Analysis of Global Optimization Methods: Genetic Algorithms vs. Hill Climbing and Simulated Annealing

Antoniev Valeriu-Gabriel, Maciuc Mihai

December 1, 2024

Abstract

This report contains the implementation and evaluation of a genetic algorithm for solving optimization problems. The algorithm is applied to the following functions: Rastrigin, Schwefel, DeJong and Michalewicz. Key details of the algorithm's implementation, including representation, selection, crossover, and mutation, are discussed. The results include an analysis of the algorithm's performance, parameter sensitivity, and its comparison across various configurations.

1 Introduction

Optimization problems are pervasive in science and engineering, often requiring methods to find the best solution from a set of feasible options. Genetic algorithms (GAs), inspired by the process of natural selection, offer a robust and adaptable approach to solving such problems. This report explores the application of a GA to find the minimum value of a function. We describe the problem setup, motivation, and the relevance of these test functions in benchmarking optimization algorithms.

2 Methods

Rastrigin function:

$$f(x) = A \cdot n + \sum_{i=1}^{n} \left[x_i^2 - A \cdot cos(2\pi x_i)\right], A = 10, x_i \in [-5.12, 5.12], i = 1..n$$

De Jong Function:

$$f(x) = \sum_{i=1}^{n} x_i^2, x_i \in [-5.12, 5.12], i = 1..n$$

Michalewicz Function:

$$f(x) = -\sum_{i=1}^{n} \sin(x_i) \cdot \left(\sin\left(\frac{i \cdot x_i^2}{\pi} \right) \right)^{2m}, m = 10, x_i \in [0, \pi], i = 1..n$$

Schwefel function:

$$f(x) = 418.9829 \cdot n - \sum_{i=1}^{n} x_i \sin\left(\sqrt{|x_i|} \right), x_i \in [-500, 500], i = 1..n$$

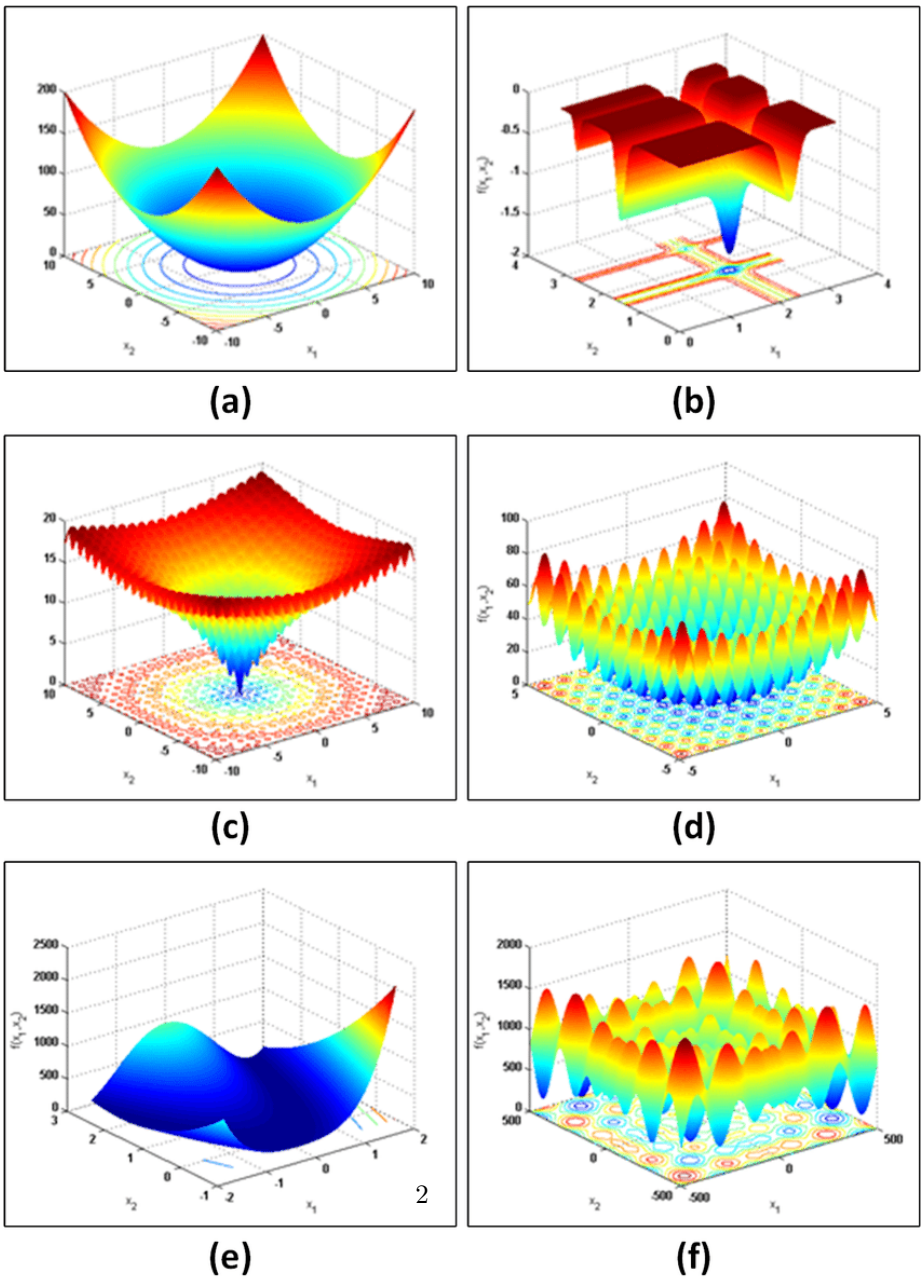

Figure 1: (a) De Jong Function, (b) Michalewicz Function, (c) Ackley Function, (d) Rastrigin Function, (e) Rosenbrock Function, (f) Schwefel Function[2]

The implemented GA operates as follows:

1. **Representation**: Solutions are encoded as binary strings. The bitstring length required for the binary encoding of a solution is given by the formula

$$n \cdot ceil\left(\log_2\left((b-a)\cdot 10^d\right)\right),$$

where n is the number of dimensions of the function to be optimized, and the other term of the product is the number of bits required to represent a single parameter of the function in binary, depending on the precision d. The most important part, decoding a solution from binary to real numbers (which is done to evaluate the solution), is achieved with the formula

$$a + x \cdot \frac{b-a}{2^L - 1},$$

where x is the base-10 representation of the bit sequence corresponding to parameter x_i of the current solution's binary representation ($i = 1..n$), $2^L - 1$ is the maximum value in base-10 that can be formed with L bits ($L = ceil\left(\log_2\left((b-a)\cdot 10^d\right)\right)$), and a and b represent the endpoints of the interval over which the function to be optimized is defined ($a < b$). This last step ensures that each parameter is converted from binary form to the space of real numbers, $\in$ [a, b].

2. **Population Initialization**: The initial population is generated randomly and consists of $popSize$ individuals (candidate solutions). Each individual is represented by a bit vector, which is generated randomly.

3. **Selection Methods**: Multiple strategies were explored, including:

   - *Roulette Wheel Selection* with two fitness scaling approaches.
     - *Roulette Wheel (Method 1):* Each individual has a selection probability inversely proportional to its fitness. Thus, better individuals have higher chances of being chosen.
     - *Roulette Wheel (Method 2):* Similar to the first method, but with a modification to the selection probability to avoid an inequitable selection process.
   - *Tournament Selection.*
     A random subset of individuals is chosen, and the best individual from this subset is selected for reproduction.
   - *Elitism.*
     The best individuals from the population are selected to be part of the new population without any modification.

4. **Crossover**: A single-point crossover with a probability P_{cr} was employed.

5. **Mutation**: Mutation was applied with a dynamic probability. The implementation of mutation involves a probability vector that is updated with

each new generation of individuals, taking into account the index of the current population. In this way, at the beginning of the algorithm, the mutation vector increases the likelihood of mutation in the initial section of each parameter in the candidate's binary representation (encouraging an explorative approach). Towards the end of the algorithm, the final section of each parameter becomes more likely to mutate. The probability vector is computed using this formula:

$$e^{\frac{-(x-m)^2}{2\sigma^2}} \cdot sin(P_{mut} \cdot 0.8)$$

where x is the position of a candidate of the algorithm scaled to the interval $[0,\ \sigma]$, m is the index inside the main generations loop, scaled to the interval $[0,\ 1]$, and sigma is a positive number. This formula is linked to the formula of normal distribution, which is also used in the Simulated Annealing algorithm.

2.1 Key Choices and Parameters

- **Fitness Evaluation**: The function values were computed for test functions using the decoded binary representation.
- **Stopping Criterion**: The algorithm terminated after a fixed number of generations.
- **Algorithm Parameters**: The main parameters of the genetic algorithm are:
  - *Initial Population Size* ($popSize$)
  - *Number of Generations* ($nrGenerations$)
  - *Mutation Probability* (P_{mut})
  - *Crossover Probability* (P_{cr})
  - *Selection Method* ($selectionMethod$)

3 Results

We analyzed the four selection versions implemented within the genetic algorithm.Then, to make a comparison, we also used the results obtained from Hill Climbing and Simulated Annealing.

The results for the Hill Climbing algorithms have been executed with 1000 repetitions, and the Simulated Annealing algorithm have been executed with parameters $T = 640$, $\sigma = 400$. All programs (HC Best Improvement/Simulated Annealing/Genetic Algorithm with the four variants of selection) have been run 40 times.

Rastrigin Function

The GA consistently found solutions close to the global minimum, with better results observed for smaller dimensionality ($n = 5$ and $n = 10$). While Hill Climbing was competitive in low-dimensional cases, it struggled with higher dimensions, showing larger deviations from the optimal solution. Simulated Annealing demonstrated intermediate performance but required significantly more computation time compared to GA.

Michalewicz Function

The Michalewicz function, known for its complex landscape with multiple local optima, highlighted the advantage of GA's diversity-maintaining mechanisms. The use of tournament selection and elitism improved convergence towards the global minima. Hill Climbing often converged prematurely to suboptimal solutions, particularly in higher dimensions ($n = 10$ and $n = 30$).

Schwefel Function

The GA showed good performance across all dimensions, effectively avoiding the local optima that hindered Hill Climbing. Simulated Annealing managed to find near-optimal solutions but required significantly more iterations, demonstrating GA's advantage in computational efficiency.

DeJong Function

The simplicity of this function allowed all methods to perform well in lower dimensions ($n = 5$ and $n = 10$). However, in higher dimensions, the GA achieved superior precision and demonstrated lower variability in results compared to Hill Climbing and Simulated Annealing.

Rastrigin function parameters:

- **Domain:** [-5.12, 5.12]
- **Population Size:** 30
- **Number of Generations:** 20000
- **Tournament Size or Number of Elites (only for Selection Method elitism or tournament):** 2
- **Mutation Probability (P_Mutation):** 0.05
- **Sigma:** 0.3
- **Crossover Probability (P_Crossover):** 0.01
- For Rastrigin function with 30 dimensions, the tests have been made with popSize=110, nrGenerations=25000, P_Mutation=0.03704 and sigma=0.23

	HC - Best	SA	GA-R1	GA-R2	GA-T	GA-E
$n = 5$						
Min. value	0.00000	0.00000	1.03531	0.00012	0.00000	0.00000
Max. value	1.23582	8.14915	9.71061	5.45731	1.23582	1.23582
Mean	0.40413	2.67206	3.96142	2.37309	0.15448	0.03090
Standard Deviation	0.50265	1.68146	2.14535	1.09662	0.41392	0.19540
Average time(sec)	1.39	7.15	0.93	0.93	1.18	0.91
$n = 10$						
Min. value	0.99496	1.98992	7.20904	3.03435	0.00000	0.00000
Max. value	5.45652	12.37491	27.72477	14.48292	7.91334	3.70747
Mean	3.61979	5.03176	13.96850	7.20508	3.61221	0.67970
Standard Deviation	0.97000	2.55480	4.51346	2.86879	1.90772	1.04526
Average time(sec)	7.62	12.21	1.68	1.68	2.00	1.66
$n = 30$						
Min. val.	18.85637	11.20057	47.04254	15.68379	10.92820	1.23582
Max. val	32.95540	28.22147	112.70178	43.57623	31.94410	17.26176
Mean	27.70595	18.15091	64.74406	29.18403	20.31507	6.29723
St. Dev.	2.99251	4.01798	14.13514	5.73216	5.87178	3.70064
Avg. time	168.66	33.58	22.33	22.25	25.63	21.77

Table 1: Rastrigin: exact global minima: n=5/10/30 $->$ 0

Michalewicz function parameters:

- **Domain:** [0, π]
- **Population Size:** 100
- **Number of Generations:** 2500
- **Tournament Size or Number of Elites (only for Selection Method elitism or tournament):** 2
- **Mutation Probability (P_Mutation):** 0.05263
- **Sigma:** 0.23
- **Crossover Probability (P_Crossover):** 0.01

	HC - Best	Simulated Annealing	GA-R1	GA-R2	GA-T	GA-E
$n = 5$						
Min. value	-4.68766	-4.68593	-0.98330	-0.00000	-4.68765	-4.68766
Max. value	-4.68153	-3.54586	-0.00000	-4.44664	-4.49147	-4.22151
Mean	-4.68681	-4.23580	-0.32555	-4.57308	-4.60333	-4.62307
Standard Deviation	0.00140	0.31140	0.28276	0.08903	0.07629	0.09472
Average time(sec)	1.71	0.01	0.42	0.45	0.57	0.44
$n = 10$						
Min. value	-9.57161	-9.24567	-0.89231	-9.61946	-9.54779	-9.61837
Max. value	-9.25546	-7.37724	-0.07809	-8.48694	-8.34763	-9.07125
Mean	-9.40824	-8.36363	-0.46285	-9.15892	-9.20299	-9.41060
Standard Deviation	0.07306	0.49831	0.21429	0.27073	0.25927	0.11494
Average time(sec)	10.24	0.02	0.88	0.89	1.11	0.88
$n = 30$						
Min. value	-27.54316	-27.14902	-1.15432	-27.92602	-27.69691	-28.86678
Max. value	-26.58271	-23.28145	-0.09809	-24.27905	-24.82141	-26.57300
Mean	-27.00167	-25.11001	-0.45314	-25.87550	-26.43913	-27.59616
Standard Deviation	0.21404	0.99451	0.23952	0.68508	0.65351	0.51837
Average time(sec)	216.71	0.25	2.48	2.49	3.02	2.47

Table 2: Michalewicz: exact global minima: n=5 $->$ -4.68765 ; n=10 $->$ -9.66015 ; n=30 $->$ -29.9

Schwefel function parameters:

- **Domain:** [-500.00, 500.00]
- **Population Size:** 110
- **Number of Generations:** 25000
- **Tournament Size or Number of Elites (only for Selection Method elitism or tournament):** 2
- **Mutation Probability (P_Mutation):** 0.03704
- **Sigma:** 0.23
- **Crossover Probability (P_Crossover):** 0.02

	HC - Best	Simulated Annealing	GA-R1	GA-R2	GA-T	GA-E
$n = 5$						
Min. value	0.00011	0.00069	0.00143	0.00070	0.00006	0.00069
Max. value	0.20865	34.44355	0.44915	0.41536	0.51840	0.41474
Mean	0.09165	1.01453	0.20199	0.22374	0.14570	0.25183
Standard Deviation	0.06684	5.42187	0.11802	0.10618	0.12798	0.09910
Average time(sec)	3.31	58.45	5.17	5.14	6.20	5.02
$n = 10$						
Min. value	0.62312	0.00201	0.29703	0.10630	0.10382	0.00202
Max. value	248.34030	34.96195	254.89464	152.88293	0.72706	0.72642
Mean	107.34842	5.45540	61.17108	9.80516	0.33587	0.33565
Standard Deviation	53.76364	12.41442	66.80400	31.92667	0.16170	0.16312
Average time(sec)	21.04	107.05	9.98	9.94	11.60	9.80
$n = 30$						
Min. value	888.09205	0.72923	1223.3374	69.94138	35.07188	0.73108
Max. value	1411.22948	119.89815	3258.46237	903.11208	696.33655	1.66345
Mean	1226.33757	30.82801	1833.65181	435.28544	288.71188	1.19695
Standard Deviation	144.79832	38.17938	415.38938	193.16499	163.43952	0.23221
Average time(sec)	483.60	299.78	29.43	29.36	34.50	29.09

Table 3: Schwefel: exact global minima: n=5/10/30 $->$ 0

DeJong function parameters:

- **Domain:** [-5.12, 5.12]
- **Population Size:** 20
- **Number of Generations:** 1000
- **Tournament Size or Number of Elites (only for Selection Method elitism or tournament):** 2
- **Mutation Probability (P_Mutation):** 0.00167
- **Sigma:** 0.3
- **Crossover Probability (P_Crossover):** 0.01

	HC - Best	Simulated Annealing	GA-R1	GA-R2	GA-T	GA-E
$n = 5$						
Min. value	0.00000	0.00000	0.00078	0.00000	0.00000	0.00000
Max. value	0.00000	0.00000	2.96866	0.00160	0.00000	0.00000
Mean	0.00000	0.00000	0.15597	0.00004	0.00000	0.00000
Standard Deviation	0.00000	0.00000	0.47467	0.00025	0.00000	0.00000
Average time(sec)	1.42	0.54	0.04	0.04	0.05	0.04
$n = 10$						
Min. value	0.00000	0.00000	0.36046	0.00000	0.00000	0.00000
Max. value	0.00000	0.00000	7.68439	0.10241	0.00640	0.00000
Mean	0.00000	0.00000	2.76745	0.00479	0.00021	0.00000
Standard Deviation	0.00000	0.00000	1.80227	0.01646	0.00104	0.00000
Average time(sec)	8.15	1.00	0.08	0.08	0.09	0.08
$n = 30$						
Min. value	0.00000	0.00000	31.54049	0.34026	0.00310	0.00000
Max. value	0.00000	0.00040	138.93472	3.76645	0.22153	0.00010
Mean	0.00000	0.00002	83.82518	1.62138	0.04077	0.00000
Standard Deviation	0.00000	0.00007	29.22889	0.94958	0.04834	0.00002
Average time(sec)	182.61	2.61	0.23	0.23	0.25	0.23

Table 4: De Jong: exact global minima: n=5/10/30 $->$ 0

4 Parameter Influence

The influence of parameters like mutation probability, selection method, and crossover probability was studied. It was observed that:

- Lower mutation rates led to premature convergence. Low mutation rates keep the genetic structures of the population too similar, which limits the ability to adapt to better solutions. The population tends to quickly align around already dominant solutions, even if they are not globally optimal.
- In roulette wheel selection, individuals with higher fitness have a much greater chance of being selected, which can lead to the dominance of a few very fit individuals early on. This reduces diversity, as other individuals have lower chances of contributing to the next generation.

  Tournament selection chooses individuals based on competition within small groups, giving greater opportunities to individuals with lower fitness as well. This allows more solutions to contribute, preserving genetic diversity.
- Higher crossover probabilities improve exploration because they increase the chance of generating new combinations of genetic material through the recombination of solutions.

5 Comparative Analysis

When compared to Hill Climbing and Simulated Annealing, the Genetic Algorithm consistently produces better results. While Hill Climbing is a local search algorithm that can easily get stuck in local optima, and Simulated Annealing relies on probabilistic decisions that may not always converge to the best solution, the GA benefits from its population-based search and genetic operators (selection, crossover, and mutation) that allow it to explore the solution space more effectively.

6 Conclusions

The genetic algorithm demonstrated effectiveness in solving complex optimization problems, with the performance depending significantly on parameter settings and problem characteristics.The Genetic Algorithm achieves superior solution quality when compared to Hill Climbing and Simulated Annealing. This makes the GA a more efficient and scalable choice for solving complex optimization problems.Future research could explore hybrid approaches combining GAs with other optimization methods.

References

[1] https://ro.wikipedia.org/wiki/Distribu%C8%9Bia_Gauss

[2] https://www.researchgate.net/figure/a-Sphere-Function-b-Michalewicz-Function-c-Ackley-Fu
fig5_269309904

[3] https://chatgpt.com/

[4] https://en.wikipedia.org/wiki/Rastrigin_function

[5] https://www.sfu.ca/~ssurjano/michal.html

[6] https://gitlab.com/eugennc/teaching/-/blob/master/GA/texample.
tex

[7] https://www.geeksforgeeks.org/genetic-algorithms/