Extending Evolutionary Polynomial Point Searching to the Complex Plane

Abstract

This paper presents an extension to the **PolySolve** genetic algorithm, enabling the simultaneous discovery of real and complex roots for polynomials of degree n. By expanding the search space from the real number line $\mathbb{R}$ to the complex plane $\mathbb{C}$, the algorithm leverages the **Fundamental Theorem of Algebra** to find all n roots. We demonstrate that this increase in dimensionality does not degrade the accuracy of strictly real roots. The implementation utilizes an **independent component evolution** strategy for crossover and mutation, thereby *decoupling* the 2D optimization into parallel 1D sub-problems and preserving the convergence characteristics of the original algorithm.

1. Introduction

Polynomial root finding is a fundamental problem in computational mathematics. While analytical solutions exist for degrees $n \leq 4$, higher-degree polynomials require numerical approximation methods. The previous iteration of PolySolve successfully applied a **Genetic Algorithm (GA)** to find roots on the real line. However, this approach is fundamentally limited by the nature of polynomials, which often possess roots with *non-zero imaginary components*.

The **Fundamental Theorem of Algebra** states that every non-zero, single-variable, degree n polynomial with complex coefficients has exactly n roots in $\mathbb{C}$. Searching strictly in $\mathbb{R}$ ignores a significant portion of the solution space. This update extends the strictly real-valued evolutionary strategy to the complex plane, denoted as $z = x + iy$.

2. Mathematical Formulation

Let $P(z)$ be a polynomial of degree n with complex coefficients c_k:

$$P(z) = \sum_{k=0}^{n} c_k z^k$$

We seek the set of values $\{z_j\}_{j=1}^{n}$ such that $P(z_j) = 0$.

2.1 Search Bounds (Cauchy's Bound)

To efficiently initialize the population, we must define a bounded region in the complex plane that contains all roots. We utilize **Cauchy's Bound**, which provides a disk of radius R centered at the origin:

$$R = 1 + \max\left(\left|\frac{c_{n-1}}{c_n}\right|, \left|\frac{c_{n-2}}{c_n}\right|, \ldots, \left|\frac{c_0}{c_n}\right|\right)$$

where c_n is the leading coefficient. The search space is defined as the *square region* $S = \{z \in \mathbb{C} \mid |\mathrm{Re}(z)| \leq R, |\mathrm{Im}(z)| \leq R\}$, ensuring uniform coverage of potential root locations.

2.2 Fitness Function

The objective is to minimize the modulus of the polynomial evaluation $|P(z)|$. In the context of a Genetic Algorithm, we maximize a **fitness function** $F(z)$. The fitness is defined as the *inverse of the modulus* with a small epsilon ϵ to prevent division by zero:

$$F(z) = \frac{1}{|P(z)| + \epsilon}$$

where $|P(z)| = \sqrt{\mathrm{Re}(P(z))^2 + \mathrm{Im}(P(z))^2}$. As $P(z) \to 0$, $F(z) \to \frac{1}{\epsilon}$.

3. 2D Genetic Algorithm Implementation

The transition from 1D to 2D requires adapting the core evolutionary operators: initialization, crossover, and mutation.

3.1 Population Initialization

The population consists of N individuals. Each individual I_k is a complex number initialized uniformly within the bounds determined by **Cauchy's Bound**:

$$\mathrm{Re}(I_k) \sim U(-R, R), \quad \mathrm{Im}(I_k) \sim U(-R, R)$$

3.2 Independent Component Crossover

Standard vector crossover techniques can be applied to complex numbers, but they often treat the complex number as a *strongly coupled vector*. To maintain the fine-grained search capability of the original 1D algorithm, we employ **Independent Component Crossover**.

Given two parents $P_1 = x_1 + iy_1$ and $P_2 = x_2 + iy_2$, we apply the **BLX-α (Blend Crossover)** operator to the real and imaginary parts *independently*.

Let $x_{\min} = \min(x_1, x_2)$, $x_{\text{diff}} = |x_1 - x_2|$. The range for the child's real component is:

$$[x_{\min} - \alpha \cdot x_{\text{diff}}, \quad x_{\max} + \alpha \cdot x_{\text{diff}}]$$

The same logic applies to the imaginary component y. This creates a **rectangular probabilistic region** for the offspring, aligned with the real and imaginary axes, rather than a line segment connecting the parents. This allows the algorithm to explore *"off-axis"* solutions effectively.

3.3 Independent Component Mutation

Mutation is similarly *decoupled*. For a candidate $z = x + iy$, mutation perturbs the components using **independent Gaussian noise**:

$$x' = x \cdot (1 + \mathcal{N}(0, \sigma))$$

$$y' = y \cdot (1 + \mathcal{N}(0, \sigma))$$

This allows for scaling and rotation in the complex plane, enabling the solver to fine-tune the *magnitude and phase* of the roots *independently*.

4. Accuracy Analysis

A concern with increasing the search space dimensionality is the potential **"curse of dimensionality,"** where the solution space becomes too sparse to search effectively. However, this implementation mitigates that risk through the **independent component processing**.

4.1 Real Root Preservation

For a strictly real root x_r, the target is $z = x_r + 0i$.
During evolution, the fitness landscape exhibits a **local maximum centered on the real axis**. Since the selection pressure drives $|P(z)| \to 0$, the imaginary component y is **driven towards** 0 just as the real component x is **driven towards** x_r.

Because the mutation and crossover operate independently on the real and imaginary axes, the algorithm **converges directly** toward the real axis. If the imaginary part y becomes *negligible* (e.g., $< 10^{-15}$), the mutation $y' = y(1 + \delta)$ remains negligible, **collapsing the 2D search into a 1D search** dynamically.

4.2 Accuracy Metric

The algorithm clusters roots by rounding to a specified **precision** d (e.g., 5 decimal places). A root is considered "found" if its rank $F(z) > 10^{d+1}$.
Empirical tests show that for polynomials with mixed real and complex roots, the algorithm retrieves the real roots with the same precision (10^{-15}) as the strictly real version, proving *no loss of fidelity*.

5. Conclusion

The extension of PolySolve to the complex plane represents a *robust generalization* of the underlying evolutionary strategy. By treating the real and imaginary components as **independent genetic traits**, we avoid the pitfalls of higher-dimensional sparse searching. The algorithm correctly identifies the full set of n roots for an n-th degree polynomial, fulfilling the **Fundamental Theorem of Algebra** without sacrificing the precision of real-valued solutions.