Testing standard benchmark functions with Hill Climbing and Simulated Annealing

Albert Alexandru

October 2023

1 Abstract

The goal of this work is to determine the global minimum of well known, multi-dimensional functions by comparing two popular optimization algorithms: Hill Climbing and Simulated Annealing. This study aims to assess these algorithms performance in terms of computing efficiency and accuracy.

2 Introduction

This paper presents a investigation into the efficacy of heuristic optimization methods for function minimization across varying dimensions. The study focuses on four widely recognized benchmark functions: De Jong 1, Schwefel's, Rastrigin's, and Michalewicz's. The objective is to assess the performance of two heuristic approaches - hill climbing algorithms (implemented in three different variations: best improvement, worst improvement, and first improvement) and simulated annealing, in the context of binary string representation of the candidate solutions.

To facilitate a thorough comparison, the experiments are conducted across three distinct dimensionalities: 5, 10, and 30. This range of dimensions enables us to investigate the robustness of the chosen methods under varying levels of complexity.

3 Methods

In this section, we will conduct a thorough analysis of the two algorithms employed: hill climbing and simulated annealing, along with their respective variations.

3.1 Hill Climbing

In the implemented approach, hill climbing operates by examining neighbors with a Hamming distance of 1. The initial solution is represented as a binary string of a specific length. For the sake of clarity, let's consider a one-dimensional function as the simplest case. The starting point of the function is the solution where all bits have a value of 0, and the end point is the solution where all bits have a value of 1. By randomly flipping a bit, we introduce a perturbation in the function's value, which may be greater or smaller depending on the position of the flipped bit. The following variants of the hill climbing (HC) method were used, all of them iteratively:

3.1.1 Best Improvement HC

The "Best Improvement" hill climbing algorithm, when applied to a binary string, works by iteratively evaluating the neighboring solutions with a Hamming distance of 1 (i.e., solutions that differ by only one bit). It selects the neighbor that yields the highest improvement in the objective function value compared to the current solution. If the objective function value of the best neighboring solution is better than the current solution, it becomes the new current solution. This process is repeated until no further improvement is possible, indicating that a local minimum has been reached.

3.1.2 Worst Improvement HC

The "Worst Improvement" hill climbing algorithm, when applied to a binary string, operates in a similar manner to the "Best Improvement" algorithm. It evaluates neighboring solutions with a Hamming distance of 1. However, unlike the "Best Improvement" algorithm, the "Worst Improvement" algorithm selects the neighbor that results in the least improvement in the objective function value compared to the current solution. If the objective function value of the worst neighboring solution is better than the current solution, it becomes the new current solution. This process continues until no further improvement is possible, indicating that a local minimum has been reached.

3.1.3 First Improvement HC

The "First Improvement" hill climbing algorithm, when applied to a binary string, begins by evaluating neighboring solutions with a Hamming distance of 1. Unlike the "Best Improvement" and "Worst Improvement" approaches, the algorithm adopts a 'first come, first serve' strategy. The algorithm initiates its search from a randomly selected position within the binary string, and it stops as soon as it encounters the first neighboring solution that offers an improvement in the objective function value compared to the current solution. If the objective function value of this first improving neighbor surpasses that of the current solution, it is promptly adopted as the new current solution. This process continues until no further improvement is possible, signifying that a local minimum has been reached.

3.2 Simulated Annealing

Simulated annealing is a stochastic optimization algorithm inspired by the annealing process in metallurgy. It shares similarities with hill climbing, yet it introduces an element of probabilistic exploration. Unlike its more deterministic counterparts, simulated annealing is willing to occasionally accept worse solutions. This adaptability helps it to escape local minima in complex and non-convex functions. However, it's worth noting that on simpler, convex functions, it may be surpassed in terms of accuracy by more deterministic approaches like hill climbers.

In the approach discussed in this implementation, experiments were conducted without employing the algorithm iteratively, although iterative use can yield better results, at the cost of execution time. Initially, a binary string is generated. Subsequently, for a certain temperature, the main loop is executed a substantial number of times, around 200,000 iterations. Within this loop, a random neighbor is generated by flipping a number of bits, and an acceptance probability is calculated, in the event that the solution is inferior to the current "best solution." This acceptance probability introduces a probabilistic element, influencing whether the solution is adopted.

Should the solution be rejected, a "no solution" variable is incremented, and at the end of the loop, the number of inner iterations is reduced by one. This loop persists until a predefined number of "no solutions" have been encountered, at which point the temperature undergoes a gradual cooling process.

4 Experimental setup description

Each function studied was executed in three distinct dimensionalities: 5, 10, and 30. Almost every experiment utilized a small ε value of 1×10^{-4}
, although the number of iterations varied significantly.

When it came to determining the minimum for De Jong 1, every algorithm and its respective version successfully found the minimum within a single iteration. In the case of Michalewicz and Schwefel's functions, 10 iterations were used for both best and worst improvement, while first improvement was tested with 100 iterations.

For the Rastrigin function, a total of 100 iterations were applied for both worst and first improvement, but a more substantial 1000 iterations were necessary to achieve an acceptable result in the 30 dimensional space.

- For Michalewicz Simulated Annealing, the chosen parameters were:
  - $\varepsilon = 1 \times 10^{-4}$
  - $T = 25$
  - $\alpha = 0.9955$
  - Minimum temperature is $\alpha = 1 \times 10^{-8}$
  - Inner iterations = 200,000

  - Neighbors were generated by flipping 4 bits.
  - The main loop terminated upon encountering 2 instances of 'no solutions.'

- For the remaining functions, simulated annealing was performed with the following parameters:
  - $\varepsilon = 1 \times 10^{-5}$
  - $T = 100$
  - $\alpha = 0.999$
  - Minimum temperature is $\alpha = 1 \times 10^{-8}$
  - Inner iterations = 200,000
  - Neighbors were generated by flipping 5 bits.
  - The main loop terminated upon encountering 5 instances of 'no solutions.'

5 Experimental results

5.1 De Jong 1 function

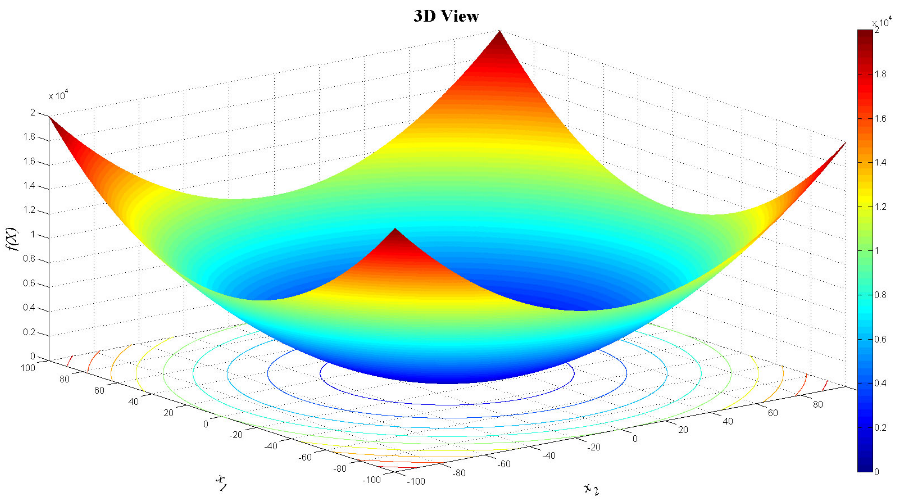

Figure 1: De Jong 1 Function, for d = 2.

$$f(\mathbf{x}) = \sum_{i=1}^{d} x_i^2, \quad x_i \in [-5.12, 5.12] for\, i = 1, \ldots, d$$

Dim	Mean	Min	Max	StdDev	Q1 - Q3	Mean Time (s)	Min Time	Max Time
5	0.00000	0.00000	0.00000	0.00000	0.00000 - 0.00000	0.00281	0.00206	0.00344
10	0.00000	0.00000	0.00000	0.00000	0.00000 - 0.00000	0.01590	0.01268	0.01995
30	0.00000	0.00000	0.00000	0.00000	0.00000 - 0.00000	0.32021	0.29484	0.34358

Table 1: Results for De Jong 1 HC Best Improvement

Dim	Mean	Min	Max	StdDev	Q1 - Q3	Mean Time (s)	Min Time	Max Time
5	0.00000	0.00000	0.00000	0.00000	0.00000 - 0.00000	0.02811	0.02328	0.03279
10	0.00400	0.00000	0.10668	0.02671	0.00000 - 0.00000	0.07242	0.06250	0.08884
30	0.00000	0.00000	0.00000	0.00000	0.00000 - 0.00000	4.26696	3.72207	4.67766

Table 2: Results for De Jong 1 HC First Improvement

Dim	Mean	Min	Max	StdDev	Q1 - Q3	Mean Time (s)	Min Time	Max Time
5	0.00000	0.00000	0.00000	0.00000	0.00000 - 0.00000	0.00462	0.00362	0.00711
10	0.00000	0.00000	0.00000	0.00000	0.00000 - 0.00000	0.02451	0.02044	0.03193
30	0.00000	0.00000	0.00000	0.00000	0.00000 - 0.00000	0.39690	0.36489	0.46920

Table 3: Results for De Jong 1 HC Worst Improvement

Dim	Mean	Min	Max	StdDev	Q1 - Q3	Mean Time (s)	Min Time	Max Time
5	0.00010	0.00001	0.00049	0.00015	0.00006 - 0.00011	2.48809	2.44728	2.66057
10	0.00013	0.00003	0.00027	0.00006	0.00010 - 0.00019	2.60205	2.53829	2.75840
30	0.00042	0.00023	0.00080	0.00019	0.00029 - 0.00049	2.94586	2.85368	3.19968

Table 4: Results for De Jong 1 SA

5.2 Schwefel's function

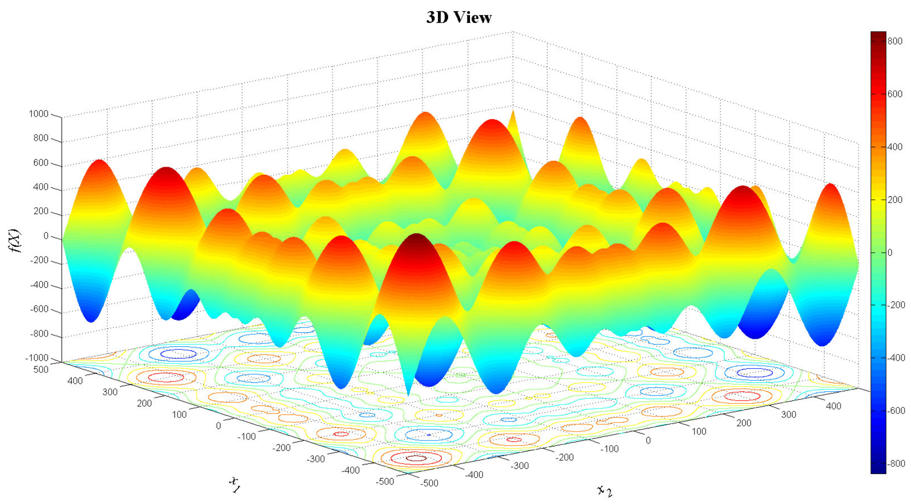

Figure 2: Schwefel's Function, for d = 2.

$$f(\mathbf{x}) = -\sum_{i=1}^{d} x_i \sin(\sqrt{|x_i|}), \quad x_i \in [-500, 500] for\, i = 1, \ldots, d$$

Dim	Mean Value	Min Value	Max Value	StdDev	Q1 - Q3	Mean Time (s)	Min Time	Max Time
5	-1908.02419	-2094.70583	-1742.20160	108.74804	-1972.24597 - -1826.51343	0.06088	0.05683	0.06727
10	-3727.90537	-4009.75338	-3346.54824	235.09926	-3841.31738 - -3619.06543	0.41555	0.38554	0.49758
30	-10599.32729	-11105.20120	-9972.67254	212.11733	-10747.67047 - -10427.34720	8.84485	8.47862	9.23810

Table 5: Results for Schwefel HC Best Improvement

Dim	Mean Value	Min Value	Max Value	StdDev	Q1 - Q3	Mean Time (s)	Min Time	Max Time
5	-1809.06826	-2006.80900	-1584.42001	78.51587	-1862.45092 - -1743.19151	0.04591	0.03824	0.05357
10	-3408.22889	-3631.39468	-3174.80765	139.27119	-3525.50642 - -3288.30941	0.13656	0.11655	0.16953
30	-9533.82448	-10452.97991	-9181.63728	353.73452	-9842.81182 - -9288.01038	1.00911	0.62043	1.43038

Table 6: Results for Schwefel HC First Improvement

Dim	Mean Value	Min Value	Max Value	StdDev	Q1 - Q3	Mean Time (s)	Min Time	Max Time
5	-1931.97274	-2094.80951	-1739.00108	87.09938	-1975.67532 - -1871.04956	0.08852	0.08123	0.09500
10	-3692.78818	-4086.70634	-3452.78753	194.25225	-3791.06209 - -3585.93613	0.52517	0.48899	0.56222
30	-10638.02042	-11053.27588	-10141.52306	248.70969	-10800.00839 - -10500.76882	10.23611	9.59578	10.84585

Table 7: Results for Schwefel HC Worst Improvement

Dim	Mean Value	Min Value	Max Value	StdDev	Q1 - Q3	Mean Time (s)	Min Time	Max Time
5	-2094.74442	-2094.91443	-2094.49970	0.11925	-2094.81076 - -2094.70708	2.45391	2.37371	2.67453
10	-4189.41261	-4189.72329	-4189.10121	0.16628	-4189.51592 - -4189.30912	2.57566	2.52442	2.67331
30	-12516.92261	-12568.64841	-12296.91321	105.20059	-12567.92363 - -12534.02508	2.95995	2.89958	3.02322

Table 8: Results for Schwefel SA

5.3 Rastrigin's function

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

Dim	Mean Value	Min Value	Max Value	StdDev	Q1 - Q3	Mean Time (s)	Min Time	Max Time
5	1.44861	0.00000	3.46737	0.89178	0.99496 - 1.99506	0.26321	0.25238	0.27365
10	6.23946	2.47241	9.69891	1.63525	5.46242 - 6.45738	1.68022	1.62143	1.83647
30	28.03236	20.33712	32.23608	3.21356	26.29117 - 30.24571	389.93805	316.20494	620.44842

Table 9: Results for Rastrigin HC Best Improvement

5.4 Michalewicz's function

$$f(\mathbf{x}) = -\sum_{i=1}^{d} \sin(x_i) \sin^{2m}\left(\frac{i x_i^2}{\pi}\right), \quad x_i \in [0, \pi] for\, i = 1, \ldots, d$$

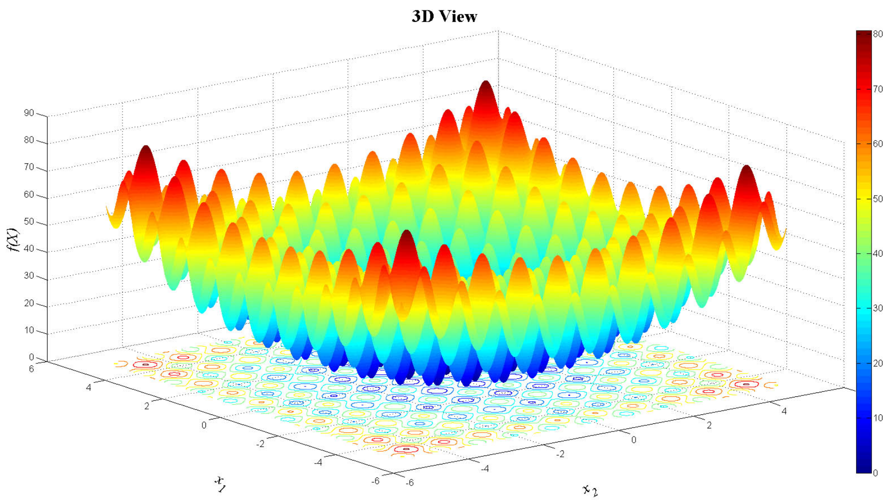

Figure 3: Rastrigin's Function, for d = 2.

Dim	Mean Value	Min Value	Max Value	StdDev	Q1 - Q3	Mean Time (s)	Min Time	Max Time
5	5.43565	1.28155	9.62755	3.07037	2.15345 - 8.32353	0.03158	0.02639	0.04694
10	15.33677	7.44935	23.49978	5.68684	10.72226 - 17.33867	0.08636	0.07368	0.10289
30	64.67462	47.50214	79.36162	8.26211	57.28073 - 72.29672	0.60509	0.49504	0.71880

Table 10: Results for Rastrigin HC First Improvement

Dim	Mean Value	Min Value	Max Value	StdDev	Q1 - Q3	Mean Time (s)	Min Time	Max Time
5	1.45830	0.00000	2.99001	0.84774	0.99496 - 1.98992	0.38163	0.36258	0.49063
10	6.58579	2.99002	9.39144	1.54509	5.69853 - 7.44720	2.12379	2.04981	2.32977
30	34.73537	25.26110	40.47385	3.60134	32.14903 - 36.84746	39.74116	37.72270	51.89658

Table 11: Results for Rastrigin HC Worst Improvement

Dim	Mean Value	Min Value	Max Value	StdDev	Q1 - Q3	Mean Time (s)	Min Time	Max Time
5	0.12359	0.00000	1.23584	0.32741	0.00001 - 0.00002	2.17162	2.11141	2.25132
10	1.48242	0.00001	8.63112	2.15649	1.23585 - 1.23591	2.33101	2.23970	2.47920
30	15.54186	6.28744	26.49976	4.07673	13.70454 - 16.70799	2.75145	2.71866	2.92706

Table 12: Results for Rastrigin SA

Dim	Mean Value	Min Value	Max Value	StdDev	Q1 - Q3	Mean Time (s)	Min Time	Max Time
5	-4.51002	-4.68459	-4.18446	0.16932	-4.56358 - -4.36944	0.02614	0.02294	0.02948
10	-8.70178	-9.29669	-8.06520	0.16332	-8.89442 - -8.45789	0.16384	0.14983	0.17629
30	-25.44161	-26.22963	-24.79306	0.37175	-25.75245 - -25.06571	3.29517	3.17412	3.40947

Table 13: Results for Michalewicz HC Best Improvement

Dim	Mean Value	Min Value	Max Value	StdDev	Q1 - Q3	Mean Time (s)	Min Time	Max Time
5	-4.16994	-4.51640	-3.82142	0.16543	-4.31623 - -4.00986	0.02679	0.02149	0.03368
10	-8.05969	-8.70117	-7.03248	0.15541	-8.28325 - -7.89904	0.08214	0.07117	0.09408
30	-22.72989	-23.96036	-20.71319	0.62411	-23.30827 - -22.08870	0.54637	0.40240	0.65311

Table 14: Results for Michalewicz HC First Improvement

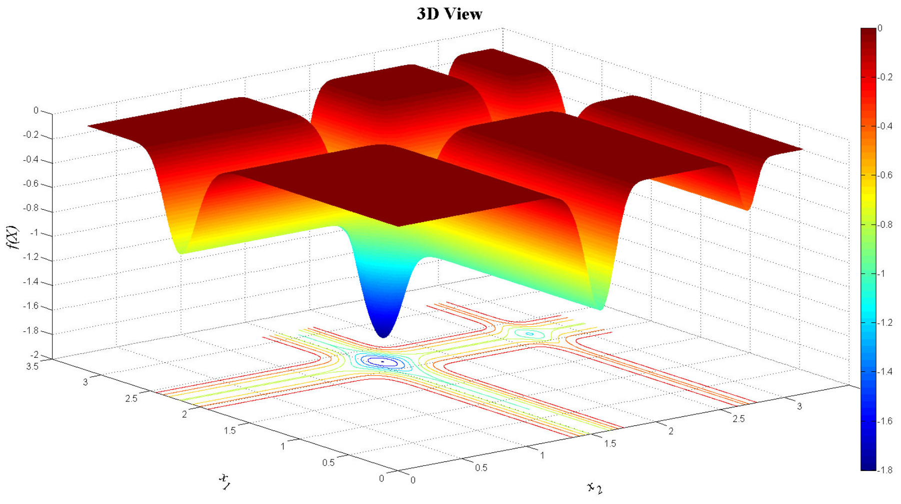

Figure 4: Michalewicz's Function, for d = 2.

Dim	Mean Value	Min Value	Max Value	StdDev	Q1 - Q3	Mean Time (s)	Min Time	Max Time
5	-4.50178	-4.68751	-4.04505	0.06434	-4.55888 - -4.47467	0.03683	0.03212	0.04920
10	-8.74388	-9.26110	-8.16409	0.22713	-8.87547 - -8.61717	0.20230	0.18868	0.21636
30	-25.57418	-27.35137	-24.53777	0.58253	-25.76553 - -25.14459	3.59323	3.45392	3.81242

Table 15: Results for Michalewicz HC Worst Improvement

Dim	Mean Value	Min Value	Max Value	StdDev	Q1 - Q3	Mean Time (s)	Min Time	Max Time
5	-4.55516	-4.68765	-4.21161	0.10779	-4.63976 - -4.53706	1.48840	1.41333	1.63201
10	-8.94762	-9.38094	-8.41197	0.25824	-9.07613 - -8.74285	1.45313	1.35056	1.66579
30	-26.01103	-27.45690	-24.42465	0.65915	-26.59862 - -25.40936	1.63445	1.42816	1.87693

Table 16: Results for Michalewicz SA

6 Algorithm Performance Comparison

The results, in the higher dimensional space, for the more complex functions clearly demonstrated the superior performance of the simulated annealer. This was most evident in the case of Rastrigin's 30-dimensional function, where the best improvement method returned a mean value of approximately 28 in over 5 minutes, while the simulated annealer achieved an average of around 15 in just under 3 seconds. A similar trend was observed for Schwefel's function, where the simulated annealer approached the global minimum, although the margin of improvement was not as pronounced.

Conversely, for the simpler functions, the difference in performance was much narrower. In fact, for De Jong 1, the simulated annealer was surpassed in both speed and accuracy by every hill climber.

In the case of Michalewicz's function in the 30-dimensional space, both the best and worst improvement methods achieved decent results comparable to those of the simulated annealer, but at the cost of taking twice as much time to execute.

In the lower-dimensional space (5 dimensions), the difference was nearly negligible, especially considering the time taken for execution. For Rastrigin, the most complex function, HC Best Improvement achieved a value of 1.44861 in just 0.27365 seconds. SA, while obtaining a superior result of 0.12359 (over 10 times better), also took 10 times more time to execute. As for Michalewicz's function, both HC Best Improvement and SA produced nearly identical outcomes (-4.51002 compared to -4.55516), but HC executed in 0.02948 seconds, whereas SA required 1.63201 seconds. Moving on to Schwefel's function, SA managed to closely approach the global minimum with a value of -2094.74442 (the global minimum being -2094.9145) in 2.45391 seconds. Meanwhile, HC Worst Improvement came close, reaching -1931.97274 in just 0.08852 seconds.

7 Conclusion

In summary, this study compared the performance of hill climber algorithms and their variations to the simulated annealer on various mathematical functions commonly used for benchmarking. We observed that hill climbers excel at simpler functions like De Jong 1 or Michalewicz's and perform quite well overall in lower number of dimensions. However, in higher-dimensional spaces, they struggle with more complex functions like Rastrigin's, often getting stuck in local minima.

On the other hand, the simulated annealer consistently outperforms hill climbers on complex functions, quickly finding better local minima. The algorithm demonstrates better overall adaptability, being able to yield very good results even in higher-order dimensional problems. This study concludes that simulated annealing is the preferred method for navigating complex and multimodal landscapes to find the global minimum.

References

[1] Function photos. https://al-roomi.org/benchmarks/unconstrained/n-dimensions/174-generalized-rastrigin-s-function

[2] Wikipedia, Simulated annealing. https://en.wikipedia.org/wiki/Simulated_annealing

[3] Wikipedia, Hill climbers. https://en.wikipedia.org/wiki/Hill_climbing

[4] Eugen Croitoru, Teaching: Genetic Algorithms https://profs.info.uaic.ro/~eugennc/teaching/ga/

[5] Robert Miles, Hill Climbing Algorithm and Artificial Intelligence - Computerphile. https://www.youtube.com/watch?v=oSdPmxRCWws&ab_channel=Computerphile