A Cellular Automata Model on Urban Land Use Dynamics

Aidan Claffey

15th May, 2018

1 Why model Urban Land Use?

As the global population continues to rise, more and more people are moving to cities, and more and more cities are growing. For example, according a *The Guardian*article titled "The great sprawl of China: timelapse images reveal 30-year growth of cities," there are 8 Chinese cities with current populations above 1 million people that saw population growths of over 1,000

While there are many different land use models out there, some rely heavily on computational power and are therefore less easy to analyze from a mathematical modeling perspective. The cellular automata model, described below, used in this paper relies on rules that determine how land changes from one type to another and an additional stochastic element.

2 What is a Cellular Automata (CA) model?

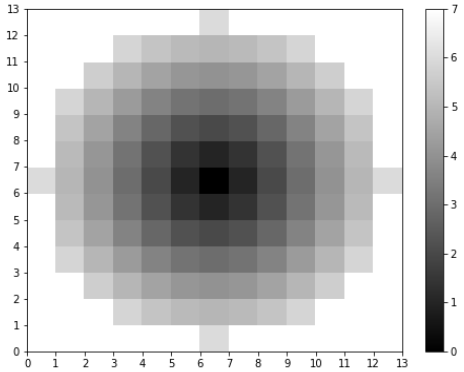
A cellular automata model essentially is an n-dimensional array of cells whereby each cell in the grid is of a certain state. For example, a cell's state space, Ω , could be as simple as $\{on, off\}$, or it could be as complicated as $\Omega = \{1,...,SN\}$ however it must be a discrete space. In a cellular automata model, each cell's state depends on the states of the neighboring cells. Furthermore, a set of transition rules will cause states to change depending on both a cell's own state and the states of neighboring cells. Thus, the neighborhood of a cell must be defined within the model. Two other key factors in a CA model are that time is discrete and all cell's states are updated simultaneously. CA models are not just used for urban growth. Since they were introduced in the 1940s, they have been used to tackle problems in many dynamic processes, like the spreading of epidemics and forest fires or the creation of chemical reactions. CA models look to characterize spatial processes rather than linear ones. (R White, G Engelen 1176-77).

3 Defining White and Engelen's Land-Use model

The model that this paper analyzes is based on a model created by White and Engelen in 1993, in a paper titled "Cellular automata and fractal urban form: a cellular modelling approach to the evolution of urban land-use patterns." Their model is defined in the following way:

1. State space
    - Each cell represents an area of land, with the cell's state representing the land's purpose.
    - A cell's state space is defined as the following: $\Omega = \{Vacant, Housing, Industrial, Commercial\}$
2. Neighborhood
    - The neighborhood, N, of a cell, i.e. which cells affect the cell in question, lays in a radius of six units around the cell, based on euclidean distance. Thus, there are 113 cells in the neighborhood.

A visualization of the neighborhood is shown below. Darkened cells are part of the neighborhood.



3. Transition rules:

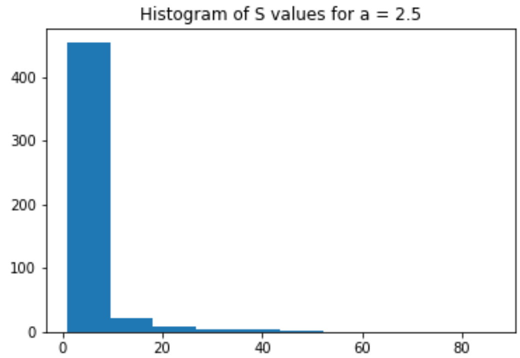
- In each iteration (i.e. from time t to time $t+1$), the number of cells that are allowed to change to each state, given in the paper by $N_i, (i = H, I, C)$, is determined *exogenously*. (At first it seemed odd that this would be determined *exogenously*, however, the capacity for cities to grow is not based on the shape of the city, but rather economic conditions facing the city)
- An initial transition ratio and a growth rate are given as parameters, so each $N_i, (i = H, I, C)$ grows by the given growth rate each iteration.
- Cells can only change from lower states to higher states. The order is *vacant*, *housing*, *industry*, and *commerce*.
- At each iteration, each cell must have potentials calculated for each possible transition. Therefore, every *vacant* cell will have three potentials calculated for it (*vacant to housing, vacant to industry, vacant to commerce*), every *housing* cell will have two potentials calculated for it (*housing to industry, housing to commerce*), and every *industry* cell will have one potential calculated for it (*industry to commerce*).
- Each potential is calculated as follows:

$$P_x(ij) = S(1 + \sum_{y \in N} m(state(y), ij, \delta) \tag{1}$$

  where
  $P_x(ij)$ is the transition potential from state i to state j for a cell x
  S is a stochastic disturbance term
  x is the cell in question
  y is a cell in a neighborhood
  δ is the euclidean distance between y and x
  m is a function that gives the weighting parameter given the transition type, the state of y, and δ.
  Vacant cells do not receive a weight
- the stochastic disturbance term, S, is given below:

$$S = 1 + (-ln(R))^a \tag{2}$$

  where
  R is a uniformly distributed random variable between 0 and 1, so $(0 < R < 1)$
  a is a parameter that allows for control of the size of the stochastic perturbations
  A histogram of 500 S values given an $a = 2.5$ is shown below:



- The function $m(state(y), ij, d)$ is defined in the paper by a set of discrete numbers. While unnecessary to list the entire set of parameters in the paper, an example is that a cell, y of type *commercial*, with a distance $d = 4$, will give a weighting parameter of 0.4 for the transition type $ij = VH$, or a transition type of *vacant to housing*
  - The discrete values for the weighting parameter m were not calibrated by any specific city. Instead they were based on observed behavior.
- Then, the cells with the N_i^{th} highest potential for each N_i are "transformed," and the next iteration begins. As an example, if the exogenously determined number of cells allowed to transform to a *commerce* state is 5, then the cells in the grid with the 5 highest *commerce* potentials, i.e. $P(iC)$ where $i = V, H, I$, are transformed to a commerce cell for the next iteration.

4 Changes to the White and Engelen model

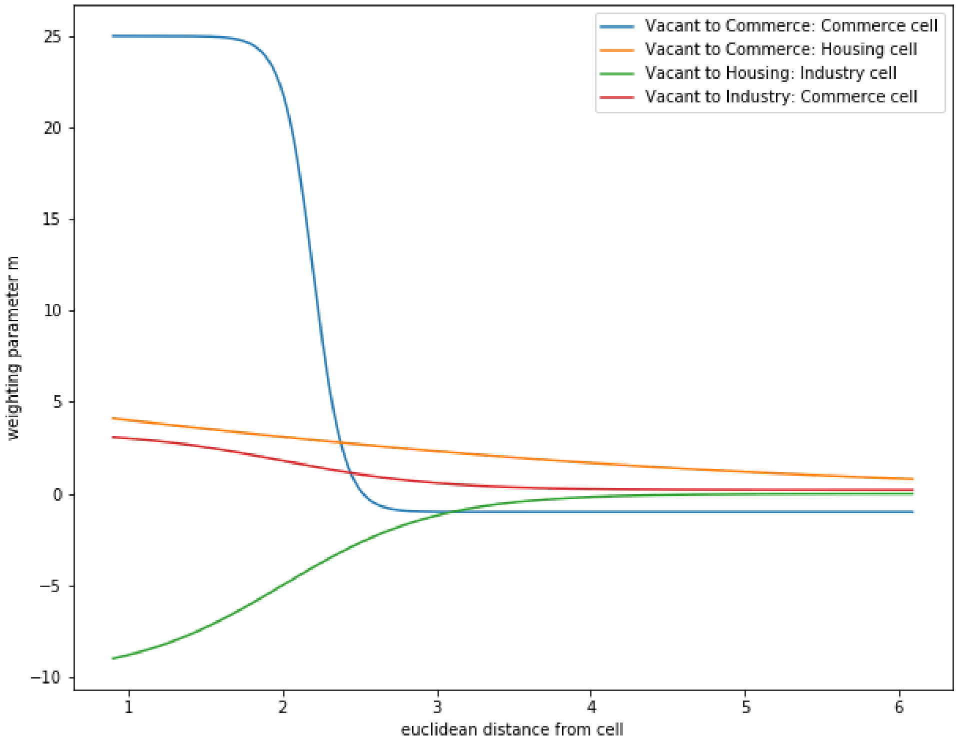
While White and Engelen's model gave a good basis for a land-use model, I had to a adapt a few parts of the model. Here are the following changes I made and the reasons why:

1. Rather than the m function being a discrete function, I instead chose to model it using continuous functions, based off of the discrete parameters from the White and Engel paper. There were two types of functions I used to model each weighting parameter as a function of distance and state:

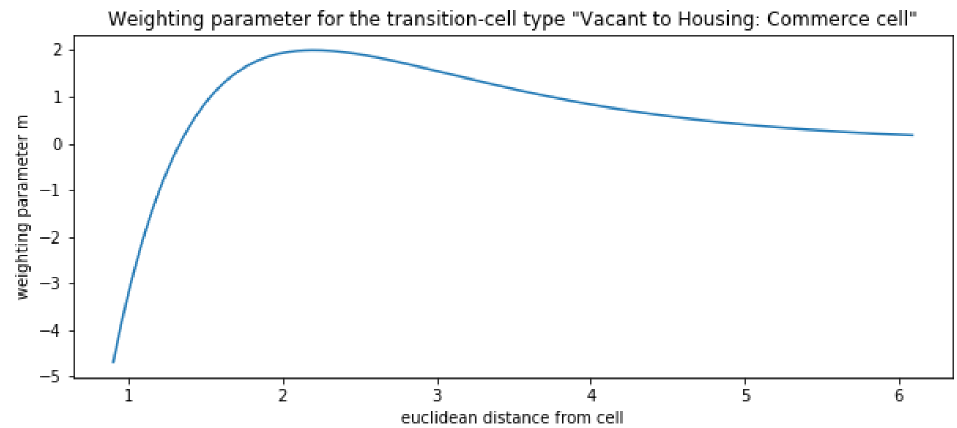
   (a) For most cell and transition types, I used a logistic model. The equation for a typical logistic graph looks like this:

$$f(\delta) = a * \frac{e^{\frac{-\delta-c}{b}}}{1+e^{\frac{-\delta-c}{b}}} + d \tag{3}$$

   where
   a, b, c, d are parameters for the equation and δ is the euclidean distance
   Shown below are a few examples of transition types with a logistic growth (or decay) graph.



(b) However, one specific cell-and-transition type required a different model. In a *vacant to housing* transition, the effect of *commerce* cells is not either strictly increasing or strictly decreasing, which is a limitation of the logistic model. The graph instead looks like below:



This is a an example of an **interatomic potential** function, which maps the potential energy between two atoms based on their distance.
The equation:

$$f(\delta) = -a(e^{-2*b*(\delta-c)} - 2e^{-b*(\delta-c)}) \tag{4}$$

where a, b, c are parameters and δ is the euclidean distance

2. Another change I made was a change to the calculation of $P_x(ij)$. Because of the $1+$ at the beginning of the definition of P, many *vacant* cells around the edges still were transitioning early in the model, since the

stochastic perturbations were more of a factor. This was happening too frequently, so instead of the original equation, I changed it to

$$P_x(ij) = S(.25 + \sum_{y \in N} m(state(y), ij, \delta) \tag{5}$$

By changing the first term from 1 to .25, the model lowers the amount of random growth on the edges of the model, while still promoting some sort of randomness.

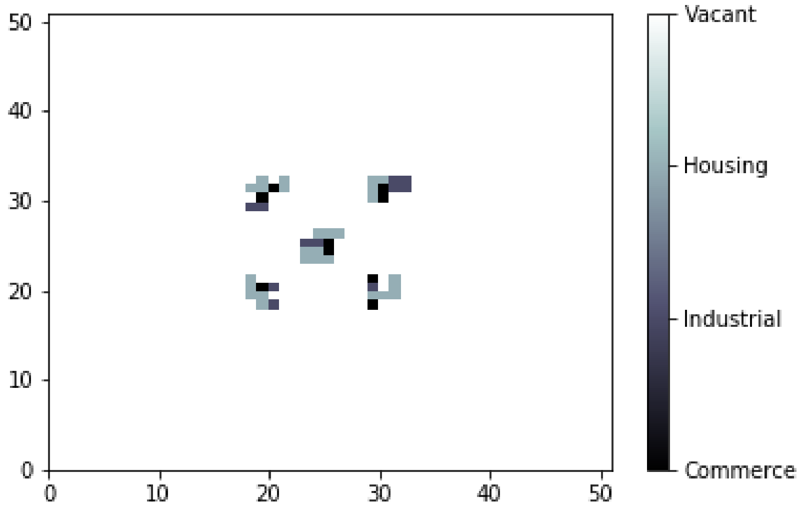
5 Two Different types of cities

I created the above White and Engelen model, with my own alterations, in python. Then, I tried to see how it models cities growing with different initial configurations. The two configurations that made the most sense were a city that began as a series of smaller towns that got sucked up into the city and a city with a single central area that then extends outwards. Two European cities work well as examples for the two different types: Berlin for the former type and Paris is for the latter type.

For each city type, I ran three simulations. The parameters were as follows:
initial ratio: $[1.0, 4.0, 7.0]$ for $[N_C, N_I, N_H]$
growth rate: 0.06
stochastic perturbation parameter: $a = 2.5$
and the model is run for 40 iterations.

5.1 The "Berlin" Model

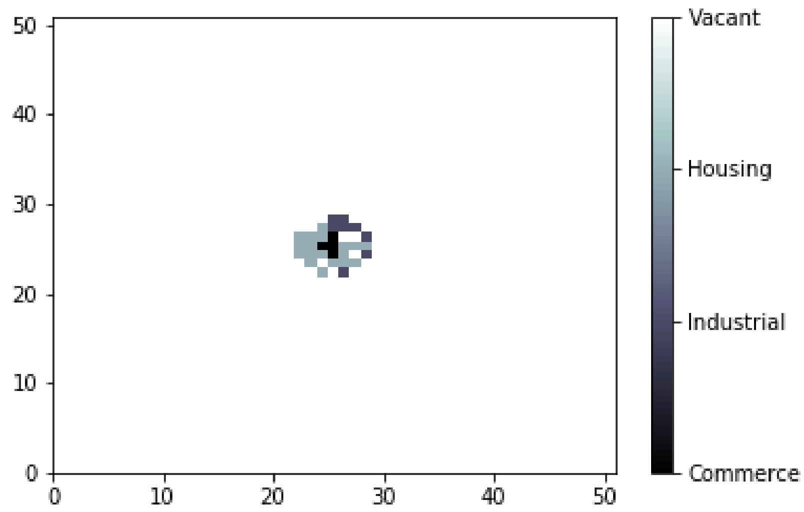
As stated before, the "Berlin" model has a series of smaller clusters around the middle of the grid at time $t = 0$. See below for a figure.



The rest of the simulations can be seen in the **Simulations** section at the end of the paper.

5.2 The "Paris" Model

The "Paris" model has a central district with nothing in the surrounding area. This simulates a traditional European walled city that needs to expand past its walls. The figure below is the initial configuration.



5.3 Analysis

A few key similarities between the two models stand out.

- Firstly, *industry* zones are more homogeneous than the other two cells. Also, *industry* zones seem to confine the growth of housing, for the most part. The only time we see a *housing* cluster form beyond an industrial one is in run 1 of the Berlin simulation.
- Additionally, clusters of *commerce* cells tend to pop up in the middle of *housing* sites.
- There also tends to be more random growth in the beginning of the model. Occasionally, like in run 2 of Berlin and in runs 2 and 3 of the Paris, an industrial zone will pop up far away from the city center.

The differences between the two models are more striking.

- The growth of Paris's housing and industrial zones tend to stick to a left-right axis seperating the two, while the Berlin simulations tend to have several clusters of each type.
- Additionally, the Paris simulations have more cells transitioning far from the city center than in the Berlin simulations.
- However, the Paris simulations all have a similar shape: a central, elliptic city with one or two clusters on the outsides. The Berlin simulations tend to have a less elliptic shape, and follow more of an *urban sprawl* pattern.

Clearly, the initial conditions play a huge role in the shaping of a city growth.

6 Investigations

6.1 Clustering coefficient

A clustering coefficient in a Cellular Automata model describes how closely cells of the same state tend to be close to each other. I interpreted this as how many cells in the previously described neighborhood are of the same type. Therefore, the clustering coefficient for cell x C_x is given by:

$$C_x = \frac{\#\, of\ cells\ with\ x's\ type\ in\ N}{total\ \#\, of\ cells\ in\ N} \tag{6}$$

Then, the average clustering coefficient, $C_i, (i = H, I, C)$, is given by:

$$C_i = \frac{\sum_{x \in grid}(c_x \; of \; type \; i}{total \; \# \; of \; cells \; of \; type \; i \; in \; grid} \tag{7}$$

Figure 9 in the **Simulations** section charts the clustering coefficient fir Berlin's 1st run with respect to time. Originally, the *housing* coefficient keeps above the *industry* coefficient. This makes sense since the housing blocks grow at a faster rate and also tend to stick together, so they are more likely to have a higher coefficient at the beginning. However, as the model moves forward in time, the *industry* coefficient takes over its counterpart. Since *industrial* cells tend to repel other types of growth, this makes sense because the *industrial* clusters tend to become more homogeneous as the model moves along.
Commerce cells tend to keep a lower clustering coefficient, since it has a smaller growth rate than the other two and likes to pop up in the middle of *housing* blocks.
Additionally, the coefficients are increasing for almost the whole time, which makes sense given that the model is growing and it tends to put similar cells together.

Figure 10 in the **Simulations** section charts the clustering coefficient for Paris's first run. It shows a similar phenomenon to the Berlin one, although the *housing* and *industry* coefficients stay at around the same rate for the middle of the model.

6.2 Standard deviation of transition potentials

Another interesting investigation is a look into how the transition potentials, $P_x(ij)$ are distributed as the simulation iterates. Figures 9 and 10 show the standard deviation for each cell type of the transition potentials to that type. A generic standard deviation formula is given by:

$$s = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2} \tag{8}$$

where
s is the standard deviation
N is the number of values
x_i is each individual values
$\bar{x}$ is the mean of all values

Looking at the graphs, we see that the standard deviation, while having many large falls and jumps, increases from start to finish. The large jumps can be attributed to random stochastic perturbations, which we know from the S factor for the potential equation can be high. Since the model only transitions the highest potentials for each transition type, a higher standard deviation means the values are more spread out. Since the values are more spread out, the perturbations have less of an effect, meaning a lower stochastic growth for the city. This is why we see more random growth for smaller time periods.

Looking at the differences between the two city types, Berlin's standard deviations grow at a slightly more uniform rate at the beginning, but then grow to become more and more random as time goes on. Paris's standard deviations do not have the same volatility. This can be seen in the model, as Paris's urban form has a more uniform growth and Berlin's does not.

7 Drawbacks of the model

There are a few drawbacks of this Cellular Automata model.

First of all, it ignores geological features that would affect city growth. For example, most major cities are built either near a river, a lake, or an ocean. This changes how cities grow, since they obviously cannot build on such water features. In more extreme circumstance, cities are built on peninsulas, like San Francisco, or on islands, like Manhattan. This obviously changes the growth patterns of an urban area and the model does not take that into account.

Also, urban green spaces like parks are not considered. A park in a city often brings *housing* clusters and pushes away *industrial* ones.

Another drawback of the model is that it does not consider technological advancement that leads to a changing makeup of cities. As seen in both city models above, the *industrial* areas often are situated in a large, dense area close to the city center. By looking at modern cities like Manhattan. we know that as time progresses and cities become more technologically advanced, the *industrial* sites are pushed to the periphery while the *commerce* districts take over the center.

A third drawback is the assumption that cells can only transform based on a hierarchy, although it is possible for a *commerce* area to transform into a *housing* cell, and it is even possible for cells to be turned into *vacant* spaces, although less likely. White and Engels in their paper mention that later versions of the model take this into account, although the details are not specified.

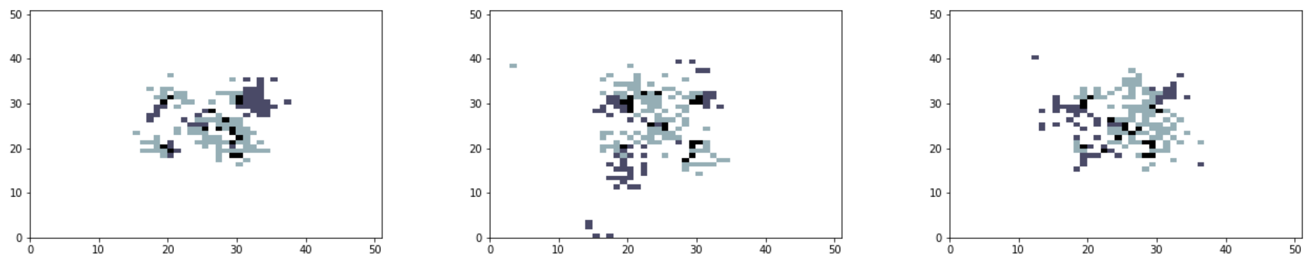
A fourth drawback, or rater a limitation, for me was the inability to use real-world data. It would involve image processing which I am not capable of. This would allow me to use real-world data rather than relying on made-up city scenarios.

8 References

Van White, Nick. 'The great sprawl of China: timelapse images reveal 30-year growth of cities'. *The Guardian*. 21 March 2017.

White, R and Engelen, G. 'Cellular automata and fractal urban form: a cellular modelling approach to the evolution of urban land-use patterns'. *Environment and Planning A, 1993*, volume 25, pages 1175-1199. 11 Feburary, 1993.

9 Simulations

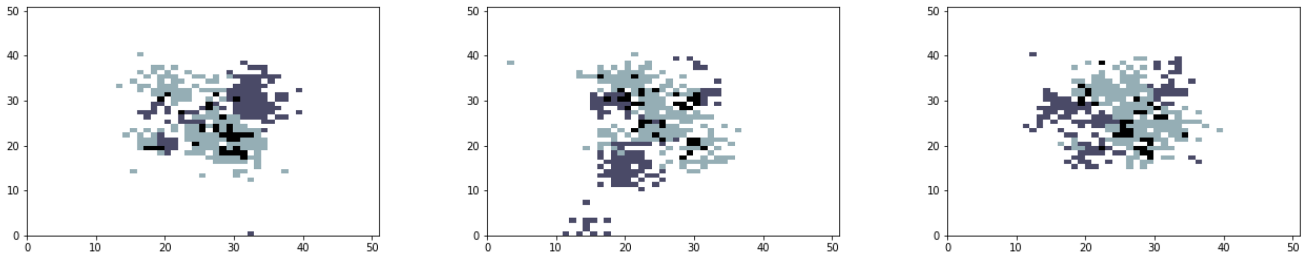


(a) run 1

(b) run 2

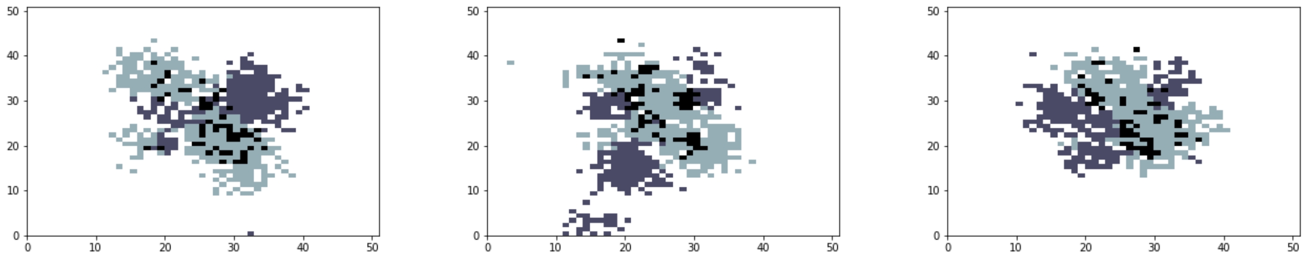
(c) run 3

Figure 1: Berlin time t = 10



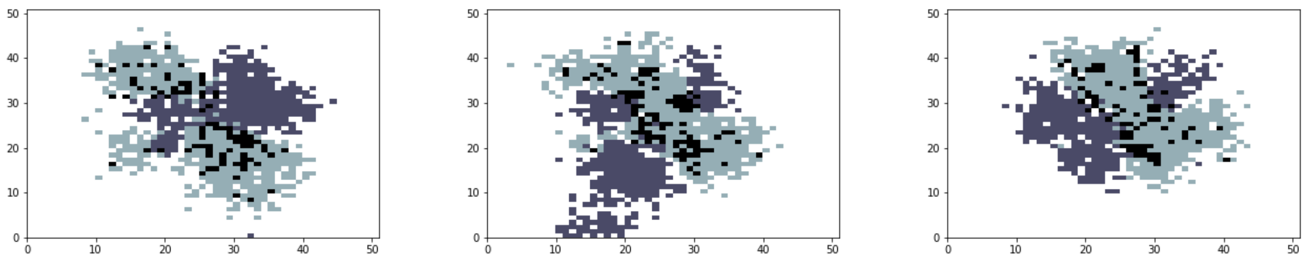
(a) run 1 (b) run 2 (c) run 3

Figure 2: Berlin time t = 20



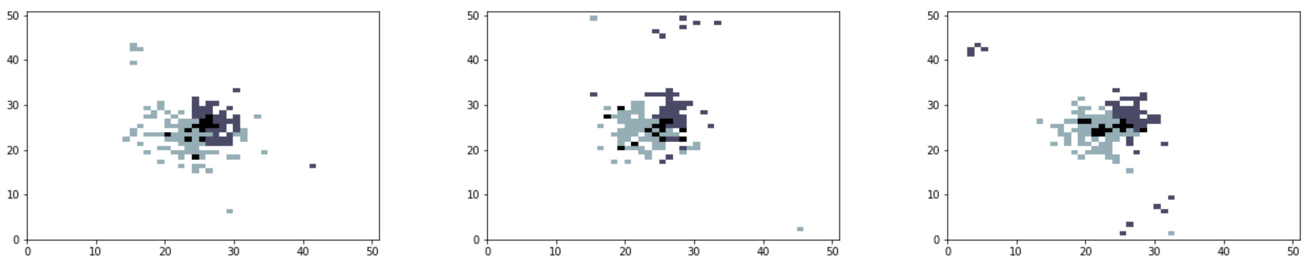
(a) run 1 (b) run 2 (c) run 3

Figure 3: Berlin time t = 30



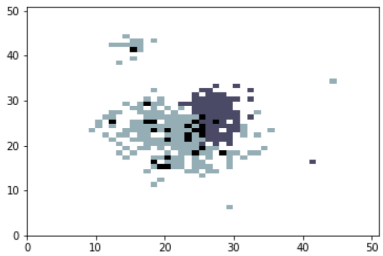
(a) run 1 (b) run 2 (c) run 3

Figure 4: Berlin time t = 40

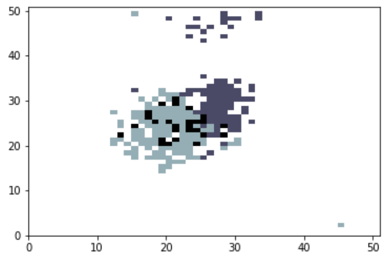


(a) run 1 (b) run 2 (c) run 3

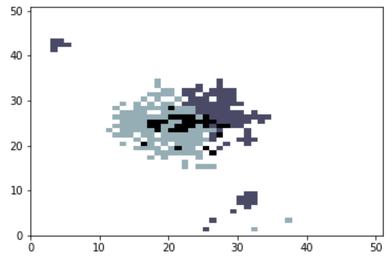
Figure 5: Paris time t = 10



(a) run 1

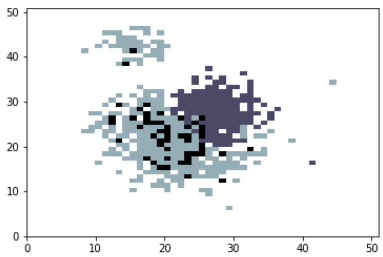


(b) run 2

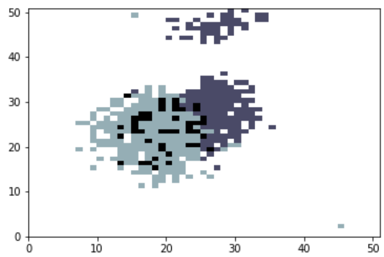


(c) run 3

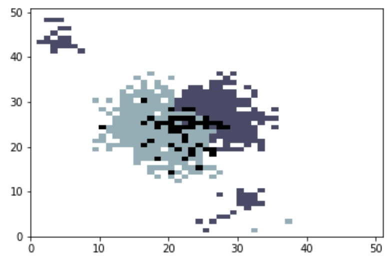
Figure 6: Paris time t = 20



(a) run 1

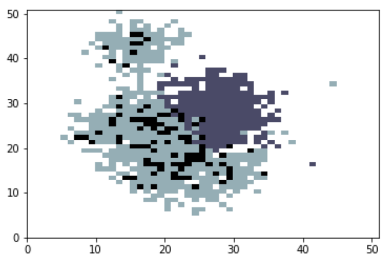


(b) run 2

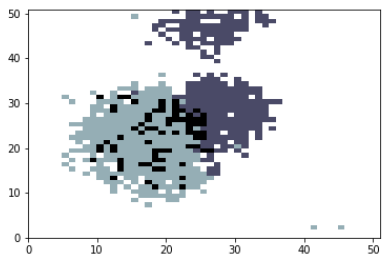


(c) run 3

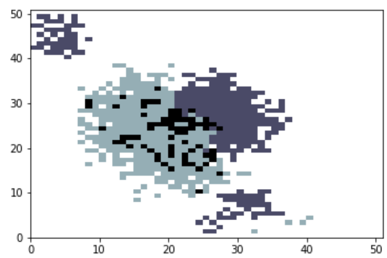
Figure 7: Paris time t = 30



(a) run 1



(b) run 2



(c) run 3

Figure 8: Paris time t = 40

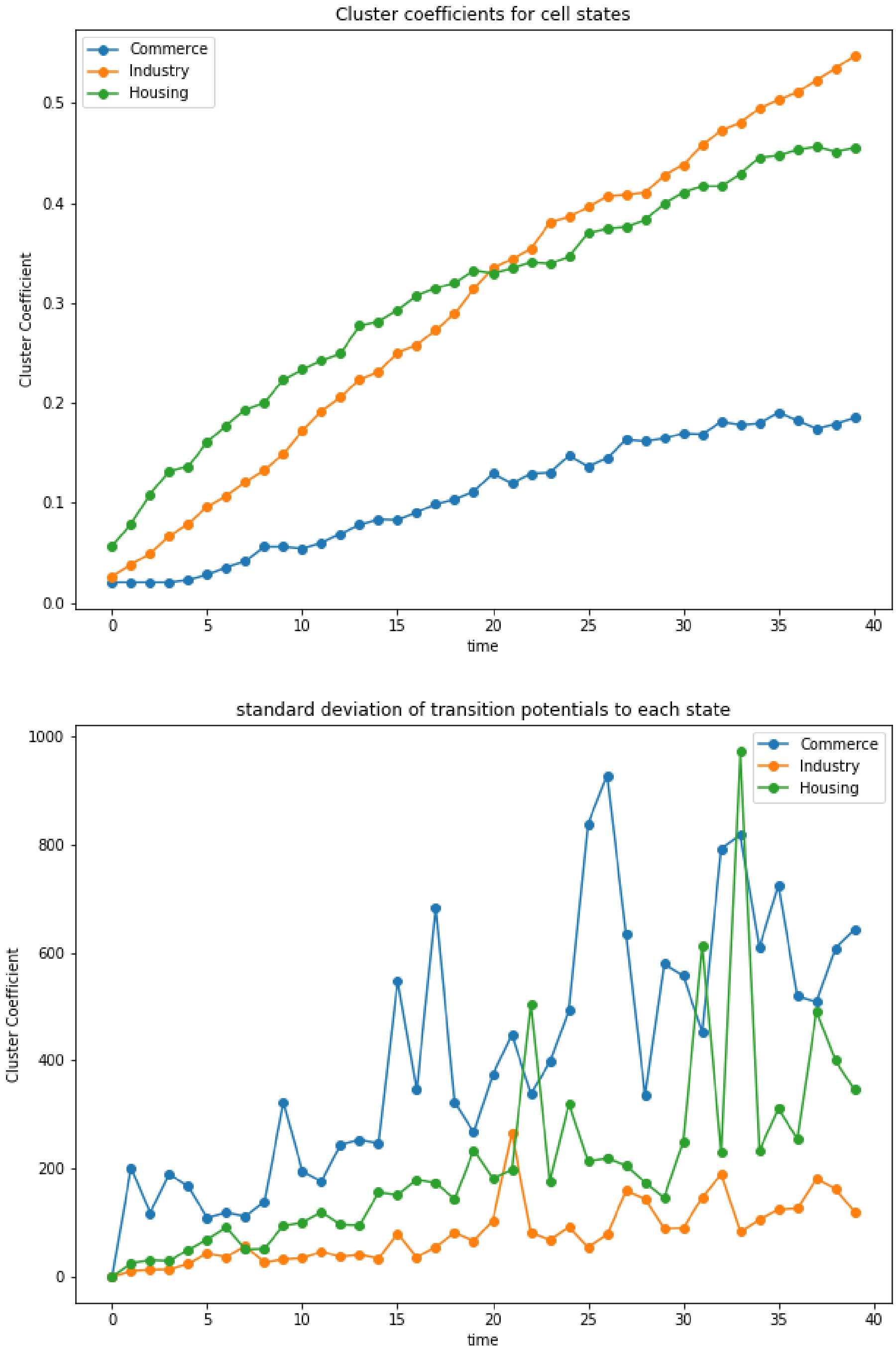


Figure 9: Berlin's numerical investigations

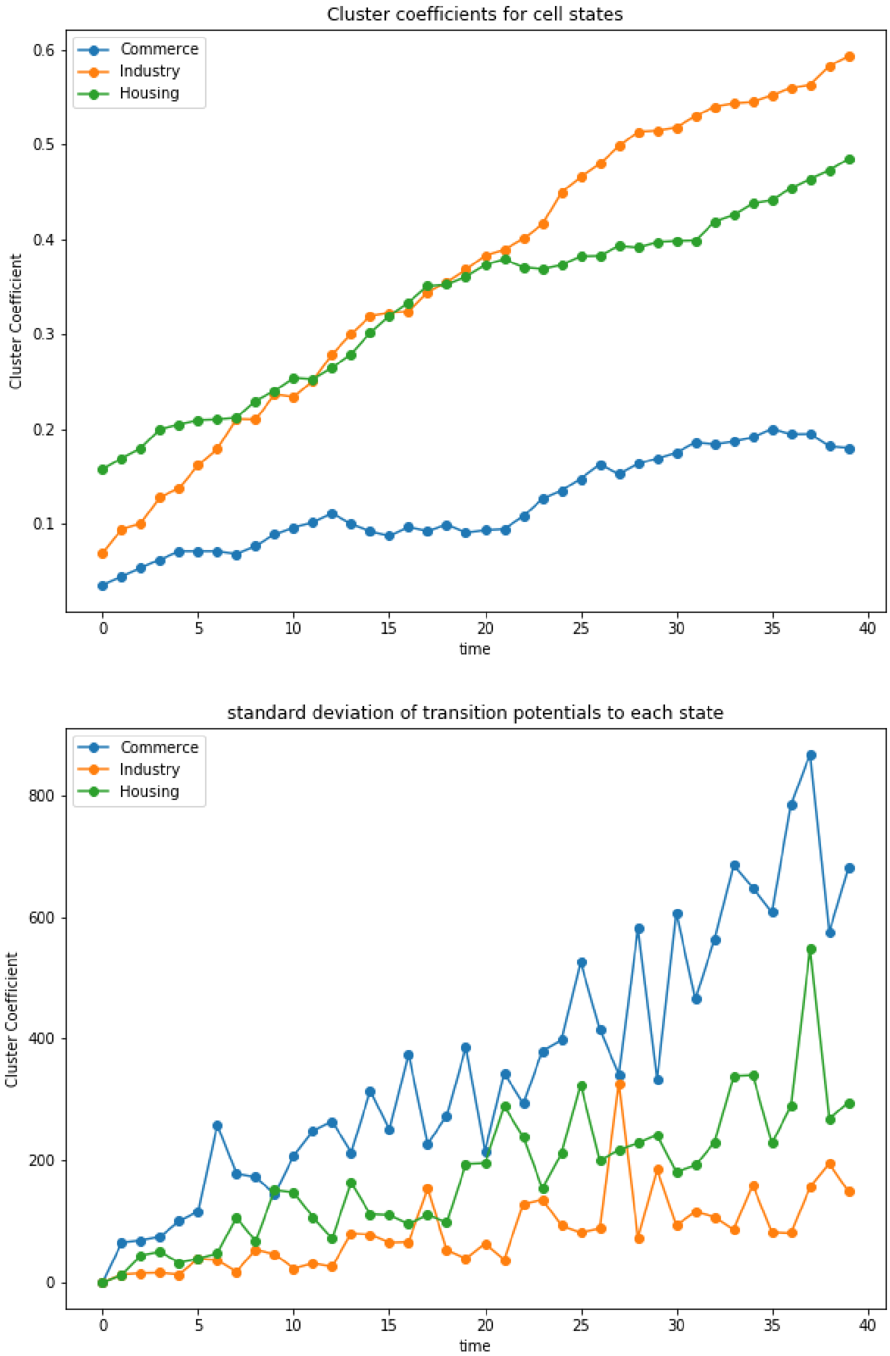


Figure 10: Paris's numerical investigations