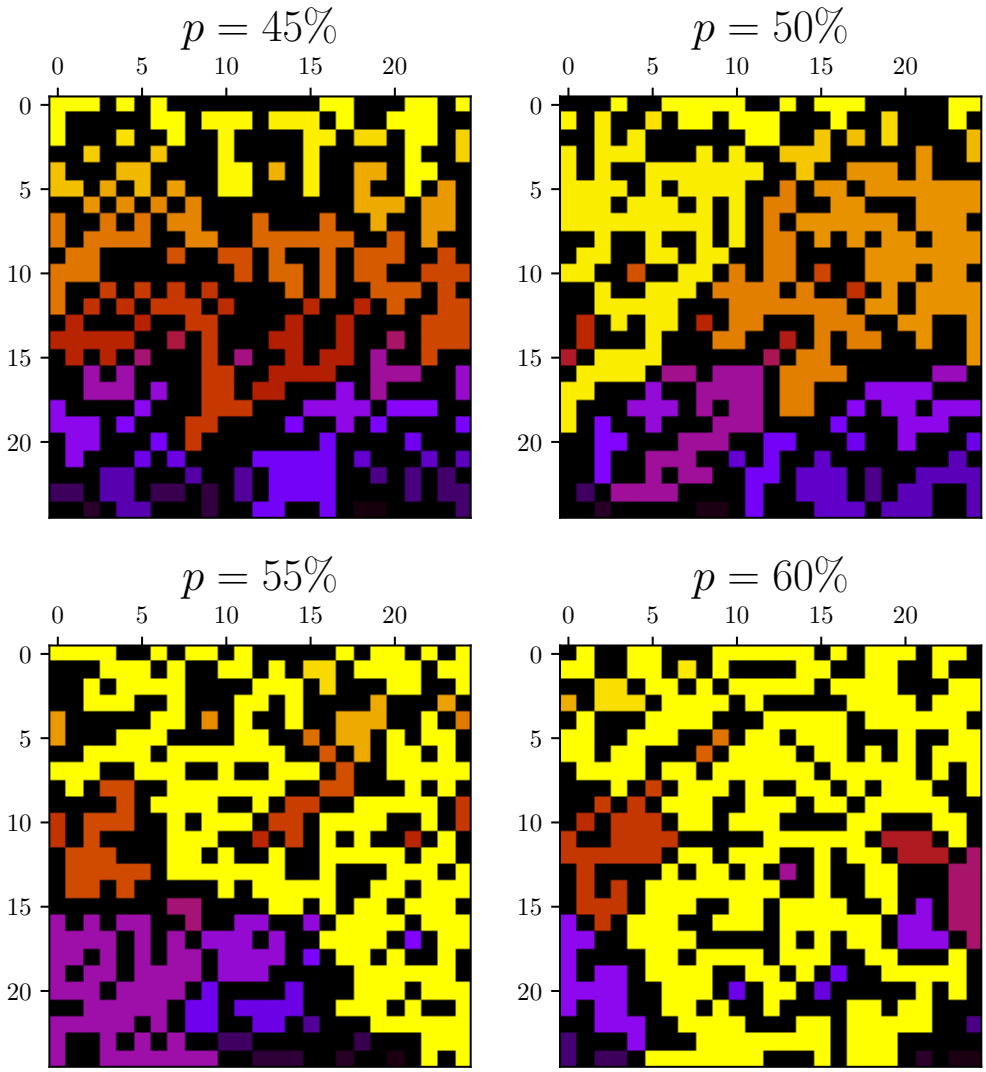
Algorithm 1 DFS for solving a percolation problem

```
1:  function DFS(G) ▷ G is a boolean graph representing the empty/full states
                      of each cell of the grid
2:      make stack S
3:      for all vertices v in the starting row/column of the grid do
4:          if v is true then
5:              S.push(v)
6:              while S is not empty do
7:                  v = S.pop()
8:                  if v is in the final row/column then
9:                      return true
10:                 else if v is not already visited then
11:                     mark v as visited
12:                     for all w in G.neighbors(v) do
13:                         S.push(w)        ▷ Prioritize the neighbor in the direction
                                               of the percolation's destination to optimize the
                                               performance for emptier grids
14:                     end for
15:                 end if
16:             end while
17:         end if
18:     end for
19:     return false
20: end function
```


$p = 55\%$
0
20
40
60
80


$p = 60\%$
0
20
40
60
80


$p = 58\%$
0
20
40
60
80



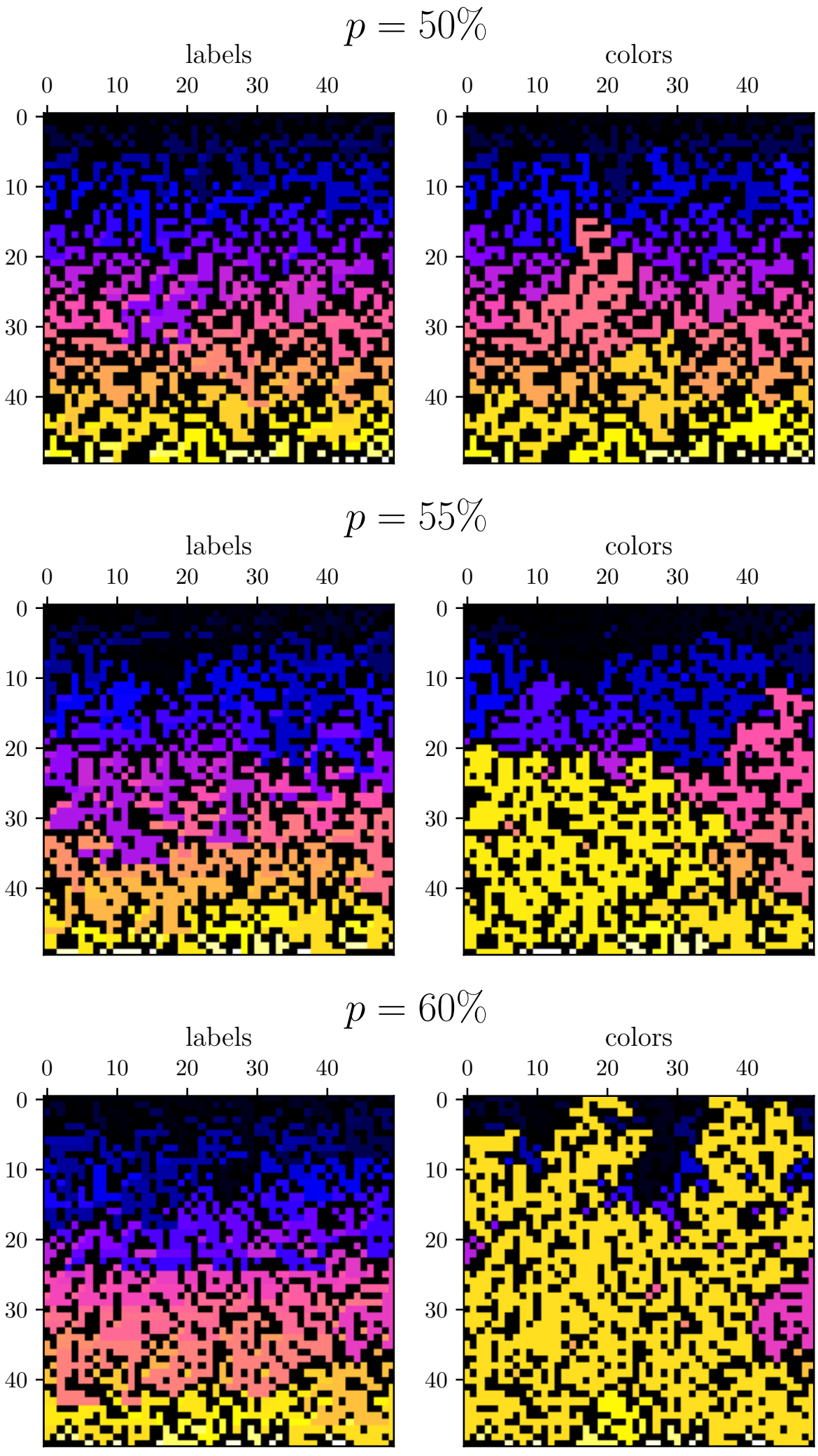
2.2 Coloring

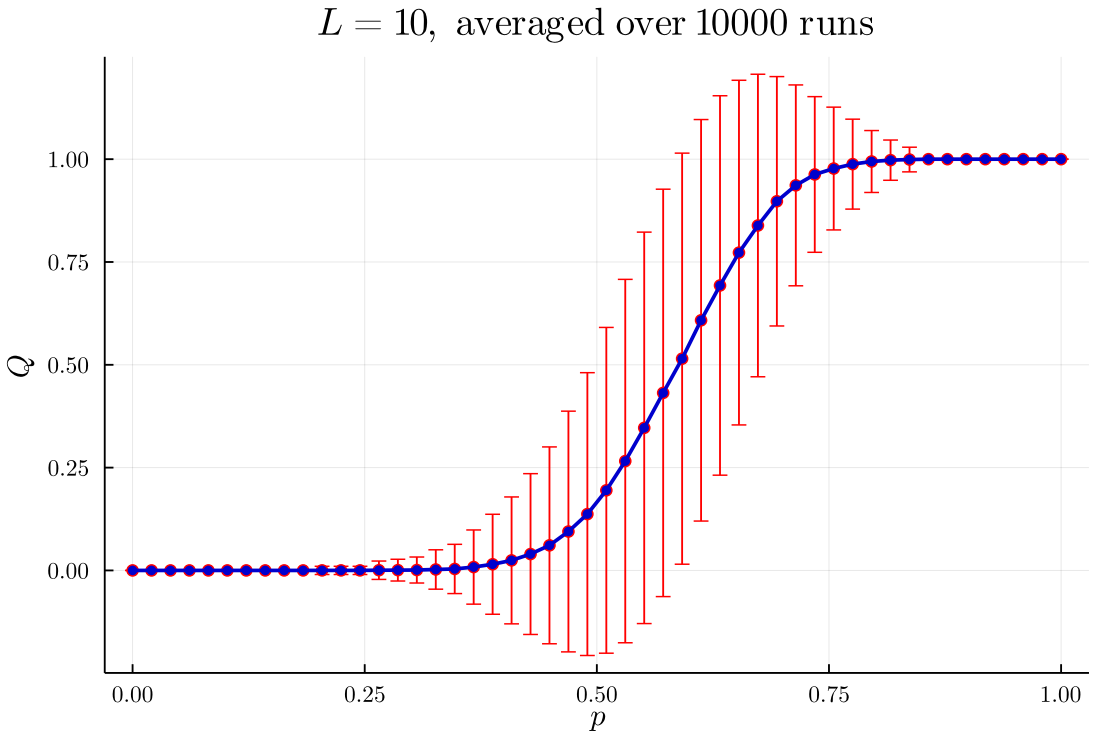
We can use (a very inefficient[1]) coloring algorithm, as described in the lecture notes, to label the clusters in the grid. Then, If a cluster connecting the two ends of the grid exists, percolation is possible.

2.3 Hoshen–Kopelman

This is a much more efficient $\mathcal{O}(L^d)$ algorithm for labeling the clusters. You can read more about it on the Wikipedia page.

[1] $\mathcal{O}(L^{2d})$ time complexity


$p = 50\%$
labels
colors
$p = 55\%$
labels
colors
$p = 60\%$
labels
colors
0
10
20
30
40



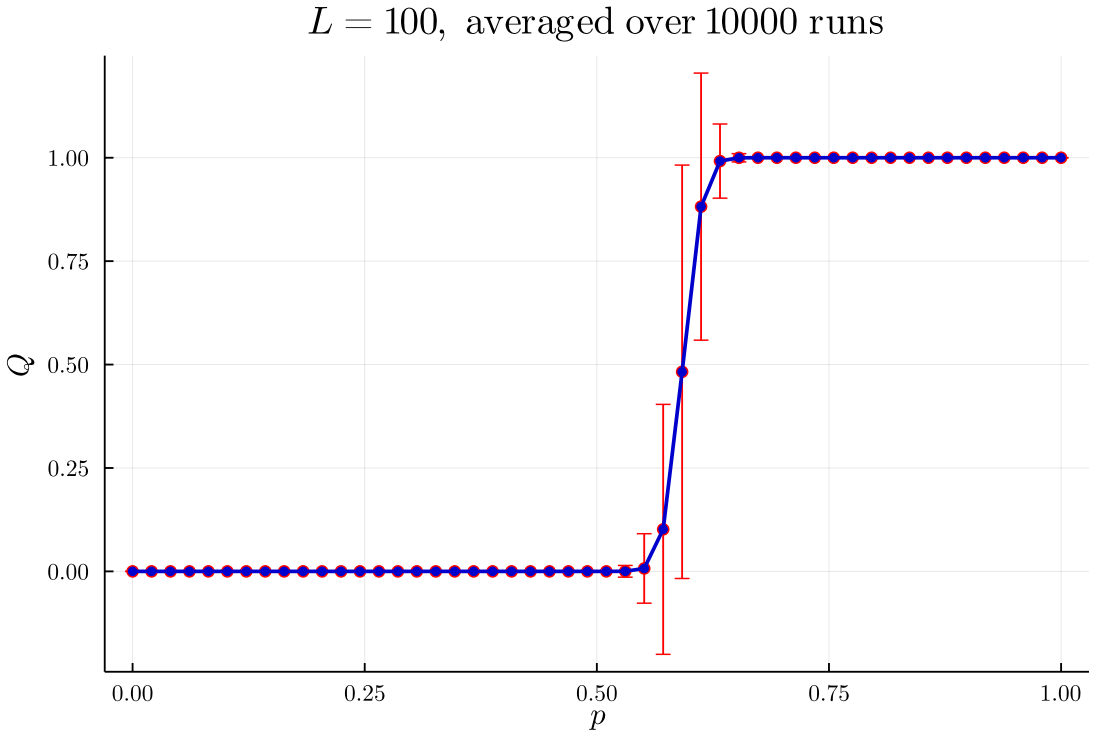
2.3.1 Percolation Probability

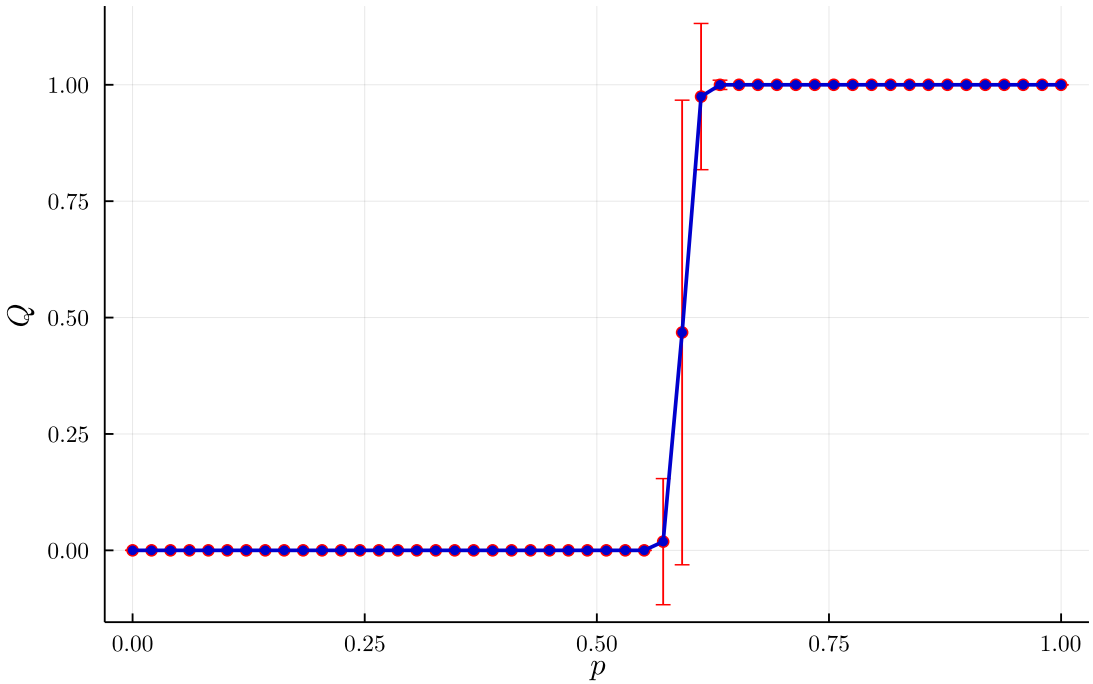
The probablity of there existing a path from the top to the bottom of a lattice (denoted by Q) exhibits critical behavior; If p is the probablity of forming bonds (or sites) in the lattice, then is a sudden jump from 0 to 1 probablity at a critical probablity p_c. In an infinite lattice, the path connecting the top and bottom of the lattice is called the *infinite open cluster*.

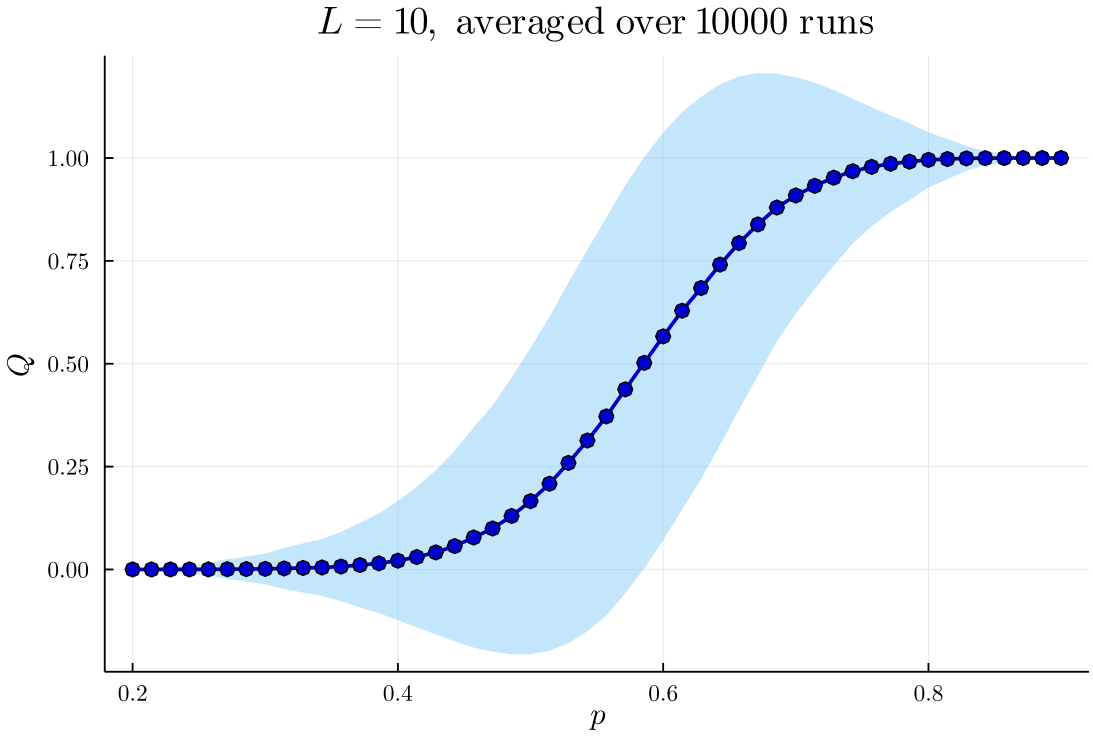
As you can see, the transition from 0 to 1 probablity at p_c becomes sharper as the side-length of the lattice increases.

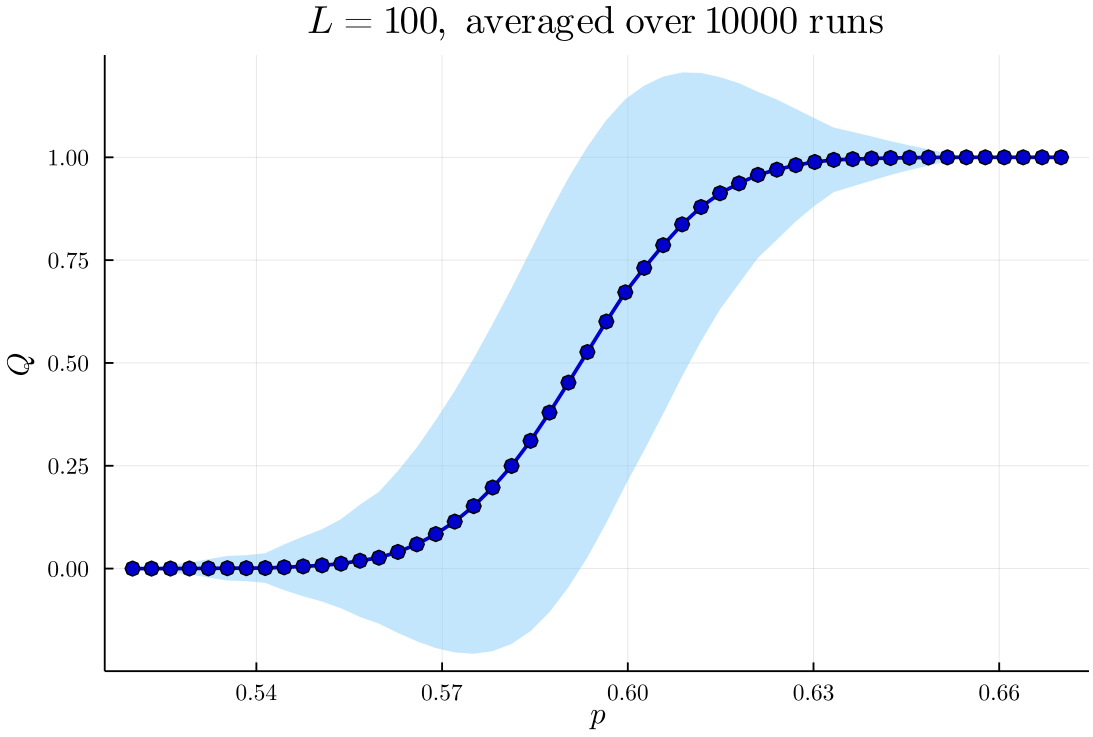
We use Q_∞ to denote the probablity of a cell being a part of an open cluster connecting the top and bottom of the lattice. To calculate Q_∞, we can modify the Hoshen–Kopelman algorithm to record cluster sizes for every label (that is, the "root" labels, or the final label that the cluster is connected to, such that Label(number) = number). Then, Q_∞ will be the sum of the sizes of the open clusters devided by the surface area of the lattice.

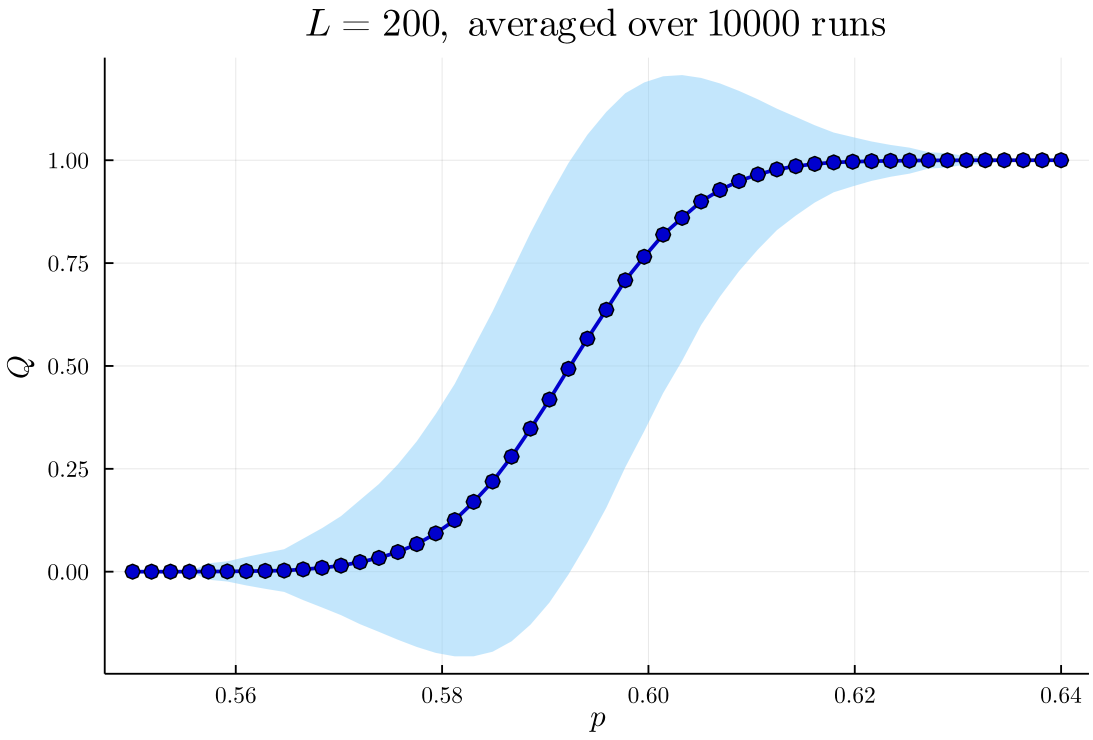
As you can see, Q_∞ exhibits the same critical behavior as Q.

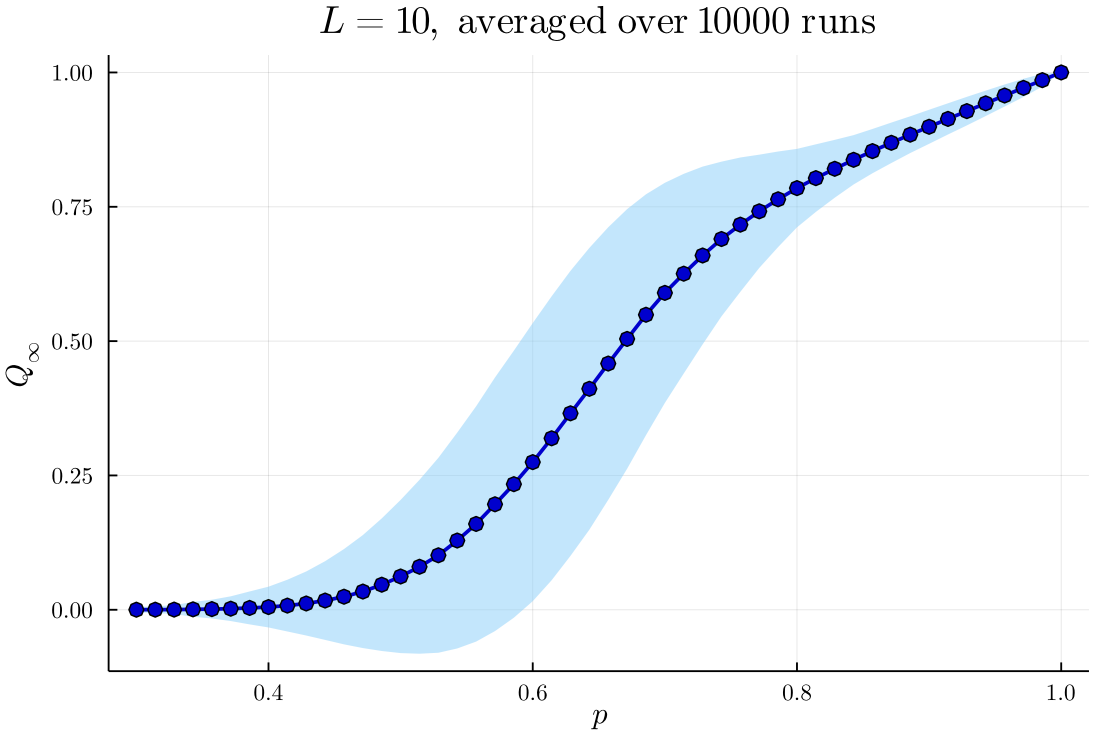

$L = 100,$ averaged over 10000 runs
1.00
0.75
0.50
0.25
0.00
Q
0.00
0.25
0.50
0.75
1.00
p

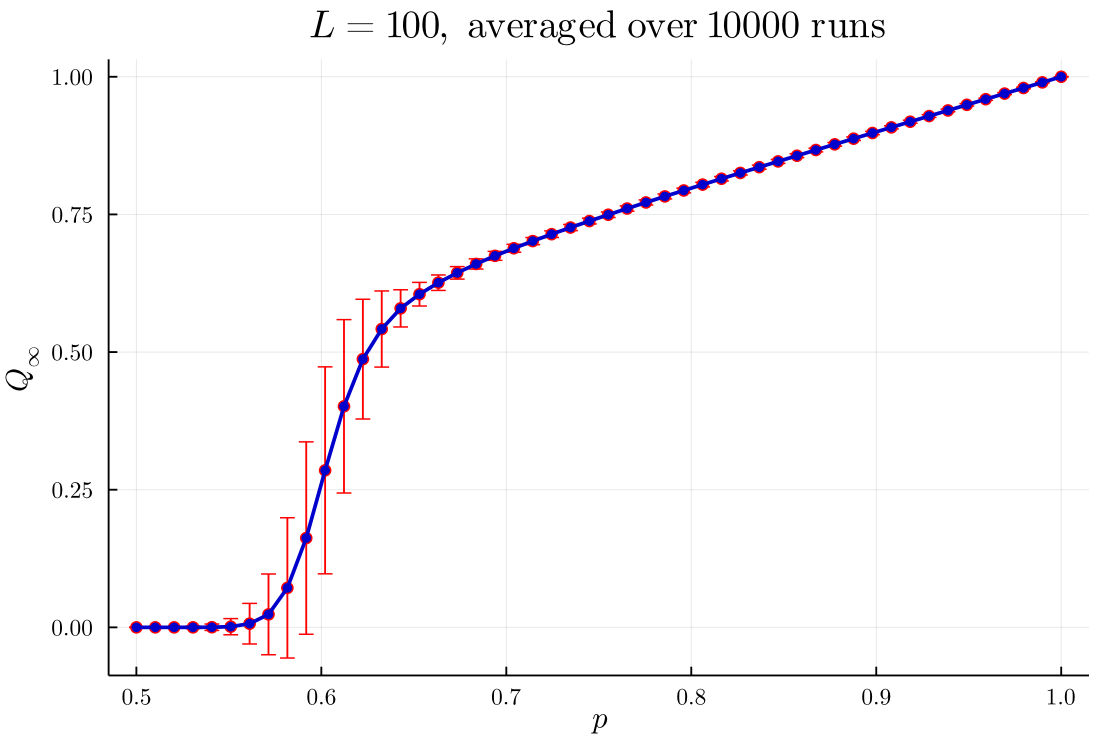

$L = 200,$ averaged over 10000 runs
1.00
0.75
0.50
0.25
0.00
Q
0.00
0.25
0.50
0.75
1.00
p


$L = 10,$ averaged over 10000 runs
1.00
0.75
0.50
0.25
0.00
Q
0.2
0.4
0.6
0.8
p


$L = 100,$ averaged over 10000 runs
1.00
0.75
0.50
0.25
0.00
Q
0.54
0.57
0.60
0.63
0.66
p


$L = 200,$ averaged over 10000 runs
1.00
0.75
0.50
0.25
0.00
Q
0.56
0.58
0.60
0.62
0.64
p


$L = 10,$ averaged over 10000 runs
1.00
0.75
0.50
0.25
0.00
Q_∞
0.4
0.6
0.8
1.0
p


$L = 100,$ averaged over 10000 runs
1.00
0.75
0.50
0.25
0.00
Q_∞
0.5
0.6
0.7
0.8
0.9
1.0
p