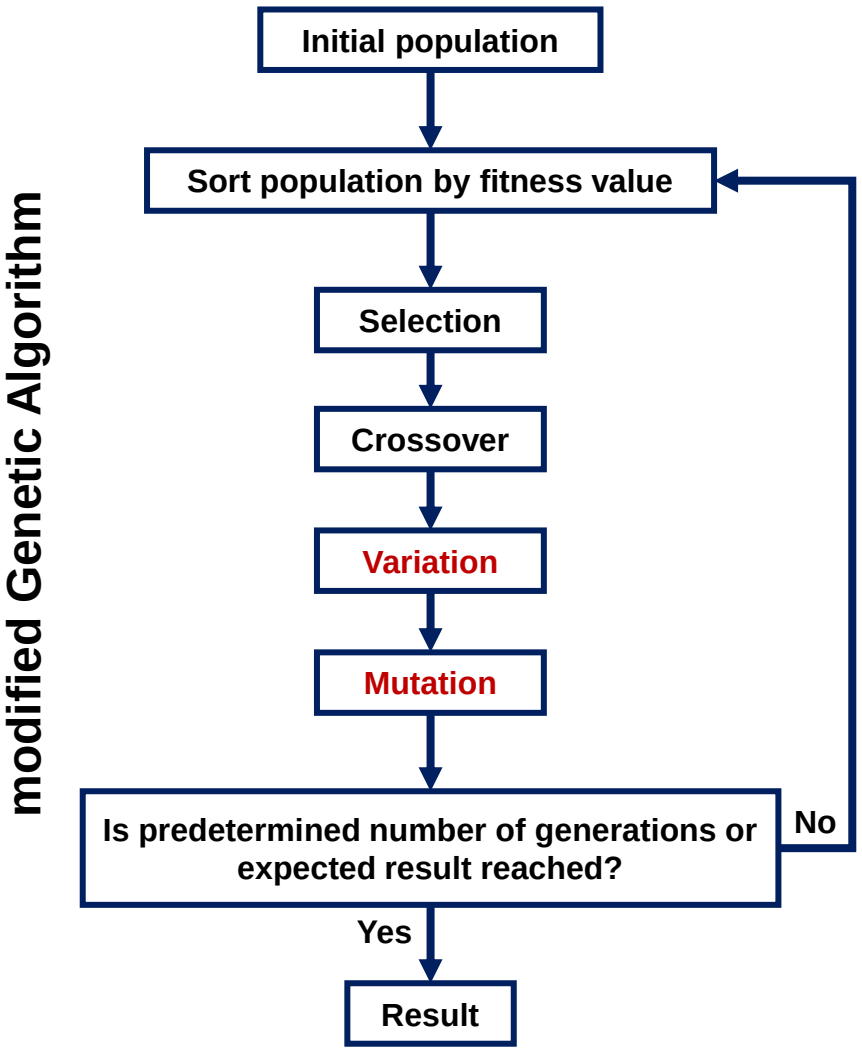
modifiedGA

Python Module for modified Genetic Algorithm

Ali Can Canbay

Ankara University Department of Physics

Modified Genetic Algorithm (mGA)



In order to address challenges related to multiple and correlated parameters, it is essential to maintain a large population and generation count in the Genetic Algorithm (GA). However, this can lead to longer computation times.

To tackle this issue, we introduced the concept of variation to the genetic algorithm, drawing inspiration from nature itself. We modified the mutation distribution to follow a Gaussian distribution, with the standard deviation adjusted based on the results of each generation.

What is variation?

The best genes undergo minor changes as they are passed on to the next generation

Modified Genetic Algorithm (mGA)

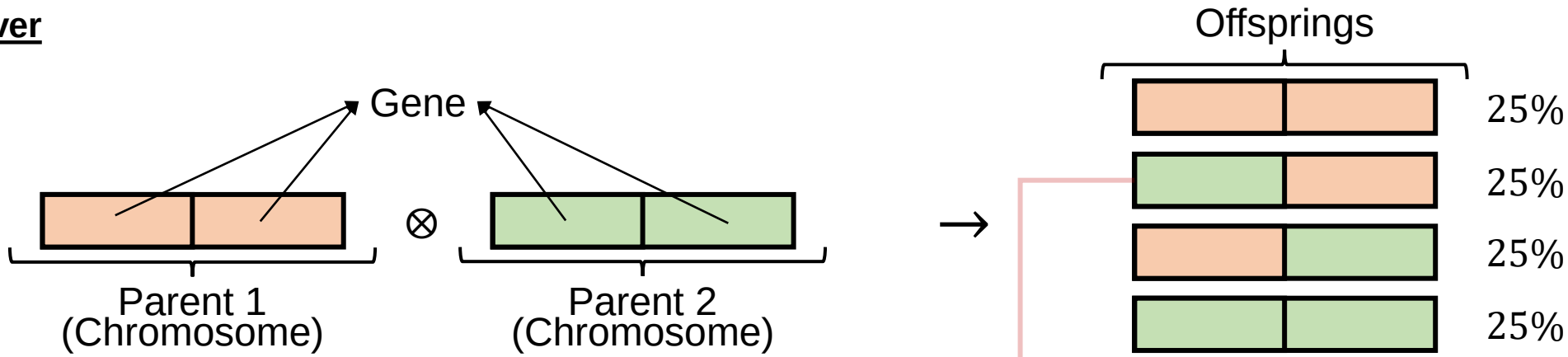
- **Initial population** consists of numbers determined to be within the domain of the variables.
- The initial population is tested whether the relevant function gives the optimized value and the **fitness** values (closeness to the result) are determined.
- The initial population is weighted according to their fitness values, and some of the highest weighted values are **selected** for the next generation.
- The parents of the offsprings which will form the next generation are selected by **weighted wheel (roulette wheel) selection** according to their fitness values.

weightless wheel

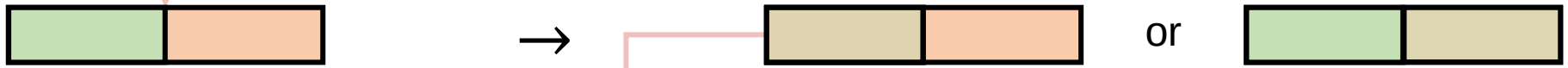
weighted wheel

Modified Genetic Algorithm (mGA)

Crossover



Variation



The variation is based on the genes of the parents.
The new gene is the average of the parent genes.

Mutation

In mutation, already existing genes are changed
within specified deviation probabilities.

Modified Genetic Algorithm (mGA)

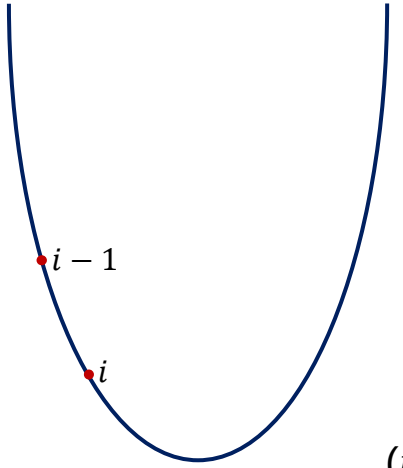
Standard Mutation:

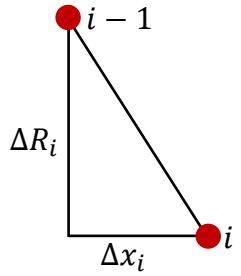
$$x_i = x_i \times random\boldsymbol{Uniform}(0.9, 1.1)$$

(i: Number of generations) These ratios may vary depending on usage

Modified Mutation:

$$x_i = x_i \times random\boldsymbol{Gauss}(\mu = 1, \sigma)$$





(i: Number of generations)

$$\Delta R_i = Result_i - Result_{i-1}$$

$$\Delta x_i = \sqrt{\sum_{k=1}^{N} \left(x_{k,i} - x_{k,i-1}\right)^2}$$

(N: Number of parameters)

$$\sigma_i = \frac{\Delta R_i}{\Delta x_i}$$

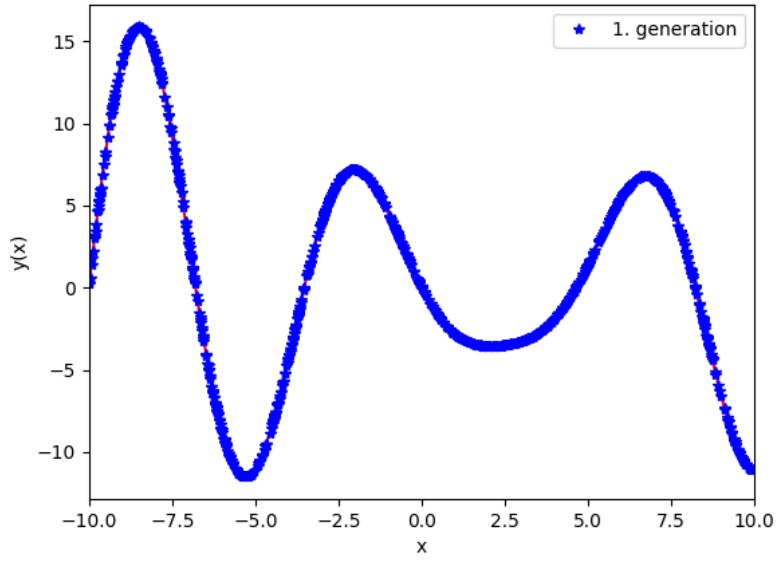
We want a fluctuation at most σ around the parameter for the mutation since variation is already applied.

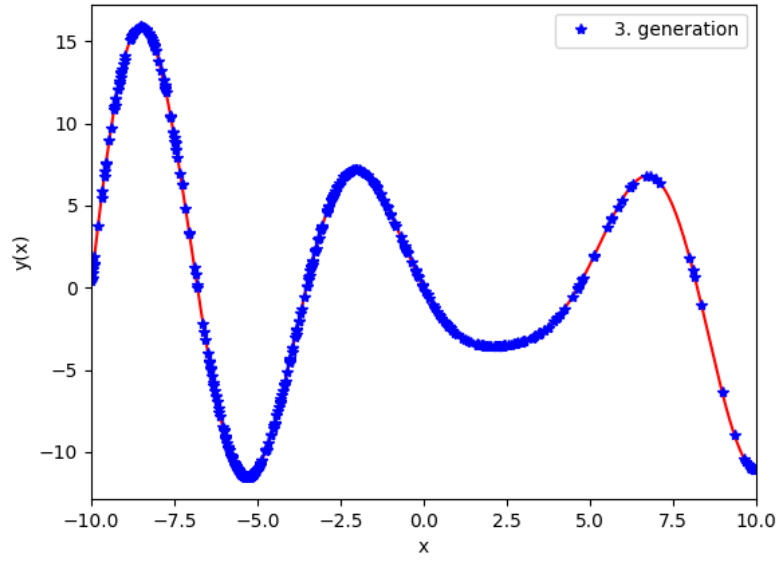
Modified Genetic Algorithm (mGA)

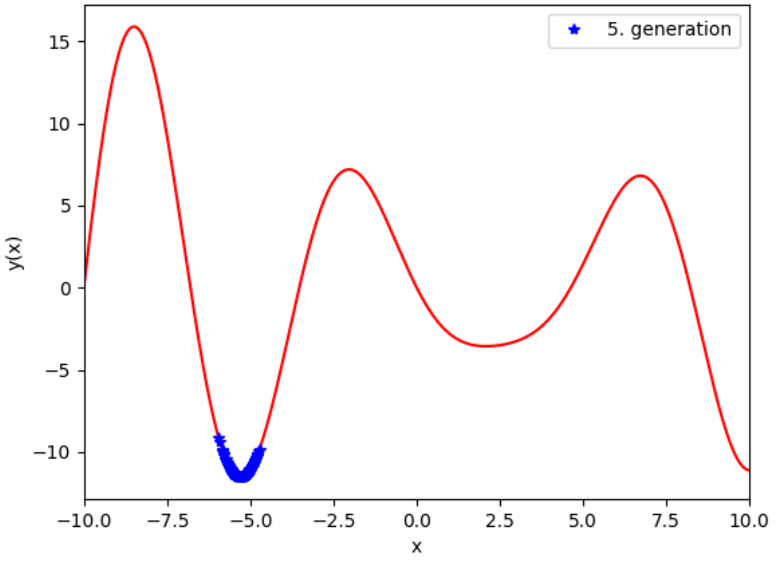
Minimize

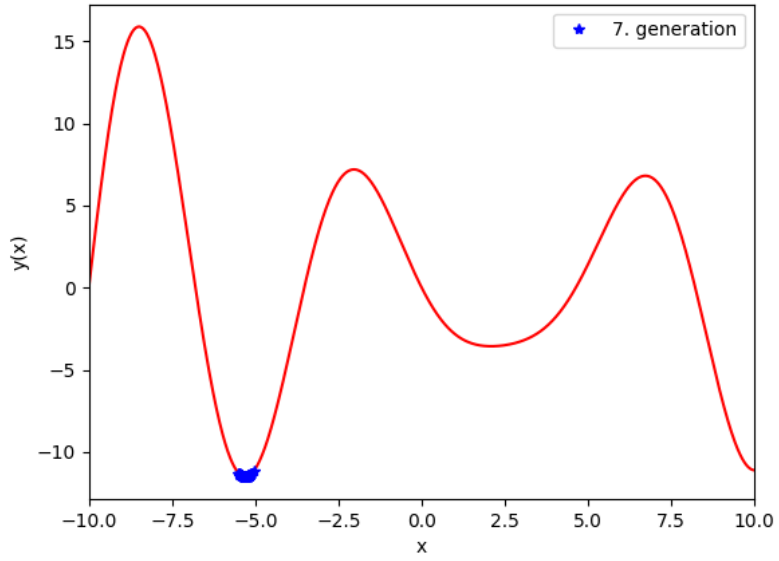
$f(x) = sin(x)(x-5) + cos(x)x$

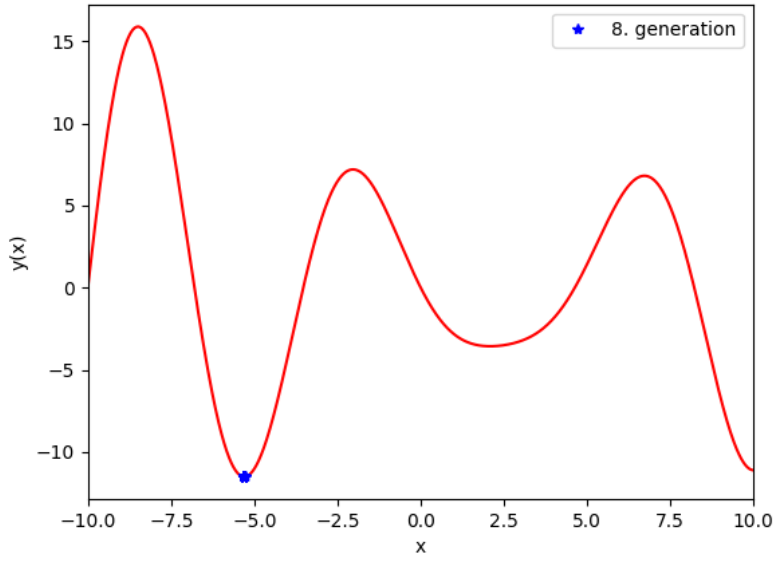
$x: [-10,10]$

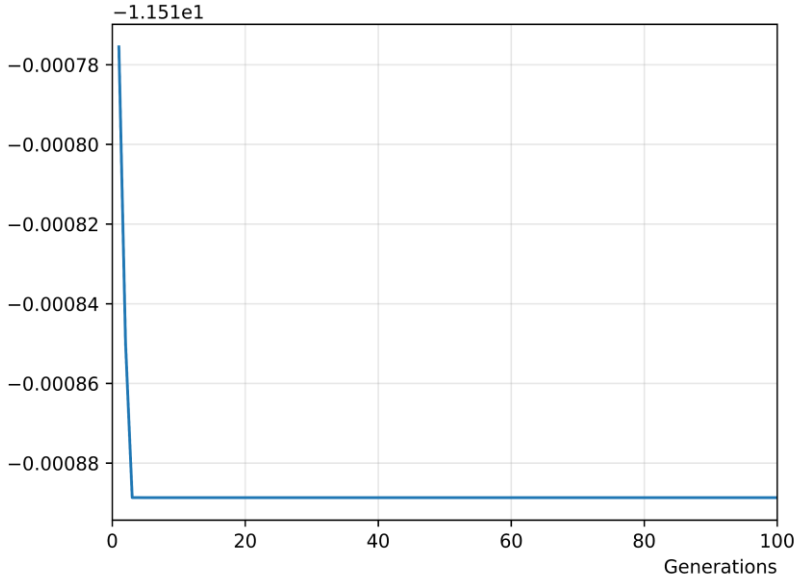












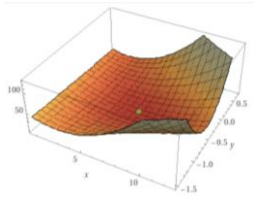
Modified Genetic Algorithm (mGA)

minimize $(x-5)^2+(y-3)^2+x^2y^2+x^2y$

Global minimum

$\min\{(x-5)^2+(y-3)^2+x^2y^2+x^2y\} \approx 3.64497208987637341623632593 7$
at $(x, y) \approx (6.612677980974700535797561234, -0.421748382735042090930087 1604)$

WolframAlpha computational intelligence.



pymoo: Multi-objective Optimization in Python [3]

pymoo is a multi-objective optimization module for python that contains many algorithms.

In order to test the algorithms under equal conditions, the following definitions were made on pymoo:

- algorithm = "ga" (Genetic Alrgorithm)
- N_offsprings = None
- get_sampling("real_random")
- crossover = get_crossover("real_sbx")
- mutation = get_mutation("real_pm")
- eliminate_duplicates = False

Population Size : 100
Number of Generations : 100
Distribution Size : 1000

		Domain		
		[-10,10]	**[-100,100]**	**[-1000,1000]**
pymoo (0.4.0)	**σ (around the correct result)**	2.492×10^{-5}	3.091×10^{-3}	1.823×10^{0}
	Δt [m:s]	10:17	09:39	09:51

		Domain		
		[-10,10]	**[-100,100]**	**[-1000,1000]**
modified	**σ (around the correct result)**	5.108×10^{-15}	5.225×10^{-15}	5.159×10^{-15}
	Δt [m:s]	01:02	01:05	01:05

This time difference is due to the time spent on importing modules, random number generations and evaluation functions. This time difference is expected to be approximately the same for different problems (with not correlated variables).

The results in both algorithms may vary according to eta and probability values. Therefore, it is possible to improve both algorithms. Here the default values are compared.