Comparison of Genetic Algorithm and Particle Swarm Optimization in Determining the Solution of Nonlinear System of Equations

Eva Mindasari¹, Sawaluddin²*, Parapat Gultom³

¹) Magister of Mathematics, Universitas Sumatera Utara, Indonesia,
²,³) Department of Mathematics, Universitas Sumatera Utara, Indonesia

¹) evaminda@gmail.com, ²) sawal@usu.ac.id, ³) parapat@usu.ac.id

Submitted : Jun 25, 2024 | Accepted : Jun 25, 2024 | Published : Jul 1, 2024

Abstract: Nonlinear systems of equations often appear in various fields of science and engineering, but their analytical solutions are difficult to find, so numerical methods are needed to solve them. Optimization algorithms are very effective in finding solutions to nonlinear systems of equations especially when traditional analytical and numerical methods are difficult to apply. Two popular optimization methods used for this purpose are Genetic Algorithm (GA) and Particle Swarm Optimization (PSO). This study aims to compare the effectiveness of GA and PSO in finding solutions to nonlinear systems of equations. The criteria used for comparison include accuracy and speed of convergence. This research uses several examples of nonsmooth nonlinear systems of equations for experimentation and comparison. The results provide insight into when and how to effectively use these two algorithms to solve nonlinear systems of equations as well as their potential combinations.

Keywords: Genetic Algorithm, Particle Swarm Optimization, optimization methods, nonlinear equation systems

INTRODUCTION

Nonlinear systems of equations are problems that are quite difficult to solve. Nonlinear equations often cannot be solved by analytical methods, so numerical methods are another solution to solve nonlinear systems of equations. Numerical methods are methods used to formulate mathematical problems so that they can be solved by ordinary calculation operations.

There are many numerical methods that can be used to solve a system of nonlinear equations, as done by (Sunandar & Indrianto, 2020), by comparing the Newton-Raphson and Secant methods. The result obtained is that the Newton-Raphson method produces a more accurate root value compared to the Secant method. In addition, the iterations needed to get a convergent value in the Newton-Raphson method are only 6 iterations, while the Secant method requires 8 iterations.

The Newton-Raphson method is good enough, but it requires the derivative of the function f(xₙ) while not all functions can be found easily. This can be overcome with computational intelligence. Computational intelligent is a research study used in the field of optimization techniques based on intelligent calculation of a structured step (Chu & Tsai, 2007). If the significance value is more than 0.05, then the residual value is thought to follow a normal distribution. If the p-value is less than 0.05, we say that the residual value does not follow a normal distribution. Table 3 displays the results of the normalcy test.

Using optimization algorithms is a very effective way considering that traditional numerical methods such as newton raphson are difficult to apply. This research will use two optimization algorithms namely genetic algorithm and particle swarm optimization to solve the nonlinear system of equations and compare the performance of both.
Optimization algorithms are very effective in finding solutions to nonlinear systems of equations especially when traditional analytical and numerical methods are difficult to apply.

The following are the steps to use optimization algorithms in finding the solution of nonlinear systems of equations:

1) define a system of nonlinear equations
   Suppose we have a system of nonlinear equations as follows:
   \[
   \begin{align*}
   f_1(x_1, x_2, ... , x_n) &= 0 \\
   f_2(x_1, x_2, ... , x_n) &= 0 \\
   \vdots \\
   f_m(x_1, x_2, ... , x_n) &= 0 \\
   \end{align*}
   \]
   the goal is to find the values \((x_1,x_2,...,x_n)\) that make all the above equations zero. but usually in the process of finding a solution we have temporary values for \(x_1,x_2,...,x_n\) that make not all equations zero.

2) form an objective function
   To convert a system of nonlinear equations into an objective function, it is necessary to create a function that measures how far the current solution we have is from the desired solution, namely all equations in the system are zero. this objective function measures the error of the expected solution. the objective function \(F(x_1, x_2, ..., x_n)\) is formed as the sum of the squares of all equations in the system.
   \[
   F(x_1, x_2, ..., x_n) = f_1(x_1, x_2, ... , x_n)^2 + f_2(x_1, x_2, ... , x_n)^2 + \cdots + f_m(x_1, x_2, ... , x_n)^2
   \]
   The objective function will be zero, if and only if \(f_i(x_1,x_2,...,x_n)=0\)

3) use optimization algorithms
   Use an optimization algorithm to minimize the objective function \(F(x_1, x_2, ..., x_n)\). The optimal solution of this objective function is the solution of the nonlinear system of equations.

**Genetic Algorithm** is a heuristic search technique based on the idea or principle of evolutionary processes, natural selection and genetics to solve an optimization problem. Individuals constantly change their genes during the evolutionary process in order to adapt to their environment, where only strong individuals would survive. American mathematician John Holland originally created this technique in 1975 and published it in his book "Adaption in Natural and Artificial Systems".

According to (Abiodun M. et al., 2011), the steps of the Genetic Algorithm can be sequenced as follows:

a. Make the domain of the problem variable as a chromosome of a certain length. Determine the chromosome population size and crossover probability;
b. Define the fitness function, which is used to measure the quality of each chromosome in the problem domain;
c. Generate a random initial chromosome population of a certain length;
d. Calculate the fitness function for each chromosome;
e. Select a pair of chromosomes to mate from the population. The parent chromosomes are selected based on their fitness values;
f. Apply genetic operations (crossover and mutation) to create a pair of daughter chromosomes (offspring);
g. Place the created offspring in the initial population;
h. Swap the initial (previous) population of chromosomes with the new one;
i. Repeat steps d through i until the stopping criterion is met where the specified number of iterations or generations is met.

**Particle Swarm Optimization** is influenced by swarms of fish, insects, or birds that can be found in nature. A swarm's members are able to communicate with one another and cooperate to identify the best course of action. J. Kennedy and R.C. Eberhart first presented this algorithm in 1995. The PSO algorithm's fundamental premise is to mimic the actions of a swarm of particles, each of which has the capacity to solve the optimization issue. The PSO algorithm refers to individual particles within a herd.
Based on both its own and the experiences of other particles in the swarm, each particle navigates the search space. The other particles will follow the path taken by the particle that discovers an efficient way to get to the best answer. (Caesar et al., 2016). There are several factors that make up the PSO algorithm, including the following:

a. The quantity of particles in a population is known as a swarm. The complexity of the problem to be solved determines the size of the swarm.
b. A particle is an individual within a swarm that explains how to address an issue. The optimal solution representation determines the position and velocity of each particle.
c. Personal Best (pBest), which is the particle's best-ever financial position determined by comparing its fitness value to the prior position.
d. The best particle position, or global best (gBest), is determined by comparing the best fitness value of every particle in the swarm.
e. Velocity (velocity), where \( v \) is a vector indicating the particle's direction of motion.
f. The influence of variations in particle velocity is managed by the inertia weight, \( w \).
g. The particle's extent in a single iteration is controlled by the acceleration coefficient. Coefficients \( C_1 \) and \( C_2 \) have the same value, which falls between 0 and 1, generally speaking. For any unique study, you can, however, decide the worth for yourself.

The update equations for the position and velocity of each particle are as follows:

\[
\begin{align*}
    v_i^{t+1} &= w \cdot v_i^{t} + c_1 \cdot r_1 \cdot (pBest_i^{t} - x_i^{t}) + c_2 \cdot r_2 \cdot (gBest_i^{t} - x_i^{t}) \\
    x_i^{t+1} &= x_i^{t} + v_i^{t}
\end{align*}
\]

Where, \( v_i \) is the velocity of particle \( i \) at time \( t \), \( x_i \) is the position of particle \( i \) at time \( t \), \( w \) is the inertial weight used to change the previous velocity of a particle, \( c_1 \) and \( c_2 \) are the cognitive and social scaling parameters, chosen such that \( c_1 = c_2 = 2.0 \) which allows \( c_1 r_1 \) or \( c_2 r_2 \) to have an average of 1, \( r_1 \) and \( r_2 \) are two pickle lines generated from the interval \([0,1]\), \( P_i \) is the best position previously reached by the \( i \)-th particle called pBest, \( P_g \) is the best position reached by all particles called gBest.

**METHOD**

The stage taken in this study to designing the model are as follows:

1. **literature study**
2. **Defining the Problem**
3. **form the objective function**
4. **create a program**
   - Simulation and Implementation of Genetic Algorithm
   - Simulation and Implementation of Particle Swarm Optimization
5. **Analysis of Results**
6. **Conclusion**
RESULT

In this study, several nonlinear systems of equations from the literature were used to demonstrate the performance of swarm optimization algorithms and genetic algorithms and compare the results. Two nonsmooth nonlinear systems of equations are chosen (Luo et al., 2008) as follows:

Example 1 : (Krzyworzcka, 1996)

\[
\begin{align*}
\frac{x_1}{4} + \frac{x_2 x_4 x_6}{4} + 0.75 &= 0 \\
0.405 e^{x_1 x_2} + 1.405 &= 0 \\
\frac{x_3 - x_4 x_6}{2} + 1.5 &= 0 \\
0.605 e^{x_1 - x_5^2} - 0.395 &= 0 \\
\frac{x_5 + x_2 x_6}{2} + 1.5 &= 0 \\
x_6 - x_1 x_5 &= 0
\end{align*}
\]

The above 6-dimensional system is solved by GA and PSO and the solution is shown in Table 1.

<table>
<thead>
<tr>
<th>Solutions</th>
<th>GA</th>
<th>PSO</th>
</tr>
</thead>
<tbody>
<tr>
<td>x_1</td>
<td>-0.914893</td>
<td>-1.002466</td>
</tr>
<tr>
<td>x_2</td>
<td>0.938498</td>
<td>1.002235</td>
</tr>
<tr>
<td>x_3</td>
<td>1.121207</td>
<td>-0.997671</td>
</tr>
<tr>
<td>x_4</td>
<td>0.862838</td>
<td>1.003026</td>
</tr>
<tr>
<td>x_5</td>
<td>-1.049455</td>
<td>-0.998478</td>
</tr>
<tr>
<td>x_6</td>
<td>0.960139</td>
<td>1.001112</td>
</tr>
<tr>
<td>f_1(x)</td>
<td>0.017525</td>
<td>-0.000307</td>
</tr>
<tr>
<td>f_2(x)</td>
<td>-3.101453</td>
<td>0.000333</td>
</tr>
<tr>
<td>f_3(x)</td>
<td>0.035429</td>
<td>0.000257</td>
</tr>
<tr>
<td>f_4(x)</td>
<td>-5.551115</td>
<td>0.000206</td>
</tr>
<tr>
<td>f_5(x)</td>
<td>0.0</td>
<td>-0.000154</td>
</tr>
<tr>
<td>f_6(x)</td>
<td>0.0</td>
<td>0.000170</td>
</tr>
</tbody>
</table>

Example 2: geometry size of thin wall rectangle girder section

\[
\begin{align*}
bh - (b - 2t)(h - 2t) - 165 &= 0 \\
\frac{bh^3}{12} - \frac{(b - 2t)(h - 2t)^3}{12} - 9369 &= 0 \\
\frac{2(h - t)^2(b - t)^2t}{h + b - 2t} - 6835 &= 0
\end{align*}
\]

where b is the width of the section, h the height of the section and t is thickness of the section.(Luo et al., 2008)
Tabel 2. Comparison of results from the application of GA and PSO to example 2.

<table>
<thead>
<tr>
<th>Solutions</th>
<th>GA</th>
<th>PSO</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>12.421091</td>
<td>12.257060</td>
</tr>
<tr>
<td>h</td>
<td>21.721846</td>
<td>22.912299</td>
</tr>
<tr>
<td>t</td>
<td>3.689394</td>
<td>2.779309</td>
</tr>
<tr>
<td>$f_1(b,h,t)$</td>
<td>32.487004</td>
<td>-0.405166</td>
</tr>
<tr>
<td>$f_2(b,h,t)$</td>
<td>0,0</td>
<td>-0.189759</td>
</tr>
<tr>
<td>$f_3(b,h,t)$</td>
<td>0,0</td>
<td>0.098563</td>
</tr>
</tbody>
</table>

Fig. 2. fitness value progression in the GA process in example 1.

Fig. 3. population diversity over iterations in the GA process in example 1.
Fig. 4. PSO convergence history in example 1

Fig. 5. fitness value progression in the GA process in example 2

Fig. 6. population diversity over iterations in the GA process in example 2
DISCUSSIONS

In the example 1, the GA generated solutions obtained accurate results for some functions \( f_i(x) \) in the system such as \( f_4(x) \) and \( f_5(x) \). However, there is instability in the GA in finding solutions. This is different from PSO where the solution tends to stabilize towards a value or the best solution.

Similarly, in example 2, according to (Luo et al., 2008), a nonlinear system of equations can have multiple solutions without considering its physical meaning. The solution obtained using the quasi-Newton method is \( h = 22.8949 \), \( b = 12.5655 \) and \( t = 2.7898 \), not much different from the solutions produced by GA and PSO in this study. GA obtained accurate results in \( f_2(b,h,t) \) and \( f_3(b,h,t) \) but the results were much different from \( f_1(b,h,t) \). This suggests that GA's performance in finding solutions is less stable than PSO's, which tends to consistently achieve convergent results. As far as speed of convergence is concerned, it can be seen from Figure 2-7 that PSO converges faster than GA.

CONCLUSION

Based on the discussion above, it can be concluded that from the two examples of nonlinear systems of equations given, it can be seen that GA tends to be more accurate than PSO. This is attributed to GA's ability to explore a wider search space and of course to overcome the local optimum. Nevertheless, in terms of speed in achieving convergence, PSO is faster. Perhaps this is because PSO uses simpler velocity and position update rules. It can be noted that the performance and stability of both algorithms are highly dependent on the exact parameter configuration, as well as the nature and complexity of the problem at hand.

REFERENCES


*name of corresponding author

