FUDMA Journal of Sciences (FJS)
ISSN online: 2616-1370
ISSN print: 2645 - 2944
Vol. 6 No. 6, December, 2022, pp 202 - 213
DOI: https://doi.org/10.33003/fjs-2022-0606-1114

ESTIMATING NONLINEAR REGRESSION PARAMETERS USING PARTICLE SWARM OPTIMIZATION AND GENETIC ALGORITHM

***[1]Emmanuel Sabastine , [2]Ikechukwu Okoye, [2]Chinenye Pauline Ezenweke, [2]Dolapo Abidemi Shobanke and [2]Isaac Adeola Adeniyi**

[1]Department of Mathematical Sciences, Federal University Lokoja, P.M.B. 1154, Lokoja, Nigeria.
[2]Department of Statistics, Federal University Lokoja, P.M.B. 1154, Lokoja, Nigeria.

*Corresponding authors' email: sabastine.emmanuel@fulokoja.edu.ng

ABSTRACT

Obtaining parameter estimates for nonlinear regression model using Gauss-Newton and gradient-based methods present some complex analytical challenges. In this paper, the effectiveness and simplicity of Particle Swarm Optimization (PSO) and Genetic Algorithm (GA) on five nonlinear regression models with varying level of complexities were investigated. The PSO and GA techniques were implemented for each model in R and the model fittings were performed based on 30 independent runs for at least 100 iterations. The performances of PSO and GA were evaluated based on computation time, residual error. The results obtained showed that PSO significantly outperformed GA in terms of computation time and accuracy of parameter estimates. However, GA required fewer iterations and produced fairly accurate results. Further investigation showed that PSO and GA are both competitive, effective, simple to implement, and can be considered reliable for obtaining the parameter estimates of nonlinear regression models.

Keywords: Regression analysis, stochastic algorithm, least squares estimation, particle swarm optimization, genetic algorithm

INTRODUCTION

Regression analysis is one of the prominent statistical techniques used across disciplines – in science and social sciences – for estimating or predicting the relationship between two or more variables. Generally, regression analysis may be classified as linear and nonlinear regression, and, oftentimes a regression model is expressed as a mathematical equation composed of some unknown parameters β with an error term $\varepsilon \sim N(\mu, \sigma^2)$ (Hsin-Hsiung Huang et al., 2010).
Suppose we specify the model as:

$$y_i = f(x_i, \beta) + \varepsilon, \qquad i = 1 \ldots n \qquad (1)$$

where f is a general function of the predictor variable(s) x_i and parameter(s) β ; y_i is the response variable. If the derivative of f with respect to the parameter(s) does not depend upon one of the $\beta's$, we say that the model is linear in parameter, which yields a widely recognized procedure having a slope and intercept. This form of model is referred to as the classical linear regression model. Here, the unknown parameter estimates can be easily obtained, analytically, using the least square estimation technique. However, in real life processes particularly in fields such as biology, chemistry, logistics and engineering, linear models hardly represent the asymptotic behavior of experimental variables, so, one may need to employ other form of relationship known as the nonlinear regression.
In nonlinear regression, dataset is fitted to a model expressed as nonlinear combination of mathematical function and variables. This form of relationship has a competitive advantage over the linear regression because of its capability to fit a broad range of functions (Sotirios and Fernando, 2015). In other words, model of this form are parsimonious, consistent and can accommodate variety of mean functions such as exponential, logarithms and trigonometry, thereby allowing a more robust curve fitting functionality (Bates and Watts, 2007). Similar to linear models, the concept involved in estimating the coefficients of nonlinear models is relatively the same, however, its non-linear nature makes it more difficult, and most-time impractical to obtain a solution analytically. To circumvent this structural (nonlinear) problem, linearization through logarithm transformation is often utilized. Yet, in some cases, transformation is not feasible (Chicco and Mazza, 2020). What then happens when a nonlinear model cannot be linearized or is intrinsically nonlinear? Researchers are compelled to seek "bailout" approaches – for instance, the use of iterative algorithms – which is one the major challenge of choosing to use the nonlinear least square regression. This inherent problem further contributed to development of numerous estimation algorithms.
In classical iterative algorithms, users are required to provide a starting value for the unknown coefficients of the problem of interest. Although there is no general rule for this selecting this starting values, a good guess is needed in order to obtain reliable results. Deterministic algorithms such as gauss-newton and gradient based methods are the commonly used and well-established practices for estimating the unknown parameter (β) value of nonlinear regression models (Madsen *et al*., 2004). These methods however are limited to a certain class of problems because they depend on stringent details (such as gradient information, matrix-invertibility, accurate guess for initial values) to arrive at reliable estimates (Bulent and Alptekin, 2004). Other limitations include, very slow convergence and definite data representation (Chicco and Mazza, 2020). More specifically, the gradient methods require that the function be differentiable while the gauss-newton methods depend on auxiliary information (often not be available) for optimal performance. In addition, these deterministic algorithms usually get stuck to local optima, instead of locating a global optimum that optimizes an objective function (Friedl and Kuczmann, 2014), thereby resulting to misleading estimates. In fact, an inaccurate guess of the boundaries of the starting values greatly influence the solution time, search speed of the algorithms (Sotirios and Fernando, 2015). Broadly speaking, these algorithms are usually centered around either of the two approaches: The Taylor series and corrections of several parameters at each

iteration on the assumption of local linearity. They attempt to solve the underlying nonlinear least square problem by successive approximations of solution, starting from an initial guess. Sadly, this approach is inconsistent because of divergence in successive iterations.

Eventually, researchers developed a list of modern techniques to overcome the difficulties faced in estimating the unknown parameter value of nonlinear regression models. These alternative methods named meta-heuristic algorithms employ a higher level procedure to provide a sufficiently good solution close enough to the exact solutions with limited assumptions and computation resources (Khanduja and Bhushan, 2021). Most of these algorithms are designed to mimic natural, social or biological activities of living things in order to find best points for an objective function. Kennedy and Eberhart (1995) are one of the early researchers that proposed a powerful method to tackle such problem in nonlinear regression. They introduced the Particle Swarm Optimization (PSO) technique to simulate the choreography of flock of birds or school of fish in search of food. Before Kennedy's PSO, John Holland, his students and colleagues introduced the Genetic algorithm (GA) in 1975 (Holland, 1975). They established the GA to imitate the adaptation and evolution of living things in nature. Holland's GA was designed to move a set of solution from one population to a new population using mechanism based on the theory of "survival of the fittest". Other meta-heuristics algorithms include, fruit-fly optimization, simulated annealing and ant-colony optimization (Desale *et al*., 2015; Rajakumar *et al*., 2016). The PSO and GA are particularly of interest because they are not restrictive to specific problem (Malik *et al*., 2021) and possess promising abilities to narrow down a very large pool of potential solutions to a subset of tangible solutions. More also, PSO and GA are applicable to enhance the estimation accuracy of real life processes (Erdoğmuş and Ekiz, 2016; Sengupta *et al*., 2018; Gupta, 2021). A number of authors (Bulent and Alptekin, 2004; Kapanoglu *et al*., 2007; Malekan and Khosravi, 2018; Ajdad *et al*., 2019; Naidu *et al*., 2018; de Almeida and Leite, 2019; Özsoy & Örkçü, 2016) have reported that PSO and GA are easy to implement because they do not require stringent information for maximum effectiveness.

Over the years, vast literatures have been published in relation to the development, applications and refinement of PSO and GA. More recently, these algorithms were used for various regression (Belhocine *et al*, 2021), classification (Liu R., 2014) and feature selection problems (Ghosh *et al*. 2020). While there is sizeable number of literatures available, to the best of our knowledge only few work have delineated the effectiveness and simplicity of both Particle swarm optimization and Genetic algorithm for unknown parameter estimates of nonlinear regression model in view of computation time, residual errors and curse of objective function. Consequently, this study is devoted to investigate the effectiveness and simplicity of Particle swarm optimization and Genetic algorithm for estimating the unknown parameters of nonlinear regression models. The rest of this paper is organized as follows: the concepts of particle swarm optimization and genetic algorithm are briefly introduced, followed by implementation and analysis of experimental problems; discussion and analysis of results. Lastly, concluding remarks are provided.

METHODOLOGY

Particle Swarm Optimization (PSO)

The Particle Swarm Optimization (PSO) is a population-based optimization algorithm modelled after the social cognitive behavior of flock of bird, bees or a fish school. Originally introduced in the mid-90s by Kennedy, Eberhart and Shi, PSO was built to mimic the foraging of a bird flock searching for food over a landscape, with an objective to learn the principles surrounding their well-organized movements and sudden regrouping along a direction in a search space also called swarm. In this case, each bird or individual in the space is called particles, and represents a potential solution. Each particle navigates through a search space with a dynamic velocity that is influenced by its previous best experience or neighbor (social) experience. With this unique ability, the particles are able to learn and race towards the optimal point by emulating the characteristics of other successful particles in the same region. In addition, each particle enjoys an allocated memory that permit them to remember and update their best position ever encountered. The information sharing mechanism continues throughout the search period until a supposed (optimal) destination is reached. Obviously, the best positions over a number of iterations equals to the optimal fitness value of an objection function.

In PSO algorithm, the process begins with a random population of candidate solution or particles. Each individual solution circulates a multidimensional search space using memorized information that is distributed among members in a swarm. Each member of the swarm is represented by a vector coordinates in a Cartesian plane. Furthermore, the particle vector is assigned a vector that determines its next movement in the search space. Mathematically, Assume we have a d-dimensional search space, we can represent the ith particle at k-iteration in the space by a position vector x_i^k and the particle velocity by v_i^k. Now, consider the best position visited for the ith particle as p_{best}^k, also the overall best position encountered so far as g_{best}. The velocity and the position of each particle is updated through the following simple mathematical formula

$$v_i^{k+1} = w * v_i^k + c_1 * r_1() * \left(p_{best}^k - x_i^k\right) + c_2 * r_2() * \left(g_{best} - x_i^k\right) \quad (2)$$

and

$$x_i^{k+1} = x_i^k + v_i^{k+1} \quad (3)$$

where $c_1 \, and \, c_2$ represents the cognitive and social parameters respectively, $r_1 \, and \, r_2$ are random numbers uniformly distributed within [0, 1]. p_{best}^k is particle k best position and g_{best} is the global best position for all particles. A two-dimensional geometry representation of PSO particles in space is displayed in figure 1

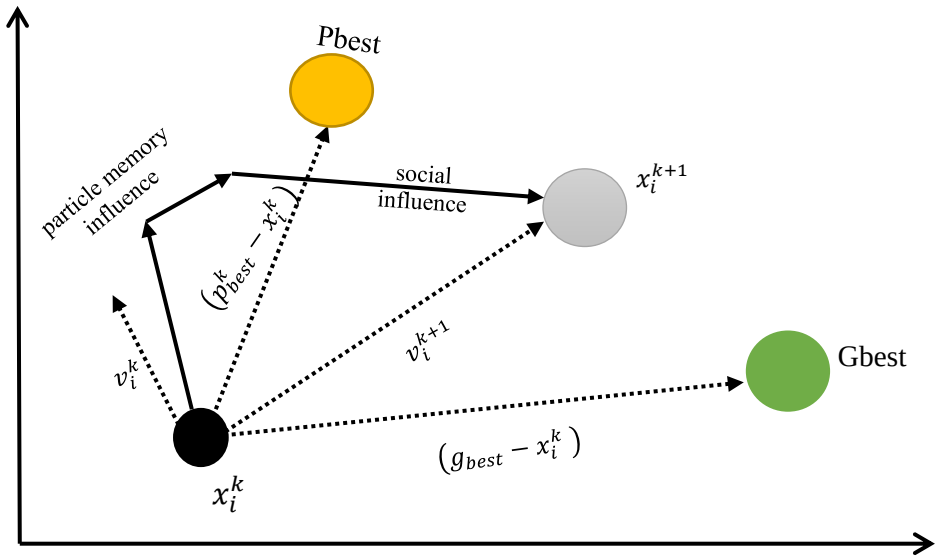

Figure 1: PSO particles geometry representation in a two-dimensional search space.

In sum, PSO algorithm attempt to solve a problem by employing a random population of potential solutions called particles that travels through a multidimensional space in search for best points (food) according to a simple mathematical formula. This implying that PSO does not require that the function be differentiable. Thus, PSO is easy to implement with simple parameters to adjust.

A basic pseudo-code for implementing PSO algorithm is given below

Initialize PSO parameters and define population size of particles

Read in the values of response y and predictor variables x

Do

for each particle

calculate the fitness value

if the fitness value is better than any best fitness available

update and set as new p_{best}

end for

select the particle with the best fitness value (p_{best}) from all the particle as g_{best}

for each particle

compute and update particle velocity v_i^{k+1}

Compute and update the particle position, x_i^{k+1}

end for

while the condition is not satisfied

return Gbest as the best estimates of the global optimum

Genetic algorithm (GA)

The Genetic algorithm is a meta-heuristic and an adaptive search algorithm designed to emulate the Darwinian theory of "survival of the fittest". It mimics the natural activities of biological species and their ability to adapt and compete effectively for resources in a search region or landscape. Developed by Holland and his colleagues in the 1970s, GA is known to for its high level procedure and elegant list of accessories called operators used to obtain optimal solution for complex problems.

Similar to PSO, the GA algorithm begins with a random population of potential solutions (say, ancestor/parents) that undergo some selection criteria and reproduction. At each

stage, a pool of candidate solutions is randomly selected to experience mutation, reproduction and recombination, then give birth to a new set of solutions (say, offspring) that replaces the "weaker" parents. Usually, the new set of offspring is better or "fitter" than its parents. To ensure this, the GA technique assigns a fitness ranking to each candidate so that a large proportion of the "fitter" individual survives and are capable to crossover and reproduce into another generation. The cycle is repeated in this manner for some number of successive generations until some conditions are met. Thus, GA is advantageous to most conventional techniques, such as not requiring gradient information, large search space exploration capability and low memory usage by eliminating weak or redundant solution at each generation. It is also offers a tangible list of operators for tuning which may guarantee better estimates.

Essentially, after the initialization of population, in each iteration or generation, the GA operates using at least three genetic operators, namely, selection, crossover and mutation. Selection operator ensures that healthy solutions are stochastically chosen in the current generation for reproducing in the next generation. Crossover operator randomly single out pairs of individual from the population to generate a superior offspring while Mutation integrates and maintain diversity among the solutions by injecting randomness into the population in order to prevent stagnation, so, sometimes, delaying the computation time of GA. Hence, these GA operators invariably contribute to the search for optimum global in a persistently evolving set of solutions which makes it an adaptive search algorithm.

A fundamental pseudo-code for implementing the Genetic algorithm is given as follows:

Initialize GA parameters and define random population

Read in the values of response y and predictor variables x

while (some condition is not satisfied) do

compute the fitness of each individual

select random pairs of individual for mating

generate new individual and crossover using the crossover operator

randomize the entire population using mutation operator

evaluate the fitness of new individuals

update and replace weak individuals with new ones

end while

return best individual found as optimal fitness value

Experimental Problems

To investigate the performance of PSO and GA, we extracted five nonlinear regression experimental problems from the National Institute of Standards and Technology (NIST) website (NIST/SEMATECH, 2022) alongside their corresponding certified values. The number of observations for each problem ranges from 6 to 35 and a maximum number of 3 parameters. While it is unstated on the reference website how each problem difficulty level was determined, we predict that the model structure/dimension is the chief characteristics. All experimental models and datasets used in this study can be found at the NIST website: http://www.itl.nist.gov/div898/strd/nls/nls_main.shtml.

Table 1: Nonlinear regression experimental problems

S/N	Model name	Difficulty level/classification	No. of observation. /parameters.	Model structure
1	Chwirut2	Lower/Exponential	54/3	$y = \frac{exp[-\beta_1 x]}{\beta_2 + \beta_3 x}$
2	Danwood	Lower/Miscellaneous	6/2	$y = \beta_1 x^{\beta_2}$
3	Misra1d	Average/Exponential	14/2	$y = \frac{\beta_1 \beta_2 x}{1 + \beta_2 x}$
4	Eckerle4	Higher/Miscellaneous	35/3	$y = \frac{\beta_1}{\beta_2} exp\left[\frac{-(x-\beta_3)^2}{2{\beta_2}^2}\right]$
5	Rat42	Higher/Miscellaneous	9/3	$y = \frac{\beta_1}{1 + exp[\beta_2 - \beta_3 x]}$

Model formulation and Implementation

It is useful to observe that least square estimation in regression attempt to minimize the error sum of squares between observed and predicted data-points, given by the formula:

$$SSE\ (\beta) = \sum_{i=1}^{n} \epsilon_i{}^2 = \sum_{i=1}^{n} (y_i - f(x_i, \beta))^2 \qquad (4)$$

Here, the $SSE\ (\beta)$ equation is likened to minimization of the objective function of an unconstrained optimization problem – one of the most frequently encountered problem in mathematics. Thus, we formulated each problem of interest to resemble the $SSE\ (\beta)$ equation. For instance, Misrald model becomes,

$$Fobj_{misrald} = \sum_{i=1}^{n} \left(y_i - \left(\frac{\beta_1 * \beta_2 x}{1 + \beta_2 x} \right) \right)^2 \qquad (5)$$

Furthermore, in order estimates the coefficients of experimental models, we developed custom functions for each model and incorporated it into the PSO and GA packages in R. The function takes in each dataset and a list of unknown parameters as argument and returns optimal values that corresponds to the unknown estimates of the model parameters. In addition, we programmed the estimation algorithm to stop when it reaches a pre-defined tolerance level of $1e-8$ or when maximum-iteration is reached. Other in-built libraries and packages employed in the process include ***"plot" and "point"*** to visualize the behavior of the solution and ***"tic & toc"*** to track the solution time. All the implementation was carried out in R programming environment (R core team, 2021). The performance results are obtained displayed on Table 3 through 7.

RESULTS

Table 2: Parameter settings for PSO and GA

PSO	**GA**
Swarm size: 500	Population size: 500
Max_Iteration:100	Generation: 100

Table 3: Chwirut2 model parameter estimates using PSO and GA method

Parameters	**Chwirut2 model**		
	PSO approach		**Certified values**
	Best	**Worst**	
$\boldsymbol{\beta_1}$	0.1628154258511	0.1468814223203	0.16657666537
$\boldsymbol{\beta_2}$	0.0050999154773	0.0050700722573	0.0051653291286
$\boldsymbol{\beta_3}$	0.0123073249707	0.0125276176092	0.012150007096
residual sum of squares	513.1504670813640	521.4695044173250	513.04802941
solution time	1.189999999999	1.7099999999991	

	GA approach		**Certified values**
	Best	**Worst**	
$\boldsymbol{\beta_1}$	0.1617181395064	0.1503462096550	0.16657666537
$\boldsymbol{\beta_2}$	0.0050338468429	0.0050605787910	0.0051653291286
$\boldsymbol{\beta_3}$	0.0124072200322	0.0124490977093	0.012150007096
residual sum of squares	513.433302714000	520.557466898800	513.04802941
solution time	17.9600000000010	17.5602100000021	

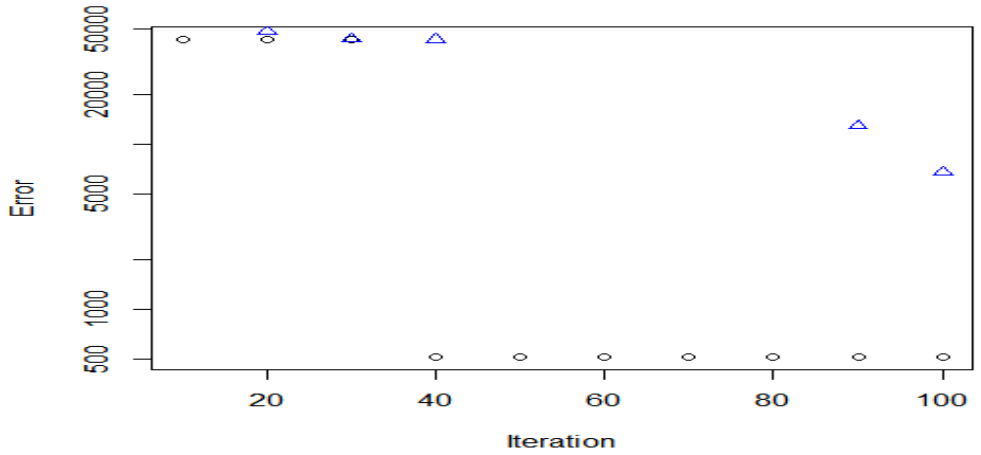

Figure 2: Residual error behavior of PSO on **Chwirut2** model

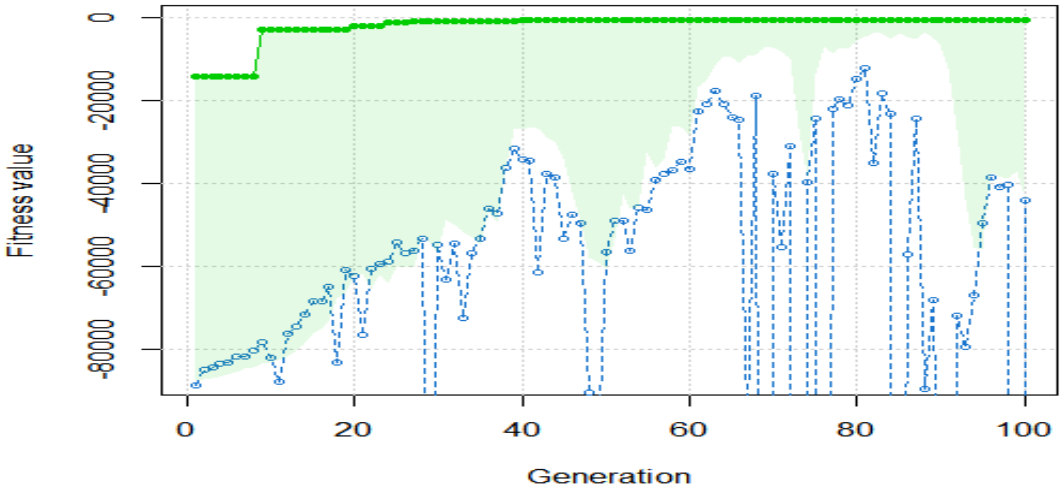

Figure 3: Residual error behavior of GA on Chwirut2 model

Table 4: DanWood model parameter estimates using PSO and GA method

Parameters	DanWood model		Certified values
	PSO approach		
	Best	Worst	
β_1	0.7688647754452	0.7688667739250	0.76886226176
β_2	3.860398508801	3.8603938891141	3.8604055871
residual sum of squares	0.004317308429	0.0043173084943	0.0043173084083
solution time	3.86039388911411	1.29999999998836	
	GA approach		Certified values
	Best	Worst	
β_1	0.7627743780613	0.7528803445936	0.76886226176
β_2	3.8774786144500	3.9054983341850	0.0038604055871
residual sum of squares	0.0044387512934	0.0068581464626	0.0043173084083
solution time	32.58	18.80	

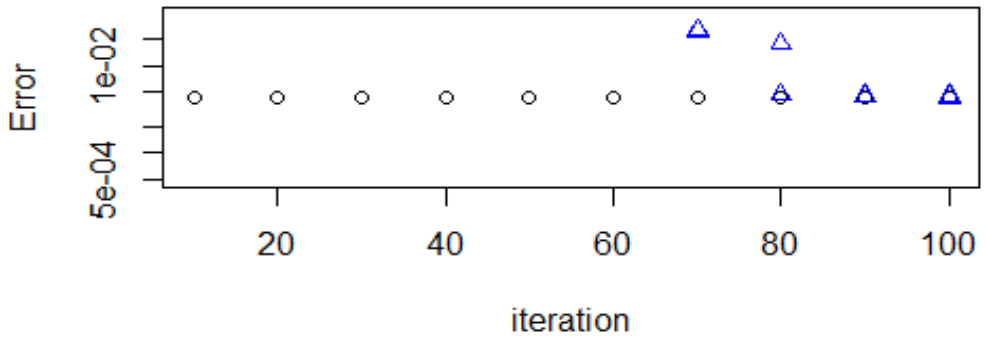

Figure 4: Residual error behavior of PSO on Danwood model

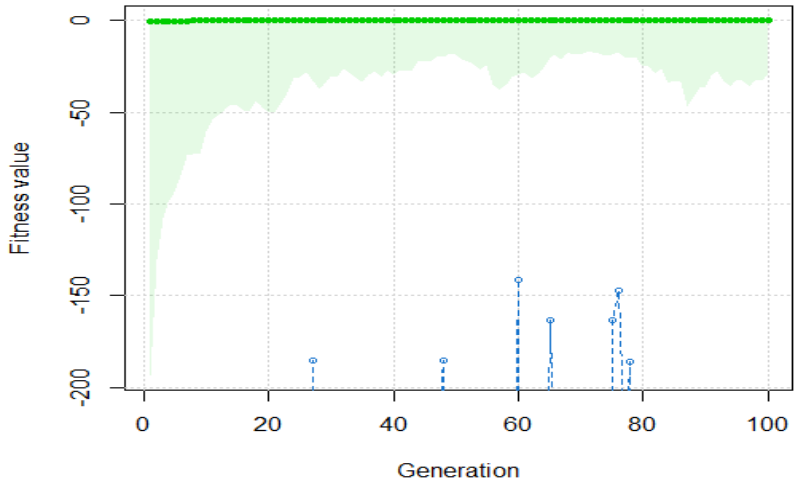

Figure 5: Residual error behavior of GA on Danwood model

Table 5: Misra1d model parameter estimates using PSO and GA method

Parameters	Misra1d model		
	PSO approach		Certified values
	Best	Worst	
β_1	437.3697077639100	437.3705698042650	437.36970754
β_2	0.0003022732443	0.0003022725033	0.0003.0227324449
residual sum of squares	0.0564192952826	0.0564192961771	0.056419295283
solution time	6.940000000002	4.3792093000000	

	GA approach		Certified values
	Best	Worst	
β_1	435.5897855361000	432.0244522139000	$437.36970754E+02$
β_2	0.0003036449772	0.0003066898270	0.00030227324449
residual sum of squares	0.0586445124135	0.0677440703519	0.056419295283
solution time	16.89	16.14	

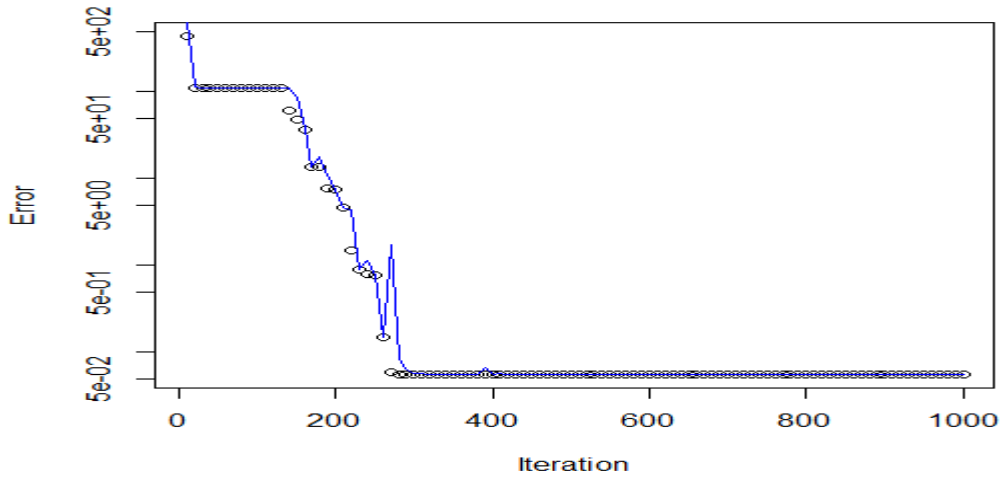

Figure 6: Residual error behavior of PSO on Misra1d model

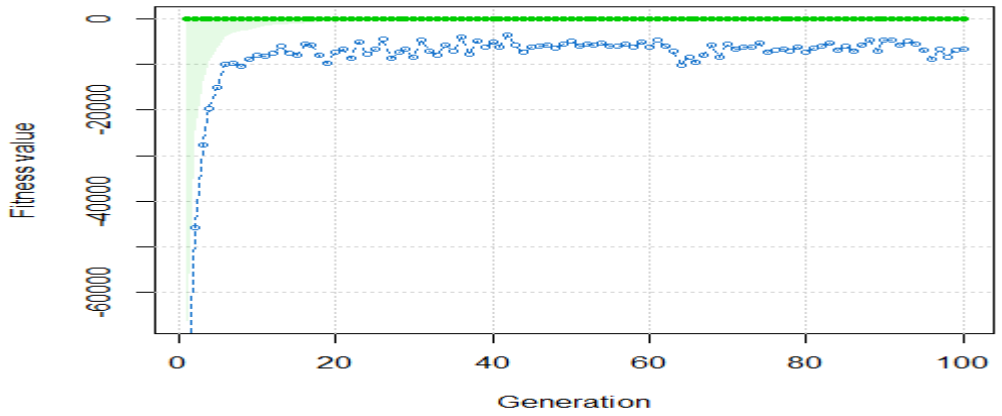

Figure 7: Residual error behavior of GA on Misra1d model

Table 6: Eckerle4 model parameter estimates using PSO and GA method

Parameters	Eckerle4 model		
	PSO approach		Certified values
	Best	Worst	
β_1	1.55438291079400	1.55438802087395	1.5543827178
β_2	4.0888329851750	4.08884628607915	4.0888321754
β_3	451.5412186098000	451.5412019034748	451.54121844
Residual sum of squares	0.001463588748742	0.00146358876006	0.001.4635887487
Solution time	1.860000000001	1.510000000002	

	GA approach		Certified values
	Best	Worst	
β_1	1.5534598114830	1.5447127801880	1.5543827178
β_2	4.0862854840560	4.0517526055230	4.0888321754
β_3	451.5425802263000	451.5741729681000	$451.54121844E+02$
residual sum of squares	0.001463809008197	0.001514428840176	0.0014635887487
solution time	17.10	23.47	

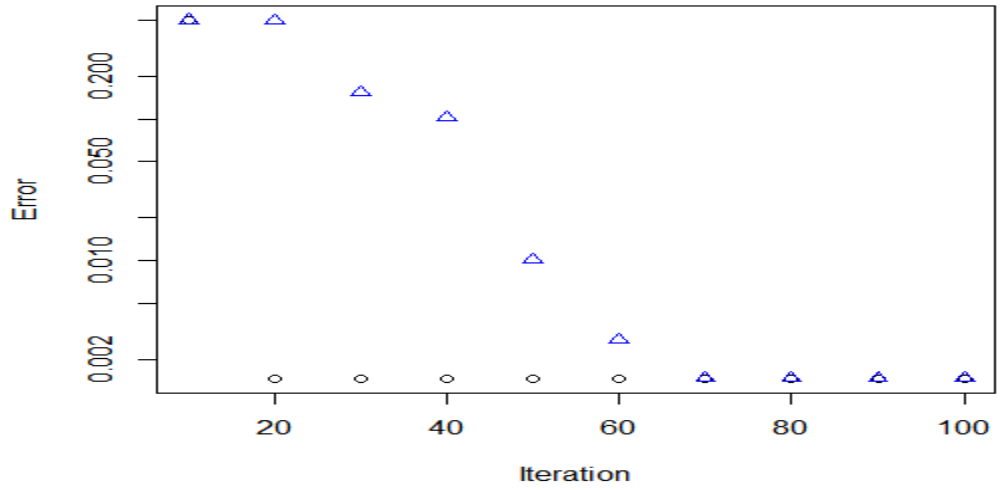

Figure 9: Residual error behavior of PSO on Eckerle4 model

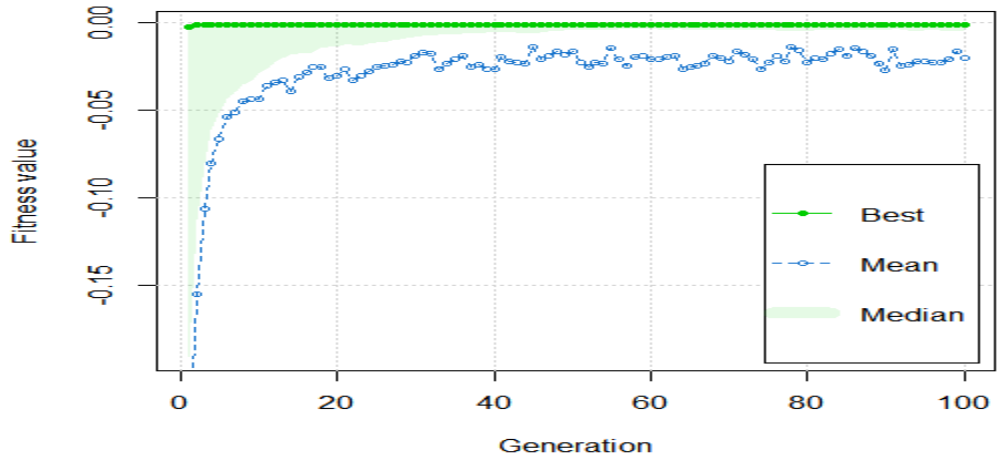

Figure 10: Residual error behavior of GA on Eckerle4 model

Table 7: Rat42 model parameter estimates using PSO and GA method

Parameters	Rat42 model		Certified values
	PSO approach		
	Best	Worst	
β_1	72.46223755243584	57.7779974512380	72.462237576
β_2	2.61807684163544	47.6198008416280	2.6180768402
β_3	0.06735920007859	1.6842030577620	0.067359200066
residual sum of squares	8.056522933811	1033.2946164260000	8.0565229338
solution time	9.48999999999069	4.229999999981374	
	GA approach		**Certified values**
	Best	**Worst**	
β_1	72.4504230022400	72.8599321178700	72.462237576
β_2	2.6206826120620	2.5714411622070	2.6180768402
β_3	0.06743212044239	0.06594189874853	0.067359200066
residual sum of squares	8.0577762138830	8.4472772131430	8.0565229338
solution time	21.63	25.07000000001	

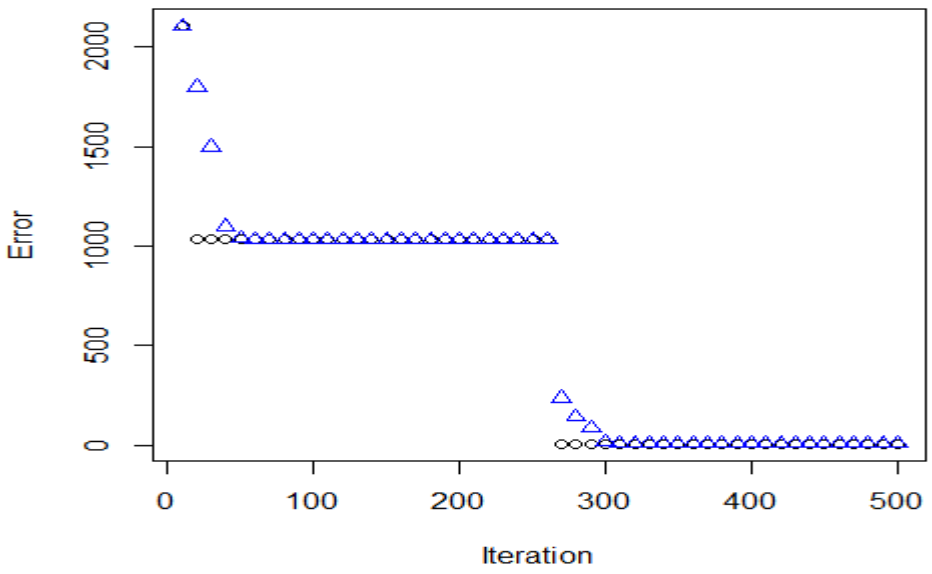

Figure 11: Residual error behavior of Rat42 for PSO

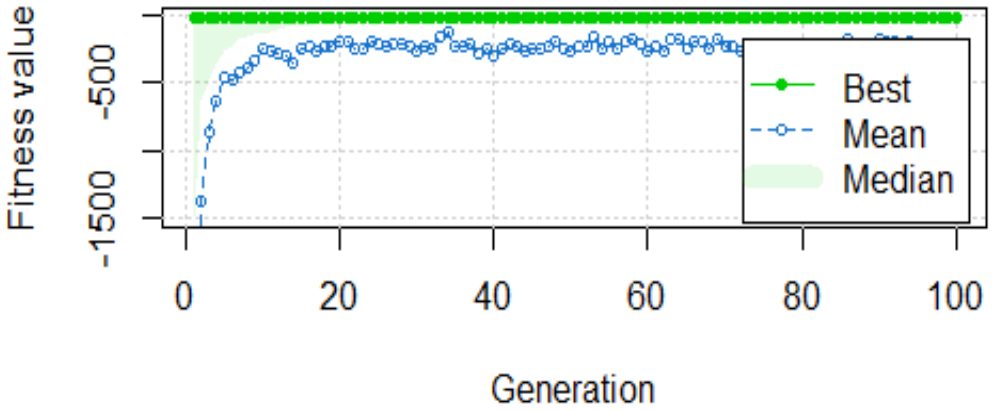

Figure 12: Residual error behavior of Rat42 for GA**.**

DISCUSSION OF RESULTS

Tables 3 through 7 and Figures 2 to 12 quantifies the performance of PSO and GA algorithm. The results showed that, in most cases, PSO outperforms GA with respect to computation time, accuracy of parameters estimates and minimum sum of squares. This finding is consistent with Chandrashaker *et al*. (2017) study which examined the correlation and efficiency of PSO over GA. They highlighted that PSO and GA is computationally superior to GA. Still, in this study, PSO struggled to maintain composure on the parameters values of Rat42 model. For instance, on Rat42 model, PSO's worst estimates for $\boldsymbol{\beta_1, \beta_2, \beta_3}$ and RSS, considerably differ from optimal values of the model, this likely suggest that the algorithm could be limited by structural composition of an objective function. In spite of this potential limitation, we observed that PSO has a simple but powerful configuration which ensures that the entire solution landscape is adequately covered, which is apparent in its best estimates for each models.

On the other hand, estimated values for GA approach are fairly comparable or similar to the optimal values of each nonlinear regression models. However, a detailed inspection (Tables 3 - 7) indicates that GA quality performance is stifled by its slow runtime, the fastest being 16.14 seconds on Misrald model (Table 6). We observed that the lag in GA's solution time may be due to its sophisticated operators and concurrent operations involved at each generation. Recently, Ajay and Ausif (2016) proposed that a fine-tuned crossover operator would likely speed up and improve the quality of GA optimum values. Also, we deduced that GA strength lie in its complex nature of search which enables it to maintain a balance in the number of iterations (generations) when tackling different structure of nonlinear regression problems, as displayed in Figures (2 – 12) above. This property indicate that functional complexity does not significantly alter the performance of GA.

These results indicate that PSO and GA algorithms are both competitive and converge to respective global optimal values of the parameters of the nonlinear regression models at reasonable rates. Based on these findings, one can conclude that the PSO and GA techniques are quite effective. It is also worthy to note that the estimated values in this study are in consonance with other studies. Finally, due to the limited sample of experimental problems considered in this study it is impossible to state that one algorithm generally supersedes another in all situations. Notwithstanding, based on results from the experimental problems, it can be concluded that the PSO and GA techniques provide an alternative and reliable approach for obtaining parameter estimates of a wide range of nonlinear regression models.

CONCLUSION

The use of particle swarm optimization and genetic algorithm for nonlinear regression parameter estimates present a simple, effective and reliable way to enhance optimal global values which further aids sound statistical judgments. In this study, we investigated the performance of particle swarm optimization and genetic algorithm for obtaining the parameter estimates of five nonlinear regression problems with varying level of difficulties (low, average, higher). We found that both PSO and GA estimates, when compared with other published sources can be considered effective for nonlinear regression unknown parameter estimate tasks. Our results show that while PSO solution speed is faster relative to GA, its number of iterations may be inconsistent for different problems. Fewer iterations and fairly accurate results could be a balance for GA's slow computation time. Accordingly, we conclude that both algorithms are reliable and effective for varying tasks involving nonlinear regression parameter estimates. Future study may explore the use of a high quality Automatic Differentiation (AD) software (Wikipedia, 2022) for hybridizing gradient based algorithms with either of PSO or GA to efficiently tackle high-dimensional nonlinear functions with rugged search spaces. Perhaps, the PSO or GA could be used to locate set of points that evolve to optimal global, then, a gradient based method that would accurately locate the deep local minimum. Future studies could also explore the usage of the PSO and GA techniques to solve optimization problems arising from other form of models such as nonlinear mixed effects models (Adeniyi *et al*., 2018), generalized mixed effects models (Ezenweke et al., 2022a), geo-spatial structured additive models (Ezenweke et al., 2022b) amongst others.

REFERENCES

Adeniyi, I. A., Yahya, W. B., & Ezenweke, C.P. (2018). A Note on Pharmacokinetics Modelling of Theophylline Concentration Data on Patients with Respiratory Diseases. *Turkiye Klinikleri Journal of Biostatistics*, *10*(1), 27-45. doi:10.5336/biostatic.2017-58451.

Adjad H., Baba YF., Mers A. A., Merron O., Bouatern A., Boutmmachte N. (2019). Particle swarm optimization for optimal-geometric optimization of linear Fresnel solar concentrations. *Renewable Energy*, 130, 992-1001.

Ajay S. & Ausif M. (2016). Improving Genetic Algorithm with fine-tuned Crossover and Scaled Architecture. *Journal of Mathematics*, 2016.

Archontoulis, S. V., & Miguez, F. E. (2015). Nonlinear regression models and applications in agricultural research. *Agronomy Journal*, *107*(2), 786-798.

Bates, D. M. & D. G. Watts. (2007). *Nonlinear Regression and its Applications*. John Wiley and Sons, New York.

Belhocine, A., Shinde, D., & Patil, R. (2021). Thermo-mechanical coupled analysis based design of ventilated brake disc using genetic algorithm and particle swarm optimization. *JMST Advances*, *3*(3), 41-54.

Bulent, A., & Alptekin E. (2004). *The genetic algorithm method for estimation in nonlinear regression.* G.U Journal of Science 17(2), 43-51.

Chandrashaker R. B., Venkat Prasad, Reddy P., & Rajeshwari M., Kavya Y. Sai (2017). Correlation of GA and PSO for Analysis of Efficient optimization. *International Journal of Advance Research and Development*, 2(4).

Chicco, G., & Mazza, A. (2020). Metaheuristic Optimization of Power and Energy Systems: Underlying Principles and Main Issues of the 'Rush to Heuristics.' *Energies*, *13*(19), 5097. http://dx.doi.org/10.3390/en13195097

de Almeida, B. S. G., & Leite, V. C. (2019). Particle swarm optimization: A powerful technique for solving engineering problems. *Swarm intelligence-recent advances, new perspectives and applications*, 1-21.

Desale, S.A., Rasool, A., Andhale, S., & Rane, P.V. (2015). Heuristic and Meta-Heuristic Algorithms and Their Relevance to the Real World: A Survey. *International journal of computer engineering in research trends*, 351. 2349-7084.

Erdoğmuş, P., & Ekiz, S. (2016). Nonlinear regression using particle swarm optimization and genetic algorithm. *International Journal of Computer Applications*, *153*(6), 28-36.

Ezenweke, C. P., Adeniyi, I. A., Edogbanya, H. O., & Yahya, W. B. (2022a). Spatial distributions and risk factors of overweight and obesity among women in Nigeria using structured geo-additive regression models: analysis of 2018 Nigeria demographic health survey. *FUDMA Journal of Sciences*, *6*(4), 112-124.

Ezenweke, C. P., Adeniyi, I. A., Yahya, W. B. & Onoja, R. E. (2022b). Determinants and Spatial Patterns of Anaemia and Haemoglobin Concentration among Pregnant Women in Nigeria Using Structured Additive Regression Models. *Spatial and Spatio-temporal Epidemiology* (Accepted).

Friedl, G., & Kuczmann, M. (2014). Population and gradient based optimization techniques, a theoretical overview. *Acta Technica Jaurinensis*, *7*(4), 378-387.

Ghosh, M., Guha, R., Alam, I., Lohariwal, P., Jalan, D. & Sarkar, R. (2020). *Binary Genetic Swarm Optimization: A Combination of GA and PSO for Feature Selection.* Journal of Intelligent Systems, 29(1), 1598-1610.

Gupta, S.K. (2021). An overview of Genetic algorithms: A structural Anlaysis. *International Journal of Innovative Science and Research Technology*, 6(5), 1305-1309.

Holland J. (1975) *Adaptation in Natural and Artificial Systems*, The University of Michigan Press, Ann Arbor, Mich, USA, 1975.

Hsin-Hsiung Huang. (2010). *Nonlinear Regression Analysis.* International Encyclopedia of Education. Oxford: Elsevier.

Kapanoglu M., Koc, I.O., and Erdogmus, S., (2007). *Genetic algorithms in parameter estimation for nonlinear regression models: an experimental approach*, Journal of Statistical Computation and Simulation, 77(10): 851-867.

Kennedy, J., & Eberhart, R. (1995, November). Particle swarm optimization. In *Proceedings of ICNN'95-international conference on neural networks* (Vol. 4, pp. 1942-1948). IEEE.

Khanduja, N., Bhushan, B. (2021). Recent Advances and Application of Metaheuristic Algorithms: A Survey (2014–2020). In: Malik, H., Iqbal, A., Joshi, P., Agrawal, S., Bakhsh, F.I. (eds) Metaheuristic and Evolutionary Computation:

Algorithms and Applications. Studies in Computational Intelligence, vol 916. Springer, Singapore. https://doi.org/10.1007/978-981-15-7571-6_10

Liu R. (2014). A Particle Swarm Optimization Based Simultaneous Learning Framework for Clustering and Classification. *Pattern Recognition, 47*, 2143 - 2152.

Madsen K., Nielsen H.B., Tingleff O. (2004). *Methods for non-linear least squares problems.* 2nd Edition Informatics and Mathematical Modelling Technical University of Denmark.

Malekan M and Khosravi A. (2018). Investigation of convective heat transfer of ferro fluid using CFD simulation and adaptive neuro-fuzzy inference system optimized with particle swarm optimization. *Powder Technology*, *333*, 364-376.

Malik, H., Iqbal, A., Joshi, P., Agrawal, S., & Bakhsh, F. I. (Eds.). (2021). *Metaheuristic and evolutionary computation: algorithms and applications* (pp. 46-61). Springer Nature Singapore.

Naidu, P. S., Bhagat, B., & Rathi, N. (2018). Particles Swarm Optimization Techniques: Principle, Comparison & Application. *International Journal of Computer Science Engineering and Information Technology Research (IJCSEITR)*, *8*, 37-48.

NIST/SEMATECH. *e-Handbook of Statistical Methods*, NIST/SEMATECH, Retrieved May 4, 2022 from https://www.itl.nist.gov/div898/handbook/pmd/section1/pmd142.htm.

Özsoy, V. S., & Örkçü, H. H. (2016). Estimating the parameters of nonlinear regression models through particle swarm optimization. *Gazi University Journal of Science*, *29*(1), 187-199.

R Core Team (2021). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Rajakumar, R., Dhavachelvan, P., & Vengattaraman, T. (2016, October). A survey on nature inspired meta-heuristic algorithms with its domain specifications. In *2016 international conference on communication and electronics systems (ICCES)* (pp. 1-6). IEEE.

Sengupta, S., Basak, S., & Peters, R. A. (2018). Particle Swarm Optimization: A survey of historical and recent developments with hybridization perspectives. *Machine Learning and Knowledge Extraction*, *1*(1), 157-191.

Wikipedia (2021). *Automatic differentiation*. Wikipedia, Retrieved September 4, 2022 from https://www.wikipedia.org/wiki/Automatic_differentiation.

Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, *1*(1), 67-82.