

Subquadrature Expansions for TSRK methods

Anne Kværnø* and Jim Verner†

January 3, 2011

Abstract.

The representation of order conditions for general linear methods formulated using an algebraic theory by Butcher, and the alternative using B-series by Hairer and Wanner for treating vector initial value problems in ordinary differential equations are well-known. Each relies on a recursion over rooted trees; yet tractable forms - for example, those which may be solved to yield particular methods - can only be obtained with extensive computation. In contrast, for Runge-Kutta methods, tractable forms have been used by various authors for obtaining methods. Here, the corresponding recursion formula for two-step Runge-Kutta methods is revised to give a tractable form which may be exploited to derive such methods and motivate the selection of efficient algorithms in an obvious way. The new recursion is utilized in a MAPLE code (which is available from the authors).

Key words. Two-step Runge-Kutta methods, B-series, order conditions, local error estimation, MAPLE code.

Mathematics subject classifications (2000) 65L05, 65L06, 65L20.

1 Introduction

Butcher [1] was the first to derive order conditions for general high-order discrete variable methods using his algebraic theory of integration methods. This was reformulated using B-series with a recursion formula by Hairer and Wanner [7, 8]. Although these forms did not easily yield tractable forms of order conditions, equivalent forms were solved by a number of authors for some types of methods. For example, a general formulation developed by Cooper and Verner [6] showed that Runge-Kutta methods could be derived by confining the nodes and weights to yield a quadrature method of order p , to select some coefficients $a_{i,j}$ to satisfy

*Department of Mathematics, NTNU, Trondheim, Norway, (anne@math.ntnu.no).

†Department of Mathematics, PIMS, Simon Fraser University, Burnaby, Canada, V5A 1S6 (jverner@pims.math.ca). The work of this author was supported by the National Science and Engineering Council of Canada grant A8147.

certain "subquadrature" conditions, and the remaining free parameters to satisfy the remaining constraints expressed as sums of polynomials of subquadrature expressions.

Here, we indicate how this might be done for some general linear methods. As an application, we express the order conditions for two-step Runge–Kutta methods in a more convenient form than those developed by Jackiewicz and Tracogna [10], Tracogna and Butcher [4], or Hairer and Wanner [9]. Moreover, this new form yields convenient implementations of algorithms for testing the order and computing norms of the leading terms of the local truncation errors. Such an algorithm is utilized in a MAPLE code which is available on request from the authors.

This article develops an algorithm to express the necessary order conditions as polynomials of subquadrature expressions. While it is applied here to two-step Runge–Kutta methods, the approach can be developed for other types of methods. In addition to indicating how the order conditions might be solved directly, this representation is used to construct tools for the analysis of such methods and for comparison with other types of methods.

In this paper, we consider the autonomous initial value problem

$$y' = f(y), \quad y(t_0) = y_0, \quad (1.1)$$

solved by a two step Runge–Kutta (TSRK) method [10]

$$Y_n = \mathbf{u}y_{n-1} + (\mathbb{1} - \mathbf{u})y_n + hA f(Y_{n-1}) + hB f(Y_n), \quad (1.2a)$$

$$y_{n+1} = \theta y_{n-1} + (1 - \theta)y_n + h\mathbf{v}^T f(Y_{n-1}) + h\mathbf{w}^T f(Y_n). \quad (1.2b)$$

where A , B are square matrices, and \mathbf{u} and $\mathbb{1} = [1, \dots, 1]^t$ are vectors of dimension s . For a diagonal matrix C , we define the nodes by

$$\mathbf{c} = C\mathbb{1} = (A + B)\mathbb{1} - \mathbf{u},$$

and subquadratures and quadratures by

$$q^{[k]} = \frac{C^k}{k} \mathbb{1} - \left[\frac{(-1)^k}{k} \mathbf{u} + A(C - I)^{k-1} \mathbb{1} + BC^{k-1} \mathbb{1} \right], \quad k = 1, 2, \dots, \quad (1.3a)$$

$$Q^{[k]} = \frac{1}{k} - \left[\frac{(-1)^k}{k} \theta + \mathbf{v}^T (C - I)^{k-1} \mathbb{1} + \mathbf{w}^T C^{k-1} \mathbb{1} \right], \quad k = 1, 2, \dots \quad (1.3b)$$

We observe that the definition of the nodes implies that vector $q^{[1]} = 0$, and also for a method of order at least 1, that scalar $Q^{[1]} = 0$.

2 Order conditions expressed by trees and subquadratures

By the use of Albrecht approach, Jackiewicz and Tracogna [10] developed the order conditions of TSRK methods expressed using recurrence on the subquadratures $q^{[k]}$, while Hairer and Wanner [9] as well as Tracogna and Butcher [4] expressed the conditions in terms of B-series and rooted trees. In this section, we will use the rooted tree theory to develop order conditions expressed by subquadratures, and thereby establish a link between the two approaches.

The exact solution $y(t_n + h)$ as well as the corresponding numerical solution y_{n+1} and the vector of stage values Y_n can (under certain conditions) be written in terms of B-series around $y(t_n)$, defined formally by

$$\mathcal{B}(a, y; h) = \sum_{\tau \in T} \frac{a(\tau)}{\sigma(\tau)} h^{\rho(\tau)} F(\tau)(y)$$

where each τ is a rooted tree in T having $\rho(\tau)$ nodes and density $\sigma(\tau)$. Each rooted tree has a *root-partition* $\tau = [\tau_1, \dots, \tau_m]$ which is defined as the set of subtrees remaining when the root is removed. For an introduction to B-series, see e.g. [3, 7]. In particular, the exact solution is given by

$$y(t_n + h) = \mathcal{B}(e, y(t_n); h)$$

with

$$e(\emptyset) = 1, \quad e(\tau) = \frac{1}{\rho(\tau)} \prod_{j=1}^m e(\tau_j) \equiv \frac{1}{\rho(\tau)} \prod_{j=1}^m \frac{1}{\gamma(\tau_j)} = \frac{1}{\gamma(\tau)}. \quad (2.4)$$

We know from [7, Chap. III], that under the condition $a(\emptyset) = 1$, the following conditions hold:

$$y(t_n + \alpha h) = \mathcal{B}(e^\alpha, y(t_n); h), \quad e^\alpha(\tau) = \alpha^{\rho(\tau)} e(\tau), \quad (2.5a)$$

$$hf(\mathcal{B}(a, y(t_n); h)) = \mathcal{B}(a', y(t_n); h), \quad (2.5b)$$

$$\mathcal{B}(b, \mathcal{B}(a, y(t_n); h); h) = \mathcal{B}(ab, y(t_n); h), \quad (2.5c)$$

with

$$a'(\emptyset) = 0, \quad a'(\bullet) = a(\emptyset) = 1, \quad a'([\tau_1, \dots, \tau_m]) = \prod_{j=1}^m a(\tau_j), \quad (2.5d)$$

$$(ab)(\emptyset) = b(\emptyset), \quad (ab)(\bullet) = b(\bullet) + b(\emptyset)a(\bullet), \quad (ab)(\tau) = \sum_{\theta \in OST(\tau)} b(\theta) \prod_{\delta \in \tau \setminus \theta} a(\delta), \quad (2.5e)$$

where \bullet denotes the tree with a single node. From [1, 8] it follows that

$$(c(a+b))(\tau) = (ca)(\tau) + (cb)(\tau), \quad (2.5f)$$

$$((ab)c)(\tau) = (a(bc))(\tau) \quad \text{whenever } a(\emptyset) = b(\emptyset) = 1. \quad (2.5g)$$

We also conclude that

$$(ab')(\tau) = (ab)'(\tau), \quad \text{if } b(\emptyset) = 1, \quad (2.5h)$$

since

$$hf(\mathcal{B}(b, \mathcal{B}(a, y; h); h)) = \begin{cases} \mathcal{B}(b', \mathcal{B}(a, y; h); h) & = \mathcal{B}(ab', y; h) \quad \text{or else} \\ hf(\mathcal{B}(ab, y; h)) & = \mathcal{B}((ab)', y; h), \end{cases}$$

and the two series are equivalent. For later use, we introduce the notation

$$a_0(\tau) = a(\tau), \quad a_{r+1}(\tau) = (e^{-1}a_r)(\tau), \quad r = 0, 1, 2, \dots \quad (2.6)$$

If $a(\emptyset)$ is either 0 or 1, we get by (2.5g)

$$a_r(\tau) = e^{-r}a_0(\tau), \quad r = 1, 2, \dots$$

Method (1.2) is of order p if $y_n = y(t_n) + \mathcal{O}(h^{p+1})$ for each n , and in [9], B-series of the approximations about the exact solution for which $y_n = y(t_n)$ yield algebraic order conditions. Hence, replacing y_{n+1} , y_n , y_{n-1} , Y_n and Y_{n-1} by $\mathcal{B}(\psi, y(t_n))$, $y(t_n)$, $y(t_{n-1})$, $\mathcal{B}(\phi, y(t_n); h)$ and $\mathcal{B}(\phi, y(t_{n-1}); h)$ in (1.2) and comparing term by term implies the following relations hold for most trees with up to p nodes:

$$\phi(\emptyset) = \mathbb{1}, \quad \phi(\tau) = e^{-1}(\tau) \mathbf{u} + A(e^{-1}\phi)'(\tau) + B\phi'(\tau), \quad (2.7a)$$

$$\psi(\emptyset) = 1, \quad \psi(\tau) = e^{-1}(\tau) \theta + \mathbf{v}^T(e^{-1}\phi)'(\tau) + \mathbf{w}^T\phi'(\tau). \quad (2.7b)$$

To be precise, the method is of order p if and only if (2.7a) is satisfied for all $\tau \in T$, $\rho(\tau) \leq p-1$, and $\psi(\tau) = e(\tau)$ for all $\tau \in T$, $\rho(\tau) \leq p$, see [4, 9]. Later, we shall invoke an exact B-series for y_{n-1} to characterize the local truncation error.

Our first aim is to reformulate (2.7) in terms of quadratures $Q^{[k]}$ and subquadratures $q^{[k]}$. For a diagonal matrix C , we write $e^C(\tau)$ to denote the *vector* for which the i -th element is $e^{c_i}(\tau)$. Then, the desired results arise from the errors:

$$y(t_n + \mathbf{c}h) - \mathcal{B}(\phi, y(t_n); h) = \mathcal{B}(q, y(t_n); h), \quad \text{with } q(\tau) = e^C(\tau) - \phi(\tau), \quad (2.8a)$$

$$y(t_n + h) - \mathcal{B}(\psi, y(t_n); h) = \mathcal{B}(\chi, y(t_n); h), \quad \text{with } \chi(\tau) = e(\tau) - \psi(\tau). \quad (2.8b)$$

Now we are ready to state the main theorem of this section:

Theorem 2.1. For all $\tau \in T$, the weights $q(\tau)$ with $0 \leq \rho(\tau) < p$ and $\chi(\tau)$ with $0 < \rho(\tau) \leq p$ are generated by the following algorithm:

For $k = 1, 2, \dots, p-1$, and $r = 0, 1, \dots, p-k-1$, then using (2.6), vectors

$$q_r([\bullet^{k-1}]) = \frac{1}{k} \left\{ [(C - rI)^k - (-rI)^k] \mathbb{1} - [(-(r+1))^k - (-r)^k] \mathbf{u} \right\} \\ - A(C - (r+1)I)^{k-1} \mathbb{1} - B(C - rI)^{k-1} \mathbb{1}. \quad (2.9a)$$

Now denote $q_r^{[k]} = q_r([\bullet^{k-1}])$ as a natural extension of (1.3a) for $r \geq 0$. Suppose for $\tau = [\tau_1, \dots, \tau_m]$ that $q_{r+1}(\tau_j)$, $j = 1, \dots, m$, have already been obtained. Then

$$q_r(\tau) = \frac{\rho(\tau)}{\gamma(\tau)} q_r^{[\rho(\tau)]} \\ + A \left[\prod_{j=1}^m \frac{(C - (r+1)I)^{\rho(\tau_j)}}{\gamma(\tau_j)} \mathbb{1} - \prod_{j=1}^m \left(\frac{(C - (r+1)I)^{\rho(\tau_j)}}{\gamma(\tau_j)} \mathbb{1} - q_{r+1}(\tau_j) \right) \right] \\ + B \left[\prod_{j=1}^m \frac{(C - rI)^{\rho(\tau_j)}}{\gamma(\tau_j)} \mathbb{1} - \prod_{j=1}^m \left(\frac{(C - rI)^{\rho(\tau_j)}}{\gamma(\tau_j)} \mathbb{1} - q_r(\tau_j) \right) \right]. \quad (2.9b)$$

Then $q(\emptyset) = 0$, $q(\tau) = q_0(\tau)$ and

$$\chi(\tau) = \frac{\rho(\tau)}{\gamma(\tau)} Q^{[\rho(\tau)]} \\ + \mathbf{v}^T \left[\prod_{j=1}^m \frac{(C - I)^{\rho(\tau_j)}}{\gamma(\tau_j)} \mathbb{1} - \prod_{j=1}^m \left(\frac{(C - I)^{\rho(\tau_j)}}{\gamma(\tau_j)} \mathbb{1} - q_1(\tau_j) \right) \right] \\ + \mathbf{w}^T \left[\prod_{j=1}^m \frac{C^{\rho(\tau_j)}}{\gamma(\tau_j)} \mathbb{1} - \prod_{j=1}^m \left(\frac{C^{\rho(\tau_j)}}{\gamma(\tau_j)} \mathbb{1} - q_0(\tau_j) \right) \right] \quad (2.9c)$$

Remark 1. In order to apply (2.9b), notice for each tree $\tau = [\tau_1, \dots, \tau_m]$ that both $q_r(\tau_j)$ and $q_{r+1}(\tau_j)$ have to be calculated for all τ_j *before* the calculation of $q_{r+1}(\tau)$. Accordingly, for each tree τ of order $\rho(\tau)$, we will compute $q_r(\tau)$ for all r required before computing corresponding values for trees of higher order.

Remark 2. The recursive computation of (2.9b) has (2.9a) as its foundation. Accordingly, all values of $\chi(\tau)$ can be represented explicitly using only subquadratures $q_r^{[k]}$ as is seen in Tables 1, 3 and 4. Nevertheless, for efficient computation of $\chi(\tau)$, values of subquadrature expressions $q_r(\tau)$ are stored as intermediate values.

Remark 3. Restricting each of A, B, C to be single algebraic entries in the MAPLE code that has been developed yields the order conditions shown in Tables 1, 3 and 4.

Table 1: Applying Theorem 2.1 to obtain Order Conditions up to order 4.

τ	$q_r(\tau)$	$\chi(\tau)$	$\gamma(\tau)$	$\sigma(\tau)$
\bullet	$q_r^{[1]}$	$Q^{[1]}$	1	1
\vdots	$q_r^{[2]}$	$Q^{[2]}$	2	1
$\begin{array}{c} \vee \\ \vdots \end{array}$	$q_r^{[3]}$	$Q^{[3]}$	3	2
$\begin{array}{c} \vdots \\ \vdots \end{array}$	$q_r^{[3]}/2 + (Aq_{r+1}^{[2]} + Bq_r^{[2]})$	$Q^{[3]}/2 + (\mathbf{v}^T q_1^{[2]} + \mathbf{w}^T q_0^{[2]})$	6	1
$\begin{array}{c} \vee \\ \vee \\ \vdots \end{array}$	$q_r^{[4]}$	$Q^{[4]}$	4	6
$\begin{array}{c} \vdots \\ \vee \\ \vdots \end{array}$	$q_r^{[4]}/2 + A(C - (r+1)I)q_{r+1}^{[2]} + B(C - rI)q_r^{[2]}$	$Q^{[4]}/2 + \mathbf{v}^T(C - I)q_1^{[2]} + \mathbf{w}^T C q_0^{[2]}$	8	1
$\begin{array}{c} \vee \\ \vee \\ \vdots \end{array}$	$q_r^{[4]}/3 + (Aq_{r+1}^{[3]} + Bq_r^{[3]})$	$Q^{[4]}/3 + (\mathbf{v}^T q_1^{[3]} + \mathbf{w}^T q_0^{[3]})$	12	2
$\begin{array}{c} \vdots \\ \vdots \\ \vdots \end{array}$	$q_r^{[4]}/6 + (Aq_{r+1}^{[3]} + Bq_r^{[3]})/2 + A^2 q_{r+2}^{[2]} + (AB + BA)q_{r+1}^{[2]} + B^2 q_r^{[2]}$	$Q^{[4]}/6 + (\mathbf{v}^T q_1^{[3]} + \mathbf{w}^T q_0^{[3]})/2 + \mathbf{v}^T(Aq_2^{[2]} + Bq_1^{[2]}) + \mathbf{w}^T(Aq_1^{[2]} + Bq_0^{[2]})$	24	1

Proof. Since $e^0(\emptyset) = 1$ and is zero for all other trees, (2.6) with (2.8a) and (2.7a) implies for each tree with $0 \leq \rho(\tau) < p$ that

$$\begin{aligned} q_r(\tau) &= e^{-r} q(\tau) = e^{-r} [e^C(\tau) - e^{-1}(\tau)\mathbf{u} - e^0(\tau)(\mathbf{1} - \mathbf{u}) - Ae^{-1}\phi'(\tau) - B\phi'(\tau)] \\ &= e^{C-rI}(\tau) - e^{-(r+1)}(\tau)\mathbf{u} - e^{-r}(\tau)(\mathbf{1} - \mathbf{u}) - A\phi'_{r+1}(\tau) - B\phi'_r(\tau), \end{aligned} \quad (2.10)$$

where the last equality results from application of (2.5f), (2.5g) and (2.5h). Next, for the tree with one node, the definition of C with (2.5a) implies

$$\phi(\bullet) = C\mathbf{1}, \quad \phi_r(\bullet) = e^{-r}\phi(\bullet) \equiv e^{-r}e^C(\bullet) = e^{C-rI}(\bullet) = (C - rI)\mathbf{1}.$$

Hence, substitution of each ‘‘bushy’’ tree, $\bar{\tau} = [\bullet^{k-1}]$, into (2.10) gives

$$q_r(\bar{\tau}) = e^{C-rI}(\bar{\tau}) - e^{-(r+1)}(\bar{\tau})\mathbf{u} - e^{-r}(\bar{\tau})(\mathbf{1} - \mathbf{u}) - A\phi'_{r+1}(\bar{\tau}) - B\phi'_r(\bar{\tau}). \quad (2.11)$$

For this tree, (2.5a) and (2.5d) imply that

$$\gamma(\bar{\tau}) = k, \quad e^{C-rI}(\bar{\tau}) = \frac{(C-rI)^k}{k} \mathbb{1} \quad \text{and} \quad \phi'_r(\bar{\tau}) = (\phi_r(\bullet))^{k-1} = (C-rI)^{k-1} \mathbb{1}.$$

Now, substitution of these values into (2.11) proves (2.9a).

For a general tree $\tau = [\tau_1, \tau_2, \dots, \tau_m]$ of order $\rho(\tau) < p$, we prove (2.9b) by the following argument. Let $\bar{\tau} = [\bullet^{\rho(\tau)-1}]$. Then, for each real α , (2.4) and (2.5a) imply that

$$e^\alpha(\tau) = \frac{\rho(\tau)}{\gamma(\tau)} e^\alpha(\bar{\tau}).$$

This result also holds for the *diagonal matrix* $\alpha = C - rI$ since it holds for each $\alpha = c_i - r$.

Now in the difference between (2.10) and $\rho(\tau)/\gamma(\tau)$ times (2.11), this relation cancels many terms, and those remaining can be rearranged to give

$$q_r(\tau) - \frac{\rho(\tau)}{\gamma(\tau)} q_r^{[\rho(\tau)]} = A \left(\frac{\rho(\tau)}{\gamma(\tau)} \phi'_{r+1}(\bar{\tau}) - \phi'_{r+1}(\tau) \right) + B \left(\frac{\rho(\tau)}{\gamma(\tau)} \phi'_r(\bar{\tau}) - \phi'_r(\tau) \right)$$

Now, inserting

$$\frac{\rho(\tau)}{\gamma(\tau)} \phi'_r(\bar{\tau}) = \left(\prod_{j=1}^m \frac{1}{\gamma(\tau_j)} \right) (C-rI)^{\rho(\tau)-1} \mathbb{1} = \left(\prod_{j=1}^m \frac{(C-rI)^{\rho(\tau_j)}}{\gamma(\tau_j)} \right) \mathbb{1}$$

and

$$\phi'_r(\tau) = \prod_{j=1}^m \phi_r(\tau_j) = \prod_{j=1}^m (e^{C-rI}(\tau_j) - q_r(\tau_j))$$

proves (2.9b).

Finally, from (2.8b) and (2.7b), for either the empty tree $\tau = \emptyset$, or otherwise for each tree with $1 \leq \rho(\tau) \leq p$ when $e^0(\tau) = 0$, we obtain

$$\chi(\tau) = e(\tau) - e^{-1}(\tau)\theta - (1-\theta)e^0(\tau) - \mathbf{v}^T e^{-1} \phi'(\tau) - \mathbf{w}^T \phi'(\tau).$$

The observation that for a bushy tree $\bar{\tau} = [\bullet^{k-1}]$, $Q(\bar{\tau}) = \chi(\bar{\tau})$, can be used with the same arguments as above with $r = 0$ to prove (2.9c). \square

3 Computing Local Truncation Coefficients

Our second aim is to find expressions for the error coefficients for a method of order p . It is tempting to use the terms $\chi(\tau)$ for trees satisfying $\rho(\tau) = p + 1$. But in this case, the assumptions below (2.7) are violated. One step of the

method can still be expressed in terms of B-series, but terms of order greater than p have contributions from errors of previous steps. To accommodate this situation, we will write the stage vector Y_n as well as the numerical solution y_{n+1} as step-dependent B-series developed around y_n , that is

$$Y_n = \mathcal{B}(\bar{\phi}_n, y_n; h), \quad y_{n+1} = \mathcal{B}(\bar{\psi}_n, y_n; h). \quad (3.12)$$

We shall assume that *ideal* starting values y_1 and Y_0 are available to propagate the exact numerical solution. (Starting values are ideal if the B-series relationship of y_1 and Y_0 to $y(t_1)$ and $y(t_0 + \mathbf{c}h)$ respectively is exactly the same as that of y_2 and Y_1 to $y(t_2)$ and $y(t_1 + \mathbf{c}h)$; see, for example, [4, page 354].) Recalling that the inverse exactly reverses a computation, (3.12) implies that $y_{n-1} = \mathcal{B}(\bar{\psi}_{n-1}^{-1}, y_n; h)$ and $Y_{n-1} = \mathcal{B}(\bar{\psi}_{n-1}^{-1} \bar{\phi}_{n-1}, y_n; h)$. On insertion of these with (3.12) into (1.2), we obtain exact representations of the numerical solution in terms of B-series about the approximation y_n . Hence, in contrast to (2.7), comparison term by term provide exact formulas for the n -dependent weights as

$$\bar{\phi}_n(\emptyset) = \mathbf{1}, \quad \bar{\phi}_n(\tau) = \bar{\psi}_{n-1}^{-1}(\tau) \mathbf{u} + A(\bar{\psi}_{n-1}^{-1} \bar{\phi}_{n-1})'(\tau) + B \bar{\phi}_n'(\tau), \quad (3.13a)$$

$$\bar{\psi}_n(\emptyset) = 1, \quad \bar{\psi}_n(\tau) = \bar{\psi}_{n-1}^{-1}(\tau) \theta + \mathbf{v}^T (\bar{\psi}_{n-1}^{-1} \bar{\phi}_{n-1})'(\tau) + \mathbf{w}^T \bar{\phi}_n'(\tau), \quad (3.13b)$$

for all trees for each $n = 1, 2, \dots$

However, in order that the numerical values propagate by (1.2) correctly, the *ideal* starting values y_1 and Y_0 for most problems (1.1) will differ from the corresponding exact solution values $y(t_1)$ and $y(t_0 + \mathbf{c}h)$ that they are selected to approximate. (See [13, 12] in which it is shown that selecting starting values equal to exact solution values prevents achieving the design order of a TSRK method when the stage-order is less than $p - 1$.) Moreover, for computation, selection of actual starting values should be coordinated with the order of the method used. In particular, if the starting method satisfies

$$Y_0 = B(\phi, y_0; h) + \mathcal{O}(h^p), \quad y_1 = y(t_0 + h) + \mathcal{O}(h^{p+1}) \quad (3.14)$$

and the method is of order p , then by (2.7) we know that

$$\bar{\phi}_n(\tau) = \phi(\tau) \quad \text{when } \rho(\tau) \leq p - 1, \quad \text{and} \quad \bar{\psi}_n(\tau) = e(\tau) \quad \text{when } \rho(\tau) \leq p. \quad (3.15)$$

If $\tilde{y}(t)$ is the solution to the differential equation (1.1) which satisfies $\tilde{y}(t_n) = y_n$, then the local error $\tilde{y}(t_n + h) - y_{n+1}$ is given by

$$B(e, y_n; h) - B(\bar{\psi}_n, y_n; h) = \sum_{\substack{\tau \in T \\ \rho(\tau) \geq p+1}} \frac{\varepsilon_n(\tau)}{\sigma(\tau)} h^{\rho(\tau)} F(\tau)(y_n)$$

for $n = 0, 1, 2, 3, \dots$ and with $\varepsilon_n(\tau) = e(\tau) - \bar{\psi}_n(\tau)$. We want to find an expression for $\varepsilon_n(\tau)$ or $\bar{\psi}_n(\tau)$, in particular for trees of order $p + 1$.

Now, consider a tree τ of order p . By (3.15), the right hand sides of (2.7a) and (3.13a) are equal, so that

$$\bar{\phi}_n(\tau) = \phi(\tau), \quad \text{for } n \geq 1 \text{ and } \rho(\tau) = p. \quad (3.16)$$

Next, notice that

$$\begin{aligned} 0 &= (\bar{\psi}_n \bar{\psi}_n^{-1})(\tau) = \bar{\psi}_n(\tau) + \bar{\psi}_n^{-1}(\tau) + \sum_{\theta=OST(\tau) \setminus \{\tau, \emptyset\}} \bar{\psi}_n^{-1}(\theta) \prod_{\delta \in \tau \setminus \theta} \bar{\psi}_n(\delta) \\ 0 &= (e e^{-1})(\tau) = e(\tau) + e^{-1}(\tau) + \sum_{\theta=OST(\tau) \setminus \{\tau, \emptyset\}} e^{-1}(\theta) \prod_{\delta \in \tau \setminus \theta} e(\delta). \end{aligned}$$

For a tree τ of order $p+1$, the last sums are equal in these two cases by (3.15), and hence

$$\bar{\psi}_n^{-1}(\tau) = e^{-1}(\tau) + (e(\tau) - \bar{\psi}_n(\tau)), \quad \text{when } \rho(\tau) = p+1.$$

For $n \geq 1$, a similar argument can be used to show

$$(\bar{\psi}_n^{-1} \bar{\phi}_n)'(\tau) = (\bar{\psi}_n^{-1} \bar{\phi}_n')(\tau) = (e^{-1} \phi')(\tau) + \bar{\phi}_n'(\tau) - \phi'(\tau), \quad \text{when } \rho(\tau) = p+1.$$

But for a tree τ of order $p+1$, $\phi_n'(\tau) \neq \phi'(\tau)$ only if $n=0$ and $\tau = [\tau_1]$. Inserting all this into (3.13b), we get

$$\varepsilon_n(\tau) = e(\tau) - \bar{\psi}_n(\tau) = \theta \varepsilon_{n-1}(\tau) + \chi(\tau) + \begin{cases} \mathbf{v}^T(\phi(\tau_1) - \bar{\phi}_0(\tau_1)) & \text{if } \tau = [\tau_1] \text{ and } n = 1, \\ 0 & \text{otherwise.} \end{cases}$$

This linear difference equation can be solved explicitly to summarize the result in the following theorem:

Theorem 3.1. For a TSRK method of order p , with $|\theta| < 1$ and the starting method satisfying (3.15) with $n=0$, the local error can be expressed as

$$\tilde{y}(t_n + h) - y_{n+1} = \mathcal{B}(e, y_n; h) - \mathcal{B}(\bar{\psi}_n, y_n; h) = \mathcal{B}(\varepsilon_n, y_n; h)$$

For trees of order $p+1$, the leading error terms are given by

$$\varepsilon_n(\tau) = \theta^n \varepsilon_0(\tau) + \frac{1 - \theta^n}{1 - \theta} \chi(\tau) + \begin{cases} \theta^{n-1} \mathbf{v}^T(\phi(\tau_1) - \phi_0(\tau_1)) & \text{if } \tau = [\tau_1], \\ 0 & \text{otherwise.} \end{cases}$$

where $\varepsilon_0(\tau)$ and $\phi_0(\tau_1)$ are given by the starting method, and $\chi(\tau)$ is given by (2.9c).

We now observe for methods with $\theta = 0$ (such as those in [11]), values of $\chi(\tau)$ exactly yield leading coefficients of the local truncation error. Otherwise, for values of $\theta \approx 0$, the values $\chi(\tau)/(1-\theta)$ approximate the size of the local truncation error coefficients for later terms ($n \gg 1$) computed. In the next two sections, these observations will illustrate how (2.9c) may be utilized.

4 Design of a MAPLE Code

In this section, we present the background for a MAPLE code that implements the algorithm of Theorem 2.1.

The code begins with a unique enumeration of all rooted trees using an algorithm suggested in 1969 by J.C. Butcher. To each tree τ , we assign $N(\tau)$ to denote its sequence number in the total ordering imposed first by the number of nodes n of a tree, and then for each n , by sequencing trees of n nodes so that in the ordered *root partition* $\tau = [\tau_1, \dots, \tau_m]$,

$$N(\tau_i) \leq N(\tau_j), \text{ if and only if } i < j.$$

This sequencing is *achieved* by cycling through all possible trees identified using the *bi-partition* of τ defined by $\tau = \tau^L \cdot \tau^R \equiv [\tau_1, \dots, \tau_{m-1}, \tau^R]$, so that $\tau^L = [\tau_1, \dots, \tau_{m-1}]$, and the root of τ^L is the root of τ . Here, we denote $\tau^L = \text{leftTree}(\tau)$ and $\tau^R = \text{rightTree}(\tau)$. To select the next tree in sequence, we cycle through previously sequenced *left trees* of $n^L = n - 1, \dots, 1$, nodes in increasing order of their sequence numbers, and for each of these through all *right trees* of $n^R = n - n^L$ nodes in increasing order of their sequence numbers. The next tree in sequence is the first for which $N(\text{rightTree}(\tau^L)) \leq N(\tau^R)$. This will ensure that the subtrees of the root partition of τ occur from left to right in non-decreasing sequence, and hence that each rooted tree will be enumerated exactly once. (There exist other choices for imposing a total ordering on the set of all rooted trees with no more than p nodes, but each will yield the same set of trees perhaps in a different order.) In the MAPLE code, we denote

- $N(\tau) = \text{treeIndex}$
- $N(\tau^L) = \text{leftIndex}$
- $N(\tau^R) = \text{rightIndex}$
- $\rho(\tau) = \text{number of nodes in } \tau = \text{rho}[\text{treeIndex}]$
- Number of trees with $\rho(\tau)$ nodes = $\text{numOc}[\rho(\tau)] - \text{numOc}[\rho(\tau) - 1]$
- Number of edges in highest branch of $\tau = \text{ht}[\text{treeIndex}]$
- Number of trees (τ_j in $\tau = [\tau_1, \dots, \tau_m]$ equal to τ_m) = $\text{mu}[\text{treeIndex}]$
- For a bushy tree with $k = \rho(\tau)$, $q_r^{[k]} = q[k, r, i]$
- Otherwise, $q_r(\tau)_i = \text{sq}[\text{treeIndex}, r, i]$

These are used to tabulate the trees with bi-partition in the sequence specified up to p (or more) nodes. Simultaneously, values of functions γ and σ defined in [2] are computed. Formulas (2.9) from Theorem 1 are written differently as follows for more efficiency and transparency in the code.

Values of the subquadratures $q_r(\tau)$ are computed recursively using stored values of $\phi'_r(\tau_j)$ for subtrees in the root partition $\tau = [\tau_1, \dots, \tau_m]$. For the empty tree, $\phi(\emptyset) = 1$ implies that $\phi_1(\emptyset) = (e^{-1}\phi)(\emptyset) = e^{-1}(\emptyset)\phi(\emptyset) = 1$ so that $\phi_r(\emptyset) = 1$. For the tree with a single node, $\phi_r(\bullet) = (C - rI)\mathbb{1}$, and so values of ϕ'_r may be computed recursively for trees of increasing order and increasing sequence number by

$$\phi'_r(\tau) = \phi'_r(\tau^L) \cdot \phi_r(\tau^R) \quad (4.17a)$$

where

$$\phi_r(\tau^R) = e_r^C(\tau^R) - q_r(\tau^R) \equiv \frac{(C - rI)^{\rho(\tau^R)}}{\gamma(\tau^R)} \mathbb{1} - q_r(\tau^R), \quad (4.17b)$$

and $q_r(\tau^R)$ is computed from (2.9a) for a bushy tree, or else

$$\begin{aligned} q_r(\tau^R) &= \frac{\rho(\tau^R)}{\gamma(\tau^R)} \left(q_r^{[\rho(\tau^R)]} \right. \\ &\quad \left. + A \left[(C - (r+1)I)^{\rho(\tau^R)-1} \mathbb{1} - \frac{\gamma(\tau^R)}{\rho(\tau^R)} \phi'_{r+1}(\tau^R) \right] \right. \\ &\quad \left. + B \left[(C - rI)^{\rho(\tau^R)-1} \mathbb{1} - \frac{\gamma(\tau^R)}{\rho(\tau^R)} \phi'_r(\tau^R) \right] \right) \end{aligned} \quad (4.17c)$$

which is equivalent to (2.9b). For each tree, the order condition may be computed using $Q^{[k]}$ in (1.3b) (denoted otherwise as $(k-1)!C^{[k]}$ in [10]) for a bushy tree from:

$$\begin{aligned} \chi([\bullet^{k-1}]) &= \frac{\rho(\bullet^{k-1})}{\gamma(\bullet^{k-1})} \left[\frac{1}{k} - \left(\frac{\theta(-1)^k}{k} + \mathbf{v}^T \phi'_1(\bullet^{k-1}) + \mathbf{w}^T \phi'_0(\bullet^{k-1}) \right) \right] \\ &\equiv Q^{[k]}, \quad k = 1, \dots, \end{aligned} \quad (4.18a)$$

or for a more general tree from

$$\begin{aligned} \chi(\tau) &= \frac{\rho(\tau)}{\gamma(\tau)} \left\{ Q^{[\rho(\tau)]} + \mathbf{v}^T \left[(C - I)^{\rho(\tau)-1} \mathbb{1} - \frac{\gamma(\tau)}{\rho(\tau)} \phi'_1(\tau) \right] \right. \\ &\quad \left. + \mathbf{w}^T \left[C^{\rho(\tau)-1} \mathbb{1} - \frac{\gamma(\tau)}{\rho(\tau)} \phi'_0(\tau) \right] \right\}. \end{aligned} \quad (4.18b)$$

(A method is of order p if $\chi(\tau) = 0$ for all τ satisfying $\rho(\tau) \leq p$.) This expression is equivalent to (2.9c), and is utilized to develop the MAPLE code.

This code was carefully designed either to generate algebraically all order conditions to any selected order, or else to compute numerically the individual error expressions, $\chi(\tau)$, or their norms of each order for a particular method.

Example 1 (Order Conditions). Place a copy of the code (available from the authors) in a directory as a file named "Trees.TSRK.maple" . Now, in a MAPLE Graphical User Interface write and then execute the following MAPLE script:

```
# Computing ORDER CONDITIONS or ERROR COEFFICIENTS
restart;
currentdir("Your directory name here");
ptest := 6:
# Choose Order Conditions or Error Coefficients:
OrderConditions := true:
AccumulatedErrorCoefficients := true:
if OrderConditions = false then
    methodfile := "method.maple":
    read methodfile;
fi:
read 'Trees.TSRK.maple';
```

The order conditions generated represent those appearing in Tables 1, 3, 4. (The code must be used without changes - eg. such as MAPLE "simplify" or "expand" commands - or else the order conditions will not be correct.)

Example 2 (Error Coefficients). For any TSRK method (see [10, 11, 5]), assign the number of stages s , and specify $theta$, the coefficients $\{\mathbf{u}, \mathbf{v}, \mathbf{w}\}$ as MAPLE data types vector or Vector, and $\{A, B\}$ as MAPLE data types matrix or Matrix in a second file named "method.maple". Now, in the MAPLE script above, select `ptest`, set `OrderConditions:=false:`, and execute the Worksheet again. This yields either individual values, or 2-norms of the error expressions, $\chi(\tau)$, of each order.

Example 3 (Local Truncation Errors for two TSRK Pairs). In 2002, Jackiewicz and Verner [11] derived some five-stage pairs of TSRK pairs of orders 8 and 7. In [5], Chan and Chan showed that by choosing nonzero values of θ and the vector u , they were able to construct four-stage TSRK *methods* of order 8. To compare these new algorithms, we derived a matching method of order 7 for this method in [5] to provide a four-stage TSRK *pair* of orders 8 and 7. For these two pairs, 2-norms of the error expressions χ_9 scaled by σ_9 for method of order 8, and those of $\hat{\chi}_8$ scaled by σ_8 for the imbedded method of order 7 are contrasted in Table 2.

<i>Method</i>	<i>Stages</i>	<i>Stage – order</i>	$ \chi_9/\sigma_9 _2$	$ \hat{\chi}_8/\sigma_8 _2$
<i>JV8(7)</i>	5	5	.7083	.0859
<i>CC8</i>	4	6	.0959	.0156

Table 2: Error Expression Norms for two TSRK pairs of orders 8(7)

The formula derived by Chan and Chan requires fewer stages, and the scaled 2-norm of the leading local truncation error is smaller (since $\theta < 0$ for this method, this advantage is more pronounced). Hence, methods of this type may be expected to be more efficient. Indeed, five steps of the Chan and Chan formula will require the same number of derivative evaluations as four steps of a JV pair. Hence, with the formula for comparison

$$\left\| \frac{\chi_9^{CC}}{\sigma_9} \right\|_2 \left(\frac{1}{5} \right)^8 < \left\| \frac{\chi_9^{JV}}{\sigma_8} \right\|_2 \left(\frac{1}{4} \right)^8,$$

the CC formula can be expected to be *considerably more* efficient with an appropriate implementation.

In summary, we now have an easily-implemented effective tool for comparing two-step Runge–Kutta methods. Even so, the studies in [12, 13] show that a TSRK pair requires more for efficient implementation: in addition to a pair of formulas of successive orders, there is a need to consider how past approximate values should be modified when the stepsize is changed. In those articles, it was shown that special starting values can overcome the inaccuracies if the stepsize is changed. Unfortunately, for those particular types of formulas, past derivative values need to be restarted whenever the stepsize is changed. In an alternative approach, it will be shown as a consequence of the formulation above that other types of TSRK methods can be designed to accommodate stepsize changes without resorting to computing new or revised past values.

5 Conclusions

Jackiewicz and Tracogna [10] developed a theory for obtaining order conditions for two-step Runge–Kutta methods based on the Albrecht approach. To illustrate their approach, they obtained a set of order conditions which were necessary and sufficient for a TSRK method to be of order six. Unfortunately, that approach did not indicate a general algorithm for obtaining those and corresponding order conditions of higher order TSRK methods. Using the algorithm developed in Section 3, we have computed the order conditions for methods up to order six (and higher). Associated tools for computing local truncation error coefficients up to order four are included in Table 1 to illustrate the pattern of these and higher order conditions for such methods. For completeness, we include error expressions for orders 5 and 6 in the Appendix. These or corresponding expressions of higher order may be used with Theorem 3.1 to determine the relative efficiency of a method.

The new form illustrates why choosing high stage-order makes solution of the order conditions easier. By examining Tables 1, 3 and 4, we observe, for a method

of stage-order 3 and order 6 that many of the terms are annihilated. Indeed, it is necessary only to satisfy subquadrature conditions $q^{[k]} = 0$, $k = 1, 2, 3$, quadrature conditions $Q^{[k]} = 0$, $1 \leq k \leq 6$, and to annihilate four particular linear combinations of the subquadrature values $q_r^{[k]}$, $k = 4, 5$ (see (2.8) in [11]). Patterns shown by the expressions found in these tables are replicated in using this process to find order conditions for higher order methods.

The main result of this paper is the construction of a simplified recursion from the composition rule developed by Butcher for characterizing the effect of applying one method after another, or equivalently the composition rule developed by Hairer, Lubich and Wanner for B-series. Here, we have established the particular simplified version for TSRK methods, and illustrated its utility through a MAPLE code which may be used to generate order conditions of arbitrarily high order in a uniform notation, or otherwise to compute individual values or norms of local truncation error expressions for such methods.

6 Appendix

Using graphs of rooted trees, we tabulate additional error coefficient expressions which may be used to obtain the order conditions of orders 5 and 6 in [10]. The forms given illustrate the structure of the algorithm specified in Section 3 used to generate them.







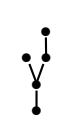
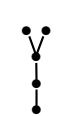

τ	$\chi(\tau)$	$\gamma(\tau)$	$\sigma(\tau)$
	$Q^{[5]}$	5	24
	$Q^{[5]}/2 + \left(\mathbf{v}^T (C - I)^2 q_1^{[2]} + \mathbf{w}^T C^2 q_0^{[2]} \right)$	10	2
	$Q^{[5]}/4 + \left(\mathbf{v}^T \left[\left((\mathbf{c} - \mathbf{1})^2 / 2 \right)^2 - \left((\mathbf{c} - \mathbf{1})^2 / 2 - q_1^{[2]} \right)^2 \right] \right. \\ \left. + \mathbf{w}^T \left[(\mathbf{c}^2 / 2)^2 - \left(\mathbf{c}^2 / 2 - q_0^{[2]} \right)^2 \right] \right)$	20	2
	$Q^{[5]}/3 + \left(\mathbf{v}^T (C - I) q_1^{[3]} + \mathbf{w}^T C q_0^{[3]} \right)$	15	2
	$Q^{[5]}/6 + \left(\mathbf{v}^T (C - I) q_1^{[3]} + \mathbf{w}^T C q_0^{[3]} \right) / 2 \\ + \mathbf{v}^T (C - I) \left[A q_2^{[2]} + B q_1^{[2]} \right] + \mathbf{w}^T C \left[A q_1^{[2]} + B q_0^{[2]} \right]$	30	1
	$Q^{[5]}/4 + \left(\mathbf{v}^T q_1^{[4]} + \mathbf{w}^T q_0^{[4]} \right)$	20	6
	$Q^{[5]}/8 + \left(\mathbf{v}^T q_1^{[4]} + \mathbf{w}^T q_0^{[4]} \right) / 2 \\ + \mathbf{v}^T \left[A(C - 2I) q_2^{[2]} + B(C - I) q_1^{[2]} \right] + \mathbf{w}^T \left[A(C - I) q_1^{[2]} + BC q_0^{[2]} \right]$	40	1
	$Q^{[5]}/12 + \left(\mathbf{v}^T q_1^{[4]} + \mathbf{w}^T q_0^{[4]} \right) / 3 \\ + \left(\mathbf{v}^T \left[A q_2^{[3]} + B q_1^{[3]} \right] + \mathbf{w}^T \left[A q_1^{[3]} + B q_0^{[3]} \right] \right)$	60	2
	$Q^{[5]}/24 + \left(\mathbf{v}^T q_1^{[4]} + \mathbf{w}^T q_0^{[4]} \right) / 6 \\ + \left(\mathbf{v}^T \left[A q_2^{[3]} + B q_1^{[3]} \right] + \mathbf{w}^T \left[A q_1^{[3]} + B q_0^{[3]} \right] \right) / 2 \\ + \mathbf{v}^T \left[A(A q_3^{[2]} + B q_2^{[2]}) + B(A q_2^{[2]} + B q_1^{[2]}) \right] \\ + \mathbf{w}^T \left[A(A q_2^{[2]} + B q_1^{[2]}) + B(A q_1^{[2]} + B q_0^{[2]}) \right]$	120	1

Table 3: Expressions giving Order Conditions of order 5

For tree number 11 recorded in the third row of Table 3, we have changed the notation to emphasize the necessity of doing element-by-element multiplication of a product of vectors. This occurs for each tree with two branches of height at least 2. It will be observed that every term for each error expression will end with a factor of the form $q_r^{[k]}$: all final vectors ending in the form $(\mathbf{c} - r\mathbf{1})^k$ vanish.











τ	$\chi(\tau)$	$\gamma(\tau)$	$\sigma(\tau)$
	$Q^{[6]}$	6	120
	$Q^{[6]}/2 + \left(\mathbf{v}^T (C - I)^3 q_1^{[2]} + \mathbf{w}^T C^3 q_0^{[2]} \right)$	12	6
	$Q^{[6]}/4 + \left(\mathbf{v}^T (C - I) \left[\left((\mathbf{c} - \mathbf{1})^2 / 2 \right)^2 - \left((\mathbf{c} - \mathbf{1})^2 / 2 - q_1^{[2]} \right)^2 \right] \right. \\ \left. + \mathbf{w}^T C \left[(\mathbf{c}^2 / 2)^2 - \left(\mathbf{c}^2 / 2 - q_0^{[2]} \right)^2 \right] \right)$	24	2
	$Q^{[6]}/3 + \left(\mathbf{v}^T (C - I)^2 q_1^{[3]} + \mathbf{w}^T C^2 q_0^{[3]} \right)$	18	4
	$Q^{[6]}/6 + \left(\mathbf{v}^T (C - I)^2 q_1^{[3]} + \mathbf{w}^T C^2 q_0^{[3]} \right) / 2 \\ + \left(\mathbf{v}^T \left[A q_2^{[2]} + B q_1^{[2]} \right] + \mathbf{w}^T \left[A q_1^{[2]} + B q_0^{[2]} \right] \right)$	36	2
	$Q^{[6]}/6 + \left(\mathbf{v}^T \left[(\mathbf{c} - \mathbf{1})^5 / 6 - [(\mathbf{c} - \mathbf{1})^2 / 2 - q_1^{[2]}][(\mathbf{c} - \mathbf{1})^3 / 3 - q_1^{[3]}] \right] \right. \\ \left. + \mathbf{w}^T \left[c^5 / 6 - [c^2 / 2 - q_0^{[2]}][c^3 / 3 - q_0^{[3]}] \right] \right)$	36	2
	$Q^{[6]}/12 + \left(\mathbf{v}^T \left[(\mathbf{c} - \mathbf{1})^5 / 6 - [(\mathbf{c} - \mathbf{1})^2 / 2 - q_1^{[2]}][(\mathbf{c} - \mathbf{1})^3 / 3 - q_1^{[3]}] \right] \right. \\ \left. + \mathbf{w}^T \left[c^5 / 6 - [c^2 / 2 - q_0^{[2]}][c^3 / 3 - q_0^{[3]}] \right] \right) / 2 \\ + \mathbf{v}^T ((C - I)^2 / 2 - q_1^{[2]}) \left[A q_2^{[2]} + B q_1^{[2]} \right] + \mathbf{w}^T (C^2 / 2 - q_0^{[2]}) \left[A q_1^{[2]} + B q_0^{[2]} \right]$	72	1
	$Q^{[6]}/4 + \left(\mathbf{v}^T (C - I) q_1^{[4]} + \mathbf{w}^T C q_0^{[4]} \right)$	24	6
	$Q^{[6]}/8 + \left(\mathbf{v}^T (C - I) q_1^{[4]} + \mathbf{w}^T C q_0^{[4]} \right) / 2 \\ + \mathbf{v}^T (C - I) \left[A(C - 2I) q_2^{[2]} + B(C - I) q_1^{[2]} \right] + \mathbf{w}^T C \left[A(C - I) q_1^{[2]} + B C q_0^{[2]} \right]$	48	1
	$Q^{[6]}/12 + \left(\mathbf{v}^T (C - I) q_1^{[4]} + \mathbf{w}^T C q_0^{[4]} \right) / 3 \\ + \left(\mathbf{v}^T (C - I) \left[A q_2^{[3]} + B q_1^{[3]} \right] + \mathbf{w}^T C \left[A q_1^{[3]} + B q_0^{[3]} \right] \right)$	72	2

Table 4: Expressions giving Order Conditions of order 6

τ	$\chi(\tau)$	$\gamma(\tau)$	$\sigma(\tau)$
	$Q^{[6]}/24 + \left(\mathbf{v}^T(C - I)q_1^{[4]} + \mathbf{w}^T C q_0^{[4]} \right)/6$ $+ \left(\mathbf{v}^T(C - I) \left[Aq_2^{[3]} + Bq_1^{[3]} \right] + \mathbf{w}^T C \left[Aq_1^{[3]} + Bq_0^{[3]} \right] \right)/2$ $+ \mathbf{v}^T(C - I) \left(A \left[Aq_3^{[2]} + Bq_2^{[2]} \right] + B \left[Aq_2^{[2]} + Bq_1^{[2]} \right] \right)$ $+ \mathbf{w}^T C \left(A \left[Aq_2^{[2]} + Bq_1^{[2]} \right] + B \left[Aq_1^{[2]} + Bq_0^{[2]} \right] \right)$	144	1
	$Q^{[6]}/5 + \left(\mathbf{v}^T q_1^{[5]} + \mathbf{w}^T q_0^{[5]} \right)$	30	24
	$Q^{[6]}/10 + \left(\mathbf{v}^T q_1^{[5]} + \mathbf{w}^T q_0^{[5]} \right)/2$ $+ \left(\mathbf{v}^T \left[A(C - 2I)^2 q_2^{[2]} + B(C - I)^2 q_1^{[2]} \right] + \mathbf{w}^T \left[A(C - I)^2 q_1^{[2]} + BC^2 q_0^{[2]} \right] \right)$	60	2
	$Q^{[6]}/20 + \left(\mathbf{v}^T q_1^{[5]} + \mathbf{w}^T q_0^{[5]} \right)/4$ $+ \left(\mathbf{v}^T \left[A \left([(\mathbf{c} - 2\mathbb{1})^2/2]^2 - [(\mathbf{c} - 2\mathbb{1})^2/2 - q_2^{[2]}]^2 \right) \right. \right.$ $\left. \left. + B \left([(\mathbf{c} - \mathbb{1})^2/2]^2 - [(\mathbf{c} - \mathbb{1})^2/2 - q_1^{[2]}]^2 \right) \right] \right)$ $+ \mathbf{w}^T \left[A \left([(\mathbf{c} - \mathbb{1})^2/2]^2 - [(\mathbf{c} - \mathbb{1})^2/2 - q_1^{[2]}]^2 \right) \right.$ $\left. \left. + B \left([c^2/2]^2 - [c^2/2 - q_0^{[2]}]^2 \right) \right] \right)$	120	2
	$Q^{[6]}/15 + \left(\mathbf{v}^T q_1^{[5]} + \mathbf{w}^T q_0^{[5]} \right)/3$ $+ \left(\mathbf{v}^T \left[A(C - 2I)q_2^{[3]} + B(C - I)q_1^{[3]} \right] + \mathbf{w}^T \left[A(C - I)q_1^{[3]} + BCq_0^{[2]} \right] \right)$	90	2
	$Q^{[6]}/30 + \left(\mathbf{v}^T q_1^{[5]} + \mathbf{w}^T q_0^{[5]} \right)/6$ $+ \left(\mathbf{v}^T \left[A(C - 2I)q_2^{[3]} + B(C - I)q_1^{[3]} \right] + \mathbf{w}^T \left[A(C - I)q_1^{[3]} + BCq_0^{[3]} \right] \right)/2$ $+ \mathbf{v}^T \left[A(C - 2I)(Aq_3^{[2]} + Bq_2^{[2]}) + B(C - I)(Aq_2^{[2]} + Bq_1^{[2]}) \right]$ $+ \mathbf{w}^T \left[A(C - I)(Aq_2^{[2]} + Bq_1^{[2]}) + BC(Aq_1^{[2]} + Bq_0^{[2]}) \right]$	180	1
	$Q^{[6]}/20 + \left(\mathbf{v}^T q_1^{[5]} + \mathbf{w}^T q_0^{[5]} \right)/4$ $+ \left(\mathbf{v}^T \left[Aq_2^{[4]} + Bq_1^{[4]} \right] + \mathbf{w}^T \left[Aq_1^{[4]} + Bq_0^{[4]} \right] \right)$	120	6

Table 4(continued): Expressions giving Order Conditions of order 6


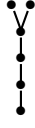

τ	$\chi(\tau)$	$\gamma(\tau)$	$\sigma(\tau)$
	$Q^{[6]}/40 + (\mathbf{v}^T q_1^{[5]} + \mathbf{w}^T q_0^{[5]})/8$ $+ (\mathbf{v}^T [Aq_2^{[4]} + Bq_1^{[4]}] + \mathbf{w}^T [Aq_1^{[4]} + Bq_0^{[4]}])/2$ $+ \mathbf{v}^T \left(A \left[A(C - 3I)q_3^{[2]} + B(C - 2I)q_2^{[2]} \right] + B \left[A(C - 2I)q_2^{[2]} + B(C - I)q_1^{[2]} \right] \right)$ $+ \mathbf{w}^T \left(A \left[A(C - 2I)q_2^{[2]} + B(C - I)q_1^{[2]} \right] + B \left[A(C - I)q_1^{[2]} + BCq_0^{[2]} \right] \right)$	240	1
	$Q^{[6]}/60 + (\mathbf{v}^T q_1^{[5]} + \mathbf{w}^T q_0^{[5]})/12$ $+ (\mathbf{v}^T [Aq_2^{[4]} + Bq_1^{[4]}] + \mathbf{w}^T [Aq_1^{[4]} + Bq_0^{[4]}])/3$ $+ (\mathbf{v}^T [A[Aq_3^{[3]} + Bq_2^{[3]}] + B[Aq_2^{[3]} + Bq_1^{[3]}]]$ $+ \mathbf{w}^T [A[Aq_2^{[3]} + Bq_1^{[3]}] + B[Aq_1^{[3]} + Bq_0^{[3]}]])$	360	2
	$Q^{[6]}/120 + (\mathbf{v}^T q_1^{[5]} + \mathbf{w}^T q_0^{[5]})/24$ $+ (\mathbf{v}^T [Aq_2^{[4]} + Bq_1^{[4]}] + \mathbf{w}^T [Aq_1^{[4]} + Bq_0^{[4]}])/6$ $+ (\mathbf{v}^T [A[Aq_3^{[3]} + Bq_2^{[3]}] + B[Aq_2^{[3]} + Bq_1^{[3]}]]$ $+ \mathbf{w}^T [A[Aq_2^{[3]} + Bq_1^{[3]}] + B[Aq_1^{[3]} + Bq_0^{[3]}]])/2$ $+ \mathbf{v}^T \left[A^2(Aq_4^{[2]} + Bq_3^{[2]}) + (AB + BA)(Aq_3^{[2]} + Bq_2^{[2]}) + B^2(Aq_2^{[2]} + Bq_1^{[2]}) \right]$ $\mathbf{w}^T \left[A^2(Aq_3^{[2]} + Bq_2^{[2]}) + (AB + BA)(Aq_2^{[2]} + Bq_1^{[2]}) + B^2(Aq_1^{[2]} + Bq_0^{[2]}) \right]$	720	1

Table 4(continued): Expressions giving Order Conditions of order 6

References

- [1] J C Butcher. An algebraic theory of integration methods. *Math. Comp.* 26:79–106, 1972.
- [2] J.C. Butcher, *The Numerical Analysis of Ordinary Differential Equations*, John Wiley, Chichester, New York 1987.
- [3] J.C. Butcher, *Numerical Methods for Ordinary Differential Equations*, John Wiley, Chichester, New York 2008.
- [4] J.C. Butcher and S. Tracogna, *Order conditions for two-step Runge–Kutta methods*, *Appl. Numer. Math.* 24 (1997), pp. 351–364.
- [5] T.M.H.Chan and R.P.K. Chan, *A simplified approach to the order conditions of integration methods*, *Computing* 77 (2006), pp. 237–252.

- [6] G.J. Cooper and J.H.Verner, *Explicit Runge–Kutta methods of high order*, SIAM J. Numer. Anal. 9 (1972), pp. 389–405.
- [7] E. Hairer, C. Lubich and G. Wanner, *Geometric Numerical Integration*, Second Edition, Springer-Verlag, Berlin, Heidelberg, 2006.
- [8] E. Hairer and G. Wanner, *On the Butcher group and general multi-value methods*, Computing 13 (1974), pp. 1–15.
- [9] E. Hairer and G. Wanner, *Order conditions for general two-step Runge–Kutta methods*, SIAM J. Numer. Anal. 34 (1997), pp. 2087–2089.
- [10] Z. Jackiewicz and S. Tracogna, *A general class of two-step Runge–Kutta methods for ordinary differential equations*, SIAM J. Num. Anal. 32 (1995), pp. 1390–1427.
- [11] Z. Jackiewicz and J.H.Verner, *Derivation and implementation of two-step Runge–Kutta pairs*, Japan J. Indus. Appl. Math. 19 (2002), pp. 227–248.
- [12] J.H.Verner, *Starting methods for two-step Runge–Kutta methods of stage-order $p-3$ and order 6*, J. Comput. Appl. Math. 185 (2006), pp. 292–307.
- [13] J.H.Verner, *Improved starting methods for two-step Runge–Kutta methods of stage-order $p-3$* , Appl. Numer. Math. 56 (2006), pp. 388–396.