On Modified Runge-Kutta Trees and Methods.

Ch. Tsitouras[a], I. Th. Famelis[b,*], T. E. Simos[c]

[a] *TEI of Chalkis, GR-34400 Psahna, Greece*
[b] *Dept. of Mathematics, TEI of Athens, GR-12210 Egaleo, Greece*
[c] *Univ. of Peloponnese, GR-22100 Tripolis, Greece*

Abstract

Modified Runge–Kutta (mRK) methods can have interesting properties as their coefficients may depend on the step-length. By a simple perturbation of very few coefficients we may produce various function-fitted methods and avoid the overload of evaluating all the coefficients in every step. It is known that, for Runge–Kutta methods, each order condition corresponds to a rooted tree. When we expand this theory to the case of mRK methods, some of the rooted trees produce additional trees, called mRK rooted trees, and so additional conditions of order. In this work we present the relative theory including a theorem for the generating function of these additional mRK trees and explain the procedure to determine the extra algebraic equations of condition generated for a major subcategory of these methods. Moreover, efficient symbolic codes are provided for the enumeration of the trees and the generation of the additional order conditions. Finally, phase-lag and phase-fitted properties are analyzed for this case and specific phase-fitted pairs of orders 8(6) and 6(5) are presented and tested.

Keywords: Rooted trees, integer partitions, truncation error, Runge Kutta Methods, Initial Value Problems, Mathematica.

1. Introduction

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

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

[*]Corresponding author

Email addresses: `tsitoura@teihal.gr` (Ch. Tsitouras), `ifamelis@teiath.gr` (I. Th. Famelis), `tsimos@mail.ariadne-t.gr` (T. E. Simos)

where the function $f : \mathbb{R} \times \mathbb{R}^m \rightarrow \mathbb{R}^m$ is assumed to be as smooth as necessary. The general $s-$stage embedded Runge–Kutta pair of orders $p(p-1)$, for the approximate solution of the problem (1) can be represented using the following Butcher tableau [2, 3]:

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

where $A \in \mathbb{R}^{s\times s}$ is strictly lower triangular, b^T, $\hat{b}^T$, and $c \in \mathbb{R}^s$ with $c = A\cdot e$, $e = [1,\ 1, \cdots, 1]^T \in \mathbb{R}^s$. The vectors b and $\hat{b}$ 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 from 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

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

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 parameter TOL, if $TOL > E_n$, the algorithm

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

provides the next step length. Whereas if $TOL < E_n$ we reject the current step and evaluate another smaller one using the same formula but with h_{n+1} now being h_n.

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

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

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

$$t_{n+1} = y_{n+1} - y_n(x_n + h_n) = \sum_{q=1}^{\infty} h_n^q \sum_{i=1}^{\lambda_q} \sigma_{qi} T_{qi} P_{qi} = h_n^{p+1} \Phi(x_n, y_n) + O(h_n^{p+2})$$

where

$$T_{qi} = (Q_{qi} - \frac{\xi_{qi}}{q!})$$

and σ_{qi} are real numbers depending on the order of the group of automorphisms on a particular labelling of tree t that corresponds to the elementary differential [5]. This order is known as the 'symmetry group' of the tree The ξ_{qi} are positive integers, Q_{qi} are algebraic functions of A, b, c and P_{qi} are differentials of f evaluated at (x_n, y_n). For a $p-$th order method the order conditions

$$T_{qi} = 0, \text{ for } q = 1, 2, \cdots, p \text{ and } i = 1, 2, \cdots, \lambda_q,$$

must hold.

The number of elementary differentials for each order is λ_q and coincides with the number of rooted trees of order q. It is known [1] that

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

More details can be found in [4].

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 common practice that a $(q-1)-$th order method has

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

minimized.

In this work we are interested on a modification of Runge -Kutta methods called Modified Runge -Kutta (mRK). mRK methods can have interesting properties as their coefficients may depend on the step-length. By a simple perturbation of very few coefficients we may produce various function-fitted methods and avoid the overload of evaluating all the coefficients in every step. For this class of methods the works of Franco [8], Vyver [24] and Simos et. al. [15] can be found in the literature .

When we expand the Runge-Kutta tree theory to the case of mRK methods, some of the rooted trees produce additional trees, called mRK rooted trees, and so additional conditions of order. Here we present the relative theory including a theorem for the generating function of these additional mRK trees and explain the procedure to determine the extra algebraic equations of condition generated for a major subcategory of these methods. Moreover, efficient symbolic codes are provided for the enumeration of the trees and the generation of the additional order conditions. Finally, phase-lag and phase-fitted properties are analyzed for this case and specific phase-fitted pairs are presented and tested.

2. Modified Runge–Kutta methods

Vanden Berghe et. al. [22] proposed the modified Runge–Kutta methods where the stages are evaluated by:

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

So the parameter vector $\gamma = [\gamma_1\ \gamma_2 \cdots \gamma_s]^T$ is introduced. The $s-$stages modified Runge–Kutta method is represented by the Butcher tableau:

$$\begin{array}{c|c|cccc}
c_1 & \gamma_1 & & & & \\
c_2 & \gamma_2 & a_{21} & & O & \\
\vdots & \vdots & \vdots & \ddots & & \\
c_s & \gamma_s & a_{s1} & \cdots & a_{s,s-1} & \\
\hline
 & & b_1 & \cdots & b_{s-1} & b_s
\end{array}$$

If $\gamma_i \neq 1$ for some $1 \leq i \leq s$, f becomes involved in the expression for truncation error coefficients T's then little can be said about algebraic order conditions for this type of methods. Modified Runge–Kutta are used considering

$$\gamma_i = 1 + \gamma_{i2} v^2 + \gamma_{i4} v^4 + \cdots,$$

where $v = \omega h$ for some real parameter ω. In that case powers of h produce extra truncation error coefficients and the corresponding truncation error becomes:

$$t_{n+1} = \sum_{q=1}^{\infty} h_n^q (\sum_{i=1}^{\lambda_q} \sigma_{qi} T_{qi} P_{qi} + \sum_{i=1}^{\widetilde{\lambda}_q} \widetilde{\sigma}_{qi} \widetilde{T}_{qi} \widetilde{P}_{qi}) = h_n^{p+1} \widehat{\Phi}(x_n, y_n) + O(h_n^{p+2})$$

where $\widetilde{T}_{qi} = \widetilde{Q}_{qi}$. Again, $\widetilde{\sigma}_{qi}$ real numbers and $\widetilde{Q}_{qi}$ are algebraic functions of $A,\ b,\ c$ and vectors

$$g_2 = [\gamma_{12},\ \gamma_{22},\ \gamma_{32}, \cdots,\ \gamma_{s2}]^T,\ g_4 = [\gamma_{14},\ \gamma_{24},\ \gamma_{34}, \cdots,\ \gamma_{s4}]^T, \text{ etc.}$$

The expressions $\widetilde{P}_{qi}$ are differentials of f and $y(x)$ evaluated at (x_n, y_n) (see Tables 2, 3) and $\widetilde{T}_{qi} = 0$ for $q = 1, 2, \cdots, p$ and $i = 1, 2, \cdots, \widetilde{\lambda}_q$ are the additional $q - th$ order conditions for the modified Runge–Kutta methods. Respectively, $\widetilde{\lambda}_q$ is the number of the additional elementary differentials, and therefore the additional order conditions for the modified Runge–Kutta methods. Franco [8] and Vyver [24] have already presented the additional equations of condition up to order five. In a more recent work París and Rández [13] following the same theory and present embedded pairs of 4(3). Moreover, Simos et. al. have presented phase–fitted mRK [14] and modified Runge–Kutta-Nyström methods [15].

It is known by Butcher theory [4] that, for the family of Runge–Kutta methods, each elementary differential, and so each order condition, corresponds to a rooted tree. So, by enumerating the Runge Kutta related rooted trees one can calculate the number of order conditions λ_q, for $q = 1, 2, \cdots, p$. In this work our aim is to comprehend and generate the mechanism of production of the additional generated trees and order conditions something that has not been presented in the literature before.

Following a simple practical approach one can see that, for the case of modified Runge–Kutta methods some of the RK rooted trees, especially those that their leaves can be collected in couples, or in groups of four, six ... etc, produce the additional modified rooted trees and so additional order conditions. These RK rooted trees are presented along with their corresponding error coefficients for orders $3 - 5$ in Table 2 and for order 6 in Table 3. In order to compute the $\widetilde{\lambda}_q$ for $q = 1, 2, \cdots, p$ we have to enumerate the total number of modified RK trees. We prove the following theorem that gives the generating function of the trees for the all modified RK methods.

Theorem 1. *Let $N_k,\ k = 1, 2, 3, \cdots$ be the number of the rooted trees with k nodes then the corresponding generating function for the modified RK trees is the following*

$$\begin{aligned} M(x) &= x \cdot \frac{1}{(1-x)^{N_1}} \cdot \frac{1}{(1-x^2)^{N_2+1}} \cdot \frac{1}{(1-x^3)^{N_3}} \cdot \frac{1}{(1-x^4)^{N_4+1}} \cdots \\ &= x \prod_{i=1}^{\infty} \sum_{j=0}^{\infty} \Pi_{i,j}\ x^{ij} \qquad (2) \end{aligned}$$

where

$$\Pi_{2i-1,j} = \left(\begin{array}{c} N_{2i-1} + j - 1 \\ j \end{array} \right), \ \Pi_{2i,j} = \left(\begin{array}{c} N_{2i} + j \\ j \end{array} \right), i = 1, 2, 3, \cdots$$

PROOF. *Following the lines Calvo and Sanz–Serna [6], we consider that each tree is produced by other trees that are grafted in new root. The difference form the ordinary rooted trees that correspond to the Runge–Kutta methods is that the number of trees with even number of codes an extra tree exists and that is the ones that leaves are collected in a couple, or in group of two, four, six .... So, collected in couples, or in groups of four, six ... etc. So, if N'_k, $k = 1, 2, 3, \cdots$ is the number of the rooted trees then $N'_{2j} = N_{2j}+1$ and $N'_{2j+1} = N_{2j}$ hold. So, as in the proof found in of Papaioannou [11]*

$$\begin{aligned} M(x) &= x \cdot \frac{1}{(1-x)^{N'_1}} \cdot \frac{1}{(1-x^2)^{N'_2}} \cdot \frac{1}{(1-x^3)^{N'_3}} \cdot \frac{1}{(1-x^4)^{N'_4}} \qquad (3) \\ &= x \cdot \frac{1}{(1-x)^{N_1}} \cdot \frac{1}{(1-x^2)^{N_2+1}} \cdot \frac{1}{(1-x^3)^{N_3}} \cdot \frac{1}{(1-x^4)^{N_4+1}} \cdots \end{aligned}$$

Substituting,

$$\frac{1}{(1-x^k)^N} = \sum_{j=0}^{\infty} \left(\begin{array}{c} N + j - 1 \\ j \end{array} \right) x^{kj}$$

we get the result.

Using this theorem we can enumerate the modified trees. In Table–1 we list the numbers of the additional equations and the total numbers of equations for various orders and in Appendix B we present the Mathematica code which implements the generating function (2) coefficients' and enumerates the modified RK trees

Table 1: Number of order conditions.

order→	1	2	3	4	5	6	7	8	9	10	11	12	13
rooted trees	1	1	2	4	9	20	48	115	286	719	1842	4766	12486
modified trees	0	0	1	2	7	18	53	149	435	1266	3734	11057	32969
total trees	1	1	3	6	16	38	101	264	721	1985	5576	15823	45455

The way, that the sets of the additional truncation order coefficients $\widetilde{T}^{(q)} = \{\widetilde{T}_{q1}, \ \widetilde{T}_{q2}, \cdots, \widetilde{T}_{q,\widetilde{\lambda}_q}\}$ are formed, is shown in Tables 2, 3 for orders

3 to 6. There, the operation "*" may be understood as component-wise multiplication:

$$[b_1 \;\; b_2 \cdots b_s]^T * [\gamma_1 \;\; \gamma_2 \cdots \gamma_s]^T = [b_1\gamma_1 \;\; b_2\gamma_2 \cdots b_s\gamma_s]^T.$$

This operation has the least priority. Parentheses, powers and dot products are always evaluated before "*". Absence of an operation sign means that we use dot product.

As an example, observe that the original truncation error coefficient $bc^5 - \frac{1}{6}$, generates three additional equations. Namely:

$$\widetilde{T}_{6,16} = b(g_2^2 * c) = 0,\; \widetilde{T}_{6,17} = b(g_2 * c^3) = 0 \text{ and } \widetilde{T}_{6,18} = b(g_4 * c) = 0.$$

Using the same methodology the additional modified order conditions and trees for higher orders may be derived. On the other hand, truncation error coefficients like $T_{2,1} = bc - \frac{1}{2}$ or $T_{3,1} = bAc - \frac{1}{6}$, that do not produce any $\widetilde{T}$'s exist.

3. Phase–Lag property and Phase-fitted modified Runge–Kutta pairs

The application of a modified Runge–Kutta method to the test problem

$$y' = i\omega y,\; \omega \in \mathrm{I\!R},\; i = \sqrt{-1}, \tag{4}$$

leads to the numerical scheme,

$$y_{n+1} = (1 - iv^2 b \cdot (I_s + ivA)^{-1}\gamma)y_n = (Q(v^2) + iR(v^2))y_n,$$

where $v = \omega h$, h is the step length, $I_s \in \mathrm{I\!R}^{s\times s}$ being the identity matrix and $Q,\ P$ are polynomials in v^2.

Actually we have

$$Q(v^2) = 1 - \tau_2 v^2 + \tau_4 v^4 - \tau_6 v^6 \pm \cdots,$$

and

$$R(v^2) = \tau_1 v - \tau_3 v^3 + \tau_5 v^5 \mp \cdots$$

with

$$\tau_0 = 1,\; \tau_1 = b\gamma,\; \tau_2 = bA\gamma,\; \tau_3 = bA^2\gamma,\; \tau_4 = bA^3\gamma, \cdots \text{ etc.}$$

Table 2: RK truncation error coefficients and trees producing additional Modified RK truncation error coefficients, trees and their corresponding additional elementary differentials for orders 3-5.

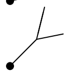
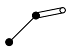
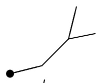
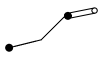
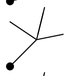
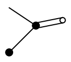
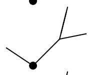
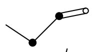
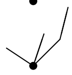
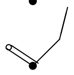
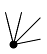
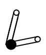
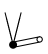
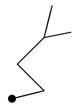
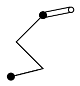
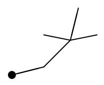
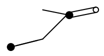
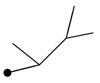
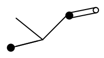
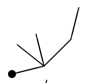
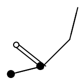
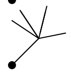
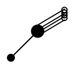
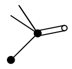
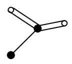
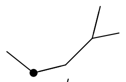
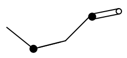
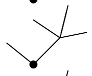
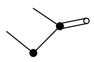
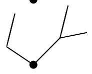
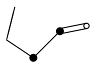
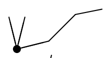
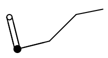
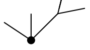
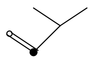
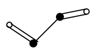
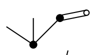
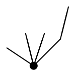
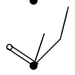
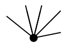
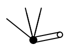
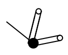
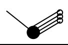
order	T	tree	$\widetilde{T}$	mod. tree	$\widetilde{P}$
3	$T_{31} = bc^2 - \frac{1}{3}$		$\widetilde{T}_{3,1} = bg_2$		$f'y$
4	$T_{41} = bc^3 - \frac{1}{4}$		$\widetilde{T}_{4,1} = b(g_2 * c)$		$f''(y, f)$
	$T_{42} = bAc^2 - \frac{1}{12}$		$\widetilde{T}_{4,2} = bAg_2$		$f'f'y$
5	$T_{5,1} = bA^2c^2 - \frac{1}{60}$		$\widetilde{T}_{5,1} = bA^2g_2$		$f'f'f'y$
	$T_{5,2} = bAc^3 - \frac{1}{20}$		$\widetilde{T}_{5,2} = bA(c * g_2)$		$f'f''(f, y)$
	$T_{5,3} = b(c * Ac^2) - \frac{1}{15}$		$\widetilde{T}_{5,3} = b(c * Ag_2)$		$f''(f, f'y)$
	$T_{5,4} = b(c^2 * Ac) - \frac{1}{10}$		$\widetilde{T}_{5,4} = b(c^2 * Ac)$		$f''(y, f'f)$
	$T_{5,5} = bc^4 - \frac{1}{5}$		$\widetilde{T}_{5,5} = bg_2^2$		$f''(y, y)$
			$\widetilde{T}_{5,6} = b(g_2 * c^2)$		$f^3(y, f, f)$
			$\widetilde{T}_{5,7} = bg_4$		$f'y$

Table 3: Order 6 RK truncation error coefficients and trees producing additional Modified RK truncation error coefficients, trees and their corresponding additional elementary differentials.

T	tree	$\widetilde{T}$	mod. tree	$\widetilde{P}$
$T_{6,1} = bA^3c^2 - \frac{1}{360}$		$\widetilde{T}_{6,1} = bA^3g_2$		$f'f'f'f'y$
$T_{6,2} = bA^2c^3 - \frac{1}{120}$		$\widetilde{T}_{6,2} = bA^2(c * g_2)$		$f'f'f''(f, y)$
$T_{6,3} = bA(c * Ac^2) - \frac{1}{90}$		$\widetilde{T}_{6,3} = bA(c * Ag_2)$		$f'f''(f, f'y)$
$T_{6,4} = bA(c^2 * Ac) - \frac{1}{60}$		$\widetilde{T}_{6,4} = bA(g_2 * Ac)$		$f'f''(y, f'f)$
$T_{6,5} = bAc^4 - \frac{1}{30}$		$\widetilde{T}_{6,5} = bAg_4$		$f'f'y$
		$\widetilde{T}_{6,6} = bA(c^2 * g_2)$		$f'f^{(3)}(f, f, y)$
		$\widetilde{T}_{6,7} = bA(g_2^2)$		$f'f''(y, y)$
$T_{6,6} = b(c * A^2c^2) - \frac{1}{72}$		$\widetilde{T}_{6,8} = b(c * A^2g_2)$		$f''(f, f'f'y)$
$T_{6,7} = b(c * Ac^3) - \frac{1}{24}$		$\widetilde{T}_{6,9} = b(c * A(c * g_2))$		$f''(f, f''(f, y))$
$T_{6,8} = b(Ac * Ac^2) - \frac{1}{72}$		$\widetilde{T}_{6,10} = b(Ac * Ag_2)$		$f''(f'f, f'y)$
$T_{6,9} = b(c^2 * A^2c) - \frac{1}{36}$		$\widetilde{T}_{6,11} = b(g_2 * A^2c)$		$f''(y, f'f'f)$
$T_{6,10} = b(c^2 * Ac^2) - \frac{1}{18}$		$\widetilde{T}_{6,12} = b(g_2 * Ac^2)$		$f''(y, f''(f, f))$
		$\widetilde{T}_{6,13} = b(g_2 * Ag_2)$		$f''(y, f'y)$
		$\widetilde{T}_{6,14} = b(c^2 * Ag_2)$		$f^{(4)}(f, f, f'y)$
$T_{6,11} = b(c^3 * Ac) - \frac{1}{12}$		$\widetilde{T}_{6,15} = b(c * g_2 * Ac)$		$f^{(3)}(y, f, f'f)$
$T_{6,12} = bc^5 - \frac{1}{6}$		$\widetilde{T}_{6,16} = b(g_2 * c^3)$		$f^{(4)}(f, f, f, y)$
		$\widetilde{T}_{6,17} = b(g_2^2 * c)$		$f^{(3)}(f, y, y)$
		$\widetilde{T}_{6,18} = b(g_4 * c)$		$f''(f, y)$

For explicit methods, these are finite series.

The phase lag of a modified Runge–Kutta method is the difference between the angles of theoretical and numerical solutions. Thus, it is defined as the argument of polynomial $Q(v^2) + iR(v^2)$, which is

$$\delta(v^2) = v - \arg(Q(v^2) + iR(v^2)).$$

A phase fitted method satisfies

$$\tan(v) = \frac{R(v^2)}{Q(v^2)} \text{ or equivalently } Q(v^2)\tan(v) = R(v^2).$$

Every conventional Runge–Kutta method of $p-$th order can be modified entering just one γ_i (say γ_2) in order to solve the previous equation.

In the present paper we choose to work with two pairs. First choice is the Runge–Kutta pair of orders 6(5) described in [17]. This pair is chosen as it has the Euclidean norm of its principal truncation error minimized, achieving $\|T^{(7)}\|_2 \approx 1.23 \cdot 10^{-5}$. It is a nine–stage FSAL (First Stage As Last) pair that uses effectively only eight stages per step and its coefficients can be found in [18].

We decide to alter only γ_2 and γ_4 so, we simultaneously solve the following equations :

$$Q(v^2)\tan(v) = R(v^2) \text{ and } \hat{Q}(v^2)\tan(v) = \hat{R}(v^2),$$

where

$$\hat{Q}(v^2) = 1 - \hat{\tau}_2 v^2 + \hat{\tau}_4 v^4 \mp \cdots \text{ and } \hat{R}(v^2) = \hat{\tau}_1 v - \hat{\tau}_3 v^3 \pm \cdots$$

with

$$\hat{\tau}_1 = \hat{b}\gamma,\ \hat{\tau}_2 = \hat{b}A\gamma,\ \text{... etc},$$

which are linear in these two coefficients. Since, the expressions taken are very lengthy, here we present a truncated to 16 digits of accuracy form:

$$\gamma_2 \approx \frac{p_2(v)}{q_2(v)}$$

where

$$p_2(v) = 0.0001684478065771679v^8 - 0.001178746962728090v^6$$

$$+0.01631830414715230v^4 - 0.2013737602727960v^2 + 1$$

and

$$q_2(v) = 0.0001713341496916277v^8 - 0.001219903580379494v^6$$

$$+0.01582838527751099v^4 - 0.2012145820456014v^2 + 1$$

$$\gamma_4 \approx \frac{p_4(v)}{p_4(v)}$$

where

$$p_4(v) = 0.0001567880312125573v^8 - 0.0001703896681492604v^6$$

$$+0.005537711639359206v^4 - 0.1763423156795334v^2 + 1$$

and

$$q_4(v) = 0.0001518439853773218v^8 - 0.0001590567442529793v^6$$

$$+0.005537711639329609v^4 - 0.1763423156795325v^2 + 1.$$

For this approximation a least squares approach technique for $v \in [0,1]$ was applied. Expanding $\gamma_2,\ \gamma_4$ in series we have:

$$\gamma_2 \approx 1 - 0.0001591782272568729v^2 + 0.0004578898911942546v^4 + O(v^6)$$

$$\gamma_4 \approx 1 + O(v^6).$$

Finally we form vectors

$$g_2 = [0, -0.0001591782272568729, 0, 0, 0, 0, 0, 0]^T$$

and

$$g_4 = [0, 0.0004578898911942546, 0, 0, 0, 0, 0]^T$$

to find that the modification of this pair attains orders 6(5). Since $b_2 = \hat{b}_2 = 0$ we may easily verify that all $\widetilde{T}$'s for the sixth order formula and $\widetilde{\hat{T}}$'s for the lower order one vanish.

Our latter choice is a 12–stages pair of orders 8(6) given in [20] that shares a very small principal truncation error coefficient $\|T^{(9)}\|_2 \approx 7.35 \cdot 10^{-7}$ and seems to outperform all other methods at stringent tolerances [19].

Now, by deciding to alter γ_2 and γ_5 and by following the same steps as above, we manage to get the rational forms for the γ's:

$$\gamma_2 \approx \frac{p_2'(v)}{q_2'(v)}$$

where

$$p_2'(v) = 0.000032766053976649983v^8 - 0.013868369214345342v^6 \\ +0.23817284571843967v^4 - 0.95295730050968709v^2 + 1$$

and

$$q_2'(v) = 0.000010830539734763642v^8 - 0.013492222154389920v^6 \\ +0.23613131714014087v^4 - 0.95045239692026381v^2 + 1$$

$$\gamma_5 \approx \frac{p_5'(v)}{q_5'(v)}$$

where

$$p_5'(v) = 0.000075447483510904058v^8 - 0.0039691890854295819v^6 \\ +0.18468143502673477v^4 - 0.89094241368242844v^2 + 1$$

and

$$q_5'(v) = 0.000057868852463954276v^8 - 0.0037899552106836919v^6 \\ +0.18442706132555963v^4 - 0.89094241368255460v^2 + 1.$$

By expanding $\gamma_2,\ \gamma_5$ in series we have:

$$\gamma_2 \approx 1 - 4.4540580527356974 \cdot 10^{-4}v^2 + 4.4490984813753475 \cdot 10^{-5}v^4 \\ -2.7148314786806795 \cdot 10^{-6}v^6 + O(v^8)$$

and

$\gamma_5 \approx 1 + 1.9849805345038804 \cdot 10^{-5}v^4 + 3.6986894486308681 \cdot 10^{-6}v^6 + O(v^8)$.

Once again we form the vectors

$$g_2 = [0, -4.4540580527356974 \cdot 10^{-4}, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]^T,$$

$g_4 = [0, 4.4490984813753475 \cdot 10^{-5}, 0, 0, 1.9849805345038804 \cdot 10^{-5}, 0, 0, 0, 0, 0, 0, 0]^T$

and

$g_6 = [0, -2.7148314786806795 \cdot 10^{-6}, 0, 0, 3.6986894486308681 \cdot 10^{-6}, 0, 0, 0, 0, 0, 0, 0]^T$

to verify that the modified pair preserves the desired initial order.

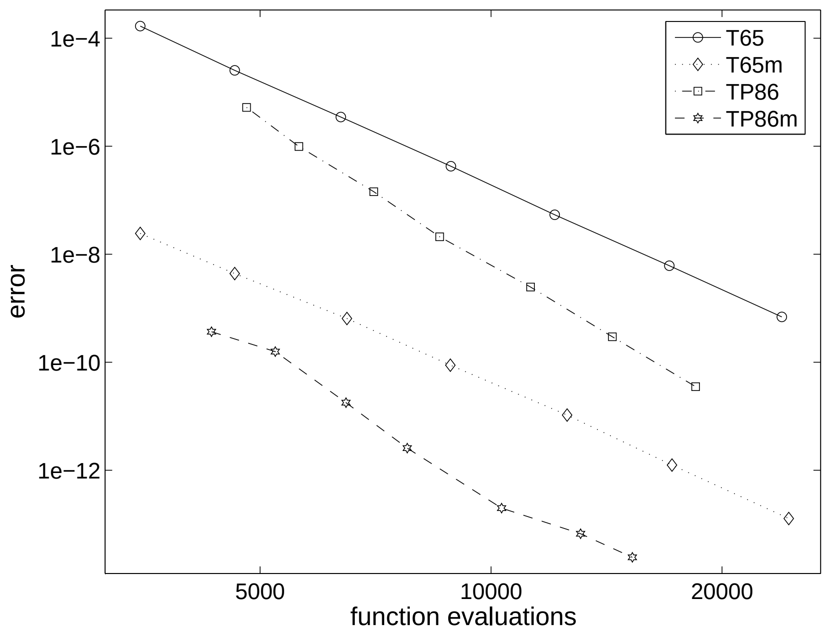


Figure 1: Bessel problem.

4. Numerical Tests

In order to test the efficiency of the modified pairs we apply the following methods:

1. T65, the pair of orders 6(5) given in [17],
2. T65m, the modification of T65 presented above,
3. TP86, the pair of orders 8(6) found in [20],
4. TP86m, the modification of TP86 presented above,

to a choice of well-known, problems found in the relevant literature. We concentrate on high order pairs but a similar modification can be used for lower order methods like the methods presented in the pioneering article of Houwen and Sommeijer [9].

4.1. Bessel equation

First we consider

$$y'' = \left(-100 + \frac{1}{4x^2}\right) y,\ y\left(1\right) = J_0\left(10x\right),\ y'\left(1\right) = -0.5576953439142885,$$

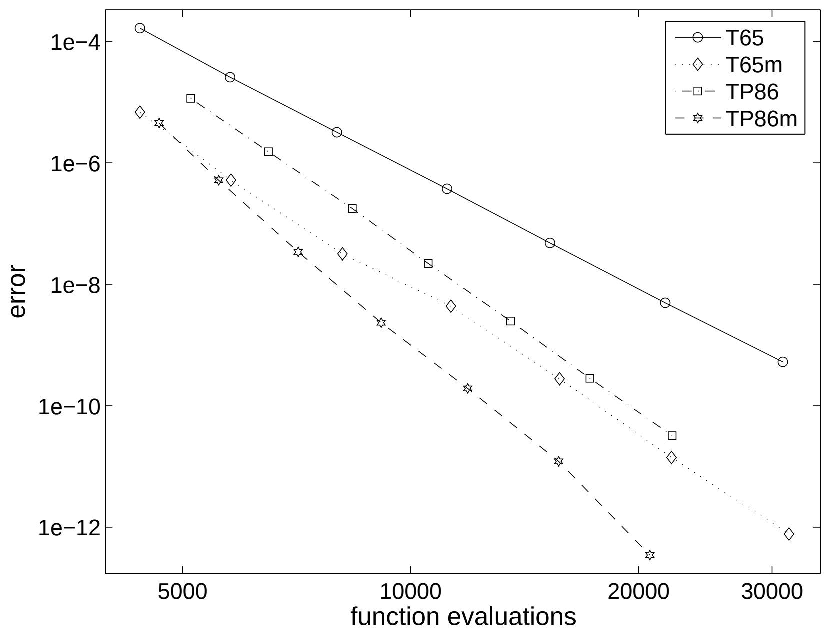


Figure 2: Inhomogeneous problem.

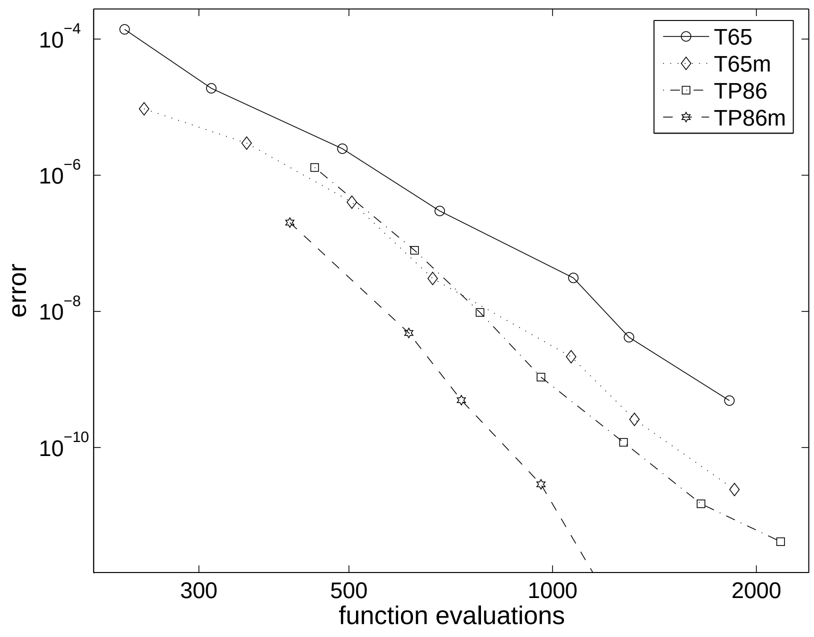


Figure 3: Duffing problem.

which has as theoretical solution

$$y(x)=\sqrt{x}J_0\left(10x\right).$$

We solve the above equation choosing $\omega=10$, in order to find the 100th root of the solution which is equal to 32.59406213134967 [21].

4.2. Inhomogeneous equation

Our second test problem is an inhomogeneous problem:

$$y''=-100y(x)+99\sin(x),\ y(0)=1,\ y'(0)=11$$

with analytical solution

$$y(t)=\cos(10x)+\sin(10x)+\sin(x).$$

We integrate it for $x\in[0,10\pi]$ as in [16]. Again, $\omega=10$ was considered for this problem.

4.3. Duffing equation

Finally, we consider the following problem

$$\begin{aligned} y'' &= -y-y^3+\frac{1}{500}\cdot\cos\left(1.01t\right), \\ y\left(0\right) &= 0.200426728067,\ y'\left(0\right)=0, \end{aligned}$$

with a theoretical solution given by Van Dooren[23]

$$\begin{aligned} y(x) &= .200179477536\cos(1.01t)+2.46946143\cdot10^{-4}\cos(3.03t) \\ &\quad +3.04014\cdot10^{-7}\cos(5.05t)+3.74\cdot10^{-10}\cos(7.07t). \end{aligned}$$

We now solve in the interval $\left[0,\frac{10.5}{1.01}\pi\right]$ as $y\left(\frac{10.5}{1.01}\pi\right)=0$, picking $\omega=1.01$.

For all problems, the 6(5) pairs are run with tolerances $10^{-3},\ 10^{-4},\cdots,$ 10^{-9}, while for the 8(6) pairs somewhat stringent tolerances $10^{-5},\ 10^{-6},\cdots,$ 10^{-11} are applied. For all methods, we measure the number of stages used and the magnitude of the end point global error. In Figures 1, 2 and 3 we summarize the results providing the corresponding efficient curves in a logarithmic scale. All computations and drawings are done using MATLAB [10].

By interpreting the results, we notice that for the Bessel equation the modified pairs are almost 4 digits more accurate. Moreover, in the inhomogeneous problem more than 2 digits of accuracy are gained when using the modified pairs. We note that for both problems T65m is clearly more efficient than T86, something that is expected since these specific problems are close to the test problem (4). But even for the nonlinear Duffing equation the modified pairs gain more than a digit of accuracy. In the past, much effort was made for the generation of Runge–Kutta pairs for much less profit [12, 17, 20].

Appendix A. Mathematica code for the enumeration of the additional order conditions

In the following lines we give the Mathematica code which implements the generating function (2) coefficients' and enumerates the modified RK trees .

```
In[1]:=Clear["@"] <<DiscreteMath`Combinatorica`
In[2]:=op[n_,k_]:=n!/(k!(n-k)!)
In[3]:=to[m_/;OddQ[m],j_]:=op[t[[m]]+j-1,j]
In[4]:=to[m_/;EvenQ[m],j_]:=op[t[[m]]+j,j]
In[5]:=RunLengthEncode[x_List]:=(Through[{First,Length}[#1]]&)/@Split[x];
In[6]:=nn=16; In[7]:=t=Table[1,{i,1,nn}];
In[8]:=tf[n_]:=Apply[Plus,
          Apply[Times,Apply[to,Map[RunLengthEncode,Partitions[n-1]],{2}],{1}]]
In[9]:=Do[t[[i]]=tf[i],{i,1,nn}]
In[10]:=t
Out[10]:=
{1,1,3,6,16,38,101,264,721,1985,5576,15823,45455,131675,384631,1131045}
```

Appendix B. Mathematica package for the generation of the additional order conditions

We implemented a Mathematica [25] package for the production of the additional trees $\widetilde{T}$. We name it `Trees16Mod` and it is listed in Appendix C. Its basic function `modtrees` gets the body of an order condition Q_{qi} as input and returns the set of modified trees produced.

```
In[10]:=<<trees16mod.m;

In[11]:=modtrees[b.a.a.c^2](* the main tree and one additional are produced *)
```

Out[11]:=$\{b.a.a.c^2, b.a.a.g_2\}$

```
In[12]:=modtrees[b.a.a.c](* nothing additional produced *)
```

Out[12]:=$\{b.a.a.c\}$

```
In[13]:=modtrees[b.c^6](* the main tree and six additional are produced *)
```

Out[13]:=$\{b.c^6, b.\left(c^4 g_2\right), b.\left(c^2 g_2^2\right), b.g_2^3, b.\left(c^2 g_4\right), b.\left(g_2 g_4\right), b.g_6\}$

In order to produce the modified order conditions of seventh order we need the package `trees16` given [7]. That package produced the order conditions T_{qi} of the conventional $q-$th order Runge–Kutta method.

```
In[14]:=<< trees16.m;

In[15]:=oc7 = RKCond[a, b, c, e, 7][[1]];(* 7th order conditions *)

In[16]:=oc7bod = Table[oc7[[j,2]], {j,1,Length[oc7]}];(* body of the conditions *)

(* The following instruction produces the modified trees of 7th order *)

In[17]:=Complement[Flatten[Table[modtrees[oc7bod[[j]]],{j,1,Length[oc7]}]],oc7bod]
```

`Out[17]:=`$\{b.\left(cA.\left(cA.g_2\right)\right), b.\left(c^3A.g_2\right), b.\left(cA.cA.g_2\right), b.\left(A.c^2A.g_2\right), b.\left(A.g_2\right)^2, b.\left(c^2A.\left(cg_2\right)\right),$
$b.\left(A.cA.\left(cg_2\right)\right), b.\left(cA.\left(c^2g_2\right)\right), b.\left(cA.\left(A.cg_2\right)\right), b.\left(cA.g_2^2\right), b.\left(cA.g_4\right), b.\left(A.g_2A.A.c\right),$
$b.\left(c^2A.A.g_2\right), b.\left(A.cA.A.g_2\right), b.\left(cA.A.\left(cg_2\right)\right), b.\left(cA.A.A.g_2\right), b.\left(c^4g_2\right), b.\left(c^2A.cg_2\right),$
$b.\left(\left(A.c\right)^2g_2\right), b.\left(cA.c^2g_2\right), b.\left(A.c^3g_2\right), b.\left(A.(cA.c)g_2\right), b.\left(cA.g_2g_2\right), b.\left(A.\left(cg_2\right)g_2\right),$
$b.\left(cA.A.cg_2\right), b.\left(A.A.c^2g_2\right), b.\left(A.A.g_2g_2\right), b.\left(A.A.A.cg_2\right), b.\left(c^2g_2^2\right), b.\left(A.cg_2^2\right), b.g_2^3,$
$b.\left(c^2g_4\right), b.\left(A.cg_4\right), b.\left(g_2g_4\right), b.g_6, b.A.\left(c^2A.g_2\right), b.A.\left(A.cA.g_2\right), b.A.\left(cA.\left(cg_2\right)\right), b.A.\left(cA.A.g_2\right),$
$b.A.\left(c^3g_2\right), b.A.\left(cA.cg_2\right), b.A.\left(A.c^2g_2\right), b.A.\left(A.g_2g_2\right), b.A.\left(A.A.cg_2\right), b.A.\left(cg_2^2\right), b.A.\left(cg_4\right),$
$b.A.A.\left(cA.g_2\right), b.A.A.\left(c^2g_2\right), b.A.A.\left(A.cg_2\right), b.A.A.g_2^2, b.A.A.g_4, b.A.A.A.\left(cg_2\right), b.A.A.A.A.g_2\}$

```
In[18]:=Length[%](* Enumerate the order conditions of 7th order *)

Out[18]:=53
```

Appendix C. The Mathematica Package Source

```
BeginPackage["Trees16mod`", {"Combinatorica`"}];

modtrees::usage = " modtrees[tr] finds all modifications of
RK-tree  "

Power2[x_, n_Integer] := Apply[Power3, Table[x, {i, 1, n}]];

modtrees[tr_] :=
   Module[{t1, q, q1, q2, i, j, pow},
          q = tr /. Power -> Power1 /. Power1[c, n_Integer] -> c^n;
          q = q /. Power1 -> Power2;
          t1 = Tuples[
                       Map[Tuples,Flatten[Select
                       [Table[Table[{modi[pow], {Position[q, c^pow][[i]]}},
                        {i, 1,Length[Position[q, c^pow]]}
                                   ], {pow, 1, powmax[q]}
                             ], # =!= {} & ],1]
                          ]
                     ];
          q2 = {};
          Do[q1 = q;
             q1 =ReplacePart[q1, Table[t1[[j, i, 1]], {i, 1, Length[t1[[j]]]}],
                                 Table[t1[[j, i, 2]], {i, 1, Length[t1[[j]]]}],
                                 Table[{i}, {i, 1, Length[t1[[j]]]}]
                            ];
             q2 = Append[q2, q1], {j, 1, Length[t1]}
            ];
          q2 = Map[First, Split[Sort[q2 /. Power3 -> Times]]]
          ];

modi[n_] :=
   Module[{p, q}, p = Partitions[n];
           q = c^p[[Map[First, Split[Map[First, Position[EvenQ[p], True]]]]]];
           Do[q = Replace[q, c^i -> Subscript[g, i], 2], {i, 2, n, 2}];
           q = Apply[Dot, q, 1];
           q = If[EvenQ[n],
                  Union[{q[[1]]},
                        Map[First, Split[Sort[Apply[Times, Delete[q, 1], 1]]]]
                       ],
                  Map[First, Split[Sort[Apply[Times, q, 1]]]]
                 ];
           q = Append[q, c^n]
         ];
powmax[tr_] :=
   Module[{p, q, j1}, q = tr; p = Position[q, c^_];
           If[p == {}, 1, Do[AppendTo[p[[j1]], 2], {j1, 1, Length[p]}];
                          Max[Extract[q, p]]]
         ];
EndPackage[];
```

References

[1] J. C. Butcher, Coefficients for the study of Runge–Kutta integration processes, J. Austr. Math. Soc. 3 (1963) 185–201.

[2] J. C. Butcher, Implicit Runge–Kutta processes, Math. Comput. 18 (1964) 50–64.

[3] J. C. Butcher, On Runge–Kutta processes of high order, J. Austral. Math. Soc. 4 (1964) 179–194.

[4] J. C. Butcher, *The Numerical Analysis of Ordinary Differential Equations. Runge–Kutta & General Linear methods*, John Wiley & sons, Chichester, 1987.

[5] J. C. Butcher, *Numerical methods for Ordinary Differential Equations, Second Edition*, Wiley, Chichester, 2008.

[6] M. P. Calvo and J. M. Sanz–Serna, High order symplectic Runge–Kutta–Nyström methods, SIAM J. Sci. Comput. 14 No 5 (1993) 1237–1252.

[7] I. Th. Famelis, S. N. Papakostas and Ch. Tsitouras, Symbolic derivation of Runge–Kutta order conditions, J. Symb. Comput. 37 (2004) 311–327.

[8] J. M. Franco, An embedded pair of exponentially fitted explicit Runge-Kutta methods, J. Comput. Appl. Math. 149 (2002) 407–414.

[9] P.J. Van der Houwen and B.P. Sommeijer, Explicit Runge–Kutta(–Nystrom) methods with reduced phase errors for computing oscillating solutions, SIAM J. Numer. Anal. 24 (1987) 595–617.

[10] MATLAB 7 (Release 14), The MathWorks inc., Natick, MA, 2004.

[11] A. Papaioannou, Enumeration of Graphs (in Greek), NTUA, Athens, 2000.

[12] S. N. Papakostas, Ch. Tsitouras, and G. Papageorgiou, A general family of explicit Runge–Kutta pairs of orders 6(5), SIAM J. Numer. Anal. 33 (1996) 917–936.

[13] A. París , L. Rández , New embedded explicit pairs of exponentially fitted Runge–Kutta methods, J. Comp. Appl. Math. 3234 (2010) 767–776.

[14] T.E. Simos and J. Vigo-Aguiar, A modified Runge-Kutta method with phase-lag of order infinity for the numerical solution of the Schrodinger equation and related problems, Comput. Chem. 25 (2001) 275-281.

[15] T. E. Simos and J. Vigo-Aguiar, Exponentially fitted symplectic integrator, Phys. Rev. E 67 (2003) No 016701

[16] T. E. Simos, I. Th. Famelis and Ch. Tsitouras, Zero dissipative, explicit Numerov type methods for second order IVPs with oscillating solutions, Numer. Algorithms 34 (2003) 27–40.

[17] Ch. Tsitouras, A parameter study of explicit Runge–Kutta pairs of orders 6(5), Appl. Math. Lett. 11 (1998) 65–69.

[18] Ch. Tsitouras, I. Th. Famelis, Phase-Fitted modified Runge-Kutta pairs of orders 6(5), ICNAAM Extended Abstracts (2006) 1962 1965.

[19] Ch. Tsitouras, Optimal Runge–Kutta pairs of orders 9(8), Appl. Numer. Math. 38 (2001), 123–134.

[20] Ch. Tsitouras and S. N. Papakostas, Cheap error estimation for Runge–Kutta pairs, SIAM J Sci. Comput. 20 (1999) 2067–2088.

[21] Ch. Tsitouras and T. E. Simos, Explicit high order methods for the numerical integration of periodic initial value problems, Appl. Math.& Comput. 95 (1998) 15–26.

[22] G. Vanden Berghe, H. De Meyer, M. Van Daele and T. Van Hecke, Exponentially–fitted explicit Runge–Kutta methods, Comput. Phys. Commun. 123 (1999) 7–15.

[23] R. Van Dooren, Stabilization of Cowell's classical difference method for numerical integration, J. Comput. Phys. 16 (1974) 186–192.

[24] H. Van de Vyver, An embedded phase-fitted modified Runge-Kutta method for the numerical integration of the radial Schrödinger equation, Phys. Lett. A 352 (2006) 278–285.

[25] S. Wolfram, *The MATHEMATICA Book, 5th ed.*, Wolfram Med., 2003.