Benchmark Function Optimization: A Comparative Analysis of Genetic Algorithms against Hill Climbing and Simulated Annealing Strategies

Buzdea Stefan
Pricop Tudor-Andrei

November 15, 2023

1 Abstract

This paper explores the efficacy of genetic algorithms in minimizing benchmark functions, comparing their performance with Simulated Annealing and Hill Climbing. We investigate variants of the latter, including the first improvement, best improvement, and worst improvement strategies. Focusing on global minimum optimization across 5, 10, and 30 dimensions, we analyze key statistics such as average and best values, tracking how function minima evolve across generations.

Our findings reveal that genetic algorithms efficiently explore expansive solution spaces within reasonable timeframes, outpacing Hill Climbing and Simulated Annealing. Notably, in higher dimensions, genetic algorithms demonstrate superior accuracy.

In conclusion, this study underscores the advantages of genetic algorithms for optimization challenges, providing a valuable perspective for heuristic algorithm selection.

2 Introduction

This study aims to compare Genetic Algorithms with Simulated Annealing and Hill Climbing across diverse scenarios. Genetic algorithms emulate natural selection, where the fittest individuals are chosen for reproduction, producing offspring in subsequent generations. Leveraging biologically inspired operators such as mutation, crossover, and selection, genetic algorithms excel in optimizing and solving search problems. Natural selection involves selecting the fittest individuals from a population, whose descendants inherit parental traits, perpetuating through new generations. The five-step genetic algorithm process includes initializing a population, defining a fitness function, selection, crossover, and mutation.

Our evaluation encompasses four benchmark functions: De Jong 1, Schwefel's, Rastrigin's, and Michalewicz's, across 5, 10, and 30 dimensions, implemented in Python. Rigorous experimentation will yield minimum values across runs, average minimum values, and execution times for each method and variant. Observing how function minima evolve over generations, we aim to discern significant differences, providing nuanced insights.

This research not only seeks to identify the most effective method for global minima discovery but also explores practical implications. In conclusion, we will succinctly summarize key insights, elucidating their relevance in the realm of optimization problems.

3 Methods

In the genetic algorithm implementation, the entire population is represented as a vector of bit vectors generated randomly. We determined the required number of bits to represent each interval, considering the function's dimensionality, and maintained a precision of 10^5, consistent with the approach used in the previously discussed methods. After generating a population of size *pop_size* chromosomes and calculating the function values at those points, the population evolution process begins over a set

number of generations. During each generation, a population of chromosomes is selected, followed by the application of the mutation and crossover operators. Holland's roulette wheel selection is employed for selection, allowing individuals to be chosen in the next population generation with a probability proportional to their fitness divided by the population's total fitness.

The fitness function, designed to measure chromosomal quality and ensure that superior candidates have a higher probability of selection, is computed using the formula:

$$f(x) = \left(\frac{e_{\min} - e(x)}{e_{\max} - e_{\min} + \epsilon} + 1 \right)^{\text{selection pressure}}$$

Here, ϵ is set to 10^{-6} to minimize its impact on the fraction.

For the mutation process, a random number between $[0, 1]$ is chosen for each bit of each individual. If it is less than $1/L$ (where L is the number of bits in a chromosome), the bit is mutated, resulting in a flip. In the crossover function, each individual is assigned a random number in $[0, 1]$. Individuals are then ordered based on these numbers, and pairs are formed only if their assigned number is less than 0.8, indicating an 80% chance of generating new descendants. Two randomly selected cutting points are employed for the crossover process. A noteworthy optimization involves a greedy method, where, from the two parents and resulting two descendants, only the best two are chosen. This adjustment, prompted by suboptimal outcomes in strategies where descendants replaced parents or all four individuals were retained, ensures a more effective genetic algorithm by prioritizing the most promising individuals. At the conclusion of all iterations through the generations, the best chromosome from the last generation is selected as the solution.

This methodology balances computational efficiency and optimization effectiveness within the genetic algorithm framework.

4 Functions

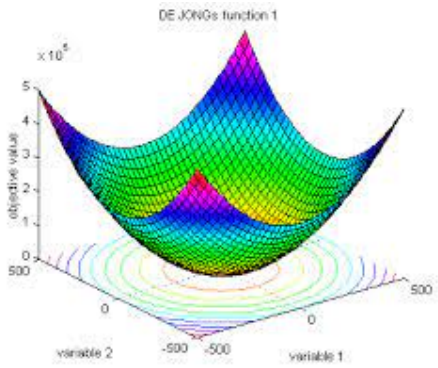


Figure 1: De Jong's function: $f_n(x) = \sum_{i=1}^{n} ix_i^2$
$-5.12 \leq x_i \leq -5.12$
global minimum: $f(x) = 0$

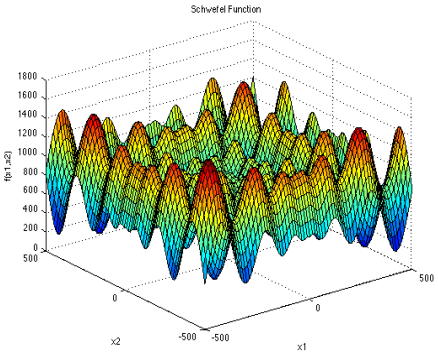


Figure 2: Schwefel's function: $f_n(x) = \sum_{i=1}^{n} -x_i sin(\sqrt{|x_i|})$
$-500 \leq x_i \leq 500$
global minimum: $f_n(x) = -n * 418.9829$

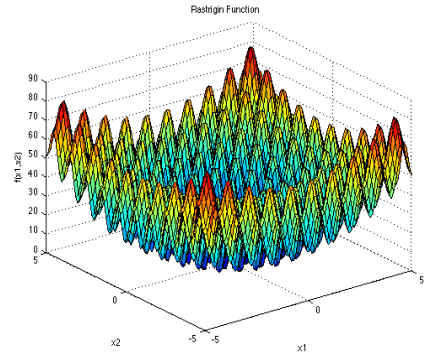


Figure 3: Rastrigin's function: $f_n(x) = 10n + \sum_{i=1}^{n}(x_i^2 - 10cos(2\pi x_i))$
$-5.12 \leq x_i \leq -5.12$
global minimum: $f(x) = 0$

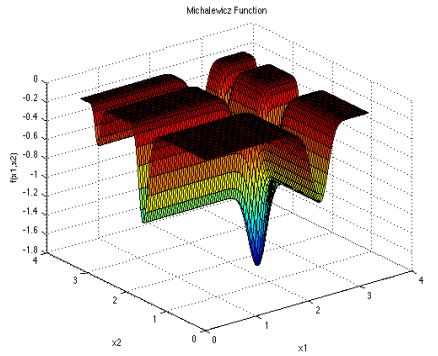
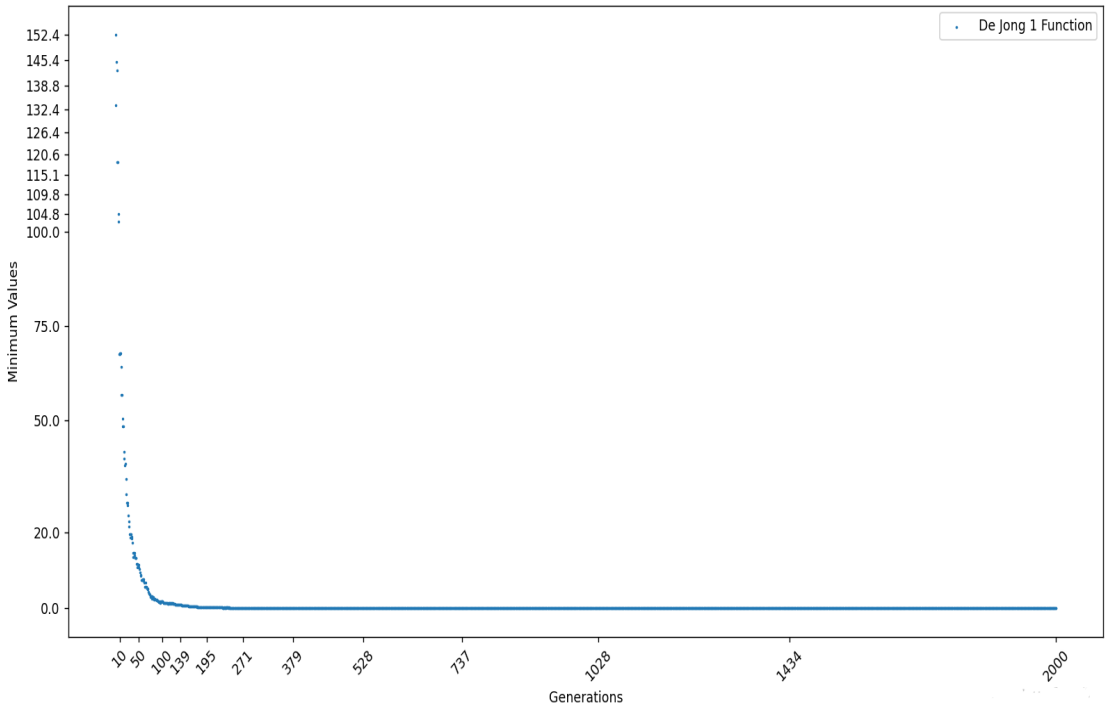
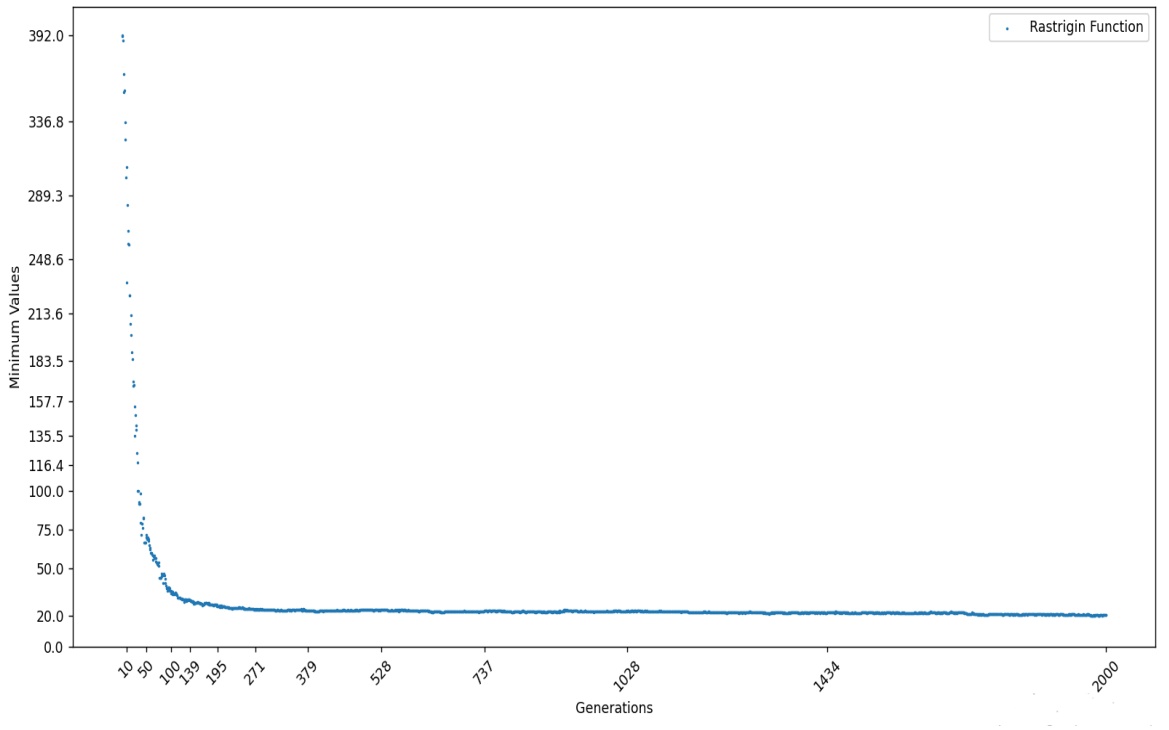


Figure 4: Michalewicz's function: $f_n(x) = -\sum_{i=1}^{n} sin(x_i)(sin\frac{ix_i^2}{\pi})^{20}$
$0 \leq x_i \leq \pi$
global minimum: $f_5(x) = -4.687, f_{10}(x) = -9.66$

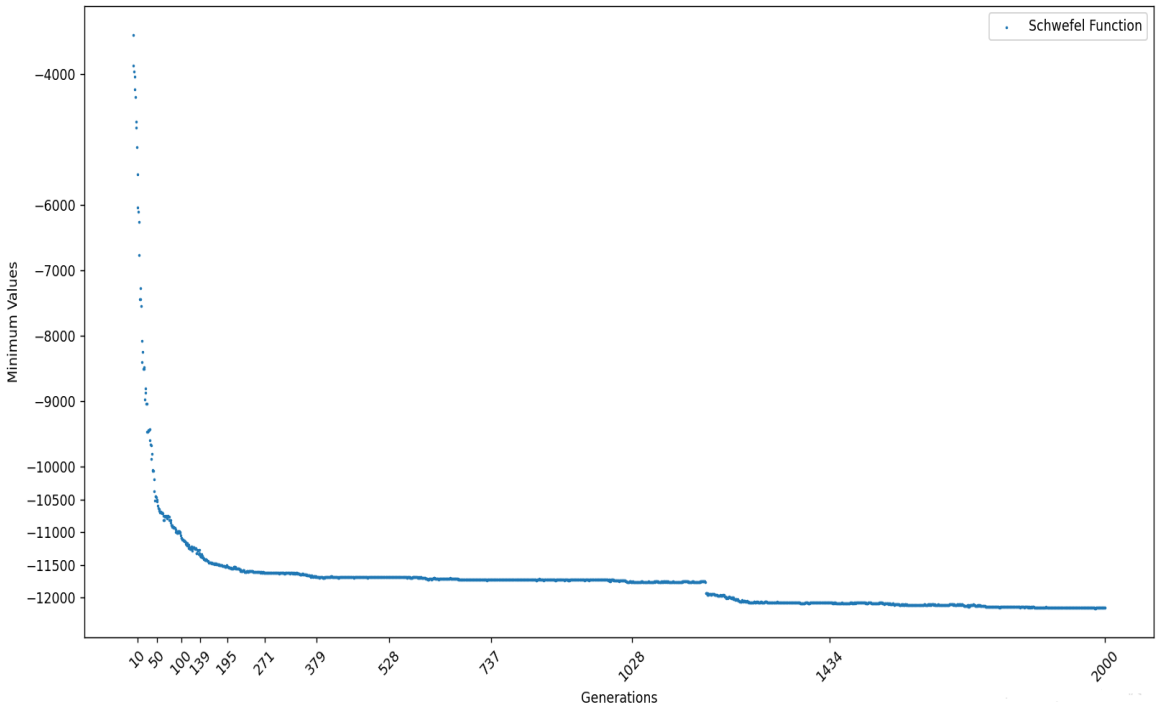
5 Experimental results



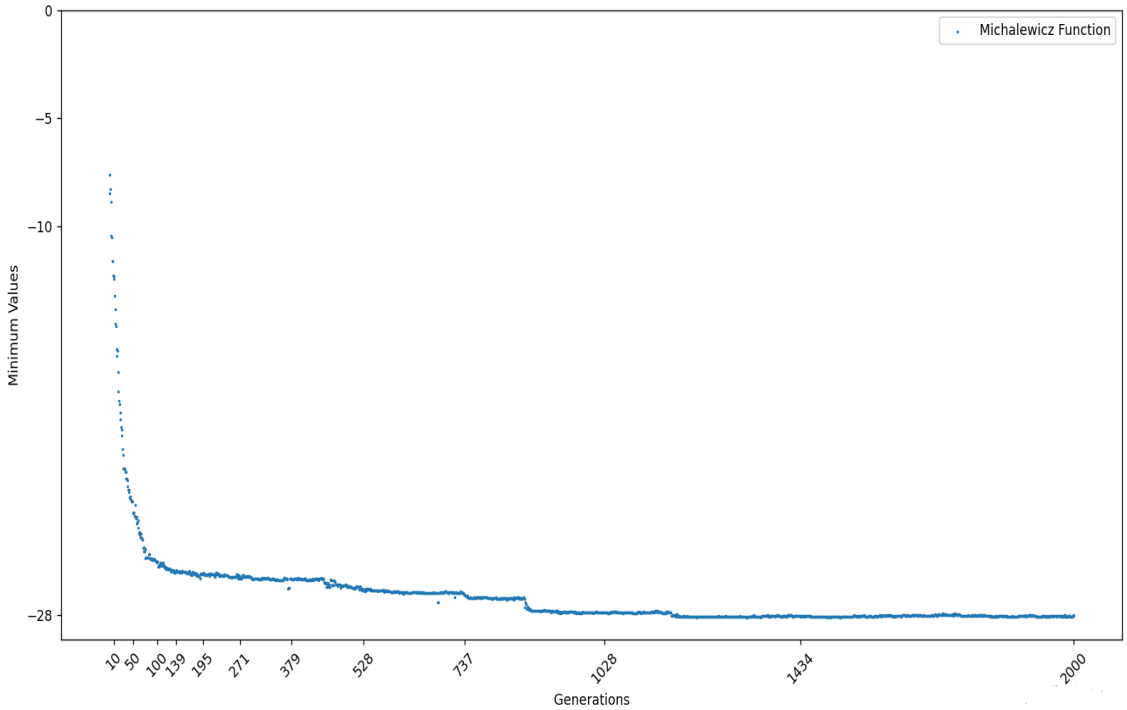
Genetic Algorithm Evolution on De Jong 1 function (30 dimensions)



Genetic Algorithm Evolution on Rastrigin function (30 dimensions)



Genetic Algorithm Evolution on Schwefel function (30 dimensions)



Genetic Algorithm Evolution on Michalewicz function (30 dimensions)

DeJong				
Dimension	Algorithm	Minim	Average	Time
30	GA	0.0024	0.0069	184 s
	HCBI	0.0	0.0	805 s
	HCFI	0.0	0.0	100 s
	SA	0.0	0.0	999 s
10	GA	0.0	0.0	121 s
	HCBI	0.0	0.0	421 s
	HCFI	0.0	0.0	79 s
	SA	0.0	0.0	744 s
5	GA	0.0	0.0	80 s
	HCBI	0.0	0.0	297 s
	HCFI	0.0	0.0	55 s
	SA	0.0	0.0	1072 s

Rastrigin				
Dimension	Algorithm	Minim	Average	Time
30	GA	15.836	19.9776	195 s
	HCBI	20.62561	27.04519	661 s
	HCFI	30.07828	38.73196	86 s
	SA	17.86534	26.27192	1297 s
10	GA	0.0	1.18684	128 s
	HCBI	2.98923	4.46861	303 s
	HCFI	4.22577	5.95132	75 s
	SA	2.04951	4.12050	1065 s
5	GA	0.0	0.04119	90 s
	HCBI	0.0	0.0	237 s
	HCFI	0.0	0.99536	64 s
	SA	0.0	0.0	1309 s

Schwefel				
Dimension	Algorithm	Minim	Average	Time
30	GA	-12533.25476	-12056.17361	179 s
	HCBI	-10865.19414	-10623.35167	2081 s
	HCFI	-10543.07683	-10127.72135	397 s
	SA	-11114.98883	-10922.64678	2400 s
10	GA	-4189.72557	-4029.09442	114 s
	HCBI	-4023.95232	-3872.213	919 s
	HCFI	-3998.89712	-3811.251	183 s
	SA	-4109.41635	-4002.11006	1670 s
5	GA	-2094.91	-2066.58240	94 s
	HCBI	-2094.91345	-2074.639	710 s
	HCFI	-2094.80931	-2063.229	155 s
	SA	-2094.91346	-2094.872	2242 s

Michalewicz				
Dimension	Algorithm	Minim	Average	Time
30	GA	-28.5601	-28.1894	187 s
	HCBI	-26.74368	-25.62685	541 s
	HCFI	-25.91867	-25.02531	76 s
	SA	-27.45123	-27.10534	1003 s
10	GA	-9.6535	-9.6247	133 s
	HCBI	-9.55213	-9.34905	236 s
	HCFI	-9.32472	-9.10362	55 s
	SA	-9.61076	-9.38035	881 s
5	GA	-4.68765	-4.6798	76 s
	HCBI	-4.68765	-4.68672	194 s
	HCFI	-4.68704	-4.67891	48 s
	SA	-4.68765	-4.68673	1150 s

6 Experimental Description

In our experimental configuration, we conducted a thorough analysis utilizing the Genetic Algorithm to pinpoint global minima across functions with varying dimensions. Our primary focus aimed at achieving precision in larger dimensions, ensuring results accurate to at least 5 decimal places. To ensure the robustness of our findings, we executed 30 runs for each case (dimensions 5, 10, and 30), calculating key statistics (minimum, average value, and average time). Across all runs, the population size (*pop_size*) and the number of generations were consistently set to 200 and 2000, respectively. Additionally, the crossover probability was maintained at 0.7, and the mutation probability was set to $1/L$ (where L represents the number of bits in a chromosome). The accompanying graphs depict the convergence of each 30-dimensional function using the genetic algorithm, illustrating the progressive improvement of the best value at each generation over time.

7 Comparison and Interpretation

In examining the results presented above, the Genetic Algorithm demonstrates exceptional performance, particularly excelling in 30 dimensions, surpassing even the values obtained by Simulated Annealing while exhibiting significantly improved execution times. However, certain scenarios, such as the De Jong 1 function with 30 dimensions, reveal instances where the Genetic Algorithm does not yield the optimal value compared to the Best Improvement Hill Climb and Simulated Annealing methods. This outcome is anticipated, considering the GA's inherent exploration of a vast solution space, designed for broad optimization rather than meticulous analysis of incremental fitness changes, unless employing advanced optimization techniques. A notable feature of the Genetic Algorithm is its capacity to foster diversity within the population, mitigating convergence to local minima. However, it is essential to acknowledge the challenge of high parameter tuning for optimal results. Compared to Simulated Annealing and Hill Climb methods, the Genetic Algorithm may exhibit a tendency to explore a limited region and occasionally become ensnared in a local minimum. This nuanced comparison provides valuable insights into the strengths and limitations of each optimization method.

8 Conclusion

In summary, Genetic Algorithms showcase robust performance in global optimization, particularly evident in higher dimensions. Despite occasional challenges in specific scenarios, their ability to explore diverse solution spaces, mitigate local minima, and offer efficient convergence underscores their significance in tackling complex optimization problems. Parameter tuning remains a consideration, but the nuanced comparison with alternative methods provides valuable insights into the broad applicability and effectiveness of Genetic Algorithms in optimization tasks.

References

[1] M.J. Usman *Correlation Study of Genetic Algorithm Operators: Crossover and Mutation Probabilities*. International Symposium on Mathematical Sciences and Computing Research, 2013.

[2] V.P. Patil, D.D. Pawar *The Optimal Crossover or Mutation Rates in Genetic Algorithm: A Review*. International Journal of Applied Engineering and Technology ISSN: 2277-212X (Online), 2015.

[3] D. Bingham, S. Surjanovic *Test Functions and Datasets: Schwefel Function*. Simon Fraser University : Virtual Library of Simulation Experiments, 2013.

[4] Bingham, D. Surjanovic, S. *Test Functions and Datasets: Rastrigin Function*. Simon Fraser University : Virtual Library of Simulation Experiments, 2013.

[5] Bingham, D. Surjanovic, S. *Test Functions and Datasets: Michalewicz Function*. Simon Fraser University : Virtual Library of Simulation Experiments, 2013.

[6] Croitoru, E. *Teaching: Genetic Algorithms*. Romania [Iasi] : Computer Science UAIC.

[7] *Simulated Annealing Explained.* Baeldung.com, 2023.

[8] Rawat, U. *Introduction to Hill Climbing — Artificial Intelligence.* geeksforgeeks.org, 2023.

[9] R.N. Greenwell, J.E. Angus, M. Finck *Optimal mutation probability for genetic algorithms.* sciencedirect.com, 1995.

[10] S. Sarmady *An Investigation on Genetic Algorithm Parameters.* sarmady.com, 2023.