Non derivative methods for optimization - Genetic algorithm and Simulated annealing

In usual context 'Optimization' refers to finding the solution for a function containing a set of parameters, by performing minimization of the difference between the ideal solution (calculated value) and the observed solution (experimental value). Many of the physical problem can be formulated in mathematical terms with unknown but determinable parameters . For example, if the growth of a plant with respect to its environment has to be studied, then we can list all the factors related to growth like sunlight, water, minerals, air (oxygen and carbon dioxide) and systematically fix all factors except one and observe the growth pattern. As an example, we can start with seven plants by placing them in a well lit, spacious environment, but varying only the water supply that each plant receives.

After 10 days of experimentation, we tabulate the height of the plant as a measure of its growth.

Label	Water (mL/day)	Growth (cm)
1	0	0
2	3	10
3	5	12.5
4	10	15.38
5	15	16.67
6	30	18.18
7	70	19.18

Table: The growth measured at the end of 10 days seen varying with quantity of water.

We see that the quantitative relationship between growth and water, is not linear but a curve reaching saturation by 30^{th} ml . The mathematical form for this non-linear curve can be obtained through detailed modeling; instead we can also write its equation from the shape of the curve, here we identify it to Michaelis-menten model.

$$\text{Growth} = \frac{a * water(ml)}{b + water(ml)}$$

Equation: Michaelis menten model:- Growth is the dependent variable and water is the independent variable,the parameters 'a' and 'b' are the constants.

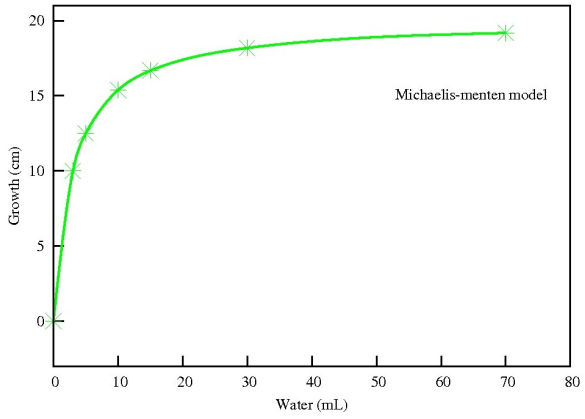


Figure: The growth of plant fitted with Michaelis menten model using a, b as 20 and 3 respectively.

As the variable 'growth' (y) is dependent on the variable 'quantity of water' (x), the former and the latter are called as dependent and independent variable respectively. The terms 'a' and 'b' in the above model are called as the constants or parameters. In fact, these constants represent a quantitative measure for the combination of other growth influencing factors i.e, sunlight, minerals, air. If we want to find the values for 'a' and 'b' from the given experimental data, we reformulate the problem as an optimization one and use 'Genetic algorithm' (GA) or 'Simulated annealing' (SA) to find the solution. Finding parameters is a simple case, much complex problems like finding the suitable growth condition for a bacterial fermentation culture to maximize product production involves optimization of multiple factors (carbon, nitrogen source, oxygen, temperature, pH etc.), and are routinely solved by optimization techniques. In the following discussions, we use the above example to understand the principles of GA and SA algorithms.

Genetic algorithm:

As the name suggests, the algorithm works analogously to how the microscopical organisms adapt/evolve with respect to its environment. If a colony of bacteria is grown in a small scale (50 mL) in the presence of Ampicillin, only few will develop the ability to degrade the antibiotic by synthesizing the enzyme 'Penicillinase'. This enzymatic resistance is attributed to its genotypic ability to mutate, cross over and reproduce successfully for several generations.

The same principle is adopted in Genetic algorithm based simulations. For above stated case, we are interested in finding optimum value for 'a' and 'b' that represents the data better. So we define 'a' and 'b' as two genes and its combination/concatenation as 'chromosome' or potential solution to the problem. At the outset, we generate a 'population' or pool (say 500) of such chromosomes with random values for 'a' and 'b'.

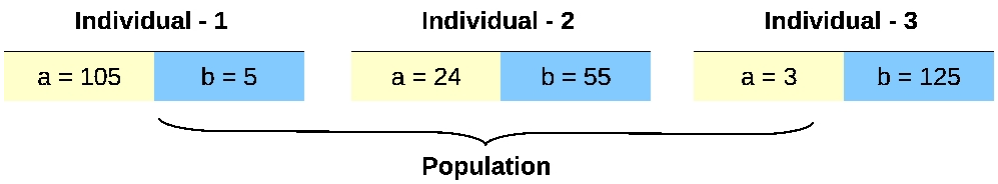


Figure : Three individuals with random values for 'a' and 'b'. Many such individuals constitute the population.

Subsequently, we allow crossing over of each chromosomes or individuals in the population by use of a random operator/ function.

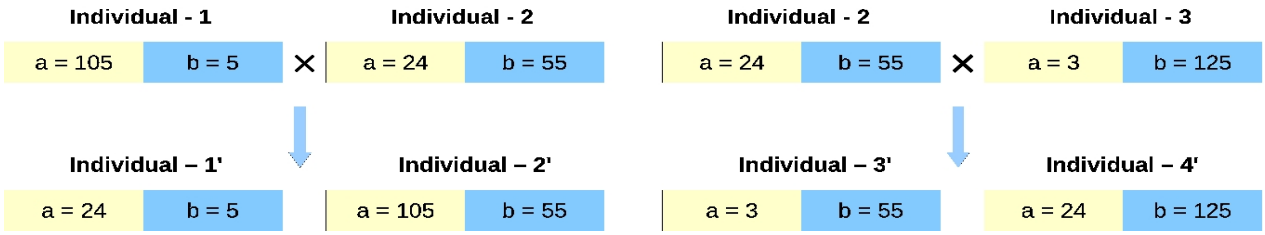


Figure: The individuals are crossed over among themselves, resulting in translocation of genes from one chromosome to another.

Following this, the existing gene values are mutated by another random operator/function.

Figure: Mutation of gene values for the above crossed over individuals

A model function is used to evaluate the fitness of each chromosome i.e, how good this individual is in representing the correct solution. So we use Michaelis-Menten model, values of a,b for each individual and evaluate the 'y' value and tabulate the results below,

x	$Y_{1''}$ a = 23 b = 4.5	$Y_{2''}$ a = 115 b = 50	$Y_{3''}$ a = 1.2 b = 59	$Y_{4''}$ a = 21 b = 130	Y_{expt}
0	0.00	0.00	0.00	0.00	**0.00**
3	9.20	6.51	0.06	0.47	**10.00**
5	12.11	10.45	0.09	0.78	**12.50**
10	15.86	19.17	0.17	1.50	**15.38**
15	17.69	26.54	0.24	2.17	**16.67**
30	20.00	43.13	0.40	3.94	**18.18**
70	21.61	67.08	0.65	7.35	**19.18**

We see that individual '1' has 'Y(calc)' closer to 'Y(expt)' values. In the following step, we define 'objective function' as the sum of the square of difference between 'Y(calc)' and 'Y(expt)' for each 'x' value; this value is also called as 'chi square' or 'sum of squared residuals'. The lesser the value for objective function, the better that individual is; such individuals are allowed to proceed to the next generation. Based on the fitness value (i.e objective function value), each individual is allowed to reproduce. More offsprings are generated for the chromosome with better fitness value. The offsprings mentioned here are the chromosomes that are exact replica of the parent chromosomes.

$$\text{objective function} = \left(1 - \frac{Y_{calc}}{Y_{expt}}\right)^2$$

Equation: Objective function:- The Y(calc) is calculated using Michaelis-Menten model and the Y(expt) is the observed experimental value.

This marks one generation, several such generations are gone through till a converged, global minimum value is achieved for the objective function. The following table explains how the objective function and parameters converges with progression in optimization.

Generation	(a)	(b)	Fitness value
1	347	145	30.2887
10	546	0	8486.9824
20	1	332	5.9590
100	75	10	16.7397
200	35	5.7	1.4660
450	19	2.98	0.0141
475	19.8	3.05	0.0012
500	19.9	3.01	0.0002

Table: The tabulated value corresponds to a single randomly picked individual among a population over different generations. As the number of generation increases, we see that the fitness value decreases and converge to a global minimum value. The 'a' and 'b' value obtained in the last generation, compares well with the model fitted values 20, 3 obtained through another method (Levenberg-Marquadt algorithm).

Simulated Annealing (SA):

SA is similar to GA, borrowing its principle from naturally occurring event called 'Annealing' seen in metallurgy. When the metals are heated to high temperature and cooled slowly, well ordered crystals without 'defect' would form, while rapid cooling, would result in defective and disordered amorphous form. This suggests, to achieve global minimum in the overall energy of a system, smaller continuous steps rather than large discrete steps are required for transitions. Similar approach is adapted in simulated annealing to achieve globally minimum solutions for a given problem. Most of the gradient

based optimization techniques have a serious problem in common,that is, if the problem is not well designed and starting values for the arbitrary parameters far away from the actual solution, they are prone to get stuck with locally minimum solutions rather than reaching the globally minimum solutions.

SA overcomes this problem, by using random operators, just like 'mutation and cross-over' operators used in GA. These operators are 'temperature' dependent and enables the optimization to jump over the 'local minima' to reach the 'global minimum'.

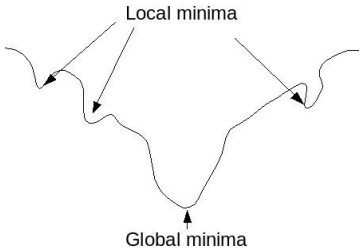


Figure: A continuous function with many local minima and one global minimum.

Using the same example used for GA, we will see the working mechanism of SA. In SA, we consider 'a' and 'b' as entities which defines the overall 'state' of the system. If 'a' and 'b' minimizes the 'energy' or the 'objective function' of the system, then it is bound to get to 'crystalline' state, else the undesired amorphous state with increased energy would result.

The algorithm starts with an initial higher temperature (~3000K) for a randomly generated state. The random values of 'a' and 'b' defines the state of the system at that temperature. The fitness or the objective function will be evaluated at this temperature for this state.

Following this, a random operator will vary the values of both 'a' and 'b' to generate a new state, the degree of variation is again dependent on current temperature. The energy of this new state is evaluated , if its 'energy' is lesser than the previous or 'old' state, the new state will be accepted. On the other hand, if the energy is higher, the new state may or may not be accepted based on the probability evaluated at this temperature.

$$\text{Probability} = e^{-\frac{(E_{i+1} - E_i)}{kT}}$$

Equation: Propability based on previous (i) and current (i+1) state at a given temperature. 'k' is the Boltzman constant.

This unique feature in SA, where higher energy states are accepted under certain probability, allows the states to escape from getting trapped in local wells. The probability is depended on the temperature of the state; at higher temperature, the higher energy states are accepted equally well as lower energy states, but as the temperature decreases, the acceptance probability of such states also decreases.

On acceptance of new state, the algorithm proceeds with the subsequent lower temperature. The complete process starting from fitness evaluation, generation of new state, fitness evaluation is repeated again for this new temperature. After several iterations, when the temperature reaches user defined minimum, the algorithm terminates.

On the other hand, if the new state is rejected based on the calculated probability, then new randomization is performed at the same temperature, this step will be go on for several times till a 'new state' meeting the acceptance criteria is found.

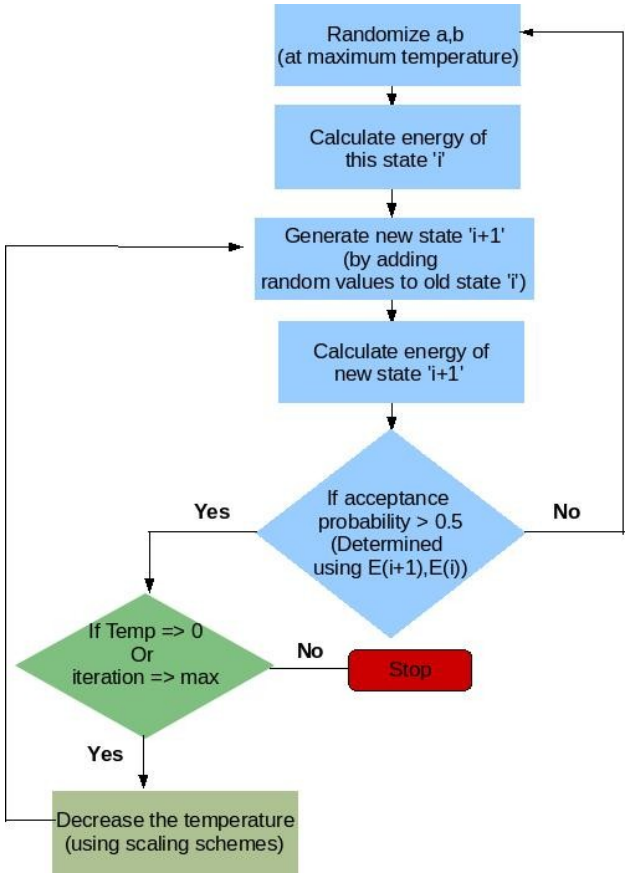
The gradual decrease in temperature may be carried out in a linear (1) or exponential (2) manner. For both the schemes, user has to define the initial (T_{max}) and final temperature (T_{min}), along with the number of step ($Cycle_{max}$) required to make this transition.

$$T_{next} = T_{current}/\mu$$

$$T_{next} = T_{current} + \frac{ln(\frac{T_{max}}{T_{min}})}{e^{(cycles_{max} - cycle_{current})}}$$

Equation: The linear and the exponential scaling schemes for decreasing the temperature

The following flow chart outlines the mechanism of SA .



Flow chart: A simple algorithm for Simulated annealing

References:

1) Genetic algorithm, http://www.iitk.ac.in/kangal/resources.shtml
2) Simulated annealing, http://www.gnu.org/software/gsl/manual/html_node/
3) Simulated annealing, http://www.gromacs.org/Documentation/