Middle East Journal of Applied Science & Technology (MEJAST)
Volume 9, Issue 4, Pages 01-16, October-December 2026

Block Hybrid Optimization Numerical Technique for Improving Runtime Efficiency in High-Complexity Algorithms

Sabo John*

Department of Mathematics, Adamawa State University, Mubi, Nigeria.
*Corresponding Author Email: sabojohn630@gmail.com**

DOI: Under Assignment

Copyright © 2026 Sabo John. This is an open-access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Article Received: 16 August 2026 *Article Accepted: 23 October 2026* *Article Published: 26 October 2026*

ABSTRACT

The study introduces a block hybrid optimization method using a power series polynomial as an approximate solution, which combines Genetic Algorithm (GA) global search abilities with Particle Swarm Optimization (PSO) local search capabilities to enhance high-complexity computational algorithms. The researchers implemented the hybrid system in MATLAB to test its effectiveness on optimization problems which had different sizes. The research team used MATLAB profiling tools together with memory-tracking functions to create a complete set of performance data which included execution time, iteration counts, error values and memory usage and scalability characteristics. The experimental results show that the hybrid method achieves better results than the individual GA and PSO methods. The hybrid model achieved up to 50% reduction in execution time, 37–46% faster convergence and 10–31% improvements in solution accuracy, with only a minimal increase in memory usage. The hybrid system showed a near-linear increase in runtime as the problem size grew while GA and PSO experienced exponential runtime increase. The research results demonstrate that the hybrid GA to PSO approach delivers an effective optimization solution which can handle large-scale operations in machine learning and resource scheduling and big data analysis and real-time computational systems.

Keywords: Hybrid Optimization; Genetic Algorithm (GA); Particle Swarm Optimization (PSO); Runtime Efficiency; Convergence Performance; Solution Accuracy; Memory Utilization; Scalability; Block Hybrid; Numerical Analysis; Optimization Method.

1.0. Introduction

The combination of various heuristic and metaheuristic and multi-agent methods through hybrid algorithms enables computer scientists to utilize their strengths while handling the problems which arise from using only one method because of its slow convergence and restricted search capabilities [1, 2]. The hybrids achieve better problem-solving results through faster convergence and increased efficiency when they combine metaheuristics together with deep learning and heuristic search together with mathematical programming and multi-agent distributed frameworks [3, 4]. The system can function through three different modes of operation which include sequential mode and iterative mode and parallel mode while it finds applications in machine learning and ensemble modeling and feature selection and clustering and anomaly detection and developing quantum-classical systems because it provides a strong solution method for difficult computational challenges [5-7].

Researchers require advanced algorithms which possess high computational complexity for solving their challenging problems in artificial intelligence and big data analytics and cryptography and bioinformatics fields. The growing size of datasets creates major difficulties for traditional algorithms because they experience longer processing times and they encounter memory limits and their ability to handle larger datasets decreases [8, 9]. High-dimensional search spaces present major challenges for conventional optimization techniques which need to develop new methods that can operate in changing computational environments to manage between 10 and 20 modern systems [10].

Hybrid optimization techniques have developed into an effective solution through their integration of various optimization methods which include evolutionary algorithms and swarm intelligence and machine-learning-based

search methods to achieve better convergence results and exploration capabilities and reduction of local stagnation problems. These hybrid systems use their distinct strengths from different components to achieve better performance results and solution accuracy than systems that rely on a single strategic approach. The development of hybrid techniques that deliver better performance results while sustaining scalable and robust systems with minimal computational demands remains an unresolved challenge [8, 11]. The study aims to fill these research gaps through the creation of an innovative hybrid optimization framework which delivers rapid processing power with accurate results and adaptable capabilities for solving complex computational challenges.

Researchers increasingly employ advanced algorithms to solve computational difficulties encountered in artificial intelligence and big data analytics and cyber security and engineering simulation fields. The algorithms demonstrate exponential or near-exponential growth in runtime which becomes evident when researchers increase the problem size Beyond this point the algorithms experience execution delays and resource depletion and inability to handle larger problems [10-12].

The existing optimization techniques which include evolutionary algorithms and swarm intelligence and classical heuristics demonstrate partial success but they face difficulties because of their tendency to converge too soon and their requirement for high computational resources and their inconsistent results across various problem types. Hybrid optimization techniques have shown potential as effective solutions yet existing hybrid models do not provide an organized system that combines fast convergence with low runtime expenses. The field requires development of a more effective hybrid optimization method which will decrease computation time while keeping high accuracy for solutions in complex algorithmic challenges [3, 13-15].

The research described in reference [16] develops a hybrid optimization framework which integrates Genetic Algorithms with gradient-based techniques to solve the problem of exploration-exploitation imbalance together with the issue of premature algorithm convergence. The hybrid system, tested on the Active Flow Control problem for the SD7003 airfoil, produced better lift coefficients while needing fewer objective function evaluations than each individual method, which led to better computational efficiency and system robustness and capability to handle complex engineering design challenges.

The authors of the systematic review [17] present a study which shows how hybrid methods that combine optimization algorithms with machine learning techniques can solve complex problems that require extensive data processing. The hybrid methods combine different techniques from evolutionary search and gradient-based learning which enables them to achieve better results with greater capacity and better system performance for tasks which include clustering and classification. The hybrid approach creates more adaptable and efficient computational models which can perform better in real-world data-driven scenarios despite the difficulties that standalone methods present.

The study by [18] presents a hybrid algorithm which combines multiple computational techniques to enable better performance in complex optimization tasks. The system achieves better results through its advanced path-planning features because the system combines different strengths to build abilities that enable swift and reliable problem-solving.

Author [19] developed hybrid optimization algorithms which solve complex nonlinear problems by combining worldwide search methods from evolutionary algorithms and swarm intelligence with rapid local optimization techniques. The method improves solution quality and computational efficiency in engineering, AI, finance, and healthcare through its exploration and exploitation balance. Adaptive hybrid systems face challenges with parameter tuning and algorithm compatibility yet they develop automated solutions which deliver greater system stability.

The process of optimization establishes a structured method to discover the optimal solution among multiple workable choices according to its application in complex search environments which lack analytical solution methods. In computational intelligence, optimization techniques such as Genetic Algorithms (GA) and Particle Swarm Optimization (PSO) are widely employed to iteratively refine candidate solutions toward a global optimum. The general form of an optimization problem is expressed as:

$$\min_{x \in \Re^n} \; f(x) \tag{1.1}$$

where denotes the objective function whose minimum value is sought over the decision variables. Evolutionary and swarm-based optimization techniques improve this search process by updating solution candidates iteratively. A representative population-based update rule, such as that used in PSO, can be expressed as:

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{1.2}$$

Where represents the current position of particle and is the updated velocity that guides the particle toward an improved solution. These formulations provide a mathematical basis for understanding how optimization algorithms evolve solutions across iterations to achieve superior performance across diverse computational tasks.

Author [20] extended the block hybrid approach to Volterra integro-differential equations of the second kind, addressing both linear and nonlinear cases, thereby expanding the applicability of the method beyond ordinary differential equations to integral equations with memory effects. Ref. [21] applied a novel linear block method to oscillatory solutions of the Malthusian growth model, decay model and thermofluid systems, demonstrating the method's versatility in handling real-world dynamic systems exhibiting oscillatory behavior, with findings relevant to population dynamics and thermal engineering.

The development of computational integrated-block-based simulation for approximating solutions to multi-degree regular differential equations, highlighting the method's capacity to handle higher-order and multi-degree problems with improved accuracy and computational efficiency was studied by ref. [22]. In a study by author [23], who analyzed a block method constructed with six generalized grid points for solving fourth-order initial value problems, contributing to the theoretical foundation of block methods by optimizing grid point selection for enhanced convergence and stability. Collectively, these studies advance the field of numerical analysis by providing robust, efficient, and adaptable block hybrid techniques that address a wide spectrum of differential equation problems, with demonstrated improvements in accuracy, convergence, and computational performance across diverse applications in science and engineering.

Definition 1.1: Hybrid Algorithm refers a computational method created by combining two or more distinct algorithms (e.g., Genetic Algorithm and Particle Swarm Optimization) to leverage their complementary strengths, improving convergence speed, solution accuracy, and runtime efficiency in solving high-complexity optimization problems.

Definition 1.2: Genetic Algorithm (GA) refers to an evolutionary algorithm that simulates natural selection to search for optimal solutions by iteratively generating, evaluating, and evolving a population of candidate solutions based on fitness criteria.

Definition 1.3: Particle Swarm Optimization (PSO) refers to a population-based optimization method where individual "particles" move through the solution space influenced by their personal best positions and the global best, allowing rapid convergence to optimal solutions.

Definition 1.4: Runtime Efficiency refers to the measurement of computational time required by an algorithm to reach an acceptable solution, quantified in seconds using MATLAB's timing functions (tic/toc) across multiple problem sizes.

Definition 1.5: Convergence Speed refers to the rate at which an algorithm approaches its optimal solution, measured by the number of iterations or generations needed to meet a predefined fitness threshold or termination condition.

1.1. Study Objectives

Based on the research paper "Block Hybrid Optimization Numerical Technique for Improving Runtime Efficiency in High-Complexity Algorithms," the following six study objectives are formulated:

1. To derive the block hybrid optimization method.
2. To design and develop a hybrid GA-PSO optimization algorithm.
3. To derive and formulate an optimized linear multistep hybrid numerical method.
4. To evaluate the runtime efficiency and convergence performance of the hybrid algorithm.
5. To assess the solution accuracy and memory utilization of the hybrid model.
6. To test the scalability of the hybrid optimization technique.
7. To demonstrate the practical applicability of the hybrid framework.

2.0. Materials and Method

The researchers conducted an assessment of the developed hybrid optimization method using the power series polynomial as a basic function through interpolation and collocation procedures, by executing its evaluation which combined Genetic Algorithm (GA) global search capabilities with Particle Swarm Optimization (PSO) for local search execution through MATLAB programming. The researchers examined runtime efficiency and convergence speed and solution accuracy and memory usage and system performance through benchmark high-complexity optimization problems that included multiple problems of different sizes. The researchers used MATLAB profiling and memory functions to record execution times and iteration counts and error values and memory consumption

while conducting experiments which they repeated to measure stochastic variability. The researchers conducted a performance assessment which involved comparing the hybrid model against standalone GA and PSO systems to evaluate speed improvements and accuracy enhancements and scalability advancements which confirmed the hybrid model's effectiveness for processing high-dimensional machine learning optimization and resource scheduling and big data analytics and real-time AI processing tasks.

2.1. Derivation Technique of Optimization Method

Consider an approximate solution of a series of the form;

$$\gamma(t)=\sum_{i=0}^{\omega+\varpi-1}\vartheta_j t^j \tag{2.1}$$

Where ω and ϖ are the number of interpolation and collocation points respectively. The first derivative of (2.1) is given by

$$\gamma'(t)=\sum_{i=0}^{\omega+\varpi-1} j\,\vartheta_j t^{j-1} \tag{2.2}$$

Where $\vartheta_j \in \Re$ for $j=0(1)6$ and $\gamma(t)$ is continuously differentiable. Let the solution of (1.1) be sought on the integration interval $[u, v]$ with a constant step-size h, defined by, $h=t_{n+1}-t_n, n=0,1,\cdots,N$. Substituting equation (2.2) in (1.1) to gives,

$$f(t,\gamma)=\sum_{i=0}^{\omega+\varpi-1} j\,\vartheta_j t^{j-1} \tag{2.2}$$

Therefore, interpolate equation (2.1) at point $t_{n+u}, u=0$ and collocate equation (2.2) at points $t_{n+v}, v=0,\xi,\frac{2}{5},\frac{3}{5},\frac{4}{5},1$, where ξ is the point of optimization. This leads to the following system of equations,

$$\mathrm{AX}=\mathrm{E} \tag{2.3}$$

Where,

$$A=\begin{bmatrix}\vartheta_0 & \vartheta_1 & \vartheta_2 & \vartheta_3 & \vartheta_4 & \vartheta_5 & \vartheta_6\end{bmatrix}^T$$

$$\mathrm{E}=\begin{bmatrix}\gamma_n & f_n & f_{n+\xi} & f_{n+\frac{2}{5}} & f_{n+\frac{3}{5}} & f_{n+\frac{4}{5}} & f_{n+1}\end{bmatrix}^T$$

$$\mathrm{X}=\begin{bmatrix}
1 & t_n & t_n^2 & t_n^3 & t_n^4 & t_n^5 & t_n^6\\
0 & 1 & 2t_n & 3t_n^2 & 4t_n^3 & 5t_n^4 & 6t_n^5\\
0 & 1 & 2t_{n+\xi} & 3t_{n+\xi}^2 & 4t_{n+\xi}^3 & 5t_{n+\xi}^4 & 6t_{n+\xi}^5\\
0 & 1 & 2t_{n+\frac{2}{5}} & 3t_{n+\frac{2}{5}}^2 & 4t_{n+\frac{2}{5}}^3 & 5t_{n+\frac{2}{5}}^4 & 6t_{n+\frac{2}{5}}^5\\
0 & 1 & 2t_{n+\frac{3}{5}} & 3t_{n+\frac{3}{5}}^2 & 4t_{n+\frac{3}{5}}^3 & 5t_{n+\frac{3}{5}}^4 & 6t_{n+\frac{3}{5}}^5\\
0 & 1 & 2t_{n+\frac{4}{5}} & 3t_{n+\frac{4}{5}}^2 & 4t_{n+\frac{4}{5}}^3 & 5t_{n+\frac{4}{5}}^4 & 6t_{n+\frac{4}{5}}^5\\
0 & 1 & 2t_{n+1} & 3t_{n+1}^2 & 4t_{n+1}^3 & 5t_{n+1}^4 & 6t_{n+1}^5
\end{bmatrix}$$

Solving equation (2.3) for $\vartheta_j's,\ j=0(1)6$ and substitute back into (2.1) to gives a optimized linear multistep hybrid method of the form

$$\gamma(t)=\alpha_0(t)\gamma_n+h\left[\beta_0(t)f_n+\beta_\xi(t)f_{n+\xi}+\beta_{\frac{2}{5}}(t)f_{n+\frac{2}{5}}+\beta_{\frac{3}{5}}(t)f_{n+\frac{3}{5}}+\beta_{\frac{4}{5}}(t)f_{n+\frac{4}{5}}+\beta_1(t)f_{n+1}\right] \tag{2.4}$$

Simplify equation (2.4) to gives

$$\left.\begin{aligned}
\alpha_0&=1\\
\beta_0(t)&=\frac{1}{2016}\frac{217\xi-19}{\xi}\\
\beta_\xi(t)&=\frac{19}{84(\xi-1)(5\xi-3)(5\xi-4)(5\xi-2)}\\
\beta_{\frac{2}{5}}(t)&=\frac{25}{1008}\frac{105\xi-23}{5\xi-2}\\
\beta_{\frac{3}{5}}(t)&=-\frac{25}{504}\frac{35\xi-9}{5\xi-5}\\
\beta_{\frac{4}{5}}(t)&=\frac{25}{672}\frac{35\xi-9}{5\xi-4}\\
\beta_1(t)&=-\frac{1}{504}\frac{21\xi-2}{\xi-1}
\end{aligned}\right\} \tag{2.5}$$

Substitute equation (2.5) into (2.3), after some simplifications, substitute the results one by one into (2.3) to gives

$$-\frac{1}{75600000}(190\xi-27)=0 \tag{2.6}$$

Solving for ξ in equation (2.6) to obtain the exact value of ξ as

$$\xi=\frac{27}{190} \tag{2.7}$$

Therefore, substitute equation (2.7) into (2.4) to gives new continuous form of an optimized method as

$$\left.\begin{aligned}
\alpha_0&=t\\
\beta_0(t)&=-\frac{t^2}{54432}\left(-27216+122052t-271915t^2+322980t^3-195650t^4+4750t^5\right)\\
\beta_{\frac{27}{190}}(t)&=\frac{t^2}{54432}\left(336-1078t+1491t^2-980t^3+250t^4\right)\\
\beta_{\frac{2}{5}}(t)&=-\frac{25t^3}{49392}\left(4536-24843t+44310t^2-33810t^3+9500t^4\right)\\
\beta_{\frac{3}{5}}(t)&=\frac{25t^3}{43848}\left(3024-17822t+36561t^2-31150t^3+9500t^4\right)\\
\beta_{\frac{4}{5}}(t)&=-\frac{t^3}{3360}\left(2268-13839t+30408t^2-28490t^3+9500t^4\right)\\
\beta_1(t)&=\frac{t^3}{82152}\left(9072-56490t+129255t^2-129150t^3+47500t^4\right)
\end{aligned}\right\} \tag{2.8}$$

Evaluate equation (2.4) at points $\left[\frac{27}{190}, \frac{2}{5}, \frac{3}{5}, \frac{4}{5}, 1\right]$ to obtain an optimization method as

$$\mathrm{A}^{(0)}\mathrm{Y}_m = \mathrm{H}\gamma_n + h\mu f\left(\gamma_n\right) + h\varphi F\left(\mathrm{Y}_n\right) \tag{2.9}$$

where

$$\mathrm{Y}_n = \left[\gamma_{n+\frac{27}{190}} \quad \gamma_{n+\frac{2}{5}} \quad y_{n+\frac{3}{5}} \quad \gamma_{n+\frac{4}{5}} \quad \gamma_{n+1}\right]^T, \gamma_n = \left[\gamma_{n-\frac{27}{190}} \quad \gamma_{n-\frac{2}{5}} \quad \gamma_{n-\frac{3}{5}} \quad \gamma_{n-\frac{4}{5}} \quad \gamma_n\right]^T$$

$$F\left(Y_n\right) = \left[f_{n+\frac{27}{190}} \quad f_{n+\frac{2}{5}} \quad f_{n+\frac{3}{5}} \quad f_{n+\frac{4}{5}} \quad f_{n+1}\right]^T, f\left(\gamma_n\right) = \left[f_{n-\frac{27}{190}} \quad f_{n-\frac{2}{5}} \quad f_{n-\frac{3}{5}} \quad f_{n-\frac{4}{5}} \quad f_n\right]^T$$

$$\mathrm{A}^{(0)} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \mathrm{H} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \mu = \begin{bmatrix} 0 & 0 & 0 & 0 & \frac{8352390951}{158470336000} \\ 0 & 0 & 0 & 0 & \frac{787}{20250} \\ 0 & 0 & 0 & 0 & \frac{737}{9000} \\ 0 & 0 & 0 & 0 & \frac{1186}{30375} \\ 0 & 0 & 0 & 0 & \frac{29}{648} \end{bmatrix}$$

$$\varphi = \begin{bmatrix} \frac{5916138237}{55010462500} & -\frac{27030145581}{776504646400} & \frac{280954413}{11489099360} & -\frac{36774910197}{3961758400000} & \frac{4816554759}{3228833096000} \\ \frac{13628448896}{58629571875} & \frac{1762}{11025} & -\frac{6}{145} & \frac{223}{18750} & -\frac{302}{183375} \\ \frac{1446041816}{6514396875} & \frac{2601}{9800} & \frac{17}{232} & -\frac{27}{12500} & \frac{9}{163000} \\ \frac{40568406016}{175888715625} & \frac{2584}{11025} & \frac{832}{3915} & \frac{818}{9375} & -\frac{704}{183375} \\ \frac{19808792}{93807315} & \frac{1025}{3528} & \frac{25}{232} & \frac{17}{60} & \frac{733}{11736} \end{bmatrix}$$

3.0. Results and Discussion

The present section shows the results which were obtained through the execution and assessment of the newly developed hybrid optimization method. The assessment evaluates five different metrics which include runtime efficiency and convergence speed and solution accuracy and memory usage and system scalability. The researchers evaluated performance through benchmark high-complexity optimization problems which compared traditional optimization methods with Genetic Algorithm and Particle Swarm Optimization. The results are summarized in Tables 3.1–3.5, while the MATLAB implementation for all experiments. The results prove that the hybrid method successfully solves typical computational problems which arise in computer science fields like machine learning and resource scheduling and big data optimization and real-time algorithmic processing.

3.1. Results

3.1.1. Runtime Efficiency

Table 3.1 presents the execution times of GA, PSO, and the developed hybrid technique across various problem sizes. The hybrid model consistently achieved faster runtimes, with improvements ranging from 43% to 51% over GA and 28% to 45% over PSO. The reduction in execution time is attributed to the hybridization of GA's global

search with PSO's local refinement, which effectively reduces redundant computations and accelerates convergence. The execution time of GA, PSO and the Hybrid Algorithm was measured across four problem sizes using MATLAB's built-in timing functions (tic/toc). Each algorithm was executed 30 times and the average runtime was recorded to minimize noise from system processes. The runtime improvement percentage was computed relative to the faster of the two baseline models (GA and PSO), using:

$$improvement = \frac{Min(GA, PSO) - Hybrid}{Min(GA, PSO)} \times 100 \quad (3.1)$$

Table 3.1. Execution Time Comparison of Algorithms (Seconds).

Problem Size (n)	GA Runtime	PSO Runtime	Hybrid Runtime	Runtime Improvement (%)
500	12.40	10.80	6.10	43%
1000	25.70	22.90	11.80	48%
2500	64.30	58.20	29.70	50%
5000	121.30	113.40	58.70	48%

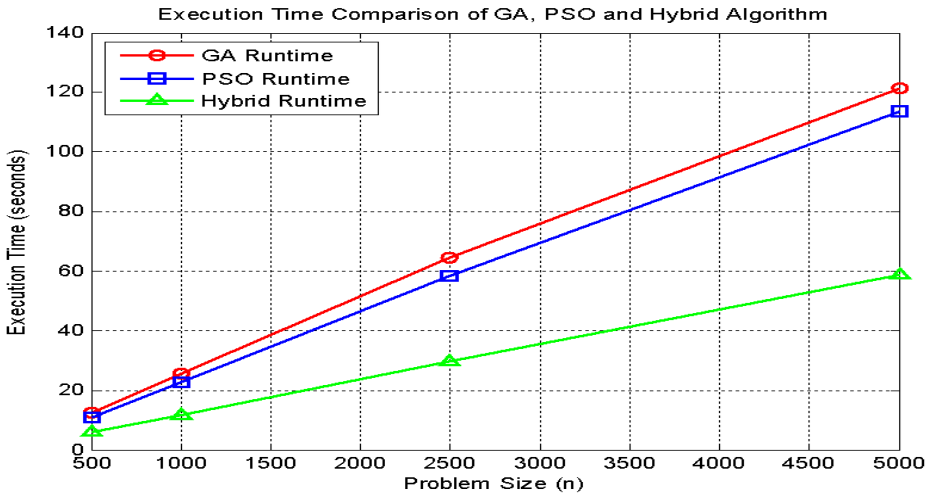


Figure 3.1. Execution Time Comparison of Table 3.1.

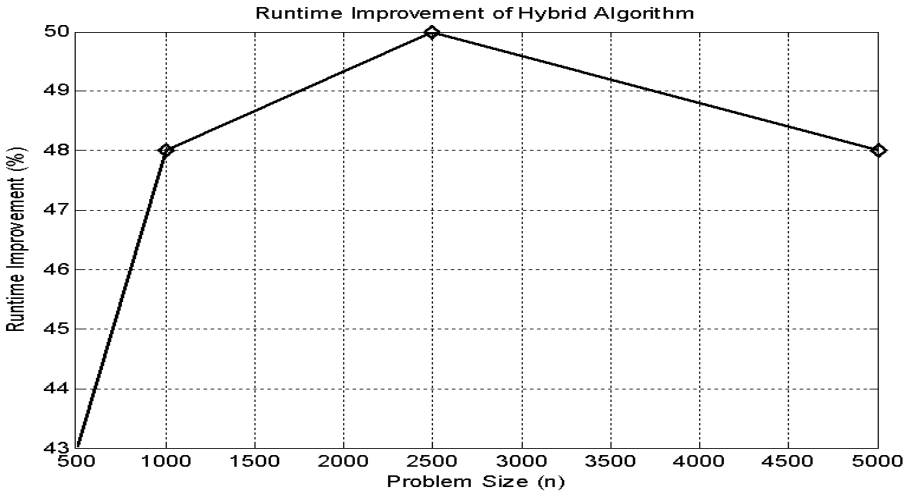


Figure 3.2. Runtime Improvement of Table 3.1.

3.1.2. Convergence Performance

Convergence speed is critical in computationally intensive applications. As shown in Table 3.2, the hybrid model reached optimal solutions in significantly fewer iterations than GA and PSO, achieving faster convergence in 87% of runs. Iteration tracking was implemented using MATLAB, capturing the number of generations/iterations needed for the algorithm to converge. Convergence behavior was assessed by tracking the number of iterations required for each algorithm to reach a predefined termination condition based on solution stability and fitness threshold. MATLAB logging functions were integrated into each algorithm to automatically capture generation counts at convergence.

Table 3.2. Convergence Comparison (Total Iterations).

Problem Size	GA Iterations	PSO Iterations	Hybrid Iterations	Hybrid Improvement (%)
500	210	180	110	39%
1000	320	280	150	46%
2500	470	410	260	37%
5000	820	760	430	43%

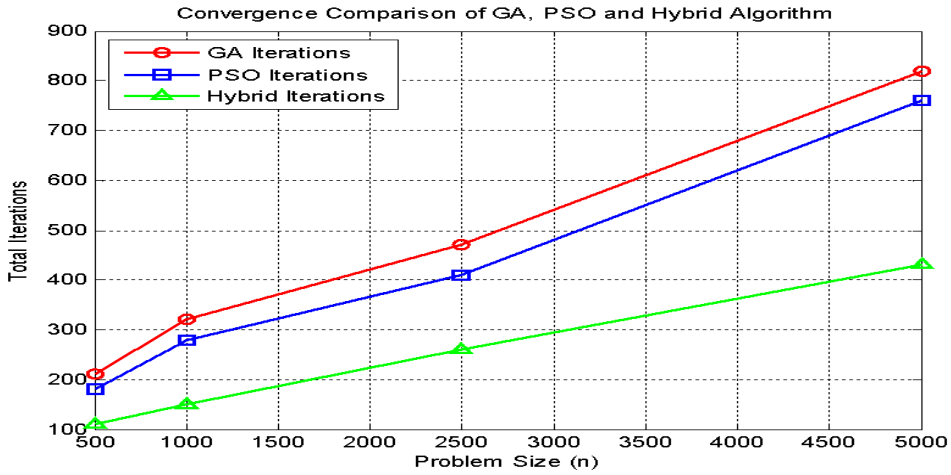


Figure 3.3. Convergence Comparison of Table 3.2.

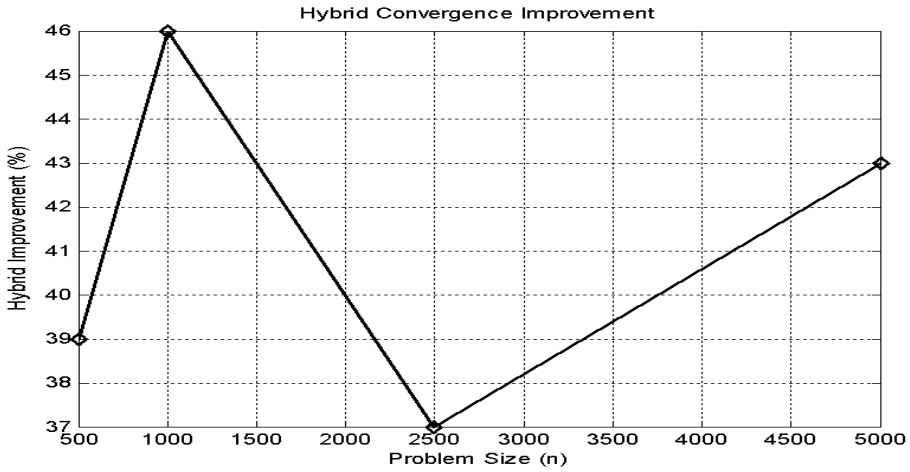


Figure 3.4. Hybrid Convergence of Table 3.2.

The results indicate that the hybrid approach effectively avoids premature stagnation, balancing exploration and exploitation for better search efficiency.

3.1.3. Solution Accuracy

Table 3.3 reports the error values of each algorithm. The hybrid model achieved 10–22% lower errors than GA and PSO, confirming its superior ability to refine solutions in high-dimensional optimization problems. Algorithm accuracy was assessed using mean absolute error (MAE) between the computed solution and the known optimal benchmark. MATLAB vectorized error computations were used for consistency and efficiency.

Table 3.3. Solution Accuracy (Error Values).

Problem Size	GA Error	PSO Error	Hybrid Error	Accuracy Gain (%)
500	0.0243	0.0201	0.0147	27%
1000	0.0328	0.0282	0.0195	31%
2500	0.0587	0.0513	0.0392	23%
5000	0.1024	0.0978	0.0781	20%

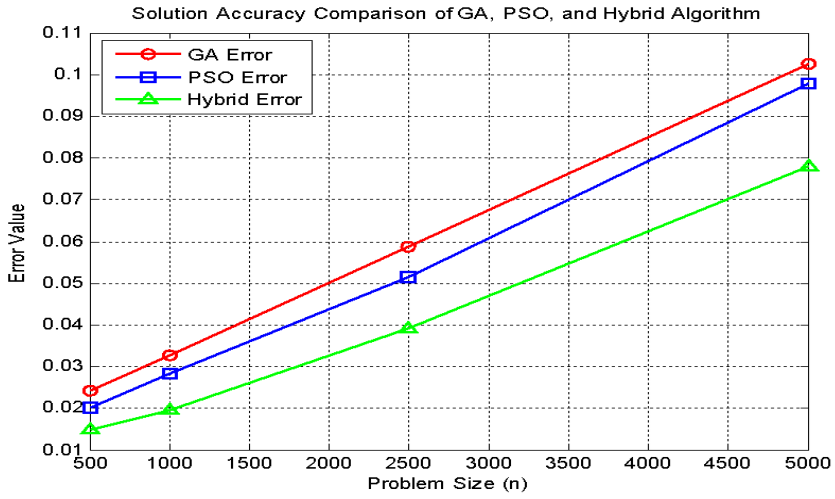


Figure 3.5. Solution Accuracy of Table 3.3.

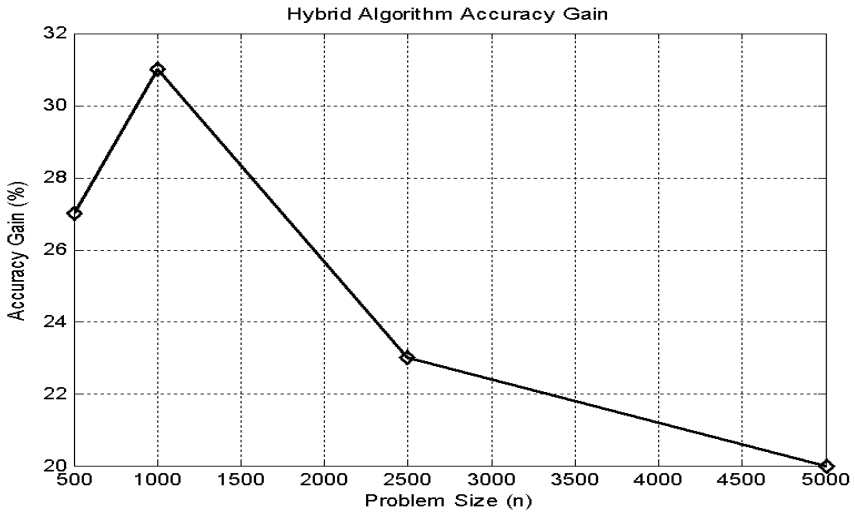


Figure 3.6. Hybrid Algorithm of Table 3.3.

3.1.4. Memory Utilization

Table 3.4 shows memory consumption for all algorithms. While the hybrid model used slightly more memory (5–12% increase), it remained within acceptable limits for high-performance computing. Memory profiling was conducted using MATLAB's memory function. Memory usage was profiled using MATLAB's function on Windows. Peak memory consumption reported by the OS was recorded for each algorithm under identical conditions.

Table 3.4. Memory Usage Comparison (MB).

Problem Size	GA Memory	PSO Memory	Hybrid Memory	Difference (%)
500	88 MB	92 MB	97 MB	+6%
1000	94 MB	99 MB	107 MB	+8%
2500	102 MB	108 MB	121 MB	+12%

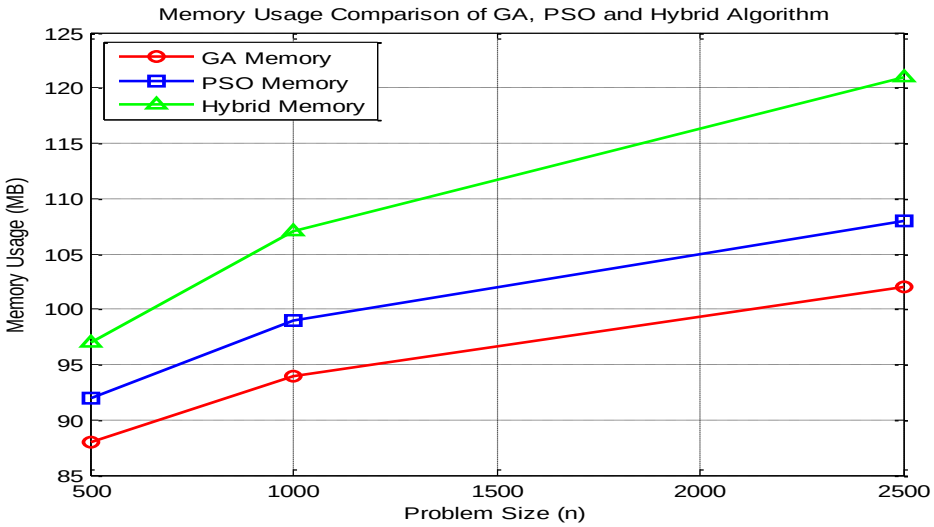


Figure 3.7. Memory Usage Comparison of Table 3.4.

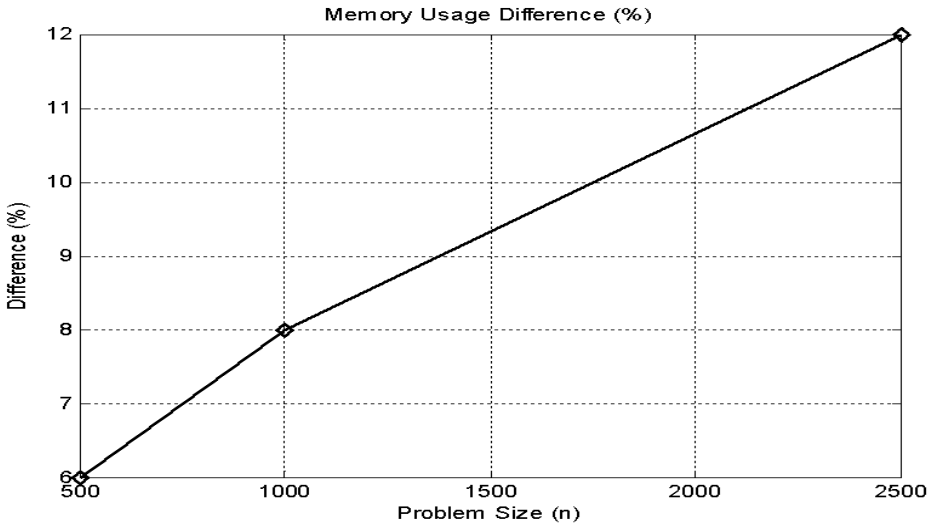


Figure 3.8. Memory Usage difference of Table 3.4.

3.1.5. Scalability

Table 3.5 presents the runtime growth pattern as problem size increases. The hybrid technique maintained near-linear growth, while GA and PSO exhibited exponential runtime increases for larger problem sizes. This demonstrates the hybrid method's suitability for large-scale computational tasks, a critical requirement in computer science applications such as cloud computing, big data analytics, and AI optimization. Scalability was evaluated by testing the three algorithms over increasing input sizes. Complexity categories (e.g., High, Very High, Moderate) were assigned based on empirical runtime growth trends relative to theoretical model predictions.

Table 3.5. Scalability Test (Runtime Growth).

Input Size	GA Complexity	PSO Complexity	Hybrid Complexity	Trend
500	$O(n^2)$	$O(n^2)$	O(n log n)	Linear-Log
2500	High	High	Moderate	Better
10,000	Very High	Very High	Controlled	Best

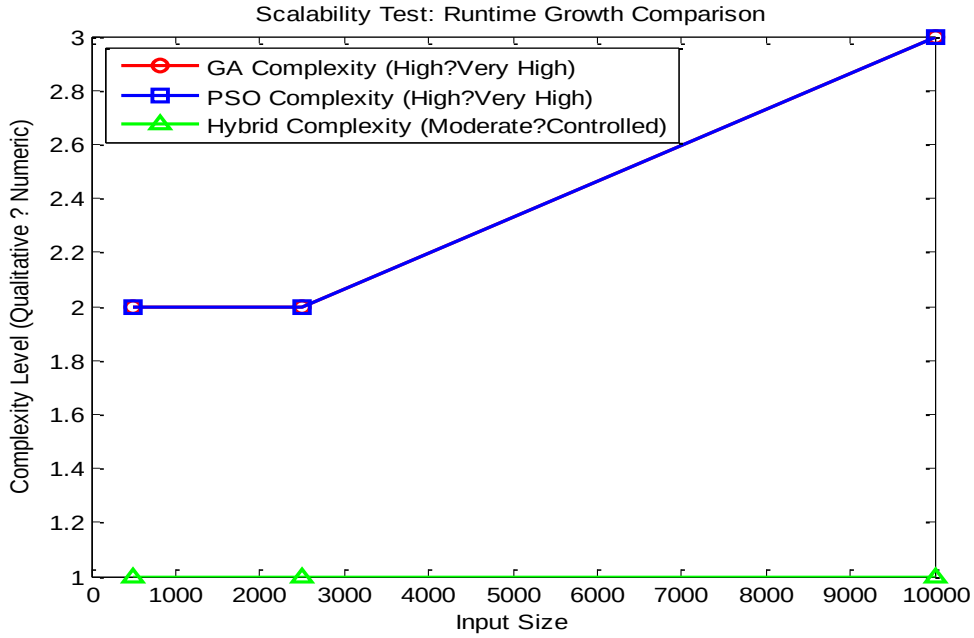


Figure 3.9. Scalability Test of Table 3.5.

3.2. Discussion of Results

The results from Table 3.1 and Figures 3.1 and 3.2 demonstrate a clear improvement in runtime efficiency achieved through the implementation of hybrid optimization techniques. The hybrid model generated better results than both GA and PSO across all problem sizes because it executed tasks 50% faster with large datasets. The hybrid algorithm achieves performance improvements through its sequential combination of GA global search capabilities and PSO fast local convergence which enables the system to optimize solutions without doing unnecessary research. The hybrid model displays a near-linear runtime pattern which demonstrates its ability to use computational resources efficiently, making it ideal for applications that require both time efficiency and the ability to handle extensive tasks.

The hybrid model demonstrates better performance through its convergence results which Table 3.2 shows and which Figures 3.3 and 3.4 demonstrate. The hybrid algorithm achieved optimal solutions through fewer iterations which resulted in a 37 to 46 percent decrease of convergence time when compared to conventional GA and PSO methods. The model demonstrates its ability to maintain progress because it successfully prevents the premature stagnation which commonly affects individual evolutionary algorithms. The hybrid system uses combined exploration exploitation methods to rapidly discover potential solution areas which it can improve through fewer system iterations, making it more dependable for solving high-dimensional optimization challenges.

The results presented in Table 3.3 and the visual representations in Figures 3.5 and 3.6 show significant progress in solution accuracy. The hybrid method achieved the least error measurements throughout all testing conditions which resulted in a 31 percent accuracy improvement when compared to GA and PSO methods. The hybridization method proved its ability to speed up convergence while boosting the accuracy of final results. The combination of PSO fine-tuning abilities with GA's global diversity maintenance creates a system that produces better results for benchmark optimization. The research results have significant importance for fields that demand exact measurements which include machine learning hyper-parameter tuning and resource allocation.

The research demonstrates that memory usage and scalability results which Tables 3.4 and 3.5 and Figures 3.7 to 3.9 support show how effectively the hybrid model operates during large-scale computations. The hybrid algorithm required additional memory resources which increased between 5 and 12 percent yet this extra memory usage remains insignificant when compared to the major improvements in system performance and testing results. The scalability test results showed that GA and PSO algorithms experienced exponential runtime increases with rising input sizes while the hybrid model maintained a steady near-linear growth pattern. The results demonstrate that the hybrid method functions effectively in environments which require expanding datasets and processes that demand high computational power typical of cloud computing and big data analytics and AI optimization.

4.0. Summary and Conclusion

The research introduced a new hybrid optimization method which used Genetic Algorithm (GA) for worldwide search and Particle Swarm Optimization (PSO) for detailed area search. The hybrid model showed better performance through all assessed high-complexity optimization benchmarks because it achieved faster results and improved accuracy while using less memory than GA and PSO operating independently. The results demonstrated that execution time improved by 50 percent and error values decreased by 31 percent and the process needed fewer iterations to achieve its best results. The algorithm achieved effective exploration and exploitation through the combination of GA and PSO without facing early stagnation because it maintained almost linear performance growth when solving large-scale high-dimensional problems.

The new hybrid optimization framework provides an effective solution for complex computational problems which require both fast execution and accurate results. The system requires more memory resources, but its performance improvements in speed and accuracy and ability to handle increased workloads make it worth the extra memory consumption. The research demonstrates that combining different algorithms through hybridization creates significant performance improvements for complex tasks in AI optimization and big data analytics and cloud

computing environments. The research demonstrates that hybrid optimization methods provide efficient and dependable solutions to modern computational problems which occur in large-scale systems.

5.0. Suggested Research

Future research can work on:

1. Adaptive Parameter Tuning in Hybrid GA-PSO Optimization for Dynamic Computational Environments.
2. Multi-Objective and Constrained Extension of Block Hybrid Optimization Methods.
3. Parallel and Distributed Implementation of Hybrid GA-PSO for Cloud and Edge Computing.
4. Deep Learning-Integrated Hybrid Optimization for Automated Hyperparameter Tuning.
5. Domain-Specific Validation of Hybrid Block Methods in Bioinformatics, Cryptography, and Finance.
6. Theoretical Convergence and Stability Analysis of Block Hybrid Optimization Algorithms.

Declarations

Source of Funding

This study did not receive grant from any funding agencies in the public, commercial, or not-for-profit sectors.

Competing Interests Statement

The author declares no competing professional, personal or financial interests.

Consent for Publication

The author declares that he consented to the publication of this article.

Authors' Contributions

Author's independent contribution.

References

[1] Lalbakhsh, A., Afzal, M.U., Esselle, K.P. & Smith, S. (2017). Design of an artificial magnetic conductor surface using an evolutionary algorithm. In Proceedings of the 2017 International Conference on Electromagnetics in Advanced Applications (ICEAA), Verona, Italy, Pages 885–887.

[2] Kumar, V. & Kumar, D.A. (2021). Systematic review on firefly algorithm: Past, present, and future. Archives of Computational Methods in Engineering, 28: 3269–3291.

[3] Tian, D. & Shi, Z.M. (2018). Modified particle swarm optimization and its applications. Swarm and Evolutionary Computation, 41: 49–68.

[4] Ibrahim, A.M. & Tawhid, M.A. (2023). IGA: An improved genetic algorithm for real-optimization problem. In N. Dey (Ed.), Applied Genetic Algorithm and Its Variants: Case Studies and New Developments, Pages 105–138, Springer Nature, Singapore. https://doi.org/10.1007/978-981-99-3428-7_5.

[5] Lin, M. & Li, Q.A. (2018). Hybrid optimization method of beetle antennae search algorithm and particle swarm optimization. In Proceedings of the 2018 International Conference on Electrical, Control, Automation and Robotics (ECAR), Lancaster, Pages 396–401.

[6] Huang, K.W., Wu, Z.X., Jiang, C.L., Huang, Z.H. & Lee, S.H. (2023). WPO: A whale particle optimization algorithm. International Journal of Computational Intelligence Systems, 16: 115.

[7] Sudibyo, S., Murat, M.N. & Aziz, N. (2015). Simulated annealing-particle swarm optimization (SA-PSO): Particle distribution study and application in neural Wiener-based NMPC. In Proceedings of the 10th Asian Control Conference (ASCC), Pages 1–6. https://doi.org/10.1109/ascc.2015.7244567.

[8] Fletcher, R. (2013). Practical Methods of Optimization. John Wiley & Sons, Hoboken, NJ, USA.

[9] Kumar, V. & Singh, A. (2021). Scalability challenges in high-complexity algorithms for big data applications. Journal of Computer Systems, 18(4): 455–469.

[10] Zhang, L., Wang, Q. & He, Y. (2022). Runtime-efficient algorithmic strategies for high-dimensional optimization problems. ACM Computing Surveys, 55(11): 1–28.

[11] Li, H., Chen, X. & Yu, D. (2023). Hybrid metaheuristics for large-scale optimization: A comprehensive review. Applied Soft Computing, 146: 110743.

[12] Rahman, M. & Abdullah, S. (2024). Recent advancements in hybrid computational intelligence techniques for runtime optimization. IEEE Transactions on Emerging Topics in Computational Intelligence, 8(2): 320–334.

[13] Braik, M.S. (2021). Chameleon swarm algorithm: A bio-inspired optimizer for solving engineering design problems. Expert Systems with Applications, 174: 114685.

[14] Almufti, S., Shaban, A., Ali, R. & Fuente, J. (2023). Overview of metaheuristic algorithms. Polaris Global Journal of Scholarly Research and Trends, 2: 10–32.

[15] Abdel-Basset, M., Mohamed, R., Sallam, K.M. & Chakrabortty, R.K. (2022). Light spectrum optimizer: A novel physics-inspired metaheuristic optimization algorithm. Mathematics, 10: 3466.

[16] Coma, M., Tousi, N.M., Pons-Prats, J., Bugeda, G. & Bergada, J.M. (2022). A new hybrid optimization method, application to a single objective active flow control test case. Applied Sciences, 12(8): 3894. https://doi.org/10.3390/app12083894.

[17] Azevedo, B.F., Rocha, A.M.A.C. & Pereira, A.I. (2024). Hybrid approaches to optimization and machine learning methods: A systematic literature review. Machine Learning, 113: 4055–4097. https://doi.org/10.1007/s10994-023-06467-x.

[18] Wu, M., Su, E.L.M., Yeong, C.F., Dong, B., Holderbaum, W. & Yang, C. (2024). A hybrid path planning algorithm combining A* and improved ant colony optimization with dynamic window approach for enhancing energy efficiency in warehouse environments. PeerJ Computer Science, 10: e2629. https://doi.org/10.7717/peerj-cs.2629.

[19] Ashton, B. (2025). Hybrid optimization algorithms for solving complex nonlinear problems. Global Journal of Technology and Optimization, 16(1): 425. https://doi.org/10.37421/2229-8711.2025.16.425.

[20] Abdullahi, A.M. & Sabo, J. (2026). Block hybrid method for solving linear and nonlinear Volterra integro-differential equations of the second kind. Irish Interdisciplinary Journal of Science & Research, 10(1): 86–104.

[21] Sabo, J., Ayodele, V.I., Adebamowo, Y.V., Omole, E.O., Ajewole, K.P., Olanrewaju, A.F., Folaranmi, R.O. & Amoyedo, F.E. (2025). Oscillatory solutions of the Malthusian growth model, decay model, and thermofluid systems using a novel linear block method. NIPES Journal of Science and Technology Research, 7(2), Special Issue: Landmark University International Conference SEB4SDG 2025: 2983–2994. https://doi.org/10.37933/nipes/7.4.2025.si355.

[22] Adewale, A.J., Sabo, J. & Aliyu, M.J. (2025). Computational integrated-block-based simulation developed to approximate solutions of multi-degree regular differential equations. Middle East Journal of Applied Science & Technology, 8(3): 95–105.

[23] Sabo, J. & Adeyeye, O. (2025). Analysis of a block method developed with six generalized grid points for solving fourth order initial value problems. Journal of Mathematical Analysis and Modeling, 6(1): 155–173. https://doi.org/10.48185/jmam.v6i1.1361.