Object Oriented Genetic Algorithm for N variable optimization

Alessandro Brovelli

November 25, 2024

Abstract

An object oriented genetic algorithm was developed in Python. The scope of the code is to optimize (maximize or minimize) a $N \geq 1$ variable fitness function in a provided search space. It can employ a set of two selection, crossover and mutation methods, set by the user. The algorithm was tested on a 2 dimensional function (Styblinsky-Tang), for which parameters were tuned for optimal accuracy for each combination of settings. Finally, a perturbative analysis was carried out to verify the stability around the optimal parameters. and speed test

1 Algorithm description

1.1 General structure

The algorithm structure is divided in three classes. The first one, *Individual*, defines a single candidate solution, expressed as a genome of N binary strings (*chromosome*) of set length. Each chromosome can be decoded into a decimal value the provided search space. The decoded genome will be the input of the N variable function used to evaluate the individual's fitness. Within itself, an individual can only mutate, given the mutation scheme and probability

A second class *Population* establishes a candidate solutions population (a list of individuals), and contains a method to create it given size, number of chromosomes and genes per chromosome. The other methods in this class allow the population to breed with a certain crossover method and probability, and to mutate by iterating through the individuals.

The main class, *BreedingProgram*, inherits *Population* and defines the settings and procedures of the genetic algorithm. It includes methods to decode each genome, evaluate the population's fitness and to select worthy individuals for breeding. By creating an object, the user can set the problem size and type (maximize or minimize) and the various selection, crossover and mutation parameters and methods, which will be explained in detail in the next section. The whole algorithm can be executed by calling the method *start evolution*, which takes as input the fitness function, the search space, the error threshold and the generation number at which to stop if there is no convergence. If desired, at the end of the evolution the generational best fitness is plotted, alongside the fitness function in the search space with the final population scattered on it (if $N = 1$ or $N = 2$).

1.2 Selection

1.2.1 Tournament

A deterministic tournament selection method was employed. A batch of fixed size is extracted at random from the population, and only the individual with the best fitness is selected. This process is repeated until the selected population reaches the size of the initial one, which is therefore maintained constant through the generations.

1.2.2 Entropy

A more advanced selection is offered by the entropy method, inspired by simulated annealing optimization methods. A fixed dimension batch is still extracted at random from the population, and each individual in it is assigned a selection probability with the so called *Metropolis criterion*, which for a minimization problem will be

$$P(i) = \begin{cases} 1 & F_i \leq F_{old} \\ \exp\left(-\frac{\Delta F}{T}\right) & F_i > F_{old} \end{cases} \tag{1}$$

where F_i fitness of i^{th} individual, F_{old} best fitness of previous generation and $\Delta F = |F_i - F_{old}|$. The parameter T represents the *temperature* and it decreases through the generations with an exponential trend

$$T = T_0 \, \alpha^n \quad 0 < \alpha < 1 \tag{2}$$

This method permits more exploration of the search space in the first genearations, when the temperature is higher, and more exploitation of the region where the solution will converge in the last generations. The selection iteration stops when the selected population reaches its former size or, to prevent being stuck in the loop when the temperature is too low, when a maximum number of batch evaluations is reached. This logic allows the population size to decrease through the generations, by letting unworthy individuals to freeze.

1.3 Crossover

Amongst the selected population, pairs of parents are chosen at random to reproduce. With a set probability they either generate a pair of children or they will live and be part of the next generation.