IFS/ACM Transactions on Machine Learning, vol. 3 , no. 1, pp. 1-8, 2026

Application of Adaptive Step Size Runge-Kutta Method in Solving Ordinary Differential Equations

Keke Hu[a,*], Asif Ali Laghari[b], Yang Liu[a], Junlin Li[a] and Haifeng Li[a]

[a]School of Mathematics and Statistics, Fuyang Normal University, Fuyang 236037 Anhui, China
[b]Department of Computer Science, Sindh Madressatul Islam University, Karachi 74000 Sindh, Pakistan

ARTICLE INFO

Keywords:
Runge-Kutta method
numerical analysis
adaptive step size
initial value problems

ABSTRACT

To address the imbalance between accuracy and efficiency in traditional fixed-step Runge-Kutta methods when dealing with nonlinear ordinary differential equations, this paper proposes an adaptive-step Runge-Kutta method. This method is based on the framework of the fourth-order Runge-Kutta method and dynamically adjusts the step size through local error estimation. It combines an expansion coefficient optimization strategy, reducing the step size when the error exceeds the limit and expanding it when the error is redundant, achieving sensitive feedback of the step size within the error limit. Compared with the fixed-step method, the adaptive-step method can more effectively avoid error accumulation and maintain extremely low error throughout the process of solving initial value problems of ordinary differential equations, effectively balancing computational accuracy and efficiency. Through numerical experiments, the effectiveness and practicality of this method have been verified. The research in this paper provides a balanced solution for initial value problems of ordinary differential equations that combines accuracy retention and computational economy.

1. Introduction

In the current era of artificial intelligence, differential equations, as key mathematical tools for describing natural dynamic processes and engineering phenomena, still play a significant role in characterizing the evolution of physical systems, predicting economic trends, and solving engineering problems [1]. For instance, Cai et al. [2] used the physics-informed neural network method to solve the forward and inverse problems of the thermal-hygroscopic coupling model of single-layer fabrics, demonstrating the complex application of differential equations in materials science. Wang et al. [3] utilized differential equation models to study the impact of grazing strategies on soil and vegetation and conducted soil moisture prediction, highlighting the practical value of differential equations in the field of ecological environment. In addition, the research on the prediction of infectious disease transmission risk based on the SEIR model [4] also fully reflects the core role of differential equations in the modeling of public health events. However, the differential equations encountered in practical applications often have highly nonlinear or complexly coupled characteristics, making it difficult to obtain exact closed-form solutions through traditional analytical methods. Therefore, the development of efficient and high-precision numerical solution methods is of great theoretical and practical significance for promoting scientific research and engineering practice.

Among various numerical solution methods, the Runge-Kutta method, with its high accuracy and wide applicability, is highly recognized and applied in multiple disciplines such as mathematical modeling, numerical analysis, and computer science. This method improves the accuracy of numerical solutions by approximating the solution of differential equations through weighted averages. However, the traditional Runge-Kutta method usually adopts a fixed step size for iterative calculations. This fixed step size strategy has inherent limitations when dealing with differential equations with rapidly changing or strongly nonlinear solution functions. On the one hand, to ensure the stability and accuracy of the numerical solution throughout the solution interval, the fixed step size often needs to be set based on the region where the solution function changes most drastically, which leads to a large amount of redundant calculations in regions where the solution function changes slowly, reducing the overall solution efficiency. On the other hand, if a larger step size is chosen to pursue computational efficiency, it may introduce unacceptable local truncation errors in regions where the solution function changes rapidly, even causing the numerical solution to diverge and resulting in insufficient accuracy or computational instability. This contradiction between accuracy and efficiency severely restricts the application of fixed step size methods in complex scenarios.

To overcome the above-mentioned shortcomings of the fixed step size method, the adaptive step size control strategy emerged and gradually became a research hotspot in the field of numerical solutions. The core idea of the adaptive method lies in dynamically adjusting the calculation strategy based on problem characteristics, an idea that has been widely explored in multiple fields. In the field of numerical solutions, Jain [5] conducted early optimization research on the Runge-Kutta-Fehlberg method, laying an important theoretical foundation for adaptive step size control. Amanatidis et al. [6] proposed a fast adaptive stochastic greedy algorithm

DOI: https://doi.org/10.70891/TML.2026.040001
ISSN of IFS/ACM TML: 3078-5510

*Corresponding author
cokecule@163.com (K. Hu)

for non-monotone submodular maximization under knapsack constraints, achieving a constant approximation ratio while maintaining high efficiency. Chen et al. [7] developed scalable distributed algorithms under the MapReduce and adaptive complexity models, providing efficient parallel solutions for large-scale submodular maximization problems. In addition, in the task of rumor detection, Wang et al. [8] introduced adaptive data augmentation and adversarial training, effectively enhancing the robustness of the model in complex environments. Particularly, Nwankwo and Dai [9] combined the adaptive explicit fourth-order Runge-Kutta-Fehlberg method with a compact finite difference scheme and successfully applied it to the pricing of American put options, demonstrating the practical value of adaptive methods in financial computing. These studies indicate that adaptive strategies can achieve a better balance between efficiency and performance. In the field of numerical solution, the adaptive step size method assesses the local truncation error at the current step size in real time and dynamically adjusts the subsequent calculation step size based on the comparison result of the error and the preset tolerance. This method can "intelligently" allocate computing resources: automatically increasing the step size in regions where the solution function changes gently to improve computational efficiency, and automatically reducing the step size in regions where the solution function changes sharply to ensure computational accuracy.

Aiming at the accuracy optimization problem of the traditional Runge-Kutta algorithm, this study proposes an adaptive step size control method based on error estimation. This method dynamically adjusts the calculation step size by real-time assessment of the local characteristics of the numerical solution and the error estimation value, maintaining efficient computation while significantly improving the accuracy of the numerical solution and the stability of the algorithm. Particularly when dealing with differential equations with sharp changes or strong nonlinear characteristics, this method demonstrates advantages that traditional fixed-step algorithms cannot match. This study is dedicated to providing a more efficient and reliable computational scheme for the numerical solution of differential equations, thereby promoting the wide application and development of this method in practical scenarios such as engineering practice.

2. Runge-Kutta Method

2.1. Basic Principles

The Runge-Kutta method, as a class of efficient numerical algorithms for solving initial value problems of ordinary differential equations, is based on the Taylor series expansion and multi-stage prediction-correction strategy. It approximates the exact solution of the differential equation through a series of carefully designed weighted averages. The core idea of this method is to construct a numerical solution with high accuracy by combining multiple prediction steps and approximations of function values and their derivatives of various orders [10, 11]. From the perspective of algorithm construction theory, the derivation of the coefficients of the Runge-Kutta method can be systematically completed through Butcher's algebraic theory or Hairer and Wanner's B-series theory [12, 13], which provides a solid mathematical foundation for understanding the accuracy order and stability of the method.

An important feature of the Runge-Kutta method is its flexibility and scalability. By adjusting the number of stages and the distribution of weights, Runge-Kutta methods of different orders can be constructed to meet different accuracy requirements and computational resource constraints [14, 15]. Low-order methods are suitable for rapid calculations with low accuracy requirements, while high-order methods can provide higher numerical accuracy while maintaining computational efficiency. It is worth noting that the basic idea of the Runge-Kutta method has given rise to new metaheuristic optimization methods such as the Runge-Kutta optimization algorithm (RUN), which demonstrate excellent exploration and exploitation balance capabilities in complex nonlinear optimization problems [16].

It is important to note that although the Runge-Kutta method has significant advantages in high-precision numerical solutions, its computational cost is relatively high, especially when the dimension of the differential equation increases or the right-hand function is complex. Therefore, in practical applications, the order and step size of the Runge-Kutta method should be reasonably selected based on the characteristics of the specific problem and the limitations of computational resources to balance the accuracy of the numerical solution and computational efficiency [17].

2.2. Classical Fourth-Order Runge-Kutta Method

Differential equations, as fundamental mathematical models for describing many phenomena in nature and engineering, have always been a key research focus in the field of numerical analysis for their numerical solutions. Among various numerical methods, the Runge-Kutta method is highly favored for its high accuracy and stability, especially the fourth-order Runge-Kutta method, which is preferred for solving initial value problems of ordinary differential equations due to its relatively simple calculation process and high numerical accuracy [18, 19, 20]. Regarding the derivation of the fourth-order Runge-Kutta method, existing studies have simplified the calculation of unknown quantities and conducted step-by-step derivations, reducing the complexity of theoretical analysis and making the method easier to understand and apply.

Specifically, for the initial value problem of differential equations of the form:

$$\begin{cases} \frac{dy}{dx} = f(x, y), \quad a < x < b, \\ y(a) = y_0. \end{cases} \tag{1}$$

The iterative formula of the classical fourth-order Runge-Kutta method is as follows:

$$\begin{cases} y_{n+1} = y_n + \frac{h}{6}\left(k_1 + 2k_2 + 2k_3 + k_4\right), \\ k_1 = f(x_n, y_n), \\ k_2 = f\left(x_n + \frac{h}{2}, \quad y_n + \frac{h}{2}k_1\right), \\ k_3 = f\left(x_n + \frac{h}{2}, \quad y_n + \frac{h}{2}k_2\right), \\ k_4 = f\left(x_n + h, \quad y_n + hk_3\right). \end{cases} \tag{2}$$

where h is the step size, k_1, k_2, k_3, k_4 are the four intermediate predicted values, y_n is the solution at the $n-th$ step, and y_{n+1} is the solution at the $(n+1)-th$ step.

The classical fourth-order Runge-Kutta method has a significant advantage in terms of accuracy, but its computational cost should not be overlooked. Especially when dealing with functions that contain complex nonlinear terms, multiple function evaluation operations in a single step iteration may lead to an exponential increase in the consumption of computing resources [21]. This characteristic requires researchers to optimize the configuration of step size and iteration times in practical applications, taking into account the mathematical characteristics of the specific problem and the available computing resources.

3. Adaptive Step Size Improvement Scheme

In the process of numerical solution of differential equations, the optimal configuration of step size parameters is directly related to the efficiency and accuracy of numerical calculation. The traditional fixed step size Runge-Kutta algorithm adopts a constant step size throughout the entire calculation process. This fixed strategy is difficult to adapt to the variation characteristics of the solution function in different regions. In fact, the solution function of differential equations often shows significant local variation differences: in regions where the solution function changes gently, a larger step size can not only maintain an acceptable accuracy level but also greatly improve computational efficiency; while in regions where the solution function changes drastically, a smaller step size must be adopted to ensure the accuracy of the numerical solution. In view of this, this study introduces a dynamic step size adjustment mechanism based on the Runge-Kutta method. By real-time monitoring the local variation characteristics of the solution function, the adaptive adjustment of the step size is realized, thereby achieving the optimal balance between efficiency and accuracy in solving complex differential equations. The local error is evaluated by comparing the differences between the full step size and half step size calculation results, and the step size expansion coefficient is adjusted according to the error feedback; through dynamic step size adjustment, under the premise of meeting the preset error limit, the balance between computational efficiency and accuracy is achieved, and the computational efficiency is improved while maintaining the computational accuracy.

3.1. Error Estimation Method

The adaptive step size Runge-Kutta method proposed in this paper is based on the framework of the fourth-order Runge-Kutta method. For the fourth-order Runge-Kutta method, its stability region is wide. In most differential equation problems, the cumulative effect of local errors can be ignored, so that the local error estimation can effectively reflect the global error trend. Therefore, in iterative calculation, the numerical stability is evaluated by analyzing the local truncation error. The local truncation error of the fourth-order Runge-Kutta method is $O(h^5)$, when $y_{k+1}^{(h)}$ is obtained by calculation with step size h, its error term is C_1h^5, where C_1 is the error term coefficient. While $y_{k+1}^{(h/2)}$ obtained by two calculations with step size $h/2$, the total error term is the accumulation of two local errors, which is approximately:

$$2C_1\left(\frac{h}{2}\right)^5 = \frac{C_1h^5}{16}. \tag{3}$$

Assuming the exact solution is $y(x_{k+1})$, the errors of the two calculations are approximately:

$$y(x_{k+1}) - y_{k+1}^{(h/2)} \approx \frac{1}{16}C_1h^5, \tag{4}$$

Its order is still $O(h^5)$, which is strictly consistent with the theoretical error order of the fourth-order method, ensuring the mathematical compatibility of error estimation and providing a quantitative basis for adaptive adjustment.

3.2. Local Error Calculation Method

The error estimation method in this paper refers to the local error, which is realized by comparing the differences between the calculation results using the current step size and the halved step size.

The specific steps are as follows: First, perform a fourth-order Runge-Kutta method calculation with step size h to obtain the result y_1. Perform two fourth-order Runge-Kutta method calculations with the halved step size $h/2$, first calculate the intermediate value y_{half}, and then use this intermediate value to perform the second calculation to obtain y_2.

$$e_{local} = |y_1 - y_2| \tag{5}$$

The local error reflects the difference between the numerical solution under the current step size and the numerical solution under a finer step size, and is an important indicator to measure whether the step size needs to be changed in the adaptive step size process of this paper.

Dynamically adjusting the step size through error estimation can significantly reduce redundant calculation steps under the premise of ensuring the solution accuracy. Especially when dealing with differential equations with drastic change characteristics, the adaptive step size method accurately matches the change rate of the local solution, and the final overall computational efficiency is usually better than that of the fixed step size algorithm. This apparent increase

Algorithm 1 Adaptive Runge-Kutta Procedure

1: Initialize parameters: Adaptive_RK($x_k, y_k, x_{\text{end}}, h_0, \epsilon, \beta_{up}$).
2: Set initial values: $x[k] = x_k$, $y[k] = y_k$, $h = h_0$.
3: If $x_k + h > x_{\text{end}}$, go to Step 4; else go to Step 5.
4: Adjust $h = x_{\text{end}} - x_k$, compute y_{k+1}, then stop.
5: Compute local error e_{local}. If $e_{\text{local}} > \epsilon$, go to Step 6; if $e_{\text{local}} < \epsilon$, go to Step 9; if equal, go to Step 12.
6: Shrink step: $h_{\text{new}} = 0.5h$, update $e_{\text{local}}^{\text{new}}$, go to Step 7.
7: If $e_{\text{local}}^{\text{new}} > \epsilon$, return to Step 6; else go to Step 8.
8: Set $h = h_{\text{new}}$, go to Step 12.
9: Expand step: $h_{\text{new}} = \beta_{\text{up}} h$, update $e_{\text{local}}^{\text{new}}$, go to Step 10.
10: If $e_{\text{local}}^{\text{new}} > \epsilon$, go to Step 11; else return to Step 9.
11: Set $h = 0.5h_{\text{new}}$, go to Step 12.
12: Compute y_{k+1} with h, update $x_{k+1} = x_k + h$, return to Step 3.

in single-step cost and substantial optimization of global efficiency reflect the advancement of the adaptive step size strategy in the allocation of computational resources.

3.3. Adaptive Step Size Adjustment Strategy

There exists a nonlinear relationship between the local error and the step size. As indicated by Equation 4, even a minor adjustment in the step size can lead to a significant change in the local error. This sensitivity provides an effective feedback mechanism for adaptive step size algorithms: when the error exceeds the allowable tolerance, reducing the step size can rapidly decrease the local error to maintain accuracy; conversely, when the error is well below the tolerance, increasing the step size can enhance computational efficiency.

Based on this principle, the adaptive strategy is driven by comparing the local error e_{local} with a predefined error tolerance ε. The specific adjustment rules are as follows:

- If $e_{\text{local}} < \varepsilon$: The current accuracy is higher than necessary, so the step size can be increased to improve efficiency.
- If $e_{\text{local}} > \varepsilon$: The current error exceeds the limit, so the step size must be reduced to meet the accuracy requirement.
- If $e_{\text{local}} = \varepsilon$: The current step size is optimal, so it is kept unchanged.

3.4. Adaptive Step Size Algorithm Flow

The detailed procedure of the adaptive Runge-Kutta method is summarized in Algorithm 1, which outlines the step size adjustment mechanism based on local error estimation and tolerance control.

3.5. Expansion Coefficient Optimization Processing

To balance computational efficiency and numerical stability, the expansion coefficient must be carefully selected. The following subsections detail the selection range, optimization objective, and the specific optimization procedure.

3.5.1. Setting of the Expansion Coefficient Range

In the step size adjustment process, a step size expansion coefficient is introduced to ensure the stability of numerical computation and avoid drastic fluctuations in step size caused by abrupt error changes. This coefficient regulates the range of step size variation to achieve a smooth transition under complex conditions, thereby improving the stability and accuracy of numerical computation. However, optimal computational efficiency cannot be achieved by relying solely on the expansion coefficient, so its optimization is necessary.

Considering the error characteristics and numerical stability requirements of the fourth-order Runge-Kutta method, the range of the expansion coefficient is set as follows:

- Expansion coefficient range: $\beta_{\text{up}} \in [1.1, 1.5]$, with interval $\Delta\beta = 0.05$;
- Contraction coefficient: $\beta_{\text{down}} = 0.5$ (fixed).

3.5.2. Optimization Objective and Evaluation Index

The total number of calculations N under the preset error tolerance ε is used as the only evaluation index. The optimization objective is defined as:

$$\beta^* = \min_{\beta \in \Omega} N(\beta) \tag{6}$$

where Ω is the set of candidate expansion coefficients, and $N(\beta)$ is the total number of iterative steps for solving the entire interval $[x_0, X]$ using the adaptive step size method with coefficient β.

The total number of steps N directly reflects the computational efficiency: the smaller N is, the fewer computations are required under the same error tolerance, and the higher the efficiency. Using N as the index is appropriate because it is closely related to practical performance, easy to obtain through numerical simulation, and avoids complicated theoretical derivation.

In traditional adaptive step size schemes, only the single-step local error is usually constrained to be less than ε. However, as the number of steps increases, the accumulation of local errors may cause the global cumulative error to gradually exceed the allowable range. Therefore, when multiple expansion coefficients yield the same computational cost, the cumulative error—defined as the sum of local errors over all steps—can be used as an additional selection criterion. Moreover, when judging whether to expand the step size, we must check not only the local error of the current step but also whether the sum of the current local error and the existing cumulative error remains less than ε. Step size expansion is allowed only if this total error satisfies the constraint, thus maintaining a reasonable balance between computational efficiency and accuracy.

3.5.3. Optimization Procedure of the Expansion Coefficient

1. **Initialization**: Set the initial-value problem of the differential equation, the solution interval $[x_0, X]$, the preset error tolerance ε, and the candidate range Ω_{up} of the expansion coefficient (the contraction coefficient is fixed at $\beta_{\text{down}} = 0.5$);
2. **Traversal calculation**: For each expansion coefficient $\beta_{\text{up}} \in \Omega_{\text{up}}$, run the adaptive step size Runge-Kutta algorithm (with $\beta_{\text{down}} = 0.5$) and record the total number of steps $N(\beta_{\text{up}})$ for the entire interval;
3. **Index comparison**: Compare $N(\beta_{\text{up}})$ for all coefficients. If multiple coefficients yield the same number of steps, further compare the cumulative error and select the one with the smaller cumulative error, denoted as β^*_{up};
4. **Verification and determination**: Substitute the optimal expansion coefficient β^*_{up} and fixed contraction coefficient $\beta_{\text{down}} = 0.5$ into the algorithm, re-verify the computational accuracy within ε , and confirm the optimal coefficient combination $\beta^* = (\beta^*_{\text{up}}, 0.5)$ after ensuring no loss of accuracy.

4. Numerical Experiments and Result Analysis

To further verify the effectiveness and advancement of the adaptive step size Runge-Kutta method, especially its excellent performance in significantly avoiding error accumulation, this section will carry out detailed explanations and demonstrations through numerical experiments. In this section, the traditional fixed step size fourth-order Runge-Kutta method and the adaptive step size Runge-Kutta method are used for comparison to numerically solve the initial value problem of differential equations.

4.1. Numerical Experiment Case

An initial value problem of differential equations is as follows:

$$\begin{cases} \frac{dy}{dx} = y - \frac{2x}{y}, & 0 < x < 1, \\ y(0) = 1. \end{cases} \tag{7}$$

Its analytical solution is:

$$y = \sqrt{2x+1}. \tag{8}$$

This analytical solution provides an accurate reference benchmark for error analysis.

4.2. Numerical Solution Scheme 1

For the initial value problem in Equation 7, both the fixed-step fourth-order Runge-Kutta method and the adaptive-step Runge-Kutta method were used to solve it, resulting in two sets of schemes. In the first set of schemes, considering that the fixed step size must meet the error limit requirements under an interval length of 1, the initial step size was set to 0.05. According to the number of calculations and cumulative errors, the optimization process of the expansion coefficient is shown in Table 1, Considering the cumulative error, the optimal expansion coefficient β_{up} is selected as 1.2 under the condition of minimizing the number of steps. The calculation results of the fixed-step part are shown in Table 2, while the calculation results of the adaptive-step are shown in Table 3.

Table 1
Expansion Coefficient Traversal Process (Scheme 1)

β_{up}	Number of Steps	Cumulative Error
1.10	12	1.08×10^{-6}
1.15	12	1.37×10^{-6}
1.20	**11**	$\mathbf{1.47\times 10^{-6}}$
1.25	11	1.56×10^{-6}
1.30	11	1.61×10^{-6}
1.35	11	1.67×10^{-6}
1.40	11	1.75×10^{-6}
1.45	11	1.78×10^{-6}
1.50	11	1.79×10^{-6}

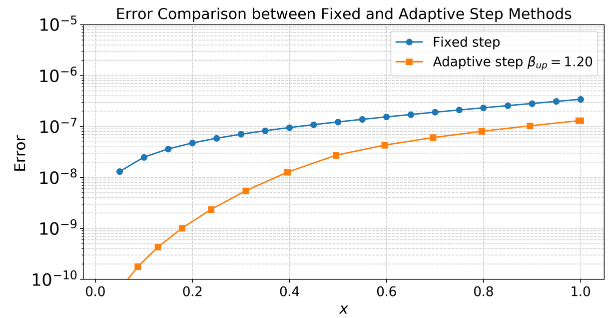

Figure 1: Accuracy comparison diagram of the two methods (Error Tolerance: 5×10^{-7})

Under the condition of meeting the preset error limit, the comparison of the accuracy of the two methods is shown in Figure 1. Under the premise of similar overall calculation accuracy (the actual error of the adaptive-step method is smaller and the accuracy is higher), the fixed-step method requires 20 iterations to complete the solution, while the adaptive-step method only needs 11 iterations, reducing the number of calculations by 45% and significantly improving the calculation efficiency.

4.3. Numerical Solution Scheme 2

To further verify the performance of the algorithm under stricter error constraints, a second set of schemes was set up, with the preset error limit increased to 1×10^{-9} and the interval length remaining at 1. To meet this stricter error limit requirement, the fixed step size was adjusted to 0.02, following the same optimization process of the expansion coefficient as described in Section 3.5, the optimal expansion coefficient was sought under the current stricter error limit conditions, and the traversal results of the expansion coefficient are shown in Table 4. Considering the cumulative

Table 2
Calculation Results with Fixed Step Size (Scheme 1)

Node	Calculation Result	Exact Value	Error	Number of Calculations
0.05	1.048808861	1.048808848	1.30×10^{-8}	1
0.10	1.095445140	1.095445115	2.49×10^{-8}	2
		......		
0.90	1.673320335	1.673320053	2.82×10^{-7}	18
0.95	1.702938947	1.702938637	3.10×10^{-7}	19
1.00	1.732051148	1.732050808	3.41×10^{-7}	20

Table 3
Calculation Results with Adaptive Step Size (Scheme 1)

Node	Calculation Result	Exact Value	Error	Number of Calculations
0.06	1.058300526	1.058300524	1.96×10^{-9}	1
0.132	1.124277552	1.124277546	6.20×10^{-9}	2
0.2184	1.198665941	1.198665925	1.56×10^{-8}	3
0.3184	1.279374881	1.279374847	3.36×10^{-8}	4
0.4184	1.355286000	1.355285948	5.24×10^{-8}	5
0.5184	1.427165095	1.427165022	7.30×10^{-8}	6
0.6184	1.495593624	1.495593528	9.61×10^{-8}	7
0.7184	1.561025427	1.561025304	1.22×10^{-7}	8
0.8184	1.623822803	1.623822650	1.53×10^{-7}	9
0.9184	1.684280450	1.684280262	1.88×10^{-7}	10
1	1.732051026	1.732050808	2.18×10^{-7}	11

Table 4
Expansion Coefficient Traversal Process (Scheme 2)

β_{up}	Number of Steps	Cumulative Error
1.10	**27**	**3.69×10^{-8}**
1.15	28	3.54×10^{-8}
1.20	28	3.45×10^{-8}
1.25	28	3.23×10^{-8}
1.30	29	2.76×10^{-8}
1.35	29	2.99×10^{-8}
1.40	29	3.04×10^{-8}
1.45	31	2.50×10^{-8}
1.50	34	1.53×10^{-8}

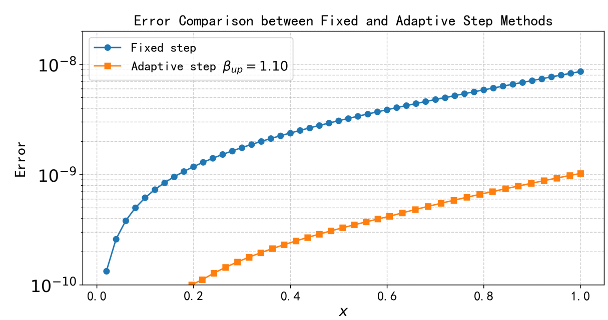

Figure 2: Accuracy comparison diagram of the two methods (Error Tolerance: 1×10^{-9})

error, the optimal coefficient beta is selected as 1.1 under the condition of minimizing the number of computations. The initial step size was set to 0.02. According to the number of calculations and cumulative errors, the fixed-step and adaptive-step methods were then numerically solved, and the corresponding calculation results were sorted into Tables 5 and 6, respectively, for comparison and analysis of the algorithm's accuracy maintenance ability and efficiency advantage under stricter error constraints.

As shown in Figure 2, the adaptive-step method has a significant advantage in calculation efficiency: compared to the 50 iterations of the fixed-step method, the adaptive-step method only requires 27 iterations to complete the solution, improving the calculation efficiency by 46%. However, although the local error of each step is strictly controlled within the preset range, with the increase in the number of iterations, the cumulative effect of local errors gradually becomes apparent, causing the global error to exceed the preset error limit at the end of the interval. This indicates that the current step size adjustment strategy based solely on local error has not fully considered the impact of global cumulative error on the final accuracy. In the future, by introducing cumulative error constraints in the step size amplification judgment, a better balance between calculation efficiency and global accuracy can be achieved.

5. Conclusion

This study proposes an adaptive-step Runge-Kutta method, which demonstrates unique technical advantages and

Table 5
Calculation Results with Fixed Step Size (Scheme 2)

Node	Calculation Result	Exact Value	Error	Number of Calculations
0.02	1.0198039029	1.0198039027	1.33×10^{-10}	1
0.04	1.0392304848	1.0392304845	2.60×10^{-10}	2
		……		
0.96	1.7088007570	1.7088007491	7.97×10^{-9}	48
0.98	1.7204650617	1.7204650534	8.28×10^{-9}	49
1.00	1.7320508162	1.7320508076	8.59×10^{-9}	50

Table 6
Calculation Results with Adaptive Step Size (Scheme 2)

Node	Calculation Result	Exact Value	Error	Number of Calculations
0.022	1.02176318197	1.02176318196	1.32×10^{-11}	1
0.0462	1.04517941047	1.04517941044	3.32×10^{-11}	2
		……		
0.919114	1.68470396852	1.68470396391	4.61×10^{-9}	25
0.961985	1.70996228488	1.70996227982	5.06×10^{-9}	26
1	1.73205081302	1.73205080757	5.45×10^{-9}	27

significant effectiveness in the numerical solution of differential equations. Through numerical experiments and specific case studies, its practical effect has been verified. The results show that compared with the fixed-step Runge-Kutta method, the adaptive-step adjustment strategy can significantly improve calculation efficiency while maintaining similar or even better calculation accuracy. The core innovation lies in the construction of an adaptive-step adjustment mechanism based on real-time error assessment, which dynamically optimizes the step size according to the error situation during the current solution process, minimizing redundant calculations while enhancing the numerical stability of the algorithm, especially in handling high-precision requirements or strongly nonlinear differential equations, it shows superior comprehensive performance.

Although the above progress has been made, there are still several areas for improvement in this study. First, the balance between the accuracy and efficiency of the error assessment mechanism, which is the core of the algorithm, still has room for improvement. Although the current error estimation model can meet the basic requirements, there is still a need for in-depth exploration in building more refined error prediction models and developing more efficient error quantification algorithms. Secondly, the optimization of step size adjustment strategies is also a key research direction. How to achieve efficient and reasonable step size adjustment while maintaining numerical stability remains a key issue to be solved in this field. Particularly, the current step size adjustment strategy based solely on single-step local error has not fully considered the impact of global cumulative error on the final accuracy, which may lead to the global error exceeding the preset error limit when solving to the end of the interval. This is also one of the key directions for future optimization. In addition, the diversity of different types of differential equations and their initial conditions may lead to significant differences in algorithm performance. To further expand the application scope of this method, future research needs to conduct systematic analysis on the differentiated equation characteristics and initial conditions, clarify the applicable boundaries, and optimize parameter configuration, so as to propose more universal adaptive strategies.

Acknowledgement

This work was supported by the Teaching Research Project of Higher Education Institutions of Anhui Province under Grant No. 2022jyxm1161, the Natural Science Research Projects of Higher Education Institutions of Anhui Province under Grant Nos. 2023AH050429 and 2023AH05-0413, the Undergraduate Teaching Engineering Projects of Fuyang Normal University under Grant Nos. 2022JYXM00-09, 2023JYXM0039, and 2025JYXM0011, the Teacher Education Collaborative Improvement Program of Fuyang Normal University under Grant No. 2025XTKYTD01, and the New Undergraduate Program Quality Improvement Project of Anhui Province under Grant No. 2024xjzlts035. The authors declare that there is no conflict of interest.

References

[1] Z. Dou, G. Bai, Z. Han, W. Li, Y. Li, PFGL-Net: A personalized federated graph learning framework for privacy-preserving disease prediction, Journal of Artificial Intelligence Research, 2025, vol. 2, no. 2, pp. 12–23.

[2] Q. Cai, Y. Xu, A physics-informed neural network method for solving forward and inverse problems of thermal-humidity coupling model in single-layer fabrics, Software Engineering, 2025, vol. 28, no. 1, pp. 73–78.

[3] K. Wang, Z. Li, Study on the impact of grazing strategies on soil and vegetation and soil moisture prediction, Software Engineering, 2024, vol. 27, no. 9, pp. 67–72.

[4] M. Ibrar, Y. Sun, SEIR model based epidemic transmission risk deep prediction, IFS/ACM Transactions on Machine Learning, 2024, vol. 1, no. 1, pp. 25–31.

[5] R. K. Jain, M. K. Jain, Optimum Runge-Kutta Fehlberg methods for first order differential equations, IMA Journal of Applied Mathematics, 1971, vol. 8, no. 3, pp. 386–396.

[6] G. Amanatidis, F. Fusco, P. Lazos, S. Leonardi, R. Reiffenhäuser, Fast adaptive non-monotone submodular maximization subject to a knapsack constraint, Journal of Artificial Intelligence Research, 2022, vol. 74, pp. 661–690.

[7] Y. Chen, T. Dey, A. Kuhnle, Scalable distributed algorithms for size-constrained submodular maximization in the MapReduce and adaptive complexity models, Journal of Artificial Intelligence Research, 2024, vol. 80, pp. 1575–1622.

[8] Y. Wang, F. Ma, Z. Yang, Y. Zhu, B. Yang, P. Shen, L. Yun, Rumor detection with adaptive data augmentation and adversarial training, Journal of Artificial Intelligence Research, 2025, vol. 82, pp. 1175–1204.

[9] C. Nwankwo, W. Dai, An adaptive and explicit fourth order Runge-Kutta-Fehlberg method coupled with compact finite differencing for pricing American put options, Japan Journal of Industrial and Applied Mathematics, 2021, vol. 38, no. 3, pp. 921–946.

[10] Q. Li, N. Wang, D. Yi, Numerical analysis, 5 ed., Huazhong University of Science and Technology Press, Wuhan, 2018.

[11] M. He, Analysis on a class of new algorithm of fourth order R-K method, College Mathematics, 2004, vol. 20, no. 1, pp. 72–76.

[12] D. Tan, Z. Chen, On a general formula of fourth order Runge-Kutta method, Journal of Mathematical Science & Mathematics Education, 2012, vol. 7, no. 2, pp. 1–10.

[13] M. Ökten Turacı, T. Öziş, Derivation of three-derivative Runge-Kutta methods, Numerical Algorithms, 2017, vol. 74, pp. 247–265.

[14] Y. Wu, X. Zhang, C. Xiong, Optimal algorithm of strongly stable Runge-Kutta in three-step order, Mathematica Applicata, 2005, vol. 18, no. S1, pp. 57–61.

[15] K. Xia, Construction and research of high-order Runge-Kutta methods, Ph.D. thesis, Shanghai Normal University, Shanghai, 2017.

[16] R. A. Khurma, The Runge-Kutta optimization algorithm: A comprehensive survey of methodology, variants, applications, and performance evaluation, Archives of Computational Methods in Engineering, 2025, vol. 32, no. 8, pp. 5075–5122.

[17] D. J. Walter, A. Manera, Adaptive burnup stepsize selection using control theory for 2-D lattice depletion simulations, Progress in Nuclear Energy, 2016, vol. 88, pp. 218–230.

[18] H. Musa, I. Saidu, M. Y. Waziri, A simplified derivation and analysis of fourth order Runge Kutta method, International Journal of Computer Applications, 2010, vol. 9, no. 8, pp. 51–55.

[19] M. H. Carpenter, C. A. Kennedy, H. Bijl, S. A. Viken, V. N. Vatsa, Fourth-order Runge-Kutta schemes for fluid mechanics applications, Journal of Scientific Computing, 2005, vol. 25, pp. 157–194.

[20] P.-W. Fok, A linearly fourth order multirate Runge-Kutta method with error control, Journal of Scientific Computing, 2016, vol. 66, no. 1, pp. 177–195.

[21] T. Xu, A. Yang, P. Guo, M. Yang, Z. Zhao, W. Wan, Optimization of digital back-propagation for coherent optical fiber communication systems using fourth-order Runge-Kutta in the interaction picture method, Optics Express, 2025, vol. 33, no. 2, pp. 2082–2100.