Analysis of Hill Climbing and Simulated Annealing algorithms on a given set of functions

Zaborilă Andrei

November the 4th, 2020

1 Introduction

This report is meant to present the differences between non-deterministic algorithms on the subject of finding the global minima of a function. Three methods based on two algorithms will be used on four different functions; based on the results, conclusions regarding the differences of time, accuracy and behaviour will be drawn.

1.1 Motivation

Studying the usage of Hill Climbing and Simulated Annealing algorithms is important because there are functions and problems that cannot be solved in a reasonable time using deterministic algorithms, while non-deterministic ones can provide an approximate answer in a decent amount of time.

2 Method

Three non-deterministic methods, two based on Hill Climbing and one on Simulated Annealing, have been used for each of the four different functions in order to analyse their accuracy. The functions differ in their number of local minima therefore it is easier to find a global minima for some, and more difficult for others. All the three methods follow a set of rules and a similar path in order to try to find the global minima. The solutions are stored only in binary format, while the evaluation of the results is made through a function that converts the binary input into a decimal value.

Best-Improvement Hill Climbing is the most time-consuming of the three methods because it iterates through all the neighbours of the point and only chooses the next point based on the best neighbour found.

First-Improvement Hill Climbing is much like the aforementioned method, the main difference being that instead of going through all the neighbours of the point and looking for the best one, it chooses the first neighbour that satisfies the given condition. It is less time-consuming but it doesn't always provide results as accurate as its counterpart. For these two methods we try to find **10000** local minima and then choose the best result from them.

Simulated Annealing is similar to the others but it provides the option for any point to jump to a "bad" neighbour (one that doesn't satisfy the given condition), based on a temperature that is lowered with each iteration. For the **Simulated Annealing** method we initialize the temperature with **100** and stop when it is lower or equal than 10^{-9}. It will be multiplied by 0.9942601, resulting in approximately **4000**[2] changes in temperature before reaching the halting condition.

A sample size of **36** was chosen for each size (**5, 10, 30**) of each function for each method and **12** threads will be used to run the experiments, on an AMD Ryzen 3700X.

3 Experiments

A precision of five decimals has been chosen for all the experiments, so all the results have been truncated where needed.

3.1 De Jong's function

3.1.1 The function

$$f(x) = \sum_{i=1}^{n} x_i^2 \qquad -5.12 \leq x_i \leq 5.12$$

Global minimum: $f(x) = 0, \quad x_i = 0, \quad i = 1 : n$

De Jong's function is also known as "The Spehere function". It has a number of local minima equal to the number of dimensions, except for the global one. It is continuous, convex and unimodal. **Figure 1** shows its two-dimensional form.[9]

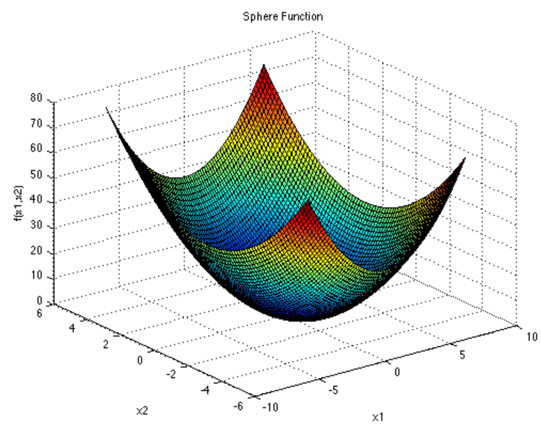

Figure 1: De Jong's function

3.1.2 Hill Climbing First-Improvement results

Dimension	Min	Max	Mean	σ	Min Time	Max Time	Mean Time
5	0	0	0	0	14.218	14.516	14.382
10	0	0	0	0	79.308	83.658	81.711
30	0	0	0	0	1246.9	1295.9	1264.77

Table 1: Results of Hill Climbing First-Improvement on De Jong's function

3.1.3 Hill Climbing Best-Improvement results

Dimension	Min	Max	Mean	σ	Min Time	Max Time	Mean Time
5	0	0	0	0	23.91	26.04	25.394
10	0	0	0	0	142.345	152.581	146.898
30	0	0	0	0	2251.37	2419.01	2327.996

Table 2: Results of Hill Climbing Best-Improvement on De Jong's function

3.1.4 Simulated Annealing results

Dimension	Min	Max	Mean	σ	Min Time	Max Time	Mean Time
5	0	0	0	0	25.688	27.604	26.745
10	0	0	0	0	36.831	40.293	38.251
30	0	0	0	0	63.315	67.667	65.270

Table 3: Results of Simulated Annealing on De Jong's function

3.1.5 Execution time comparison

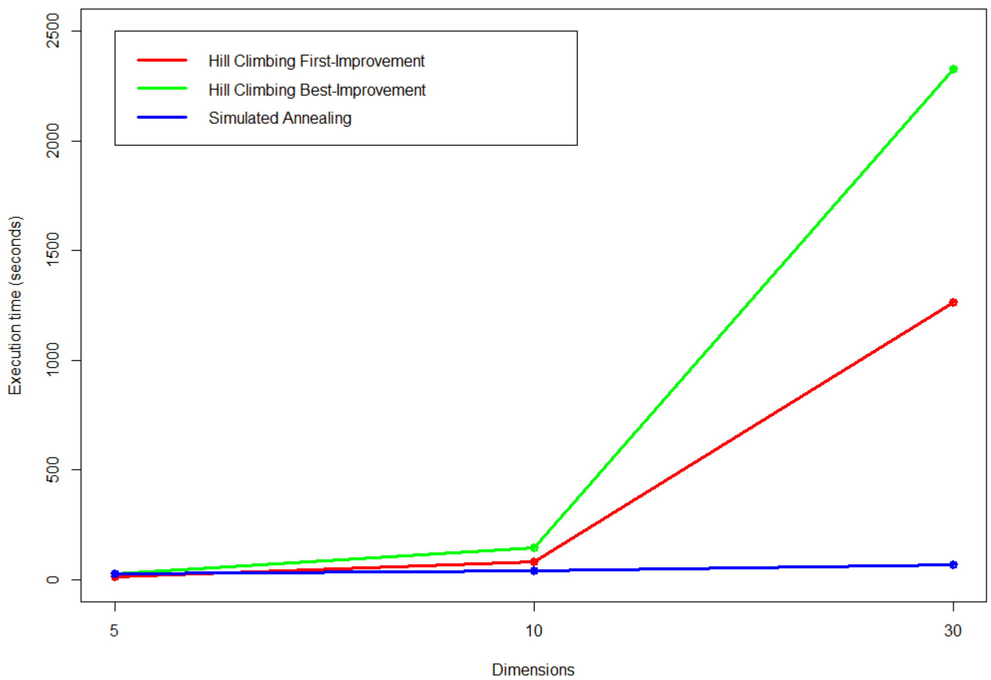

Figure 2: De Jong's function execution time comparison

3.2 Schwefel's function

3.2.1 The function

$$f(x) = \sum_{i=1}^{n} -x_i \cdot \sin(\sqrt{|x_i|}) \qquad -500 \le x_i \le 500$$

Global minimum: $f(x) = n \cdot 418.9829, \quad x_i = 420.9687, \quad i = 1 : n$

The Schwefel function is complex, with many local minima. **Figure 3** shows the two-dimensional form of the function.[10]

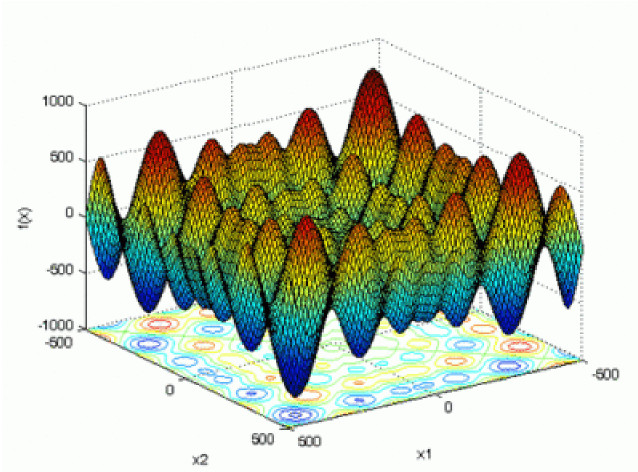

Figure 3: Schwefel's function

3.2.2 Hill Climbing First-Improvement results

Dimension	Min	Max	Mean	σ	Min Time	Max Time	Mean Time
5	-2094.81	-2094.6	-2094.762	0.07217	26.091	29.665	28.209
10	-4155.18	-3939.85	-4058.069	60.05254	160.32	265.784	172.251
30	-11093.4	-10828.4	-10992.85	62.46128	3010.77	4511.38	3142.548

Table 4: Results of Hill Climbing First-Improvement on Schwefel's function

3.2.3 Hill Climbing Best-Improvement results

Dimension	Min	Max	Mean	σ	Min Time	Max Time	Mean Time
5	-2094.91	-2094.81	-2094.893	0.037796	48.636	53.033	50.876
10	-4189.52	-4071.08	-4154.266	20.33704	299.643	311.893	306.550
30	-11794.6	-11341.9	-11487.99	146.2265	5504.15	8247.31	5708.184

Table 5: Results of Hill Climbing Best-Improvement on Schwefel's function

3.2.4 Simulated Annealing results

Dimension	Min	Max	Mean	σ	Min Time	Max Time	Mean Time
5	-2094.71	-1355.47	-1798.275	174.3262	31.666	34.446	33.075
10	-3825.76	-3219.18	-3535.088	171.7948	47.337	50.84	49.266
30	-11786.4	-9869.01	-10904.62	449.3071	94.308	98.106	96.407

Table 6: Results of Simulated Annealing on Schwefel's function

3.2.5 Execution time comparison

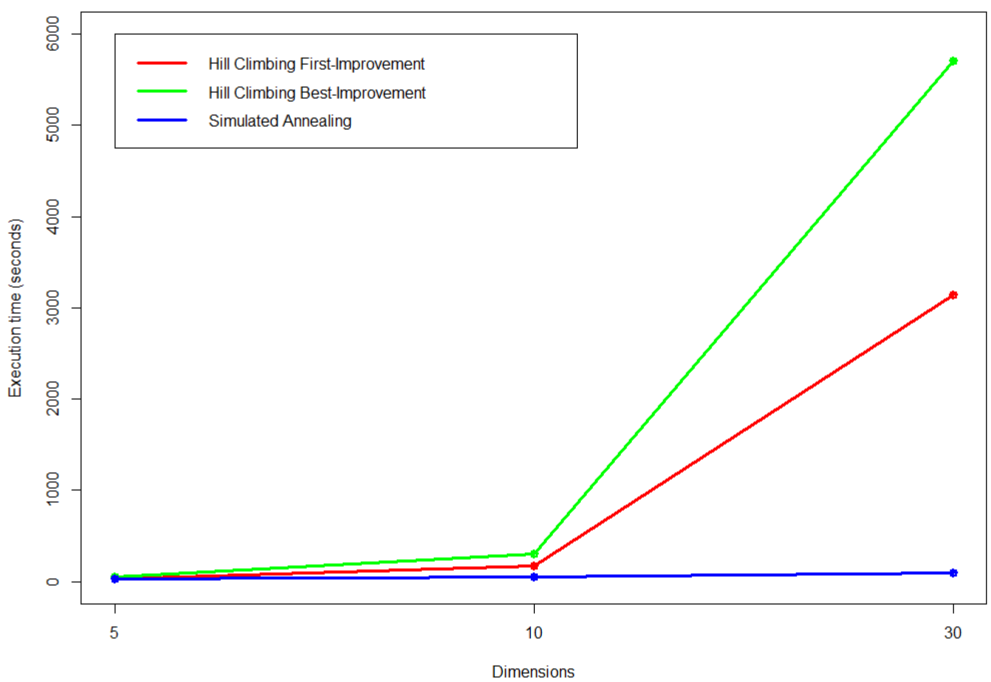

Figure 4: Schwefel's function execution time comparison

3.3 Rastrigin's function

3.3.1 The function

$$f(x) = 10 \cdot n + \sum_{i=1}^{n}(x_i^2 - 10 \cdot \cos(2 \cdot \pi \cdot x_i)) \qquad -5.12 \leq x_i \leq 5.12$$

Global minimum: $f(x) = 0, \quad x_i = 0, \quad i = 1 : n$

The Rastrigin function has several local minima. It is highly multimodal, but locations of the minima are regularly distributed. **Figure 5** shows the two-dimensional form of the function.[11]

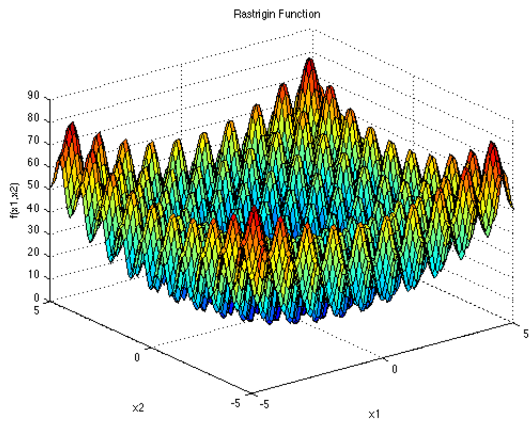

Figure 5: Rastrigin's function

3.3.2 Hill Climbing First-Improvement results

Dimension	Min	Max	Mean	σ	Min Time	Max Time	Mean Time
5	0	0.99495	0.33165	0.47568	11.619	12.888	12.387
10	4.62533	5.27031	5.10906	0.28324	66.207	109.972	72.33739
30	33.1509	34.7911	33.92544	0.83044	1096.78	1206.48	1160.72

Table 7: Results of Hill Climbing First-Improvement on Rastrigin's function

3.3.3 Hill Climbing Best-Improvement results

Dimension	Min	Max	Mean	σ	Min Time	Max Time	Mean Time
5	0	0	0	0	20.057	22.51	21.498
10	3.28039	4.36009	3.97019	0.52596	117.963	132.789	126.104
30	26.2402	26.5776	26.52137	0.12752	1993.99	2135.61	2078.675

Table 8: Results of Hill Climbing Best-Improvement on Rastrigin's function

3.3.4 Simulated Annealing results

Dimension	Min	Max	Mean	σ	Min Time	Max Time	Mean Time
5	1.6402	15.2209	8.65099	4.05877	28.214	30.877	30.154
10	6.61551	28.4615	17.64116	5.92929	41.171	45.386	43.480
30	15.5701	64.0306	36.24808	10.63367	72.064	78.324	75.859

Table 9: Results of Simulated Annealing on Rastrigin's function

3.3.5 Execution time comparison

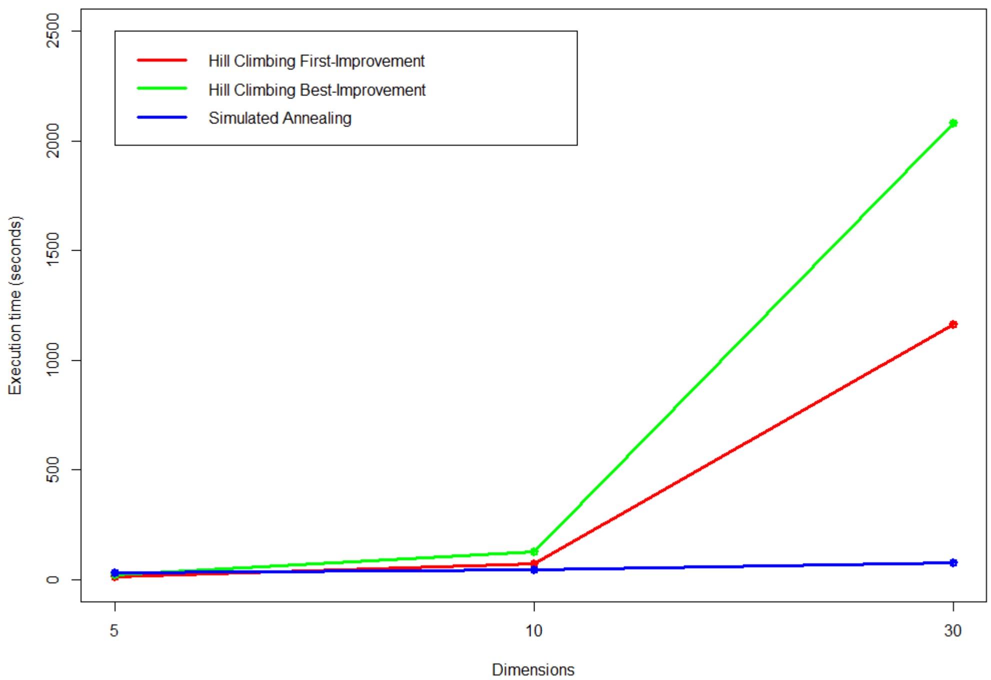

Figure 6: Rastrigin's function execution time comparison

3.4 Michalewicz's function

3.4.1 The function

$$f(x) = -\sum_{i=1}^{n} \sin(x_i) \cdot \left(\sin\left(\frac{i \cdot x_i^2}{\pi}\right)\right)^{2 \cdot m} \qquad i = 1 : n,\ m = 10,\ 0 \leq x_i \leq \pi$$

Global minimum: $f(x) = -4.687$, for n = 5
$f(x) = -9.66$, for n = 10
$f(x) = -29.63$, for n = 30

The Michalewicz function has d! local minima, where d is the number of dimensions, and it is multimodal. The parameter m defines the steepness of the valleys and ridges; a larger m leads to a more difficult search. The recommended value of m is m = 10. The function's two-dimensional form is shown in **Figure 7**.[12]

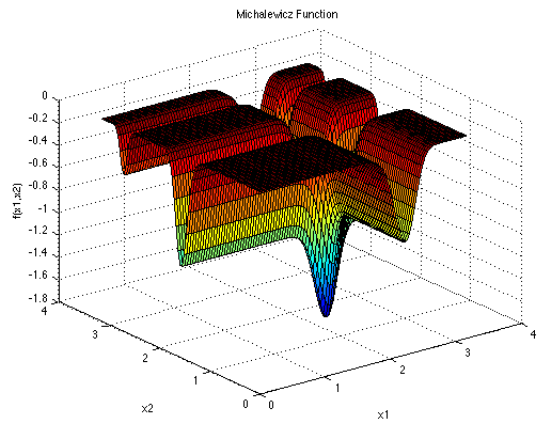

Figure 7: Michalewicz's function

3.4.2 Hill Climbing First-Improvement results

Dimension	Min	Max	Mean	σ	Min Time	Max Time	Mean Time
5	-4.68766	-4.68717	-4.68759	0.0001	15.898	16.84	16.534
10	-9.49385	-9.18631	-9.34286	0.07681	97.989	105.769	102.667
30	-27.0496	-26.4546	-26.84332	0.15172	1858.46	1913.6	1887.912

Table 10: Results of Hill Climbing First-Improvement on Michalewicz's function

3.4.3 Hill Climbing Best-Improvement results

Dimension	Min	Max	Mean	σ	Min Time	Max Time	Mean Time
5	-4.68766	-4.68623	-4.68760	0.00024	27.088	29.173	28.569
10	-9.55645	-9.38976	-9.49635	0.05488	168.477	180.142	176.342
30	-27.6434	-27.136	-27.38289	0.14993	3148.07	3371.13	3288.474

Table 11: Results of Hill Climbing Best-Improvement on Michalewicz's function

3.4.4 Simulated Annealing results

Dimension	Min	Max	Mean	σ	Min Time	Max Time	Mean Time
5	-4.60829	-3.52101	-4.16069	0.27619	39.2	40.931	40.192
10	-9.0896	-7.17672	-8.29631	0.50365	62.219	67.194	64.501
30	-26.8552	-23.5297	-25.52226	0.86358	139.339	197.553	145.406

Table 12: Results of Simulated Annealing on Michalewicz's function

3.4.5 Execution time comparison

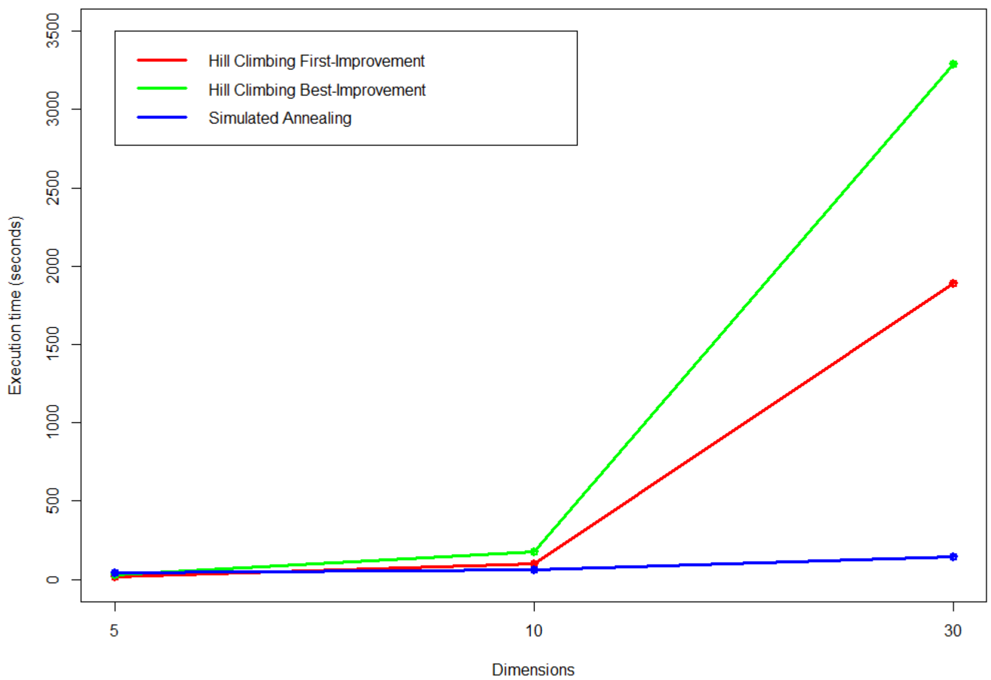

Figure 8: Michalewicz's function execution time comparison

4 Observations

As we can see, **Hill Climbing Best-Improvement** usually provides the best answer, but it also takes the longest to find it. **Hill Climbing First-Improvement** is almost always two times faster than its counterpart but it provides worse answers. These two were actually quite far from the real global minima on **Rastrigin's function**, mostly because it has lots of local minima, but a noteworthy observation can be drawn from that function.
We can see in **Table 13** the difference between the results of **Hill Climbing Best-Improvement** and **Simulated Annealing** on **Rastrigin's function**, on size **30**.

Approach	Min	Max	Mean	σ	Min Time	Max Time	Mean Time
HCB	26.2402	26.5776	26.52137	0.12752	1993.99	2135.61	2078.675
SA	15.5701	64.0306	36.24808	10.63367	72.064	78.324	75.859

Table 13: Comparison between HCB and SA on Rastrigin's function, size 30

Knowing that **Rastrigin's function's** global minima is **0** we can draw the conclusion that **Simulated Annealing** was more closer than **Hill Climbing Best-Improvement**. Another clear difference is the time one, the first method taking about 27x more time than the second one, and yet providing a worse result. This massive difference is caused by the **temperature** involved in Simulated Annealing. Providing a chance to jump to a "bad" neighbour enlarges the reaches of the search area. While the first method could get stuck in a local minima, the second one has the chance to escape from there. This also explains why σ is way bigger in the second case, its results are more random and more dispersed.

Another interesting observation is the behaviour of the used algorithms on **De Jong's function**. All three of them provided very good results, which, truncated to the used precision, result in the exact global minima of the function. The main difference between them is the time they took. As said before, **Hill Climbing First-Improvement** is about two times as fast as its counterpart, but the third method barely took longer than one minute on the largest size. If we were to choose another precision, there would have been a difference, but with the chosen one they all equal to **0**. This is caused by the selected function having only one real minima, the Hill Climbing algorithms both finding it from the first iteration, thus wasting time with the other iterations that only find the same result.

5 Conclusions

We can draw a few interesting conclusion from the experiments and from the aforementioned observations, mostly regarding their behaviour, their total execution time and their accuracy.

5.1 The behaviour of the algorithms

Based on the results of the three selected methods, we can clearly see that **Simulated Annealing** always has a higher σ. This is because its temperature mechanism allows for a larger area of search, providing more dispersed results. Whilst seemingly undermining the accuracy of the results on certain functions such as **De Jong's**, its approach is extremely well-suited for functions like **Schwefel's** or **Rastrigin's**. Thus we can draw the conclusion that a more "chaotic" behaviour is better used on functions that have a lot of local minima, while methods with a more predictable one, such as the Hill Climbing ones, are better for functions with a few local minima.

It can be seen in **Figure 9** below that Simulated Annealing provides results with a smaller error on **Rastrigin's function** on larger sizes, whilst not being on par with the Hill Climbing algorithms on **Michalewicz's function** and providing the results with the biggest error.

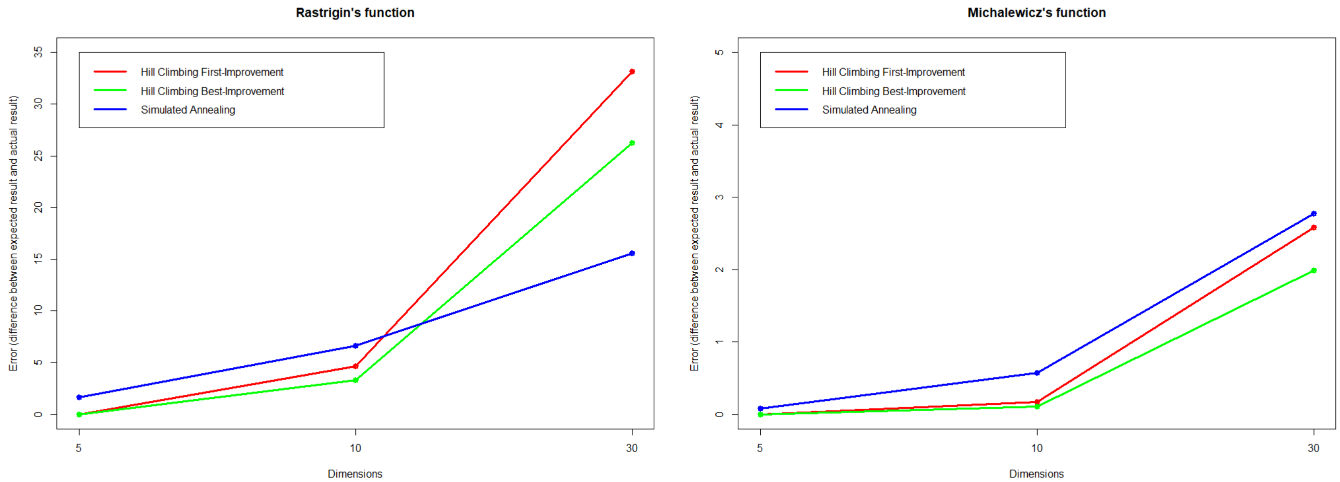

Figure 9: Difference between expected result and actual result in Rastrigin's function and Michalewicz's function

5.2 The time difference

Non-deterministic algorithms are used mainly because they are less time-consuming. Deterministic ones are supposed to be 100% accurate, while taking absurd amounts of time on more complex functions or problems.
Of the four chosen functions, **Schwefel's** was the one to take the most time. **Table 14** shows a time comparison between the used methods on **Schwefel's function**, on size 30. (This is also represented in **Figure 4**)

Approach	Min Time	Max Time	Mean Time
HCB	5504.15	8247.31	5708.184
HCF	3010.77	4511.38	3142.548
SA	94.308	98.106	96.407

Table 14: Time comparison between the three approaches

I believe this table best illustrates the time difference between the three chosen approaches. While **Hill Climbing Best-Improvement** takes the most time, its other variation, **First-Improvement** only takes half as much. While the time these two took can be measured in tens of minutes, even hours, **Simulated Annealing** barely took approximately one minute and a half. The difference of time between Hill Climbing algorithms and Simulated Annealing, with the chosen conditions, is immense, and as seen before the difference between the results does not really justify the time difference.

5.3 The accuracy

As big of a difference as there is between the time it takes for each method to find the minima, the accuracy greatly differs. In **Table 15** we can see the results for **Schwefel's function**, on size 30, provided by each approach.

Approach	Min	Max	Mean	σ
HCF	-11093.4	-10828.4	-10992.85	62.46128
HCB	-11794.6	-11341.9	-11487.99	146.2265
SA	-11786.4	-9869.01	-10904.62	449.3071

Table 15: Comparison of results between the three approaches

By looking only at the "Min" column it's easy to see that the methods provide similar results, **Hill Climbing First-Improvement** being the farthest from the actual global minima. The main difference between the three methods used is the **Standard Deviation(σ)**. The last column clearly shows that **Hill Climbing First-Improvement's** results don't vary that much, **Hill Climbing Best-Improvement's** vary a little more and **Simulated Annealing's** vary the most. This can be best observed in **Figure 10**, where the difference between **Min** and **Max** based on the results on **Schwefel's function** is represented.

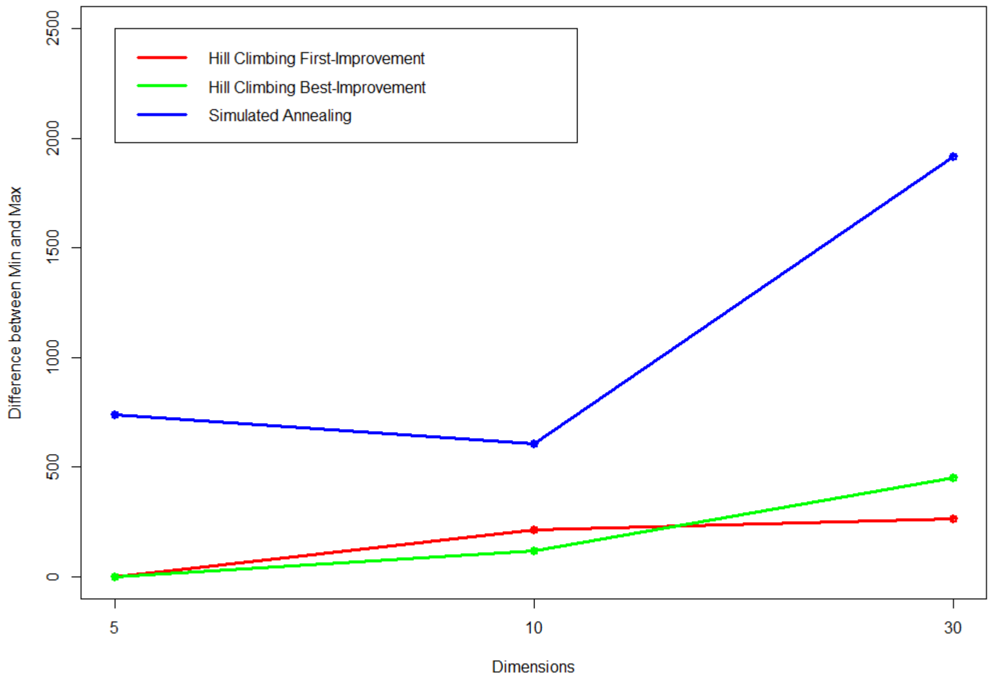

Figure 10: Difference between Min and Max on Schwefel's function

While for the Hill Climbing methods the difference is quite low, for the third method it is quite big, a difference of almost 2000, which is close to 15% of the final result.

In conclusion, for more dispersed results **Simulated Annealing** should be used, and for more congregated results **Hill Climbing** algorithms are more well-suited.

References

[1] Thread information
How to get thread id. https://en.cppreference.com/w/cpp/thread/get_id

[2] Logarithm Calculator
Finding out the number of temperature changes. https://www.rapidtables.com/calc/math/Log_Calculator.html

[3] Hill Climbing information
Information about Hill Climbing and its uses. https://en.wikipedia.org/wiki/Hill_climbing

[4] Pseudocode of Hill Climbing and Simulated Annealing
Main examples of Hill Climbing and Simulated Annealing pseudocodes. https://profs.info.uaic.ro/~eugennc/teaching/ga/#Notions02

[5] Pseudocode of Simulated Annealing
Another example of a Simulated Annealing pseudocode. https://www.researchgate.net/figure/The-pseudo-code-of-simulated-annealing-algorithm_fig2_309537833

[6] Using random
Information on using random numbers in C++. http://www.cplusplus.com/reference/cstdlib/rand/

[7] Measuring time
Information on how to measure time in C++ using clocks. https://en.cppreference.com/w/cpp/chrono/c/clock

[8] String Streams
Information on how to use string streams in C++. http://www.cplusplus.com/reference/sstream/stringstream/

[9] De Jong's function
Information about De Jong's function. http://www.geatbx.com/docu/fcnindex-01.html#P89_3085 De Jong's function image
Image for De Jong's function and further information. https://www.sfu.ca/~ssurjano/spheref.html

[10] Schwefel's function
Information about Schwefel's function. http://www.geatbx.com/docu/fcnindex-01.html#P150_6749 Schwefel's function image
Image for Schwefel's function and further information. https://esa.github.io/pagmo2/docs/cpp/problems/schwefel.html

[11] Rastrigin's function
Information about Rastrigin's function. http://www.geatbx.com/docu/fcnindex-01.html#P140_6155 Rastrigin's function image
Image for Rastrigin's function and further information. https://www.sfu.ca/~ssurjano/rastr.html

[12] Michalewicz's function
Information about Michalewicz's function. http://www.geatbx.com/docu/fcnindex-01.html#P204_10395 Michalewicz's function image
Image for Michalewicz's function and further information. https://www.sfu.ca/~ssurjano/michal.html

[13] Merging text files
Site that merges text files, used for easier processing of the results. https://www.filesmerge.com/merge-text-files

[14] LaTeXinformation
Mathematical symbols in LaTeX. https://oeis.org/wiki/List_of_LaTeX_mathematical_symbols
Floats in LaTeX. https://www.overleaf.com/learn/latex/Positioning_of_Figures
Text formatting in LaTeX. https://www.overleaf.com/learn/latex/bold,_italics_and_underlining
Example of LaTeXcode. https://gitlab.com/eugennc/teaching/-/blob/master/GA/texample.tex
Example of LaTeXexported pdf. https://gitlab.com/eugennc/teaching/-/blob/master/GA/texample.pdf