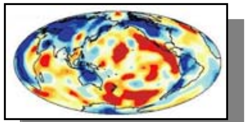
Genetic Algorithms

- A biological analogy for optimization problems
- Bit encoding, models as strings
- Reproduction and mutation -> natural selection
- Pseudo-code for a simple genetic algorithm

$$\sigma(d,m) = k\frac{\rho(d,m)\theta(d,m)}{\mu(d,m)}$$

Genetic Algorithms



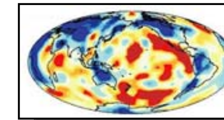
The goal of **genetic algorithms (GA)**:

- Propagate members of a population through many generations
- By carefully influencing wedding plans, decide which partner will have kids
- always keep a whole population (ensemble of models) and try to improve their overall quality
- evalute each member for fitness, allow only the fittest to have sex
- the weakest members will die out

GA's mimic natural selection (Darwin)

$$\sigma(d,m) = k\frac{\rho(d,m)\theta(d,m)}{\mu(d,m)}$$

GA - terminology



- in GA a population is considered as a large ensemble of chromosomes (each member, i.e. model, is described by a long string, like DNA)
- a chromosome is composed of individual *genes* (this would correspond to the parameters of our problem)
- the only way individuals are changed is by crossover (the kids of two adults) and some mutation (a random perturbation of the genetic material)

Example:

crossover

Parent 1: 0 1 0 0 1 1 0 | 1 0 0 0 0 0 1 0

Parent 2: 1 0 0 0 0 0 1 | 1 1 0 0 0 0 0 1

Child 1: 0 1 0 0 1 1 0 1 1 0 0 0 0 0 1

Child 2: 1 0 0 0 0 0 1 1 **1** 0 0 0 0 1 0

mutation

$$\sigma(d,m) = k\frac{\rho(d,m)\theta(d,m)}{\mu(d,m)}$$

GA - real life

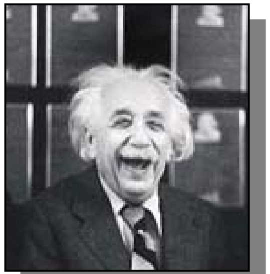
Let's make a real life example.

Human genotypes are

Eye color	Hair color	Skin color	height	streng th	Brain size	music ality	coordi nation	maths

Phenotypes of Albert Einstein

brown	black	bright	Avg.	Avg.	Avg.	high	low	mediu m



Phenotypes of Sharon Stone

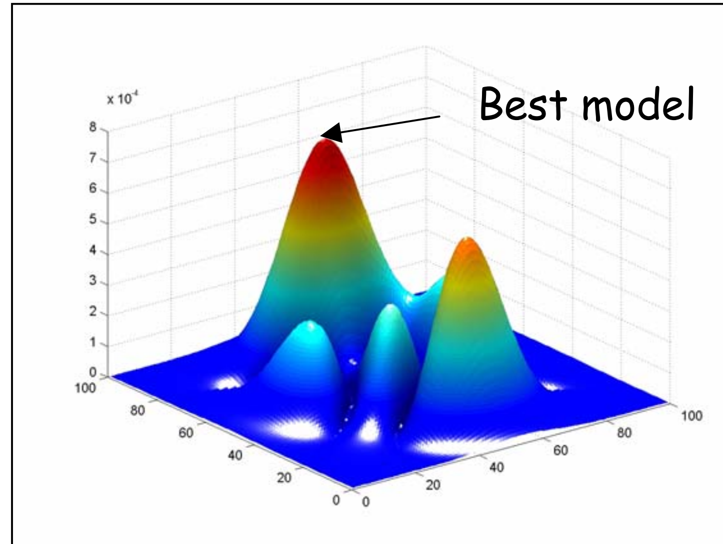
green	blond e	fair	Avg.	Avg.	Avg	high	good	good

Let us not speculate what their baby would look like …

$$\sigma(d,m) = k\frac{\rho(d,m)\theta(d,m)}{\mu(d,m)}$$

What is an individual?

The goal of a GA is to find (or better to breed) the best individual from the available genetic pool.



Examples:

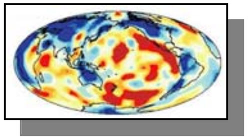
In our *peaks* function, each individual would be described by two numbers, the x and y coordinate. A fundamental (but actually not absolutely necessary) aspect of GA's is the coding of an individual into string. If you use a Binary representation, it s called **bit coding**

	x	y
Model 1: (3,25)	-> 001011	010101
Model 2: (16,97)	-> 100101	101010
…		
Model N: (23,45)	-> 001010	110111

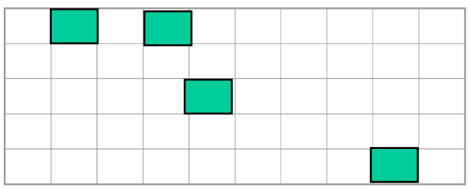
This can of course be carried out for an arbitrary number of parameters (properties).

$$\sigma(d,m) = k\frac{\rho(d,m)\theta(d,m)}{\mu(d,m)}$$

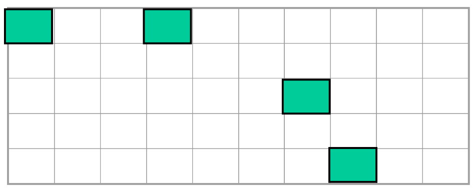
Lets get (geo)physical .....



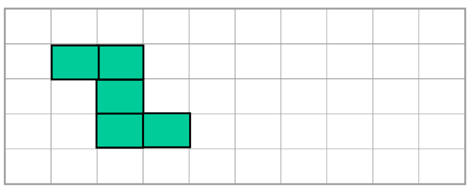
A natural example for a string representation
for a geophysical inverse problem is our treasure hunt.



01010000 010



10010000 0100



00000 0000

Here the actual binary coding would make sense. Here 0 would mean sand and 1 would represent gold. What if we can to describe a function taking on arbitrary values?

Example: We want to invert for a depth dependent velocity model, described by layer thickness d and velocity c. Then a model vector m would look like:

$m = (d_1,v_1, d_2,v_2, d_3,v_3, d_4,v_4, d_5,v_5, d_6,v_6, \; .... \; d_n,v_n)$

... and would simply be described by a long bit-string. It is your choice how many bits you use for the possible range of values for each parameter.

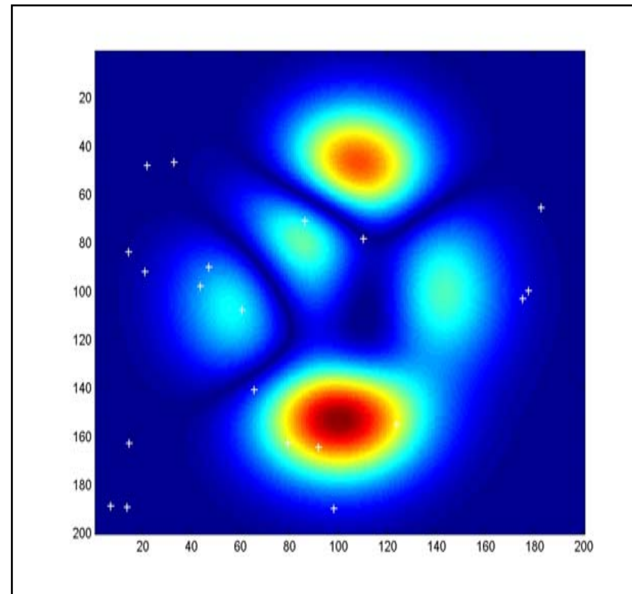
$$\sigma(d,m) = k\frac{\rho(d,m)\theta(d,m)}{\mu(d,m)}$$

Initial population

First step: For illustration we try to find the maximum of the peaks function with only two parameters (x,y). In contrast to simulated annealing we are not working with one model which we perturb, but we work with a model population.

We decide we work on an initial model size n=20.

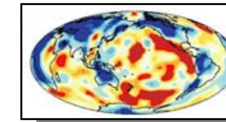
The initial population was not influenced by the (quality, misfit, probability, etc.) function which will be responsible for their future development. The model parameters will be (but don't have to be) **bit-coded**.



Initial population (white crosses)

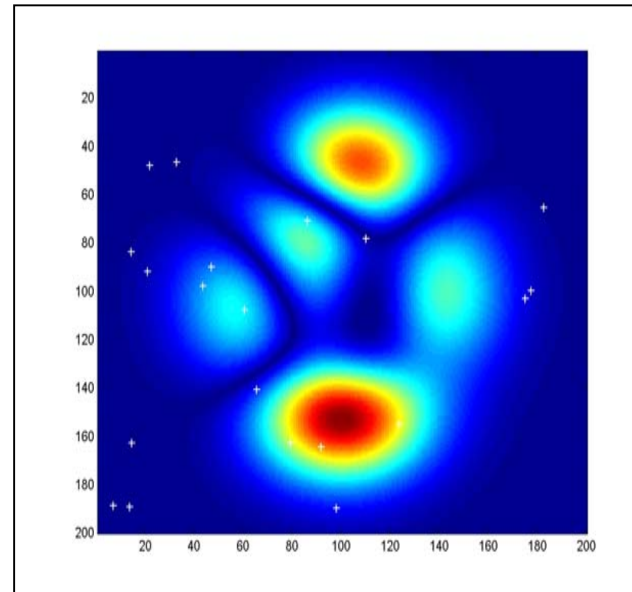
$$\sigma(d,m) = k\frac{\rho(d,m)\theta(d,m)}{\mu(d,m)}$$

Quality, Fitness, Probability



Second step: Calculate the quality (misfit, probability, etc.) of each individual of the population:

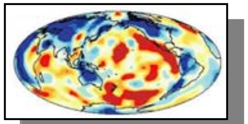
	x	y	Quality
Model 1:	24	12	0.12
Model 2:	3	67	0.24
Model 3:	87	34	0.45
.			
.			
.			
Model n:	21	56	0.23



Note that in **geophysical inverse problems** the calculation of the quality would correspond to the solution of a forward problem and calculation of the data misfit.

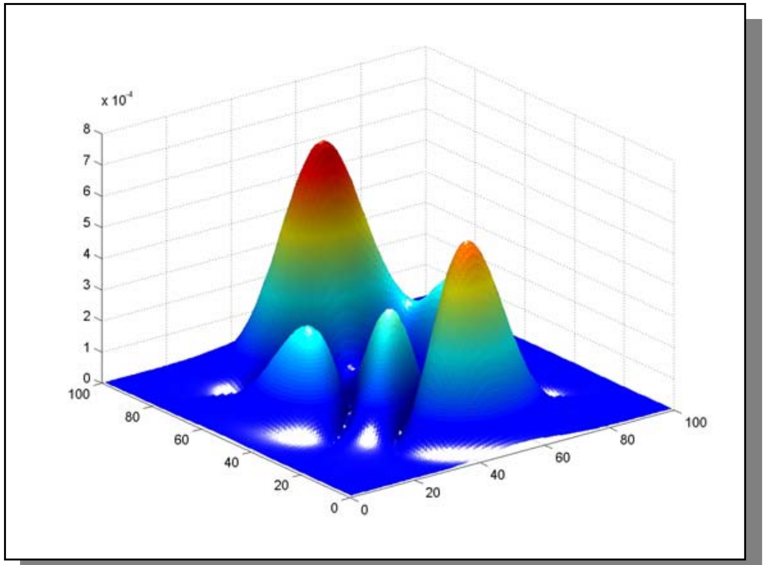
$$\sigma(d,m) = k\frac{\rho(d,m)\theta(d,m)}{\mu(d,m)}$$

Order - ranking

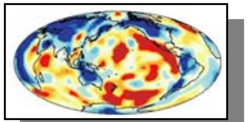


Third step: Now rank the individuals (models) according to fitness (quality, probability, etc):

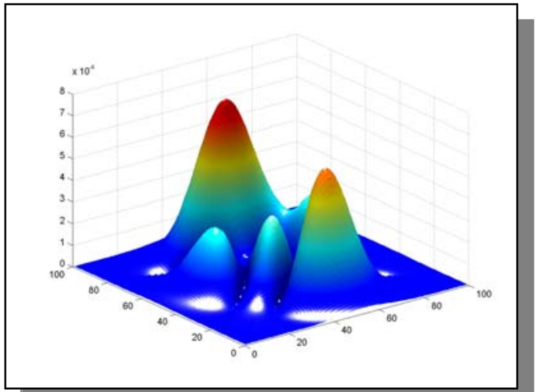
Rank	Model	x	y	Quality
1	18	12	34	0.81
2	12	45	56	0.67
3	3	1	56	0.65
.				
.				
.				
N	7	24	98	0.01



$$\sigma(d,m) = k\frac{\rho(d,m)\theta(d,m)}{\mu(d,m)}$$

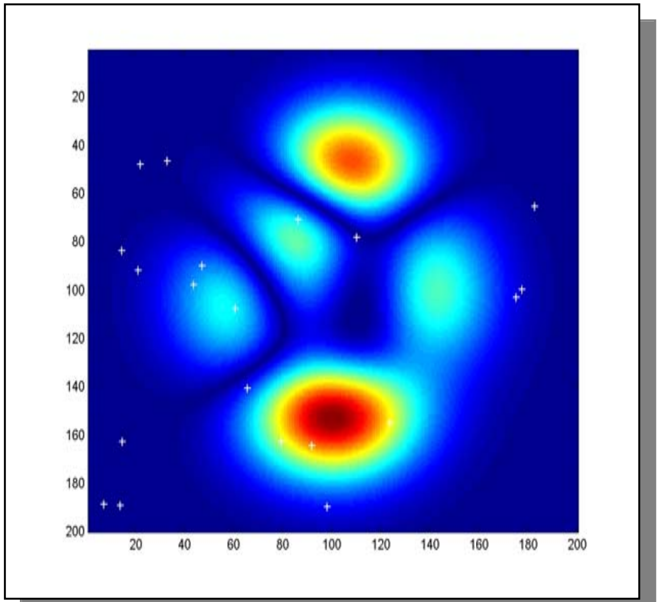


Ranking



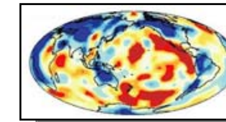
Fourth step: Decide how many of the bad models you want to sent to Siberia (e.g. half). Then **reproduction** begins with the remaining models.

Rank	Model	x	y	Quality
1	18	12	34	0.81
2	12	45	56	0.67
3	3	1	56	0.65
.				
N/2	5	11	44	0.58
.				
.	-> Siberia			
.				
N				



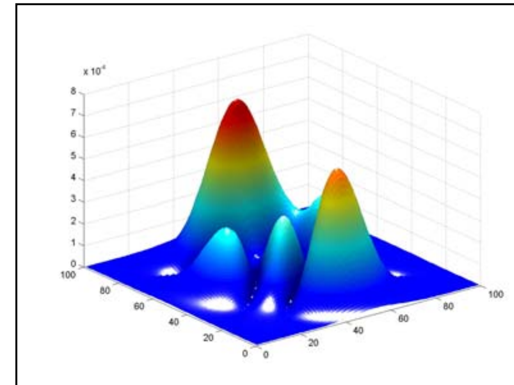
$$\sigma(d,m) = k\frac{\rho(d,m)\theta(d,m)}{\mu(d,m)}$$

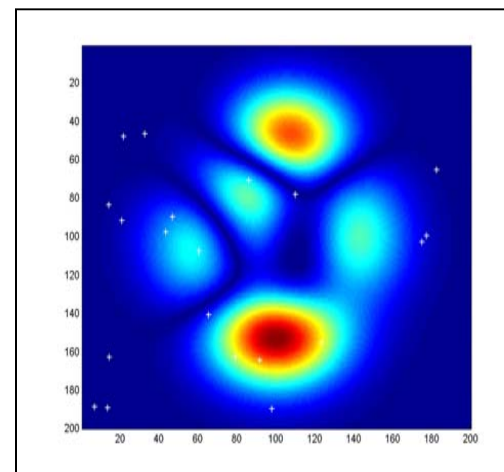
Fitness



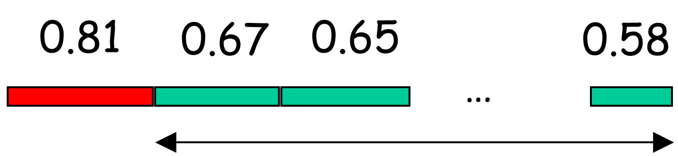
Fifth step: Go through the remaining n/2 models and find a partner with a probability proportional to its fitness.

Rank	Model	x	y	Quality
1	18	12	34	0.81
2	12	45	56	0.67
3	3	1	56	0.65
.				
N/2	5	11	44	0.58





Finding a partner for Model 18:

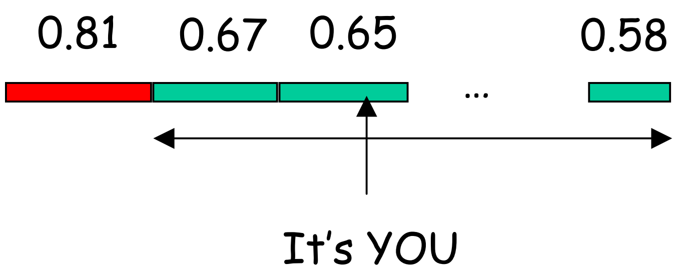


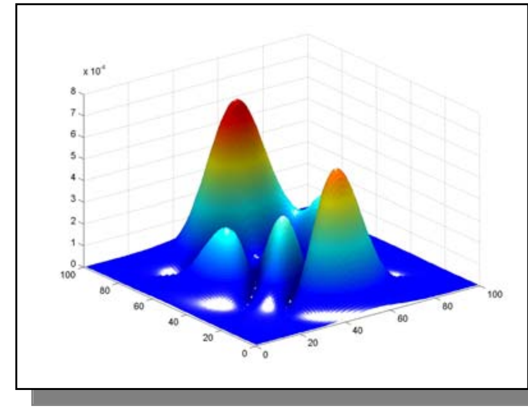
Generate a random number between 0 and max. A "better" model will be picked more often.

$$\sigma(d,m) = k\frac{\rho(d,m)\theta(d,m)}{\mu(d,m)}$$

Reproduction

Finding a partner for Model 18:





Model 3 was picked as partner for model 18, now they mate:

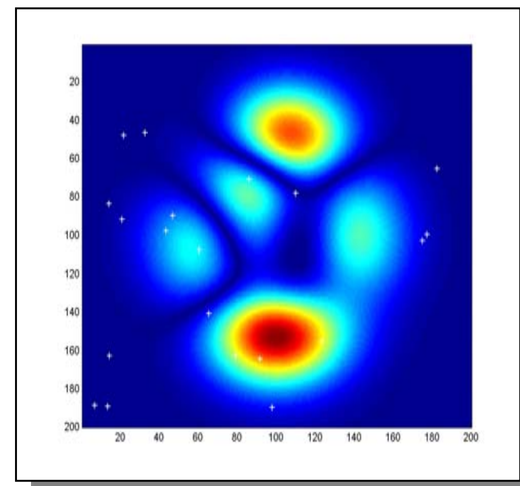
Model 18 (12,34) -> 001010 101001

Model 3 (1,56) -> 101100 010110

Randomly choose a crossover point, we get the babies:

Baby 1: 00101010 | 0110 -> (24, 56)

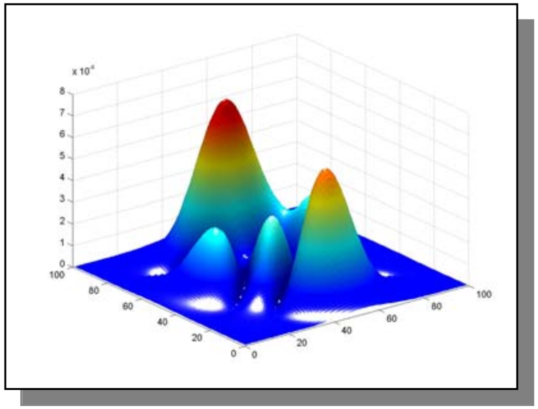
Baby 2: 10110001 | 1001 -> (76,87)



$$\sigma(d,m) = k\frac{\rho(d,m)\theta(d,m)}{\mu(d,m)}$$

Mutation

We keep the n/2 best models for the next population and we have n/2 new members, the babies. Each baby has an occasional flip of one of its bits with a (usually low) probability.



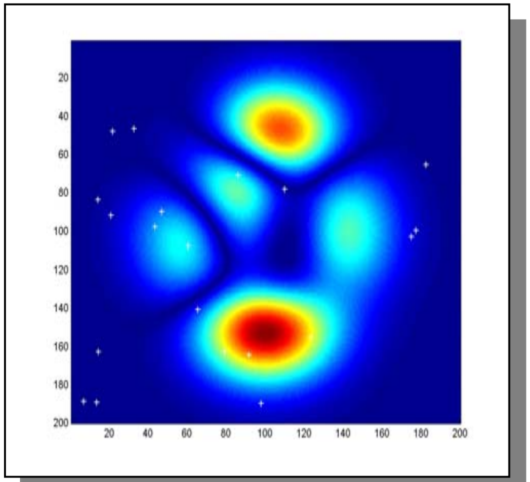
Baby 1: 00101010 | 0110 -> (24, 56)

Baby 2: 10110001 | 1001 -> (76,87)

↓

Baby 1: 00**1**11010 | 0110 -> (**13**, 56)

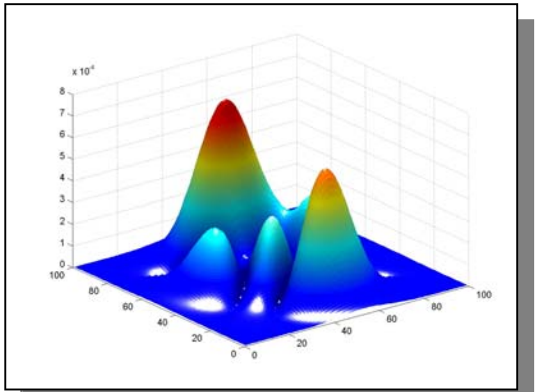
Baby 2: 10110001 | 1001 -> (76,87)



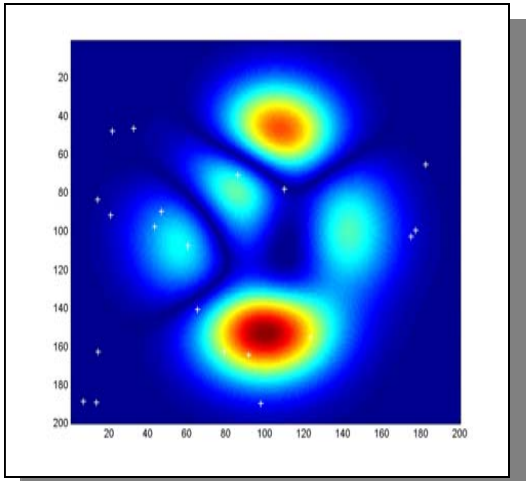
Just one of the babies underwent mutation and it affected only one of the parameters.

$$\sigma(d,m) = k\frac{\rho(d,m)\theta(d,m)}{\mu(d,m)}$$

Iteration



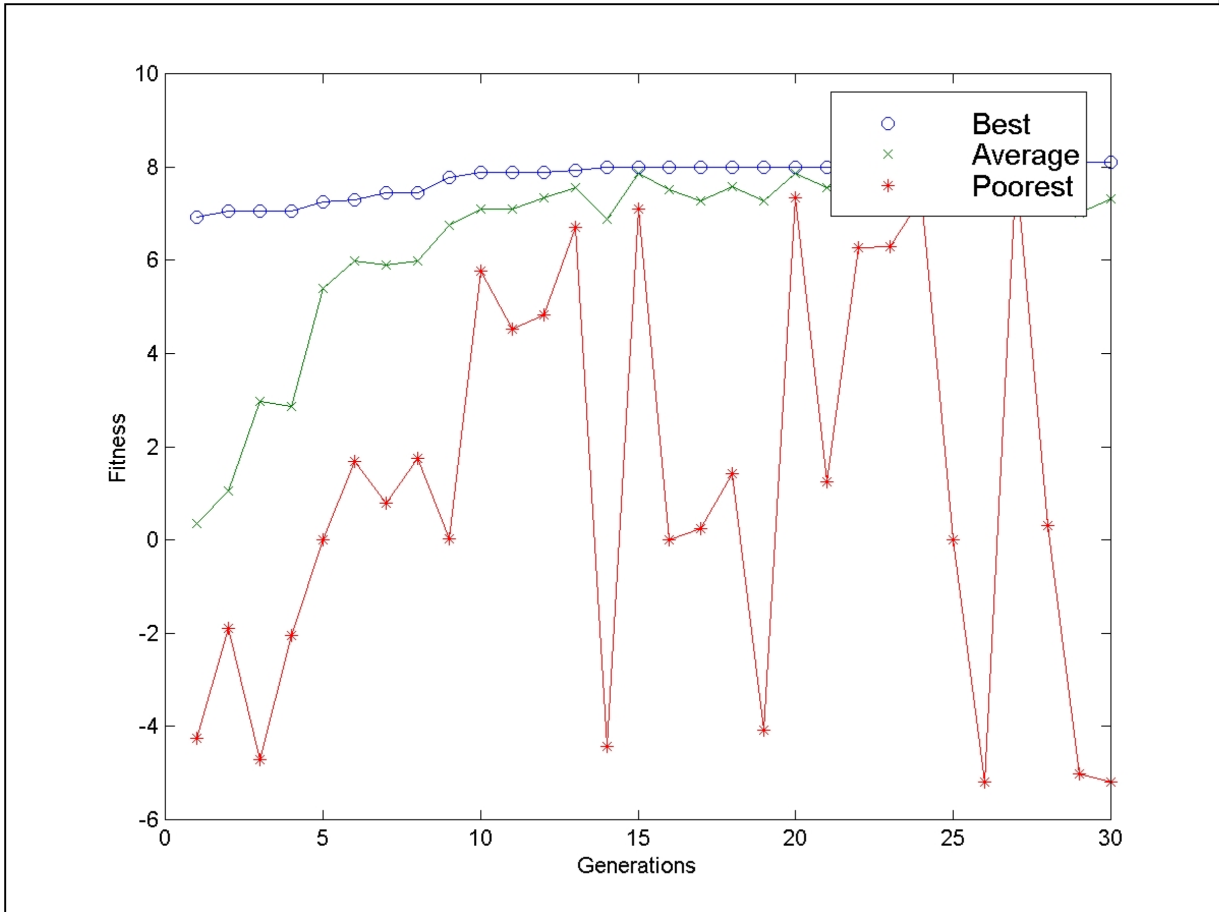
The process of **reproduction, crossover and mutation** is repeated until either the average fitness (quality, probability, etc.) of the population has reached an acceptable value or one of the members (the star) has such incredible quality that you decide he is the winner.

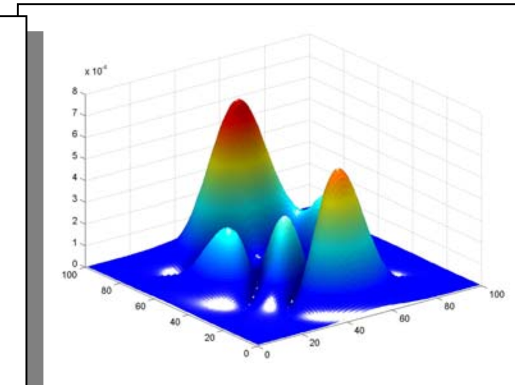


In principle the average quality of the population should increase with each generation. However, this strongly depends on some of the parameters (e.g. the probability for mutation) and the nature of the fitness (quality, probability) function.

$$\sigma(d,m) = k\frac{\rho(d,m)\theta(d,m)}{\mu(d,m)}$$

Example for peaks





Summary

A Genetic algorithm is an optimization method based on the biological analogy of “survival of the fittest”.

In contrast to simulated annealing where only one model is perturbed and walked through the model space in genetic algorithms an ensemble of models is always considered.

Through analogies of genetic reproduction, crossover, mutation the quality of the average population and the individuals is improved over several generations.

Genetic algorithms can be considered as a special case of the more general **evolutionary algorithms**.