mathematics

MDPI

Article

Evolutionary Optimisation of Runge–Kutta Methods for Oscillatory Problems

Zacharias A. Anastassi

Institute of Artificial Intelligence, School of Computer Science and Informatics, De Montfort University, Leicester LE1 9BH, UK; zacharias.anastassi@dmu.ac.uk

Abstract

We propose a new strategy for constructing Runge–Kutta (RK) methods using evolutionary computation techniques, with the goal of directly minimising global error rather than relying on traditional local properties. This approach is general and applicable to a wide range of differential equations. To highlight its effectiveness, we apply it to two benchmark problems with oscillatory behaviour: the (2+1)-dimensional nonlinear Schrödinger equation and the N-Body problem (the latter over a long interval), which are central in quantum physics and astronomy, respectively. The method optimises four free coefficients of a sixth-order, eight-stage parametric RK scheme using a novel objective function that compares global error against a benchmark method over a range of step lengths. It overcomes challenges such as local minima in the free coefficient search space and the absence of derivative information of the objective function. Notably, the optimisation relaxes standard RK node bounds ($c_i \in [0, 1]$), leading to improved local stability, lower truncation error, and superior global accuracy. The results also reveal structural patterns in coefficient values when targeting high eccentricity and non-sinusoidal problems, offering insight for future RK method design.

Keywords: evolutionary computation; parametric Runge–Kutta methods; particle swarm optimisation; (2+1)-dimensional nonlinear Schrödinger equation; N-Body problem

MSC: 65L06; 90C56; 90C26

check for updates

Academic Editor: Jinhai Li

Received: 8 July 2025
Revised: 18 August 2025
Accepted: 19 August 2025
Published: 31 August 2025

Citation: Anastassi, Z.A. Evolutionary Optimisation of Runge–Kutta Methods for Oscillatory Problems. *Mathematics* **2025**, *13*, 2796. https://doi.org/10.3390/math13172796

Copyright: © 2025 by the author. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Even though the class of Runge–Kutta (RK) methods has been well established and studied for decades, a complete determination of all their coefficient values based on their performance remains unresolved to this day, except for RK methods with a small number of stages. This is due to the large number and the strong nonlinearity of the conditions that must be satisfied to achieve high-order accuracy. The main approach to address this complexity is to predetermine a set of coefficients, often by setting them to zero, to simplify the resulting system enough to be solvable [1] and, to the same end, to consider additional simplifying assumptions [2].

Traditionally, once the order conditions have been satisfied, the remaining free coefficients are used to satisfy other local properties, such as increased phase-lag order and/or amplification error order, phase fitting, amplification error fitting, exponential/trigonometric fitting, and minimisation of the local truncation error, among others. The problems considered here, the (2+1)-dimensional nonlinear Schrödinger (NLS) equation, the Two-Body (TB) problem and its generalisation, i.e., the N-Body (NB) problem, all exhibit

oscillatory and periodic behaviour. As such, methodologies targeting oscillatory and/or periodic systems have been widely applied. For example, a phase- and amplification-fitted Runge–Kutta–Nyström pair has been derived in [3] for the numerical solution of oscillatory systems. For the same type of problems, Abdulsalam et al. [4] developed a multistep Runge–Kutta–Nyström method with frequency-dependent coefficients. More specific problem approaches have been adopted as well, and trigonometrically-fitted methods have been developed for the Schrödinger equation in [5] and orbital problems in [6]. Further minimisation of the local truncation error coefficients or other quantities, e.g., the Lagrangian of the system, has also been investigated [7], while invariant domain preserving and mass conservative explicit Runge–Kutta methods have also been developed [8]. Finally, explicit integrators that obey local multisymplectic conservation laws, based on partitioned Runge–Kutta methods, have been investigated in [9].

However, a key limitation of these classical approaches is their reliance on local properties. That is, they ensure the method performs well locally, perhaps by integrating simple problems, based on the assumption that acquiring certain local features implies good global behaviour. This assumption, although common, is not always verified for complex or long-time integration problems. The global error in the real problem remains the ultimate metric for assessing a method's performance, and this is the approach we adopt here. Rather than analysing local accuracy or stability conditions, we directly target the global error of the problem.

For this reason, we consider a parametric explicit RK method with eight stages and sixth-order accuracy, which has four free coefficients and can be optimised to efficiently solve the NLS, TB, and NB problems. In a previous work [10], we attempted to optimise this method for the NLS equation only. However, several significant barriers were encountered: the highly nonlinear behaviour of the optimisation landscape, the presence of local minima in the free coefficient space, the expensive computation of the global error via time integration, and the black-box nature of the objective function, due to the absence of derivative information. These factors make a systematic investigation of the optimisation problem particularly difficult and suggest the need for non-traditional approaches.

Utilising artificial intelligence methodologies to solve scientific and engineering problems is increasingly common; see [11] and references therein. Computational intelligence techniques [12], and in particular evolutionary computation (EC) methods [13], are well-suited for optimising parametric numerical methods. Particle swarm optimisation (PSO) is among the most prominent of these methods [14], and differential evolution techniques are also gaining popularity [15]. There is research that involves combining optimisation and RK methods, but not with the ultimate goal of developing an optimised RK method. For example, in [16], the RK method is used to define the update rule in a metaheuristic optimiser. Notwithstanding, there have been limited applications of EC techniques to optimise finite difference schemes and even fewer for Runge–Kutta methods, despite the fact that the complex nature and size of the system of nonlinear equations would benefit from such techniques. Babaei in [17] develops new RK methods for the solution of initial value problems; however, the process does not involve any order conditions. Neelan and Nair [18] developed hyperbolic Runge–Kutta methods with an optimised stability region using evolutionary algorithms. Optimisation of parametric RK methods has also been achieved using differential evolution by Kovalnogov et al. in [19] and by Jerbi et al. in [20].

In this article, we use the particle swarm optimisation algorithm [21] to determine the optimal values of the four free coefficients in our parametric RK method [10]. PSO, which continues to evolve, is well-suited to overcoming the aforementioned obstacles [22], as it does not require derivative information and allows for efficient global exploration of the coefficient space. It also provides a systematic way to experiment with otherwise

impractical constraints, such as relaxing the node bounds $c_i \in [0,1]$ to a more general interval $c_i \in [a,b] \supset [0,1]$. The use of PSO not only enhances the ability to identify effective RK coefficients but also enables the discovery of optimal methods tailored to real-world problems in quantum physics, astronomy, and related fields.

In summary, the main contributions of this article are the formulation of an evolutionary optimisation framework for Runge–Kutta methods that targets global accuracy rather than local properties, the introduction of a benchmark-based objective function, the relaxation of conventional coefficient bounds to enhance stability and error properties, and the identification of structural patterns in optimal coefficients for highly oscillatory regimes.

The structure of this paper is as follows:

- In Section 2, we introduce the Runge–Kutta framework, formulate the parametric optimisation problem, and describe the implementation of the evolutionary computation methods;
- In Section 3, we present the numerical problems used for training and testing the methods;
- In Section 4, we report the optimal Runge–Kutta coefficients obtained for each training problem;
- In Section 5, we analyse the error and stability properties of the resulting methods;
- In Section 6, we present numerical results evaluating the performance of the optimised methods;
- In Section 7, we discuss key trends and implications of the results;
- In Section 8, we summarise the main contributions.

2. Evolutionary Optimisation of Parametric RK Methods

2.1. Overview of Runge–Kutta Methods

The problems under investigation fall under the same category of initial value problems, presented in the autonomous form:

$$\begin{aligned} \frac{du}{dt} &= f(u) \\ u(0) &= u_0 \end{aligned} \tag{1}$$

where u implicitly contains t and potentially x and y and all derivatives with respect to variables other than t, if they exist, which can be semi-discretised via finite differences or replaced by a theoretical solution if available. The resulting problem can, in that case, be solved using the method of lines.

An s-stage explicit Runge–Kutta method for the solution of Equation (1) is presented below:

$$\begin{cases} u_{n+1} = u_n + h \sum_{i=1}^{s} b_i k_i \\ k_i = f(t_n + c_i h,\ u_n + h \sum_{j=1}^{i-1} a_{ij} k_j),\ i = 1, \ldots, s \end{cases} \quad or \quad \begin{array}{c|c} c & A \\ \hline & b \end{array} \tag{2}$$

where $A = [a_{ij}] \in \mathbb{R}^{s \times s}$ is strictly lower triangular, $b = [b_i]^T \in \mathbb{R}^s$ (weights), $c = [c_i] \in \mathbb{R}^s$ (nodes), and all coefficients $a_{ij}, b_i, c_i \in \mathbb{R}$, for $i = 1, 2, \ldots, s$ and $j = 1, 2, \ldots, i-1$, define the method [2]. Here, $h = \Delta t$ is the step length in time, and f is defined in the form $u' = f(x, y, t, u)$.

2.2. Parametric Runge–Kutta Method

We consider a general parametric method, which was developed in [10] for the solution of Equation (1). It is an eight-stage, sixth-order explicit RK method and has 10 fixed coefficients, namely $c_3 = \frac{1}{6}$, $c_6 = \frac{1}{2}$, $c_8 = 1$, $b_2 = 0$, $b_3 = 0$, $a_{42} = 0$, $a_{52} = 0$, $a_{62} = 0$, $a_{72} = 0$,

$a_{82} = 0$, while 29 coefficients depend on one or more of $\{c_2, c_4, c_5, c_7\}$, which serve as the four degrees of freedom in the optimisation.

In [10], we attempted to optimise the method using a standard optimisation technique, with a progressively decreasing step length Δc from 0.1 to 0.001 for all four free coefficients c_2, c_4, c_5, c_7. The optimisation yielded the following results as optimal: $\{c_2 = 0.007$, $c_4 = 0.208$, $c_5 = 0.441$, $c_7 = 0.915\}$. Using evolutionary optimisation, we will identify improved variants of the method described above, illustrating the limitations of conventional optimisation in finding globally optimal values.

2.3. The Optimisation Problem

The objective of the optimisation is to explore the free coefficient space thoroughly and efficiently across a range of step lengths to ensure that the optimal RK method performs well across all step lengths. Many setups were successful, with varying efficiencies. For example, when a reliable benchmark method exists across all accuracy levels and can act as a benchmark, then it is useful to optimise the mean ratio of the errors of the method under development to those of the benchmark method.

We define the objective function C as the mean ratio of the maximum norm of the global error produced by the new method over that of the benchmark method, averaged over several runs with different step lengths. Formally, the objective function is defined as

$$\underset{c_2, c_4, c_5, c_7}{\text{minimise}} \quad C = \frac{1}{N} \sum_{i=1}^{N} \frac{\|E_i\|_\infty}{\|\bar{E}_i\|_\infty} \tag{3}$$

C: The objective function to be minimised. It represents the average relative error of the new method compared with the benchmark method with the same stages and order.
N: The total number of runs, each corresponding to a specific step length h_i.
$\|E_i\|_\infty$: The maximum norm of the global error of the new method in the i-th run.
$\|\bar{E}_i\|_\infty$: The maximum norm of the global error of the benchmark method in the i-th run.

The benchmark method is only run once per run. We chose the optimal method developed in [10], also presented in Section 4, to act as the benchmark method, due to its consistent performance across all step lengths.

In order to ensure a uniformly good performance for all accuracies, we select a range of different step lengths h_i that produce results with varying accuracies, avoiding both excessively large (unstable) and excessively small (round-off dominated) values. The best results were obtained from the geometric sequence:

$$h_i = \frac{2}{5} \cdot 2^{-(i-1)}, \quad i = 1, \ldots, 6, \quad t \in [0, 2.5], \quad \text{for the NLS Equation (10)} \tag{4a}$$

$$h_i = 2^{-\frac{i-1}{2}}, \quad i = 1, \ldots, 10, \quad t \in [0, 100], \quad \text{for the Two-Body problem (17)} \tag{4b}$$

Several variations were also examined, but the above combinations provided the most reliable results over a set of individual runs.

Each run involves solving a problem using both the new and the benchmark methods under identical conditions (e.g., same problem, same step length). The global error is measured as the maximum absolute difference between the numerical and exact (or reference) solution over time. By averaging the ratio $\|E_i\|_\infty / \|\bar{E}_i\|_\infty$ across all N runs, we obtain a scalar performance measure C that reflects the overall accuracy of the new method relative to the benchmark. A value of $C < 1$ indicates that the new method, on average, outperforms the benchmark in terms of maximum global error.

For optimising the four coefficients to perform well when integrating the NLS Equation (10), described in Section 3.1, we use the same equation over a shorter interval as a training problem, in contrast to the test problem.

To improve the performance of the methods in the N-Body problem (19), described in Section 3.3, instead of using it as a training problem and test problem, we select the Two-Body problem (17), described in Section 3.2, to tune the coefficients, as it is computationally cheaper yet shares similar dynamics. We experimented with three different eccentricities, but $e = 0.05$ yielded the best results, as it is the dominant eccentricity in our problem, as explained in Section 3.3.

Alternatively, we tested the following objective function:

$$\underset{c_2, c_4, c_5, c_7}{\text{minimise}} \quad C = \frac{1}{N} \sum_{i=1}^{N} \frac{\|E_i\|_\infty}{h_i^{\hat{p}}} \tag{5}$$

where $\hat{p}$ is the experimental order of accuracy, although it can be empirically tuned, as its purpose in the above formula is to balance the scale of the ratios within the objective function (an equivalent objective function was proposed in [20]). However, despite offering independence from a benchmark method, the function was less consistent and required fine-tuning of the parameter $\hat{p}$ for comparable results, so it was not adopted in our main optimisation.

We have one main boundary constraint for all coefficients $c_i \in [0, 1]$, which is the typical constraint when developing Runge–Kutta methods. However, this constraint offers limited practical benefit for the problems studied. As we will show, removing this bound allows for smaller maximum global error, lower local error constants, and larger stability intervals, all simultaneously. The efficient implementation of PSO allows this type of experimentation with different setups, enabling the removal of traditional constraints that are not essential. Even when bounds are enforced, PSO yields better results than classical optimisation techniques.

To conclude, we present the optimisation problem under investigation below:

- Minimise the objective function C, defined in (3),
- with respect to the four coefficients c_2, c_4, c_5, c_7,
- in combination with the step lengths defined in (4),
- subject to $c_i \in [0, 1]$ (optional).

2.4. Implementation of Particle Swarm Optimisation

Because the goal is to minimise the global error, and the objective function lacks derivative information, derivative-based methods are inapplicable. For instance, optimisation methods such as gradient descent could be considered; however, the absence of derivative information and the problem's strong nonlinearity render derivative-free approaches more appropriate. This motivates the use of derivative-free global optimisation strategies, such as evolutionary computation methods.

Particle Swarm Optimisation (PSO), inspired by the social behaviour of bird flocks and fish schools, is a population-based stochastic method that iteratively improves candidate solutions (in our case, quadruples of free coefficients) through individual experience and social learning. PSO is a well-established and widely used optimisation technique, known for its robustness, simplicity, and flexibility across diverse problem domains. We selected PSO for its balance between exploration and exploitation and relatively few control parameters [21–23]. Its straightforward tuning requirements and fast convergence behaviour further support its practical appeal. Additionally, its structure naturally supports parallel evaluation of candidates, which is beneficial given the computational cost of calculating the global error via time integration. While other evolutionary methods, such as differential evolution, have been applied in similar optimisation problems, PSO best meets our requirements for efficiency, robustness, and ease of integration into the optimisation framework.

We used a standard PSO algorithm described by the update rules below (Algorithm 1).

Algorithm 1 Neighbourhood-Best PSO with Adaptive Parameters

The position of particle i at iteration $t+1$ is updated as

$$\mathbf{x}_i^{t+1} = \mathbf{x}_i^t + \mathbf{v}_i^{t+1} \tag{6}$$

The velocity is updated according to

$$\mathbf{v}_i^{t+1} = \omega^t \, \mathbf{v}_i^t + c_1 r_1 \left(\mathbf{p}_{i,\text{best}} - \mathbf{x}_i^t \right) + c_2 r_2 \left(\mathbf{p}_{\text{neigh},i} - \mathbf{x}_i^t \right) \tag{7}$$

where

- ω^t is the inertia weight at iteration t;
- c_1 and c_2 are the cognitive and social acceleration coefficients;
- $r_1, r_2 \sim U(0,1)$ are independent uniform random numbers drawn for each particle and dimension at every iteration;
- $\mathbf{p}_{i,\text{best}}$ is the best position found by particle i;
- $\mathbf{p}_{\text{neigh},i}$ is the best position found in the neighbourhood of particle i;
- N^t is the neighbourhood size at iteration t.

The inertia weight is adapted at each iteration as

$$\omega^{t+1} = \begin{cases} \min\Big(\max(2\omega^t, \omega_{\min}),\, \omega_{\max}\Big), & \text{if stallCounter} < 2 \\ \min\Big(\max(0.5\omega^t, \omega_{\min}),\, \omega_{\max}\Big), & \text{if stallCounter} > 5 \\ \omega^t, & \text{otherwise} \end{cases} \tag{8}$$

where $\omega_{\min}$ and $\omega_{\max}$ bound the inertia range, and `stallCounter` counts consecutive iterations without improvement. The neighbourhood size N^t is adapted as

$$N^{t+1} = \begin{cases} N_{\min}, & \text{if improvement found} \\ \min(N_{\max},\, N^t + N_{\min}), & \text{otherwise} \end{cases} \tag{9}$$

where $N_{\min}$ and $N_{\max}$ are the minimum and neighbourhood sizes, respectively.

In all experiments, the parameters were set as follows: $c_1 = 1.49$, $c_2 = 1.49$, $\omega_{\min} = 0.1$, $\omega_{\text{init}} = \omega_{\max} = 1.1$, $N_{init} = N_{\min} = S/4$, and $N_{\max}$ equal to the swarm size S. These choices follow commonly recommended guidelines about parameter values, such as the cognitive and social acceleration coefficients [23,24] and update rules, specifically the inertia and neighbourhood update rules [25], while accommodating the specific requirements of our problems. For example, there are numerous articles and surveys advocating for the use of neighbourhoods in the PSO algorithm [26,27]. Other authors propose even more advanced variants [28,29], however, we prefer the technique described above, which accommodates ease of integration without compromising on efficiency or robustness.

The population size is now generally expected to be much larger than the 20–50 particles suggested in early PSO studies, as observed in many applications [30]. In our study, the criteria for selecting a population size S are, in order of priority: (i) achieving the lowest possible objective-function values among the best solutions, (ii) maintaining solution diversity, and (iii) if these two are comparable, maximising the computational efficiency of the optimisation process. We analysed the sensitivity of these criteria with respect to $S = 1000, 2000, 4000$ when training for the Two-Body problem (e = 0.05) for the step lengths in (4a) and objective function (5), based on nine independent runs per population size in this representative test configuration. The analysis revealed equivalent objective function values with means $\mu_1 = 0.00214$, $\mu_2 = 0.00256$, $\mu_4 = 0.00207$ and standard deviations $\sigma_1 = 0.00040$, $\sigma_2 = 0.00050$, $\sigma_4 = 0.00055$ for $S = 1000, 2000, 4000$ respectively. The decid-

ing factor in selecting the population size was the diversity of the solutions: the average pairwise Euclidean distances were $d_1 = 0.114$, $d_2 = 0.520$, and $d_4 = 0.344$, indicating that the solutions for $S = 2000$ were more spread out and less prone to clustering in a narrow region. Furthermore, for $S = 4000$, the system exhibited instability due to high memory usage from parallelisation. Thus, $S = 2000$ provided the most robust and diverse results. For completeness, the average number of function evaluations was $c_1 = 1.93 \times 10^5$, $c_2 = 2.90 \times 10^5$, and $c_4 = 8.02 \times 10^5$.

Nonetheless, it was necessary to reinitiate the population between runs, as this alone could redirect the swarm to more optimal regions of the search space. Each run converged in approximately 100–300 iterations, which resulted naturally without imposing any hard iteration limit, with a maximum of 50 stall iterations, where the best solution has not changed by more than the function tolerance, which is set to machine epsilon, approximately 2.2×10^{-16} for our machine.

During the exploration phase of the optimisation, we observed that the optimal solution exhibited high mobility across the search space, overcoming local barriers and valleys. Different settings and initial populations led to distinct optimal solutions, and this behaviour further indicates the complex nature of the optimisation problem.

More than 20 runs were attempted for each of the four problems, the NLS Equation (10), and the Two-Body problem (17) with three different eccentricities, $e = 0.00$, $e = 0.05$, and $e = 0.25$. The solution derived from each of these runs was tested over the full interval for either the NLS Equation (10) or the N-Body problem (19), depending on the problem for which it was optimised. All experiments were conducted on a computer with an Intel i9 processor (14 cores) and 32 GB of RAM. MATLAB R2024b was used for numerical computations, and Maple 2024 was used for symbolic computations.

Before presenting the optimised coefficients obtained from the training process, we first introduce the numerical problems used in both the optimisation and testing stages.

3. Training and Test Problems

We now describe the numerical problems used to train and test the methods, namely the (2+1)-dimensional nonlinear Schrödinger equation, the Two-Body problem and the N-Body problem.

3.1. (2+1)-Dimensional Nonlinear Schrödinger (NLS) Equation

We consider the (2+1)-dimensional nonlinear Schrödinger (NLS) equation of the form:

$$\begin{aligned} &iu_t + au_{xx} + bu_{yy} + c|u|^2u + du = 0 \\ &u(x, y, t = 0) = u_0(x, y) \end{aligned} \tag{10}$$

where $i = \sqrt{-1}$, $a, b, c, d \in \mathbb{R}$, $u(x, y, t) : \mathbb{R}^3 \to \mathbb{C}$ is a complex function of the spatial variables x, y and the temporal variable t, and $u_0(x, y) : \mathbb{R}^2 \to \mathbb{C}$. The term iu_t governs temporal evolution, the terms au_{xx} and bu_{yy} describe spatial dispersion in the x and y directions, respectively, and the nonlinear term $c|u|^2u$ arises in physical applications such as Bose–Einstein condensates and optical beam propagation [31].

Equation (10) has a dark soliton solution given by

$$u(x, y, t) = \sqrt{-\frac{\gamma}{c}} \tanh\left(\sqrt{\frac{\gamma}{2\beta}}\left(k_1x + l_1y - 2at + \xi_0\right)\right) e^{i\left(k_2x + l_2y - c_2t + \eta_0\right)} \tag{11}$$

where $\beta = ak_1^2 + bl_1^2$, $\gamma = c_2 + d - ak_2^2 - bl_2^2$, and we select $a = 1$, $b = 1$, $c = -1$, $d = 1$, $k_1 = 1$, $k_2 = 1$, $l_1 = 0$, $l_2 = 0$, $c_2 = 2$, $\xi_0 = 0$, $\eta_0 = 0$ [32].

The analytical solution of the above special case is

$$u(x, y, t) = \sqrt{2}\tanh(x - 2t)e^{i(x-2t)} \tag{12}$$

and the density is given by

$$\left|u(x, y, t)\right|^2 = 2\left|\tanh(x - 2t)\right|^2$$

Furthermore, the solution $u(x, y, t)$ of Equation (10) satisfies the mass conservation law [33]:

$$M[u](t) = \int_{\mathbb{R}^2} |u(x, y, t)|^2 dxdy \equiv M[u](0) \tag{13}$$

For the numerical solution of Equation (10),

$$\begin{aligned}
&\text{let } (x, y, t) \in [x_L, x_U] \times [y_L, y_U] \times [0, T] \text{ and} \\
&u_{l,m}^n \text{ denotes an approximation of } u(x_l, y_m, t_n), \text{ where} \\
&x_l = x_L + l\,\Delta x,\ l = 0, 1, \ldots, L - 1,\ L = \tfrac{x_U - x_L}{\Delta x} + 1,\ \Delta x > 0 \\
&y_m = y_L + m\,\Delta y,\ m = 0, 1, \ldots, M - 1,\ M = \tfrac{y_U - y_L}{\Delta y} + 1,\ \Delta y > 0 \\
&t_n = n\,h,\ n = 0, 1, \ldots, N - 1,\ N = \tfrac{T}{h} + 1,\ h > 0
\end{aligned} \tag{14}$$

We use the method of lines for $(x, y, t) \in [-20, 30] \times [-1, 1] \times [0, T]$ for the integration of problem (10), where $T = 2.5$ for the training problem, and $T = 5$ for the test problem. Even shorter training intervals could yield good results, but not consistently, and the resulting RK methods may lack sufficient stability characteristics. As we are interested in the RK method's performance, we use Equation (12) to evaluate both the initial and boundary conditions and the second derivatives u_{xx} and u_{yy} of the same equation. The maximum global solution error and the maximum relative mass error of the numerical solution are defined by Equations (15) and (16) respectively.

For the NLS equation, if $u_{l,m}^n$ is the numerical approximation, $u(x_l, y_m, t_n)$, we define the maximum global solution error as

$$\|E\|_\infty = \max_{l,m,n}(|u_{l,m}^n - u(x_l, y_m, t_n)|) \tag{15}$$

and the maximum relative mass error over the region D as

$$\max_n \left(\left| \frac{M_D^n - M_D^0}{M_D^0} \right| \right) \tag{16}$$

where M_D^n is approximating $M[u](t_n)$ over $D = [-20, 30] \times [-1, 1]$ for $t \in [0, T]$, and M_D^0 is approximating $M[u](0)$ over the same region D.

3.2. Two-Body Problem

The classical planar Newtonian Two-Body problem is used as a training problem to tune the four free coefficients before testing it on the N-Body problem.

In Cartesian form:

$$\begin{cases} u_1'' = -\frac{u_1}{r^3}, \;\; u_2'' = -\frac{u_2}{r^3} \\ u_1(0) = 1 - e, \;\; u_1'(0) = 0 \\ u_2(0) = 0, \;\; u_2'(0) = \sqrt{\frac{1+e}{1-e}} \end{cases} \tag{17}$$

where $r = \sqrt{u_1^2 + u_2^2}$ and e is the orbital eccentricity.

The theoretical solution is provided by:

$$\begin{cases} u_1(t) = \cos(\theta) - e \\ u_2(t) = \sqrt{1 - e^2}\, \sin(\theta) \end{cases} \tag{18}$$

where θ can be found by solving the equation $\theta - e\, \sin(\theta) - t = 0$ [34].

As we consider the Two-Body problem a training problem, we select a relatively short interval of integration: $t \in [0, 100]$. We tested the trained methods in the Two-Body problem for $t \in [0, 1000]$ and the N-Body problem, as described in the next section.

3.3. N-Body Problem

The N-Body problem is the problem that investigates the movement of N bodies under Newton's law of motion. It is expressed by the system

$$u_i'' = G \sum_{j=1, j\neq 0}^{N} \frac{m_j\, (\overrightarrow{u_j} - \overrightarrow{u_i})}{|\overrightarrow{u_j} - \overrightarrow{u_i}|^3}, \quad i = 1, 2, \ldots, N \tag{19}$$

where G is the gravitational constant, m_j is the mass of body j, and $\overrightarrow{u_i}$ and $\overrightarrow{u''}_i$ denote the vectors of the position and acceleration of body i, respectively.

We solve the five outer body problem, a specific configuration of the N-Body problem. This system consists of the Sun and the four inner planets (denoted as Sun+4), the four outermost planets, and Pluto. Table 1 shows the masses (relative to the solar mass), the orbital eccentricities and revolution periods (in Earth years), and the initial position and velocity components of the six bodies. The distances are in astronomical units, and the gravitational constant is $G = 2.95912208286 \cdot 10^{-4}$.

As analytical solutions are unavailable, we compare against reference high-precision numerical results, produced by applying the 10-stage implicit Runge–Kutta method of Gauss with 20th algebraic order, for $t \in [0, 10^6]$ (about 2738 years), following the procedure described in [35].

Table 1 implies that the dominant eccentricity of the bodies with the highest frequencies, or lowest revolution periods, that is, Jupiter, Saturn, and Uranus, is $e \approx 0.05$. This observation motivates our choice of training eccentricity, as further explained in Section 6. Additional methodologies for investigating the performance and behaviour of the methods in astronomical problems can be found in [36].

Table 1. Five outer body problems: The basic properties of the bodies, their orbits, and their initial position and velocity, where Sun+4 denotes the Sun and the four inner planets.

Body	Mass (Solar Masses)	Orbital Ecc.	Rev. Period (Years)	Init. Position	Init. Velocity
Sun+4	1.00000597682	-	-	0 0 0	0 0 0
Jupiter	0.000954786104043	0.0484	11.86	−3.5023653 −3.8169847 −1.5507963	0.00565429 −0.00412490 −0.00190589
Saturn	0.000285583733151	0.0541	29.46	9.0755314 −3.0458353 −1.6483708	0.00168318 0.00483525 0.00192462
Uranus	0.0000437273164546	0.0472	84.01	8.3101420 −16.2901086 −7.2521278	0.00354178 0.00137102 0.00055029
Neptune	0.0000517759138449	0.0086	164.79	11.4707666 −25.7294829 −10.8169456	0.00288930 0.00114527 0.00039677
Pluto	$1/(1.3 \cdot 10^8)$	0.2488	248.59	−15.5387357 −25.2225594 −3.1902382	0.00276725 −0.00170702 −0.00136504

4. Optimal RK Coefficients

Table 2 lists the optimal coefficient values obtained for each training problem. These include the nonlinear Schrödinger equation and the Two-Body problem with eccentricities $e = 0.00$, 0.05, and 0.25 for both bounded ($c_i \in [0, 1]$) and unbounded cases. For comparison, the Original method from [10] is also included.

In total, seven new optimised RK methods were constructed:

- NLS (unbounded) and NLS in [0, 1] (bounded),
- TB I, TB II, TB III (unbounded, trained for $e = 0.00$, 0.05, and 0.25),
- TB Ib, TB IIb (bounded counterparts of TB I and TB II).

Notably, the coefficients of the bounded NLS method are relatively close to those of the Original method, while the unbounded version differs significantly. The unbounded Two-Body methods (TB I and TB II) also show substantial deviations from their bounded counterparts (TB Ib and TB IIb). In contrast, TB III, trained on a problem with high eccentricity $e = 0.25$, naturally satisfies $c_i \in [0, 1]$ and therefore has no bounded variant.

These variations in coefficient values reflect the complex, nonlinear relationship between the free parameters and the global behaviour of the method. They also illustrate the influence of the training problem, particularly the solution's eccentricity and oscillatory character, on the structure of the optimal coefficients.

Table 2. Coefficients of developed methods. Entries where $c_i \notin [0,1]$ are bolded.

Method	c_2	c_4	c_5	c_7
Original [10]	0.0070000000000000	0.2080000000000000	0.4410000000000000	0.9150000000000000
NLS	0.1846192563425903	0.2247603817048323	0.5706075347944812	**1.0582458651706454**
NLS in $[0,1]$	0.0092477488555108	0.2166123597181641	0.5440778163831921	0.9805511474091542
TB I ($e=0$)	**−0.0094525465942426**	**−0.1057207949231992**	0.4183913927663807	**1.5601765389027689**
TB Ib in $[0,1]$	0.1366360718492776	0.0408398027965382	0.1818797871926411	0.5520864798690709
TB II ($e=0.05$)	0.0157893424540159	**−0.1580319415439770**	0.3356276745332045	0.8364470804118889
TB IIb in $[0,1]$	0.0010067780917901	0.2592782871426017	0.4231941925978401	0.6844694952540733
TB III ($e=0.25$)	0.0955480567625531	0.3901353596573214	0.1789726462498607	0.2883289058072621

5. Error and Stability Analysis

In this section, we present the local truncation error and stability analysis.

5.1. Error Analysis

The local truncation error analysis of the parametric method defined in Equation (2) and Section 2.2, based on the Taylor expansion series of the difference $\tau_{n+1} = u^{n+1} - u(t_{n+1})$, yields the principal term of the local truncation error, which is given by

$$\tau_{n+1} = p_7 \frac{d^7 u}{dt^7} h^7 + \mathcal{O}(h^8) \tag{20}$$

where

$$\begin{aligned} p_7 = &-\Big(64\Big(30c_4^3 - 35c_4^2 + 11c_4 - 1\Big)\Big)^{-1} \\ &\Big((10c_4c_5c_7 - 5c_4c_5 - 5c_4c_7 - 5c_5c_7 + 3c_4 + 3c_5 + 3c_7 - 2)\Big)^{-1} \\ &25\Bigg(\Bigg(\Big(c_7 - \tfrac{3}{5}\Big)c_5^2 + \Big(-\tfrac{151c_7}{210} + \tfrac{53}{126}\Big)c_5 + \tfrac{218c_7}{1575} - \tfrac{218}{2625}\Bigg)c_4^4 \\ &+\Bigg(\Big(-\tfrac{37c_7}{30} + \tfrac{23}{30}\Big)c_5^2 + \Big(\tfrac{2977c_7}{3150} - \tfrac{26933}{47250}\Big)c_5 + \tfrac{5801}{47250} - \tfrac{3107c_7}{15750}\Bigg)c_4^3 \\ &+\Bigg(\Big(\tfrac{124c_7}{225} - \tfrac{397}{1125}\Big)c_5^2 + \Big(-\tfrac{7019c_7}{15750} + \tfrac{1451}{5250}\Big)c_5 + \tfrac{527c_7}{5250} - \tfrac{3053}{47250}\Bigg)c_4^2 \\ &+\Bigg(\Big(-\tfrac{118c_7}{1125} + \tfrac{232}{3375}\Big)c_5^2 + \Big(\tfrac{694c_7}{7875} - \tfrac{439}{7875}\Big)c_5 + \tfrac{328}{23625} - \tfrac{166c_7}{7875}\Bigg)c_4 \\ &+\Big(\tfrac{8c_7}{1125} - \tfrac{16}{3375}\Big)c_5^2 + \Big(\tfrac{92}{23625} - \tfrac{16c_7}{2625}\Big)c_5 + \tfrac{4c_7}{2625} - \tfrac{8}{7875}\Bigg) \end{aligned} \tag{21}$$

indicating global accuracy of order six. The p_7 values for the newly developed methods are presented in Table 3.

5.2. Stability Analysis

For the analysis of the (absolute) stability of method (2), we consider the problem

$$u' = \lambda u, \quad \lambda \in \mathbb{C} \tag{22}$$

Applying method (2) to the above equation, we obtain the numerical solution $u_{n+1} = R(v)u_n$, where $v = \lambda h$, and $R(v)$ is called the stability polynomial [2]. We have the following definitions:

Definition 1. *For the method of Equation* (2)*, we define the stability interval along the real axis as* $I_R = [v_R, 0]$*, where*

$$v_R = \max\{-\delta : \delta \geqslant 0,\ |R(v)| \leqslant 1,\ \text{for all}\ v \in [-\delta, 0]\ \text{with}\ \mathrm{Im}(v) = 0\}$$

Definition 2. *For the method of Equation (2), we define the stability interval along the imaginary axis as* $I_I = [-v_I, v_I]$*, where*

$$v_I = \max\{\delta \geqslant 0 : \ |R(v)| \leqslant 1, \text{ for all } v \in [-\delta, \delta] \text{ with } \mathrm{Re}(v) = 0\}$$

Definition 3. *The region of absolute stability is defined as the set* $S = \{v \in \mathbb{C} : |R(v)| \leqslant 1\}$

Following the above procedure, we compute its stability polynomial, which is given by

$$\begin{aligned} R = & \Big(34560(10c_4c_5c_7 - 5c_4c_5 - 5c_4c_7 - 5c_5c_7 + 3c_4 + 3c_5 + 3c_7 - 2)(6c_4 - 1)\Big(5c_4^2 - 5c_4 + 1\Big)\Big)^{-1} \\ & \Big(10368000Q + 10368000Qv + 5184000Qv^2 + 1728000Qv^3 + 432000Qv^4 + 86400Qv^5 \\ & +14400Qv^6 + \Big(\Big((13500c_7 - 8100)c_5^2 + (-7650c_7 + 4650)c_5 + 840c_7 - 504\Big)c_4^4 \\ & +\Big((-16650c_7 + 10350)c_5^2 + (9330c_7 - 5878)c_5 - 846c_7 + 526\Big)c_4^3 \\ & +\Big((7440c_7 - 4764)c_5^2 + (-4062c_7 + 2634)c_5 + 258c_7 - 166\Big)c_4^2 \\ & +\Big((-1416c_7 + 928)c_5^2 + (744c_7 - 492)c_5 - 24c_7 + 16\Big)c_4 + 96c_5\Big(c_5 - \tfrac{1}{2}\Big)\Big(c_7 - \tfrac{2}{3}\Big)\Big)v^7 \\ & +1125\Big(\Big(-\tfrac{2}{5} + c_5\Big)c_4 - \tfrac{2c_5}{5} + \tfrac{1}{5}\Big)\Big(\Big(c_7 - \tfrac{3}{5}\Big)c_4 - \tfrac{2c_7}{5} + \tfrac{4}{15}\Big)c_4\Big(\Big(c_5 - \tfrac{4}{15}\Big)c_4 - \tfrac{4c_5}{15} + \tfrac{1}{15}\Big)v^8\Big) \end{aligned}$$

where

$$Q = \Big(\Big(\Big(c_7 - \tfrac{1}{2}\Big)c_5 - \tfrac{c_7}{2} + \tfrac{3}{10}\Big)c_4 + \Big(-\tfrac{c_7}{2} + \tfrac{3}{10}\Big)c_5 + \tfrac{3c_7}{10} - \tfrac{1}{5}\Big)\Big(-\tfrac{1}{6} + c_4\Big)\Big(c_4^2 - c_4 + \tfrac{1}{5}\Big)$$

Using this polynomial and the coefficients of all methods in Table 2, we conduct a numerical computation of the real and imaginary stability intervals, according to Definitions 1 and 2, respectively, and a step length of $dv = 0.0001$, presenting the intervals in Table 3. Similarly, we present the stability regions of all methods in Figure 1, as the set of grid points that satisfy Definition 3. More specifically, we evaluated $v = x + iy$ over the domain $x \in [-6.5, 0.5]$, $y \in [-5, 5]$, using a step length of 0.01. Here, x and y represent the real and imaginary parts of v, respectively. For clarity, only the boundary of each stability region is shown, though each region comprises all points for which $|R(v)| \leqslant 1$.

Table 3. Properties of developed methods. Here, v_R and v_I define the stability interval along the real axis $[v_R, 0]$ and along the imaginary axis $[-v_I, v_I]$, whereas p_7 stands for the coefficient of the principal term of the local truncation error, shown in (20). Figure 1 provides a graphical representation of the stability intervals.

Method	v_R	v_I	p_7
Original [10]	−3.986	0.001	3.4×10^{-5}
NLS	−4.442	0.002	4.4×10^{-6}
NLS in $[0, 1]$	−4.057	0.001	1.1×10^{-5}
TB I ($e = 0$)	−5.926	0.001	7.4×10^{-5}
TB Ib ($e = 0$) in $[0, 1]$	−3.912	0.000	1.3×10^{-4}
TB II ($e = 0.05$)	−3.709	0.311	-5.3×10^{-5}
TB IIb ($e = 0.05$) in $[0, 1]$	−3.807	0.000	1.4×10^{-5}
TB III ($e = 0.25$)	−4.180	0.001	1.2×10^{-4}

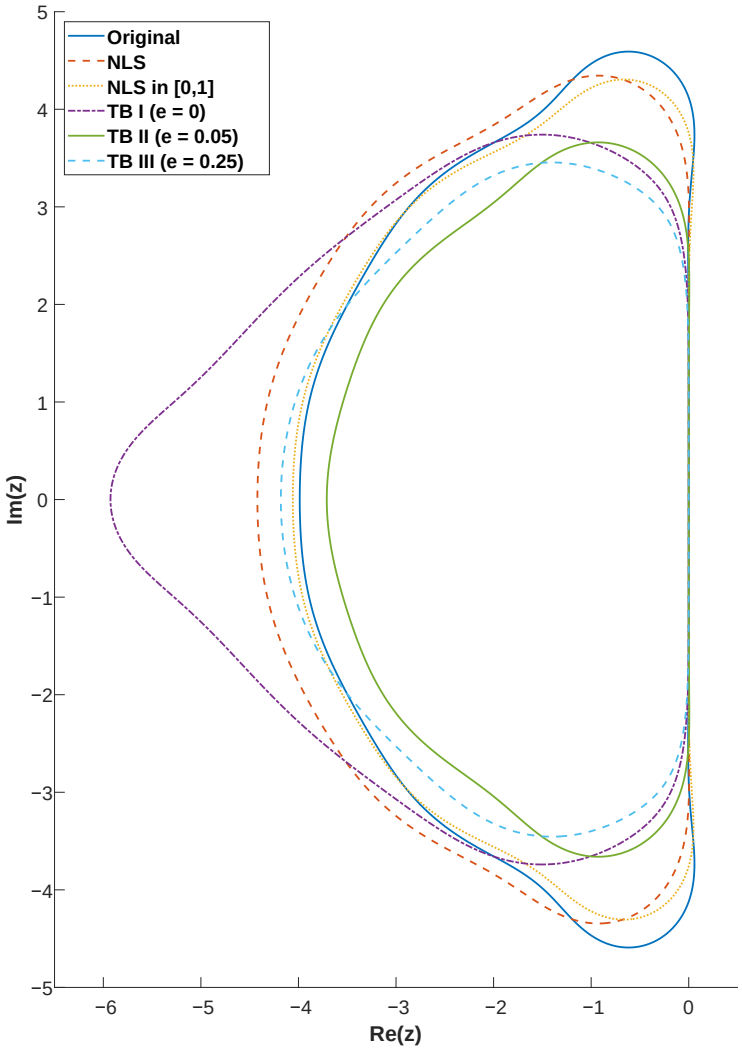


Figure 1. Boundaries of the stability regions of the methods presented in Table 2, where each region consists of the interior and boundary points satisfying Definition 3. The stability interval along the real axis $[v_R, 0]$ corresponds to the intersection of the stability region and the negative real semi-axis, whereas the stability interval along the imaginary axis $[-v_I, v_I]$ corresponds to the intersection of the stability region and the imaginary axis. The numerical values of v_R and v_I are reported in Table 3. Further details are provided in Section 5.2.

5.3. Error and Stability Characteristics and Interpretation

In Table 3, we note that the new methods show improved stability and local truncation error characteristics. For example, the new method NLS, compared with the Original method, exhibits larger real and imaginary stability intervals (indicated by larger absolute values of v_R and v_I, respectively) and a smaller absolute coefficient for the principal term of the local truncation error (denoted by p_7). A similar improvement is evident when comparing the unbounded TB I with its bounded counterpart, TB Ib (developed under the condition $c_i \in [0, 1]$). The unbounded method exhibits a real stability interval approximately two units larger and a truncation error coefficient approximately half an order of magnitude smaller. Furthermore, TB II has a non-vanishing imaginary stability interval in contrast to the bounded counterpart TB IIb, while retaining a similar real stability interval and error coefficient. We observe that, even though the methods were developed for optimal global error, they also exhibit good, albeit not optimal, local properties. Whether the local properties are optimal depends solely on whether they contribute to the minimisation of the global error.

6. Numerical Results

6.1. Overview of Compared Methods

We measure the efficiency of the new methods against existing methods from the literature. We refer to methods 2–8 as new methods, meaning Runge–Kutta methods constructed via PSO in this study. Methods 3, 5, and 7 have bounded nodes such that $c_i \in [0,1]$, $i = 1,2,\ldots,8$, whereas methods 2, 4, 6, and 8 were optimised without this constraint, even though method 8 ultimately satisfies it.

The compared methods are presented below:

1. Original: Method developed in [10] for the NLS equation (Table 2, row 1)
2. NLS: New method optimised by PSO for the NLS equation, without boundary constraints (Table 2, row 2)
3. NLS in $[0,1]$: New method optimised for the NLS equation with bounded nodes (Table 2, row 3)
4. TB I: New method optimised for the Two-Body problem with $e = 0$, unbounded (Table 2, row 4)
5. TB Ib: New method optimised for the Two-Body problem with $e = 0$ with bounded nodes (Table 2, row 5)
6. TB II: New method optimised for the Two-Body problem ($e = 0.05$) and the N-Body problem, unbounded (Table 2, row 6)
7. TB IIb: New method optimised for the Two-Body problem ($e = 0.05$) and the N-Body problem with bounded nodes (Table 2, row 7)
8. TB III: New method optimised for the Two-Body problem with $e = 0.25$, unbounded (Table 2, row 8)
9. Papageorgiou: The method of Papageorgiou et al., with phase-lag and amplification error orders of 10 and 11, respectively [37]
10. Kosti I: The method of Kosti et al. (I), with phase-lag and amplification error orders of 8 and 9, respectively [38]
11. Dormand: The classic method of Dormand et al. [2]
12. Kosti II: The method of Kosti et al. (II), with phase-lag and amplification error orders of 8 and 5, respectively [39]
13. Fehlberg: The classic method of Fehlberg et al. [2]
14. Triantafyllidis: The method of Triantafyllidis et al., with phase-lag and amplification error orders of 10 and 7, respectively [40]
15. Kovalnogov: The parametric method of Kovalnogov et al., optimised with differential evolution [19]
16. Jerbi: The parametric method of Jerbi et al., optimised with differential evolution [20]

6.2. Performance on the NLS Equation

Table 4 presents the maximum global solution error and the maximum relative mass error of the selected methods when solving the NLS Equation (10) for $\Delta x = \Delta y = 0.1$ and six values of Δt ranging from $\frac{1}{80}$ to $\frac{2}{5}$. We exclude methods developed specifically for the Two-Body and N-Body problems. The table indicates the superior performance of the PSO-optimised NLS method compared with all other methods, including the Original method developed specifically for the NLS equation, for both small and large step lengths. At the largest step length $\Delta t = 2/5$, all other methods either diverged or produced a maximum global error exceeding one, indicating instability in contrast to the NLS method, which remained stable and accurate.

Table 4. Maximum global solution error and maximum relative mass error of the selected methods in the NLS Equation (10) for $\Delta x = \Delta y = 0.1$ and six different Δt values.

Method	Solution Error	Mass Error	Solution Error	Mass Error
	$\Delta t = 1/80$		$\Delta t = 1/40$	
NLS	3.48×10^{-11}	6.46×10^{-12}	1.89×10^{-9}	7.38×10^{-10}
NLS in $[0, 1]$	1.15×10^{-10}	5.71×10^{-11}	1.97×10^{-9}	1.07×10^{-9}
Original [10]	7.39×10^{-11}	3.73×10^{-11}	5.64×10^{-9}	3.14×10^{-9}
Papageorgiou [37]	2.39×10^{-10}	8.09×10^{-11}	1.63×10^{-8}	5.21×10^{-9}
Kosti I [38]	5.43×10^{-10}	1.37×10^{-10}	3.54×10^{-8}	8.27×10^{-9}
Dormand [2]	2.74×10^{-8}	1.21×10^{-8}	2.07×10^{-6}	1.14×10^{-6}
Kosti II [39]	7.28×10^{-8}	3.83×10^{-8}	2.23×10^{-6}	1.17×10^{-6}
Fehlberg [2]	1.11×10^{-8}	5.82×10^{-9}	6.42×10^{-7}	3.28×10^{-7}
Triantafyllidis [40]	4.69×10^{-8}	2.32×10^{-8}	2.71×10^{-6}	1.31×10^{-6}
Kovalnogov [19]	3.90×10^{-10}	1.05×10^{-10}	2.69×10^{-8}	6.84×10^{-9}
Jerbi [20]	6.25×10^{-10}	1.89×10^{-10}	4.22×10^{-8}	1.20×10^{-8}
	$\Delta t = 1/20$		$\Delta t = 1/10$	
NLS	2.09×10^{-7}	5.16×10^{-8}	2.37×10^{-5}	4.20×10^{-6}
NLS in $[0, 1]$	2.24×10^{-7}	8.15×10^{-8}	2.62×10^{-5}	5.33×10^{-6}
Original [10]	4.78×10^{-7}	1.99×10^{-7}	4.51×10^{-5}	1.20×10^{-5}
Papageorgiou [37]	1.22×10^{-6}	3.17×10^{-7}	9.92×10^{-5}	1.81×10^{-5}
Kosti I [38]	2.42×10^{-6}	4.92×10^{-7}	1.74×10^{-4}	2.72×10^{-5}
Dormand [2]	1.27×10^{-4}	7.03×10^{-5}	5.23×10^{-3}	2.88×10^{-3}
Kosti II [39]	6.55×10^{-5}	3.44×10^{-5}	1.78×10^{-3}	9.25×10^{-4}
Fehlberg [2]	3.28×10^{-5}	1.56×10^{-5}	1.23×10^{-3}	3.89×10^{-4}
Triantafyllidis [40]	1.39×10^{-4}	6.20×10^{-5}	5.04×10^{-3}	1.37×10^{-3}
Kovalnogov [19]	2.09×10^{-6}	4.14×10^{-7}	1.81×10^{-4}	2.35×10^{-5}
Jerbi [20]	3.10×10^{-6}	7.33×10^{-7}	2.89×10^{-4}	4.24×10^{-5}
	$\Delta t = 1/5$		$\Delta t = 2/5$	
NLS	1.54×10^{-3}	4.87×10^{-4}	6.83×10^{-1}	8.00×10^{-2}
NLS in $[0, 1]$	1.35×10^{-3}	2.78×10^{-4}	-	-
Original [10]	3.21×10^{-3}	5.49×10^{-4}	-	-
Papageorgiou [37]	9.07×10^{-3}	8.96×10^{-4}	-	-
Kosti I [38]	1.36×10^{-2}	1.38×10^{-3}	-	-
Dormand [2]	9.44×10^{-2}	2.90×10^{-2}	-	-
Kosti II [39]	6.19×10^{-2}	2.06×10^{-2}	-	-
Fehlberg [2]	5.09×10^{-2}	2.72×10^{-2}	-	-
Triantafyllidis [40]	3.65×10^{-1}	1.97×10^{-1}	-	-
Kovalnogov [19]	1.70×10^{-2}	1.13×10^{-3}	-	-
Jerbi [20]	2.89×10^{-2}	2.18×10^{-3}	-	-

As shown in Table 4, the NLS method performs best overall. Figure 2 illustrates this by plotting the maximum global solution error against the function evaluations for the best seven methods solving the NLS Equation (10), using the setup described in Section 3.1. Both newly constructed methods, NLS and NLS in [0, 1], outperform the Original method across all tested step lengths. The unbounded NLS method also exhibits superior stability, achieving better accuracy at larger step lengths and faster convergence in that regime compared with the bounded variant.

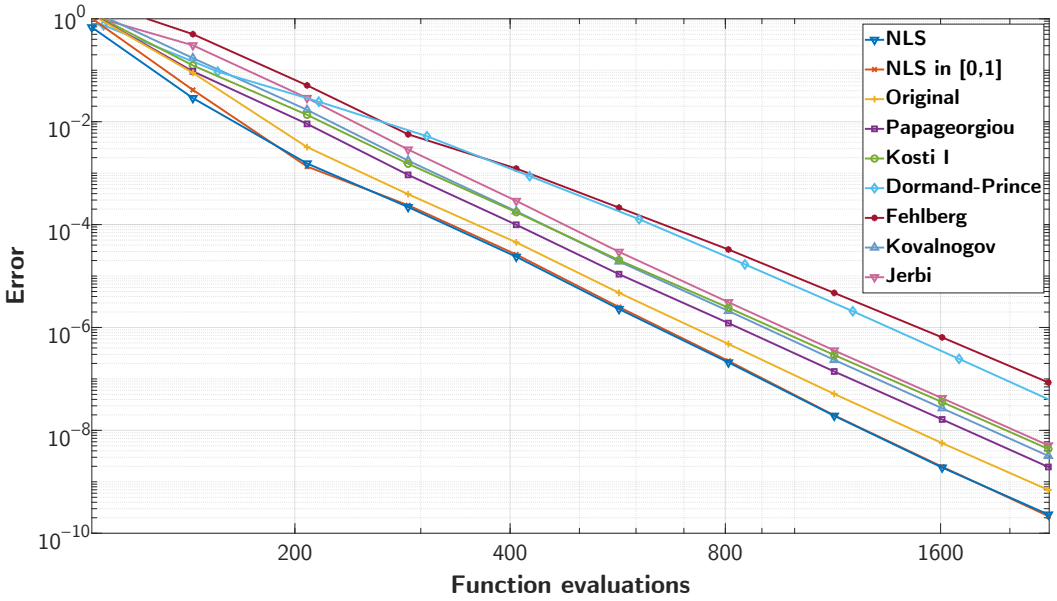


Figure 2. Global error vs. function evaluations for the most efficient methods solving the NLS Equation (10). The NLS-trained methods display a clear advantage in efficiency compared with the Original and other methods, as their curves lie closer to the bottom left of the plot. See Section 6.2 for further details.

6.3. Performance on the Two-Body Problem

Although the Two-Body problem was used for training before testing on the N-Body problem, we also evaluate the methods on the former problem over a longer integration interval, as described in Section 3.2.

Figures 3–5 show the efficiency results for the Two-Body problem (17) for eccentricities $e = 0$, $e = 0.05$, and $e = 0.25$, respectively. In each case, we present the three best TB methods, i.e., the ones without the constraint $c_i \in [0, 1]$, along with the corresponding bounded variant trained for that eccentricity, where available. For example, in Figure 5, there is no such method, as the coefficients of the best method for $e = 0.25$, TB III, already satisfy the constraint, as seen in the last row of Table 2.

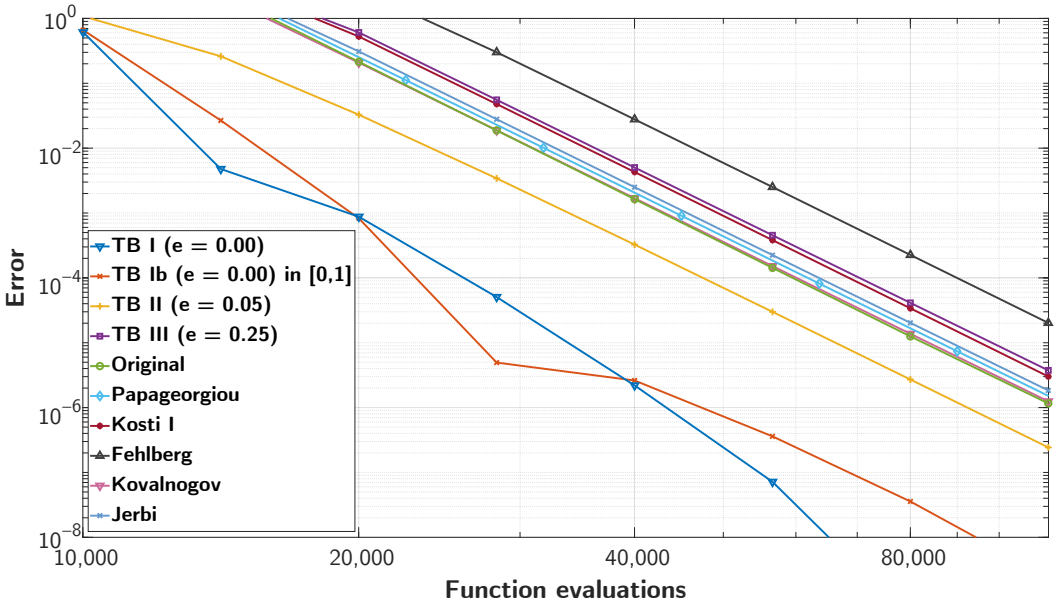


Figure 3. Global error vs. function evaluations for the most efficient methods solving the Two-Body problem (17) with eccentricity $e = 0$. The TB I and TB Ib methods, trained specifically for this eccentricity, show clear improvements in efficiency. See Section 6.3 for details.

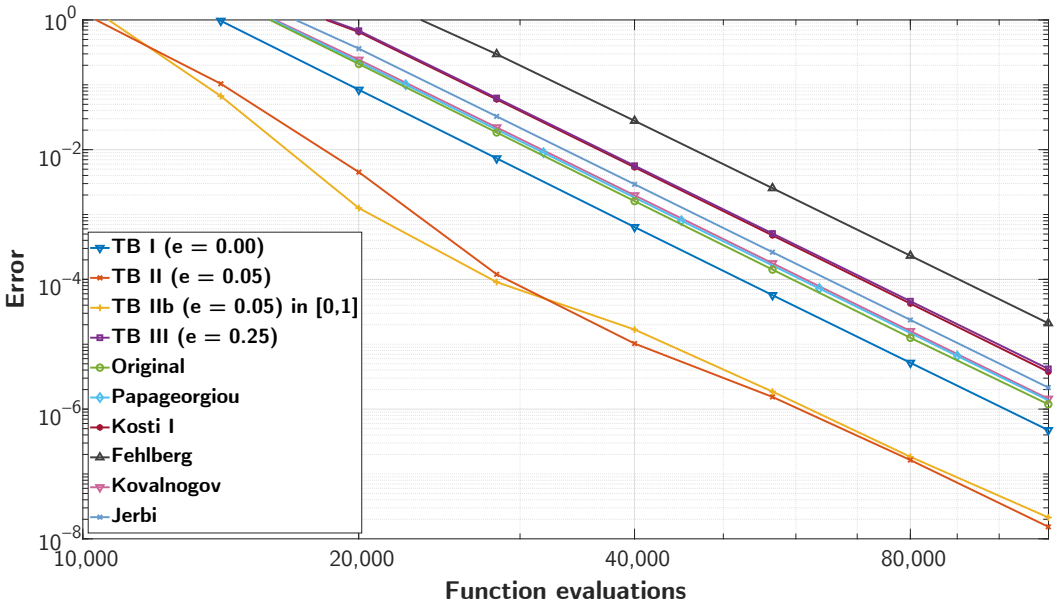


Figure 4. Global error vs. function evaluations for the most efficient methods solving the Two-Body problem (17) with eccentricity $e = 0.05$. The TB II and TB IIb methods, trained on this eccentricity, outperform all others, including those trained for different eccentricities. See Section 6.3 for details.

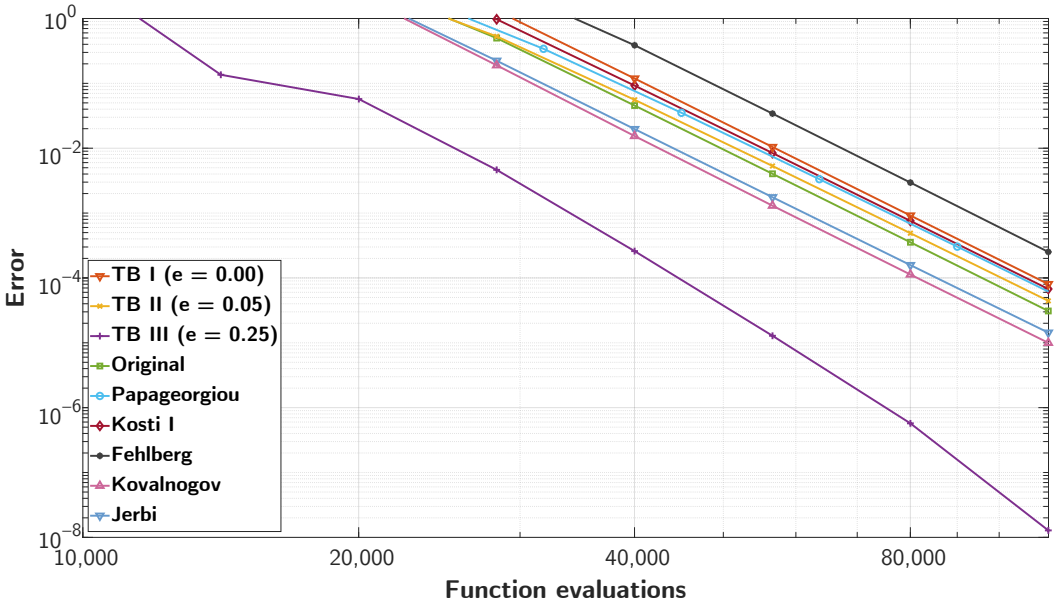


Figure 5. Global error vs. function evaluations for the most efficient methods solving the Two-Body problem (17) with eccentricity $e = 0.25$. The best-performing method, TB III, satisfies the boundary constraint and outperforms all other methods by up to three orders of magnitude. See Section 6.3 for details.

The plots illustrate that the most efficient methods achieve lower global error with fewer function evaluations across varying values of e. For smaller e, the problem is less stiff, allowing certain methods to maintain accuracy with relatively low computational cost. As e increases, reflecting greater oscillatory behaviour and complexity, methods trained specifically for those eccentricities require fewer function evaluations compared to the other methods to achieve accuracy, thus becoming more efficient. This variation highlights that method efficiency depends on the problem regime defined by e and the corresponding targeted training of the methods. In conclusion, the methods trained on the corresponding

eccentricity outperform all others, including those trained for different eccentricities, by up to three orders of magnitude in global error.

6.4. Performance on the N-Body Problem

To assess generalisability to more complex systems, we test the methods trained for the Two-Body problem on the N-Body problem. Table 5 presents the maximum global error for each method applied to the N-Body problem (19) for eight different step lengths in a geometric sequence. We include all methods except the two methods developed for the NLS equation. As observed in Section 3.3, Table 1, the dominant eccentricity of the bodies with the highest frequencies, which are the main contributors to the maximum global error [34], is approximately $e \approx 0.05$. Indeed, the results show that the best performance in the five outer planet problem is achieved by using the method whose coefficients have been trained for the simpler Two-Body problem with $e = 0.05$. Methods produced by training the coefficients in the Two-Body problem with orbital eccentricities $e = 0$ or $e = 0.25$ yielded much poorer results.

Table 5. Maximum global solution error of the selected methods in the N-Body problem (19) for eight different step lengths.

Method\Step Length	$\mathbf{250\sqrt{2}}$	**250**	$\mathbf{125\sqrt{2}}$	**125**	$\mathbf{62.5\sqrt{2}}$	**62.5**	$\mathbf{31.25\sqrt{2}}$	**31.25**
TB II ($e = 0.05$)	1.63×10^{-1}	4.63×10^{-3}	1.74×10^{-4}	4.60×10^{-5}	5.78×10^{-6}	6.23×10^{-7}	6.18×10^{-8}	3.93×10^{-9}
TB IIb ($e = 0.05$) in $[0, 1]$	1.06×10^{-1}	7.04×10^{-4}	3.22×10^{-4}	4.55×10^{-5}	4.81×10^{-6}	4.86×10^{-7}	6.47×10^{-8}	2.06×10^{-8}
TB I ($e = 0$)	2.11	2.00×10^{-1}	1.78×10^{-2}	1.59×10^{-3}	1.43×10^{-4}	1.30×10^{-5}	1.19×10^{-6}	1.11×10^{-7}
TB Ib ($e = 0$) in $[0, 1]$	3.18	3.50×10^{-1}	3.19×10^{-2}	2.90×10^{-3}	2.63×10^{-4}	2.42×10^{-5}	2.24×10^{-6}	2.13×10^{-7}
TB III ($e = 0.25$)	8.17	1.49	1.42×10^{-1}	1.29×10^{-2}	1.16×10^{-3}	1.04×10^{-4}	9.42×10^{-6}	8.10×10^{-7}
Original [10]	5.00	4.79×10^{-1}	4.10×10^{-2}	3.55×10^{-3}	3.08×10^{-4}	2.71×10^{-5}	2.65×10^{-6}	4.57×10^{-7}
Papageorgiou [37]	2.33	2.28×10^{-1}	2.02×10^{-2}	1.79×10^{-3}	1.61×10^{-4}	1.45×10^{-5}	1.31×10^{-6}	1.20×10^{-7}
Kosti I [38]	8.84	1.43	1.36×10^{-1}	1.22×10^{-2}	1.09×10^{-3}	9.67×10^{-5}	8.59×10^{-6}	7.63×10^{-7}
Dormand [2]	-	-	-	-	6.25×10^{-2}	1.87×10^{-2}	4.00×10^{-3}	7.67×10^{-4}
Kosti II [39]	-	-	9.24×10^{-1}	1.67	3.10×10^{-1}	5.49×10^{-2}	9.70×10^{-3}	1.71×10^{-3}
Fehlberg [2]	-	5.63	6.82×10^{-1}	6.21×10^{-2}	5.59×10^{-3}	4.99×10^{-4}	4.43×10^{-5}	3.93×10^{-6}
Triantafyllidis [40]	-	-	3.81	4.51×10^{-1}	4.12×10^{-2}	3.71×10^{-3}	3.34×10^{-4}	3.02×10^{-5}
Kovalnogov [19]	5.22	5.52×10^{-1}	5.02×10^{-2}	4.49×10^{-3}	4.03×10^{-4}	3.63×10^{-5}	3.28×10^{-6}	2.99×10^{-7}
Jerbi [20]	8.37	8.05×10^{-1}	7.38×10^{-2}	6.61×10^{-3}	5.91×10^{-4}	5.31×10^{-5}	4.79×10^{-6}	4.35×10^{-7}

A desirable property of a numerical method is to maintain its theoretical order of accuracy over a long range of step lengths that do not necessarily include the unstable region (very large step lengths) and the region where the round-off error becomes significant (very small step lengths).

We estimate the experimental order of accuracy $\hat{p}$ using

$$\hat{p} = \frac{\log\left(\frac{\|E_i\|_\infty}{\|E_{i+1}\|_\infty}\right)}{\log\left(\frac{h_i}{h_{i+1}}\right)} \tag{23}$$

Table 6 shows that all new methods exhibit consistent convergence behaviour and retain the order on average. While some fluctuations exist, a drop in error is typically followed by an increase, so the achieved accuracy often exceeds the values predicted by the average order, which is also confirmed in Figure 6.

Table 6. Experimental order of accuracy computed from successive errors and step lengths for the N-Body problem (19), according to Equation (23) and $h_i = 250 \cdot 2^{\frac{2-i}{2}}$, $i = 1, \ldots, 8$.

Method\Step Length	h_1	h_2	h_3	h_4	h_5	h_6	h_7	h_8
TB II ($e = 0.05$)	10.3	9.5	3.8	6.0	6.4	6.7	8.0	
TB IIb ($e = 0.05$) in $[0, 1]$	14.5	2.3	5.6	6.5	6.6	5.8	3.3	
TB I ($e = 0$)	6.8	7.0	7.0	7.0	6.9	6.9	6.9	
TB Ib ($e = 0$) in $[0, 1]$	6.4	6.9	6.9	6.9	6.9	6.9	6.8	
TB III ($e = 0.25$)	4.9	6.8	6.9	7.0	6.9	6.9	7.1	
Original [10]	6.8	7.1	7.1	7.1	7.0	6.7	5.1	
Papageorgiou [37]	6.7	7.0	7.0	7.0	7.0	6.9	6.9	
Kosti I [38]	5.3	6.8	7.0	7.0	7.0	7.0	7.0	
Dormand [2]	-	-	-	-	3.5	4.5	4.8	
Kosti II [39]	-	-	5.0	4.9	5.0	5.0	5.0	
Fehlberg [2]	-	6.1	6.9	7.0	7.0	7.0	7.0	
Triantafyllidis [40]	-	-	6.2	6.9	7.0	7.0	6.9	
Kovalnogov [19]	6.5	6.9	7.0	7.0	6.9	6.9	6.9	
Jerbi [20]	6.8	6.9	7.0	7.0	7.0	6.9	6.9	

As seen in Figure 6, the slope of each method, representing its experimental order of accuracy, agrees with the results presented in Table 6 for the N-Body problem. Notably, the behaviour of all methods for the N-Body problem (19) closely mirrors that of the Two-Body problem for eccentricity $e = 0.05$. Even though TB II and TB II in $[0, 1]$ were trained on a much simpler problem over a significantly shorter integration interval, they retain their performance characteristics when applied to the more complex system. It is noteworthy that the training TB problem (17) has about 16 full oscillations, whereas for the N-Body problem (19), the fastest revolving body, Jupiter, completes about 230 revolutions.

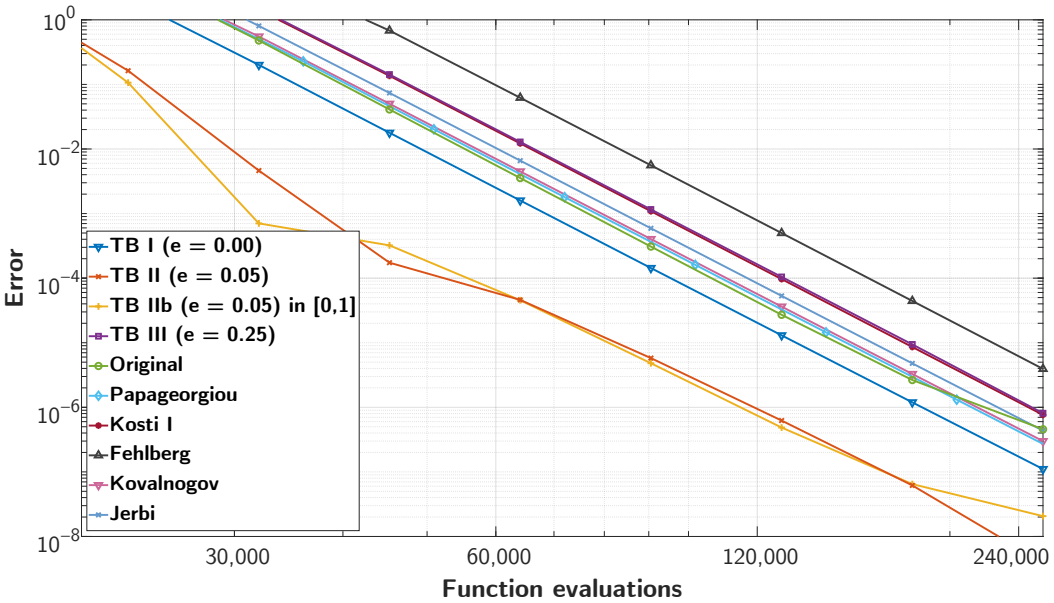


Figure 6. Global error vs. function evaluations for the most efficient methods solving the N-Body problem (19). The experimental order of each method, reflected in the slope of its curve, agrees with the results in Table 6. The performance closely resembles that observed for the Two-Body problem with $e = 0.05$, despite the increased complexity and longer integration interval of the N-Body problem. See Section 6.4 for details.

7. Discussion

There are several aspects that contributed to the increased performance of the evolutionarily optimised methods compared with methods optimised by classical techniques or

those designed to possess local properties. First of all, the objective function under minimisation considers the efficiency of the new method compared with a benchmark method, so the objective function is explicitly designed to maximise the efficiency of the new method relative to an established benchmark, embedding performance comparison into the optimisation itself. The objective function also ensures that it covers the efficiency across a range of accuracies, which also covers small accuracies, thus improving the stability characteristics. To reduce optimisation time and enhance results, the training was performed on simpler versions of the problems, e.g., using short integration intervals or alternative but related problems. Moreover, other integration parameters, such as the step lengths, were tuned individually for each problem to maximise efficiency. This allowed experimentation with relaxed conditions, such as removing typical RK bounds on the coefficients.

The evolutionary optimisation allowed the new methods to be more efficient than their classically optimised counterparts in both local truncation error coefficients and stability intervals. The newly developed methods outperformed existing literature methods tailored to periodic or oscillatory problems. Notably, this was accomplished without explicitly enforcing local properties such as phase lag or amplification error.

A particularly striking result is the training efficiency, as the Two-Body problem required significantly less computational time and a shorter integration interval compared with the more complex N-Body problem. Comparing the function evaluations and CPU time for different accuracies, we determined that the integration of the training problem required approximately 70 times fewer evaluations and less time than the testing problem. In particular, for the Two-Body and N-Body problems, the evolutionary algorithm successfully optimised methods for high eccentricities, where solutions deviate significantly from sinusoidal form, overcoming the limitations of classical methodologies that rely on local properties such as phase-lag minimisation, amplification-error control, and trigonometric fitting.

The evolutionary optimisation process, using the selected PSO parameters, is sufficiently robust to require minimal adjustment when applied to a different problem, aside from possible changes to the step length range.

Nevertheless, some limitations of the PSO-based approach remain. The computational cost is dominated by evaluating the global error through time integration, which may increase substantially when scaling to higher-order methods with more free parameters. Additionally, PSO can be prone to premature convergence in complex or multimodal search landscapes, potentially resulting in suboptimal solutions. To address these issues, hybrid optimisation strategies or adaptive parameter tuning could be considered in future work. Nevertheless, PSO remains an effective and practical choice for the current problem, balancing exploration and exploitation while allowing parallel candidate evaluation.

Ideally, full optimisation of a parametric RK method would be achieved without any predetermined coefficients, and, after solving the system of equations, all coefficients would then depend on a set of free parameters, acting as degrees of freedom. However, this approach remains infeasible with current hardware, due to its high memory and computational demands. Alternative strategies, such as applying simplifying assumptions, can make the process more tractable but introduce additional constraints that limit the extent of optimisation. In practice, it is not yet possible to eliminate the use of predetermined coefficients or simplifying assumptions in the development of parametric RK methods. Nevertheless, reducing their use remains a future goal, as doing so would better highlight the potential of evolutionary optimisation.

However, the methodology has the potential to be applied to a wider range of differential problems. Its direct operation in the parameter space of Runge–Kutta coefficients means that different problem classes could be addressed by selecting appropriate training

problems and parameter constraints. While we have not yet tested these extensions, the general framework is flexible enough to be explored in future work.

8. Conclusions

In this article, we proposed a new optimisation strategy for selecting RK coefficients based on evolutionary computation techniques. This strategy seeks to optimise the global behaviour over a training problem, rather than local properties as targeted by classical approaches. In some cases, the training problem requires nearly two orders of magnitude less computational effort than the testing problem. We also introduced a new objective function designed to improve the method's performance relative to a benchmark. The strategy considers a wide range of accuracies, systematically and efficiently, overcoming challenges such as local minima in the free coefficient space and the lack of derivative information for the objective function. We removed typical coefficient bounds that govern RK methods, such as $c_i \in [0, 1]$, and achieved excellent results in local stability, error characteristics, and global performance. Finally, the methodology uncovers structural patterns in coefficient values when optimising for high eccentricities, where solutions deviate from sinusoidal behaviour, offering insights that can guide coefficient predetermination in classical method design.

The methodology is general and can be applied to any type of problem. Here, we used the NLS equation and the N-Body problem (the latter tested over a long-time integration interval), which are important problems in quantum physics and astronomy, respectively. Future work will focus on applying the methodology to additional problem classes, incorporating a broader set of representative training problems to further enhance the generalisability of the resulting methods.

Funding: This research received no external funding.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Acknowledgments: We thank the anonymous reviewers for their useful comments and remarks.

Conflicts of Interest: The author declares no conflict of interest.

References

1. Hairer, E.; Nørsett, S.P.; Wanner, G. *Solving Ordinary Differential Equations I: Nonstiff Problems*, 2nd ed.; Springer: Berlin/Heidelberg, Germany, 1993. [CrossRef]
2. Butcher, J.C. *Numerical Methods for Ordinary Differential Equations*, 3rd ed.; John Wiley & Sons: Chichester, UK, 2016. [CrossRef]
3. Demba, M.A.; Senu, N.; Ramos, H.; Kumam, P.; Watthayu, W. A Phase- and Amplification-Fitted 5(4) Diagonally Implicit Runge–Kutta–Nyström Pair for Oscillatory Systems. *Bull. Iran. Math. Soc.* **2023**, *49*, 24. [CrossRef]
4. Abdulsalam, A.; Senu, N.; Majid, Z.A.; Nik Long, N.M.A. Higher-Order Multi-Step Runge-Kutta-Nyström Methods with Frequency-Dependent Coefficients for Second-Order Initial Value Problem $u'' = f(x, u, u')$. *Comput. Methods Differ. Equ.* **2024**, *12*, 808–826. [CrossRef]
5. Fang, Y.; Yang, Y.; You, X.; Ma, L. Modified THDRK Methods for the Numerical Integration of the Schrödinger Equation. *Int. J. Mod. Phys. C* **2020**, *31*, 2050149. [CrossRef]
6. Zhai, W.; Fu, S.; Zhou, T.; Xiu, C. Exponentially-Fitted and Trigonometrically-Fitted Implicit RKN Methods for Solving $y'' = f(t, y)$. *J. Appl. Math. Comput.* **2022**, *68*, 1449–1466. [CrossRef]
7. Van Daele, M.; Vanden Berghe, G. Geometric Numerical Integration by Means of Exponentially-Fitted Methods. *Appl. Numer. Math.* **2007**, *57*, 415–435. [CrossRef]
8. Ern, A.; Guermond, J.-L. Invariant-Domain-Preserving High-Order Time Stepping: I. Explicit Runge-Kutta Schemes. *SIAM J. Sci. Comput.* **2022**, *44*, A3366–A3392. [CrossRef]
9. Ryland, B.N.; McLachlan, R.I.; Frank, J. On the Multisymplecticity of Partitioned Runge-Kutta and Splitting Methods. *Int. J. Comput. Math.* **2007**, *84*, 847–869. [CrossRef]

10. Anastassi, Z.A.; Kosti, A.A.; Rufai, M.A. A Parametric Method Optimised for the Solution of the (2+1)-Dimensional Nonlinear Schrödinger Equation. *Mathematics* **2023**, *11*, 609. [CrossRef]
11. Babaei, M. A Swarm-Intelligence Based Formulation for Solving Nonlinear ODEs: $\gamma\beta$II-(2+3)P Method. *Appl. Soft Comput.* **2024**, *162*, 111424. [CrossRef]
12. Engelbrecht, A.P. *Computational Intelligence: An Introduction*, 2nd ed.; Wiley: Hoboken, NJ, USA, 2007. [CrossRef]
13. Bäck, T.; Fogel, D.B.; Michalewicz, Z. (Eds.) *Evolutionary Computation 2—Advanced Algorithms and Operators*; Institute of Physics Publishing: Bristol, UK, 2000. [CrossRef]
14. Tang, J.; Liu, G.; Pan, Q. A Review on Representative Swarm Intelligence Algorithms for Solving Optimization Problems: Applications and Trends. *IEEE/CAA J. Autom. Sin.* **2021**, *8*, 1627–1643. [CrossRef]
15. Liu, Z.-Z.; Wang, Y.; Yang, S.; Cai, Z. Differential Evolution with a Two-Stage Optimization Mechanism for Numerical Optimization. In Proceedings of the 2016 IEEE Congress on Evolutionary Computation (CEC), Vancouver, BC, Canada, 24–29 July 2016; pp. 3170–3177. [CrossRef]
16. Ahmadianfar, I.; Heidari, A.A.; Gandomi, A.H.; Chu, X.; Chen, H. RUN Beyond the Metaphor: An Efficient Optimization Algorithm Based on Runge-Kutta Method. *Expert Syst. Appl.* **2021**, *181*, 115079. [CrossRef]
17. Babaei, M. An Efficient ODE-Solving Method Based on Heuristic and Statistical Computations: αII-(2+3)P Method. *J. Supercomput.* **2024**, *80*, 20302–20345. [CrossRef]
18. Neelan, G.A.A.; Nair, M.T. Hyperbolic Runge-Kutta Method Using Evolutionary Algorithm. *J. Comput. Nonlinear Dyn.* **2018**, *13*, 101003. [CrossRef]
19. Kovalnogov, V.N.; Fedorov, R.V.; Khakhalev, Y.A.; Simos, T.E.; Tsitouras, C. A Neural Network Technique for the Derivation of Runge-Kutta Pairs Adjusted for Scalar Autonomous Problems. *Mathematics* **2021**, *9*, 1842. [CrossRef]
20. Jerbi, H.; Ben Aoun, S.; Omri, M.; Simos, T.E.; Tsitouras, C. A Neural Network Type Approach for Constructing Runge-Kutta Pairs of Orders Six and Five That Perform Best on Problems with Oscillatory Solutions. *Mathematics* **2022**, *10*, 827. [CrossRef]
21. Kennedy, J.; Eberhart, R. Particle Swarm Optimization. In Proceedings of the IEEE International Conference on Neural Networks (ICNN'95), Perth, WA, Australia, 27 November–1 December 1995; Volume 4, pp. 1942–1948. [CrossRef]
22. Eberhart, R.C.; Shi, Y. Particle Swarm Optimization: Developments, Applications and Resources. In Proceedings of the IEEE Conference on Evolutionary Computation (ICEC), Seoul, Republic of Korea, 27–30 May 2001; pp. 81–86. [CrossRef]
23. Pedersen, M.E.H. *Good Parameters for Particle Swarm Optimization*; Technical Report no. HL1001; Hvass Laboratories: Naples, FL, USA, 2010. [CrossRef]
24. Isiet, M.; Gadala, M. Sensitivity Analysis of Control Parameters in Particle Swarm Optimization. *J. Comput. Sci.* **2020**, *41*, 101086. [CrossRef]
25. Iadevaia, S.; Lu, Y.; Morales, F.C.; Mills, G.B.; Ram, P.T. Identification of Optimal Drug Combinations Targeting Cellular Networks: Integrating Phospho-Proteomics and Computational Network Analysis. *Cancer Res.* **2010**, *70*, 6704–6714. [CrossRef]
26. Shami, T.M.; El-Saleh, A.A.; Alswaitti, M.; Al-Tashi, Q.; Summakieh, M.A.; Mirjalili, S. Particle Swarm Optimization: A Comprehensive Survey. *IEEE Access* **2022**, *10*, 10031–10061. [CrossRef]
27. Suganthan, P.N. Particle Swarm Optimiser with Neighbourhood Operator. In Proceedings of the 1999 Congress on Evolutionary Computation (CEC 1999), Washington, DC, USA, 6–9 July 1999; pp. 1958–1962. [CrossRef]
28. Wang, H.; Sun, H.; Li, C.; Rahnamayan, S.; Pan, J. Diversity Enhanced Particle Swarm Optimization with Neighborhood Search. *Inf. Sci.* **2013**, *223*, 119–135. [CrossRef]
29. Li, X. Adaptively Choosing Neighbourhood Bests Using Species in a Particle Swarm Optimizer for Multimodal Function Optimization. In *Lecture Notes in Computer Science*; Springer: Berlin/Heidelberg, Germany, 2004; Volume 3102, pp. 105–116. [CrossRef]
30. Piotrowski, A.P.; Napiorkowski, J.J.; Piotrowska, A.E. Population Size in Particle Swarm Optimization. *Swarm Evol. Comput.* **2020**, *58*, 100718. [CrossRef]
31. Fibich, G. *The Nonlinear Schrödinger Equation: Singular Solutions and Optical Collapse*; Springer: Cham, Switzerland, 2015. [CrossRef]
32. Feng, D.; Jiao, J.; Jiang, G. Optical solitons and periodic solutions of the (2+1)-dimensional nonlinear Schrödinger's equation. *Phys. Lett. A* **2018**, *382* 2081–2084. [CrossRef]
33. Guardia, M.; Hani, Z.; Haus, E.; Maspero, A.; Procesi, M. Strong nonlinear instability and growth of Sobolev norms near quasiperiodic finite gap tori for the 2D cubic NLS equation. *J. Eur. Math. Soc.* **2023**, *25*, 1497–1551. [CrossRef]
34. Anastassi, Z.A.; Simos, T.E. A Trigonometrically Fitted Runge-Kutta Method for the Numerical Solution of Orbital Problems. *New Astron.* **2005**, *10*, 301–309. [CrossRef]
35. Anastassi, Z.A.; Simos T.E. Numerical Multistep Methods for the Efficient Solution of Quantum Mechanics and Related Problems. *Phys. Rep.* **2009**, *482–483*, 1–240. [CrossRef]
36. Panopoulos, G.A.; Anastassi, Z.A.; Simos T.E. A New Eight-Step Symmetric Embedded Predictor-Corrector Method (EPCM) for Orbital Problems and Related IVPs with Oscillatory Solutions. *Astron. J.* **2013**, *145*, 75. [CrossRef]

37. Papageorgiou, G.; Tsitouras, C.; Papakostas, S.N. Runge-Kutta pairs for periodic initial value problems. *Computing* **1993**, *51*, 151–163. [CrossRef]
38. Kosti, A.A.; Colreavy-Donnelly, S.; Caraffini, F.; Anastassi, Z.A. Efficient Computation of the Nonlinear Schrödinger Equation with Time-Dependent Coefficients. *Mathematics* **2020**, *8*, 374. [CrossRef]
39. Kosti, A.A.; Anastassi, Z.A.; Simos, T.E. An optimized explicit Runge-Kutta method with increased phase-lag order for the numerical solution of the Schrödinger equation and related problems. *J. Math. Chem.* **2010**, *47*, 315. [CrossRef]
40. Triantafyllidis, T.V.; Anastassi, Z.A.; Simos, T.E. Two optimized Runge-Kutta methods for the solution of the Schrödinger equation. *MATCH Commun. Math. Comput. Chem.* **2008**, *60*, 753–771. Available online: https://match.pmf.kg.ac.rs/electronic_versions/Match60/n3/match60n3_753-771.pdf (accessed on 1 July 2025).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.