mathematics

MDPI

Article

On Reusing the Stages of a Rejected Runge-Kutta Step

Vladislav N. Kovalnogov [1], Ruslan V. Fedorov [1], Tamara V. Karpukhina [1], Theodore E. Simos [1,2,3,4,5,6,*] and Charalampos Tsitouras [7]

[1] Laboratory of Inter-Disciplinary Problems of Energy Production, Ulyanovsk State Technical University, 32 Severny Venetz Street, 432027 Ulyanovsk, Russia; kvn@ulstu.ru (V.N.K.); r.fedorov@ulstu.ru (R.V.F.); tkarpukhina@yandex.ru (T.V.K.)
[2] Center for Applied Mathematics and Bioinformatics, Gulf University for Science and Technology, West Mishref 32093, Kuwait
[3] Department of Medical Research, China Medical University Hospital, China Medical University, Taichung City 40402, Taiwan
[4] Data Recovery Key Laboratory of Sichun Province, Neijiang Normal University, Neijiang 641100, China
[5] Section of Mathematics, Department of Civil Engineering, Democritus University of Thrace, 67100 Xanthi, Greece
[6] Department of Mathematics, University of Western Macedonia, 50100 Kozani, Greece
[7] General Department, National & Kapodistrian University of Athens, Euripus Campus, 34400 Evia, Greece; tsitourasc@uoa.gr
* Correspondence: tsimos.conf@gmail.com

Abstract: Runge-Kutta (RK) pairs are amongst the most popular methods for numerically solving Initial Value Problems. While using an RK pair, we may experience rejection of some steps through the interval of integration. Traditionally, all of the evaluations are then dropped, and we proceed with a completely new round of computations. In this work, we propose avoiding this waste and continuing by reusing the rejected RK stages. We focus especially on an RK pair of orders six and five. After step rejection, we reuse all the previously evaluated stages and only add three new stages. We proceed by evaluating the output using a smaller step. By this technique, we manage to significantly reduce the cost in a set of problems that are known to pose difficulties in RK algorithms with changing step sizes.

Keywords: initial value problem; Runge-Kutta; step control

MSC: 65L05; 65L06

Citation: Kovalnogov, V.N.; Fedorov, R.V.; Karpukhina, T.V.; Simos, T.E.; Tsitouras, C. On Reusing the Stages of a Rejected Runge-Kutta Step. *Mathematics* **2023**, *11*, 2589. https://doi.org/10.3390/math11112589

Academic Editor: Rodica Luca

Received: 9 May 2023
Revised: 2 June 2023
Accepted: 4 June 2023
Published: 5 June 2023

1. Introduction

A fundamental issue in mathematical analysis, the Initial Value Problem (IVP) has applications in a variety of disciplines, including physics, engineering, and economics. The IVP has the following form

$$y' = f(x, y),\ y(x_0) = y_0 \in \mathbb{R}^m,\ x \in [x_0, x_e], \tag{1}$$

with $f : \mathbb{R} \times \mathbb{R}^m \to \mathbb{R}^m$ continuously differentiable.

The IVP's purpose is to determine the solution to a differential equation given the starting condition, which is the value of the dependent variable at a specified initial time. The most frequent approach for solving the IVP is to use numerical methods, which entail estimating the solution of the differential equation using discrete time steps.

The Runge-Kutta (RK) method is one of the most prominent and commonly used numerical methods for solving the IVP. The RK technique was developed in the late 1800s by German mathematicians Carl Runge and Martin Kutta. It is a class of numerical methods that estimate the value of the solution at the end of a time step by taking a weighted average of function values at different points in the time step. The algebraic order of the RK method relates to the number of function evaluations for each time step, which impacts the technique's accuracy.

For solving the IVP, the RK approach offers significant benefits over other numerical methods. It is a basic and easy-to-implement approach that is understandable to non-mathematicians. It is also a flexible approach that may be used to solve a broad variety of differential equations, including stiff problems that are difficult to solve with other numerical methods. Furthermore, the RK approach is a self-starting technique, which implies that no extra equations or conditions must be solved.

However, the RK method's accuracy is dependent on the sequence of the process, which increases the technique's computing cost and complexity. To overcome this issue, the Runge-Kutta pair, a version of the RK technique, was devised. A Runge-Kutta pair is a pair of RK methods using the same body of function evaluations but distinct weight coefficients that are used to estimate the solution and error at each time step. The estimated error is utilised to adjust the step size in the following time step, leading to a more accurate and efficient technique than the single RK method.

The general form of an s-stage Runge-Kutta pair of orders p and $p-1$ can be written as follows (remark that clearly $s > p$ for $p > 4$):

$$\begin{aligned} y_{n+1} &= y_n + h_n \sum_{i=1}^{s} b_i k_{ni}, \quad p-\text{th order aproximation}, \\ \hat{y}_{n+1} &= y_n + h_n \sum_{i=1}^{s} \hat{b}_i k_{ni}, \quad (p-1)-\text{th order aproximation}, \end{aligned}$$

with

$$k_{ni} = f(x_n + c_i h_n, y_n + h_n \sum_{j=1}^{i-1} a_{ij} k_{nj}),\ i = 1, 2, \cdots, s,$$

where y_n is the approximate solution at time x_n, h_n is the time step size, k_{ni} are the intermediate function values, and b_i, $\hat{b}_i$, c_i, and a_{ij} are the RK coefficients. We assume that the coefficients form the column vectors b^T, $\hat{b}^T$ $c \in \mathbb{R}^s$ and the matrix $A \in \mathbb{R}^{s \times s}$. The error estimate can be calculated as follows:

$$e_{n+1} = \|y_{n+1} - \hat{y}_{n+1}\| = \|h_n \sum_{i=1}^{s} (b_i - \hat{b}_i) k_{ni}\|. \tag{2}$$

By comparing the error estimate to a specified tolerance level TOL and modifying the time-step size accordingly, the step size may be adjusted. This enables a more efficient and precise technique for resolving the IVP. For achieving this, we use the formula

$$h_{n+1} = 0.9 \cdot h_n \cdot \left(\frac{TOL}{e_{n+1}} \right)^{(1/p)}. \tag{3}$$

In case $e_{n+1} > TOL$, the step is rejected, i.e., we do not move forward from x_n to x_{n+1}. Then, we again use (3) but replace the left-hand side with h_n. This is a classical procedure: see [1] (p. 167).

There have been several studies on the performance and accuracy of Runge-Kutta pairs in solving the IVP. Among the first such studies, Fehlberg compared the performance of different RK pairs of order 5 and 6 on a set of test problems and found that a particular RK pair, known as the Fehlberg pair, was the most efficient and accurate pair for a wide range of problems [2]. These pairs by Fehlberg are quadrature defective. Thus, for quadratic problems, the lower and higher order results coincide, and error control fails.

In another study, Prince and Dormand introduced a new RK pair of order 6 and 5 that avoided this defect and showed improved performance compared to other RK pairs [3].

In recent years, there have been developments in using RK pairs in combination with adaptive time-stepping algorithms to solve the IVP. One such algorithm is the Dormand–Prince method, which uses an embedded RK pair of order 5 and 4 with adaptive time-step size control to improve the accuracy and efficiency of the method [4]. Another algorithm is the Tsitouras method, which uses an embedded RK pair of order 5 and 6 with adaptive

time-step size control; it has been shown to be more efficient and accurate than other RK pairs on certain types of problems [5].

In conclusion, the Runge-Kutta pair is a powerful and versatile numerical method for solving the IVP. It provides a more accurate and efficient method than using a single RK method by using a pair of methods of the same order with different coefficients to estimate the value of the solution and the error at each time step. There have been several studies on the performance and accuracy of RK pairs, with some pairs designed specifically for stiff equations and others optimized for efficiency and accuracy on a wide range of problems. Adaptive time-stepping algorithms, such as the Dormand–Prince and Tsitouras pairs, have been developed to further improve the accuracy and efficiency of the RK pair method.

2. The New Step-Size Control Scheme

After using the devised Equations (2) and (3) for changing the step size, we may experience step rejection. Then, traditionally, all the stages are completely wasted and we proceed by evaluating a new smaller step. This unjustified loss will be discussed here. We intend to suggest extension of existing pairs in order to handle these stages of a rejected step again and reduce the cost.

Thus, after an s-stage RK pair is applied and the error estimation fails to stay below ϵ_{n+1}, we reduce the current step to have a new length, say τh_n, $\tau < 1$. We then extend the pair by adding some more stages. After having at hand the s stages of the underlying pair, given by (1), we additionally compute

$$k_{ni} = f(x_n + c_i h_n, y_n + h_n \sum_{j=1}^{i-1} a_{ij} k_{nj}),\ i = s+1, s+2, \cdots, \tilde{s},$$

and combine all these $\tilde{s}$ stages in a new pair of formulas

$$\begin{aligned} y^*_{n+1} &= y_n + h_n \sum_{i=1}^{\tilde{s}} b^*_i k_{ni}, \quad p-\text{th order aproximation,} \\ \hat{y}^*_{n+1} &= y_n + h_n \sum_{i=1}^{\tilde{s}} \hat{b}^*_i k_{ni}, \quad (p-1)-\text{th order aproximation.} \end{aligned}$$

i.e., the new weights $b^*, \hat{b}^*$ are used in order to approximate $y^*_{n+1} \approx y(x_n + \tau h_n)$.

The error estimate now becomes

$$e^*_{n+1} = \|y^*_{n+1} - \hat{y}^*_{n+1}\| = \|h_n \sum_{i=1}^{\tilde{s}} (b^*_i - \hat{b}^*_i) k_{ni}\|,$$

and the slightly modified formula

$$h_{n+1} = 0.9 \cdot h_n \cdot \left(\frac{TOL}{e^*_{n+1}} \right)^{(1/p)}.$$

is used for advancing the integration.

In consequence, we spend only $(\tilde{s} - s)$ stages after a step rejection instead of the s-stages lost with the standard step-size-control algorithm. Obviously, we clearly demand $(\tilde{s} - s) < s$.

For safety reasons, we only apply the new algorithm in case

$$TOL < \epsilon_{n+1} < \lambda \cdot TOL,$$

for some $\lambda > 1$. Otherwise, we continue with the standard policy (2)–(3).

We assume that the new extended set of coefficients now forms the column vectors b^{*T}, $\hat{b}^{*T}$ $c \in \mathbb{R}^{\tilde{s}}$ and the new extended matrix $A \in \mathbb{R}^{\tilde{s} \times \tilde{s}}$.

3. Evaluation of a Mew-Extended Runge-Kutta Pair

In the present work, we concentrate on a pair of orders six and five. We choose the DLMP6(5) pair that appeared in [6]. This pair seems to be among the best choices for moderate tolerances (i.e., $10^{-4} > TOL > 10^{-9}$, see [5]). For this case, $s = 9$, but since

$$a_{9j} = b_j,\ j = 1, 2, \cdots, 9,$$

the last stage of each step can be used as the first stage of the next step. This is called FSAL (First Stage As Last), and then DLMP6(5) only actually spends 8 stages per step. The same number of stages is wasted after a step rejection using (2) and (3).

We set $\tau = 0.8$ and $\tilde{s} = 12$. The new extended pair (named DLMP6(5)ext) shares an enhanced set of coefficients, as shown in the previous subsection. When a step is rejected, we evaluate these three new stages, $k_{n,10}, k_{n,11}$ and $k_{n,12}$, and combine them with the new set of weights $b^*, \hat{b}^*$ to form the new approximations for the point $x_n + \tau h_n$.

Thus, we have to evaluate the coefficients

$$a_{10,1}, a_{10,3}, a_{10,4}, \cdots, a_{10,9}, a_{11,1}, a_{11,3}, a_{11,4}, \cdots, a_{11,10}, a_{12,1}, a_{12,3}, a_{12,4}, \cdots, a_{12,11},$$

$$c_{10}, c_{11}, c_{12},\ b_1^*, b_4^*, b_5^*, \cdots, b_{12}^*, \hat{b}_1^*,\ \hat{b}_4^*, \hat{b}_5^*, \cdots, \hat{b}_{11}^*,$$

since for the considered type of pairs, it is appropriate to set

$$a_{10,2} = a_{11,2} = a_{12,2} = b_2^* = b_3^* = \hat{b}_2^* = \hat{b}_3^* = \hat{b}_{12}^* = 0.$$

The coefficients of DLMP6(5) along with the above ones have to satisfy the order conditions listed in Tables 1 and 2.

Table 1. Order conditions for the higher-order weights of DLMP6(5)ext, i.e., b^*.

$d_{1,1} = b^*\mathbb{I} - \tau$,	$d_{2,1} = b^*c - \frac{\tau^2}{2}$,	$d_{3,1} = \frac{1}{2}\left(b^*c^2 - \frac{\tau^3}{3}\right)$,
$d_{3,2} = b^*Ac - \frac{\tau^3}{6}$,	$d_{4,1} = \frac{1}{6}\left(b^*c^3 - \frac{\tau^4}{4}\right)$,	$d_{4,2} = b^*Ac^2 - \frac{\tau^4}{12}$,
$d_{4,3} = b^*(c * Ac) - \frac{1}{24}$,	$d_{4,4} = b^*A^2c - \frac{\tau^4}{24}$,	$d_{5,1} = \frac{1}{24}\left(b^*c^4 - \frac{\tau^5}{5}\right)$,
$d_{5,2} = \frac{1}{2}\left(b^*(c^2 * Ac) - \frac{\tau^5}{10}\right)$,	$d_{5,3} = \frac{1}{2}\left(b^*(Ac)^2 - \frac{\tau^5}{10}\right)$,	$d_{5,4} = \frac{1}{2}\left(b^*(c * Ac^2) - \frac{\tau^5}{15}\right)$,
$d_{5,5} = b^*(c * A^2c) - \frac{\tau^5}{10}$,	$d_{5,6} = \frac{1}{6}\left(b^*Ac^3 - \frac{\tau^5}{20}\right)$,	$d_{5,7} = b^*A(c * Ac) - \frac{\tau^5}{40}$,
$d_{5,8} = \frac{1}{2}\left(b^*A^2c^2 - \frac{\tau^5}{60}\right)$,	$d_{5,9} = b^*A^3c - \frac{\tau^5}{120}$,	$d_{6,1} = \frac{1}{120}\left(b^*c^5 - \frac{\tau^6}{6}\right)$,
$d_{6,2} = \frac{1}{6}\left(b^*(c^3 * Ac) - \frac{\tau^6}{12}\right)$,	$d_{6,3} = 2\left(b^*\left(c * (Ac)^2\right) - \frac{\tau^6}{24}\right)$,	$d_{6,4} = \frac{1}{4}\left(b^*(c^2 * Ac^2) - \frac{\tau^6}{18}\right)$,
$d_{6,5} = \frac{1}{2}\left(b^*(c^2 * A^2c) - \frac{\tau^6}{36}\right)$,	$d_{6,6} = \frac{1}{2}\left(b^*(Ac * Ac^2) - \frac{\tau^6}{36}\right)$,	$d_{6,7} = \frac{1}{2}\left(b^*(Ac * A^2c) - \frac{\tau^6}{72}\right)$,
$d_{6,8} = \frac{1}{6}\left(b^*(c * Ac^3) - \frac{\tau^6}{24}\right)$,	$d_{6,9} = b^*(c * A(c * Ac)) - \frac{\tau^6}{48}$,	$d_{6,10} = \frac{1}{2}\left(b^*(c * A^2c^2) - \frac{\tau^6}{72}\right)$,
$d_{6,11} = b^*(c * A^3c) - \frac{\tau^6}{144}$,	$d_{6,12} = \frac{1}{24}\left(b^*Ac^4 - \frac{\tau^6}{30}\right)$,	$d_{6,13} = \frac{1}{2}\left(b^*A(c^2 * Ac) - \frac{\tau^6}{60}\right)$,
$d_{6,14} = \frac{1}{2}\left(b^*A(Ac)^2 - \frac{\tau^6}{120}\right)$,	$d_{6,15} = \frac{1}{2}\left(b^*A(c * Ac^2) - \frac{\tau^6}{90}\right)$,	$d_{6,16} = b^*A(c * A^2c) - \frac{\tau^6}{180}$,
$d_{6,17} = \frac{1}{6}\left(b^*A^2c^3 - \frac{\tau^6}{120}\right)$,	$d_{6,18} = b^*A^2(c * Ac) - \frac{\tau^6}{240}$,	$d_{6,19} = \frac{1}{2}\left(b^*A^3c^2 - \frac{\tau^6}{360}\right)$,
$d_{6,20} = b^*A^4c - \frac{\tau^6}{720}$.		

Table 2. Order conditions for the lower-order weights of DLMP6(5)ext, i.e., $\hat{b}^*$.

$\hat{d}_{1,1} = b^*\mathbb{I} - \tau$,	$\hat{d}_{2,1} = \hat{b}^*c - \frac{\tau^2}{2}$,	$\hat{d}_{3,1} = \frac{1}{2}\left(\hat{b}^*c^2 - \frac{\tau^3}{3}\right)$,
$\hat{d}_{3,2} = \hat{b}^*Ac - \frac{\tau^3}{6}$,	$\hat{d}_{4,1} = \frac{1}{6}\left(\hat{b}^*c^3 - \frac{\tau^4}{4}\right)$,	$\hat{d}_{4,2} = \hat{b}^*Ac^2 - \frac{\tau^4}{12}$,
$\hat{d}_{4,3} = \hat{b}^*(c * Ac) - \frac{1}{24}$,	$\hat{d}_{4,4} = \hat{b}^*A^2c - \frac{\tau^4}{24}$,	$\hat{d}_{5,1} = \frac{1}{6}\left(\hat{b}^*c^4 - \frac{\tau^5}{5}\right)$,
$\hat{d}_{5,2} = \frac{1}{2}\left(\hat{b}^*(c^2 * Ac) - \frac{\tau^5}{10}\right)$,	$\hat{d}_{5,3} = \frac{1}{2}\left(\hat{b}^*(Ac)^2 - \frac{\tau^5}{10}\right)$,	$\hat{d}_{5,4} = \frac{1}{2}\left(\hat{b}^*(c * Ac^2) - \frac{\tau^5}{15}\right)$,
$\hat{d}_{5,5} = \hat{b}^*(c * A^2c) - \frac{\tau^5}{10}$,	$\hat{d}_{5,6} = \frac{1}{6}\left(\hat{b}^*Ac^3 - \frac{\tau^5}{20}\right)$,	$\hat{d}_{5,7} = \hat{b}^*A(c * Ac) - \frac{\tau^5}{40}$,
$\hat{d}_{5,8} = \frac{1}{2}\left(\hat{b}^*A^2c^2 - \frac{\tau^5}{60}\right)$,	$\hat{d}_{5,9} = \hat{b}^*A^3c - \frac{\tau^5}{120}$.	

In these Tables, we mention with d_{ji} the $j-$th-order conditions. By $*$, we denote component-wise multiplication among vectors. This multiplication has lesser priority than the dot product. We also use the notation $c^2 = c * c$, $c^3 = c * c * c$, etc. By $A^2 = A \cdot A$ or $Ac = A \cdot c$, we denote the classical dot product. We also set $\mathbb{I} = [1, 1, \cdots, 1]^T \in \mathbb{R}^{\tilde{s}}$.

For solving the above equations, we make some simplifying assumptions. Namely

$$A\mathbb{I} = c, (\ A \cdot c = \frac{1}{2}c^2)_{(3+)},\ (A \cdot c^2 = \frac{1}{3}c^3)_{(3+)},$$

and

$$b \cdot (A + C - \tau I_{\tilde{s}}) = 0_{\tilde{s}}.$$

In the above, we use the symbolization $v_{(i+)}$ with the elements of some vector v but with its first $(i-1)$ elements dropped. We set $C = \mathrm{diag}(c)$, and by $I_{\tilde{s}}$ and $0_{\tilde{s}}$, we mean, respectively, the identity and zero matrix in $\mathbb{R}^{\tilde{s}\times\tilde{s}}$.

Then, there is a severe reduction in the number of the order conditions to be solved. We are able to analytically solve these remaining order conditions with respect to the new coefficients. Indeed, when the above assumptions hold, only the order conditions

$$d_{6,1}, d_{5,1}, d_{4,1}, d_{3,1}, d_{2,1}, d_{1,1}, d_{5,4}, d_{5,5}, d_{6,16}, d_{6,8}, d_{6,11}, d_{6,19},$$

and

$$\hat{d}_{5,1}, \hat{d}_{4,1}, \hat{d}_{3,1}, \hat{d}_{2,1}, \hat{d}_{1,1}, \hat{d}_{5,4}, \hat{d}_{5,5},$$

have to be solved. The task is becoming a lot easier. We even manage to solve the seventh-order equations of condition for the higher-order formula. The resulting pair is shown in Table 3. These coefficients can be retrieved in Mathematica [7] format from http://users.uoa.gr/~tsitourasc/dlmp65ext.m (accessed on 3 June 2023).

An interpolant of the sixth-order at the same cost can be constructed for this type of pair [8]. Then, we may freely choose the magnitude of τ and vary the lengths of the new step after rejection. By fixing $\tau = 0.8$ and achieving locally seventh-order accuracy after rejection, we obtain better results overall.

Table 3. The coefficients of the new extended pair.

$c_2 = \frac{1}{9}, c_3 = \frac{1}{6}, c_4 = \frac{1}{4},$	$c_5 = \frac{5}{9}, c_6 = \frac{1}{2}, c_7 = \frac{48}{49},$
$c_8 = 1, c_9 = 1, c_{10} = \frac{4}{139},$	$c_{11} = \frac{17}{38}, c_{12} = \frac{4}{5},$
$b_1^* = -0.06075441182658404,$	$b_2^* = b_3^* = 0,$
$b_4^* = 0.25108031811087983,$	$b_5^* = 0.59459248062264663,$
$b_6^* = -0.58130691768291823,$	$b_7^* = -0.01117792906462664,$
$b_8^* = 0.001953125,$	$b_9^* = 0.00453876219794998,$
$b_{10}^* = 0.18340955527240297,$	$b_{11}^* = 0.33291925465838509,$
$b_{12}^* = 0.08474576271186441,$	$\hat{b}_1^* = -0.0607545222182737630,$
$\hat{b}_2^* = \hat{b}_3^* = 0 = \hat{b}_{12}^*,$	$\hat{b}_4^* = 0.362681592201453867,$
$\hat{b}_5^* = 1.18886870906761734,$	$\hat{b}_6^* = -1.20278300666332157,$
$\hat{b}_7^* = -0.357600832335522983,$	$\hat{b}_8^* = 0.232809581363277529$
$\hat{b}_9^* = 0.0760545523116338381$	$\hat{b}_{10}^* = 0.163215379071331048$
$\hat{b}_{11}^* = 0.314851188060490077$	$\hat{b}_{12}^* = 0.0826573591413146190, b_1 = \frac{203}{2880},$
$b_2 = b_3 = 0 = b_9,$	$b_4 = \frac{30208}{70785},$
$b_5 = \frac{177147}{164560},$	$b_6 = -\frac{536}{705},$
$b_7 = \frac{1977326743}{3619661760},$	$b_8 = -\frac{259}{720},$
$\hat{b}_1 = \frac{36567}{458800},$	$\hat{b}_4 = \frac{9925984}{27063465},$
$\hat{b}_5 = \frac{85382667}{117968950},$	$\hat{b}_6 = -\frac{310378}{808635},$
$\hat{b}_7 = \frac{262119736669}{345979336560},$	$\hat{b}_8 = -\frac{1}{2}, \hat{b}_8 = -\frac{101}{2294},$
$a_{21} = 0.111111111111111111,$	$a_{31} = 0.0416666666666666667,$

Table 3. *Cont.*

$a_{32} = 0.125,$	$a_{41} = 0.0625,$
$a_{j2} = 0, j = 4, 5, \cdots, 12,$	$a_{43} = 0.1875,$
$a_{51} = 0.384087791495198903,$	$a_{53} = -1.33744855967078189,$
$a_{54} = 1.50891632373113855,$	$a_{61} = 0.417370572207084469,$
$a_{63} = -1.46730245231607629,$	$a_{64} = 1.60862026257121625,$
$a_{65} = -0.0586883824622244241,$	$a_{71} = -0.906581932271243731,$
$a_{73} = 1.98165828767968130,$	$a_{74} = 0.967924991130227440,$
$a_{75} = 7.90644976448593311,$	$a_{76} = -8.96985927428990425,$
$a_{81} = -1.23125466844812894,$	$a_{83} = 2.33058398998453494,$
$a_{84} = 1.69577556052661329,$	$a_{85} = 10.8007435894539014,$
$a_{86} = -12.5648566499630329,$	$a_{87} = -0.0309918215538877730,$
$a_{9j} = b_j, j = 1, 2, \cdots, 8,$	$a_{101} = 0.0276060694624219017,$
$a_{103} = -0.18678058047598361,$	$a_{104} = 0.391371551663676298,$
$a_{105} = 1.09230024433914178,$	$a_{106} = -1.22247349711209067,$
$a_{107} = -0.556216395594661712,$	$a_{108} = 0.356521739130434783,$
$a_{109} = 0.126447847004327,$	$a_{111} = 0.0192549367566782782,$
$a_{114} = 0.496087246358859837,$	$a_{115} = -1.18052838103602307,$
$a_{116} = 1.29939201810168170,$	$a_{117} = 0.586956521739130435,$
$a_{118} = -0.367816091954022989,$	$a_{119} = -0.142156862745098039,$
$a_{1110} = 0.281632150794417543,$	$a_{121} = -0.820970265019910839,$
$a_{124} = -0.653270781790705787,$	$a_{125} = 4.32243201762434916,$
$a_{126} = -5.36952327363607790,$	$a_{127} = -1.10690062359555245,$
$a_{128} = 0.688006483439893015,$	$a_{129} = 0.274081679397217048,$
$a_{1210} = 0.562729086953349127,$	$a_{1211} = 1.38529454069957502,$
$a_{113} = -0.545453116962992122,$	$a_{123} = 1.51812113592786359$

4. Numerical Results

We have chosen the detest problems D4, D5, and E2 as the set to perform our tests on. Integrating these problems by any $6(5)$ pair, we observe that almost 25% of the computational cost is wasted upon rejected steps. The problems chosen are given below.

Problem D4

This is the two-body problem with eccentricity $e = 0.7$, and it has the following form [9]

$$y_1' = y_3,\ y_2' = y_4,\ y_3' = -\frac{y_1}{(y_1^2+y_2^2)^{3/2}},\ y_4' = -\frac{y_2}{(y_1^2+y_2^2)^{3/2}},$$
$$y_1(0) = e = \frac{7}{10},\ y_2(0) = 0,\ y_3(0) = 0,\ y_4(0) = \sqrt{\frac{17}{3}},\ x \in [0, 20].$$

Notice that subscripts above denote the component of the problem and not time steps. We solve the above equations in the interval $[0, 20]$, and the theoretical solution is given in [10].

After we run this problem for tolerances $TOL = 10^{-4}, 10^{-5}, \cdots, 10^{-9}$ and $\lambda = 7$, we record in Tables 4 and 5 all the characteristics of the results. In the final column, we present an efficiency measure according to the formula [11,12]

$$\text{efficiency} = (\text{total function evaluations}) \cdot (\text{end point error})^{1/6}. \tag{4}$$

This information is perhaps more significant because it ensures that we do not simply decrease the number of rejected steps with a very conservative step-size algorithm, which, in reverse, decreases efficiency.

Table 4. Results of DLMP6(5) over the problem D4 using the standard step-size-change algorithm.

TOL	Accepted Steps	Rejected Steps	Extended Steps	End Point Error	Efficiency (4)
10^{-4}	60	24	0	1.0×10^{-3}	213.6
10^{-5}	78	29	0	2.1×10^{-4}	208.5
10^{-6}	108	39	0	2.5×10^{-5}	201.7
10^{-7}	151	48	0	2.6×10^{-6}	186.5
10^{-8}	215	58	0	2.2×10^{-7}	169.2
10^{-9}	303	10	0	2.2×10^{-8}	132.1

Table 5. Results of DLMP6(5) over the problem D4 using the new step-size-change algorithm.

TOL	Accepted Steps	Rejected Steps	Extended Steps	End Point Error	Efficiency (4)
10^{-4}	58	7	13	3.3×10^{-4}	150.3
10^{-5}	76	5	13	6.7×10^{-5}	141.0
10^{-6}	107	0	20	4.2×10^{-6}	119.0
10^{-7}	155	0	26	2.0×10^{-6}	151.4
10^{-8}	224	0	36	1.7×10^{-7}	144.0
10^{-9}	307	0	9	2.1×10^{-8}	131.2

Problem D5

This is again the two-body problem (i.e., Kepler problem) as above but with eccentricity $e = 0.9$, and the initial conditions are now

$$y_1(0) = e = \frac{9}{10},\ y_2(0) = 0,\ y_3(0) = 0,\ y_4(0) = \sqrt{19},\ x \in [0, 20].$$

The corresponding results are given in Tables 6 and 7.

Table 6. Results of DLMP6(5) over the problem D5 using the standard step-size-change algorithm.

TOL	Accepted Steps	Rejected Steps	Extended Steps	End Point Error	Efficiency (4)
10^{-4}	90	29	0	4.3×10^{-3}	384.1
10^{-5}	118	45	0	3.0×10^{-4}	337.0
10^{-6}	160	58	0	3.8×10^{-5}	319.4
10^{-7}	223	75	0	$3.7.1 \times 10^{-6}$	296.8
10^{-8}	316	92	0	2.9×10^{-7}	265.9
10^{-9}	449	64	0	2.6×10^{-8}	224.1

Table 7. Results of DLMP6(5) over the problem D5, using the new step-size-change algorithm.

TOL	Accepted Steps	Rejected Steps	Extended Steps	End Point Error	Efficiency (4)
10^{-4}	89	18	10	4.0×10^{-3}	356.9
10^{-5}	117	23	13	4.9×10^{-5}	224.3
10^{-6}	158	3	27	2.2×10^{-5}	232.8
10^{-7}	227	0	40	2.4×10^{-6}	229.0
10^{-8}	327	0	52	2.0×10^{-7}	215.7
10^{-9}	457	0	36	2.2×10^{-8}	200.5

Problem E2

This is the Van der Pol oscillator driven by the equations

$$y_1' = y_2,\ y_2' = (1 - y_1^2)y_2 - y_1,\ y_1(0) = 2, y_2(0) = 0,\ x \in [0, 20].$$

The corresponding results are given in Tables 8 and 9.

Table 8. Results of DLMP6(5) over the problem E2 using the standard step-size-change algorithm.

TOL	Accepted Steps	Rejected Steps	Extended Steps	End Point Error	Efficiency (4)
10^{-4}	50	25	0	2.5×10^{-4}	150.9
10^{-5}	71	31	0	2.2×10^{-5}	136.9
10^{-6}	102	32	0	1.7×10^{-6}	117.4
10^{-7}	146	42	0	1.8×10^{-7}	113.5
10^{-8}	207	41	0	1.9×10^{-8}	102.1
10^{-9}	298	33	0	2.0×10^{-9}	94.2

Table 9. Results of DLMP6(5) over the problem E2 using the new step-size-change algorithm.

TOL	Accepted Steps	Rejected Steps	Extended Steps	End Point Error	Efficiency (4)
10^{-4}	51	4	14	5.7×10^{-5}	97.3
10^{-5}	73	6	16	3.2×10^{-5}	123.9
10^{-6}	104	0	21	2.4×10^{-6}	105.7
10^{-7}	149	3	24	1.7×10^{-7}	97.4
10^{-8}	210	4	19	1.4×10^{-8}	88.1
10^{-9}	302	1	19	1.7×10^{-9}	86.2

Arenstorf orbit

A fascinating orbit displays a spacecraft's steady journey around the Earth and Moon [1] (p. 297). This can be given in the form of (1) as follows

$$\begin{aligned} y_1' &= y_3, \\ y_2' &= y_4, \\ y_3' &= -\psi' \frac{\psi \cos x + y_1}{P_1} + \psi \frac{\psi' \cos x - y_1}{P_2}, \\ y_4' &= -\psi' \frac{\psi \sin x + y_2}{P_1} + \psi \frac{\psi' \sin x - y_2}{P_2}, \end{aligned}$$

with

$$\begin{aligned} P_1 &= \left((y_1 + \psi \cos x)^2 + (y_2 + \psi \sin x)^2\right)^{1.5}, \\ P_2 &= \left((y_1 - \psi' \cos x)^2 + (y_2 - \psi' \sin x)^2\right)^{1.5}, \\ \psi &= 0.012277471, \psi' = 0.987722529, \end{aligned}$$

and initial values

$$y_1(0) = 0.994,\ y_3(0) = 0,\ y_2(0) = 0,\ y_4(0) = -1.007585106379082,$$

and the solution is periodic with period $x_A = 17.0652165601579625589$. We solved the Arenstorf problem and recorded the cost (i.e., the stages used) and the errors observed for x_A.

The corresponding results are given in Tables 10 and 11.

Interpreting the results, we observe about 28%-29% improvement in the efficiency for the problems and accuracies chosen. This can be found by dividing the corresponding values in the last column of Tables 4–11: e.g., for problem D4 and $TOL = 10^{-4}$, we observe that the standard algorithm ((2) and (3)) is about $\frac{213.6}{150.3} = 42.1\%$ more expensive than the new algorithm. In all 24 runs, we experience better efficiency with the new algorithm. Another interesting issue is that the error is decreasing, since after step rejection, we apply locally a seventh-order approximation.

The new algorithm proposed in the paper could be applied to fractional-order differential equations after some proper modifications [13,14].

Table 10. Results of DLMP6(5) over the Arenstorf orbit using the standard step-size-change algorithm.

TOL	Accepted Steps	Rejected Steps	Extended Steps	End Point Error	Efficiency (4)
10^{-4}	61	23	0	5.4×10^{-1}	605.9
10^{-5}	81	29	0	2.6×10^{-1}	704.6
10^{-6}	111	42	0	3.4×10^{-3}	474.6
10^{-7}	155	49	0	4.4×10^{-4}	450.0
10^{-8}	219	47	0	3.6×10^{-5}	387.4
10^{-9}	315	44	0	4.3×10^{-6}	366.6

Table 11. Results of DLMP6(5) over the Arenstorf orbit using the new step-size-change algorithm.

TOL	Accepted Steps	Rejected Steps	Extended Steps	End Point Error	Efficiency (4)
10^{-4}	62	16	5	6.6×10^{-1}	595.8
10^{-5}	81	16	7	2.7×10^{-1}	640.2
10^{-6}	113	12	18	5.1×10^{-4}	297.6
10^{-7}	157	0	26	1.6×10^{-4}	310.4
10^{-8}	225	0	30	9.0×10^{-6}	272.7
10^{-9}	320	0	25	3.0×10^{-6}	316.9

5. Conclusions

A new step-control algorithm was presented in this article that, after the rejection of a Runge-Kutta step, avoids the unnecessary waste of the already-made computations. Numerical results on problems with a high percentage of rejected steps show the clear advantages of the new step-size policy.

Author Contributions: Conceptualization, V.N.K., R.V.F., T.V.K., T.E.S. and C.T.; Methodology, T.E.S. and C.T.; Software, T.E.S. and C.T.; Validation, V.N.K., R.V.F., T.V.K., T.E.S. and C.T.; Formal analysis, T.E.S. and C.T.; Investigation, V.N.K., R.V.F., T.V.K., T.E.S. and C.T.; Resources, T.E.S. and C.T.; Data curation, V.N.K., R.V.F., T.V.K., T.E.S. and C.T.; Writing—original draft, C.T.; Writing—review & editing, T.E.S. and C.T.; Visualization, V.N.K., R.V.F., T.V.K., T.E.S. and C.T.; Supervision, T.E.S.; Project administration, T.E.S.; Funding acquisition, V.N.K. All authors contributed equally. All authors have read and agreed to the published version of the manuscript.

Funding: The research was supported by a Mega Grant from the Government of the Russian Federation within the framework of federal project No. 075-15-2021-584.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The coefficients of the method presented can be retrieved from http://users.uoa.gr/~tsitourasc/dlmp65ext.m (accessed on 3 June 2023).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Hairer, E.; Norsett, S.P.; Wanner, G. *Solving Ordinary Differential Equations I: Nonstiff Problems*; Springer: Berlin/Heidelberg, Germany, 1987.
2. Fehlberg, C. *Classical Fifth-, Sixth-, Seventh-, and Eighth-Order Runge-Kutta Formulas with Stepsize Control*; NASA Technical Report-287; National Aeronautics and Space Administration (NASA): Washington, DC, USA, 1969.
3. Prince, P.J.; Dormand, J.R. High order embedded Runge-Kutta Formulae. *J. Comput. Appl. Math.* **1981**, *7*, 67–75. [CrossRef]
4. Dormand, J.R.; Prince, P.J. A family of embedded Runge-Kutta formulae. *J. Comput. Appl. Math.* **1980**, *6*, 19–26. [CrossRef]
5. Tsitouras, C. A parameter study of explicit Runge-Kutta pairs of orders 6(5). *Appl. Math. Lett.* **1998**, *11*, 65–69. [CrossRef]
6. Dormand, J.R.; Lockyer, M.A.; McGorrigan, N.E.; Prince, P.J. Global error estimation with Runge-Kutta triples. *Comput. Math. Appl.* **1989**, *18*, 835–846. [CrossRef]
7. Wolfram Research, Inc. *Mathematica*; Version 11.3.0; Wolfram Research, Inc.: Champaign, IL, USA, 2018.
8. Verner, J.H. Differentiable Interpolants for high-Order Runge-Kutta Methods. *SIAM J. Numer. Anal.* **1993**, *30*, 1446–1466. [CrossRef]
9. Tsitouras, C. A tenth order symplectic Runge–Kutta–Nyström method. *Celest. Mech. Dyn. Astron.* **1999**, 74, 223–230. [CrossRef]

10. Enright, W.; Pryce, J.D. Two FORTRAN packages for assessing initial value methods. *ACM Trans. Math. Softw.* **1987**, *13*, 1–27. [CrossRef]
11. Kovalnogov, V.N.; Fedorov, R.V.; Chukalin, A.V.; Simos, T.E.; Tsitouras, C. Evolutionary Derivation of Runge–Kutta Pairs of Orders 5(4) Specially Tuned for Problems with Periodic Solutions. *Mathematics* **2021**, *9*, 2306. [CrossRef]
12. Shampine, L.F. Some practical Runge-Kutta formulas. *Math. Comp.* **1986**, *46*, 135–150. [CrossRef]
13. Zhao, K. Solvability and GUH-stability of a nonlinear CF-fractional coupled Laplacian equations. *AIMS Math.* **2023**, *8*, 13351–13367. [CrossRef]
14. Zhao, K. Existence and UH-stability of integral boundary problem for a class of nonlinear higher-order Hadamard fractional Langevin equation via Mittag-Leffler functions. *Filomat* **2023**, *37*, 1053–1063.