Programming remark: how to handle periodic boundaries

- The easiest way to handle periodic boundary conditions in C++ is to use *modulo* operator, denoted by in C++ by %. It has the property that for integers a and b,

$$a\%b = \text{remainder in the division } \frac{a}{b}$$

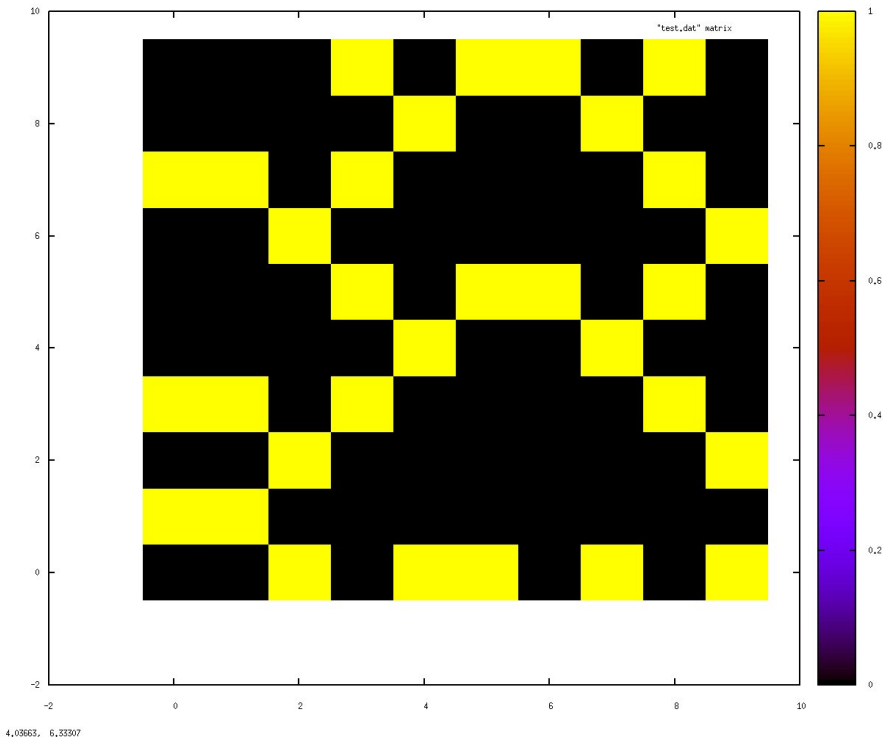
- This means that if you want, for example, value of the right neighbour of the site i using periodic boundary conditions, you can write `world[(i+1)%M]`, and it will be correct for all $i = 0, \ldots M-1$.
- If you want left neighbour, write `world[(i-1+M)%M]`. Adding M is necessary because we want to make the index positive before applying modulo operator.
- This generates some computational overhead compared to "if" statement, but is more convenient.

Programming remark: visualization with gnuplot

In gnuplot, if you have datafile "test.dat" like this, with rows separated by spaces,

0 0 1 0 1 1 0 1 0 1
1 1 0 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0 0 1
1 1 0 1 0 0 0 0 1 0
0 0 0 0 1 0 0 1 0 0
0 0 0 1 0 1 1 0 1 0
0 0 1 0 0 0 0 0 0 1
1 1 0 1 0 0 0 0 1 0
0 0 0 0 1 0 0 1 0 0
0 0 0 1 0 1 1 0 1 0

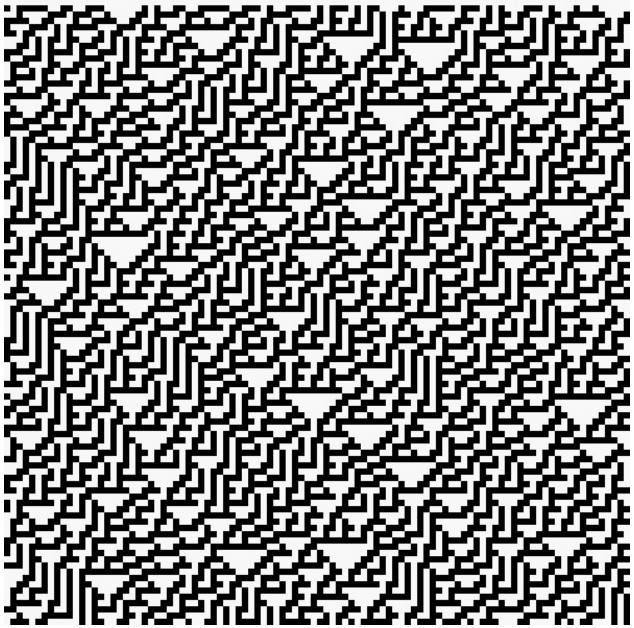
then in gnuplot, if you write plot “test.dat” matrix with image
you will obtain the following figure:



In spatiotemporal diagrams of CA, it is customary to have time axis (vertical) increasing from top to bottom. You can do it by typing in gnuplot: set yrange [] reverse
See file rule18-defects.cpp for an example of this applied to rule 18 (read comments in the file first).

Application remark: rule 30

Elementary cellular automaton rule 30, to be further studied in the assignment, generates patterns with triangles, like shown on the left.

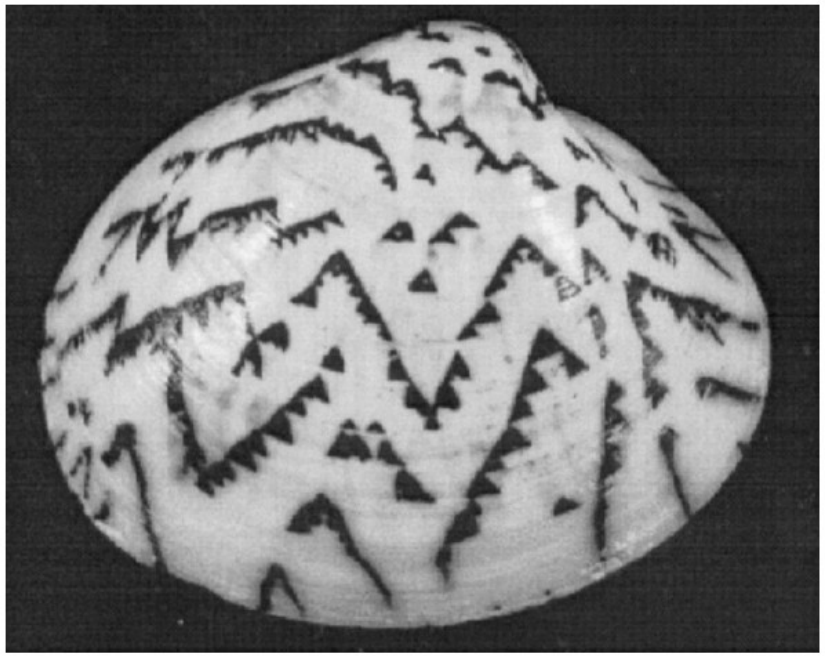


It is somewhat resembles patterns present on the shell of cone snail species *Conus textile* (right above).

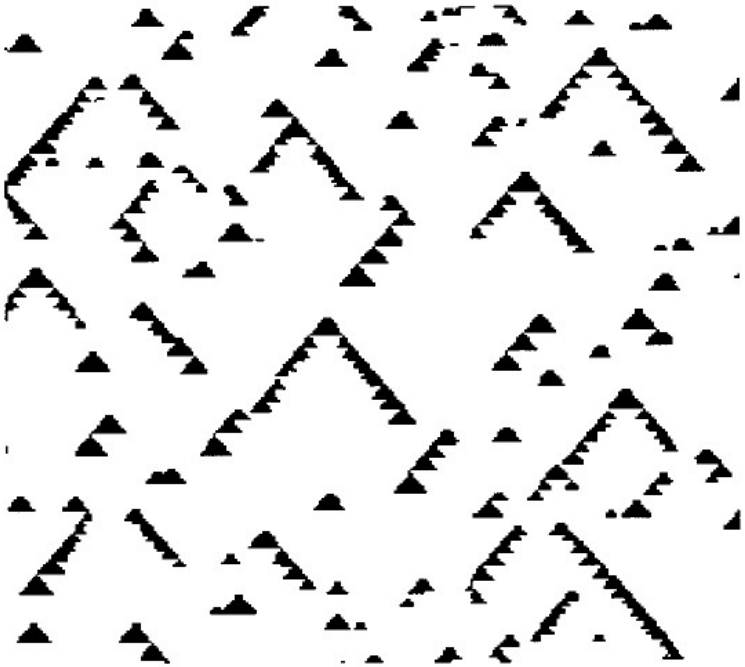
Further reading on shell patterns

"Mollusc Shell Pigmentation: Cellular Automaton Simulations and Evidence for Undecidability" I. Kusch and M. Markus, *J. of Theoretical Biology* vol. 178, Issue 3, 1996, p. 333-340.

(a)



(b)



Left: example of *Lioconcha castrensis* (mollusk from Philippines).
Right: pattern produced by CA constructed by authors.

Application remark: Nagel-Schreckenberg model

(Nagel, K.; Schreckenberg, M. (1992) *Journal de Physique I*. 2 (12): 2221)

Cars occupy sites of a lattice with periodic boundary conditions. They move according to four subrules:

1. Acceleration: All cars not at the maximum velocity have their velocity increased by one unit.
2. Slowing down: All cars are checked to see if the distance between it and the car in front (in units of cells) is smaller than its current velocity (which has units of cells per time step). If the distance is smaller than the velocity, the velocity is reduced to the number of empty cells in front of the car to avoid a collision. For example, if the velocity of a car is now 5, but there are only 3 free cells in front of it, with the fourth cell occupied by another car, the car velocity is reduced to 3.
3. Randomization: The speed of all cars that have a velocity of at least 1, is now reduced by one unit with a probability of p.
4. Car motion: all cars are moved forward the number of cells equal to their velocity. For example, if the velocity is 3, the car is moved forward 3 cells.

Nagel-Schreckenberg model - more formal definition

Each cell is either empty or occupied by a car with velocity $v_i(t)$ where $v_i(t) \in \{0, 1, \ldots, v_{max}\}$. If $x_i(t)$ is position of car i at time t, then the gap between car i and the next car, $i+1$, is defined as

$$d_i(t) = x_{i+1}(t) - x_i(t)$$

Car velocities are updated as follows:

$$v_i(t + 1/2) = \min\{v_i(t) + 1, d_i(t) - 1, v_{max}\}$$

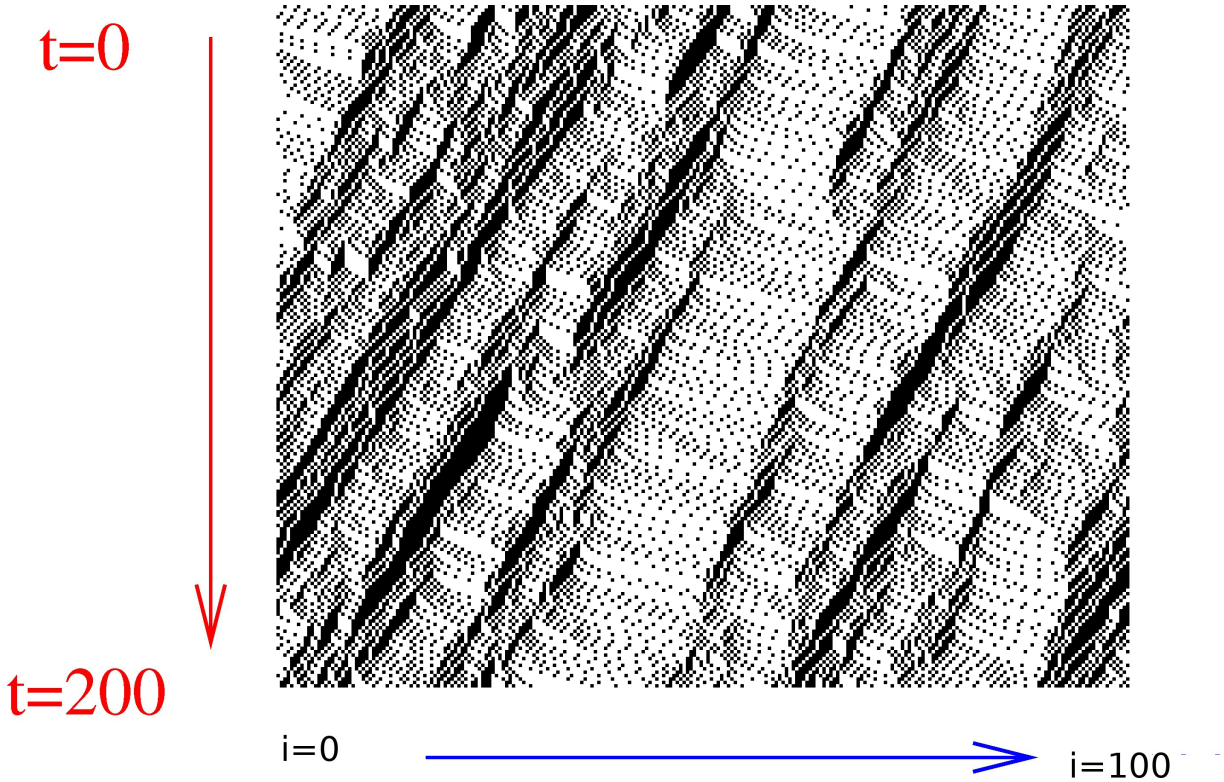
$$v_i(t+1) = \begin{cases} \max\{v_i(t + 1/2) - 1, 0\} \text{ with probability } p, \\ v_i(t + 1/2) \text{ with probability } 1 - p \end{cases}$$

Cars move according to the rule

$$x_i(t + 1) = x_i(t) + v_i(t + 1).$$

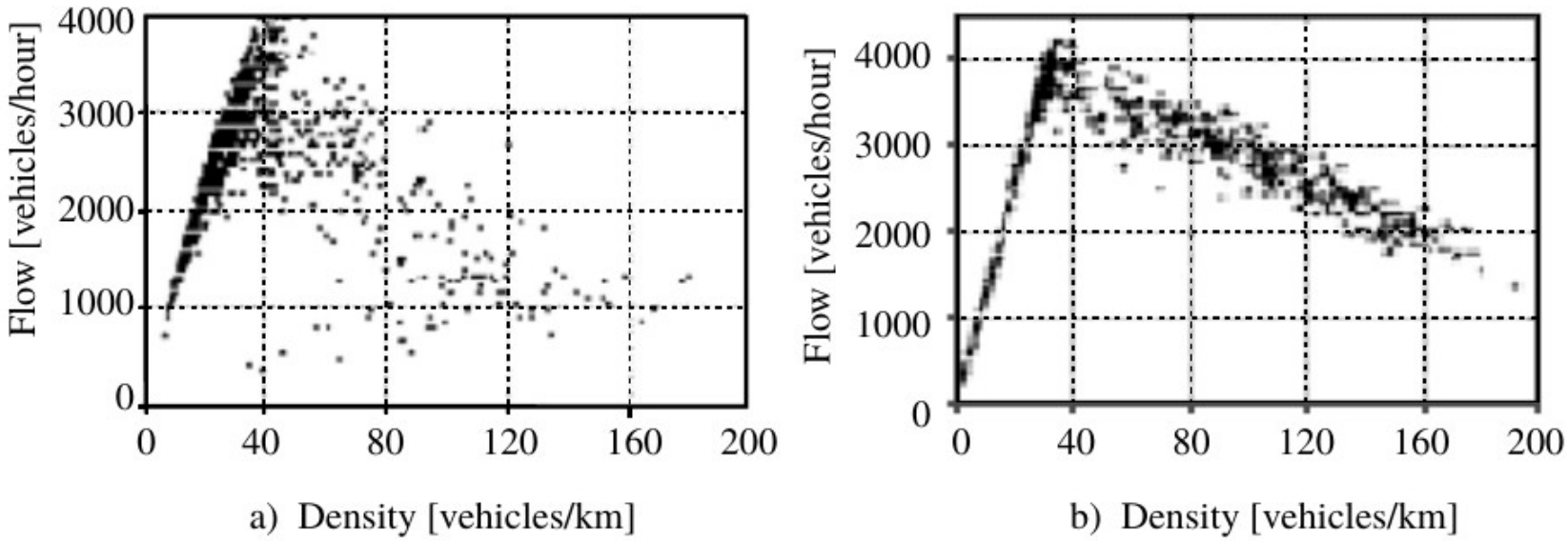
Nagel-Schreckenberg model - simulation

Spatiotemporal pattern produced by Nagel-Schreckenberg model for 100 cells, for car density 0.35 and $p = 0.3$.



Nagel-Schreckenberg model - simulation compared to real world data

Nagel-Schreckenberg model, somewhat modified to include realistic features (e.g., multiple lanes) can produce fundamental diagram very close to real traffic.

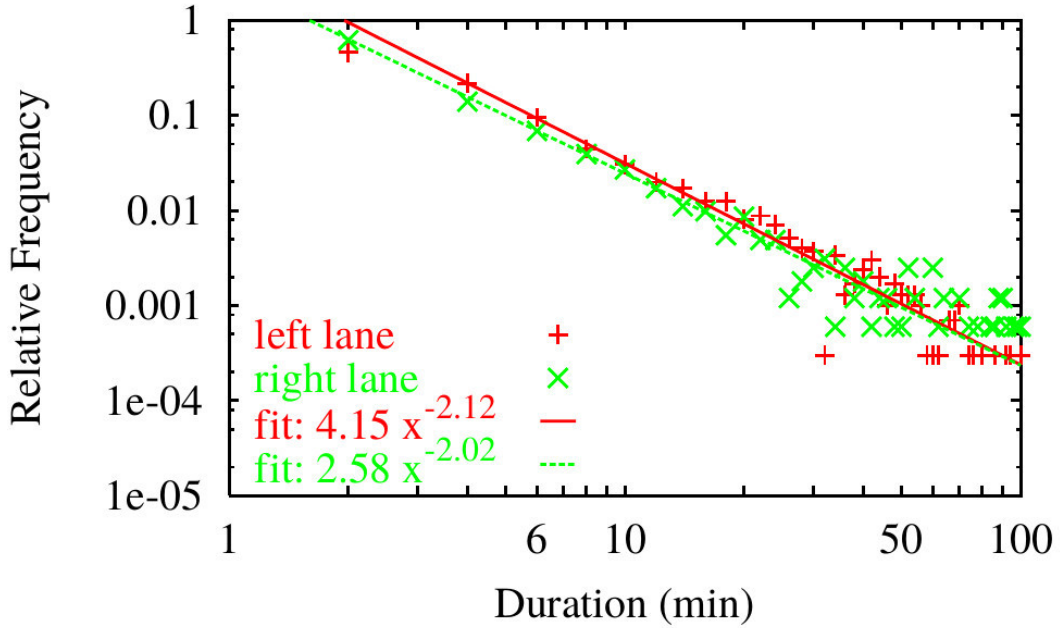


(a) real field data from German highway (b) simulations.

source: SMARTEST Partnership. Final Report SMARTEST Contract No: RO-97-SC.1059, University of Leeds, Great Britain, 2000.

Power laws in traffic

Power laws can be found in traffic data and traffic models. Example below shows the distribution of duration of traffic jams.

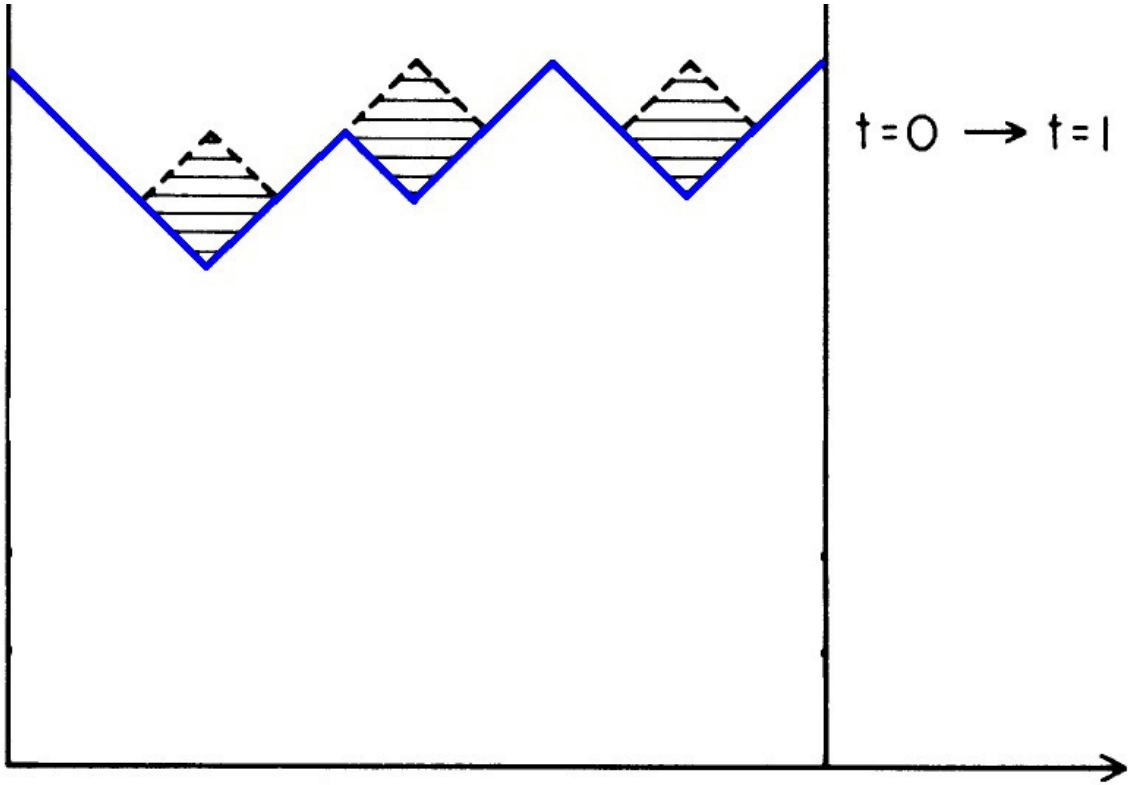


Typical scaling law for the distribution of durations, for which the vehicle flow $Q(x,t)$ at the cross section x of a freeway exceeds the threshold $Q_{\text{thres}} = 1400$ vehicles/h. The distribution follows a power law and has been determined from two-minute averages of single vehicle data of the Dutch freeway A9 from Rottepolderplein to Badheuvedorp close to Amsterdam.

from D. Helbing and B. Tilch, *The European Physical Journal B* 2009, Vol. 68, Issue 4, pp 577–586

Application remark: surface deposition model

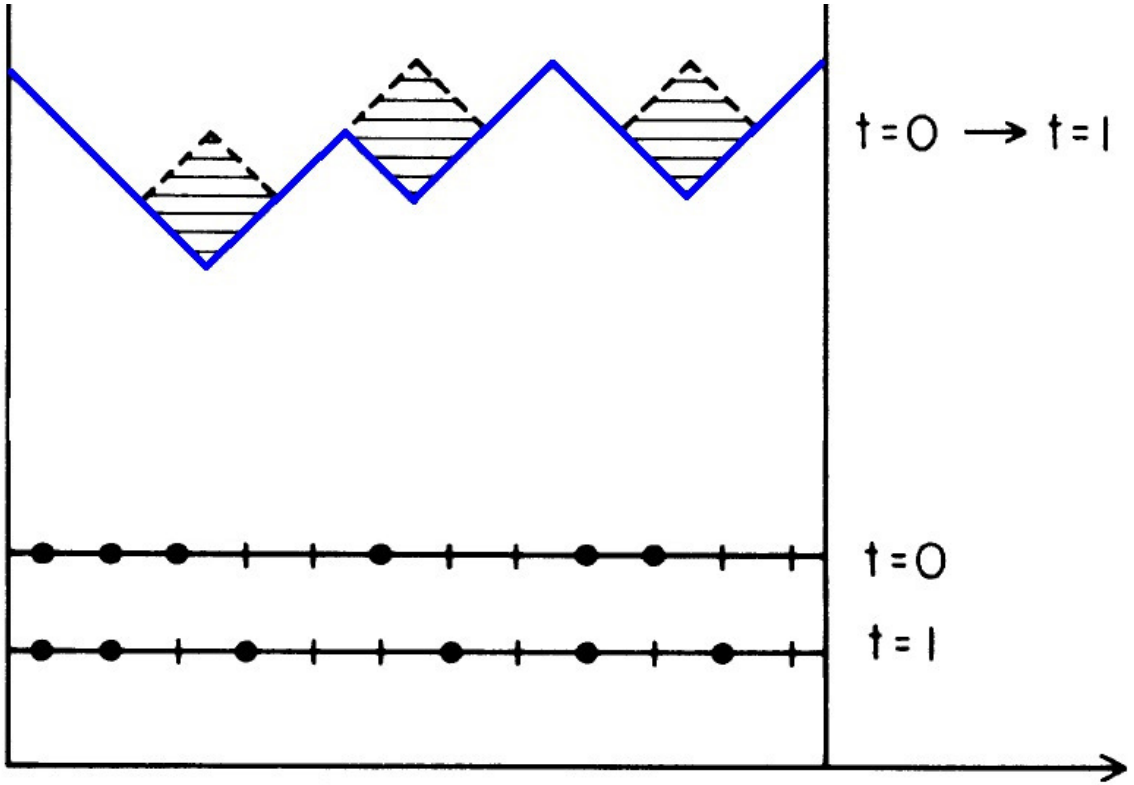
Consider the following model of surface deposition: at each time step, all local surface minima are filled with new particles.



(J. Krug and H. Spohn, *Phys. Rev. A*, 38:4271–4283, 1988.)

Surface deposition model - particle interpretation

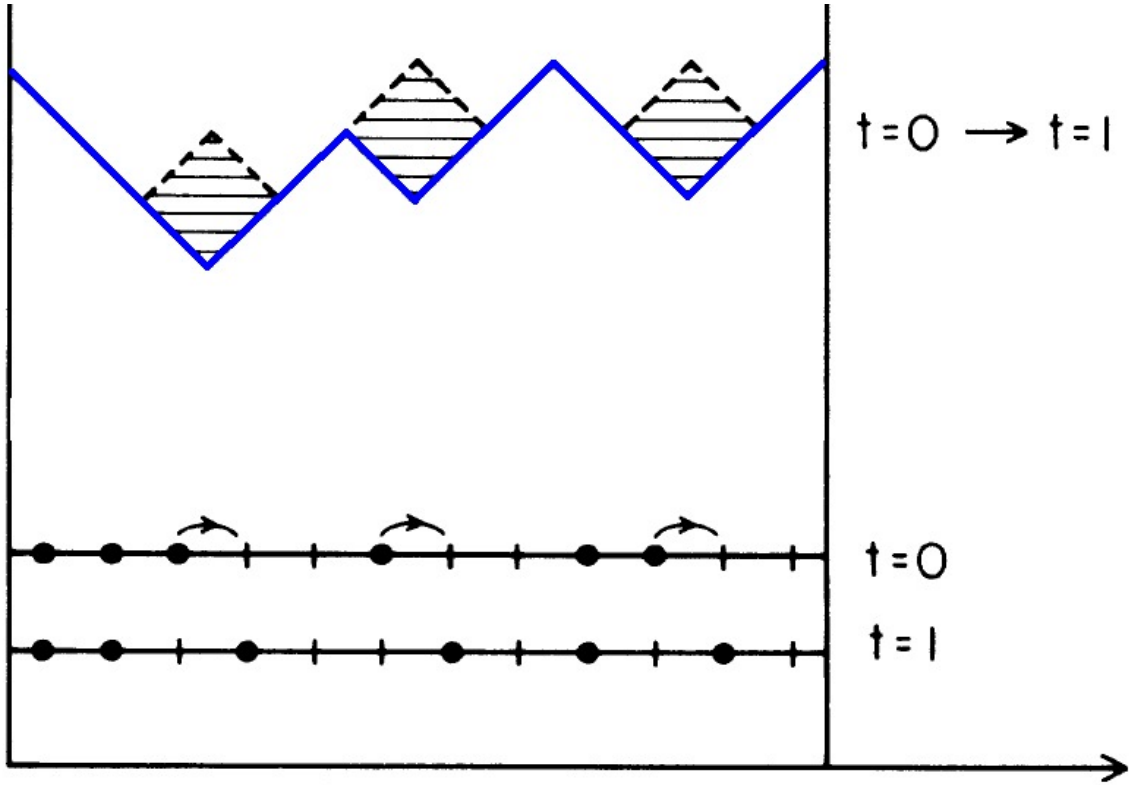
Suppose we treat segments with negative slope as site occupied by imaginary particles, and segments with positive slope as empty.



(J. Krug and H. Spohn, *Phys. Rev. A*, 38:4271–4283, 1988.)

Surface deposition model - particle movement

Particles appear to move to the right if the right neighbour is empty!



(J. Krug and H. Spohn, *Phys. Rev. A*, 38:4271–4283, 1988.)

We know from previous lectures that this is just rule 184.

$$s_{t+1}(i) = s_t(i-1) + s_t(i)s_t(i+1) - s_t(i-1)s_t(i).$$

In the above:

- $s_t(i) = 1$, site i occupied by a "particle", slope of the surface $= -1$
- $s_t(i) = 0$, site i empty, slope of the surface $= 1$

Let $x_t(i)$ be the slope of the i-th segment of the surface at time t. Then it is easy to see that we have

$$x_t(i) = 1 - 2s_t(i)$$

$$s_t(i) = \frac{1}{2}(1 - x_t(i))$$

Changing variables to x we obtain

$$x_{t+1}(i) = \frac{1}{2}x_t(i-1) + \frac{1}{2}x_t(i+1) - \frac{1}{2}x_t(i)\,x_t(i+1) + \frac{1}{2}x_t(i-1)\,x_t(i)$$

Surface deposition model - surface width

We thus obtained a cellular automaton model where $x_t(i) \in \{-1, 1\}$ such that

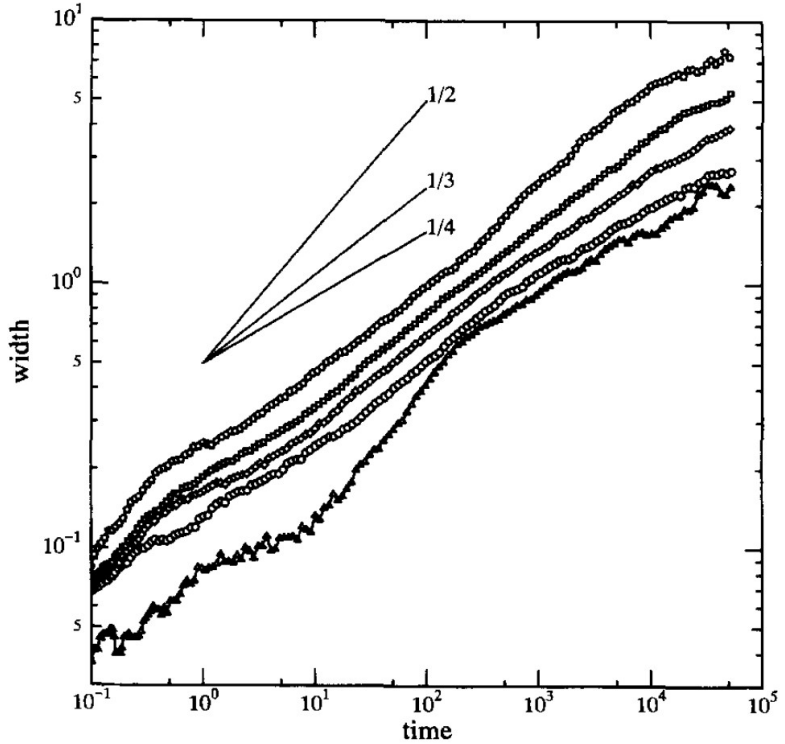
$$x_{t+1}(i) = \frac{1}{2}x_t(i-1) + \frac{1}{2}x_t(i+1) - \frac{1}{2}x_t(i)\,x_t(i+1) + \frac{1}{2}x_t(i-1)\,x_t(i)$$

It is obviously equivalent to rule 184.
Now, if we start with the surface which is "balanced", that is, having the same number of segments with positive and negative slope, it will have its highest point of height h_{max} and lowest point of height h_{min}. Define the width of the surface at time t to be $w_t = h_{max} - h_{min}$.
We expect

$$w_t \to 1 \text{ as } t \to \infty$$

How does it converge to 1? Perhaps as a power law? A question to consider, and perhaps to investigate as a part of the final project.

In rule 184, surface width decreases with time. However, in real world deposition processes, an opposite phenomenon often takes place, namely *surface roughening*. In the process of the crystal growth from vapour, the width of the surface actually grows, and the width $w \approx t^{\beta}$ with β varying from $1/8$ to $1/2$ depending on the model (the width is usually defined somewhat differently than on the previous slide). We will talk about such models later.



K. Nishino et al., *Surface Science* Volume 374, Pages 291297 (1997)