Homework 2 Report

Bernard Pirlea

December 5, 2019

1 Introduction

1.1 Motivation

As long as humankind exists, we strive for perfection in many areas. We want to reach a maximum degree of happiness with the least amount of effort. In our economy, profit and sales must be maximized and costs should be as low as possible. Therefore, optimization is one of the oldest of sciences which even extends into daily life.

Think of a situation where there are a complex and large set of problems to be solved. Even the most powerful computing systems take a very long time (even years) to come across such difficult problems. In such a situation, Genetic algorithms prove to be an effective tool to provide usable near-optimal solutions in a short amount of time.

1.2 Description of paper

The purpose of this paper is to see how a Genetic Algorithm really finds a better solution based on evolution. Implementing a genetic algorithm to find the minimum value of 4 different functions.

2 Algorithm

2.1 Description

In genetic algorithms, we have a pool or a population of possible solutions to a given problem. These solutions then undergo some genetic operations, producing new children and the process is repeated over various generations. Each individual (or candidate solution) is assigned a fitness value based on an evaluation function value and the fitter individuals are given a higher chance to mate and yield more "fitter" individuals.

This way, we keep "evolving" better individuals or solutions over generations, till we reach a termination criterion.

2.2 Representation

Basic Terminology and its implementation:

Population - It is a subset of all the possible (encoded) solutions for a given problem. Simply, this is the set of individuals and each individual is a solution to the problem we want to solve. To represent the population I used a 2 dimensional vector of integers.

Chromosomes A chromosome is one such solution to the given problem. The solution is represented as a vector of bits. Gene A gene is one element of a chromosome. Genes are joined into a vector to form a Chromosome (solution). Either 1 or 0 in our implementation.

Fitness Function - evaluates how close a given solution is to the optimum solution of the desired problem. It determines how fitting a particular solution is. The function we used to determine the fitness is:

$$1.1 * max - f(chromosome)$$

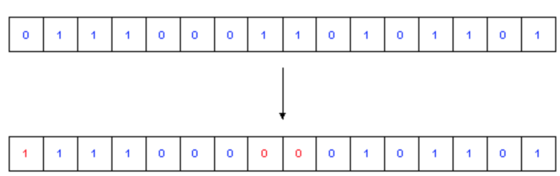
2.3 Selection

During each successive generation, a portion of the existing population is selected to breed a new generation. As for the this I used the Wheel of Fortune in which every individual can become a parent with a probability which is proportional to its fitness. Therefore, fitter individuals have a higher chance of mating and propagating their features to the next generation. Therefore, such a selection strategy applies a selection pressure to the more fit individuals in the population, evolving better individuals over time.

2.4 Operators

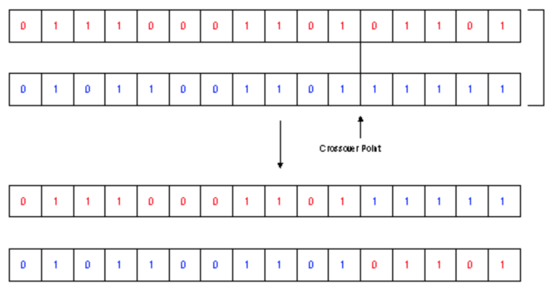
Mutation Modifies one or more genes from a randomly chosen chromosome. The probability of mutation is given by the pm parameter. The number of genes that are mutated is around

$$pm * chromosomelength * popsize$$



Crossover Combines the genes of 2 chromosomes. The probability of mutation is given by the pc parameter. The number of genes that are mutated is around:

$$pc * popsize$$

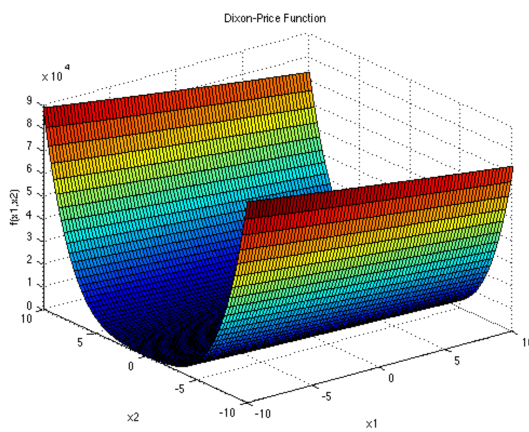


3 Functions

I choose the following functions to test the algorithm:

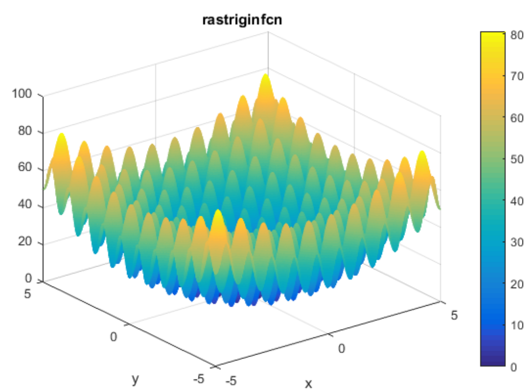
$$f(X) = (x_1 - 1)^2 + \sum_{i=2}^{n} i \left(2x_i^2 - x_{i-1}\right)^2, -10 \leq x_i \leq 10$$

Figure 1: Dixon Price



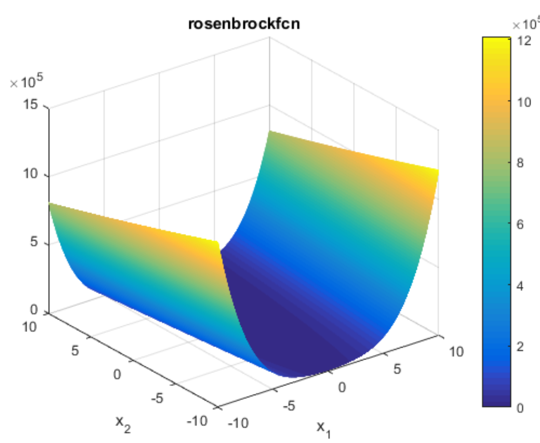
$$f(x) = A \cdot n + \sum_{i=1}^{n} \left[x_i^2 - A \cdot cos(2\pi x_i) \right], A = 10, x_i \in [-5.12, 5.15]$$

Figure 2: Rastrigin's



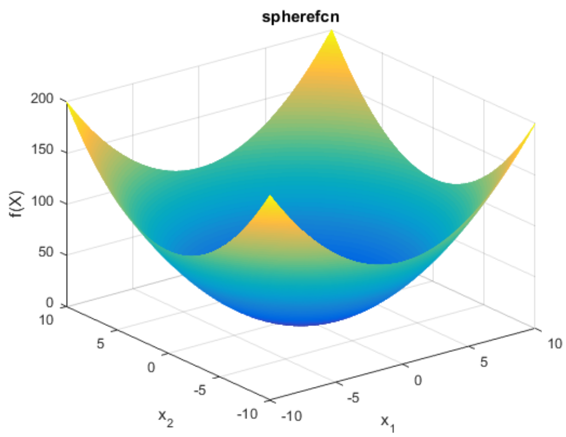
$$f(x, y) = \sum_{i=1}^{n} [b(x_{i+1} - x_i^2)^2 + (a - x_i)^2], -5 \leq x_i \leq 10$$

Figure 3: Rosenbrock



$$f(\mathbf{x}) = f(x_1, x_2, ..., x_n) = \sum_{i=1}^{n} x_i^2, -5.12 \leq x_i \leq 5.12$$

Figure 4: Sphere



4 Experiment

As for the experiment, I executed algorithm 30 times, on a population of 100 individual, over 1000 generations. The pm = 0.01 and pc = 0.25. As I realised that the genetic algorithm tends to converge, on each solution given after I ran a Hill Climbing Improvement. I wrote down the solutions given by the algorithm. From these solutions I took the minimum, the maximum, the average, the median, calculated the standard deviation.

5 Results

alg	min	max	avg	median	σ	time of ex
GA2	0.00097	0.00097	0.00097	0.00097	0.0	38.8144
GA5	0.04572	3.48199	0.80152	0.60288	0.58331	61.9943s
GA10	0.56755	24.411	3.54642	1.35148	5.45373	89.482s
GA30	1.84926	33.755	8.97462	5.2265	7.92448	231.06s

Figure 5: GA on Dixon Price

alg	min	max	media	median	σ	time of ex
GA2	0.02112	0.0661	0.02847	0.02321	0.01074	40.2617s
GA5	0.28685	46.9328	9.61179	5.76548	10.44914	65.1822s
GA10	22.1074	479.168	151.94724	105.185	112.89118	96.8215s
GA30	632.482	11479.5	6241.54073	6216.87	3073.43835	244.315s

Figure 6: GA on Rosenbrock

alg	min	max	media	median	σ	time of ex
GA2	0.0	0.63154	0.15788	0.0	0.1955	41.0101s
GA5	0.31577	2.8673	1.498	1.43365	0.64646	64.4525s
GA10	3.9473	12.7182	7.73859	7.6332	2.36485	96.0841s
GA30	36.3742	75.0731	56.44712	57.17335	10.92745	219.755s

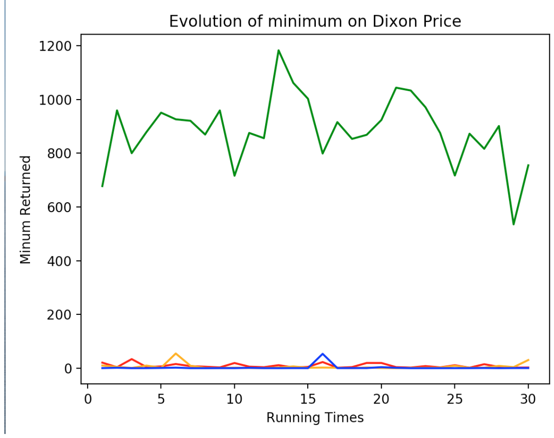
Figure 7: GA on Rastrigin

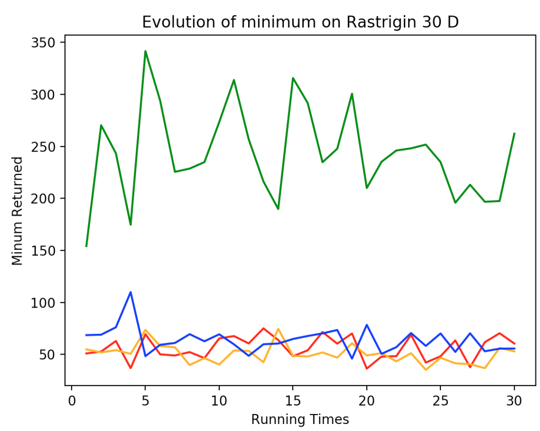
alg	min	max	media	median	σ	time of ex
GA2	0.0	0.0032	0.00096	0.0008	0.00106	38.4289s
GA5	0.0016	0.0064	0.00389	0.0032	0.00169	58.4472s
GA10	0.0048	0.0112	0.00805	0.008	0.00204	93.2865s
GA30	0.016	0.0352	0.02421	0.024	0.00456	220.846s

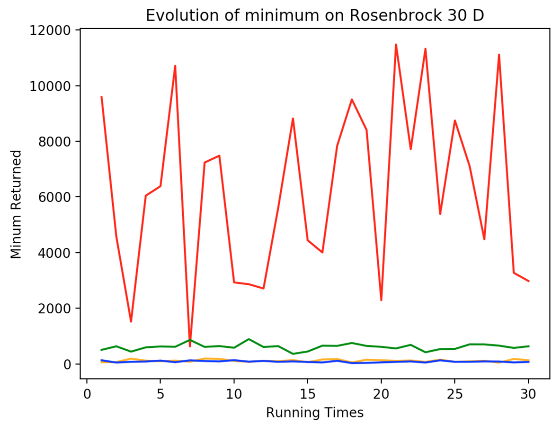
Figure 8: GA on Sphere

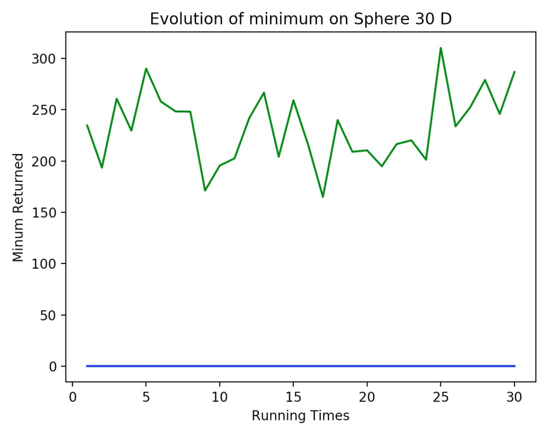
6 Compare

GA - red
HCB - orange
HCF - blue
SA - green









6.1 Influence of parameters

Mutation probability is expected to introduce diversity since it is able to insert new individuals into the population. The higher this rate, the greater the population strongly changes. That will keep shaking things up enough so that other parts of the solution space will be explored and the global optimum can be achieved. The random mutation guarantees to some extent that we see a wide range of solutions. But done too often ruins our algoritm.
Crossover probability provides an exchange of design characteristics between paired individuals. The lower the rate, the less the population is destroyed. A large enough of crossover probability leads to suboptimal solutions. Good use of these parameters increases the chance that the genetic algorithm will find the optimum solution, and improves the value of the best solution found even when the optimum solution is not found.
Population of small size allows an initial faster convergence, but a worse final

result. This can be explained because the quality of final solution needs more population diversity -it depends on the population size- to avoid premature stagnation.

7 Conclusions

I think that overall, the Genetic algorithm did better than the others, looking at the average of the minimum. It is more consistent and gives a better solution in most cases. The results on Rosenbrock may be worse because of the high values of the fitness, the not so well chosen parameters as maybe it needed more time to converge. In many situations GAs locate the neighborhood of the global optimum extremely efficiently but have problems converging onto the optimum itself, that's why I use an Improvement over the minimum, which turned out to do the trick for 3 of the function. The random mutation guarantees to some extent that we see a wide range of solutions. But done too often ruins our algoritm.

References

https://profs.info.uaic.ro/ pmihaela/GA/laborator3.html
Wikipedia
Quora
https://www.phy.ornl.gov/csep/mo/node34.html
http://benchmarkfcns.xyz/
https://medium.com/datadriveninvestor/an-insight-to-genetic-algorithms-part-i-a7f5a5d6d214
http://www.it-weise.de/projects/book.pdf