Simulations in Statistical Physics

Course for MSc physics students

Janos Török

Department of Theoretical Physics

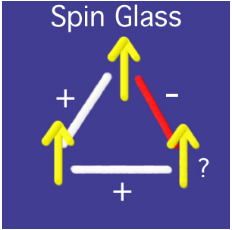
October 20, 2015

Glassy behavior, frustration

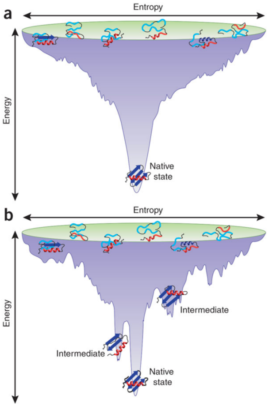
- Model glass: spin-glass:

$$H = -\frac{1}{2}\sum_{\langle i,j\rangle} J_{ij} S_i S_j$$

- where J_{ij} are random quenched variables with 0 mean (e.g. $\pm J$ with probability half)



Rugged energy landscape.



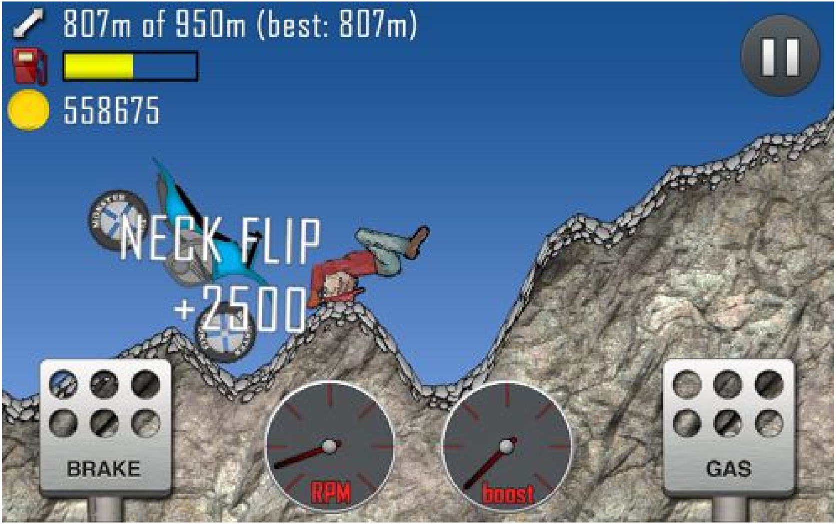
Rugged energy landscape

- Typical example NP-complete problems:
  - Traveling salesman
  - Graph partitioning
  - Spin-glass
- No full optimization is possible (do we need it?)
- Very good minimas can be obtained by optimization
  - Simulated annealing
  - Genetic algorithm

Simulated annealing

- Cool down the system slowly
- Speed is crucial and many experiments are needed
- No guarantee that we find something meaningful
- Warm up and down if needed, if the system quenched into a local minimum
- One needs a Hamiltonian (or a fitness function) and an elementary move
  - Spin glass: Metropolis
- Traveling salesman
  - Minimal travelling path for visiting a number of cities
  - Elementary move: swap two cities ($T \sim$ alcohol)

Hill climb


807m of 950m (best: 807m)
558675
NECK FLIP
+2500
BRAKE
RPM
boost
GAS

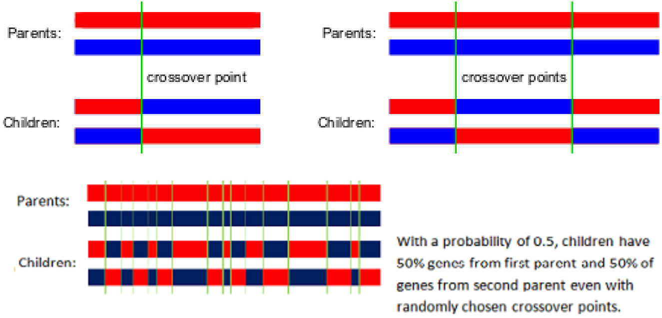
Travelling salesman

Genetic algorithm

- Learn from nature
- Let the fittest to survive
  - Fitness function, e.g. energy, length, etc.
- Combine different strategies
- State is represented by a vector (genetic code or genotype)
  - Phasespace, city order, neural network parameters, etc.
- Offsprings have two parents with shared genetic code
- Mutations
- Those who are not fit enough die out
  - Keep the number of agents fixed

Genetic algorithm: Reproduction

- Two parents and two children



Genetic algorithm terminology

- Chromosome: Carrier of the genetic representation
- Gene: Smallest units in the chromosome with individual meaning
- Parents: Pair of chromosomes, wich produce offsprings
- Population: Set of chromosomes from which the parents are selected. Its size should be larger than the length of the chromosome
- Selection principle: The way parents are selected (random, elitistic)
- Crossover: Recombination of the genes of the parents by mixing
- Crossover rate: The rate by which crossover takes place ($\sim$90%)
- Mutatation: Random change of genes
- Mutation rate: The rate by which mutation takes place ($\sim$1%)
- Generation: The pool after one sweep.

Genetic algorithm schema

1. Start with a randomly generated population
2. Calculate the fitnesses
3. Selection
    - Random
    - Best fitness (keep top 50% and generate new 50%)
    - Roulette (Monte-Carlo) selection
4. Crossover: offsprings must be viable (Sometimes difficult)

Parents

1	2	3	4	5	6	7	8	9

9	8	7	6	5	4	3	2	1

Offspring

					6	7	8	

9	5	4	3	2	6	7	8	1

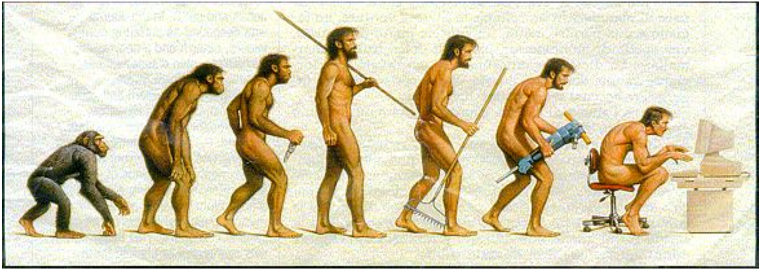
Genetic algorithm schema

1. Start with a randomly generated population
2. Calculate the fitnesses
3. Selection
   - Random
   - Best fitness (keep top 50% and generate new 50%)
   - Roulette (Monte-Carlo) selection
4. Crossover: offsprings must be viable (Sometimes difficult)
   - One-point
   - Two-point
   - Uniform
   - Mutation: small rate

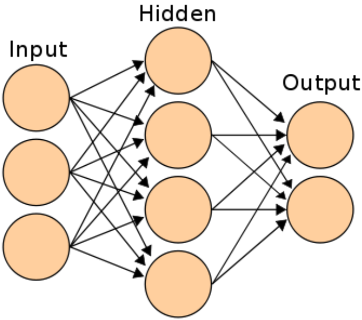
1	2	3	4	5	6	7	8	9

1	2	8	4	5	6	7	3	9

Genetic algorithm example

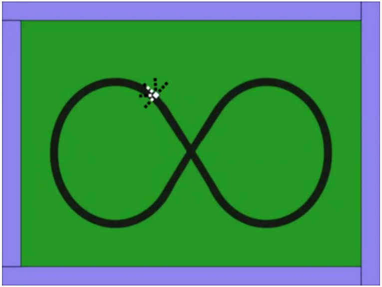


Neural networks



- Input pattern
- Output pattern
- Adaptive wights
- Approximating non-linear functions

- Machine learning
- Pattern recognition
- Handwriting
- Speech recognition



Neural networks

- Input vector I
- Output vector $O(I)$
- Transition matrix $W_{ij} \in [-1, 1]$
- Learning using a cost function
- Test goodness

Neural networks: Learning

- Supervised learning
- Data training:
  - Superwised learning
  - Fitness function, energy:

    $$E = T(I) - O(I),$$

    where $T(I)$ is the target vector for input I
  - Minimize E for available set of $\{I, I(O)\}$ pairs
- Test goodness:
  - Use only part of $\{I, I(O)\}$ pairs for learning, the rest is for testing.
- Used for: pattern recognition, classification, etc.

Neural networks: Learning

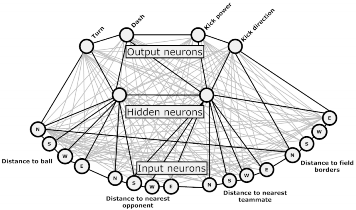
- Unsupervised learning
- Cost function may depend on task
- Cost function is deviation from mean data

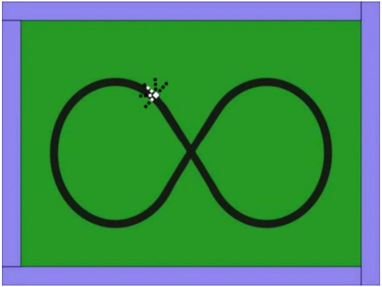
$$C = E[(x - f(x))^2]$$

- Test goodness:
  - Some self consistent limit on the cost function
- Used for: estimation, filtering, etc.

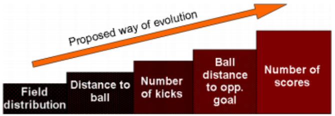
Neural networks: Learning

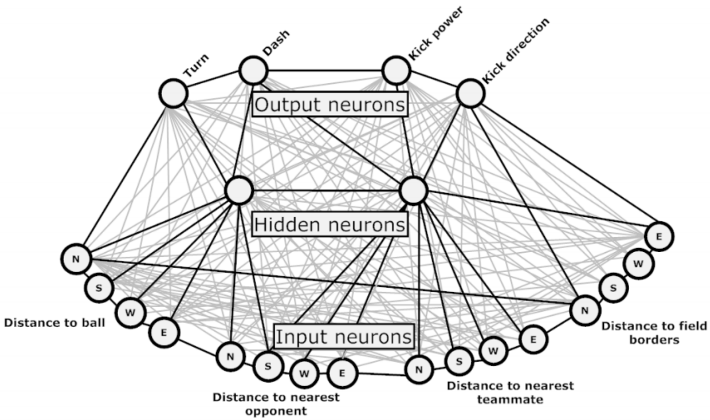
- Reinforcement learning
- Cost function is a long time performance on an agent making decisions based on the neural network.
- Test goodness:
  - Compare with other agents which can be algorithmical or based on neural networks
- Used for: control problems, AI, complex optimization





Genetic algorithm example

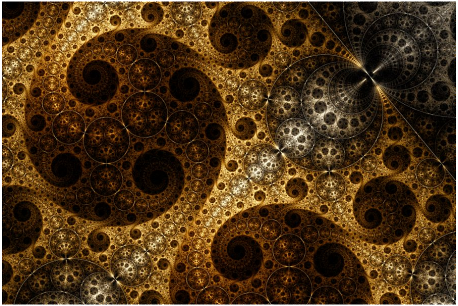




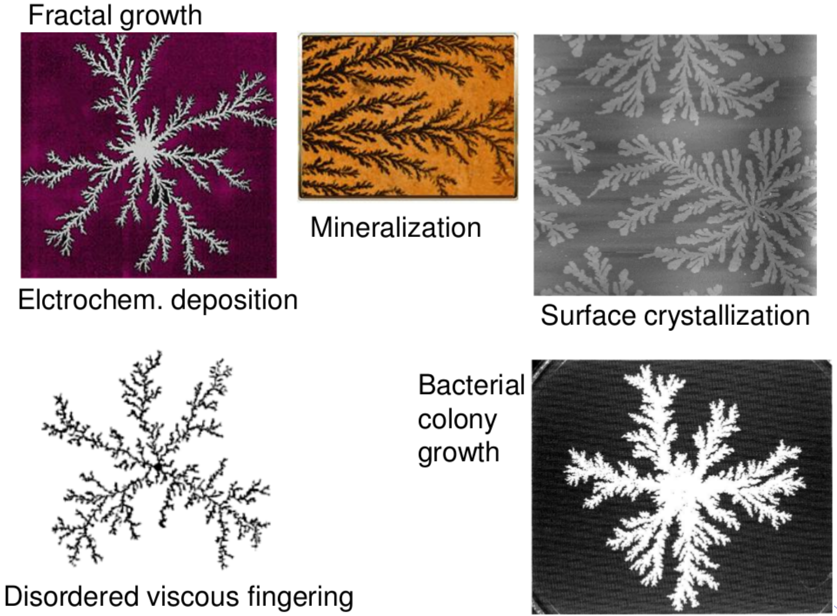
Neural networks

- Learning algorithms:
  - Linear regression
  - Genetic algorithm
  - Simulated annealing

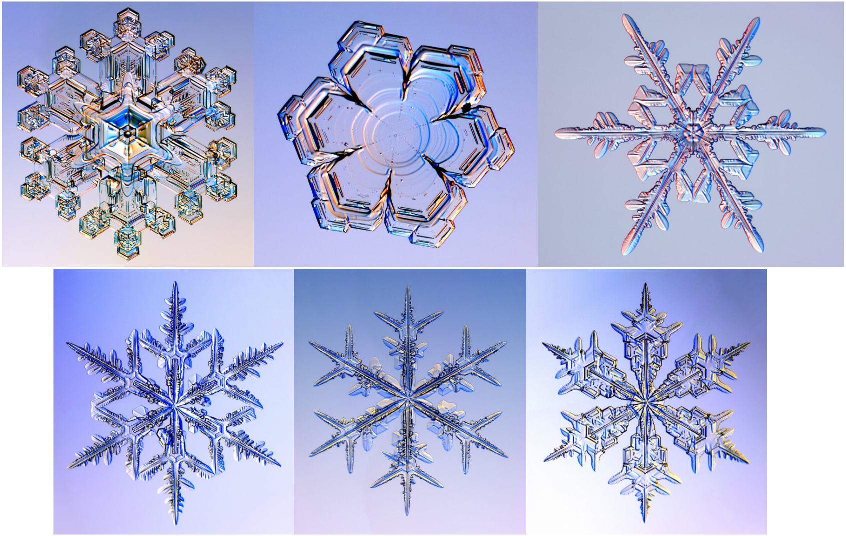
Fractals



Fractal growth



Snowflakes



Fractal growth

Laplacian or gradient governed growth

- Scalar field (electrostatic field, density, through diffusion)

$$\Delta u = 0$$

- Velocity of the interface Γ proportional with the gradient

$$\mathbf{v}|_\Gamma = -C\nabla u|_\Gamma$$

- Boundary condition: potential is curvature (κ) dependent

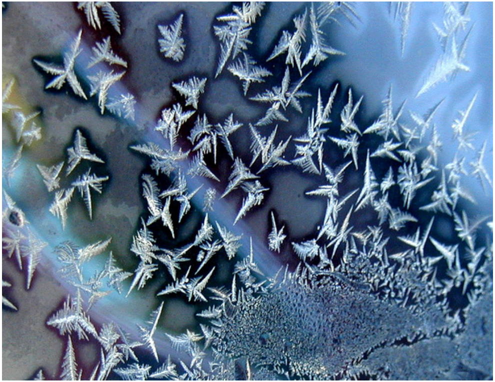
$$u|_\Gamma = f(\nabla u, \kappa)$$

- Disorder: small fluctuations

Fractal growth

Laplacian or gradient governed growth

- Scalar field (electrostatic field, density, through diffusion)
- Velocity of the interface Γ proportional with the gradient
- Boundary condition: potential is curvature (κ) dependent
- Disorder: small fluctuations



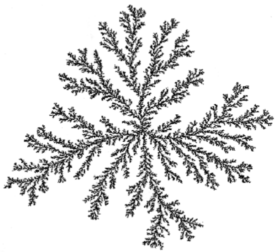
Fractal growth

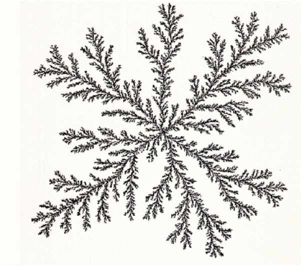
Consequences:

- *Positive growth feedback*: If there is a bump, gradient increases (peak effect), growth gets faster
- *Screening*: Faster bump will screen the slower one
- *Branching*: If tip is far a new bump may grow.
- *Tip splitting*: Tip gets instable and splits

Fractal

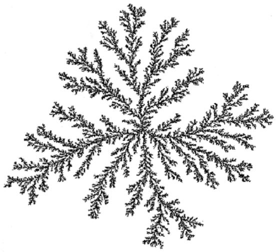
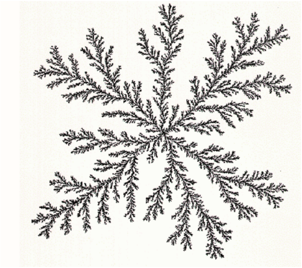
- Self-similarity
- Repeating pattern
- Scaling patterns





Diffusion Limited Aggregation

- Starting from a seed
- Particles come from infinity with diffusion
- If incoming particle touches cluster it gets stuck to it
- Samples: 1m and 100m particles

Scale

Diffusion Limited Aggregation: Algorithm

Basic:

- Start with a seed at (0,0)
- Particles start far from the aggregate and diffuse till they get adjacent to existing cluster

Advanced:

- Start with a seed at (0,0)
- Start random walker on a circle just big enough to cover the cluster
- Define a kill ring big enough or use reentry distribution
- Regions of large jumps, on a larger scale lattice

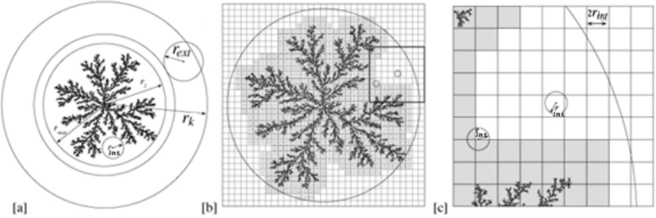


FIG. 1: (a) Schematic representation of the "optimized random trajectories". (b) A DLA aggregate and a mesh of cells $2r_{int} \times 2r_{int}$. Long steps are forbidden in the gray boxes and allowed in the white ones. Also, two long steps are illustrated. (c) A zoom of the region inside the large square in (b).

Diffusion Limited Aggregation: Algorithm

- Start with a seed at (0,0)
- Start random walker on a circle just big enough to cover the cluster
- Define a kill ring big enough or use reentry distribution
- Regions of large jumps, on a larger scale lattice

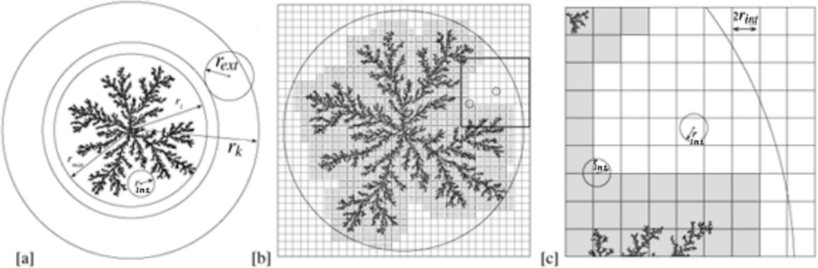
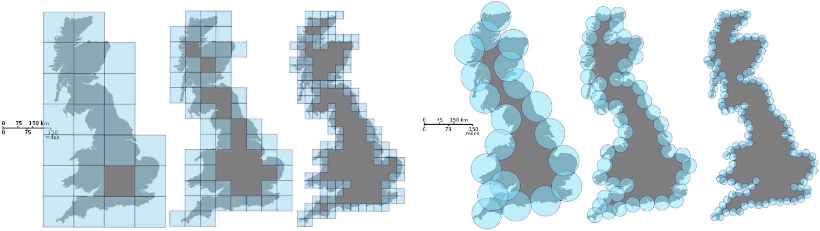
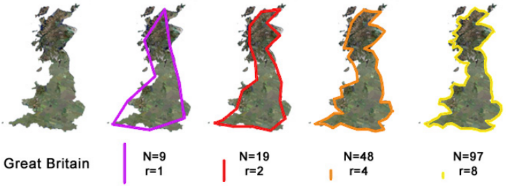


FIG. 1: (a) Schematic representation of the "optimized random trajectories". (b) A DLA aggregate and a mesh of cells $2r_{int} \times 2r_{int}$. Long steps are forbidden in the gray boxes and allowed in the white ones. Also, two long steps are illustrated. (c) A zoom of the region inside the large square in (b).

Dimension

- $d = 0$ point, $d = 1$ line, $d = 2$ plane, etc. Containing space.
- Dimension of a finite object: Cover it
- Hausdorff (fractal) dimension
- Minkowski–Bouligand dimension





Fractal dimension

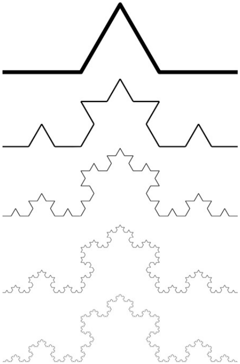
- Fractal dimension
  - Cover the object with boxes of size ε, the fractal dimension is:

$$D = \dim(S) \equiv \lim_{\varepsilon \to 0} \frac{\log N(\varepsilon)}{\log 1/\varepsilon}$$

- Differences:
  - Minkowski–Bouligand: Regular lattice is used
  - Hausdorff: Spheres of given size are used.
- In practice

$$N(\varepsilon) \propto \varepsilon^{D}$$

Fractal dimension: Example



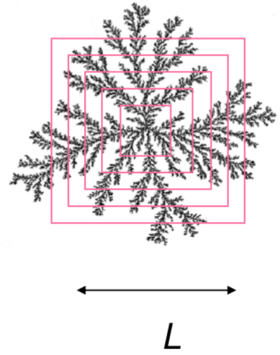
Koch curve

- Start from unit segment
- Hausdorff dimension: cover it with spheres of size $l = 3^{-i}$
- Number of spheres needed $N_l = 4^i$ (take level i!)
- Fractal dimensions:

$$D = \frac{\log N_l}{\log 1/l} = \frac{i \log(4)}{+i \log(3)} = \log_3(4)$$

Fractal dimension: Other methods

- Sandbox method: $M \propto L^D$

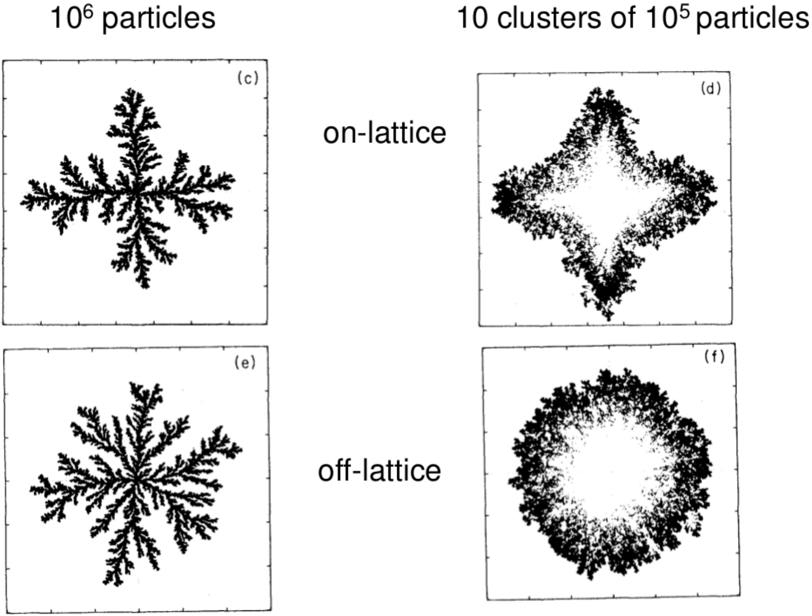


- Correlation functions

$$C(r) = \langle \rho(r)\rho(0) \rangle \propto r^{-\alpha}$$

$$D = d - \alpha$$

DLA: Lattice effects

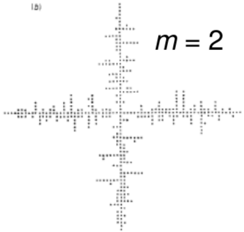


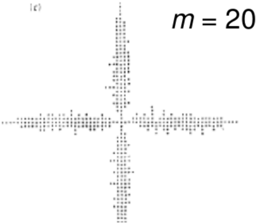
DLA: Lattice effects

DLA on a lattice is anisotropic but splitting tips are observed!
Randomness suppresses the stabilizing effect.

No much difference between lattice and off lattice DLA (a)

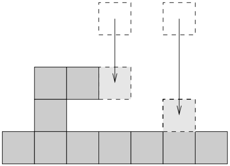
What if we suppress randomness?
„Noise reduction": The growth happens only after the *m*-th particle arrives at the growth site. Ordinary DLA: m=1



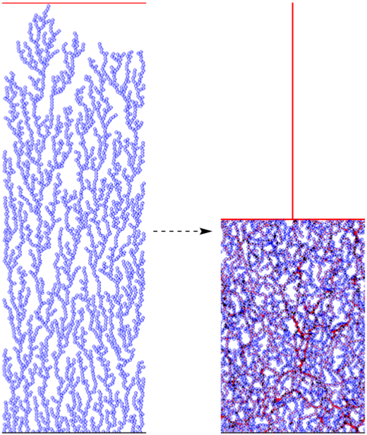


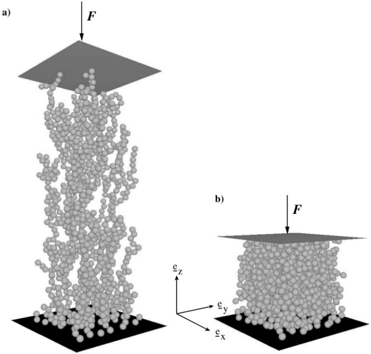
Ballistic deposition

- Lattice



- Off lattice

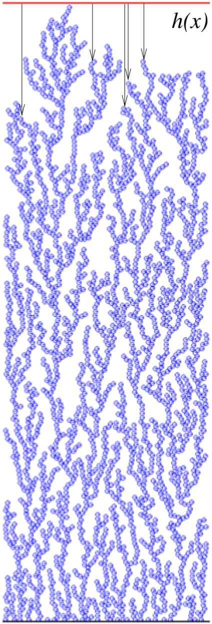




Surface growth models

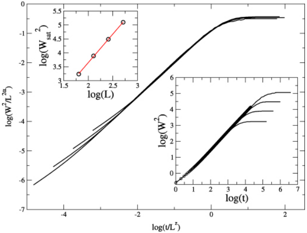
- Not the whole object but only its surface is interesting (e.g. coastline)
- Object starts from a d-dimensional substrate
- Object grows in the $d+1$th dimension.
- Object is described by $h(\mathbf{x})$ ($\mathbf{x}$ is a d-dimensional position vector) height function which is the maximum surface position at $\mathbf{x}$.
- Width of the surface

$$w(L,t)=\sqrt{\frac{1}{L}\int_0^L [h(x,t)-\bar{h}(t)]^2 dx}$$



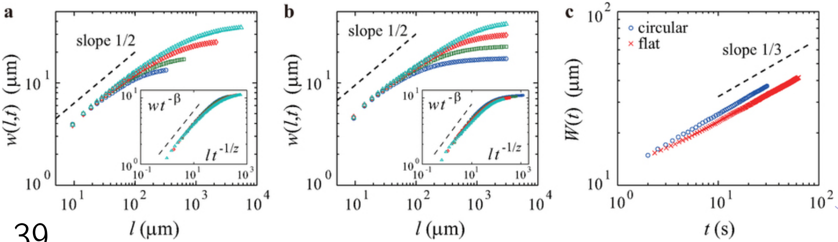
Family-Vicsek scaling

- Change of width in time



- Scaling relation:

$$w(L,t) \propto L^{\alpha} f(t/L^{z})$$



Theory: The KPZ-equation

- Surface growth $\dot{h}(\mathbf{x}, t)$
- Function of: position(?), height, gradient, Laplace of height, noise

$$\dot{h}(\mathbf{x}, t) = f[\mathbf{x}, h(\mathbf{x}, t), \nabla h(\mathbf{x}, t), \Delta h(\mathbf{x}, t), \ldots, \eta(\mathbf{x}, t)]$$

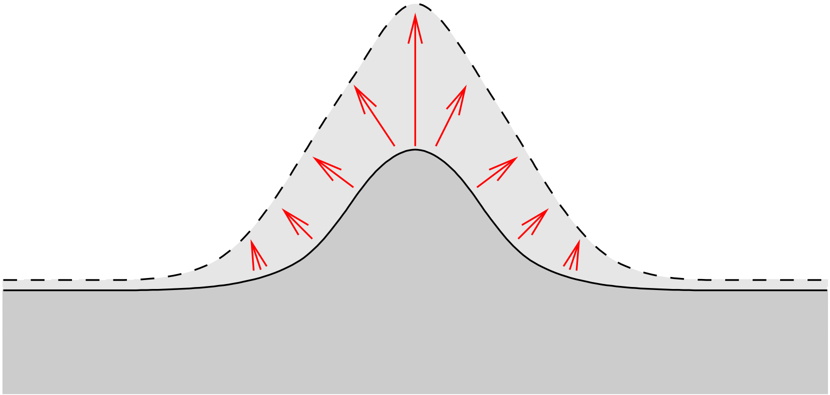
- Normally:

$$\dot{h}(\mathbf{x}, t) = f[h(\mathbf{x}, t), \nabla h(\mathbf{x}, t), \Delta h(\mathbf{x}, t), \eta(\mathbf{x}, t)]$$

- Gaussian noise:

$$\langle \eta(\mathbf{x}, t)\eta(\mathbf{x}', t')\rangle = A\delta(t - t')\delta(x - x')$$
$$P(\eta) = \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{\eta^2}{2\sigma}\right)$$

The Kadar-Parisi-Zhang equation



- Growth is lateral, up to second order

$$\dot{h}(\mathbf{x}, t) = f[(\nabla h(\mathbf{x}, t))^2, \Delta h(\mathbf{x}, t), \eta(\mathbf{x}, t)]$$

The Kadar-Parisi-Zhang equation

$$\dot{h}(\mathsf{x}, t) = \nu \Delta h(\mathsf{x}, t) + \lambda(\nabla h(\mathsf{x}, t))^2 + \eta(\mathsf{x}, t)$$

- Nonlinear
- Stochastic
- Partial differential equation

Discretization in 1D of the KPZ-equation

Space discretization (1+1 dimensions):

$$x_i = i\Delta x,\ h_i = h\left(x_i\right)$$

$$\frac{\partial h}{\partial x}\left(x_i\right) = \frac{h_{i+1} - h_{i-1}}{2\Delta x} + O\left(\Delta x^2\right)$$

$$\left[\frac{\partial h}{\partial x}\left(x_i\right)\right]^2 = \frac{\left(h_{i+1} - h_{i-1}\right)^2}{4\Delta x^2} + O\left(\Delta x^2\right)$$

$$\frac{\partial^2 h}{\partial x^2}\left(x_i\right) = \frac{h_{i+1} - 2h_i + h_{i-1}}{\Delta x^2} + O\left(\Delta x^2\right)$$

$$\Downarrow$$

$$\frac{\mathrm{d}h_i}{\mathrm{d}t} = \frac{1}{\Delta x^2}\left[\nu\left(h_{i+1} - 2h_i + h_{i-1}\right) + \frac{\lambda}{4}\left(h_{i+1} - h_{i-1}\right)^2\right] + \text{noise.}$$

Numerical solution of the KPZ-equation

- ξ is a random number with zero mean (can be Gaussian, or uniform)
- Due to noise Euler scheme is enough:

$$
\begin{aligned}
h_i(t+\Delta t) =& h_i(t) + \nu \frac{\Delta t}{(\Delta x)^2} \left[h_{i+1}(t) - 2h_i(t) + h_{i-1}(t) \right] + \\
&+ \frac{\lambda}{4} \left[h_{i+1}(t) - h_{i-1}(t) \right] + \xi_i
\end{aligned}
$$

- Critical exponents and and universality classes $\alpha = 1/2$, $z = 3/2$