Runge–Kutta pairs of orders $5(4)$ using the minimal set of simplifying assumptions.

Ch. Tsitouras

TEI of Chalkis, Dept. of Applied Sciences, GR34400, Psahna, Greece.

Abstract. The Runge–Kutta pairs of orders 5 and 4 are the most popular ones in the relevant literature. In order to derive such method we have to solve a system of nonlinear equations for its coefficients. A common practice for achieving this is to admit various simplifying assumptions. These restrict the generality of the solution. Here we proceed using the minimal set of simplifications ever made for the derivation of the coefficients for a pair of these orders. The result is a pair that outperforms other known pairs in the bibliography when tested to standard set of problems of DETEST.

Keywords: Runge–Kutta; Truncation Error; Non Linear Algebraic Systems; Free Parameters.
PACS: 02.60D.Lj; 02.60.Pn

INTRODUCTION

We consider the numerical solution of the non-stiff initial value problem,

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

where the function $f : \Re \times \Re^m \to \Re^m$ is assumed to be as smooth as necessary. Traditionally, explicit embedded Runge-Kutta methods produce an approximation to the solution of (1) only at the end of each step.

The general $s-$stage embedded Runge-Kutta pair of orders $p(p-1)$, for the approximate solution of the problem (1) can be defined by the following Butcher scheme [3, 4]

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

where $A \in \Re^{s\times s}$, is strictly lower triangular, $b^T,\ \hat{b}^T,\ c \in \Re^s$ with $c = A\cdot e$, $e = [1,\ 1, \cdots, 1]^T \in \Re^s$. The vectors $\hat{b},\ b$ define the coefficients of the $(p-1)-$th and $p-$th order approximations respectively.

Starting with a given value $y(x_0) = y_0$, this method produces approximations at the mesh points $x_0 < x_1 < x_2 < \cdots < x_f$. Throughout this paper, we assume that local extrapolation is applied, hence the integration is advanced using the $p-$th order approximation. For estimating the error, two approximations are evaluated at each step x_n to $x_{n+1} = x_n + h_n$. These are:

$$\hat{y}_{n+1} = y_n + h_n \sum_{j=1}^{s} \hat{b}_j f_j \text{ and } y_{n+1} = y_n + h_n \sum_{j=1}^{s} b_j f_j,$$

where

$$f_i = f(x_n + c_i h_n,\ y_n + h_n \sum_{j=1}^{i-1} a_{ij} f_j),\ i = 1,2,\cdots,s. \tag{2}$$

The local error estimate $E_n = \|y_n - \hat{y}_n\|$ of the $(p-1)-$th order Runge-Kutta pair is used for the automatic selection of the step size. Given a Tolerance $TOL > E_n$, the algorithm

$$h_{n+1} = 0.9\cdot h_n \cdot (\frac{TOL}{E_n})^{\frac{1}{p}}$$

furnishes the next step length. In case $TOL < E_n$ then we reject the current step and try again with the left side of above formula being h_n.

In case that $c_s = 1$, $a_{s,j} = b_j$ for $j = 1,2,\cdots,s-1$ and $b_s = 0 \neq \hat{b}_s$ then the **F**irst stage of each step is the **S**ame **A**s the **L**ast one of the previous stage. This device was possibly first used in [7, pg. 22] and it is called FSAL. The pair shares effectively only $s-1$ stages per step then.

Let $y_n(x)$ be the solution of the local initial value problem

$$y'(x) = f(x, y_n(x)),\ x \geq x_n,\ y_n(x_n) = y_n$$

Then E_{n+1} is an estimate of the error in the local solution $y_n(x)$ at $x = x_{n+1}$. The local truncation error t_{n+1} associated with the higher order method is

$$t_{n+1} = y_{n+1} - y_n(x_n + h_n) = \sum_{q=1}^{\infty} h_n^q \sum_{i=1}^{\lambda_q} T_{qi} P_{qi} = h_n^{p+1} \Phi(x_n, y_n) + O(h_n^{p+1})$$

where $T_{qi} = Q_{qi} - \xi_{qi}/q!$ with Q_{qi} algebraic functions of A, b, c and ξ_{qi} positive integers. P_{qi} are differentials of f evaluated at (x_n, y_n) and $T_{qi} = 0$ for $q = 1,2,\cdots,p$ and $i = 1,2,\cdots,\lambda_q$. λ_q is the number of elementary differentials for each order and coincides with the number of rooted trees of order q. It is known that

$$\lambda_1 = 1,\ \lambda_2 = 1,\ \lambda_3 = 2,\ \lambda_4 = 4,\ \lambda_5 = 9,\ \lambda_6 = 20,\ \lambda_7 = 48\cdots,\ \text{etc [2].}$$

The set $T^{(q)} = \{T_{q1},\ T_{q2},\cdots,T_{q,\lambda_q}\}$ is formed by the $q-$th order truncation error coefficients. It is usual practice a $(q-1)-$th order method to have minimized

$$\|T^{(q)}\|_2 = \sqrt{\sum_{j=1}^{\lambda_q} T_{qj}^2}.$$

DERIVATION OF RK PAIRS OF ORDERS $5(4)$

The construction of an effectively $6-$stages FSAL Runge-Kutta pair of orders $5(4)$ requires the solution of a nonlinear system of 25 order conditions. $\lambda_1 + \cdots + \lambda_5 = 17$ equations for the higher order formula and $\lambda_1 + \cdots + \lambda_4 = 8$ equations for the lower order formula. There are 28 unknowns. Namely $c_2 - c_6$, $b_1 - b_6$, $\hat{b}_1 - \hat{b}_7$, $a_{32}, a_{42}, a_{43}, a_{52}, a_{53}, a_{54}$ and $a_{62} - a_{65}$.

We proceed setting $c_6 = 1$ and an arbitrary value for $\hat{b}_7$. Then the only assumption we make is

$$b \cdot (A + C - I_s) = 0 \in \Re^{1\times s}$$

with $C = \text{diag}(c)$ and $I_s \in \Re^{s\times s}$ the identity matrix. This is the minimal set of simplifying assumptions for pairs of orders $5(4)$. It is worth mentioning that in the family of methods introduced here

$$A \cdot c \neq \frac{c^2}{2},\ \text{and}\ b_2 \neq 0,$$

contrary to the common practice of every $5(4)$ pair appeared until now [1, 5, 7, 8].

The implicit algorithm that derives a pair of the new family follows

The algorithm producing the coefficients of the new pair

Set $c_6 = 1$ and get an arbitrary $\hat{b}_7 \neq 0$. Select free parameters c_2, c_3, c_4, c_5. Then

1. Solve $b \cdot e = 1,\ b \cdot c = 1/2,\ b \cdot c^2 = 1/3,\ b \cdot c^3 = 1/4,\ b \cdot c^4 = 1/5$ for $b_1, b_2, b_3, b_4,$ and b_5.
2. Solve $\hat{b} \cdot e = 1,\ \hat{b} \cdot c = 1/2,\ \hat{b} \cdot c^2 = 1/3,\ \hat{b} \cdot c^3 = 1/4$ for $\hat{b}_1, \hat{b}_2, \hat{b}_3,$ and $\hat{b}_4$.
3. Solve $b \cdot (A + C - I_s) = 0$ for $a_{62}, \cdots, a_{65}$.
4. Solve $b \cdot A^3 c = 1/120,\ b \cdot A^2 c^2 = 1/60,\ b \cdot (c^2 A c) = 1/12,\ b \cdot (cAc) = 1/8,$ for a_{52}, a_{53}, a_{54} and a_{43}. Adjust the values of a_{62}, a_{63} and a_{64}
5. Solve $\hat{b} \cdot A^2 c = 1/24,\ \hat{b} \cdot (cAc) = 1/4,\ \hat{b} \cdot Ac = 1/6$ for $\hat{b}_5, \hat{b}_6$ and a_{42}. Adjust the values of a_{52}, a_{53}, a_{43} and a_{62}, a_{63}, a_{64}. Reevaluate all $\hat{b}$'s.

6. Solve $\hat{b}\cdot Ac^2 = 1/12$ for b_6. Reevaluate $a_{52}, a_{53}, a_{54}, a_{43}$, $\hat{b}_5, \hat{b}_6$ and a_{42}. Adjust all b's.
7. Solve $b\cdot(Ac)^2 = 1/20$ for a_{32}. Compute the final values of the coefficients.
8. Compute explicitly $a_{21}, a_{31}, a_{41}, a_{51}, a_{61}$ from $A\cdot e = c$

The equations $1-6$ are linear to the coefficients. The seventh equation is a rational function over a_{32}. The numerator of that function is a polynomial of ninth degree and may furnish some real solutions for a_{32}. A Mathematica [13] implementation of the above algorithm requires reevaluation of the coefficients and the order conditions in every step of the algorithm above. In a small computer it needs about $3-4$ seconds to derive the coefficients.

THE NEW RUNGE–KUTTA PAIR

Applying the above described algorithm we constructed a method presented in Table–1. This method shares a rather large value of b_2 and it is clearly far from any other pair appeared until now. The Norm of the principal truncation error is $\|T^{(6)}\|_2 \approx 5.23\cdot 10^{-4}$ while the corresponding value for the Dormand and Prince pair [5] is $\|T^{(6)}\|_2 \approx 3.99\cdot 10^{-4}$.

TABLE 1. The coefficients of the new pair.

$c_2 = 0.231572163526079$	$c_3 = 0.212252555252816$	$c_4 = 0.596693497318054$
$c_5 = 0.797009955708112$	$c_6 = c_7 = 1$	$b_1 = 0.091937670648056$
$b_2 = 1.156529958312496$	$b_3 = -0.781330409541651$	$b_4 = 0.197624776163019$
$b_5 = 0.271639883438847$	$b_6 = 0.063598120979232$	$\widehat{b}_1 = 0.092167469090589$
$\hat{b}_2 = 1.131750860603267$	$\hat{b}_3 = -0.759749304413104$	$\hat{b}_4 = 0.205573577541223$
$\hat{b}_5 = 0.264767065074229$	$\hat{b}_6 = 0.040490332103796$	$\hat{b}_7 = \frac{1}{40},\ b_7 = 0$
$a_{32} = -0.059103796886580$	$a_{42} = 4.560080615554683$	$a_{43} = -4.006458683473722$
$a_{52} = -2.443935658802774$	$a_{53} = 2.631461258707441$	$a_{54} = 0.524706566208284$
$a_{62} = 9.516251378071800$	$a_{63} = -8.467630087008555$	$a_{64} = -0.987888827522473$
$a_{65} = 0.867009765724064$	$a_{i1} = c_i - \sum_{j=2}^{j=i-1} a_{ij},\ j = 1,2,\cdots,6$	$a_{7i} = b_i,\ i = 1,2,\cdots,6$

We run the Runge–Kutta pair for the 25 DETEST [6] non–stiff problems and for tolerances 10^{-2}, 10^{-3}, $\cdots, 10^{-6}$. For stringent tolerances it is preferred to use higher order pairs. DETEST was implemented through MATLAB2009a on a Pentium IV computer running Windows XP at 3.4GHz. For comparison purposes the DP5(4) pair [5] was also run for the same tolerances. We present the results in Table–2.

TABLE 2. Efficiency gains of NEW5(4) relative to DP5(4), for the range of tolerances 10^{-2}, $\cdots$, 10^{-6}.

g.e.	A1	A2	A3	A4	A5	B1	B2	B3	B4	B5	C1	C2	C3	C4	C5	D1	D2	D3	D4	D5	E1	E2	E3	E4	E5
-1									4							4	2	0	-1	-3		5			
-2			0			3	0		3	2						1	0	-3	-4		1	3	8		
-3	0		-1		0	1	0	0	3	1	0	0	0	0	3	-2	-3				1	2	7		
-4	0		-2	3	0		0	0	2	1	0	0		0	2						1	1	7	1	
-5	0	4		2	0		0	1	1		0	0		0	1						1			2	-1
-6	0	3					0	1			0			0										3	-1
8%	0	3	-1	3	0	2	0	1	3	1	0	0	0	0	2	1	0	-2	-3	-3	1	3	7	2	-1

These results were developed according to the guidelines given in [10]. So, we notify the percentage difference in the number of function evaluations required for achieving a given maximum global error for each problem. Unity represents 10%. Numbers have been rounded to the nearest digit. Positive numbers mean that the first method is superior. Zero entries indicate a difference less than 5%. The final row, gives the mean value of efficiency gain for all tolerances in a problem. The left most lower number is the average efficiency gain for all problems. Empty places in the tables are due to the unavailability of data for the respective tolerances. See [9] for more details.

The coefficient $\hat{b}_7$ does not affect $\|T^{(6)}\|_2$ and it was chosen equal to the one given for the pair of [5]. We finally observe that the new method is in average 8% more efficient than Dormand–Prince $5(4)$ for the DETEST problems. It is a remarkable improvement over pairs of same orders and origin.

From a users point of view the question is "what pair is best for low tolerances?". It seems that the classical pairs of orders $5(4)$ that use effectively six stages per step [7, 5, 8] is not the answer. According to our tests for the above tolerances the pair of orders $5(4)$ given in [1] and the pair of orders $6(4)$ given in [12] perform about 25% better.

REFERENCES

1. P. Bogacki and L. F. Shampine, An efficient Runge-Kutta (4,5) pair, Comput. Math. Applic., 32, 15-28 (1996)
2. J. C. Butcher, Coefficients for the study of Runge-Kutta integration processes, J. Austr. Math. Soc., 3, 185-201 (1963)
3. J. C. Butcher, Implicit Runge-Kutta processes, Math. Comput., 18, 50-64 (1964)
4. J. C. Butcher, On Runge-Kutta processes of high order, J. Austral. Math. Soc., 4, 179-194 (1964)
5. J. R. Dormand and P. J. Prince, A family of embedded Runge-Kutta formulae, J. Comput. Appl. Math., **6**, 19–26 (1980)
6. W. H. Enright and J. D. Pryce, Two Fortran Packages for Assessing Initial Value Methods, ACM TOMS, 13, 1-27 (1987)
7. E. Fehlberg, Low order classical Runge-Kutta formulas with stepsize control and their application to some heat-transfer problems, TR R-287, NASA, (1969)
8. S. N. Papakostas and G. Papageorgiou, A family of fifth order Runge-Kutta pairs, Math. Comput. 65(1996) 1165-1181.
9. S. N. Papakostas, Ch. Tsitouras and G. Papageorgiou, A general family of Runge-Kutta pairs of orders 6(5), SIAM J. Numer. Anal., **33**, 917–926 (1996)
10. P. Sharp, Numerical comparisons of some explicit Runge–Kutta pairs, ACM TOMS, **17**, 387-409 (1991)
11. Ch. Tsitouras, Optimized explicit Runge–Kutta pair of orders 9(8), Appl. Numer. Math., 38, 123-134 (2001)
12. Ch. Tsitouras and S. N. Papakostas, Cheap error estimation for Runge–Kutta pairs, SIAM J Sci., Comput., 20, 2067–2088 (1999)
13. Wolfram Research, Inc., Mathematica, Version 6.0, Champaign, IL (2007).