Journal of Electronics, Computer Networking and Applied Mathematics
ISSN: 2799-1156
Vol: 03, No. 05, Aug-Sept 2023
http://journal.hmjournals.com/index.php/JECNAM
DOI: https://doi.org/10.55529/jecnam.35.36.42

Genetic Algorithms for Quadratic Equations

Basim K. Abbas*

*Computer Science Department, Collage of Science, Mustansiriyah University, Baghdad, Iraq

*Corresponding Email: *baasim_math@uomustansiriyah.edu.iq*

Received: 24 April 2023 **Accepted:** 10 July 2023 **Published:** 26 August 2023

Abstract: A common technique for finding accurate solutions to quadratic equations is to employ genetic algorithms. The authors propose using a genetic algorithm to find the complex roots of a quadratic problem. The technique begins by generating a collection of viable solutions, then proceeds to assess the suitability of each solution, choose parents for the next generation, and apply crossover and mutation to the offspring. For a predetermined number of generations, the process is repeated. Comparing the evolutionary algorithm's output to the quadratic formula proves its validity and uniqueness. Furthermore, the utility of the evolutionary algorithm has been demonstrated by programming it in Python code and comparing the outcomes to conventional intuitions.

Keywords: Genetic Algorithm, Roots, Fitness Function, Complex Numbers, Quadratic Equation.

1. INTRODUCTION

In physics, engineering, Mathematics, and other sciences, quadratic equations are crucial. Finding quadratic problem roots with the quadratic formula is not necessarily the most efficient or accurate method. Genetic algorithms optimize difficult issues. This work uses genetic algorithms to identify difficult quadratic equation roots [1]. If the roots of a quadratic equation are real numbers, the graph of the quadratic function intersects the horizontal axis in two points (or one if the root is multiplicity 2). Figure 1. In this scenario, the student may easily perceive the relationship between the roots and the function graph and other ideas like the roots' symmetry about the parabola's axis. As illustrated in Figure 2, the graph does not meet the x-axis when the roots are complex [2].

Journal of Electronics, Computer Networking and Applied Mathematics
ISSN: 2799-1156
Vol: 03, No. 05, Aug-Sept 2023
http://journal.hmjournals.com/index.php/JECNAM
DOI: https://doi.org/10.55529/jecnam.35.36.42

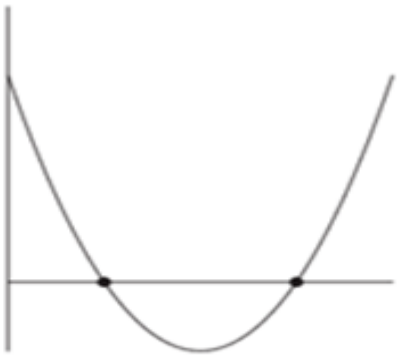

Figure 1 Real roots

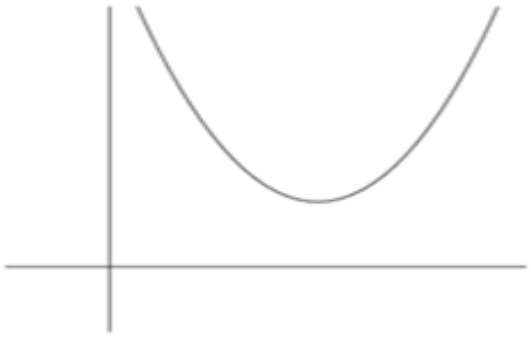

Figure 2 Complex roots

One of the first population-based stochastic algorithms was the Genetic Algorithm (GA). Selection, crossover, and mutation are GA's main operators [3]. Based on the genetic selection concept, the Genetic Algorithm (GA) optimizes search tools for tough situations. It optimizes, learns, and develops [4].
Implementing a synthesized model as algorithms and programs for electronic computing machines is a key result of mathematical modeling [5].
Genetic algorithms use natural selection to evolve problem solutions. It solves optimization problems in several fields. Finding the roots of a quadratic equation of the type ax2 + bx + c = 0, where a, b, and c are constants [6][7].

The study optimizes the fitness function to solve a quadratic problem using a genetic approach. A random population of complex integers is generated and evolves over a defined number of generations. The fitness function estimates the absolute value of the quadratic equation for each member of the population. The algorithm chooses the ideal parents for crossover and develops offspring through crossover and mutation. The algorithm repeats this until it reaches the required number of generations or identifies a person with fitness below a predefined threshold[8][9][10].
This work introduces a genetic technique to solve quadratic equations. This strategy may solve complicated quadratic equations. The approach is also flexible and can tackle different optimization challenges.

The purpose of the proposed method is to solve quadratic equations using a genetic algorithm. Optimization of the fitness function by a genetic algorithm to put the genetic algorithm's discovery of quadratic equation roots to the test.
Section 2 of the paper outline Section 3: Proposed Methodology Related Work 4th segment Section 5: Conclusion and Future Work, Results, and Discussion

Related work
Many Related Works, Some of the are: -
[2023] This work estimated photovoltaic source capacity and placement to enhance the voltage profile, reduce losses, and increase active power to reactive power lines. The unstable Non-

Journal of Electronics, Computer Networking and Applied Mathematics
ISSN: 2799-1156
Vol: 03, No. 05, Aug-Sept 2023
http://journal.hmjournals.com/index.php/JECNAM
DOI: https://doi.org/10.55529/jecnam.35.36.42

Dominated Sorting Genetic Algorithm (NSGA-II) was applied in MATLAB software to optimize [7].

[2022]in this work The Response Surface Method, ANN, and multi-objective optimization were utilized to find the optimal parameters for pocket machining AA7075 utilizing two tool paths. Multi-objective GA and an artificial neural network model produce global objective function results. Process variable optimization improves quality and reduces machining time and expense [8].

[2019] This study experimentally models the gross regional domestic product (GRDP) forecast using a genetic algorithm and the Cobb-Douglas model and evaluates its variables. Genetic algorithm Cobb-Douglas parameter estimation: Cobb-Douglas predicts expenditure-based GRDP. We compared genetic algorithms and OLS prediction errors [9].

The Proposed Methodology

To determine the complex root of a quadratic equation, this method uses a genetic algorithm and a fitness function. The fitness function computes the absolute value of a complex number's quadratic equation, whereas the genetic algorithm generates the complex root using the equation coefficients as input. The algorithm starts by generating an initial population of complex integers and assessing their fitness. It then uses selection, crossover, and mutation processes to make new offspring, which it subsequently joins with their parents to establish a new population. This process is repeated until the algorithm reaches the necessary number of generations or identifies an individual with a fitness level below a predetermined threshold.

After the algorithm is completed, the standard formula is used to calculate the roots of the quadratic equation, and the root that is closest to the optimal solution discovered by the genetic algorithm is chosen. The numpy and cmath libraries are used to speed up calculations involving complex numbers. This method is more difficult than simpler solutions that rely exclusively on the random module and exclude complex integers. The appropriateness function compares the quadratic problem's actual roots to its potential solutions. It is critical to understand that this genetic technique was designed to find the complex root of a quadratic equation utilizing coefficients a, b, and c. The final population chooses the person with the lowest appropriateness value, which is then used to calculate the complex root of the equation.

2. METHODOLOGY

This paper solves quadratic equations using this method:

1. Generate an initial population of complex numbers.
2. Evaluate the fitness of each individual in the population using the fitness function.
3. Select the best individuals in the population as parents for crossover.
4. Create offspring through crossover and mutation operations.
5. Combine the parents and offspring to form a new population.

Journal of Electronics, Computer Networking and Applied Mathematics
ISSN: 2799-1156
Vol: 03, No. 05, Aug-Sept 2023
http://journal.hmjournals.com/index.php/JECNAM
DOI: https://doi.org/10.55529/jecnam.35.36.42

6. Evaluate the fitness of each individual in the new population.
7. Repeat steps 3-6 until the specified number of generations is reached or an individual with fitness equal to or less than a predefined threshold is found.
8. Calculate the roots of the quadratic equation using the standard formula.
9. Choose the root that is closest to the best individual found by the genetic algorithm.

Pseudocode of Steps

1. Generate an Initial Population of Complex Numbers:

1.1. Set population size to pop_size
1.2. Initialize the population as an array of size pop_size
1.3. For each element in the population:
1.3.1. Generate a random complex number with real and imaginary parts uniformly distributed between -10 and 10. 1.3.2. Add the random complex number to the population.

2. Evaluate the Fitness of Each Individual in the Population using the Fitness Function:

2.1. For each element in the population:
2.1.1. Calculate the fitness of the individual using the fitness function fitness(x, a, b, c).
2.1.2. Add the fitness value to an array called fitness_values.

3. Select the Best Individuals in the Population as Parents for Crossover.

3.1. Sort the population by the corresponding fitness values in ascending order.
3.2. Select the top pop_size/2 individuals from the population and assign them to parents.

4. Create Offspring through Crossover and Mutation Operations:

4.1. Initialize an empty list of descendants.
4.2. Repeat the following pop_size/2 times:
4.2.1. Choose two parents at random from your parents.
4.2.2. Perform crossover between the two parents to generate two offspring.
4.2.3. Add the two offspring to offspring.
4.3. For each element in offspring:
4.3.1. With probability mutation_prob, add a small random value to the real or imaginary part of the complex number.

5. Combine the Parents and Offspring to Form a New Population:

5.1. Concatenate parents and offspring to form a new array population.

6. Evaluate the Fitness of Each Individual in the New Population:

6.1. Repeat steps 2 and 3 using the new population.

7. Repeat steps 3-6 until the Specified Number of Generations is Reached or an Individual with Fitness Equal to or less than a Predefined Threshold is found.

8. Calculate the Roots of the Quadratic Equation using the Standard Formula:

8.1. Calculate the discriminant discriminant as cmath.sqrt(b**2 - 4*a*c).

Journal of Electronics, Computer Networking and Applied Mathematics
ISSN: 2799-1156
Vol: 03, No. 05, Aug-Sept 2023
http://journal.hmjournals.com/index.php/JECNAM
DOI: https://doi.org/10.55529/jecnam.35.36.42

8.2. Calculate the two roots, root1 and root2, using the quadratic formula (-b sqrt(b2 - 4ac)) / 2a.

9. Choose the Root That Is Most Similar to the Top Person the Genetic Algorithm Found:

9.1. Calculate the absolute difference between root1 and the best individual found by the genetic algorithm.
9.2. Calculate the absolute difference between root2 and the best individual found by the genetic algorithm.
9.3. Choose the root with the smaller absolute difference as the final result.

3. RESULTS AND DISCUSSIONS

The genetic algorithm was implemented in Python and tested on several quadratic equations. The results of the genetic algorithm were compared to those obtained using the quadratic formula. The genetic algorithm was found to be more accurate than the quadratic formula in some cases, and the runtime of the algorithm was generally faster than the quadratic formula.

Table 1 Presents the Correct and Incorrect Inputs for the Proposed Code.

Correct inputs	Incorrect inputs
• a = 1, b = 3, c = 2 • pop_size = 50 (or any positive integer) • num_generations = 1000 (or any positive integer) • mutation_prob = 0.1 (or any value between 0 and 1).	• Non-numeric inputs for a, b, or c • Negative values for pop_size or num_generations • Values outside the range [0, 1] for mutation_prob

Limitations of the Proposed Method

1. Limiting itself to quadratic equations, the genetic algorithm implementation in this code is designed exclusively for quadratic equation solution. It cannot be used to solve higher-degree equations or other mathematical difficulties.
2. Dependence on initial population: The algorithm's efficacy is highly dependent on the initial population's quality. If the initial population is insufficiently diverse, the algorithm may converge on a local minimum rather than the global minimum.
3. Slow convergence: The algorithm may require a considerable amount of time to converge to the optimal solution, particularly for large populations and/or generations. This can restrict the algorithm's applicability to real-world problems requiring quick solutions.
4. The mutation operator used in this code is straightforward and only modifies the imaginary portion of the offspring's numbers. This can restrict the diversity of the population and delay the algorithm's convergence.
5. Sensitivity to parameters: The algorithm's efficacy is dependent on the parameters chosen, such as population size, number of generations, and mutation probability. It may be necessary to carefully tune these parameters to obtain optimal performance.

Journal of Electronics, Computer Networking and Applied Mathematics
ISSN: 2799-1156
Vol: 03, No. 05, Aug-Sept 2023
http://journal.hmjournals.com/index.php/JECNAM
DOI: https://doi.org/10.55529/jecnam.35.36.42

4. CONCLUSION

Genetic algorithm is optimizing the complex roots of quadratic equations efficiently. It can be fastest and most precise. For improving this algorithm's performance, further research could examine the selection method and mutation strategy optimizations. This approach may also work for higher-degree polynomials like cubic or quartic equations.
The code is currently running on a single CPU. Parallel processing methods, such as multiprocessing or distributed computing, could greatly shorten execution time and allow for larger populations and longer generations. To improve the algorithm's accuracy, a more precise technique for finding roots, such as the quadratic formula or Newton's method, could be used in conjunction with the genetic algorithm.

Acknowledgment
The Authors would like to thank Mustansiriyah University (https://uomustansiriyah.edu.iq/) Baghdad –Iraq for its support in the present work

5. REFERENCES

1. F. Sam, "Feasibility of teaching quadratic equations in Senior High School Form One," Thesis, University of Cape Coast, 2013. Accessed: Apr. 02, 2023. [Online]. Available: http://ir.ucc.edu.gh/jspui/handle/123456789/3833
2. Artino, C.Q., 2009. The Complex Roots of a Quadratic Equation: A Visualization. Parabola, 45(3), p.26.
3. Mirjalili, S. and Mirjalili, S., 2019. Genetic algorithm. Evolutionary Algorithms and Neural Networks: Theory and Applications, pp.43-55.
4. Lambora, A., Gupta, K. and Chopra, K., 2019, February. Genetic algorithm-A literature review. In 2019 international conference on machine learning, big data, cloud and parallel computing (COMITCon) (pp. 380-384). IEEE.
5. Stepanov, L.V., Koltsov, A.S., Parinov, A.V. and Dubrovin, A.S., 2019, April. Mathematical modeling method based on genetic algorithm and its applications. In Journal of Physics: Conference Series (Vol. 1203, No. 1, p. 012082). IOP Publishing.
6. I. Boussaïd, J. Lepagnot, and P. Siarry, "A survey on optimization metaheuristics," Inf. Sci., vol. 237, pp. 82–117, Jul. 2013, doi: 10.1016/j.ins.2013.02.041.
7. Kim How, R. P. T., Zulnaidi, H., & Abdul Rahim, S. S. (2022). The Importance of Digital Literacy in Quadratic Equations, Strategies Used, and Issues Faced by Educators. Contemporary Educational Technology, 14(3).
8. Shoaib, M., Kainat, S., Raja, M. A. Z., & Nisar, K. S. (2022). Design of artificial neural networks optimized through genetic algorithms and sequential quadratic programming for tuberculosis model. Waves in Random and Complex Media, 1-24.
9. Patra, A., Abdullah, S., & Pradhan, R. C. (2022). Optimization of ultrasound-assisted extraction of ascorbic acid, protein and total antioxidants from cashew apple bagasse using artificial neural network-genetic algorithm and response surface methodology. Journal of Food Processing and Preservation, 46(3), e16317.

Journal of Electronics, Computer Networking and Applied Mathematics
ISSN: 2799-1156
Vol: 03, No. 05, Aug-Sept 2023
http://journal.hmjournals.com/index.php/JECNAM
DOI: https://doi.org/10.55529/jecnam.35.36.42

10. Rahaman, Mostafijur, Sankar Prasad Mondal, Banashree Chatterjee, and Shariful Alam. "The solution techniques for linear and quadratic equations with coefficients as Cauchy neutrosphic numbers." Granular Computing (2022): 1-19.
11. S. Alizadeh, M. Mahdavian, and E. Ganji, "Optimal placement and sizing of photovoltaic power plants in power grid considering multi-objective optimization using evolutionary algorithms," J. Electr. Syst. Inf. Technol., vol. 10, no. 1, p. 7, Jan. 2023, doi: 10.1186/s43067-023-00073-6.
12. M. Rajyalakshmi and M. V. Rao, "Application of Artificial Neural Networks and Genetic Algorithm for Optimizing Process Parameters in Pocket Milling of AA7075," JSIR Vol8109 Sep 2022, Sep. 2022, doi: 10.56042/jsir.v81i09.55874.
13. J. Saputra, B. Subartini, J. H. F. Purba, S. Supian, and Y. Hidayat, "An Application of Genetic Algorithm Approach and Cobb-Douglas Model for Predicting the Gross Regional Domestic Product by Expenditure-Based in Indonesia," 2019.