An Enhanced Genetic Algorithm for Root-Finding in Nonlinear Equation

Pavithra Krishna[1], Sreedeep C D[2]

Abstract

This paper presents an Enhanced Genetic Algorithm (EGA) designed for the efficient computation of both real and complex roots of nonlinear equations. The proposed approach improves upon the conventional Genetic Algorithm by integrating Sobol quasi-random sampling with a region-filtering mechanism that discards non-promising areas of the search space prior to the evolutionary process. This strategy produces a uniformly distributed initial population while substantially shrinking the effective search domain, leading to improved computational efficiency. Extensive numerical experiments on a variety of nonlinear functions show that the Enhanced GA consistently achieves faster convergence than the standard GA and Quasi-GA, without compromising solution accuracy. These numerical results demonstrate that structured initialization combined with domain reduction significantly enhances evolutionary root-finding performance, establishing the Enhanced GA as a robust and efficient tool for solving nonlinear equations.

AMS Subject Classification: 65H04, 65H05

Keywords: Genetic Algorithm, Quasi Genetic Algorithm, Sobol sequence, Region filtering.

1 Introduction

Finding the roots of nonlinear equations of the form

$$f(x) = 0 \tag{1.1}$$

where f maps real variables to complex values, is a fundamental problem in mathematics. Such equations appear in a wide range of scientific and applied fields, including medical and biomedical sciences, where nonlinear models are commonly used to represent physiological systems, pharmacokinetic behaviour, medical imaging processes, and disease progression models. In many practical situations, these functions exhibit challenging features such as multiple roots, rapid or slow variations, oscillatory behaviour, and high sensitivity to small perturbations in input values. Although certain nonlinear problems admit exact solutions through analytical methods, these characteristics often make closed-form solutions difficult or impractical to obtain [2,11]. Consequently, numerical analysis plays a crucial role by offering iterative techniques that systematically compute accurate approximate solutions [2,11]. In medical applications, the availability of efficient and reliable numerical root-finding methods is particularly important for ensuring precision and stability in computational modelling, simulation, and data-driven decision-making.

With the increasing complexity of modern applications and the growing demand for computational efficiency, there is a strong need for algorithms capable of dealing with nonlinear functions that are irregular, highly sensitive, or non-differentiable[4,10]. This has led to significant interest in developing effective numerical techniques for solving equation (1.1). A variety of root-finding methods are available, including classical approaches such as the Newton method and Brent's method [8,4,10]. However, these methods are calculus-based and require a suitably chosen initial approximation to ensure convergence.

[1]Department of Mathematics, Amrita School of Physical Sciences, Amritapuri, India.
[2]Department of Mathematics, Amrita School of Physical Sciences, Amritapuri, India.

To overcome the difficulties associated with classical root-finding methods, several **derivative-free optimization techniques** have been developed, including Genetic Algorithms (GA), Differential Evolution (DE), and Particle Swarm Optimization (PSO) [3]. Among these, the Genetic Algorithm (GA) [11, 5] is a population-based root-finding approach inspired by Darwin's theory of natural selection. It simulates evolutionary processes through genetic operators such as selection, crossover, and mutation to iteratively improve candidate solutions [11]. Unlike traditional numerical methods, GA does not require derivative information and exhibits reduced sensitivity to initial guesses, making it particularly effective for solving complex nonlinear problems [8].

To further improve the efficiency of GA, especially during the initial search phase, an enhanced variant known as the **Quasi Genetic Algorithm (QGA)** has been introduced. In QGA, the initial population is generated using **quasi-random sequences**, such as Sobol sequences, rather than purely random sampling. This strategy produces a more uniform coverage of the search space and enhances population diversity at the outset [7, 6]. As a result, QGA enables more effective domain exploration and often achieves improved convergence behaviour and higher solution accuracy compared to the standard GA [7, 6].

In this work, we propose an Enhanced Genetic Algorithm (Enhanced GA) that integrates a region filtering strategy with quasi-random initialization to improve search efficiency. The search domain is first partitioned into multiple subregions. Within each subregion, Sobol quasi-random sequences are generated to achieve uniform and low discrepancy sampling across the domain. For every subregion, the minimum fitness value is evaluated and compared against a predefined fitness threshold. Subregions whose minimum fitness exceeds this threshold are discarded, as they are considered less promising. Only the Sobol points belonging to the retained subregions are used to construct the initial population. This selective initialization effectively guides the GA to begin its search within more promising areas of the solution landscape, thereby enhancing convergence behaviour and computational efficiency. Moreover, the proposed framework extends to the complex plane, enabling the algorithm to locate complex roots of nonlinear functions as well.

The paper is organized as follows. Section 2 presents the algorithms of the Enhanced GA for computing both real and complex roots. Section 3 provides a comparative analysis of the standard GA, QGA, and the Enhanced GA supported by numerical results demonstrating that Enhanced GA achieves superior time efficiency. It also discusses additional numerical experiments focused on locating complex roots using the Enhanced GA framework. Finally, Section 4 concludes the paper by summarizing key findings and remarks.

2 Algorithm

The following algorithm is designed to identify either real or complex roots of a nonlinear function, depending on the specified search domain. When complex roots are required, the algorithm operates over a region in the complex plane, which is partitioned accordingly. If only real roots are sought, the search domain is restricted to the real line, and the partitioning is performed along the real axis instead.

Algorithm 1 Enhanced Genetic Algorithm

Input: Function $f(z)$; real range $[a_r, b_r]$; imaginary range $[a_i, b_i]$; population size pop_size; maximum generations max_gens; number of subregions n_{sub}; Sobol points per subregion n_{sobol}; fitness tolerance $fitness\_tol$; tolerance tol; crossover probability p_c; mutation probability p_m.
Initialization:
Divide $[a_r, b_r]$ and $[a_i, b_i]$ into n_{sub} equal intervals.
Form n_{sub}^2 rectangular subregions R_{ij}.
for each subregion R_{ij} **do**
    Generate n_{sobol} quasi-random points $z_k \in R_{ij}$ using Sobol sequence.
    Compute fitness values $F_k = |f(z_k)|$.
    if min $(F_k) > fitness\_tol$ then
        Discard R_{ij}.
    else
        Add all z_k to the initial population P_0 .
    end if
end for
for $g = 1$ to max_gens **do**
    for each individual $z_i \in P_g$ **do**
        Compute fitness $f_i = |f(z_i)|$.
    end for
    Select parent pairs (p_1, p_2) using tournament selection.
    With probability p_c, apply crossover to obtain offspring (c_1, c_2).
    With probability p_m, mutate offspring to obtain.(c_1', c_2').
    Form the next generation population P_{g+1}.
    if min $|f(z)| < tol$ **or** $g = max\_gens$ **then**
        Terminate.
    end if
end for
Output: Best solution $z^*$and corresponding fitness $|f(z^*)|$

3 Numerical Examples

In this section, we present six numerical examples to demonstrate the performance of the proposed Enhanced GA. The examples are divided into two categories. The first three examples are used for a comparative numerical analysis of real root computation, as discussed in Section 3.1, where the Enhanced GA is compared with the standard GA and the QGA. The remaining three examples, presented in Section 3.2, focus on the computation of complex roots using the Enhanced GA.

Together, these examples illustrate the effectiveness of the proposed method for both real and complex root-finding problems.

3.1 Comparative Numerical Analysis of GA, QGA, and Enhanced GA

To assess the performance of the proposed Enhanced GA, we examine a collection of test problems and compare its ability to compute real roots against the standard GA and the QGA. Numerical examples are provided to illustrate the improved time efficiency of the proposed method relative to the other two approaches. To facilitate a systematic and fair comparison, we specify the essential parameters that impact the performance of these evolutionary processes in Tables 1 and 2. For all algorithms, the key parameters include the function domain, crossover and mutation probabilities, and the ratio of parents to offspring. Additionally, for the Enhanced GA, we specify the number of subregions into which the search domain is partitioned, the number of Sobol sequence points generated per subregion, and the fitness threshold used for region filtering. Fitness threshold is 0.5 for the first two problem and 0.1 for the third problem.

To ensure consistency across all methods, the overall population size for the GA and the QGA is chosen to match the number of Sobol points retained after filtering in the Enhanced GA. Given the inherent randomness in all three algorithms, the runtime is evaluated by performing 50 independent runs of each method and computing the average runtime. Furthermore, across these 50 runs, the best and worst fitness values are reported to illustrate the peak performance and robustness of each algorithm. Table 3 summarizes the test problems.

Table 1 Experimental Parameter Setup.

Parameter	**Value**
pop_size	400
max_gen	300
Selection	Tournament Selection
p_c	0.3
Crossover	Linear Crossover
Mutation	Random
p_m	0.1
Parents to Offspring Ratio	1:1
tol	10^{-7}

Table 2 Additional Parameter Selection for Enhanced GA.

Parameter	**Value**
n_{sub}	7
n_{sobol}	400
$fitness\_tol$	0.5 and 0.1

Table 3 Test Functions used in this study (adopted from [11, 8]).

Nonlinear Functions	**Domain**	**Solutions**
$f_1(x) = (xe^{2x^2} - sin^2x + 50cosx + 5)^4$	$(-2,2)$	-1.20514063803
$f_2(x) = \frac{2(0.3-x)}{(1+x)^2} - 48$	$(-4,-1)$	-1.25450124821
$f_3(x) = (x^2 - e^{60x} - 3x + 2)^5$	$(-1,1)$	0.01127242773

Example 1. Find the root of the nonlinear function

$$f_1(x) = (xe^{2x^2} - sin^2x + 50cosx + 5)^4$$

in the domain $(-2, 2)$.

Methods	**Time(s)**	**Best fitness**	**Worst fitness**
GA	0.045616	3.678×10^{-15}	9.387×10^{-10}
QGA	0.047331	4.170×10^{-18}	9.257×10^{-10}
EGA	0.0399533	3.097×10^{-19}	9.101×10^{-10}

Example 2. Find the root of the nonlinear function

$$f_2(x) = \frac{2(0.3-x)}{(1+x)^2} - 48$$

in the domain $(-4, -1)$.

Methods	**Time(s)**	**Best fitness**	**Worst fitness**
GA	0.137171	2.206×10^{-11}	9.971×10^{-10}
QGA	0.196351	3.990×10^{-11}	9.904×10^{-10}
EGA	0.128168	5.315×10^{-12}	9.955×10^{-10}

Example 3. Find the root of the nonlinear function

$$f_3(x) = (x^2 - e^{60x} - 3x + 2)^5$$

in the domain $(-1,1)$.

Methods	**Time(s)**	**Best fitness**	**Worst fitness**
GA	0.023365	2.791×10^{-17}	2.114×10^{-2}
QGA	0.024520	1.035×10^{-18}	1.053×10^{-3}
EGA	0.013767	6.740×10^{-18}	9.597×10^{-10}

3.2 Complex Root Computation for Nonlinear Functions

As previously discussed, the proposed Enhanced GA framework can be naturally extended to compute complex roots of nonlinear functions. This section presents numerical experiments that validate this capability. All algorithmic parameters remain as specified in the preceding section, except for the additional Enhanced GA specific parameters. In the examples below, the domain is partitioned into 15 subregions (n_{sub} = 15). The fitness tolerance is set to 2.3 for the first two problems and 0.1 for the third. The Sobol sample size generated per subregion (n_{sobol}) is chosen as 200 for the first problem and 400 for the remaining two, ensuring stable performance across all test cases. As in the previous section, Table 4 provides an overview of the test functions and the best complex root obtained over their respective domains using the Enhanced GA.

Table 4 Test Functions used in this study (adopted from [9]).

Nonlinear Functions	**Domain**	**Solutions**
$f_4(x) = x^3 + 3x^2 + 24x + 364$	$(1,3), (6,8)$	$1.99999999999 + 6.92820323027j$
$f_5(x) = x^3 - 9x + 28$	$(0,4), (-4,0)$	$2.00000000000 - 1.73205080752j$
$f_6(x) = x^3 + 4x^2 - 10$	$(-3,0), (-3,0)$	$-2.68261500667 - 0.35825935991j$

Example 4. Find the complex root of the nonlinear function

$$f_4(x) = x^3 + 3x^2 + 24x + 364$$

Where the real range is $(1,3)$ and the imaginary range is $(6,8)$.

Method	**Time(s)**	**Best fitness**	**Worst fitness**
EGA	0.213324	1.802×10^{-10}	9.870×10^{-4}

Example 5. Find the complex root of the nonlinear function

$$f_5(x) = x^3 - 9x + 28$$

where the real range is $(0,4)$ and the imaginary range is $(-4,0)$.

Method	**Time(s)**	**Best fitness**	**Worst fitness**
EGA	0.189225	7.480×10^{-11}	5.529×10^{-6}

Example 6. Find the complex root of the nonlinear function

$$f_6(x) = x^3 + 4x^2 - 10$$

where the real range is $(-3,0)$ and the imaginary range is $(-3,0)$.

Method	**Time(s)**	**Best fitness**	**Worst fitness**
EGA	0.195290	9.416×10^{-11}	6.411×10^{-5}

4 Conclusion

This work presents an enhanced genetic algorithm for computing the roots of nonlinear equations. The proposed modification focuses on the initialization phase, where the search domain is partitioned into subregions, Sobol quasi-random points are generated within each subregion, and regions with low potential are filtered out prior to evolution. Extensive numerical experiments using a range of nonlinear test functions demonstrate that the Enhanced Genetic Algorithm outperforms existing evolutionary approaches in terms of computational efficiency for real-valued problems, with performance improvements depending on the number of subregions and the density of Sobol sampling. Furthermore, the method reliably identifies complex roots across all tested problems, confirming its robustness in the complex domain. Overall, the results establish the proposed approach as an efficient and dependable framework for solving nonlinear equations with real and complex solutions, including those encountered in **computational models relevant to medical and biomedical applications**, where accuracy and efficiency are of critical importance.

References

[1] Abbas, Basim. (2023). Genetic Algorithms for Quadratic Equations. Journal of Electronics, Computer Networking and Applied Mathematics. 36-42. 10.55529/jecnam.35.36.42

[2] Argyros, I. K., Computational Theory of Iterative Methods, Elsevier, 2007.

[3] Gong, W., Liao, Z., Mi, X., Wang, L., & Guo, Y. (2021). Nonlinear equations solving with intelligent optimization algorithms: A survey. Complex System Modelling and Simulation, 1(1), 15-32.

[4] Gouri M Krishna, & Sreedeep C D. (2026). An Improved Secant Type Method for Solving Nonlinear Optimization Problems. https://doi.org/10.5281/zenodo.18213827

[5] Haupt, R. L., & Haupt, S. E. (2004). Practical genetic algorithms. John Wiley & Sons.

[6] Lei, G. (2002). Adaptive random search in quasi-Monte Carlo methods for global optimization. Computers & Mathematics with Applications, 43(6-7), 747-754

[7] Maaranen, H., Miettinen, K., & M¨akel¨a, M. M. (2004). Quasi-random initial population for genetic algorithms. Computers & Mathematics with Ap plications, 47(12), 1885-1895.

[8] Shan, X. K., & Bremser, K. (2019). Find a Function Root by Genetic Algorithm (Paper 3099-2019). In Proceedings of the SAS® Global Forum 2019 Conference. SAS Institute Inc.

[9] Siwach, A., & Malhotra, R. (2024). Complex Roots-Finding Method of Nonlinear Equations. Contemporary Mathematics, 2848-2857.

[10] Sreedeep, C. D., Argyros, I. K., Gopika Dinesh, K. C., Shrestha, N., and Argyros, M., Convergence analysis of a power series-based iterative method having seventh-order convergence, Matematychni Studii, 64(2), 179–193, 2025.

[11] Zafar, F., Cordero, A., Mujtaba, S., & Torregrosa, J. R. (2025). A Hybrid Steffensen Genetic Algorithm for Finding Multi-Roots of Nonlinear Equations and Applications to Biomedical Engineering. Algorithms, 18(9), 582