# Evolution Strategies and Genetic Algorithms

Hadelin de Ponteves

February 10, 2024

**Abstract**

This document provides an in-depth exploration of Evolution Strategies (ES) and Genetic Algorithms (GA), two cornerstone methodologies in the field of evolutionary computation. By dissecting their mathematical underpinnings, operational mechanisms, and application domains, we aim to offer a clear and comprehensive understanding of these algorithms. The discussion includes their optimization objectives, selection processes, and the role of genetic operators in evolving solutions towards optimality.

## Contents

# 1 Introduction

Evolution Strategies (ES) and Genetic Algorithms (GA) are inspired by the biological processes of natural selection and genetics. These algorithms simulate the evolution of individuals in a population towards optimal solutions for complex problems.

# 2 Foundational Principles

## 2.1 Evolution Strategies (ES)

Evolution Strategies focus on optimizing real-valued, continuous functions and rely heavily on mutation and selection as their primary operators.

## 2.2 Genetic Algorithms (GA)

Genetic Algorithms are versatile optimization tools that encode potential solutions to a problem in a chromosome-like data structure, utilizing crossover, mutation, and selection to evolve solutions.

# 3 Mathematical Formulations

## 3.1 Representation

### 3.1.1 ES Representation

In ES, solutions are represented as vectors in the real-valued search space. Each individual is typically denoted as:

$$\mathbf{x} = (x_1, x_2, \ldots, x_n), \tag{1}$$

where $\mathbf{x}$ is a vector representing an individual in the population.

### 3.1.2 GA Representation

In GA, individuals are often represented as binary strings, though other encodings (like real-valued or permutation-based) are also common:

$$\mathbf{x} = (b_1, b_2, \ldots, b_n), \tag{2}$$

where $b_i$ represents the binary genes of the individual.

## 3.2 Fitness Evaluation

The fitness function, $f(\mathbf{x})$, evaluates the quality of each individual, guiding the selection process towards optimal solutions.

### 3.3 Selection

Selection is the process of choosing which individuals pass their genes to the next generation based on their fitness. Mathematically, it can be represented as a probabilistic function that favors individuals with higher fitness:

$$P(\text{selection of individual } i) = \frac{f(i)}{\sum_{j=1}^{N} f(j)}, \tag{3}$$

where $N$ is the population size, and $f(i)$ is the fitness of the $i$-th individual.

### 3.4 Crossover (GA)

Crossover combines the genetic information of two parents to produce offspring. For binary-encoded GAs, a simple one-point crossover can be defined as:

$$\text{child} = \text{crossover}(\text{parent}_1, \text{parent}_2), \tag{4}$$

where parts of the parents' genomes are exchanged at a randomly chosen crossover point.

### 3.5 Mutation

Mutation introduces random changes to individuals, promoting genetic diversity. For real-valued representations (common in ES), mutation might be performed by adding a normally distributed random value:

$$x_i' = x_i + \mathcal{N}(0, \sigma^2), \tag{5}$$

where $x_i'$ is the mutated gene, and $\sigma^2$ is the variance of the normal distribution.

## 4 Algorithmic Procedure

### 4.1 Evolution Strategies Procedure

1. Initialize a population of solutions randomly.
2. Evaluate the fitness of each solution.
3. Select parents based on fitness.
4. Produce offspring through mutation.
5. Optional: Apply recombination among selected individuals.
6. Replace the population with offspring based on survival selection criteria.
7. Repeat until a termination condition is met.

### 4.2 Genetic Algorithms Procedure

1. Initialize a population of binary-encoded solutions randomly.
2. Evaluate the fitness of each solution.
3. Select parents based on fitness.
4. Produce offspring through crossover and mutation operations.
5. Replace the population with offspring based on survival selection criteria.
6. Repeat until a termination condition is met.

## 5 Conclusion

Evolution Strategies and Genetic Algorithms provide robust frameworks for solving optimization problems through mechanisms inspired by natural evolution. While ES emphasizes mutation and continuous parameter spaces, GA is distinguished by its use of crossover and binary representation, making each suitable for different types of optimization challenges.