


Article

Runge–Kutta Pairs of Orders 6(5) with Coefficients Trained to Perform Best on Classical Orbits

Yu-Cheng Shen ¹, Chia-Liang Lin ², Theodore E. Simos ^{3,4,5,6,7,*}  and Charalampos Tsitouras ⁸ ¹ Department of Preschool Education, School of Educational Sciences, Huaiyin Campus, Huaiyin Normal University, Huaian City 223300, China; roscoeshen@gmail.com² Department of Visual Communications, School of Arts, Huzhou University, Huzhou 313000, China; tronic1983@gmail.com³ College of Applied Mathematics, Chengdu University of Information Technology, Chengdu 610225, China⁴ Scientific and Educational Center “Digital Industry”, South Ural State University, 76 Lenin Ave., 454 080 Chelyabinsk, Russia⁵ Department of Medical Research, China Medical University Hospital, China Medical University, Taichung City 40402, Taiwan⁶ Data Recovery Key Laboratory of Sichuan Province, Neijiang Normal University, Neijiang 641100, China⁷ Section of Mathematics, Department of Civil Engineering, Democritus University of Thrace, 67100 Xanthi, Greece⁸ General Department, GR34-400 Euripus Campus, National & Kapodistrian University of Athens, 15772 Athens, Greece; tsitourasc@uoa.gr

* Correspondence: tsimos.conf@gmail.com

Abstract: We consider a family of explicit Runge–Kutta pairs of orders six and five without any additional property (reduced truncation errors, Hamiltonian preservation, symplecticness, etc.). This family offers five parameters that someone chooses freely. Then, we train them in order for the presented method to furnish the best results on a couple of Kepler orbits, a certain interval and tolerance. Consequently, we observe an efficient performance on a wide range of orbital problems (i.e., Kepler for a variety of eccentricities, perturbed Kepler with various disturbances, Arenstorf and Pleiades). About 1.8 digits of accuracy is gained on average over conventional pairs, which is truly remarkable for methods coming from the same family and order.

Keywords: initial value problem; Kepler-type orbits; Runge–Kutta; differential evolution

MSC: 65L05; 65L06; 90C26; 90C30



Citation: Shen, Y.-C.; Lin, C.-L.; Simos, T.E.; Tsitouras, C. Runge–Kutta Pairs of Orders 6(5) with Coefficients Trained to Perform Best on Classical Orbits. *Mathematics* **2021**, *9*, 1342. <https://doi.org/10.3390/math9121342>

Academic Editor: Arsen Palestini

Received: 26 May 2021

Accepted: 8 June 2021

Published: 10 June 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. 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

The initial value problem (IVP) is

$$y' = f(x, y), y(x_0) = y_0 \quad (1)$$

with $x_0 \in \mathbb{R}$, $y, y' \in \mathbb{R}^m$ and $f : \mathbb{R} \times \mathbb{R}^m \rightarrow \mathbb{R}^m$.

Runge–Kutta (RK) pairs are amongst the most popular numerical methods for addressing (1). They are characterized by the following Butcher tableau [1,2]:

$$\begin{array}{c|c} c & A \\ \hline & b \\ & \widehat{b} \end{array}$$

with $b^T, \widehat{b}^T, c \in \mathbb{R}^s$ and $A \in \mathbb{R}^{s \times s}$. Then, the method shares s stages, and when $c_1 = 0$ and A is strictly lower triangular, it is evaluated explicitly. The approximated solution steps from (x_n, y_n) to $x_{n+1} = x_n + h_n$ by producing two estimations for $y(x_{n+1})$. Namely, y_{n+1} and \widehat{y}_{n+1} , given by

$$y_{n+1} = y_n + h_n \sum_{i=1}^s b_i f_i$$

and

$$y_{n+1} = y_n + h_n \sum_{i=1}^s \hat{b}_i f_i$$

with

$$f_i = f(x_n + c_i h_n, y_n + h_n \sum_{j=1}^{i-1} a_{ij} f_j),$$

for $i = 1, 2, \dots, s$. These two approximations y_{n+1} and \hat{y}_{n+1} are of algebraic orders p and $q < p$ respectively. Thus, a local error estimation

$$\epsilon_n = h_n^{p-q-1} \cdot \|y_{n+1} - \hat{y}_{n+1}\|$$

is formed in every step and is combined in an algorithm for changing the step size:

$$h_{n+1} = 0.9 \cdot h_n \cdot \left(\frac{t}{\epsilon_n}\right)^{1/p},$$

where t is a tolerance given by the user. When $\epsilon_n < t$, the above formula is used for the new step forward. In reverse, we also use it, but the solution is not advanced and h_{n+1} is a new version of h_n . Details can be found in [3]. As an abbreviation, these methods are named RKp(q) pairs.

Runge–Kutta methods were introduced back in the late 19th century [4,5]. After 1960, RK pairs appeared. Fehlberg gave the first celebrated such pairs of orders 5(4), 6(5) and 8(7) [6,7]. Dormand and Prince followed in the early 1980s [8,9]. Our group has also presented a series of successful RK pairs [10–13].

Runge–Kutta pairs are suited for efficiently solving almost every non-stiff problem of the form (1). The variety of pairs is explained by the accuracy required. Thus, when less accuracy is required, the lowest RK pairs are more efficient. In contrast, for stringent accuracies at quadruple precision, a high-order pair should be chosen [14].

Here we focus on RK6(5) pairs, which are preferred for moderate to higher accuracies. We are especially interested in problems (1) that resemble Kepler-like orbits. Thus, we will propose a particular RK6(5) pair for addressing this type of problem.

2. Producing Runge–Kutta Pairs of Orders 6(5) and Training Their Coefficients

Runge–Kutta pairs of orders six and five are amongst the most frequently used. The coefficients have to satisfy 54 order conditions. Thus, families of solutions have been discovered over the years. Here we chose the Verner-DLMP [15,16] family, which has the advantage of being solved linearly. Then, we freely chose the coefficients c_2, c_4, c_5, c_6, c_7 and \hat{b}_9 . Pairs from this family have been proven to perform most efficiently in various classes of problems [17].

We proceed by explicitly evaluating the remaining coefficients. The algorithm is discussed in [10] and is given as a Mathematica [18] package in the Appendix A.

Although $s = 9$, the family spends only eight stages per step since the ninth stage is used as the first stage of the next step. This property is called FSAL (first stage as last).

The next question to be answered is regarding how to select the free parameters. Traditionally we try to minimize the norm of the principal term of the local truncation error. That is, the coefficients of h^7 in the residual of Taylor error expansions corresponding to the sixth-order method of the underlying RK pair.

We intend to derive a particular RK6(5) pair belonging to the family of interest here. The resulting pair has to perform best on Kepler orbits and other problems of this nature. Thus, we concentrate on the particular orbit

$$\begin{aligned} {}^1y' &= {}^3y, \\ {}^2y' &= {}^4y, \\ {}^3y' &= -\frac{{}^1y}{\left(\sqrt{{}^1y^2 + {}^2y^2}\right)^3}, \\ {}^4y' &= -\frac{{}^2y}{\left(\sqrt{{}^1y^2 + {}^2y^2}\right)^3}, \end{aligned}$$

with $x \in [0, 10\pi]$, $y(0) = \left[1 - ecc, 0, 0, \sqrt{\frac{1+ecc}{1-ecc}}\right]^T$ and the theoretical solution

$${}^1y(x) = \cos(v) - ecc, \quad {}^2y(x) = \sin(v)\sqrt{1 - ecc^2}.$$

In the above, $v = ecc \cdot \sin(u) + x$, ecc is the eccentricity, and the components of y are denoted by the left superscript. They should not be confused with $y_1 = [{}^1y_1, {}^2y_1, {}^3y_1, {}^4y_1]^T$, $y_2 = [{}^1y_2, {}^2y_2, {}^3y_2, {}^4y_2]^T$, y_3, \dots , which represent the vectors approximating the solution at x_1, x_2, x_3, \dots .

This problem can be solved with an RK6(5) pair from the family we are interested in here. After a certain run, we recorded the number fev of function evaluations (stages) needed and the global error ge observed over the mesh (grid) in the interval of integration. Then, we formed the efficiency measure

$$u = fev \cdot ge^{1/6}. \quad (2)$$

Running DLMP6(5) twice for Kepler we obtained the efficiency measures \hat{u}_1 and \hat{u}_2 as reported in Table 1. For example, we needed 1121 stages in order to achieve a global error of $2.14 \cdot 10^{-6}$ when running Kepler for $ecc = 0$, $x_{end} = 10\pi$ and $tol = 10^{-7}$. Thus, we obtained $\hat{u}_1 = 1123 \cdot 2.14 \cdot 10^{-6} \approx 127.22$, as reported in Table 1. Analogously, for a second run with $ecc = 0.6$, $x_{end} = 20\pi$ and $tol = 10^{-11}$ we observed $\hat{u}_2 = 833.27$.

Let us suppose that any new pair NEW6(5) furnishes corresponding efficiency measures \tilde{u}_1 and \tilde{u}_2 for the same runs. Then, as a fitness function we form the sum

$$u = \frac{\hat{u}_1}{\tilde{u}_1} + \frac{\hat{u}_2}{\tilde{u}_2},$$

and try to maximize it. That is, the fitness function is actually two whole runs of initial value problems. The value u changes according to the selection of the free parameters c_2, c_4, c_5, c_6, c_7 and \hat{b}_9 .

The original idea is based on [19]. For the minimization of u we used the differential evolution technique [20]. We have already tried this approach and obtained some interesting results in producing Numerov-type methods for integrating orbits [21]. In this latter work we trained the coefficients of a Numerov-type method on a Kepler orbit. We observed very pleasant results over a set of Kepler orbits as well as other known orbital problems.

Table 1. Efficiency measures for both runs and pairs.

	ecc	x_{end}	tol	DLMP6(5)	NEW6(5)	Ratio \hat{u}/\tilde{u}
First run	0	10π	10^{-7}	$\hat{u}_1 = 127.22$	$\tilde{u}_1 = 50.64$	2.51
Second run	0.6	20π	10^{-11}	$\hat{u}_2 = 833.27$	$\tilde{u}_2 = 386.64$	2.16

The optimization furnished six values for the free parameters. They are included with the rest of the coefficients in the resulting pair NEW6(5) presented in Table 2.

Table 2. Coefficients of the proposed NEW6(5) pair, accurate for double precision computations.

$c_2 = 0.173146279530013,$	$c_3 = 0.163620769891761,$	$c_4 = 0.245431154837642,$
$c_5 = 0.452502877641229,$	$c_6 = 0.902924768667267,$	$c_7 = 0.8101151362080617,$
$c_1 = 0, c_8 = c_9 = 1,$	$b_1 = 0.0794169052387116,$	$b_2 = b_3 = 0,$
$b_4 = 0.320063598496390,$	$b_5 = 0.179217292937057,$	$b_6 = -0.2872484367615202,$
$b_7 = 0.573172758378662,$	$b_8 = 0.135377881710699,$	$b_9 = 0$
$\hat{b}_1 = 0.148854176113754,$	$\hat{b}_2 = \hat{b}_3 = 0,$	$\hat{b}_4 = 0.291009331941132,$
$\hat{b}_5 = 0.229278395578701,$	$\hat{b}_6 = -0.1155397766857130,$	$\hat{b}_7 = 0.429687174664803,$
$\hat{b}_8 = 0.0167106983873234,$	$\hat{b}_9 = 0.064345053530889,$	
$a_{21} = 0.173146279530013,$	$a_{31} = 0.0863111204651556,$	$a_{32} = 0.077309649426606,$
$a_{41} = 0.061357788709411,$	$a_{42} = 0,$	$a_{43} = 0.184073366128232,$
$a_{51} = 0.178735636864969,$	$a_{52} = 0,$	$a_{53} = -0.430121641642955,$
$a_{54} = 0.703888882419215,$	$a_{61} = -0.3492563988707026,$	$a_{62} = 0,$
$a_{63} = 4.2286674995349015,$	$a_{64} = -5.131590895887595,$	$a_{65} = 2.155104563890663,$
$a_{71} = -0.004184382566843,$	$a_{72} = 0,$	$a_{73} = 1.062724280290705,$
$a_{74} = -1.188530484293243,$	$a_{75} = 0.8944565948851806,$	$a_{76} = 0.045649127892262,$
$a_{81} = -0.518393300452978,$	$a_{82} = 0,$	$a_{83} = 4.607278279969559,$
$a_{84} = -5.004120306973807,$	$a_{85} = 1.510536380616834,$	$a_{86} = -0.399249451366671,$
$a_{87} = 0.803948398207063,$	$a_{9j} = b_j, j = 1, 2, \dots, 8.$	

Interpreting Table 1, we observe that DLMP6(5) was 151% and 116% more expensive than NEW6(5) for the first and second run, respectively. The norm of the principal truncation error coefficients was $\|T^{(7)}\|_2 \approx 2.64 \cdot 10^{-4}$, which is much larger than the corresponding value $\|T^{(7)}\|_2 \approx 4.37 \cdot 10^{-5}$ for DLMP6(5). The interval of absolute stability for the new pair was $(-4.24, 0]$ while for DLMP6(5) it was $(-4.21, 0]$.

In conclusion, no extra property seemed to hold. The pair given in Table 2 does not possess any interesting properties. It is difficult to believe a special performance could be obtained after seeing its traditional characteristics.

3. Numerical Tests

We tested the following pairs chosen from the family studied above.

1. The DLMP6(5) pair, given in [15].
2. NEW6(5), presented here.

DLMP6(5) is the best representative of conventional RK pairs. Everything else presented until now is hardly more efficient [10]. Both pairs were run for tolerances of $10^{-5}, 10^{-6}, \dots, 10^{-11}$, and the efficiency measures (2) were recorded for each one. We set NEW6(5) as the reference pair. Then we divided each efficiency measure of DLMP6(5) with the corresponding efficiency measure of NEW6(5). Numbers greater than 1 indicate that NEW6(5) is more efficient. Thus, we can interpret the number 1.1 as DLMP6(5) being 0.1 = 10% more expensive than NEW6(5) while an entry of 2 means that DLMP6(5) is 100% more expensive (i.e., has twice the cost for achieving the same accuracy).

The problems we tested were as follows.

1. The Kepler problem

This problem is explained above. We ran it for five different eccentricities (i.e., $ecc = 0, 0.2, 0.4, 0.6, 0.8$), while we recorded the efficiency measures using the endpoint errors for $x_{end} = 10\pi$ and $x_{end} = 20\pi$.

The efficiency measure ratios of DLMP6(5) vs. NEW6(5) for Kepler are presented in Tables 3 and 4.

Table 3. Efficiency measure ratios of DLMP6(5) vs. NEW6(5) for Kepler in $[0, 10\pi]$.

$\frac{tol-\>}{ecc}$	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	Mean
0	1.90	2.36	3.66	3.06	2.46	3.18	3.59	2.89
0.2	1.15	1.11	1.14	1.01	1.07	1.46	2.07	1.29
0.4	1.09	1.10	1.11	1.51	0.80	1.27	1.71	1.23
0.6	1.49	1.49	1.86	1.27	1.18	1.43	1.73	1.49
0.8	1.13	1.17	1.32	1.21	1.36	1.08	2.73	1.43

Table 4. Efficiency measure ratios of DLMP6(5) vs. NEW6(5) for Kepler in $[0, 20\pi]$.

$\frac{tol-\>}{ecc}$	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	Mean
0	1.87	2.34	2.59	2.35	2.67	3.36	3.36	2.65
0.2	1.22	1.27	1.33	1.24	1.29	1.67	2.39	1.49
0.4	1.31	1.55	1.21	0.96	1.08	2.36	2.27	1.54
0.6	1.38	1.32	1.17	1.10	0.98	1.14	1.99	1.30
0.8	1.16	1.24	1.85	1.25	1.08	0.94	1.61	1.30

2. The perturbed Kepler

This problem describes the motion of a planet according to Einstein's general relativity theory and the Schwarzschild potential is applied. The equations are:

$$\begin{aligned} {}^1y'' &= -\frac{{}^1y}{\sqrt{{}^1y^2 + {}^2y^2}^3} - (2 + \delta)\delta \frac{{}^1y}{\sqrt{{}^1y^2 + {}^2y^2}^5}, \\ {}^2y'' &= -\frac{{}^2y}{\sqrt{{}^1y^2 + {}^2y^2}^3} - (2 + \delta)\delta \frac{{}^2y}{\sqrt{{}^1y^2 + {}^2y^2}^5}, \end{aligned}$$

and the analytical solution is

$${}^1y = \cos(x + \delta x), \quad {}^2y = \sin(x + \delta x).$$

We transformed this problem into a system of four first-order equations and solved for $x_{end} = 10\pi$ and $x_{end} = 20\pi$. After recording the endpoint errors and the costs, we present the efficiency measures ratios of DLMP6(5) vs. NEW6(5) for perturbed Kepler in Tables 5 and 6.

Table 5. Efficiency measures ratios of DLMP6(5) vs. NEW6(5) for perturbed Kepler in $[0, 10\pi]$.

$\frac{tol-\>}{\delta}$	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	Mean
0.01	1.90	2.48	2.39	2.33	3.04	3.64	3.49	2.75
0.02	1.93	2.69	2.19	2.11	2.58	3.65	3.45	2.66
0.03	1.91	2.76	2.14	2.07	2.50	3.45	3.47	2.61
0.04	1.87	2.63	2.18	2.12	2.53	3.28	3.57	2.60
0.05	1.87	2.40	2.32	2.30	2.47	3.24	3.72	2.62

Table 6. Efficiency measures ratios of DLMP6(5) vs. NEW6(5) for perturbed Kepler in $[0, 20\pi]$.

$\frac{tol-\geq}{\delta}$	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	Mean
0.01	1.88	2.44	2.35	2.18	2.43	3.11	3.34	2.53
0.02	1.86	2.49	2.29	2.14	2.38	3.09	3.36	2.52
0.03	1.85	2.44	2.33	2.17	2.43	3.17	3.39	2.54
0.04	1.83	2.38	2.36	2.17	2.44	3.21	3.43	2.55
0.05	1.82	2.32	2.37	2.16	2.43	3.19	3.44	2.53

3. The Arenstorf orbit

Another interesting orbit describes the stable movement of a spacecraft around Earth and Moon ([22], pg. 129).

$$\begin{aligned} {}^1y'' &= {}^1y + 2 \cdot {}^2y' - \zeta' \cdot \frac{{}^1y + \zeta}{P_1} - \zeta \cdot \frac{{}^1y - \zeta'}{P_2}, \\ {}^2y'' &= {}^2y + 2 \cdot {}^1y' - \zeta' \cdot \frac{{}^2y}{P_1} - \zeta \cdot \frac{{}^2y}{P_2}, \end{aligned}$$

with

$$\begin{aligned} P_1 &= \sqrt{({}^1y + \zeta)^2 + {}^2y^2}, P_2 = \sqrt{({}^1y - \zeta')^2 + {}^2y^2}, \\ \zeta &= 0.012277471, \zeta' = 0.987722529, \end{aligned}$$

initial values

$${}^1y(0) = 0.994, {}^1y'(0) = 0, {}^2y(0) = 0, {}^2y'(0) = -2.00158510637908252,$$

and with $x_A = 17.0652165601579625589$ the solution is periodic.

We also transformed this problem to a system of four first-order equations and solved it to x_A and $2x_A$. After recording the endpoint errors and the costs we present the efficiency measures ratios of DLMP6(5) vs. NEW6(5) for Arenstorf in Table 7.

Table 7. Efficiency measure ratios of DLMP6(5) vs. NEW6(5) for Arenstorf.

$\frac{tol-\geq}{x_{end}}$	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	Mean
x_A	1.13	1.42	1.89	1.34	0.90	1.51	1.71	1.41
$2x_A$	0.96	1.82	1.64	1.07	1.09	1.62	1.58	1.40

4. The Pleiades

Finally, we considered the problem “Pleiades” as given in ([22], pg. 245).

$${}^iy'' = \sum_{i \neq j} \frac{\mu_j ({}^iy - {}^iy)}{\rho_{ij}}, \quad {}^iz'' = \sum_{i \neq j} \frac{\mu_j ({}^iz - {}^iz)}{\rho_{ij}},$$

with

$$\rho_{ij} = \sqrt{({}^iy - {}^iy)^2 + ({}^iz - {}^iz)^2}, i, j = 1, \dots, 7.$$

The initial values are

$$\begin{aligned} {}^1y(0) &= 3, {}^2y(0) = 3, {}^3y(0) = -1, {}^4y(0) = -3, {}^5y(0) = 2, {}^6y(0) = -2, {}^7y(0) = 2, \\ {}^1z(0) &= 3, {}^2z(0) = -3, {}^3z(0) = 2, {}^4z(0) = 0, {}^5z(0) = 0, {}^6z(0) = -4, {}^7z(0) = 4, \end{aligned}$$

$${}^1y'(0) = 0, {}^2y'(0) = 0, {}^3y'(0) = 0, {}^4y'(0) = 0, {}^5y'(0) = 0, {}^6y'(0) = 1.75, {}^7y'(0) = -1.5,$$

$${}^1z'(0) = 0, {}^2z'(0) = 0, {}^3z'(0) = 0, {}^4z'(0) = -1.25, {}^5z'(0) = 1, {}^6z'(0) = 0, {}^7z'(0) = 0,$$

We again transformed this problem to a system of fourteen first-order equations and solved it to $x_{end} = 3$ and 4. We recorded the endpoint errors after we estimated the solution there by a very accurate integration using Mathematica and quadruple precision. The efficiency measure ratios of DLMP6(5) vs. NEW6(5) for Pleiades can be found in Table 8.

Table 8. Efficiency measures ratios of DLMP6(5) vs. NEW6(5) for Pleiades.

$\frac{tol-\geq}{x_{end}}$	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	Mean
3	1.12	1.02	0.97	1.03	0.89	1.10	1.63	1.11
4	1.13	1.03	0.97	0.96	0.94	1.15	1.64	1.12

We estimated 168 (i.e., 12 problems times 7 tolerances times two end points) efficiency measures for each pair. In average we observed a ratio of 1.98, meaning that DLMP6(5) is about 98% more expensive! This is quite remarkable since a great deal of effort has been put over the years towards achieving 10–20% efficiency [23,24]. In reverse, this means that about $\log_{10} 1.98^6 \approx 1.8$ digits were gained on average at the same costs.

4. Conclusions

This paper is concerned with training the coefficients of a Runge–Kutta pair for addressing a certain kind of problem. We concentrated on problems with Kepler-type orbits and an extensively studied family of Runge–Kutta pairs of orders six and five. After optimizing the free parameters (coefficients) in a couple of runs on Kepler orbits, we concluded to a certain pair. This pair was found to outperform other representatives from this family in a wide range of relevant problems.

Author Contributions: All authors contributed equally. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

Appendix A

The following Mathematica package implements the algorithm producing the coefficients of the method for double precision. In the input we provide the free coefficients c_2, c_4, c_5, c_6, c_7 and \hat{b}_9 . In the output we get four vectors, namely b, \hat{b}, c and the matrix A .

```
NEW65[cc2_, cc4_, cc5_, cc6_, cc7_, bbb9_] :=
Module[{b, a, c, bb, vandh, vandl, ac, ac2, cc, ii, ba, cond, soh, b1, b4, b5, b6,
b7, b8, bb1, bb4, bb5, bb6, bb7, bb8, bb9, c2, c3, c4, c5, c6, c7, a21, a31,
a32, a41, a43, a51, a53, a54, a61, a63, a64, a65, a71, a73, a74, a75, a76,
a81, a83, a84, a85, a86, a87, so3, so5, sol, so6, so7, so8},
c2 = Rationalize[cc2, 10^-16]; c4 = Rationalize[cc4, 10^-16];
c5 = Rationalize[cc5, 10^-16]; c6 = Rationalize[cc6, 10^-16];
c7 = Rationalize[cc7, 10^-16]; bb9 = Rationalize[bbb9, 10^-16];
b={b1,0,0,b4,b5,b6,b7,b8,0};
a={{0,0,0,0,0,0,0,0,0},
{a21,0,0,0,0,0,0,0,0},
```

```

{a31,a32,0,0,0,0,0,0,0},
{a41,0,a43,0,0,0,0,0,0},
{a51,0,a53,a54,0,0,0,0,0},
{a61,0,a63,a64,a65,0,0,0,0},
{a71,0,a73,a74,a75,a76,0,0,0},
{a81,0,a83,a84,a85,a86,a87,0,0},
{b1,0,0,b4,b5,b6,b7,b8,0};
c={0,c2,c3,c4,c5,c6,c7,1,1};
bb={bb1,0,0,bb4,bb5,bb6,bb7,bb8,bb9};
vandh={b.c==1/2,b.c^2==1/3,b.c^3==1/4,b.c^4==1/5,b.c^5==1/6};
vandl={bb.c==1/2,bb.c^2==1/3,bb.c^3==1/4,bb.c^4==1/5};
ac=a.c-c^2/2;
ac2=a.c^2-c^3/3; cc=DiagonalMatrix[c]; ii=IdentityMatrix[9];
ba=b.(a+cc-1*ii);
cond=
{b.(cc-1*ii).a.(cc-c4*ii).(cc-c5*ii).c-Integrate[(x-1)*Integrate[(x-c5)*(x-c4)*x,
{x,0,x}],{x,0,1}],
b.(cc-ii).a.(cc-c4*ii).(cc-c5*ii).c-Integrate[(x-1)*Integrate[(x-c4)*(x-c5)*x,
{x,0,x}],{x,0,1}] /.{a43->0,a53->0,a63->0,a73->0},
bb.a.(cc-c5*ii).(cc-c4*ii).c-Integrate[Integrate[(x-c5)*(x-c4)*x,{x,0,x}],
{x,0,1}]];
(* start procedure *)
soh=Solve[vandh,{b4,b5,b6,b7,b8}];
b4=Together[soh[[1,1,2]]];
b5=Together[soh[[1,2,2]]];
b6=Together[soh[[1,3,2]]];
b7=Together[soh[[1,4,2]]];
b8=Together[soh[[1,5,2]]];
sol=Solve[Join[{(bb.a)[[3]]==0},vandl],{bb4,bb5,bb6,bb7,bb8}];
bb4=Together[sol[[1,1,2]]];
bb5=Together[sol[[1,2,2]]];
bb6=Together[sol[[1,3,2]]];
bb7=Together[sol[[1,4,2]]];
bb8=Together[sol[[1,5,2]]];
c3=2/3*c4; a43=c4^2/(2*c3); a32=c3^2/(2*c2);
so5=Solve[{ac[[5]]==0,ac2[[5]]==0},{a53,a54}];
a53=Together[so5[[1,1,2]]];a54=Together[so5[[1,2,2]]];
a87=Together[Solve[{ba[[7]]==0},{a87}][[1,1,2]]];
a76=Together[Solve[{cond[[2]]==0},{a76}][[1,1,2]]];
a86=Together[Solve[{ba[[6]]==0},{a86}][[1,1,2]]];
ac=Together[ac]; ac2=Together[ac2];
ba=Together[ba]; cond=Together[cond];
so3=Solve[{ba[[3]]==0,cond[[1]]==0,cond[[3]]==0},{a63,a73,a83}];
a63=Together[so3[[1,1,2]]];
a73=Together[so3[[1,2,2]]];
a83=Together[so3[[1,3,2]]];
so6=Solve[{ac[[6]]==0,ac2[[6]]==0},{a64,a65}];
a64=Together[so6[[1,1,2]]];a65=Together[so6[[1,2,2]]];
so7=Solve[{ac[[7]]==0,ac2[[7]]==0},{a74,a75}];
a74=Together[so7[[1,1,2]]];a75=Together[so7[[1,2,2]]];
so8=Solve[{ac[[8]]==0,ac2[[8]]==0},{a84,a85}];
a84=Together[so8[[1,1,2]]];a85=Together[so8[[1,2,2]]];
b1=1-b4-b5-b6-b7-b8;
bb1=1-bb4-bb5-bb6-bb7-bb8;
a21=c2;
a31=c3-a32;
a41=c4-a43;
a51=c5-a53-a54;
a61=c6-a63-a64-a65;

```



```

a71=c7-a73-a74-a75-a76;
a81=1-a83-a84-a85-a86-a87;
Return[SetAccuracy[{b, bb, c, a}, 16] // Chop]]

```

References

- Butcher, J.C. On Runge-Kutta processes of high order. *J. Austral. Math. Soc.* **1964**, *4*, 179–194. [\[CrossRef\]](#)
- Butcher, J.C. *Numerical Methods for Ordinary Differential Equations*; John Wiley & Sons: Chichester, UK, 2003.
- Tsitouras, C.; Papakostas, S.N. Cheap Error Estimation for Runge-Kutta pairs. *SIAM J. Sci. Comput.* **1999**, *20*, 2067–2088. [\[CrossRef\]](#)
- Runge, C. Ueber die numerische Auflösung von Differentialgleichungen. *Math. Ann.* **1895**, *46*, 167–178. [\[CrossRef\]](#)
- Kutta, W. Beitrag zur näherungsweise Integration von Differentialgleichungen. *Z. Math. Phys.* **1901**, *46*, 435–453.
- Fehlberg, E. Klassische Runge-Kutta-Formeln fünfter und siebenter Ordnung mit Schrittweiten-Kontrolle. *Computing* **1969**, *4*, 93–106. [\[CrossRef\]](#)
- Fehlberg, E. Klassische Runge-Kutta-Formeln vierter und niedrigerer Ordnung mit Schrittweiten-Kontrolle und ihre Anwendung auf Wärmeleitungsprobleme. *Computing* **1970**, *6*, 61–71. [\[CrossRef\]](#)
- Dormand, J.R.; Prince, P.J. A family of embedded Runge-Kutta formulae. *J. Comput. Appl. Math.* **1980**, *6*, 19–26. [\[CrossRef\]](#)
- Prince, P.J.; Dormand, J.R. High order embedded Runge-Kutta formulae. *J. Comput. Appl. Math.* **1981**, *7*, 67–75. [\[CrossRef\]](#)
- Tsitouras, C. A parameter study of explicit Runge-Kutta pairs of orders 6(5). *Appl. Math. Lett.* **1998**, *11*, 65–69. [\[CrossRef\]](#)
- Famelis, I.T.; Papakostas, S.N.; Tsitouras, C. Symbolic derivation of Runge-Kutta order conditions. *J. Symbolic Comput.* **2004**, *37*, 311–327. [\[CrossRef\]](#)
- Tsitouras, C. Runge-Kutta pairs of orders 5(4) satisfying only the first column simplifying assumption. *Comput. Math. Appl.* **2011**, *62*, 770–775. [\[CrossRef\]](#)
- Medvedev, M.A.; Simos, T.E.; Tsitouras, C. Fitted modifications of Runge-Kutta pairs of orders 6(5). *Math. Meth. Appl. Sci.* **2018**, *41*, 6184–6194. [\[CrossRef\]](#)
- Tsitouras, C. Optimized explicit Runge-Kutta pair of orders 9(8). *Appl. Numer. Math.* **2001**, *38*, 121–134. [\[CrossRef\]](#)
- Dormand, J.R.; Lockyer, M.A.; McCorrigan, N.E.; Prince, P.J. Global error estimation with Runge-Kutta triples. *Comput. Math. Appl.* **1989**, *18*, 835–846. [\[CrossRef\]](#)
- Verner, J.H. Some Runge-Kutta formula pairs. *SIAM J. Numer. Anal.* **1991**, *28*, 496–511. [\[CrossRef\]](#)
- Simos, T.E.; Tsitouras, C. Evolutionary derivation of Runge-Kutta pairs for addressing inhomogeneous linear problems. *Numer. Algor.* **2020**. [\[CrossRef\]](#)
- Wolfram Research, Inc. *Mathematica, Version 11.1*; Wolfram Research, Inc.: Champaign, IL, USA, 2017.
- Tsitouras, C. Neural Networks With Multidimensional Transfer Functions. *IEEE Trans. Neural Netw.* **2002**, *13*, 222–228. [\[CrossRef\]](#)
- Storn, R.; Price, K. Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. *J. Glob. Optim.* **1997**, *11*, 341–359. [\[CrossRef\]](#)
- Liu, C.; Hsu, C.W.; Tsitouras, C.; Simos, T.E. Hybrid Numerov-type methods with coefficients trained to perform better on classical orbits. *Bull. Malays. Math. Sci. Soc.* **2019**, *42*, 2119–2134. [\[CrossRef\]](#)
- Hairer, E.; Nørsett, S.P.; Wanner, G. *Solving Ordinary Differential Equations I, Nonstiff Problems*, 2nd ed.; Springer: Berlin, Germany, 1993.
- Papageorgiou, G.; Papakostas, S.N.; Tsitouras, C. A general family of explicit Runge-Kutta pairs of orders 6(5). *SIAM J. Numer. Anal.* **1996**, *33*, 917–936.
- Papakostas, S.N.; Tsitouras, C. High phase-lag order Runge-Kutta and Nyström pairs. *SIAM J. Sci. Comput.* **1999**, *21*, 747–763. [\[CrossRef\]](#)