Computer simulations in Physics

Optimization

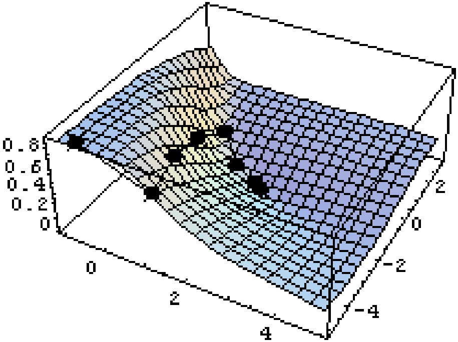
Janos Török

Department of Theoretical Physics

April 27, 2023

Optimization

- General problem of finding the ground state
- Phase-space:
- Arbitrary number of dimensions
- Methods:
  - Steepest Descent
  - Stimulated Annealing
  - Genetic algorithm



Optimization

- General problem of finding the ground state
- Phase-space:
- Arbitrary number of dimensions
- Methods:
  - Steepest Descent
  - Stimulated Annealing
  - Genetic algorithm
- Implementation
  - C: GSL
  - python: scipy.optimize
  - Both are very flexible and can be used with numerical or analytical derivatives

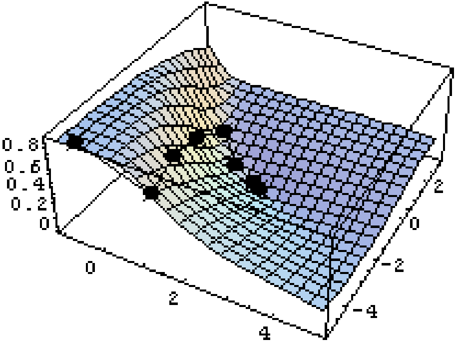
Gradient based optimization

- Given $f(\mathrm{x})$, with $\mathrm{x} = \{x_1, x_2, \dots x_n\}$
- Gradient $\nabla f(\mathrm{x}) \equiv \mathrm{g}(\mathrm{x}) = \{\partial_1 f, \partial_2 f, \dots \partial_n f\}$
- Second order partial derivatives: square symmetric matrix called the *Hessian matrix*:

$$\nabla^2 f(\mathrm{x}) \equiv H(\mathrm{x}) \equiv \begin{pmatrix} \partial_1\partial_1 f & \dots & \partial_1\partial_n f \\ \vdots & \ddots & \vdots \\ \partial_1\partial_n f & \dots & \partial_n\partial_n f \end{pmatrix}$$

General Gradient Algorithm

1. Test for convergence
2. Compute a search direction
3. Compute a step length
4. Update x



Steepest descent algorithm

1. Start from x_0
2. Compute $\mathrm{g}(\mathrm{x}_k) \equiv \nabla f(\mathrm{x}_k)$. If $||\mathrm{g}(\mathrm{x}_k)|| \leq \varepsilon_g$ then stop, otherwise, compute normalized search direction $\mathrm{p}_k = -\mathrm{g}(\mathrm{x}_k)/||\mathrm{g}(\mathrm{x}_k)||$
3. Compute α_k such that $f(\mathrm{x}_k + \alpha \mathrm{p}_k)$ is minimized
4. New point: $\mathrm{x}_{k+1} = \mathrm{x}_k + \alpha \mathrm{p}_k$
5. Test for $|f(\mathrm{x}_{k+1} - f(\mathrm{x}_k))| \leq \varepsilon_a + \varepsilon_r |f(\mathrm{x}_k)|$ and stop if fulfilled in two successive iterations, otherwise go to 2.

Conjugate Gradient Method

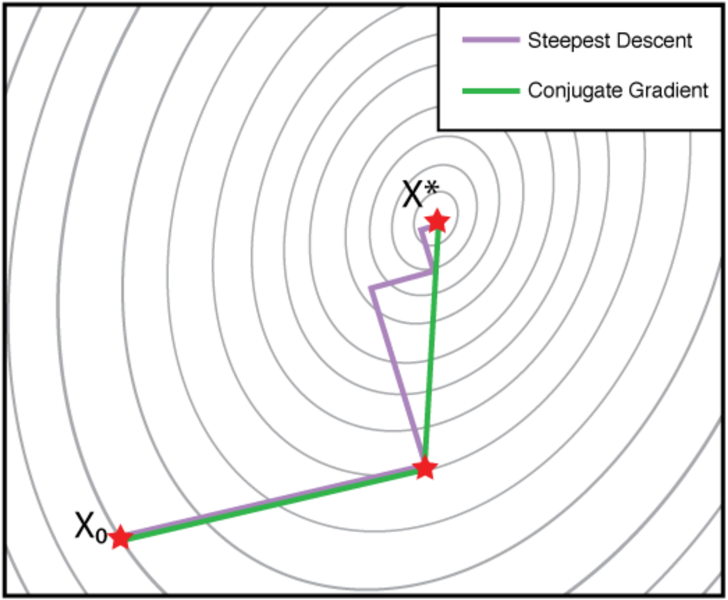
- The iteration
$$x_{n+1} = x_k - \gamma_n \nabla f(x_k),$$
- We can select γ such that if the function is quadratic in all directions it goes immediately into the minimum
- Idea: almost all minima are quadratic close to the minimum

$$\gamma_n = \frac{|(x_n - x_{n-1})^T [\nabla f(x_n) - \nabla f(x_{n-1})]|}{||\nabla f(x_n) - \nabla f(x_{n-1})||^2}$$

Conjugate Gradient Method

1. Start from x_0
2. Compute $g(x_k) \equiv \nabla f(x_k)$. If $||g(x_k)|| \leq \varepsilon_g$ then stop, otherwise Go to 6
3. $p_0 = -g_0$
4. Compute $g(x_k) \equiv \nabla f(x_k)$. If $||g(x_k)|| \leq \varepsilon_g$ then stop, otherwise continue
5. Compute the new conjugate gradient direction $p_k = -g_k + \beta_k p_{k-1}$, where
$$\beta = \left(\frac{||g_k||}{||g_{k-1}||} \right)^2$$
6. Compute α_k such that $f(x_k + \alpha p_k)$ is minimized
7. New point: $x_{k+1} = x_k + \alpha p_k$
8. Test for $|f(x_{k+1} - f(x_k))| \leq \varepsilon_a + \varepsilon_r |f(x_k)|$ and stop if fulfilled in two successive iterations, otherwise go to 4.

Conjugate Gradient Algorithm



Modified Newton's method

Second order method

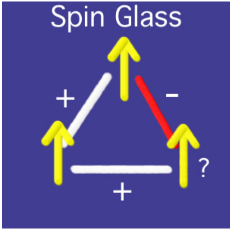
1. Start from $\mathbf{x}_0$
2. Compute $\mathbf{g}(\mathbf{x}_k) \equiv \nabla f(\mathbf{x}_k)$. If $||\mathbf{g}(\mathbf{x}_k)|| \leq \varepsilon_g$ then stop, otherwise, continue
3. Compute $H(\mathbf{x}_k) \equiv \nabla^2 f(\mathbf{x}_k)$ and the search direction $\mathbf{p}_k = -H^{-1}\mathbf{g}_k$
4. Compute α_k such that $f(\mathbf{x}_k + \alpha \mathbf{p}_k)$ is minimized
5. New point: $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha \mathbf{p}_k$
6. Go to 2.

Glassy behavior, frustration

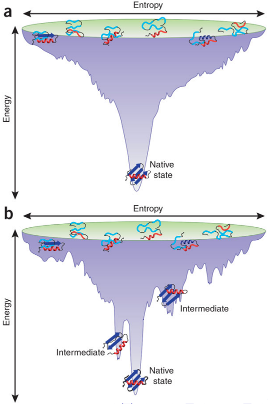
- Model glass: spin glass:

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

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

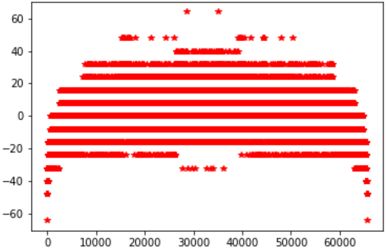


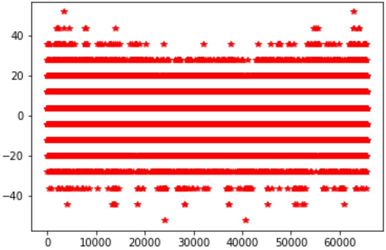
Rugged energy landscape.



Energy landscape

Ising vs. spin glass (X Axis: binary representation of number)

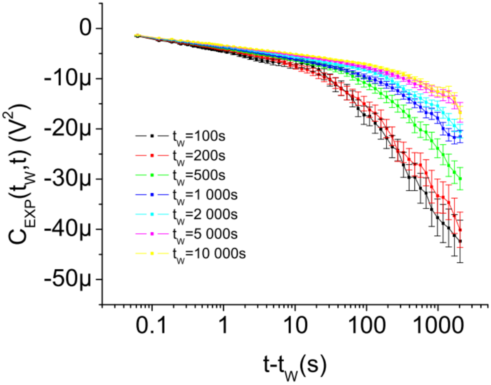




Spin glass: Aging

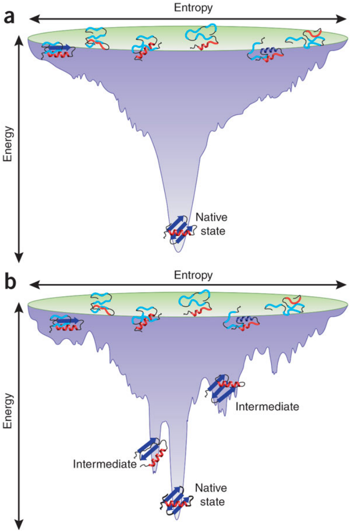
- Heat up the sample where it equilibrates fast
- Quench it below T_c
- Wait t_w
- Measure a parameter $q(t_w, t_w + t)$
- Often q is a covariance (X observable):

$$q(s,t) = E(X_t X_s) - E(X_t)E(X_s)$$



Spin glass: Trap model (Bouchaud)

- The evolution of the particle system is represented by a Markov process in a random energy landscape
- The process will spend most time into deep valleys of lowest energy where it will be trapped
- The time spent in these valleys is random and aging will appear when the mean time spent in these valleys diverges
- Order parameter: the magnetization and the two point spin correlation between spins at the same site in two different replicas



Rugged energy landscape

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

Optimization

- General optimization
- Parameters of the system x (input)
  - for networks: adjacency matrix, degree distribution
  - for pattern recognition: data, or processed data (e.g Fourier spectrum, etc.)
- Optimized property: $y = f(\mathrm{x})$, we search for $f(.)$ which gives the desired y
  - any measurable quantity
  - classification of data (e.g. $y = 1$ for cat, $y = 2$ for dog, etc.)
- Loss function, $L(f)$, the quantity to be minimized (Energy/Hamiltonian)
  - Least square: $L(f) = [y - f(\mathrm{x})]^2$
  - Hamming distance: $L(f) = \begin{cases} 1 & \text{if } f(\mathrm{x}) = y \\ 0 & \text{otherwise} \end{cases}$

Simulated annealing

- Loss function: e.g. energy E
- Minimize energy like in a physical system
- Vary parameter set w in an egodic way (all possible values must be reachable)
- Observe detailed balance:

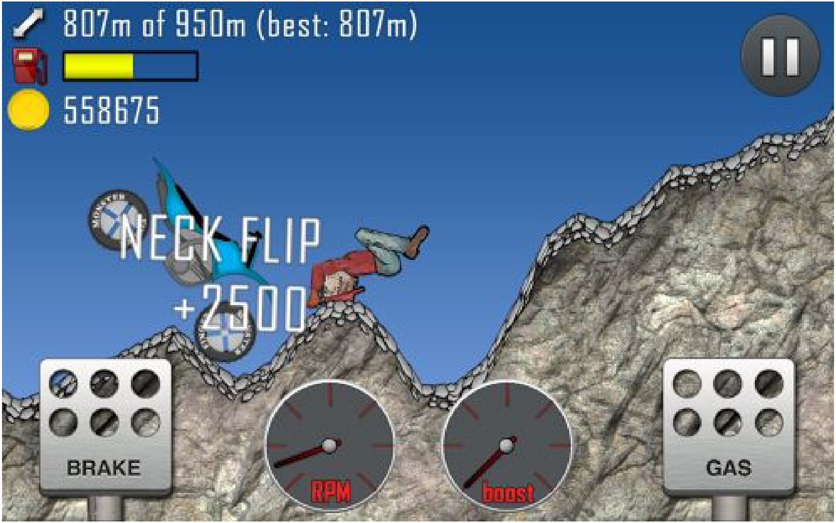
$$p(i \to j) = \begin{cases} 1 & \text{if } E_j < E_i \\ \exp[\beta(E_i - E_j)] & \text{otherwise} \end{cases}$$

- where $\beta \simeq 1/T$
- Slowly decrease T

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

Hill climb


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

Travelling salesman

- N cities on the 2d space
- Distance between the cities is the Euclidean distance (birds flight)
- The traveller must visit all cities once
- The trajectory is circular so the traveller must return to the starting city
- The optimized quantity is the travelled distance

Travelling salesman

- Minimal travelling path for visiting a number of cities
- Elementary move: swap two cities ($T \sim$ alcohol)

Genetic algorithm

- Learn from nature


Genetic algorithm

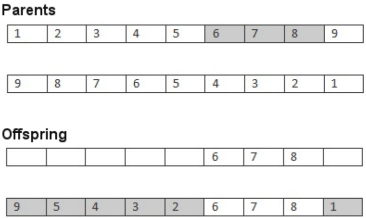
- Learn from nature
- Let the fittest to survive
  - Fitness function, e.g. energy, length, etc.
- Combine different strategies
- State is represented by a vector (genetic code or genotype)
  - Phasespace, city order, neural network parameters, etc.
- Offsprings have two parents with shared genetic code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

Genetic algorithm terminology

- Chromosome: Carrier of the genetic representation
- Gene: Smallest units in the chromosome with individual meaning
- Fitness: The measure of the success of an individual with a given chromosome
- Population: Set of chromosomes from which the parents are selected. Its size should be larger than the length of the chromosome
- Parents: Pair of chromosomes, wich produce offsprings
- 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.

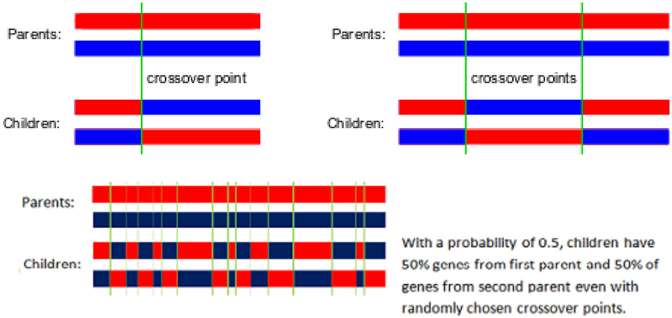
Genetic algorithm schema

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)



Genetic algorithm: Reproduction

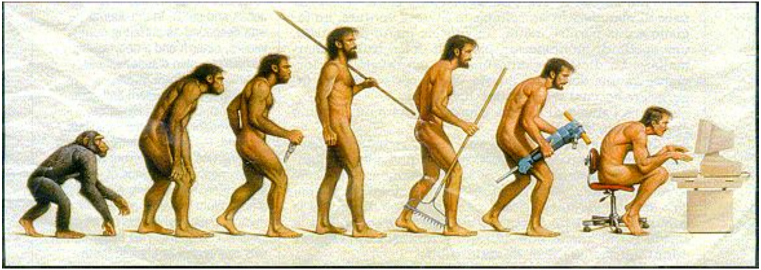
- Two parents and two children



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
5. Mutation: small rate

Genetic algorithm example



Genetic algorithm for Travelling Salesman

- Natural storage: order of the towns, e.g. $(1, 7, 4, 2, 3, 5, 6)$ not suitable for crossover.
- Good encoding can be cut at any point.
- Solution: *ordinal representation*
- In representation i means take element i from the rest of the list of cities.

Jean-Yves Potvin: Genetic algorithms for the traveling salesman problem, Université de Montréal

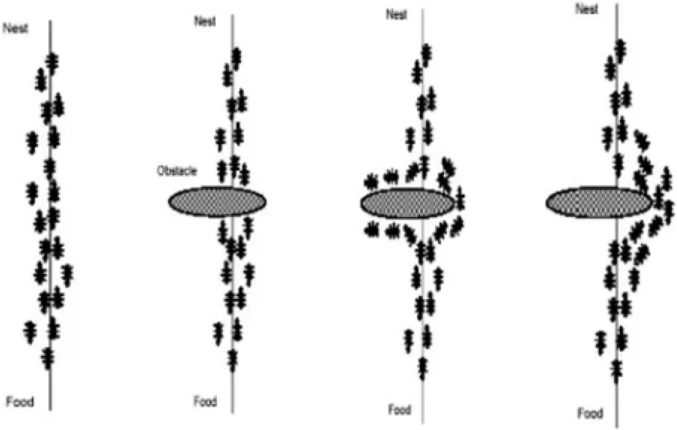
Ordinal representation: Example

- The city numbers are gradually taken from the standard list of cities
- The code is the actual number in the maimed list
- Any number sequence with values $([1, N][1, N-1] \cdots [1, 2]1)$ is valid

City order	Maimed list	Ordinal
1 5 2 4 6 3	**1** 2 3 4 5 6	1
1 **5** 2 4 6 3	2 3 4 **5** 6	1 4
1 5 **2** 4 6 3	**2** 3 4 6	1 4 1
1 5 2 **4** 6 3	3 **4** 6	1 4 1 2
1 5 2 4 **6** 3	3 **6**	1 4 1 2 2
1 5 2 4 6 **3**	**3**	1 4 1 2 2 1

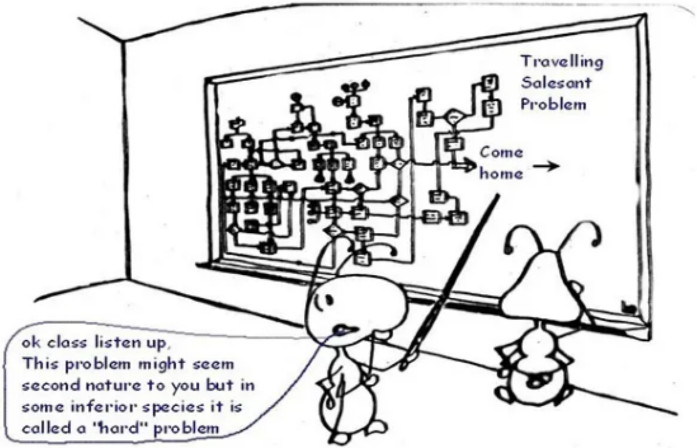
Ant colony optimization

- Once again learn from nature:



Ant colony optimization

- Once again learn from nature:
  - Ants explore
  - Deposit pheromone
  - Pheromone dissipates with time
  - Shorter paths with more usage will prevail



Ant colony optimization: algorithm

1. N ants
2. Initialize pheromone concentration h which is between all city paires by small random values
3. Ants explore the cities:
   - Ants may only go to unvisited cities
   - Probability to go from city i to j is proportional to

     $$p_i \sim h_{ij}^{\alpha} / d_{ij}^{\beta},$$

     where d_{ij} is the distance between cities i and j, α, β are parameters
4. Ants deposit pheromone on their travelled paths. The amount of deposited pheromone is $1/d_{ij}$
5. Pheromone decay: multiply all elements of the h matrix with parameter $\gamma < 1$
6. Repeat from 3

https://github.com/Akavall/AntColonyOptimization

Ant colony optimization: assessment

- Advantages
  - Inherent parallelization
  - Generally rapid solution
  - Very good for dynamic optimization
- Disadvantages
  - Individual behaviour is stochastic and not representative
  - Theory is kind of impossible
  - Steady state is incertain