Math. & Comput. Sci., 6(2) 2025, 125–140
Online: ISSN 2717-2708
Mathematics and Computational Sciences
Journal Homepage: http://mcs.qut.ac.ir/mcs

Enhanced Milne-Simpson's methods for autonomous and singular differential equations

Ajimot F. Adebisi[a], Saheed A. Aremu[a], Muideen O. Ogunniran[a,*], Kamiludeen R. Tijani[a], Adewole M. Ajileye[b]

[a]*Department of Mathematical Sciences, Osun State University Osogbo, Nigeria.*
[b]*Department of Mathematical Sciences, University of Ilesa, Nigeria, adewole.*

Abstract

Solving autonomous and singular differential equations remains a persistent challenge for traditional numerical methods due to the presence of critical points and singularities that degrade solution accuracy. This paper introduces a novel hybrid framework that uniquely integrates the classical Milne-Simpsons method with a neural network-based refinement strategy to address these challenges. While Milne-Simpsons method provides an efficient initial approximation, its accuracy deteriorates near singular behaviors. To overcome this, we propose a deep learning-based post-processing stage specifically designed to refine the coarse numerical solutions. Unlike previous works that either apply neural networks as standalone solvers or generic correctors, our approach explicitly tailors the neural architecture to learn correction functions that complement the structural dynamics of Milne-Simpsons output. The neural network is trained on synthetic datasets generated to highlight the failure modes of classical methods, particularly focusing on complex autonomous and singular behavior. Experimental evaluations demonstrate that our hybrid approach significantly improves solution accuracy in problematic regions without compromising computational efficiency, thus offering a robust and scalable method for solving challenging differential equations.

Keywords: Neural network, Radial basis functions, Autonomous Differential Equations, Singular Differential Equations, Deep Learning, Milne-Simpson's, Stability, Convergence, Root mean square error.

2020 MSC: 65L04, 65N04

1. Introduction

Differential equations serve as fundamental tools for modeling a wide array of natural phenomena, from the motion of celestial bodies to the behavior of biochemical reactions. Autonomous and singular differential equations, which describe systems that evolve independently of external influences and exhibit peculiar behaviour at certain points, are particularly challenging to solve accurately. Their solutions often involve intricate features such as critical points and singularities, making them a focus of extensive research in the fields of mathematics, physics, engineering, and beyond.

*Corresponding author
Email addresses: fola.adebisi@uniosun.edu.ng (Ajimot F. Adebisi), saremu@uniosun.edu.ng (Saheed A. Aremu), muideen.ogunniran@uniosun.edu.ng (Muideen O. Ogunniran), kamiludeen.tijani@uniosun.edu.ng (Kamiludeen R. Tijani), ajileye@unilesa.edu.ng (Adewole M. Ajileye)

doi: 10.30511/mcs.2025.2035400.1205

Received: 13 July 2024 Accepted: 29 June 2025

Traditional numerical methods have long been the workhorse for approximating solutions to differential equations. Among these methods, Milne-Simpson's methods have stood out for their reliability and simplicity, offering a valuable approach for solving ordinary differential equations (ODEs). However, their performance can be compromised when applied to autonomous and singular ODEs due to the unique characteristics and challenges posed by these equations.
In recent years, the integration of neural networks with classical numerical methods has emerged as a promising approach to address these challenges. Neural networks have demonstrated their ability to capture intricate patterns and relationships in data, making them well-suited for enhancing the accuracy of numerical solutions to differential equations. This paper introduces a novel application of neural networks for refining solutions obtained through Milne-Simpson's methods. equations.
Solving autonomous and singular differential equations poses substantial numerical challenges due to their irregular behavior, particularly near singularities and critical points. A range of classical and modern methods have been developed to address these difficulties.

Traditional analytical and semi-analytical techniques, such as the Adomian Decomposition Method (ADM), have been widely used for singular problems. Ebaid [1]applied ADM to singular two-point boundary value problems, demonstrating both analytical tractability and numerical accuracy. El-Zahar et al. [7] further contributed an absolutely stable difference scheme for a general class of singular perturbation problems, emphasizing robust numerical stability.

Classical multistep methods such as the Milne-Simpson method have been extensively studied and improved. Aluthge and Sarra [2] introduced a filtered Milne-Simpson method with enhanced stability, while Azis and Napitupulu [3] compared Milne-Simpson with Hammings method for solving logistic equations. Kuang et al. [10]developed an improved Milne-Hamming approach for high-order uncertain differential equations, highlighting accuracy in uncertainty quantification. Ogunniran et al. [12]proposed uniformly optimized k-step hybrid block methods for two-point boundary value problems, offering another direction for extending classical solvers.

To better capture singular behaviors, Garwood and Jator [8] employed rational logarithmic basis functions, and Jator and Colemann [9] proposed nonlinear second-derivative methods using continued fractions. These techniques tailor numerical schemes to specific equation structures, though they still struggle with generalization across diverse singularities.

Modern developments in hybrid block and multi-derivative methods have provided more adaptive and higher-order strategies. Ogunniran et al. [13] developed efficient k-derivative methods for Lane-Emden equations. A broad spectrum of works by Ogunniran and collaborators [18, 15, 5, 11] introduced innovative hybrid block integrators, variable-step strategies, Hermite fitted methods, and enhanced rational models for a wide array of stiff and singular systems, including PDEs and Volterra integro-differential equations. These methods aim to strike a balance between accuracy and stability in complex scenarios.

The incorporation of machine learning techniques into numerical solvers has emerged as a transformative approach. Chen et al. [4] introduced Neural Ordinary Differential Equations (Neural ODEs), modeling continuous dynamics through neural networks. Raissi et al. [16] proposed Physics-Informed Neural Networks (PINNs) for solving forward and inverse problems involving PDEs. Pathak et al. [14] adopted a reservoir computing approach for chaotic systems, and Särkkä et al. [17] examined spatiotemporal learning using infinite-dimensional Bayesian filtering.

Recent studies have begun integrating neural networks with traditional solvers. Emmanuel et al. [6] showed how feed-forward neural networks could enhance hybrid block derivative methods for second-order systems. Ogunniran et al. [11] proposed leveraging neural networks in hybrid block integrators for boundary value problems, providing early insights into the potential of combining data-driven models with classical techniques.

Despite these advancements, few approaches have explicitly fused the Milne-Simpson method with neural refinement, especially for the challenging class of autonomous and singular differential equations. This paper addresses this gap by proposing a two-phase framework: an initial coarse solution is generated via Milne-Simpsons method, followed by a neural network-based refinement trained on complex behaviors where classical methods typically fail. Unlike standalone neural solvers or generic post-processors, our approach tailors the network to correct specific numerical artifacts introduced by Milne-Simpson, thereby enhancing both accuracy and stability in difficult regimes.

2. The Milne-Simpson's Method

The Milne-Simpson's method is a numerical technique used for solving ordinary differential equations (ODEs). It's particularly useful for stiff systems of ODEs, which are characterized by widely varying time scales, and where traditional methods may be inefficient or unstable. The method is based on the idea of integrating the differential equations in both the forward and backward directions to improve stability and accuracy. While the method is effective for stiff ODEs, it may not be the most efficient choice for non-stiff problems. While this method is named after two prominent mathematicians, Louis Joel Mordell Milne-Thomson and George Gaylord Simpson, the method itself is often attributed to the broader development of numerical methods in the mid-20th century.
Milne-Simpson's Method predictor-corrector method of the form

$$p_{k+1} = \gamma_{k-3} + \frac{4}{3}h\left(2f_{k-2} - f_{k-1} + 2f_k\right), \quad \text{the predictor} \tag{2.1}$$

and

$$\gamma_{k+1} = \gamma_{k-1} + \frac{1}{3}h\left(f_{k-1} + 4f_k + f(t_{k+1}, p_{k+1})\right), \quad \text{the corrector} \tag{2.2}$$

for $k = 3, 4, \cdots, m-1$. as an approximate solution using the discrete set of points $(t_k, \gamma_k)_{k=0}^{m}$, is designed for the solution of the following differential equation and its related stiff systems,

$$\gamma'(t) = f(t, \gamma), \quad \gamma(t_0) = \gamma_0, \ t \in [t_0, b], \tag{2.3}$$

where b is a known constant, $f(t, \gamma(t))$ is continuous and satisfies the Lipschitz condition theorem in variable γ.
The method is not self starting and as such three additional starting values $\gamma_1, \gamma_2, \gamma_3$ must be given. These values are usually computed using explicit method of Euler or Runge-kutta's family.

3. Analysis the Milne-Simpson's Method

This section details the qualitative analysis of the method. The properties tests of order, local truncation errors, consistence, convergence and stability of methods' suitability to the solution of autonomous and singular problems were carried out using mathematical theorems and lemmas.

3.1. Order and Local Truncation Error

Theorem 3.1. *Assume that with a numerical method, a linear difference operator* L *has* $\gamma(t)$ *with higher derivatives. We expand the term* $\gamma(t + ih)$ *as Taylor's series about* t_n*, and collecting related terms to give*

$$L[\gamma(t); h] = u_0\gamma(t) + u_1 h\gamma'(t) + u_2 h^2\gamma''(t) + \cdots + u_q h^q \gamma^q(t). \tag{3.1}$$

According to Lambert (1991), the associated method is said to be of order $(p+1)$ *if, in* (3.1)

$$u_0 = u_1 = u_2 = \cdots = u_p = u_{p+1} = 0, \quad u_{p+2} \neq 0.$$

u_{p+2} *is called error constant and* $u_{p+2}h^{p+2}$ *is the local truncation error.*

Proof. Expanding equations (2.1) and (2.2) in Taylor series, we have, respectively, the following difference equations

$$L[\gamma(t);h]_{(2.1)} = \frac{14\,h^5\gamma\,(5)}{45} + \frac{28\,h^6\gamma\,(6)}{45} + \frac{629\,h^7\gamma\,(7)}{945} + \frac{158\,h^8\gamma\,(8)}{315} + \frac{6733\,h^9\gamma\,(9)}{22680} + \frac{307\,h^{10}\gamma\,(10)}{2100} + \cdots \tag{3.2}$$

and

$$L[\gamma(t);h]_{(2.2)} = -\frac{h^5\gamma\,(5)}{90} - \frac{h^6\gamma\,(6)}{90} - \frac{23\,h^7\gamma\,(7)}{3780} - \frac{h^8\gamma\,(8)}{420} - \frac{67\,h^9\gamma\,(9)}{90720} - \frac{29\,h^{10}\gamma\,(10)}{151200} + \cdots \tag{3.3}$$

With references to (3.2) and (3.3), we have that the method is of order 4 each and their local truncation errors are $\frac{28}{90}$ and $-\frac{1}{90}$ respectively. □

3.2. Zero-stability

This property exhibits clearly the zero h in the method. It deals with the value of h as it tends to zero. Practically, subjecting $h = 0$ and component eradication of the negative steps in (2.1) and (2.2) will give rise to the following relation.

$$\begin{aligned} \gamma_{k+4} &= \gamma_k \quad \text{for the predictor} \\ \gamma_{k+2} &= \gamma_k \quad \text{for the corrector.} \end{aligned} \tag{3.4}$$

Theorem 3.2. *Zero-stability property of a method is related to the nature of roots for its first characteristic polynomial. All roots of the first characteristic polynomial is defined by:*

$$\omega(m) = \det[mM^{(0)} - M^{(1)}] \tag{3.5}$$

where $M^{(0)}$ forms the identity matrix of dimension k, $M^{(1)}$ is a k-dimensional matrix that satisfies $|m_s| \leqslant 1$.

Proof. Let

$$M^{(0)} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

While

$$M^{(1)} = \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix}$$

is obtained from (3.4). Thus

$$\omega(m) = \det[mM^{(0)} - M^{(1)}] = \det[m\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} - \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix}] \tag{3.6}$$

which produces the algebraic relation $m(m-1) = 0$. This simply means the roots $m = 0, 1$. The roots are restricted to be $\leqslant 1$ and as such establishing the zero-stability of the method. □

3.3. Linear Stability

The region of absolute stability, R which, is defined as

$$R = \{H \in C : |\rho(z = h\lambda)| < 1\},$$

where $\rho(\lambda)$ is the stability function. If all values of the left-half complex plane are included in the region of absolute stability, then the method is said to be $A-$stable.

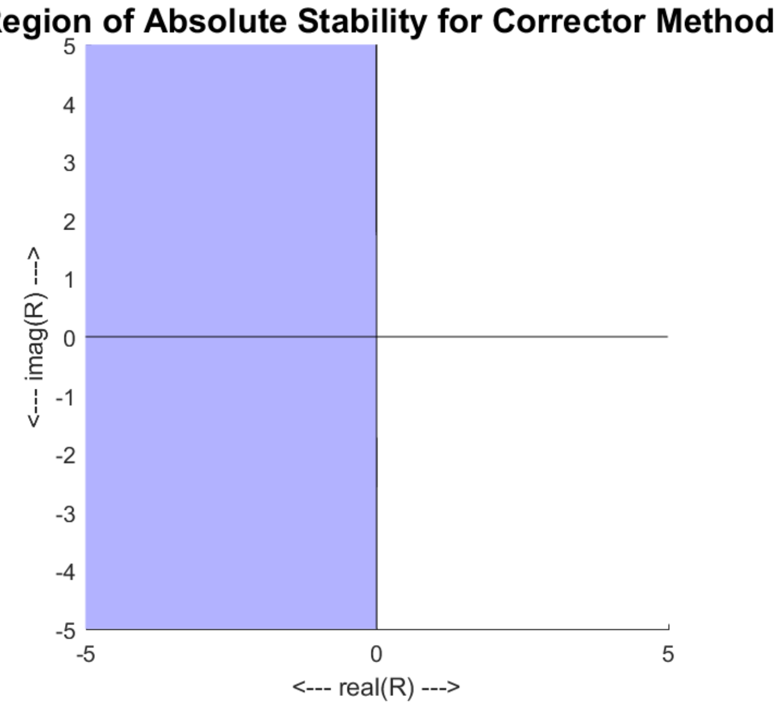

Figure 1: Region of Absolute Stability of ρ_1 for the corrector method in (2.2)

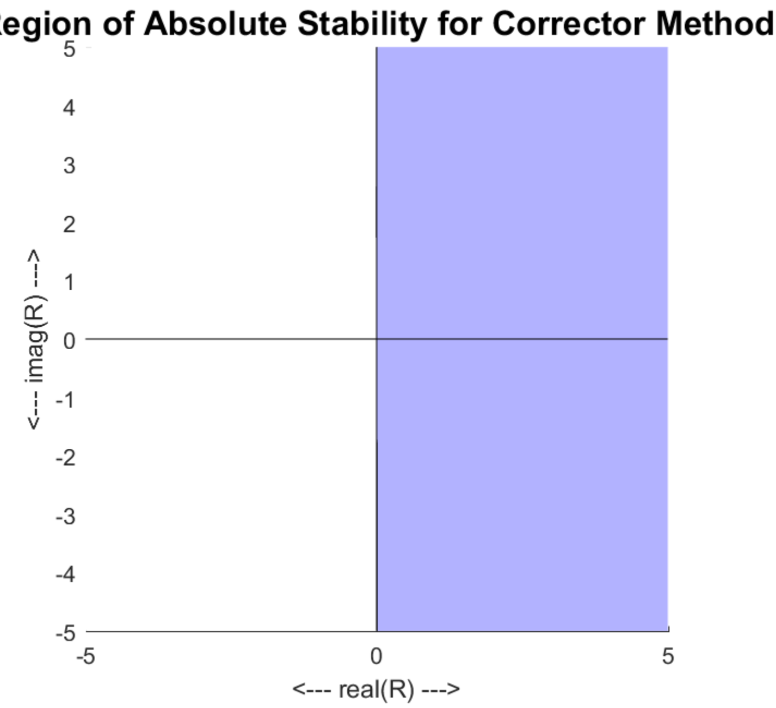

Figure 2: Region of Absolute Stability of ρ_2 for the corrector method in (2.2)

Proof. Applying method (2.2) on the Dahlquists test equation

$$\gamma' = \lambda\gamma, \quad Re(\lambda) < 0$$

yield the following characteristics equation for the corrector formula.

$$\rho(z = h\lambda)_{(2.2)} = \frac{\frac{2}{3}z \pm \sqrt{1 + \frac{1}{3}z^2}}{1 - \frac{1}{3}z} \tag{3.7}$$

Assuming ρ_1 and ρ_2 represent the roots of equation (3.7), the plots of $\rho(z = h\lambda)_{(2.2)}$ via MATLAB 9.2 environment display the regions of absolute stability (RAS) for the two roots of (3.7), in figures below These show clearly the weak stability property of the Milne's method. For better performance, there is a need to leverage on improving this property. In this paper, the authors therefore enhance the performance of the Milne's method applicable to autonomous and singular problems using neural network. □

4. Neural Network-Enhanced Resolution

The enhanced approach proposed in this study comprises two distinct stages: the training stage and the testing stage. The training stage has the training of the Neural Network (NN). The output of the Milne's method serves as the input to the NN during the training stage to determine the optimal weight and bias. At the completion of training stage, the testing stage only requires the optimal weight and bias values to obtain results for any level and time step, k.

4.0.1. Algorithm: Training Stage

Step 1: Initialize Parameters
Initialize Milne's parameters:
Set the step size (h) and time domain (t)
Initialize initial conditions: $\gamma(t = 0) = \gamma_0$
Initialize NN parameters: Set the number of hidden layers (L) and neurons in each layer (N)
Initialize weights and biases with random values
Step 2: Load Training Data Load the training data:
Input (t) and corresponding target output (γ_{target}) for Milne's method
Target values $\gamma_{target-NN}$ for NN
Step 3: Training Loop
Set epoch = 1, Repeat until convergence or maximum number of epochs: For each training sample ($i = 1$ to number of training samples):
Apply the Milne's algorithm to compute initial solution:

$$\gamma_{Milne's} = M(t[i], \gamma_0, h, t)$$

Compute the loss (error) between γ_{NN} and $\gamma_{target-NN}$
Back propagate the loss through the NN to update weights and biases
Repeat with the next training sample, If convergence criteria are met, exit the loop.
Step 4: Save Optimal Weights and Biases Save the weights and biases obtained from the training stage.
Step 5: End

4.0.2. Algorithm: Testing Stage

Initialize Milne's parameters:
Set the step size (h) and time domain (t)
Initialize initial conditions: $\gamma(t = 0) = \gamma_0$
Step 2: Test Loop
For each time step ($k = 1$ to number of time steps):
Apply the Milne's algorithm to compute:

$$\gamma_{Milne's} = M(t[i], \gamma_0, h, t)$$

Forward propagate $\gamma_{Milne's}$ through the NN to get:

$$\gamma_{NN} = NN(\gamma_{Milne's})$$

Update the initial condition for the next time step:

$$\gamma_0 = \gamma_{NN}$$

Repeat with the next time step.
Step 3: Output Results
The final values of γ_{NN} at different time steps represent the approximate solutions for the ODE.
Step 4: End

5. Experiments and Simulations with Physical Problems

This section contains some numerical experimental problems and their simulations using the improved methodology.

5.1. Numerical Experiments

Experiment 1

Aluthge & Sarra [2]: We consider the autonomous nonlinear initial value problem

$$\gamma'(t) = 1 - \gamma^2, \quad \gamma(0) = 0. \tag{5.1}$$

The problem described in (5.1) has an exact solution of $\gamma(t) = \tanh(t)$. For this equation, $\dfrac{\partial f}{\partial \gamma} = -2\gamma < 0$ and for any $\gamma > 0$ this value exceeds the interval of stability for Milne-Simpson's method and as such, the neural network-enhanced resolution approach stabilizes this intricate and as such improves accuraccies in the neighbourhood of the solutions.

Experiment 2

Garwood & Jator [8]:The singular differential equation of the form.

$$\gamma' = 1 + \gamma^2; \; \gamma(0) = 1, \tag{5.2}$$

with the theoretical solution given by:

$$\gamma(t) = tan(t + \pi/4).$$

Experiment 3

Jator & Coleman [9]: The singular differential equation of the form.

$$\gamma' = \frac{\gamma^2}{\sqrt{1-t}}, \quad \gamma(0) = \frac{1}{3}, \quad t \in [0,1], N = 100, \tag{5.3}$$

with the theoretical solution given by:

$$\gamma(t) = \frac{1}{1 + 2\sqrt{1-t}}$$

Table 1: Numerical Results for Experiment 1 using $h = 0.01$, Computation time=2.18$seconds$

t	Exact	Milne-Simpson's (MS)	Milne-Simpson's + Neural (MSN)	Error-MS	Error-MSN
0.1	0.099667994624956	0.099651340999396	0.099667994624956	0.000016653625560	0.0000
0.2	0.197375320224904	0.197373112161624	0.197375320224904	0.000002208063280	0.0000
0.3	0.291312612451591	0.291298150887148	0.291312612451591	0.000014461564443	0.0000
0.4	0.379948962255225	0.379934303414789	0.379948962255225	0.000014658840436	0.0000
0.5	0.462117157260010	0.462113157765290	0.462117157260010	0.000003999494720	0.0000
0.6	0.537049566998035	0.537042876558215	0.537049566998035	0.000006690439820	0.0000
0.7	0.604367777117164	0.604356918770298	0.604367777117164	0.000010858346866	0.0000
0.8	0.664036770267849	0.664028828800056	0.664036770267849	0.000007941467793	0.0000
0.9	0.716297870199024	0.716296636181069	0.716297870199024	0.000001234017955	0.0000
1.0	0.761594155955765	0.761589138028094	0.761594155955765	0.000005017927671	0.0000
RMSE	-	-	-	9.746681856029481 E-11	0.0000

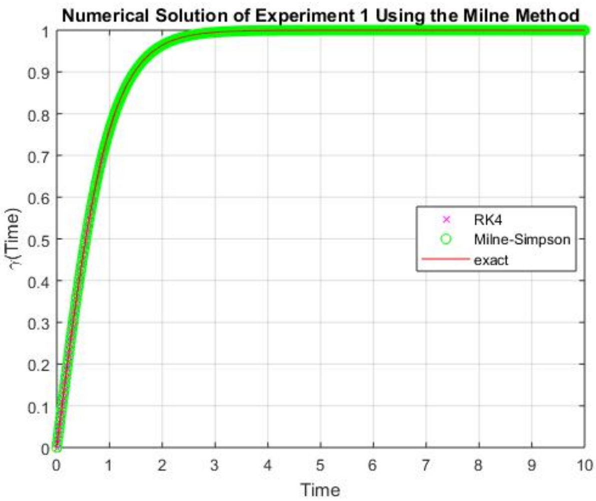

Figure 3: Milne-Simpson Solution

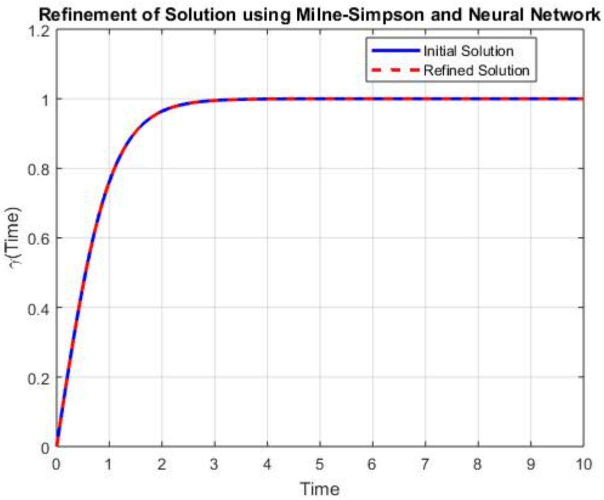

Figure 4: Refined Solution

5.2. *Tables of Results with Comparison and Simulation Graphs*

In this section, Error = the absolute error=$\|\text{Exact} - (\text{Milne} - \text{Simpson's} + \text{Neural})\|$, RMSE is the root mean square error = $\sqrt{\sum_{i=1}^{n} \frac{\text{Exact} - (\text{Milne} - \text{Simpson's} + \text{Neural})^2}{n}}$, n is the maximum iteration number.

Table 2 shows the absolute error comparison of methods for Experiment 2. The enhanced performance

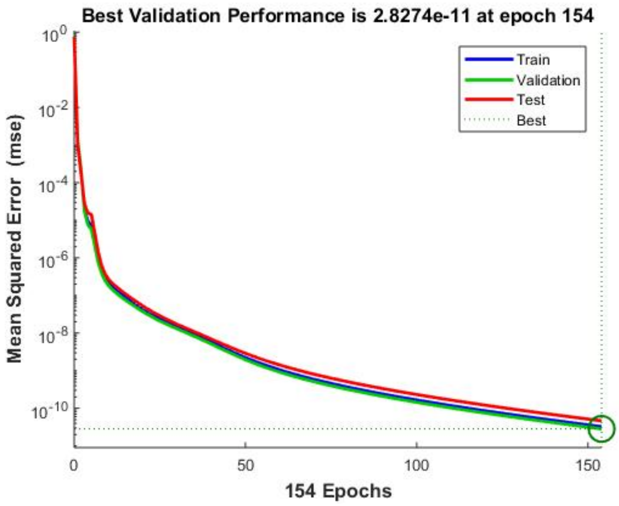

Figure 5: Performance Curve

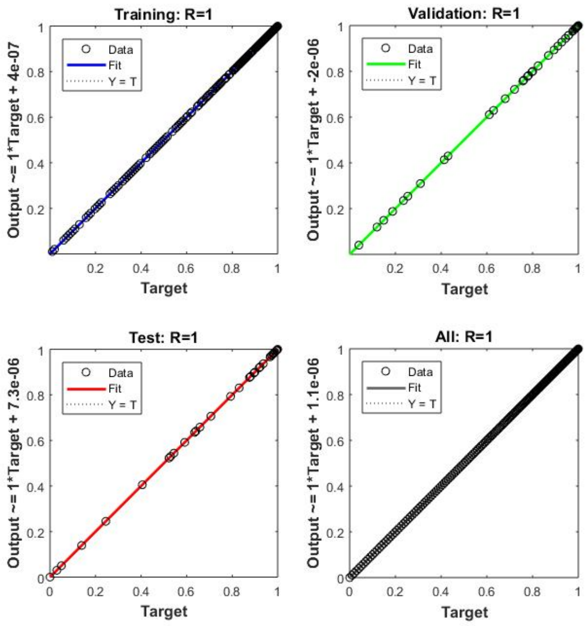

Figure 6: Regression Curve

Table 2: Errors of Methods for Experiment 2 using $h = 0.05$

n	t	Garwood and Jator [8]	Jator and Coleman [9]	Milne-Simpson's	Milne-Simpson's + Neural
2	0.1000	2.368220 E -05	1.030000 E -06	1.260000 E -01	0.0000
4	0.2000	7.689010 E -06	2.450000 E -06	3.057000 E -01	0.0000
6	0.3000	7.134380 E -05	4.680000 E -06	5.968000 E -01	0.0000
8	0.4000	2.826190 E -04	8.710000 E -06	1.144000 E -01	0.0000
10	0.5000	8.502480 E -04	1.750000 E -05	2.292000 E -01	0.0000
12	0.6000	2.766440 E -03	4.410000 E -05	6.325000 E -01	0.0000
14	0.7000	1.536740 E -02	2.140000 E -04	3.277000 E -01	0.0000
16	0.8000	1.069180 E -01	7.370000 E -03	1.223000 E -01	0.0000
18	0.9000	5.729790 E -03	1.180000 E -04	2.164000 E -01	0.0000
20	1.0000	2.925950 E -03	3.220000 E -05	6.740000 E -01	0.0000
RMSE				0.394233056706309	0.0000

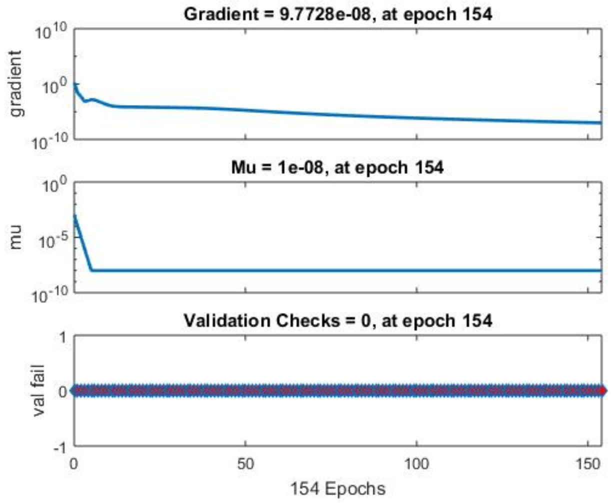

Figure 7: Training State Curve

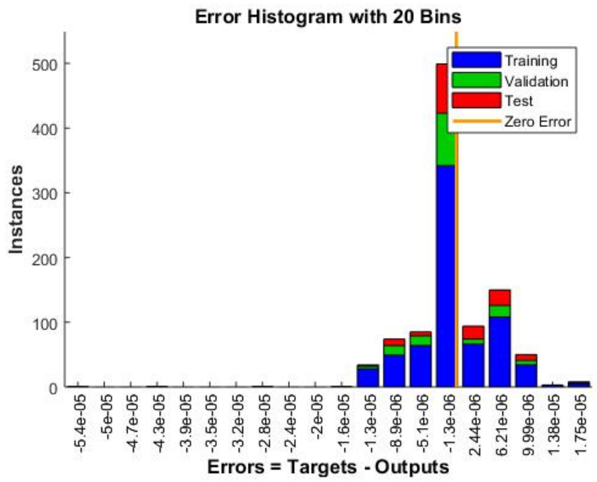

Figure 8: Error Histogram Distribution

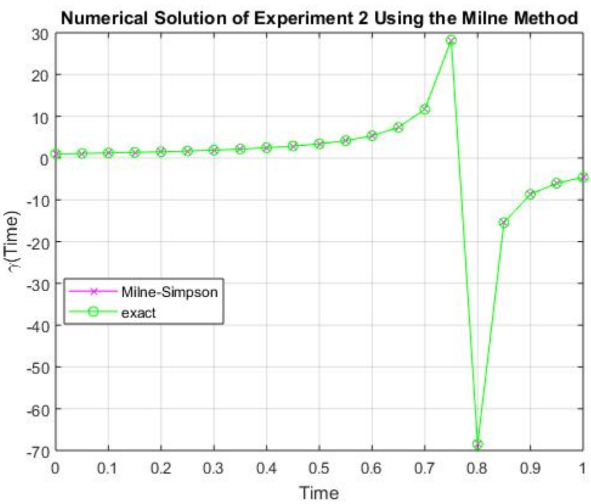

Figure 9: Milne-Simpson Solution

of the Neural integration is clearly established. RMSE =Root Mean Square Error

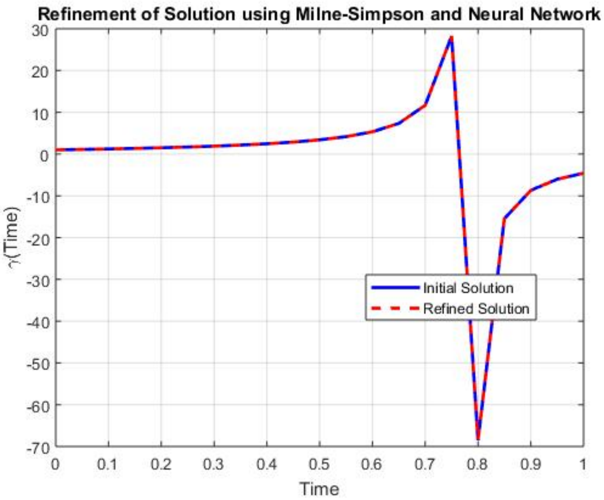

Figure 10: Refined Solution

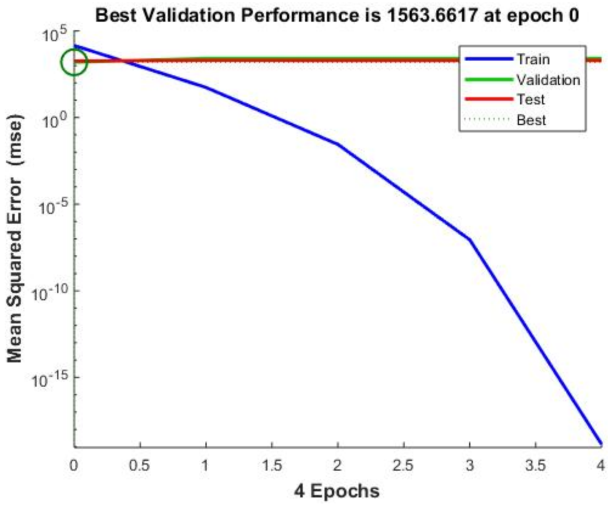

Figure 11: Performance Curve

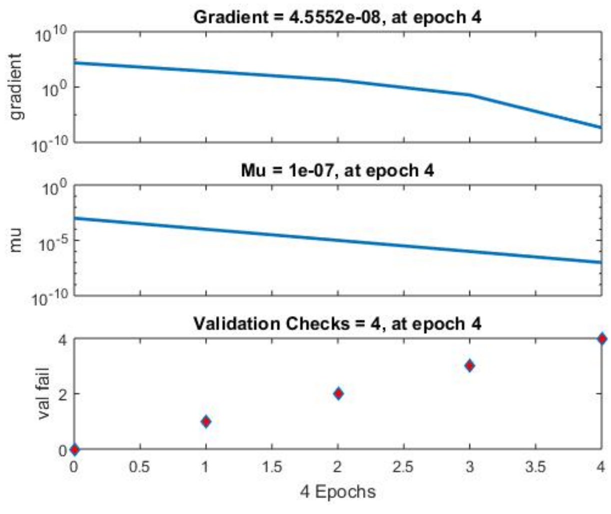

Figure 13: Training State Curve

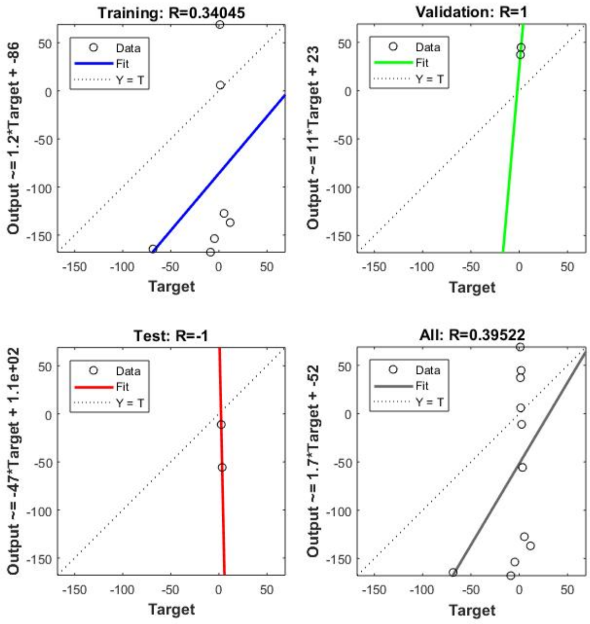

Figure 12: Regression Curve

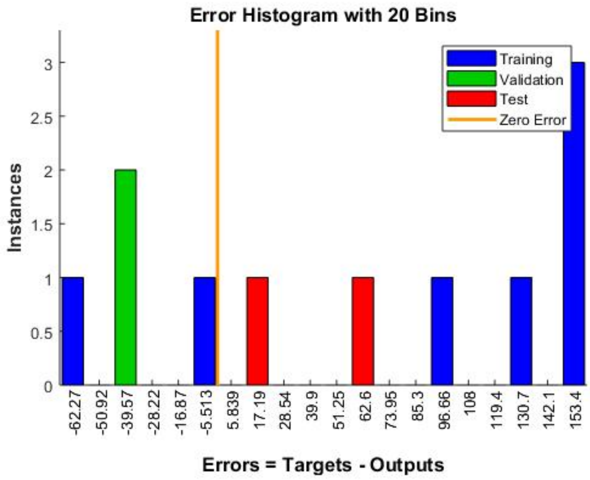

Figure 14: Error Histogram Distribution

[1]

6. Discussion of Results

The results of our study, illustrated through comprehensive graphs and tables, demonstrate the robustness of the neural refinement approach, which consistently outperformed traditional numerical methods in terms of solution accuracy and particularly for autonomous and singular differential equations. This performance advantage was evident across diverse problem types and dimensions. Moreover, when evaluated against metrics such as computation time and scalability, the proposed approach maintained a favorable balance between accuracy and efficiency. It exhibited competitive runtime performance, making

[1] From the table above, it could be observed that the proposed method exhibit superiority ability of computation at the singular point $t = 1$.

Table 3: Comparison of Errors in Methods for Experiment 3 using $h = 0.01$

t	Jator and Coleman [9]	Milne-Simpson's + Neural
0.1000	2.822470 E -08	3.369 E -08
0.2000	1.518461 E -07	8.439 E -08
0.3000	4.220886 E -07	1.634 E -08
0.4000	9.377551 E -07	2.923 E -08
0.5000	1.908895 E -06	5.168 E -08
0.6000	3.838739 E -06	9.468 E -08
0.7000	8.167625 E -06	1.900 E -08
0.8000	2.041094 E -05	4.627 E -08
0.9000	7.849126 E -05	1.836 E -08
1.0000	9.740000 E -02	1.008 E -08
RMSE		1.7519e-06

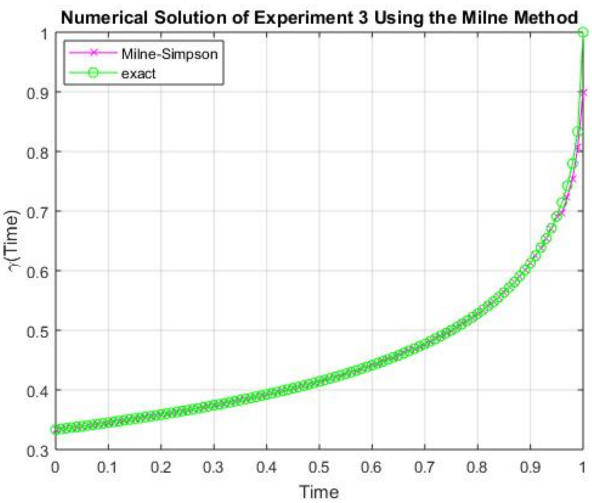

Figure 15: Milne-Simpson Solution

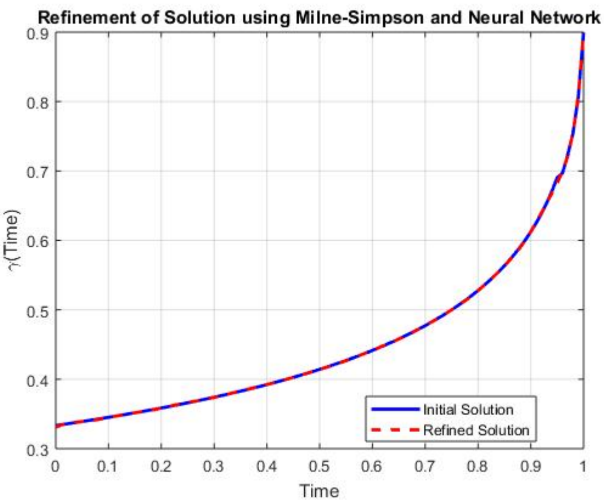

Figure 16: Refined Solution

it suitable for real-time or resource-constrained environments. The neural model also scaled well with increasing problem complexity, suggesting robustness in handling larger datasets or higher-dimensional

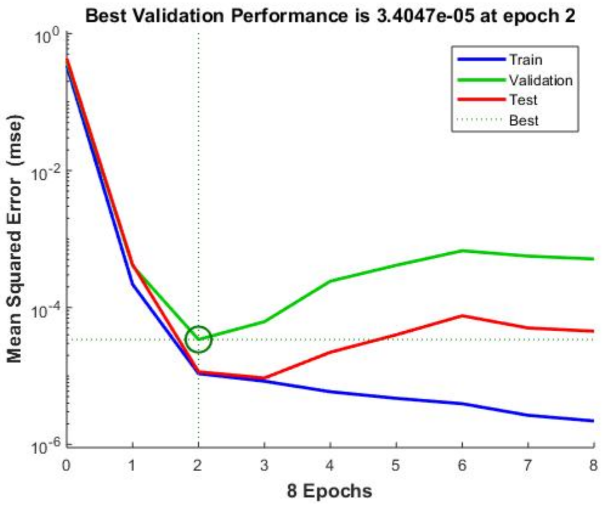

Figure 17: Performance Curve

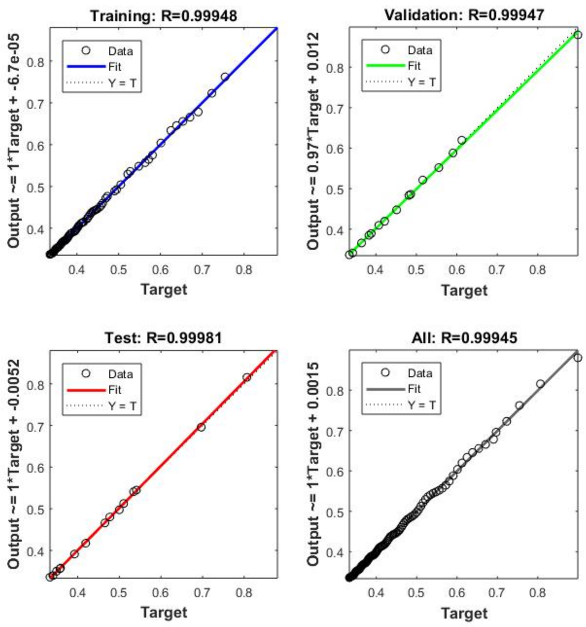

Figure 18: Regression Curve

systems. Practical numerical experiments, especially on challenging benchmark problems, further validated the adaptability and generalizability of the method to a broad range of real-world applications, offering a compelling alternative to purely traditional solvers.

A reduced Root Mean Square Error (RMSE) indicates that the numerical solutions closely match the exact or reference solutions, which significantly enhances the overall accuracy of the method. This level of precision is especially critical when dealing with complex differential equations, such as autonomous and singular types, where small errors can amplify and lead to misleading results. In practical applications, such as engineering simulations or real-time control systems, reduced RMSE ensures more reliable and predictable outcomes. It also lowers the risk of system failure in sensitive domains like aerospace, healthcare, and environmental modeling. Additionally, high accuracy reduces the need for post-processing or correction algorithms, streamlining computational workflows. With lower RMSE, models converge faster, reducing computation time and resource usage which is an important factor for time-sensitive or resource-constrained environments. It further improves model calibration and tuning by minimizing dis-

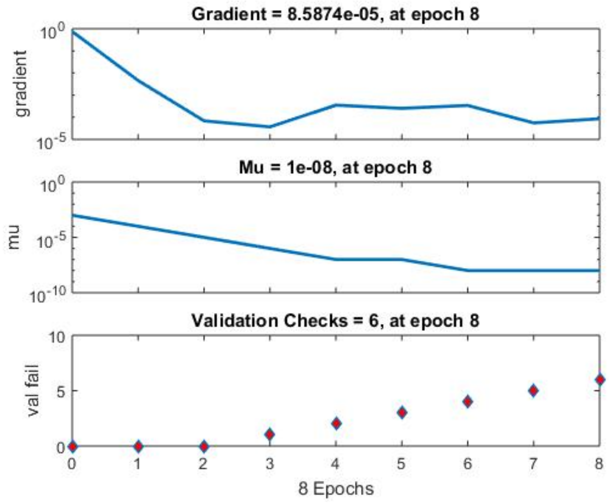

Figure 19: Training State Curve

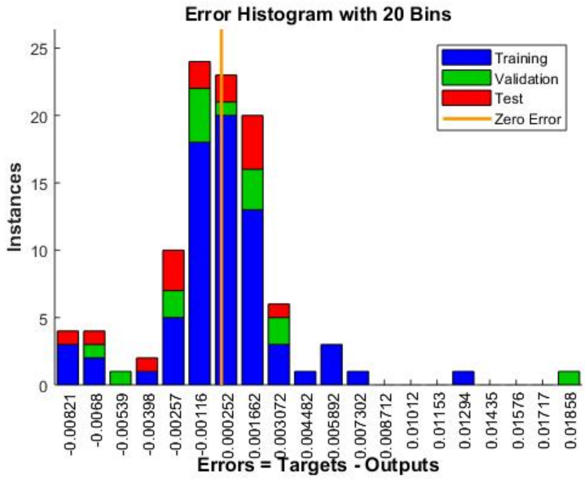

Figure 20: Error Histogram Distribution

crepancies during training or parameter estimation. The method's ability to maintain low RMSE across varied problem domains also highlights its scalability and adaptability to broader real-world problems. This robustness allows the neural refinement approach to outperform traditional methods not just in accuracy but also in efficiency. Ultimately, lower RMSE values translate to greater confidence in numerical simulations used for decision-making and predictive modeling.

7. Conclusion

In this study, we introduced a novel approach for addressing the challenges posed by autonomous and singular differential equations using a combination of Milne-Simpson's methods and neural network-based refinement. These equations, common in modeling complex systems, often exhibit intricate behavior near equilibrium points and singularities, making their accurate numerical solution a daunting task. Traditional numerical methods can falter in such scenarios. To overcome these challenges, we integrated neural networks into the solution process. We began by applying Milne-Simpson's methods to obtain initial solutions, followed by a neural refinement step designed to capture subtle features and improve accuracy. Our approach was validated through extensive experiments on a diverse range of problem domains, demonstrating its effectiveness in enhancing solution accuracy while maintaining computational efficiency. Due to the successful integration of neural networks with Milne-Simpson's methods, avenues

are opened for further research which may include exploration of the optimization of neural network architectures and training strategies.
Future research could focus on extending the neural refinement approach to solve partial differential equations (PDEs), which are prevalent in modeling multi-dimensional and time-dependent phenomena in science and engineering. Another promising direction is adapting the method for stiff systems, where traditional solvers often struggle with stability and efficiency. Incorporating physics-informed neural networks or adaptive time-stepping strategies may further enhance the frameworks applicability. Broadly, this work advances computational mathematics by integrating deep learning with classical numerical methods to improve accuracy and scalability. Its interdisciplinary potential spans applications in physics, engineering, biology, and finance, enabling more reliable simulations and data-driven decision-making.

References

[1] Abdelhalim E. (2011). A new analytical and numerical treatment for singular two-point boundary value problems via the Adomian decomposition method. Journal of Computational and Applied Mathematics, 235, 1914-1924. 1
[2] Aluthge, A. and Sarra, S.A. (2023). A Filtered Milne-Simpson ODE Method with Enhanced Stability Properties. Journal of Applied Mathematics and Physics, 11, 192-208. 1, 5.1
[3] Azis, D. & Napitupulu, M. (2019). Comparison of Milne-Simpson Method and Hamming Method in Logistic Equation Settlement on Pert Prediksi the People of Bandar Lampung City. J. Phys, Conf. Ser. 1338 012038 1
[4] Chen, R. T. Q., Rubanova, Y., Bettencourt, J., & Duvenaud, D. (2018). Neural Ordinary Differential Equations. Advances in Neural Information Processing Systems (NeurIPS). 1
[5] Emmanuel, S., Sathasivam, S. & Ogunniran, M. O. (2024). Multi-derivative hybrid block methods for singular initial value problems with application. Scientific African: 24, e02141, 1-21. 1
[6] Emmanuel, S., Sathasivam, S. & Ogunniran, M. O. (2024). Leveraging Feed-Forward Neural Networks to Enhance the Hybrid Block Derivative Methods for System of Second-Order Ordinary Differential Equations, Journal of Computational and Data Science. 1
[7] Essam R. El-Zahar, A. M. Alotaibi, Abdelhalim Ebaid, Dumitru Baleanu, José Tenreiro Machado & Y. S. Hamed (2020). Absolutely stable difference scheme for a general class of singular perturbation problems. Advances in Difference Equations, 411. 1
[8] Garwood, J. J. & Jator, S. N. (2016). Using rational logarithmic basis functions to solve singular differential equation. Tenth MSU Conference on Differential Equations and Computation Simulations, 23, 1-7. 1, 5.1, 2
[9] Jator, S. N. & Colemann (2017). A non-linear second derivative method with variable step-size based on continued fractions for singular IVPs. Cogent Mathematics, 4: 1335498. 1, 5.1, 2, 3
[10] Junxian K., Mingliang W., Jiajun H., & Yuhong S. (2023). Improved Milne-Hamming Method for Resolving High-Order Uncertain Differential Equations. Applied Mathematics and Computation, 457, 128199, 1
[11] Ogunniran, M. O., Tijani, K. R., Moshood, L. O., Ojo, R. O., Yakusak, N. S., Muritala, F., Kareem, K. O. & Oluwayemi, M. O. (2025). Harnessing neural networks in hybrid block integrator for efficient solution of boundary value problems. Thermal Advances, 2, 100022. 1
[12] Ogunniran, M. O., Olaleye, G. C. Taiwo, O. A., Shokri, A. & Nonlaopon, K. (2023). Generalization of a Class of Uniformly Optimized k-step Hybrid Block Method for Solving Two-point Boundary Value Problem. Results in Physics, 44, 106–147. 1
[13] Ogunniran, M. O., Haruna, Y. & Adeniyi, R. B. (2019). Efficient k-Derivative Methods for Lane-Emden Equations and Related Stiff Problems. Nigerian Journal of Mathematics and Applications, A28, 1–17. 1
[14] Pathak, J., Hunt, B. R., Girvan, M., Lu, Z., & Ott, E. (2018). Model-free prediction of large spatiotemporally chaotic systems from data: A reservoir computing approach. Physical Review Letters, 120(2), 024102. 1
[15] Rahmatan, H., Shokri, A., Ahmad, H., Botmart, T. (2022). Subordination Method for the Estimation of Certain Subclass of Analytic Functions Defined by the-Derivative Operator,Journal of Function Spaces, 2022. 1
[16] Raissi, M., Perdikaris, P., & Karniadakis, G. E. (2019). Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. Journal of Computational Physics, 378, 686-707. 1
[17] Särkkä, S., Solin, A., & Hartikainen, J. (2019). Spatiotemporal learning via infinite-dimensional Bayesian filtering and smoothing: A look at Gaussian process regression through Kalman filtering. IEEE Signal Processing Magazine, 36(4), 30-43. 1
[18] Sunday, J., Shokri, A. and Marian, D. (2022). Variable step hybrid block method for the approximation of Kepler problem. Fractal and Fractional, 6(6), 343. 1