Differential Evolution and $(1+\lambda)$ Evolution Strategies

Huynh Phuong Nhu
19520208

October 29, 2021

1 INTRODUCTION

In this project, Differential Evolution and Evolution Strategies need to be implemented in order to find the optimized answer for 5 objective functions:

1. f1: Sphere (d = 2 variables and 10 variables).
2. f2: Rastrigin (d = 2 variables và 10 variables).
3. f3: Rosenbrock (d = 2 variables và 10 variables)
4. f4: Griewank (d = 2 variables và 10 variables).
5. f5: Ackley (d = 2 variables và 10 variables).

Then experiment is executed by the change of Population Size ($N or \lambda$), number of variables (d) corresponding to each mentioned above function.
On each modification on $N and d$, random state seeds are tested ten times, and then recorded data such as the history of population, history of population's fitness, etc., are stored in a zipped archive of files. After that, that optimum value received during the process of executing is recorded and calculated the mean and standard deviation.

1.1 Differential Evolution

Differential Evolution (DE) was proposed in 1995 by Storn and Price, which is a population-based metaheuristic search algorithm that optimizes a problem by iteratively improving a candidate solution based on an evolutionary process.
The search for optimized values is done in a stochastic way by applying mutation, crossover, and selection operators to drive the population toward better solutions in the design space. We first create a population by randomly initializing all individuals inside the function search domain. After that, for each individual i in the current population, pick three others that also belong to the population to create a mutant vector v, which is used to crossover with i in order to craft a new offspring o; and if this o vector have better fitness than i, it will completely replace the i vector in the population. Doing the step of looping through every individual until the max number of evaluations is reached or the algorithm has found the optimal value for the objective function.

1.2 Evolution Strategies

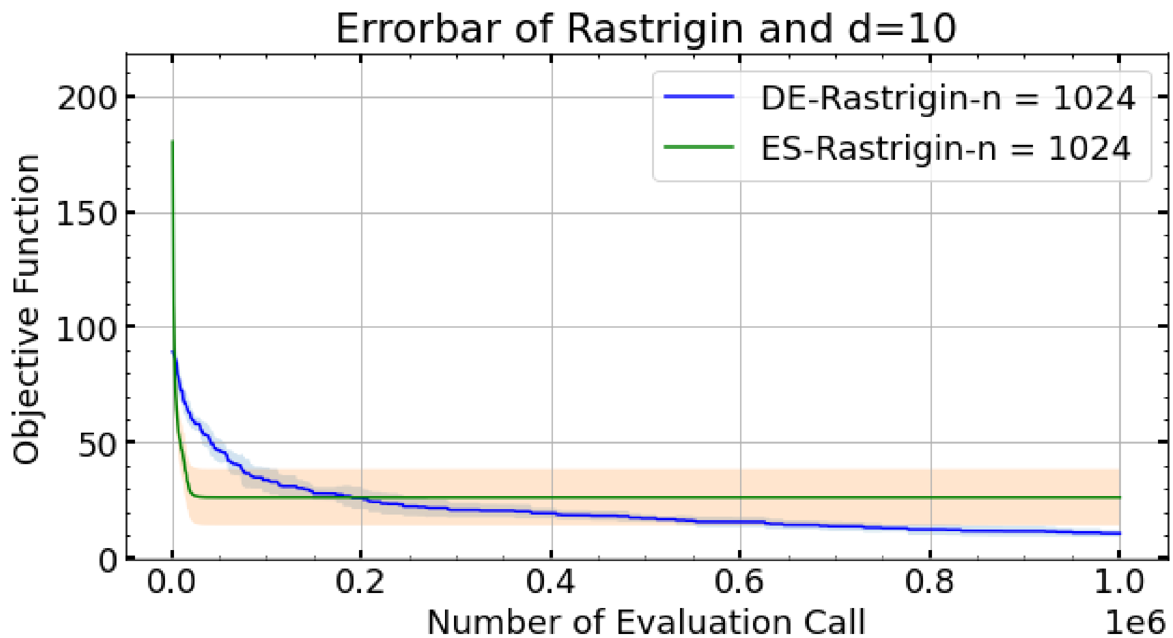
Evolution Strategies was invented in the early 1960s by Rechenberg and Schewefel, which is inspired by natural selection and usually used in solving continuous domain though can be applied in all fields of optimization.
Unlike DE, ES is more focused on the use of mutation rather than a crossover; instead of initializing the entire population, only one individual is created and mutated by two parameters: sigma σ and lambda λ. The number of individuals in the population after this process depends on the type of ES that is used. In this lab, $(1+\lambda)$ES is picked.
The next step, whether how many offsprings or parent vector we are having, only the one that has the best fitness value is kept; if it is an offspring, then this will become the next vector for mutate, then we will increase the search domain by c_{INC} time; vice versa, the search domain will be shrunk.

1.3 Source code and data

The notebook link where I stored and ran the experiment: `https://colab.research.google.com/drive/1Mo4REKtMdIF9Jvf_tGllshn7mMBRtEti?usp=sharing`.
The log and output of all experiments are stored in `https://drive.google.com/drive/folders/1bvRSU3RhMAb8erBo8F9QaKmgTbxT1IYV?usp=sharing`. All the graphs are stored in the Graphics folder, and GIFs are in the gifs folder.

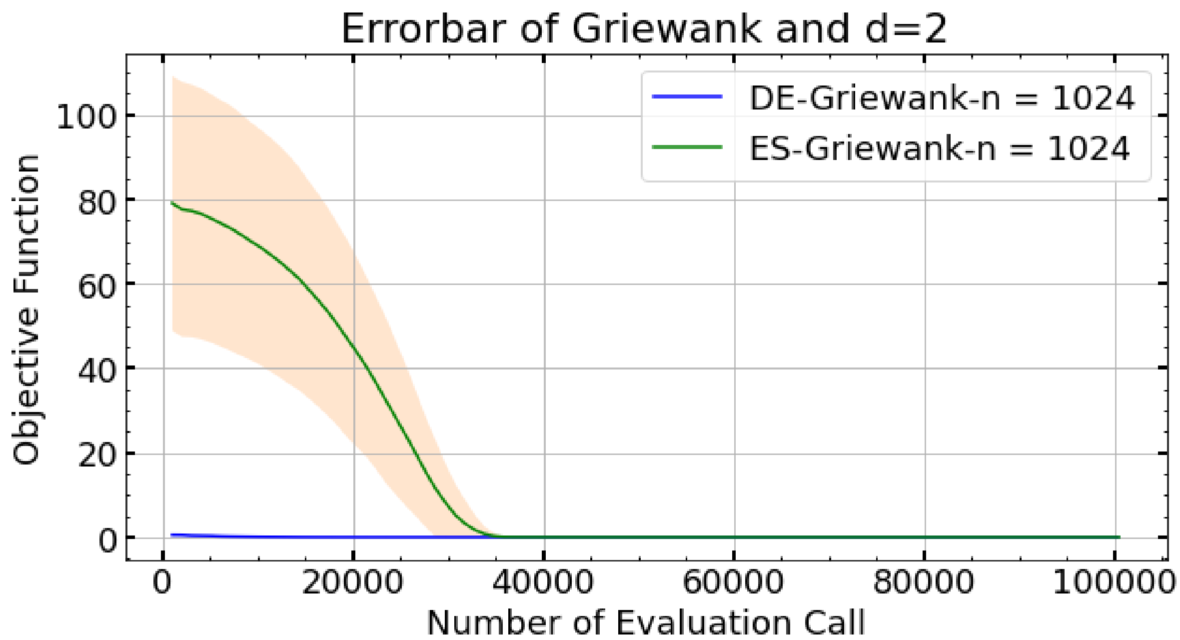
2 EXPERIMENTS

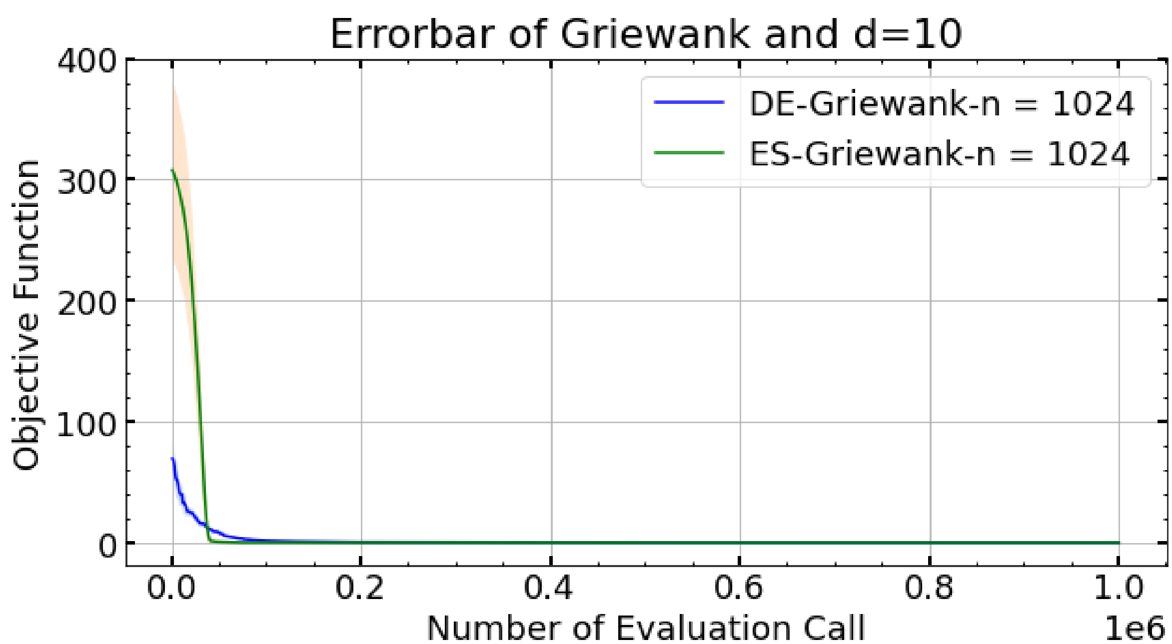
During the time of running the experiment, DE took a lot more time than ES. This may occur due to the fact that in each generation, more vectors need to be created and calculated; the latter depends on the former ones. Whether in the case of ES, only the best individual in the generation is kept and mutated into others.
Regarding the initialization of both algorithms, we notice that DE has a wider spread and covers more area than ES; that is why, most of the time, it is much easier to escape local optima and find the way to the optimum. Although it took a lot less time than DE, ES does not perform well in the case of smaller population size and small sigma when it needs to deal with many local minima function like Griewank, Rastrigin which has quite a similar value at minima.



As in the above case, ES has a better start and drops faster than DE, but it can not get out of the local optima and end up stuck there while DE is almost able to find the optimum value for Rastrigin.

With the implement of early convergence and stop running if the optimal fitness value is obtained, DE use less evaluations than ES and have a better objective function values overall.





Looking at the errorbar for Griewank after implementing the two algorithms, we can clearly see that the value for objective function at the beginning of ES varies a lot because it only takes the fittest individual while the value of DE is the best in the current population.

3 Conclusions

The change in dimensions complicates the problem, the increase in population help ES to find the correct answer. If resources are concerned, and a good approximate solution is needed, then Evolution Strategies is a better choice. But when it comes to the optimal solution or it is worth searching more for the optimal one, Differential Evolution is definitely the measure you should go towards.

References

[1] https://www.frontiersin.org/articles/10.3389/fbuil.2020.00102/full