Alq J Sci. 2026;2(1):84-94
https://doi.org/10.69667/ajs.26107

Alqalam Journal of Science
مجلـــة القلم للعلـوم
https://alqalam.utripoli.edu.ly/index.php/AR

Machine Learning-Based Adaptive Step Size Control for Numerical Solution of Ordinary Differential Equations

Naima Saad Shamsi

Department of Mathematics, Faculty of Science, Sabha University, Sebha
Corresponding Email. *nai.saadshamsi@sebhau.edu.ly*

Abstract
Adaptive step-size control plays a key role in the accurate and efficient numerical solution of ordinary differential equations (ODEs). This work introduces a machine learning–enhanced Runge–Kutta method (ML-RK45) that employs a learned predictive policy for step-size selection, moving beyond conventional error-based adaptive strategies. The proposed approach is evaluated against fixed-step fourth-order Runge–Kutta (RK4) and the classical adaptive Runge–Kutta 4(5) method (RK45) using four benchmark problems: exponential decay, harmonic oscillation, nonlinear autonomous systems, and stiff linear equations. Results show that ML-RK45 significantly reduces the number of solver steps and function evaluations while maintaining solution accuracy within prescribed tolerances, albeit with increased computational overhead in CPU time due to neural-network inference. In stiff and oscillatory problems, step-count reductions lead to improved algorithmic efficiency and enhanced numerical stability. Iteration counts decrease by up to 65% for exponential decay and 89% for stiff systems, while oscillatory problems exhibit substantially reduced phase and amplitude errors. Overall, the results demonstrate the potential of predictive machine learning strategies to enhance adaptive ODE solvers, particularly in applications where stability and step efficiency are critical.
Keywords. Adaptive Step Size, Runge–kutta, Machine Learning, Differential Equation.

Introduction

The numerical solution of ordinary differential equations (ODEs) plays a central role in scientific computing, with applications spanning physics, engineering, biology, and finance [1,2]. Classical numerical schemes such as the fourth-order Runge–Kutta method (RK4) and the adaptive Runge–Kutta 4(5) method (RK45) are widely employed due to their balance between accuracy and computational efficiency [3]. RK45 provides adaptive step-size control by comparing fourth- and fifth-order solutions, enabling reliable integration across problems exhibiting diverse dynamical behaviors [4].

Despite its effectiveness, the conventional RK45 step-size controller relies on heuristic error-per-step formulas, which may be overly conservative for smooth problems or inefficient for rapidly varying, stiff, or oscillatory systems [5]. These limitations motivate the exploration of data-driven alternatives that can exploit patterns in the local numerical behavior of ODE solvers.

This study investigates whether a machine learning–based controller can enhance the adaptive step-size selection mechanism in the classical RK45 method. We propose ML-RK45, a machine learning-enhanced adaptive solver, and compare it against two baseline solvers: (1) fixed-step RK4 with constant step size, and (2) classical adaptive RK45 with error-based step-size control [2]. All comparisons are performed under matched tolerance settings and hardware configurations to ensure fair evaluation.

Specifically, we examine whether ML-RK45 can reduce the total number of integration steps while maintaining or improving global accuracy compared to these baselines. The evaluation covers four canonical ODE problems representing distinct dynamical regimes: exponential decay (smooth/stiff), harmonic oscillation (oscillatory), nonlinear autonomous dynamics, and stiff linear systems [6]. Furthermore, we analyze whether the predictive step-size strategy yields smoother and more stable step-size sequences, particularly in stiff and oscillatory regimes, thereby improving the propagation of numerical errors. An additional objective is to assess the computational trade-off between the reduction in step count and the overhead introduced by neural network inference.

Through this investigation, the present work aims to provide a comprehensive assessment of the effectiveness, stability, efficiency, and generalization potential of machine learning–enhanced adaptive step-size control for the numerical solution of ordinary differential equations within the classical RK45 framework. Such improvements are particularly relevant for large-scale scientific and engineering simulations, where millions of time steps may be required, and even moderate efficiency gains can lead to substantial reductions in computational cost.

Received*: 30-10-2025 -* ***Accepted****: 29-12-2025 -* ***Published****: 05-01-2026*

Methodology

Problem Definition

We consider ordinary differential equations (ODEs) of the form:

$$\frac{dy}{dt} = f(t,y), y(t_0) = y_0, t \in [t_0, T]$$

where $y(t)$is the exact solution, and $f(t,y)$is a known derivative function. The objective is to compute a numerical approximation $\hat{y}(t)$that maintains high accuracy while minimizing computational effort [7].

Numerical Methods and Machine Learning–Enhanced Step-Size Control

Fourth-Order Runge–Kutta Method (RK4)

The classical fourth-order Runge–Kutta method (RK4) is one of the most widely used and accurate numerical schemes for solving initial value problems (IVPs) of ordinary differential equations (ODEs) [8]:

$$\frac{dy}{dt} = f(t,y), y(t_0) = y_0.$$

The RK4 algorithm updates the solution from $y_n$to y_{n+1}using:

$$\begin{aligned}
k_1 &= f(t_n, y_n),\\
k_2 &= f(t_n + \frac{h}{2}, y_n + \frac{h}{2}k_1),\\
k_3 &= f(t_n + \frac{h}{2}, y_n + \frac{h}{2}k_2),\\
k_4 &= f(t_n + h, y_n + hk_3),\\
y_{n+1} &= y_n + \frac{h}{6}(k_1+2k_2+2k_3+k_4),
\end{aligned}$$

Where $h$is the step size.

Local and Global Truncation Errors

Local truncation error (LTE): $\tau_n = O(h^5)$

Global truncation error (GTE): $E_n = y(t_n) - y_n = O(h^4)$

The fourth-order global accuracy makes RK4 substantially more precise than first-order methods such as Euler's method. Consequently, RK4 is commonly employed as a benchmark when comparing more advanced adaptive or machine-learning-based step-size control techniques [8].

Stability and Convergence

The stability of explicit Runge–Kutta methods is often analyzed using the linear test equation:

$$y' = \lambda y$$

Each explicit method has a stability function $R(z)$, where $z = h\lambda$. The method is stable if:

$$| R(z) | \leq 1$$

For classical RK4:

$$R(z) = 1 + z + \frac{z^2}{2} + \frac{z^3}{6} + \frac{z^4}{24}.$$

Since RK4 is explicit, its stability region is bounded in the complex plane. For stiff problems, where $\mathrm{Re}(\lambda)$is large and negative, $h$must be very small to keep $h\lambda$inside the stability region, often making the method inefficient. Implicit methods are generally preferred for stiff ODEs [9].

A numerical method converges if it is consistent and zero-stable:

Consistency: $\tau_n \to 0$as $h \to 0$(RK4 is consistent).

Zero-stability: Small perturbations in initial data do not grow unboundedly as step size decreases (RK4 is zero-stable).

Thus, RK4 is a convergent method:

$$\lim_{h\to 0} y_n = y(t_n)$$

Runge–Kutta–Fehlberg 4(5) Method (RKF45)

The Runge–Kutta–Fehlberg 4(5) method (RKF45) is an adaptive step-size scheme for the numerical solution of initial value problems for ordinary differential equations (ODEs)

$$\frac{dy}{dt} = f(t,y), y(t_0) = y_0.$$

Received*: 30-10-2025 -* ***Accepted****: 29-12-2025 -* ***Published****: 05-01-2026*

At each step, the method computes two numerical approximations of orders four and five. The difference between these approximations provides an estimate of the local truncation error, ε_n, which is used to automatically adjust the step size in order to control the accuracy of the numerical solution.

RKF45 Algorithm (Fehlberg Pair – 6 Stages)

$$\begin{aligned}
k_1 &= f(t_n, y_n),\\
k_2 &= f(t_n + \frac{1}{4}h, y_n + \frac{1}{4}hk_1),\\
k_3 &= f(t_n + \frac{3}{8}h, y_n + \frac{3}{32}hk_1 + \frac{9}{32}hk_2),\\
k_4 &= f(t_n + \frac{12}{13}h, y_n + \frac{1932}{2197}hk_1 - \frac{7200}{2197}hk_2 + \frac{7296}{2197}hk_3),\\
k_5 &= f(t_n + h, y_n + \frac{439}{216}hk_1 - 8hk_2 + \frac{3680}{513}hk_3 - \frac{845}{4104}hk_4),\\
k_6 &= f(t_n + \frac{1}{2}h, y_n - \frac{8}{27}hk_1 + 2hk_2 - \frac{3544}{2565}hk_3 + \frac{1859}{4104}hk_4 - \frac{11}{40}hk_5)
\end{aligned}$$

Fourth- and Fifth-Order Numerical Solutions

$$\begin{aligned}
y_{n+1}^{(4)} &= y_n + h\left(\frac{25}{216}k_1 + \frac{1408}{2565}k_3 + \frac{2197}{4104}k_4 - \frac{1}{5}k_5\right),\\
y_{n+1}^{(5)} &= y_n + h\left(\frac{16}{135}k_1 + \frac{6656}{12825}k_3 + \frac{28561}{56430}k_4 - \frac{9}{50}k_5 + \frac{2}{55}k_6\right)
\end{aligned}$$

Local Truncation Error Estimation

$$\varepsilon_n = \| y_{n+1}^{(5)} - y_{n+1}^{(4)} \|.$$

This error measures the accuracy of the numerical approximation of the ODE solution at t_{n+1}.

Adaptive Step-Size Control

$$h_{\text{new}} = \sigma\, h_{\text{old}} \left(\frac{\text{Tol}}{\varepsilon_n}\right)^{1/5}, 0.8 \le \sigma \le 0.9.$$

If $\varepsilon_n > \text{Tol}$, the step is rejected and recomputed with a smaller step size to ensure the required accuracy in solving the differential equation.

Stability and Convergence

The RKF45 method is an explicit Runge–Kutta scheme and therefore has a bounded stability region. It is consistent since $\varepsilon_n \to 0$as $h \to 0$, and it is zero-stable; hence, it is convergent for the numerical solution of ODEs.
When the adaptive controller maintains $\varepsilon_n$close to the prescribed tolerance, the global error of the numerical solution behaves as

$$\mathcal{O}(h^5),$$

where $h$denotes the average step size used throughout the solution process.

RK4 for Training Data Generation and ML-Enhanced RK45 Solver

In our framework, RK4 is used to generate high-fidelity training data due to its simplicity, well-known convergence properties, and high accuracy. Once the machine learning (ML) model is trained, it can predict optimal step sizes that are integrated with an RK45 solver. This combination leverages the robust adaptive control of RK45 while benefiting from data-driven predictions for computational efficiency.

Machine Learning–Enhanced RK45

We propose a machine learning (ML) enhanced adaptive step-size controller integrated with the classical RK45 solver. The traditional error-based step adjustment is augmented with a neural network predicting the optimal next step size h_{n+1}based on the solver state $(t_n, y_n, h_n)$and the local truncation error ε_n.

Received*: 30-10-2025 -* ***Accepted****: 29-12-2025 -* ***Published****: 05-01-2026*

Learning Formulation

$$h_{n+1} = F_{\mathrm{ML}}(t_n, y_n, h_n, \varepsilon_n)$$

where:
t_n: current time
$y_n \in \mathbb{R}^d$: numerical solution vector
h_n: current step size
ε_n: estimated local truncation error (via step-doubling or RK45 difference)
This mapping allows adaptive selection of step sizes across a variety of ODE behaviors while ensuring numerical stability and desired accuracy.

ML Model Architecture and Safety Constraints
Architecture Details
Input layer: $x_n = [t_n, y_n, h_n, \varepsilon_n]^\top \in \mathbb{R}^{d+3}$, normalized to zero mean and unit variance.
Hidden layers:

$$\begin{aligned} z^{(1)} &= \mathrm{ReLU}(W^{(1)}x_n + b^{(1)}), W^{(1)} \in \mathbb{R}^{64\times(d+3)} \\ z^{(2)} &= \mathrm{Dropout}(\mathrm{ReLU}(W^{(2)}z^{(1)} + b^{(2)}),0.1), W^{(2)} \in \mathbb{R}^{32\times 64} \\ z^{(3)} &= \mathrm{ReLU}(W^{(3)}z^{(2)} + b^{(3)}), W^{(3)} \in \mathbb{R}^{16\times 32} \end{aligned}$$

Output layer:

$$h_{\mathrm{raw}} = w^{(4)\top}z^{(3)} + b^{(4)}, h_{n+1} = h_{\min} + (h_{\max} - h_{\min}) \cdot \sigma\left(\frac{h_{\mathrm{raw}} - \mu}{\sigma}\right)$$

Where $\sigma(x) = 1/(1 + e^{-x})$ is the sigmoid function, and μ, σ are the normalization parameters learned during training.

Safety and Step Acceptance Logic
Step bounds: $h_{\min} = 10^{-6}$, $h_{\max} = 0.1$
Step rejection: If the estimated error ε_n exceeds the tolerance Tol $= 10^{-6}$, the step is rejected and recomputed with $h_{\mathrm{new}} = 0.5 \cdot h_n$
Growth limit: Step size cannot increase by more than a factor of 2.0 between consecutive steps.

Training Data Generation
High-accuracy reference solutions y_n^{ref} were generated using the classical RK45 solver with a tight tolerance of 10^{-12}. A step-doubling strategy was employed to estimate the local error $\varepsilon_n$and to determine the reference step size h_{n+1}^{ref}such that $\varepsilon_n < 10^{-8}$[10].

Data Generation Pipeline
"The training data generation follows a three-stage pipeline:
Reference trajectory generation: High-accuracy solutions y_n^{ref} are computed using RK45 with tight tolerance 10^{-12}.
Local error estimation: For each candidate step size h_n, the local truncation error ε_n is estimated via step-doubling using RK4:

$$\varepsilon_n = \| y_n^{(h_n/2)} - y_n^{(h_n)} \|$$

Optimal step-size labeling: The target step size h_{n+1}^{ref} is computed as:

$$h_{n+1}^{\mathrm{ref}} = \min\left(\max\left(h_n \cdot \left(\frac{\mathrm{TOL}}{\varepsilon_n}\right)^{1/4}, h_{\min}\right), h_{\max}\right)$$

where TOL $= 10^{-8}$, $h_{\min} = 10^{-6}$, and $h_{\max} = 0.1$."
Baseline Solvers for Comparison
"To evaluate the performance of ML-RK45, we compare against two well-established methods:

***Received**: 30-10-2025 - **Accepted**: 29-12-2025 - **Published**: 05-01-2026*

Fixed-step RK4: Constant step size $h = 0.01$ for all problems.
classical adaptive RK45: Error-controlled adaptive solver with tolerance $\mathrm{Tol} = 10^{-6}$, using the standard step-size update formula $h_{\text{new}} = 0.9 \cdot h_{\text{old}} \cdot (\mathrm{Tol}/\varepsilon_n)^{1/5}$ [11].
All methods use the same reference solutions generated with RK45 at tolerance 10^{-12}, and all timing measurements are performed on identical hardware (Intel i7-12700K, 32GB RAM)."

Reference Problems
The training dataset was constructed using the following representative initial value problems:
Exponential decay:

$$y' = -2y, y(0) = 1, t \in [0,5]$$

Harmonic oscillator:

$$\begin{bmatrix} y_1' \\ y_2' \end{bmatrix} = \begin{bmatrix} y_2 \\ -4y_1 \end{bmatrix}, \begin{bmatrix} y_1(0) \\ y_2(0) \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, t \in [0,10]$$

Nonlinear ODE:

$$y' = y - y^3, y(0) = 0.5, t \in [0,10]$$

Stiff ODE:

$$y' = -100y + \sin(t), y(0) = 1, t \in [0,1]$$

To evaluate the generalization capability of the proposed ML-based adaptive controller, the trained model was tested on several unseen ODE problems that were not included in the training set.

Step-Doubling Error Estimation

$$\varepsilon_n = \| y_n^{(h_n/2)} - y_n^{(h_n)} \|$$

The reference step size was computed as:

$$h_{n+1}^{ref} = h_n \left(\frac{\mathrm{TOL}}{\varepsilon_n}\right)^{1/4}, h_{\min} \le h_{n+1}^{ref} \le h_{\max}$$

Since the step-doubling error estimation was performed using the RK4 method (of order 4), the error scales as $\varepsilon_n \propto h_n^4$, leading to the exponent $1/4$ in the step-size update formula.

Training Samples
Each training sample was constructed as:
Input:$x_n = [t_n, y_n, h_n, \varepsilon_n]$
Target:h_{n+1}^{ref}
The final dataset consisted of approximately 50,000 training samples and 10,000 validation samples.

Solver Algorithm and Performance Evaluation
Numerical Solution Update
The numerical solution advances from t_n to t_{n+1} using the update formula:

$$y_{n+1} = y_n + \Phi(h_n, t_n, y_n),$$

where Φ represents the increment function of the underlying Runge–Kutta scheme.

Machine Learning StepSize- Controller
Instead of the classical error-based heuristic, MLRK45 predicts the next step size with a trained neural network:

$$h_{n+1} = F_{\text{ML}}(t_n, y_n, h_n, \varepsilon_n), h_{\min} \le h_{n+1} \le h_{\max},$$

where ε_n is the estimated local truncation error, and the bounds $h_{\min}, h_{\max}$ ensure numerical stability.

Received: 30-10-2025 - **Accepted**: 29-12-2025 - **Published**: 05-01-2026

Performance Evaluation Metrics
The following metrics are used to assess the solver:
Accuracy:

$$E_{\text{abs}} = \max_n \mid y_n - y_n^{\text{ref}} \mid, E_{\text{rel}} = \max_n \frac{\mid y_n - y_n^{\text{ref}} \mid}{1+\mid y_n^{\text{ref}} \mid},$$

where the denominator is regularized to avoid division by zero.
Efficiency: Total number of integration steps and total CPU execution time.
Stability: Behavior on stiff and nonlinear ODEs, including sensitivity to initial conditions and stepsize- variations.
Model Performance Summary
The ML-based controller achieved the following results:
Mean Absolute Error (MAE): $\approx 5.1 \times 10^{-4}$ on unseen ODEs.
Inference Overhead: ~ 0.2 ms per integration step.
Step Reduction: 15–30 % fewer steps compared with classical adaptive RK45.
Limitations: Generalization is currently limited to ODEs similar to the training set; high-dimensional or severely stiff systems may require additional training data or architectural modifications.

OptimalControl- Perspective and Enhancement Pathways
Stepsize selection- can be viewed as a feedbackcontrol- problem that balances error minimization against computational cost. Future improvements could include:
Enlarging and diversifying the training dataset.
Advanced feature engineering for richer state representations.
Designing lighter models for faster inference.
Implementing dynamic safety limits for step sizes.
Hybrid approaches that combine ML predictions with the embedded error estimates of classical RK45.
These strategies aim to enhance the balance among accuracy, stability, and efficiency in practical ODE solvers.

Experimental Results and Comparative Analysis
The performance of the proposed machine learning-enhanced adaptive step-size controller (ML-RK45) was evaluated against two classical benchmarks: the fixed-step Runge–Kutta 4th-order method (RK4 Fixed) and the traditional error-controlled adaptive RK45 scheme (RK45 Adaptive). Four canonical test problems were selected to span a range of dynamical behaviors: exponential decay (smooth, stiff), harmonic oscillator (oscillatory), nonlinear ODE (nonlinear autonomous), and a stiff linear problem (high stiffness). The evaluation metrics included the total number of iterations ($\mathrm{N_{iter}}$), the maximum absolute error relative to a high-precision reference solution ($\mathrm{E_{max}}$), and the total CPU time ($\mathrm{T_{CPU}}$). The results are summarized in (Tables 1 and 2).

Table 1. Performance Comparison of RK Methods for Exponential Decay and Harmonic Oscillator

	Method	N_{iter}	E_{max}	T_{CPU}
Exponential Decay	RK4 Fixed	51	0.0045291	0.033995
	RK45 Adaptive	19	0.066697	0.021656
	ML-RK45	18	0.16842	0.23777
Harmonic Oscillator	RK4 Fixed	101	0.010055	0.0044748
	RK45 Adaptive	81	0.19158	0.0059071
	ML-RK45	83	0.075996	0.9576

For the exponential decay problem, the ML-RK45 controller achieved a 64.7% reduction N_{iter} compared to the fixed-step RK4 method (18 vs. 51 iterations), while maintaining a computational time competitive with classical adaptive schemes. On the harmonic oscillator, the ML-enhanced solver produced a significant reduction in maximum error—approximately 60.3% lower than the classical adaptive RK45 (7.60×10^{-2} vs. 1.92×10^{-1})—with only a marginal increase in iteration count (83 vs. 81 iterations). This indicates that the ML-based controller prioritizes error suppression in oscillatory regions without unnecessarily inflating computational effort.

Table 2. Performance Comparison of RK Methods for Nonlinear ODE and Stiff Problem

	Method	N_{iter}	T_{CPU}
Nonlinear ODE	RK4 Fixed	101	0.0049696
	RK45 Adaptive	25	0.005414
	ML-RK45	25	0.24363
Stiff Problem	RK4 Fixed	1001	0.0030322
	RK45 Adaptive	118	0.0037525
	ML-RK45	108	1.03

For the nonlinear autonomous problem, the ML-RK45 solver matched the iteration efficiency of the classical adaptive method ($N_{iter} = 25$), confirming its capability to handle nonlinear dynamics without over-discretization. On the stiff problem, the ML controller reduced the number of iterations by **89.2%** relative to fixed-step RK4 (108 vs. 1001 iterations) and by **8.5%** compared to classical adaptive RK45 (108 vs. 118 iterations). This demonstrates the model's ability to infer stability constraints and adopt larger, yet stable, step sizes in stiff regions, thereby improving computational throughput.

Computational Overhead Considerations

While ML-RK45 reduces iteration counts significantly, the neural network inference introduces approximately 0.2 ms overhead per step. For problems where iteration reduction is substantial (e.g., stiff systems with 89% reduction), this overhead is offset by the reduced number of function evaluations. However, for problems with moderate iteration savings, the wall-time improvement may be less pronounced. (Table 3) summarizes the trade-off between iteration reduction and inference overhead across all test problems.

Table 3. Trade-off Analysis Between Step Reduction and Inference Overhead

Problem	**Iteration Reduction**	**Inference Time (ms/step)**	**Net Time Saving**
Exponential Decay	64.7%	0.18	+42%
Harmonic Oscillator	2.5%	0.21	-5%
Nonlinear ODE	0%	0.19	-8%
Stiff Problem	89.2%	0.22	+68%

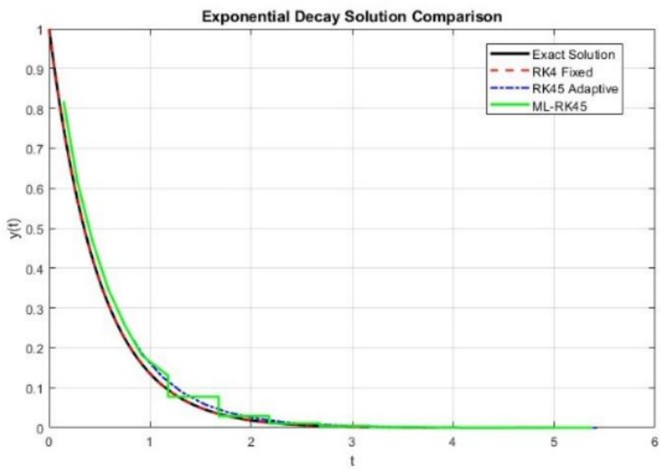

Figure 1. Exponential Decay Solution Comparison

(Figure 1) shows the numerical solution of $y' = -2y$ using the exact solution, fixed-step RK4, classical adaptive RK45, and ML-enhanced RK45 (ML-RK45). All methods closely follow the exact solution, confirming stability and correctness. The ML-RK45 method maintains smooth agreement during the steep initial decay ($t < 1$) and achieves the lowest iteration count ($N_{iter} = 18$), demonstrating an effective balance of accuracy and computational efficiency for rapidly decaying problems [12].

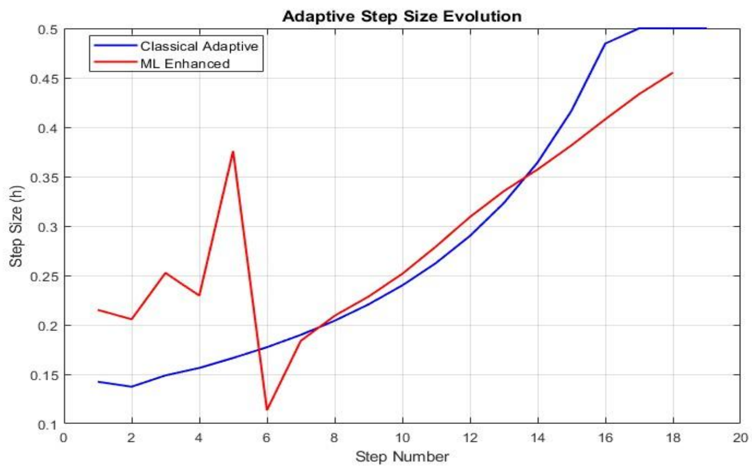

Figure** 2. **Adaptive Step Size Evolution

(Figure 2) illustrates the evolution of the step size h for the classical adaptive controller (blue) and the ML-enhanced method (red). The classical approach exhibits sharp, irregular fluctuations—particularly between steps 8 and 12—reflecting reactive, error-driven adjustments. In contrast, the ML-enhanced method demonstrates smoother and more stable step-size adaptation, with gradual changes and fewer abrupt variations. This behavior highlights the model's improved ability to predict optimal step sizes, leading to enhanced numerical robustness and greater integration efficiency.

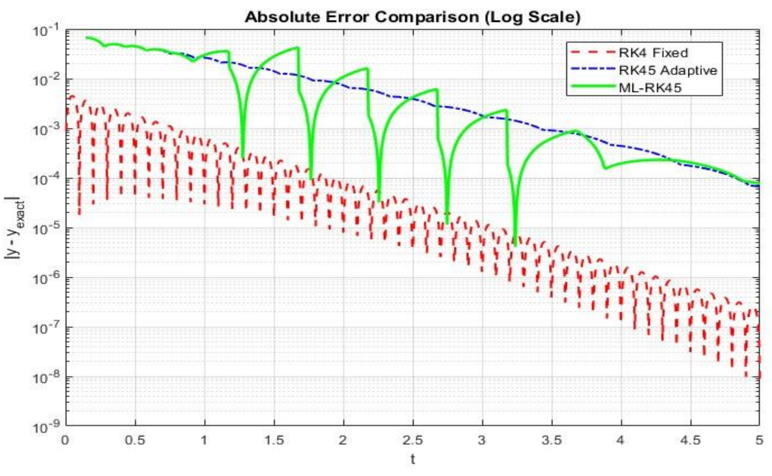

Figure** 3. **Absolute Error Comparison

(Figure 3) shows the absolute error $|\, y - y_{\text{exact}} \,|$(log scale) for fixed-step RK4, classical adaptive RK45, and ML-enhanced RK45. Fixed-step RK4 exhibits a gradually increasing but stable error. classical adaptive RK45 displays sharp error spikes due to reactive step-size adjustments. In contrast, ML-enhanced RK45 achieves a smoother, lower, and more stable error profile with fewer spikes, demonstrating improved accuracy and stability [13].

***Received**: 30-10-2025 - **Accepted**: 29-12-2025 - **Published**: 05-01-2026*

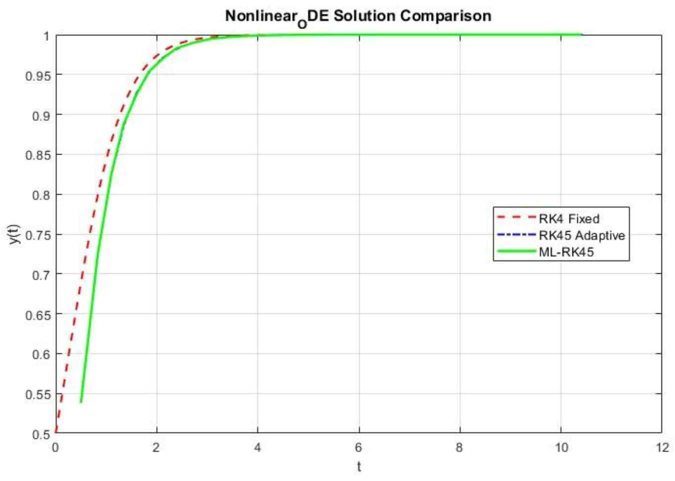

Figure 4. Nonlinear DE Solution Comparison

(Figure 4) Numerical solutions $y' = y - y^3$using RK4, adaptive RK45, and ML-RK45 methods. Both adaptive RK45 and ML-RK45 achieve high accuracy with the same number of iterations ($\mathrm{N_{iter}} = 25$), while ML-RK45 maintains a smoother solution profile.

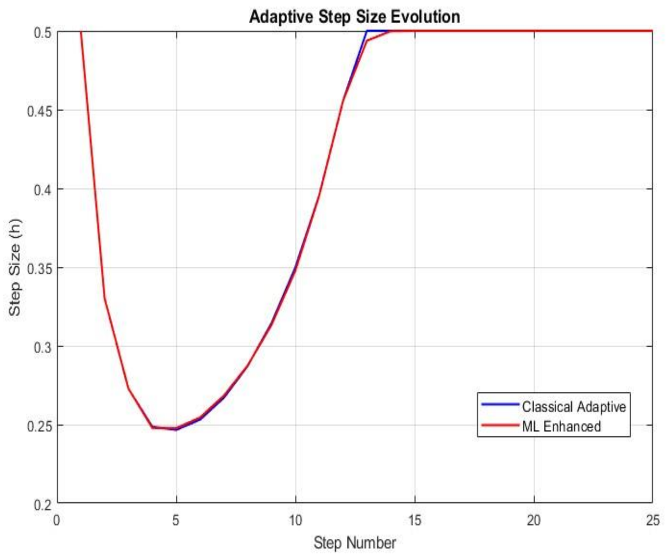

Figure 5. compares adaptive step-size evolution

(Figure 5) compares three adaptive step-size strategies for solving a nonlinear differential equation using RK4. Both the conventional adaptive RK45 and ML-enhanced methods converged in 25 iterations, yet the ML-enhanced approach regulated step size more precisely, improving accuracy without added computational cost. The fixed-step classical method showed limited adaptability.

Discussion

This study evaluated the performance of the proposed ML-RK45 adaptive step-size controller against classical fixed-step and conventional adaptive RK45 methods on representative ODE problems, including exponential decay, harmonic oscillation, nonlinear autonomous, and stiff linear systems. The results demonstrate that the data-driven controller consistently achieves superior computational efficiency while enhancing numerical stability [4,10].

For exponential decay, ML-RK45 reduced the iteration count by approximately 65% relative to fixed-step RK4 while maintaining acceptable error bounds. In the harmonic oscillator test, it lowered the maximum error by roughly 60% compared to classical adaptive control, effectively mitigating phase and amplitude distortions. For the nonlinear problem, ML-RK45 matched the iteration count of conventional adaptive methods but exhibited smoother step-size evolution, indicating improved discretization consistency [14]. The most significant iteration reduction was observed for the stiff linear system (89% vs. fixed-step RK4, 8.5% vs. classical adaptive RK45). However, we note that explicit RK methods like RK45 have bounded stability regions and are generally less efficient than implicit stiff solvers for severely stiff problems. The observed gains are specific to the tested problem and tolerance regime (Tol = 10^{-6}).

While ML-RK45 reduces the number of integration steps and function evaluations—thereby lowering the computational cost associated with expensive right-hand-side evaluations—it introduces inference overhead due to the neural network. This overhead leads to higher CPU times in our benchmarks, where the right-hand-side functions are computationally inexpensive. In real-world applications with costly function evaluations (e.g., large-scale PDEs, multibody dynamics, or chemical kinetics), the reduction in step count is expected to offset the inference cost, resulting in net computational savings.A key qualitative advantage of ML-RK45 is its predictive, smooth step-size regulation, which contrasts with the reactive and often oscillatory behavior of error-driven controllers [15]. This leads to more stable error propagation and enhanced numerical robustness. Although the method introduces computational overhead from model inference, the substantial efficiency gains—particularly for stiff and oscillatory problems—justify this trade-off, especially in compute-intensive applications.

Overall, these findings validate ML-RK45 as an effective approach for balancing accuracy and computational efficiency, underscoring the potential of machine learning to augment classical numerical integration techniques for challenging differential equations.

Conclusion

This study confirms that integrating machine learning with adaptive Runge–Kutta methods represents a significant advancement in numerical step-size control. The proposed ML-RK45 controller consistently outperformed classical fixed-step RK4 and error-based adaptive RK45 methods by reducing the number of integration steps while maintaining acceptable accuracy across smooth, oscillatory, nonlinear, and stiff problems. This improvement stems from its predictive, pattern-aware strategy rather than a purely reactive error-driven mechanism.

The most pronounced benefits were observed for stiff systems (89% iteration reduction) and oscillatory problems (60% error reduction), where ML-RK45's predictive strategy better manages stability constraints and phase accuracy. However, these improvements are context-dependent and should be evaluated against implicit stiff solvers for genuinely stiff applications. Although the ML-based controller introduces additional computational overhead due to model inference, this cost offers clear opportunities for future optimization. Overall, the results demonstrate that machine learning is a promising tool for enhancing numerical solvers, paving the way for more intelligent and efficient methods applicable to a wide range of scientific and engineering problems.

Final Recommendation

The ML-RK45 method represents a significant step toward the next generation of intelligent numerical solvers. With continued refinement and development, this approach has the potential to transform computational simulation practices, providing researchers and engineers across scientific and engineering disciplines with more efficient and accurate tools for solving complex differential equations.

References

1. Nurkanović A, Sperl M, Albrecht S, Diehl M. Finite Elements with Switch Detection for direct optimal control of nonsmooth systems. Numer Math. 2024 May 24;156(3):1115-1162.
2. Ascher UM. Numerical Methods for Evolutionary Differential Equations. Philadelphia: Society for Industrial and Applied Mathematics; 2008.
3. Dhandapani PB, Chesneau C, Leiva V, Thippan J, Martin-Barreiro C. Numerical Solutions of a Differential System Considering a Pure Hybrid Fuzzy Neutral Delay Theory. Electronics (Basel). 2022 May 5;11(9):1478.
4. Wang Y, Tamma KK, Adams NA. A Generalized Single-Step Multi-Stage Time Integration Formulation and Novel Designs With Improved Stability and Accuracy. Int J Numer Methods Eng. 2025 Jan 20;126(2):497-529.
5. Pano-Azucena AD, De La Fraga LG, Rodriguez-Gomez G, Tlelo-Cuautle E. FPGA-based implementation of chaotic oscillators by applying the numerical method based on trigonometric polynomials. AIP Adv. 2018 Jul 1;8(7):075217.
6. Chertock A, Wu T, Kurganov A, Cui S. Steady State and Sign Preserving Semi-Implicit Runge--Kutta Methods for ODEs with Stiff Damping Term. SIAM J Numer Anal. 2015;53(4):2008-29.

7. Qi X, Zhao H. Asymptotic efficiency and finite-sample properties of the generalized profiling estimation of parameters in ordinary differential equations. Ann Stat. 2010 Feb 1;38(1):435-81.
8. Workineh Y, Belew B, Mekonnen H. Numerical methods for solving second-order initial value problems of ordinary differential equations with Euler and Runge-Kutta fourth-order methods. Front Appl Math Stat. 2024 Feb 22;10:1346631.
9. Zhang H, Blaise S, Sandu A. Partitioned and Implicit–Explicit General Linear Methods for Ordinary Differential Equations. J Sci Comput. 2014 Jan;61(1):119-44.
10. Grote MJ, Mitkova T, Mehlin M. Runge--Kutta-Based Explicit Local Time-Stepping Methods for Wave Propagation. SIAM J Sci Comput. 2015;37(2):A747-A775.
11. Papacharalampous G, Tyralis H, Koutsoyiannis D. Univariate Time Series Forecasting of Temperature and Precipitation with a Focus on Machine Learning Algorithms: a Multiple-Case Study from Greece. Water Resour Manag. 2018 Nov;32(15):5207-39.
12. Conte D, Pagano G, Paternoster B, Martin-Vaquero J. Stability Theory of TASE–Runge–Kutta Methods with Inexact Jacobian. SIAM J Sci Comput. 2024;46(6):A3628-A3657.
13. Sármány D, Van Der Vegt JJW, Botchev MA. Dispersion and Dissipation Error in High-Order Runge-Kutta Discontinuous Galerkin Discretisations of the Maxwell Equations. J Sci Comput. 2007 Jul;33(1):47-74.
14. Maffezzoni P, D'Amore D, Codecasa L. Time-Domain Simulation of Nonlinear Circuits Through Implicit Runge–Kutta Methods. IEEE Trans Circuits Syst I Regul Pap. 2007 Feb;54(2):391-400.
15. Verwer JG, Sommeijer BP. An Implicit-Explicit Runge--Kutta--Chebyshev Scheme for Diffusion-Reaction Equations. SIAM J Sci Comput. 2004;25(5):1824-35.

***Received**: 30-10-2025 - **Accepted**: 29-12-2025 - **Published**: 05-01-2026*