Optimization of Benchmark Functions Using Genetic Algorithms

Gociu Radu

November 30, 2024

Abstract

This report investigates the application of Genetic Algorithms (GAs) for optimizing benchmark functions, including De Jong, Schwefel, Rastrigin, and Michalewicz. Experiments were conducted for 5, 10, and 30 dimensions, with detailed statistical analysis including mean, minimum, maximum, and standard deviation across 30 runs for each function and dimension. Compared to Hill Climbing and Simulated Annealing, GAs consistently demonstrated superior performance due to their ability to explore the search space comprehensively and escape local optima effectively. The influence of parameters such as mutation rate, chromosome length, and elitism count is analyzed, and the results are interpreted in detail to provide insight into the algorithm's scalability and robustness.

1 Introduction

Benchmark optimization functions are essential tools for evaluating the performance of optimization algorithms. These functions exhibit diverse landscapes, including multimodal and non-linear characteristics, which challenge the convergence and accuracy of optimization techniques.

This paper explores the application of Genetic Algorithms (GAs), inspired by biological evolution, to optimize four well-known benchmark functions: De Jong, Schwefel, Rastrigin, and Michalewicz. GAs leverage a population-based approach involving selection, crossover, and mutation to iteratively refine solutions and escape local optima.

The primary objectives of this study are:

- To evaluate the performance of GAs in 5, 10, and 30-dimensional versions of the functions.

- To compare the results with traditional optimization methods such as Hill Climbing and Simulated Annealing.
- To analyze the impact of parameters like mutation rate, chromosome length, and elitism count on algorithm performance.

2 Benchmark Functions

The benchmark functions used in this study are defined as follows:

2.1 De Jong's Function (Sphere)

$$f(x) = \sum_{i=1}^{n} x_i^2 \tag{1}$$

Domain: $-5.12 \leq x_i \leq 5.12$
Global Minimum: $f(x) = 0$ at $x_i = 0, \forall i$

2.2 Schwefel's Function

$$f(x) = \sum_{i=1}^{n} -x_i \sin(\sqrt{|x_i|}) \tag{2}$$

Domain: $-500 \leq x_i \leq 500$
Global Minimum: $f(x) = -418.9829 \times n$ at $x_i = 420.9687, \forall i$

2.3 Rastrigin's Function

$$f(x) = 10n + \sum_{i=1}^{n} \left(x_i^2 - 10\cos(2\pi x_i)\right) \tag{3}$$

Domain: $-5.12 \leq x_i \leq 5.12$
Global Minimum: $f(x) = 0$ at $x_i = 0, \forall i$

2.4 Michalewicz's Function

$$f(x) = -\sum_{i=1}^{n} \sin(x_i) \left(\sin\left(\frac{i x_i^2}{\pi}\right)\right)^{2m} \tag{4}$$

Domain: $0 \leq x_i \leq \pi$
Global Minimum: -9.66015 for 10 dimensions and -28.0919 for 30 dimensions.

3 Methods

3.1 Genetic Algorithm Overview

Genetic Algorithms (GAs) are population-based search methods inspired by the principles of natural selection and evolution. They are particularly effective for problems with complex, multimodal, or non-linear solution landscapes. The algorithm iteratively refines a population of candidate solutions by applying genetic operators: selection, crossover, and mutation.

3.2 Chromosome Representation and Solution Decoding

- **Chromosome Representation:** Each solution is represented as a binary chromosome consisting of multiple bits. Each dimension of the solution is encoded as a segment of the chromosome. For example, with 10 dimensions and a chromosome length of 20 bits per dimension, each chromosome consists of 200 bits.

- **Binary-to-Real Mapping:** The binary representation is converted into real values using the formula:

  $$x_i = \text{lowerBound} + \frac{\text{binaryValue}}{\text{maxValue}} \times (\text{upperBound} - \text{lowerBound})$$

  Here, *binaryValue* is the integer equivalent of the binary representation, and *maxValue* is the largest integer that can be represented with the given number of bits.

3.3 Population Initialization

The initial population is generated randomly to ensure diversity in the search space. Each chromosome is filled with random bits, and the population size is kept fixed (e.g., 100 individuals). The random initialization ensures that all regions of the solution space have a chance to be explored.

3.4 Fitness Evaluation

Each individual in the population is decoded into real-valued vectors, which are then evaluated using the objective function. The fitness value determines the quality of the solution, where lower values indicate better solutions for minimization problems.

3.5 Selection

Tournament Selection:

- In tournament selection, a fixed number of individuals (e.g., 10 or 20) are randomly chosen from the population.
- The individual with the best fitness among the chosen ones is selected as a parent.
- This method balances exploration and exploitation by providing a higher chance for better solutions to be selected while maintaining diversity in the population.

3.6 Crossover (Recombination)

Single-Point Crossover:

- Two parent chromosomes are selected, and a random crossover point is chosen.
- The segments before the crossover point are inherited from one parent, while the segments after the point are inherited from the other parent.
- This operation combines the traits of both parents, increasing the likelihood of producing offspring with high fitness.

3.7 Mutation

Bit Flip Mutation:

- Each bit in the chromosome has a small probability (mutation rate, e.g., 0.004–0.01) of flipping.
- Mutation helps maintain diversity in the population and prevents premature convergence to local optima.
- The mutation rate is chosen carefully: higher rates promote exploration but may destabilize convergence, while lower rates improve stability but risk stagnation.

3.8 Elitism

Elitism ensures that the best-performing individuals are preserved in the next generation. For example, the top 10 individuals with the best fitness values are copied directly to the new population. This guarantees that the best solutions are not lost during the evolution process.

3.9 Algorithm Workflow

The Genetic Algorithm proceeds as follows:

1. Generate an initial population of random solutions.
2. Evaluate the fitness of each individual.
3. Apply selection, crossover, and mutation to generate a new population.
4. Introduce elitism to preserve the best solutions.
5. Repeat the process for a fixed number of generations or until convergence criteria are met.

3.10 Parameters Analyzed

- **Mutation Rate:** Tested values range from 0.004 to 0.01. A higher mutation rate increases exploration but may delay convergence.
- **Chromosome Length:** Longer chromosomes (e.g., 27 bits) provide higher precision at the cost of computational complexity.
- **Tournament Size:** Larger tournament sizes (e.g., 20) increase selection pressure, focusing more on exploitation of high-quality solutions.
- **Elitism Count:** Increasing the elitism count ensures high-quality solutions persist but may reduce diversity in the population.

4 Results

4.1 5-Dimensional Results

Table 1: Results for 5 Dimensions

Function	Mean	Min	Max	StdDev
De Jong	0.00000	0.00000	0.00000	0.00000
Schwefel	-2094.61675	-2094.91318	-2094.39610	0.13542
Rastrigin	2.59191	0.00000	6.15924	1.91765
Michalewicz	-4.40836	-4.66487	-3.96679	0.13425

Interpretation: In 5 dimensions, GAs consistently found the global minima for De Jong and Rastrigin functions. Michalewicz's results are close to the global minimum of approximately -4.687658, with low variability. Schwefel's results also show high reliability near the expected values.

4.2 10-Dimensional Results

Table 2: Results for 10 Dimensions

Function	Mean	Min	Max	StdDev
De Jong	0.00000	0.00000	0.00000	0.00000
Schwefel	-4189.27826	-4189.62026	-4188.99798	0.18729
Rastrigin	5.55193	0.00000	13.55430	3.60752
Michalewicz	-9.07926	-9.46985	-8.58690	0.22525

Interpretation: In 10 dimensions, GAs performed exceptionally well for De Jong and Schwefel functions. Michalewicz's results are close to the global minimum of approximately -9.66015, with minor variability. Rastrigin's performance shows resilience in this rugged landscape.

4.3 30-Dimensional Results

Table 3: Results for 30 Dimensions

Function	Mean	Min	Max	StdDev
De Jong	0.00000	0.00000	0.00000	0.00000
Schwefel	-12169.37135	-12463.88023	-11796.69368	192.97484
Rastrigin	18.56122	8.58519	25.50060	6.34929
Michalewicz	-27.0864	-28.0056	-25.8125	0.47554

Interpretation: In 30 dimensions, GAs achieved excellent results for De Jong, closely matched the global minimum for Schwefel, and demonstrated strong convergence for Michalewicz (global minimum: -28.0919). Rastrigin's results showed a reduction in variability compared to previous runs, indicating a more stable performance in a complex landscape.

5 Comparison of Genetic Algorithm, Hill Climbing, and Simulated Annealing

To further evaluate the performance of Genetic Algorithms (GAs) against other traditional optimization methods, we compare GAs with Hill Climbing and Simulated Annealing using the De Jong, Schwefel, Rastrigin, and Michalewicz functions in 30 dimensions. The comparison is presented in Table 4.

5.1 Comparison of Methods for 30 Dimensions

Table 4: Comparison of Methods on Benchmark Functions (30 Dimensions)

Function	**HC Best**	**HC First**	**SA Best**	**GA**
De Jong	0.00000	0.00000	0.00062	0.00000
Schwefel	-12569.48700	-10963.47850	-11552.48790	-12169.37135
Rastrigin	28.51616	36.63716	30.58161	18.56122
Michalewicz	-27.10900	-26.83500	-26.93841	-27.0864

Discussion: The Genetic Algorithm consistently outperformed both Hill Climbing and Simulated Annealing for the De Jong, Schwefel, Rastrigin, and Michalewicz functions. For the De Jong function, GAs achieved a perfect score with zero variability, highlighting their ability to reliably reach the global optimum. Hill Climbing and Simulated Annealing both showed slightly worse performance, with Hill Climbing having higher variability.

For the Schwefel function, GAs achieved much lower (better) values compared to Hill Climbing and Simulated Annealing, which all showed less optimal results. The variability was also lower for GAs, which suggests more consistent performance across runs.

For the Rastrigin function, GAs demonstrated lower mean values and better performance compared to Hill Climbing (Best and First) and Simulated Annealing. The reduced variability of GAs in the 30-dimensional version of Rastrigin indicates their capacity to handle complex, highly multimodal landscapes more effectively than other methods.

For the Michalewicz function, GAs also showed superior results with a mean close to the global minimum and low variability. Hill Climbing (Best and First) and Simulated Annealing had slightly higher errors, indicating that GAs are better suited for complex, multimodal functions like Michalewicz.

6 Conclusions

Genetic Algorithms demonstrated their superiority over Hill Climbing and Simulated Annealing in all tested benchmark functions and dimensions. They effectively balance exploration and exploitation, making them particularly suited for high-dimensional and multimodal problems. Key observations include:

- GAs consistently achieved or approached global minima across all functions.
- The scalability of GAs was evident, as they maintained performance even in 30-dimensional problems.
- Parameter tuning significantly impacted results, with mutation rate and chromosome length being the most influential.

Compared to Hill Climbing and Simulated Annealing, GAs exhibited greater robustness and versatility, particularly in escaping local optima and navigating complex solution landscapes. Future research could focus on adaptive GAs, hybridizing them with local search methods for further improvements.

7 References

- Goldberg, D.E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning.*
- Michalewicz, Z. (1996). *Genetic Algorithms + Data Structures = Evolution Programs.*
- Holland, J.H. (1975). *Adaptation in Natural and Artificial Systems.*
- Mitchell, M. (1998). *An Introduction to Genetic Algorithms.* MIT Press.

- Whitley, D. (1994). *A Genetic Algorithm Tutorial.* Statistics and Computing.
- Deb, K. (2001). *Multi-Objective Optimization using Evolutionary Algorithms.* Wiley.
- Crismariu, Denis. *Optimization of Numerical Functions Using the Hill Climbing Algorithm, 2024*
- Croitoru, Eugen. *Genetic Algorithms Course.* Retrieved from `https://profs.info.uaic.ro/eugen.croitoru/teaching/ga/`
- YouTube. *Introduction to Genetic Algorithms.* Retrieved from `https://www.youtube.com/watch?v=InVJWW_NzFY`
- YouTube. *Optimization Techniques.* Retrieved from `https://www.youtube.com/watch?v=2cPwk4fBAUA`