

Order jumps in Spectral Deferred Corrections

Eugen Bronasco^{1*†}, Joscha Fregin^{2*†}, Daniel Ruprecht² and
Gilles Vilmart³

^{1*}Department of Mathematical Sciences, Chalmers University of
Technology and University of Gothenburg, Chalmersplatsen 4,
Gothenburg, SE-412 96, Gothenburg, Sweden.

²Chair Computational Mathematics, Institute of Mathematics,
Hamburg University of Technology, Am Schwarzenberg-Campus 3,
Hamburg, 21073, Hamburg, Germany.

^{3*}Section de mathématiques, Université de Genève, CP 64, Genève 4,
1211, Genève, Switzerland.

*Corresponding author(s). E-mail(s): bronasco@chalmers.se;
joscha.fregin@tuhh.de;

Contributing authors: ruprecht@tuhh.de; gilles.vilmart@unige.ch;

[†]These authors contributed equally to this work.

Abstract

Spectral deferred corrections (SDC) are time-stepping schemes where a low-order integrator is used iteratively to produce a higher order numerical approximation. SDC methods are well-known to obtain at least one additional order of accuracy per iteration. We analyse the phenomenon of “order jumps”, which were observed experimentally in previous work, where the order increases by two in a single iteration. Our analysis relies on Butcher series for the analysis of Runge–Kutta methods (RKM) and includes the case of collocation and discontinuous collocation RKM. We prove that order jumps can be obtained by using appropriate inconsistent, implicit, parallelisable error discretisations. We also investigate the stability properties of order-jump SDC methods and show that, in general, they reduce to that of explicit RKM. Suitable combinations of error discretisations, informed by our analysis, could balance order gains and stability and deliver improved SDC methods in the future.

Keywords: Spectral Deferred Corrections, Runge-Kutta, B-Series, Gauss, Radau, Lobatto

1 Introduction

Spectral deferred corrections (SDC) have received significant amounts of attention since their introduction [1]. Many different variants have emerged, including semi-implicit SDC [2], fast-wave slow-wave SDC [3], GMRES-SDC [4], parallel SDC [5], low storage SDC [6], SDC with penalization of the correction term [7], fast converging SDC [8], SDC for second order initial value problems [9] and Integral Deferred Corrections (IDC) [10]. The name IDC is sometimes used equivalently to SDC, but, in [10], IDC describes a deferred correction variant that uses higher order integrators for the corrections and introduces new stages in between existing ones. For an overview of SDC methods see [11].

SDC is useful for at least moderately stiff problems, since its purely explicit variants are typically not competitive with respect to computational efficiency [12]. Its main advantage is its iterative nature, which allows to construct high-order integrators with special features like semi-implicit splitting [2], multi-rate time-stepping [13] or parallelism [14]. Several works show computational gains from tailored variants of SDC for different applications involving both ordinary and partial differential equations, at least if reasonably high accuracy is required, e.g. for simulations of fast ions in fusion reactors [15], geo-physical flows [16] or computational fluid dynamics [17].

There are at least four critical design choices during the construction of an SDC method.

1. The number and type of quadrature nodes of the underlying collocation method (e.g. Gauss, equidistant or some other set of nodes).
2. The procedure to calculate the initial guesses at the quadrature nodes that SDC requires to start iterating (e.g. simply copying the value brought forward from the previous time step or performing one initial prediction using the sweeper, see next item).
3. The type of sweeper (also sometimes called preconditioner) that updates SDC's stages in every iteration. To construct the sweeper, there exist two main approaches. Originally, SDC used a discretization consistent with a differential equation for the error [1, 10]. However, taking an algebraic perspective of SDC, it was later shown that inconsistent discretizations (that do not correspond to a consistent discretisation of the error equation) can be used to improve convergence [8]. We will refer to the sweeper or *error equation discretization* as EED. Note that some variants of SDC use an EED that changes in each iteration [8, 18].
4. The last step of SDC is to calculate the solution at the endpoint of the timestep from the stage values. Extrapolation is proposed in [1] where the interpolating polynomial defined by the stage values is evaluated. An alternative is to use collocation nodes where the last stage is equal to the end point and to simply copy the solution. Quadrature uses the weights provided by the underlying quadrature rule and applies them to function evaluations of the stages as is usually done in Runge-Kutta methods (RKM).

Links between SDC and RKM have already been established. A relation between IDC and RKM is shown by Christlieb et al. [10]. Butcher series were used to study the convergence of general deferred correction schemes [19] even before the introduction of

SDC.. For selecting SDC methods, a Shu-Osher form has been described [20, 21]. The structure of Butcher tableaux that emerges from this approach has first been described in [22–24] which predates the original introduction of SDC in [1] by almost a decade. Butcher tableaux for IMEX-SDC have been derived in [25].

The present paper shows how to incorporate all those SDC variants in the RKM framework. We derive SDC from the Picard iteration perspective and prove, using Butcher series, that all SDC variants with K iterations have at least order K . Importantly, our proof shows that this is independent of the chosen nodes or error equation discretisation (EED). Our theory also shows that, in some cases, we can guarantee a gain of two orders of convergence per iteration for parallelizable EEDs. This was observed numerically [18] but we provide a rigorous proof. We also show that initialising SDC with a given EED is equivalent to initializing SDC by copying the initial value to all nodes and performing one iteration using the EED.

Related work on the theory of SDC.

Most convergence results for SDC consider either equidistant nodes and consistent EED [10, 19, 26–29] or arbitrary nodes and inconsistent EED [4, 7, 22, 30]. Few works address the combination of arbitrary nodes and consistent EEDs, one example being [3]. Consistent, high order EED are also discussed in [10, 19, 26, 27, 29, 31].

SDC can be initialised using a low order method [1] or by copying the initial state [3, 23]. Using quadrature to obtain the end point can increase the order of the method by one if the order of the underlying method is not yet reached [7]. Arbitrary order one-step and multi-step methods for equidistant nodes as EEDs for SDC were analyzed in [26, 27]. They find that, in contrast to non-equidistant nodes, the order of SDC increases by the order of the EED scheme with each iteration. They also seem to be the first to suggest to change the EED in every iteration, which is later found necessary to construct parallel SDC variants that converge robustly for stiff problems [18].

[30] consider non-equidistant grids and show that the maximum order is bounded by the underlying method and not the number of nodes. Their findings hold also hold for inconsistent EED. [4] prove that GMRES-SDC with s Gauss nodes can achieve order $2s$. A convergence proof for scaled EED, i.e. SDC where the coefficient matrix for the error discretisation is scaled by a constant factor, can be found in [7]. Examples of efficient inconsistent EED's are given in [5, 8]. [28] consider SDC methods where the error is discretised either with implicit, explicit Euler or implicit trapezoidal rule. They prove that SDC's order increases by one per iteration for Euler or two for the trapezoidal rule and also find that the error equation does not need to be discretised with a consistent method. [29] discretise the error equation using arbitrary order RKM to construct consistent EED when using equidistant nodes.

[10] expand the IDC methods introduced in [29] by allowing multistep methods to discretise the error equation. They cast their IDC as RKM and build Butcher tableaux to find the local truncation error and show that, for non-equidistant nodes, a high order ($p \geq 2$) EED does not necessarily increase the order of SDC by more than one per iteration. [19] interprets deferred correction methods with n equidistant nodes as n

steps of an integrator and use the theory of Butcher series to prove the order increase per iteration.

Already in 1996, Jackson et al. [32] used Butcher series to show that, when a fixed-point iteration is used to solve for the stages of an implicit RKM, so-called *iterated RKM*, the order of the overall method increases by one per iteration. This is, in essence, an SDC iteration without an EED and therefore our analysis can be seen as a generalization of their result to general EEDs. Ruprecht and Speck [3] prove that SDC gains one order per iteration when using implicit-explicit methods as EED, but the proof works for standard SDC as well. Their proof breaks down for inconsistent EED because their bound on the supremum norm of the matrix containing the collocation weights is no longer guaranteed to be true. Finally, [22] give a proof that after k iterations SDC yields at least order $p = \min(k, r)$, where r is the minimal stage order of the underlying collocation method.

Order jumps in SDC.

An *order jump* refers to a phenomenon in SDC methods where the order of accuracy of the solution increases by more than one in a single iteration. Such order jumps have been observed in previous works [10, 26, 27, 29]. They prove that order jumps by order p occur, when the error equation is discretised using a p order method and is applied within an SDC method using equidistant node distributions (see e.g. Lemma 3.9, Lemma 3.10 in [10]). Order jumps with non-equidistant grids are briefly discussed by Christlieb et al. [10] (Example 3.13) and were observed numerically for parallel SDC methods [18] where the diagonal EED called MIN-SR-NS is introduced. Bolten and Wimmer [33] observe order jumps when applying SDC to differential-algebraic equations and explicitly acknowledge that “the occurrence of order jumps cannot be explained, and no prediction is yet possible for them”. We illustrate the phenomenon in Tables 1 and 2. Table 1 show the theoretical convergence order of SDC using the MIN-SR-NS diagonal EED by Caklovic et al [18] while Table 2 shows theoretical convergence order for the order-jump EED characterized later in Theorem 2. Convergence orders are computed by assembling the Butcher table for each SDC method, as shown below in (9), and confirming algebraic order conditions using the **Julia** framework by Ketcheson et al. [34]. Rows indicate different number s of RadauIIA nodes while columns indicate the number of SDC iterations k . Order jumps, increases of convergence order by two with a single iteration, are marked in bold. While the diagonal EED only shows a single order jump in iteration $k = s - 1$, the order-jump EED increases order by two in every iteration, up to the order $2s - 1$ of the underlying collocation method. To our knowledge, no theory exists that predicts order jumps for non-equidistant nodes for general EEDs.

Structure of the paper.

This work is organised as follows. In chapter 2 we derive SDC based on a perturbed initial value problem and bridge a connection to RKM. We introduce Butcher series and their connection to SDC. Chapter 3 contains the main theoretical results of this work. Using Butcher series, we prove that an SDC method with K iterations has at least order K . We proceed by using linear theory to find an EED that increases the

$s \setminus k$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	2	3	3	3	3	3	3	3	3	3	3	3	3	3	3
3	1	3	4	5	5	5	5	5	5	5	5	5	5	5	5
4	1	2	4	5	6	7	7	7	7	7	7	7	7	7	7
5	1	2	3	5	6	7	8	9	9	9	9	9	9	9	9
6	1	2	3	4	6	7	8	9	10	11	11	11	11	11	11
7	1	2	3	4	5	7	8	9	10	11	12	13	13	13	13
8	1	2	3	4	5	6	8	9	10	11	12	13	14	15	15

Table 1 Convergence order of SDC using the MIN-SR-NS EED by Caklovic et al. [18]. The method uses s Radau IIA nodes (lines) for a maximum order of $2s - 1$ and k iterations (columns). Thick numbers indicate order jumps.

$s \setminus k$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	2	3	3	3	3	3	3	3	3	3	3	3	3	3	3
3	2	4	5	5	5	5	5	5	5	5	5	5	5	5	5
4	2	4	6	7	7	7	7	7	7	7	7	7	7	7	7
5	2	4	6	8	9	9	9	9	9	9	9	9	9	9	9
6	2	4	6	8	10	11	11	11	11	11	11	11	11	11	11
7	2	4	6	8	10	12	13	13	13	13	13	13	13	13	13
8	2	4	6	8	10	12	14	15	15	15	15	15	15	15	15

Table 2 Convergence order of SDC using the order-jump EED characterized in this paper that ensures two orders per iteration. The method uses s Radau IIA nodes (lines) for a maximum order of $2s - 1$ and k iterations (columns). Thick numbers indicate order jumps.

order of SDC by two in each iteration, independent of the node set. We then derive conditions on EED that increases the order of SDC by two for nonlinear problems if a collocation or discontinuous collocation method is used as the underlying method. We construct a diagonal order-jump EED for parallel SDC methods which performs an order jump on each iteration. In Chapter 4 we discuss linear stability properties of the new methods and possible improvements of the stability domain. We numerically illustrate that the convergence order predicted in our theory is achieved. We then discuss convergence of the iterations of the SDC methods to the underlying method with a given timestep.

2 Spectral deferred corrections

Consider the system of ordinary differential equations (ODE),

$$\frac{du(t)}{dt} = f(u(t)), \quad (1)$$

with vector field¹ $f : \mathbb{R}^d \rightarrow \mathbb{R}^d$ assumed sufficiently differentiable and with initial condition $u(0) = u_0$. Consider a classical implicit Runge-Kutta methods (RKM) of

¹For simplicity of presentation, we assume that the ODE is autonomous. Note that this is without loss of generality, as a non-autonomous system can be made autonomous by considering the augmented system with time equation $\frac{ds(t)}{dt} = 1$, see [35, Section III.1.1].

the form

$$u_{n+1} = u_n + \Delta t \sum_{i=1}^s b_i f(u_n^{[i]}), \quad (2a)$$

$$u_n^{[i]} = u_n + \Delta t \sum_{j=1}^s a_{ij} f(u_n^{[j]}), \quad i = 1, \dots, s, \quad (2b)$$

where $\Delta t = t_{n+1} - t_n$ is the time step and the coefficients b_i and a_{ij} define the RKM and we define $c_i := \sum_{j=1}^s a_{ij}$. The coefficients of the RKM can be compactly arranged in a Butcher tableau

$$\begin{array}{c|c} \mathbf{c} & \mathbf{A} \\ \hline & \mathbf{b}^\top \end{array},$$

where $\mathbf{c} = (c_i)_{i=1}^s$, $\mathbf{b} = (b_i)_{i=1}^s$, and $\mathbf{A} = (a_{ij})_{i,j=1}^s$ [36]. In this paper, we focus on *collocation* Runge-Kutta methods [35, Section II.1.2] with arbitrary collocation nodes $c_1, \dots, c_s$ where the Runge-Kutta coefficients are given by

$$b_i = \int_0^1 l_i(x) dx, \quad a_{ij} = \int_0^{c_i} l_j(x) dx, \quad i, j = 1, \dots, s,$$

where $l_i(x) = \prod_{j \neq i} \frac{x - c_j}{c_i - c_j}$ are the Lagrange interpolation polynomials related to the nodes $c_i, i = 1, \dots, s$. It includes, in particular, the widely used Gauss, Radau IIA, Lobatto IIIA methods which are implicit Runge-Kutta methods of order $2s$, $2s - 1$, and $2s - 2$. Our study also includes the variant of *discontinuous collocation* methods [35, Section II.1.2] which includes Radau IA, Lobatto IIIB, and Lobatto IIIC.

Consider the Dahlquist's test equation $\frac{du(t)}{dt} = \lambda u(t)$ with $\lambda \in \mathbb{C}$ and $\text{Re}(\lambda) \leq 0$. For any method, we can find the stability function $R(z)$, with $z = \lambda \Delta t$, that advances the numerical solution of the Dahlquist equation forward in time $u_{n+1} = R(z)u_n$. Following [37] (Definition 2.1) we call the set $S = \{z \in \mathbb{C} : |R(z)| \leq 1\}$ the stability domain of a given method. If $z \in S$, then the numerical solution $(u_n)_{n \in \mathbb{N}}$ remains bounded for $n \rightarrow \infty$. A method is called A-stable if $S \subset \mathbb{C}^-$. A method is called L-stable if it is A-stable and additionally satisfies [38]

$$\lim_{z \rightarrow \infty} R(z) = 0. \quad (3)$$

A method that satisfies $a_{sj} = b_j, j = 1, \dots, s$ is L-stable [37, Prop. 3.8]. These methods are called stiffly accurate [39]. For instance, Radau IIA and Lobatto IIIA are L-stable and stiffly accurate RKM.

An alternative to using the classical Newton or quasi-Newton method for solving the non-linear system (2b) and compute the internal stages $u_n^{[i]}$ is to consider SDC methods. As emphasized in the introduction, there are many possible variants of SDC methods. Here we consider the following derivation based on the integral formulation

$$u(t) = u(t_n) + \int_{t_n}^t f(u(s)) ds = u(t_n) + \int_{t_n}^t f(\tilde{u}(s)) ds + \int_{t_n}^t (f(u(s)) - f(\tilde{u}(s))) ds \quad (4)$$

of (1), where $\tilde{u}(s)$ stands for the approximation of the solution $u(s)$. Performing a Picard iteration, we obtain the following iteration $u^{k+1}(t)$ to approximate $u(t)$,

$$u^{k+1}(t) = u_n + \int_{t_n}^t f(u^k(s)) ds + \int_{t_n}^t (f(u^{k+1}(s)) - f(u^k(s))) ds, \quad k = 0, 1, 2, \dots, \quad (5)$$

where u_n is a known numerical solution at time t_n . Applying Runge-Kutta discretizations to each of the two above integrals results in the following SDC method,

$$u_n^{[k+1,i]} = u_n + \underbrace{\Delta t \sum_{j=1}^s a_{ij} f(u_n^{[k,j]})}_{\text{collocation term}} + \underbrace{\Delta t \sum_{j=1}^s \tilde{a}_{ij}^k (f(u_n^{[k+1,j]}) - f(u_n^{[k,j]}))}_{\text{correction term}}, \quad (6)$$

where $k = 1, \dots, K$ for a fixed number of iterations K and where $u_n^{[K,i]}$ approximates $u_n^{[i]}$. The Runge-Kutta coefficients $\mathbf{A}_\Delta^k = (\tilde{a}_{ij}^k)_{i,j=1}^s$ correspond to the so called *error equation discretization* (EED) where the corresponding term in (6) can be interpreted as a correction term in the fixed point Picard iterations. The output of the SDC method is then given by

$$u_{n+1} = u_n + \Delta t \sum_{i=1}^s b_i f(u_n^{[K,i]}). \quad (7)$$

The iterative method (6) and (7) defines the class of SDC methods $(\mathbf{A}_\Delta^0, \dots, \mathbf{A}_\Delta^K, \mathbf{A}, \mathbf{b})$ with $\mathbf{A}_\Delta^k = (\tilde{a}_{ij}^k)_{i,j=1}^s$, that we shall analyse in this paper.

Remark 1. Equation (4) can be interpreted as $u(t) = \tilde{u}(t) + \tilde{\delta}(t)$ where $\tilde{u}(t)$ and $\tilde{\delta}(t)$ solve the following partitioned problem,

$$\frac{d\tilde{u}(t)}{dt} = f(t, \tilde{u} + \tilde{\delta}), \quad \frac{d\tilde{\delta}(t)}{dt} = f(t, \tilde{u} + \tilde{\delta}) - f(t, \tilde{u}), \quad (8)$$

where $\tilde{u}(t)$ corresponds to the SDC approximation of $u(t)$ and $\tilde{\delta}(t)$ is the error of the approximate solution $\tilde{u}$ and we assume that $\tilde{\delta}(t_n) = 0$.

The SDC method (6) and (7) is itself a Runge-Kutta method with the following augmented Butcher tableau with $(K+1)s$ internal stages,

$$\begin{array}{c|cccc} \tilde{\mathbf{c}} & \mathbf{A}_\Delta^0 & & & \\ \mathbf{c} & \mathbf{A} - \mathbf{A}_\Delta^1 & \mathbf{A}_\Delta^1 & & \\ \vdots & & & \ddots & \\ \mathbf{c} & & & & \mathbf{A} - \mathbf{A}_\Delta^K & \mathbf{A}_\Delta^K \\ \hline & & & & & \mathbf{b}^\top \end{array}, \quad (9)$$

where $\mathbf{A}$ is the coefficients of the original Runge-Kutta method (2b) and $\tilde{\mathbf{c}} = (\tilde{c}_i)_{i=1}^s$ with $\tilde{c}_i = \sum_{j=1}^s \tilde{a}_{ij}^0$. Natural choices for the EED $\mathbf{A}_\Delta^k$ are based on implicit or explicit

Euler or diagonal to enable parallelization

$$\mathbf{A}_\Delta^k = \begin{pmatrix} \Delta\tau_1 & & \\ \vdots & \ddots & \\ \Delta\tau_1 & \dots & \Delta\tau_s \end{pmatrix}, \quad \mathbf{A}_\Delta^k = \begin{pmatrix} 0 & & \\ \Delta\tau_2 & 0 & \\ \vdots & \ddots & \ddots \\ \Delta\tau_2 & \dots & \Delta\tau_s & 0 \end{pmatrix}, \quad \mathbf{A}_\Delta^k = \alpha_k \text{diag}(\mathbf{c}), \quad (10)$$

with $\Delta\tau_i = c_i - c_{i-1}$ and $\Delta\tau_1 = c_1$, for some constants $\alpha_k \in \mathbb{R}$.

Remark 2. Let $(\mathbf{A}_\Delta^0, \dots, \mathbf{A}_\Delta^K, \mathbf{A}, \mathbf{b})$ define an implicit SDC method. If the Butcher tableau (9) is not singular and satisfies the stiffly accurate condition $a_{si} = b_i$ for $i = 1, \dots, s$, then (3) is satisfied. This is an immediate consequence of the classical result [37, Prop. 3.8] for standard Runge–Kutta methods, see also [22] in the context of SDC. Preserving the A-stability or L-stability of the underlying Runge–Kutta method $(\mathbf{A}, \mathbf{b})$ is however not straightforward when constructing SDC methods, as discussed in Section 4.

Remark 3. For the sake of simplicity, we initialize the SDC method (6), (7) by copying the value brought forward from the previous timestep, so that $u_n^{[0,i]} = u_n$, $i = 1, \dots, s$. However, other choices are possible [40], for example

$$u_n^{[0,i]} = u_n + \Delta t \sum_{j=1}^s \tilde{q}_{ij} f(u_n^{[0,j]})$$

for a choice of coefficients $\tilde{q}_{ij}$ that yields a higher order of approximation of u_{n+1} and satisfies $\mathbf{c} = \tilde{\mathbf{c}}$ to approximate the internal stages. This initialization corresponds to setting $\mathbf{A}_\Delta^0 = (\tilde{q}_{ij})_{i,j=1}^s$ in our formulation of SDC: using (6), we find that

$$u_n^{[1,i]} = u_n + \Delta t \sum_{j=1}^s a_{ij} f(u_n) + \Delta t \sum_{j=1}^s \tilde{q}_{ij} (f(u_n^{[1,j]}) - f(u_n)) = u_n + \Delta t \sum_{j=1}^s \tilde{q}_{ij} f(u_n^{[1,j]}),$$

where $\sum_{j=1}^s a_{ij} = c_i = \tilde{c}_i = \sum_{j=1}^s \tilde{q}_{ij}$.

Remark 4. We emphasize that the assumption $\sum_{j=1}^s \tilde{a}_{ij}^k = c_i$ used in Remark 3 for $k = 0$ is not needed to define the EED $\mathbf{A}_\Delta^k$ with $k \geq 1$. For instance, the second and third choices in (10) do not satisfy this assumption that the EED is consistent with the nodes $\mathbf{c}$ of the original Runge–Kutta method.

Remark 5. There are different possible choices for the weights $\mathbf{b}$ in the linear combination of the stages (7) that computes the approximation u_{n+1} at the end of the timestep. For instance, following [1, Eq. 4.2], one can consider the extrapolation formula $u_{n+1} = \sum_{i=1}^s \ell_i(1) u_n^{[K,i]}$, where ℓ_i are the Lagrange polynomials to the nodes c_i . For stiffly accurate Runge–Kutta methods, like Radau IIA or Lobatto IIIA methods, where $a_{sj} = b_j$, setting $u_{n+1} = u_n^{[K,s]}$ can produce L-stable SDC methods [41, Theorem 3.1]. Using the coefficients b_i of the underlying RKM can be advantageous compared to $u_{n+1} = u_n^{[K,s]}$, because it adds one order of accuracy without extra cost. The outlined

choices can all be realized by choosing different coefficients $\mathbf{b}$ [42]. while the coefficients $\mathbf{A}, \mathbf{A}_{\Delta}^k, \mathbf{c}$ remain the same.

2.1 Butcher series for SDC

Butcher series are a powerful tool for the numerical analysis of Runge-Kutta methods and hence are a natural choice to study of SDC. Expanding in Taylor series as $h \rightarrow 0$ the exact solution and numerical solution after one step and assuming $u(t_n) = u_n$, we obtain

$$\begin{aligned} u_{n+1} &= u_n + \Delta t \sum_{i=1}^s b_i f(u_n) + \Delta t^2 \sum_{i,j=1}^s b_i a_{ij} f'(u_n) f(u_n) + \mathcal{O}(\Delta t^3), \\ u(t_{n+1}) &= u_n + \Delta t f(u_n) + \Delta t^2 \frac{1}{2} f'(u_n) f(u_n) + \mathcal{O}(\Delta t^3). \end{aligned} \quad (11)$$

The difference $u_{n+1} - u(t_{n+1}) = \mathcal{O}(h^{p+1})$ corresponds to the *local error* of an order p method. In particular, we see from (11) that Runge-Kutta method with coefficients $(\mathbf{A}, \mathbf{b})$ is of order (at least) 1 if and only if $\sum_{i=1}^s b_i = 1$ and of order (at least) 2 if and only if

$$\sum_{i=1}^s b_i = 1, \quad \sum_{i,j=1}^s b_i a_{ij} = \frac{1}{2}.$$

Rooted trees [43] systematically represent the elementary differentials which arise in the Taylor expansion of the exact solution and the Runge-Kutta method. The advantage of Butcher series is that they are able to represent both the exact and numerical solution [44] based on the seminal work of J. Butcher [36]. We follow the presentation in [35, Section III].

Definition 1 (rooted tree). *A rooted tree is defined recursively as a vertex called the root and an unordered monomial of rooted trees called the children of the root.*

The set of all rooted trees is denoted by T . Examples of trees with roots at the bottom are



Monomials of trees are called *forests* and denoted by $S(T)$. The empty monomial is written as $\mathbf{1}$. Let $\{\tau_1 \cdots \tau_n\}$ denote the tree with children $\tau_1, \dots, \tau_n \in T$. For example,

$$\{\mathbf{1}\} = \bullet, \quad \{\mathbf{V}\} = \begin{array}{c} \bullet \\ \diagup \quad \diagdown \\ \bullet \quad \bullet \end{array}, \quad \{\bullet, \mathbf{1}\} = \begin{array}{c} \bullet \\ \diagup \\ \bullet \end{array}, \quad \{\mathbf{1}, \mathbf{V}\} = \begin{array}{c} \bullet \\ \diagup \quad \diagdown \\ \bullet \quad \begin{array}{c} \bullet \\ \diagup \quad \diagdown \\ \bullet \quad \bullet \end{array} \end{array}.$$

We also define several functions over T that are useful for a systematic representation of order conditions.

Definition 2 (tree functions). *The size, factorial, symmetry, and elementary differential of a tree $\tau = \{\tau_1^{m_1} \cdots \tau_k^{m_k}\}$ with distinct trees $\tau_1, \dots, \tau_k \in T$ with multiplicities $m_1, \dots, m_k \in \mathbb{N}$ are defined as follows:*

1. the size is defined as the number of vertices in τ , that is, $|\tau| := 1 + \sum_{i=1}^k m_i |\tau_i|$,

2. the factorial is defined as the product of the size of the tree and the factorials of the children of the root, that is, $\tau! = |\tau| \prod_{i=1}^k (\tau_i!)^{m_i}$ with $\bullet! = 1$,
3. the symmetry is defined as the number of automorphisms of the tree, that is, $\sigma(\tau) := \prod_{i=1}^k m_i! \sigma(\tau_i)^{m_i}$ with $\sigma(\bullet) = 1$.
4. the elementary differential $F_{\Delta t f}(\tau) : \mathbb{R}^d \rightarrow \mathbb{R}^d$ is defined as

$$F_{\Delta t f}(\tau) = \Delta t \sum_{i_1, \dots, i_N=1}^d F_{\Delta t f}(\tau_1)^{i_1} \cdots F_{\Delta t f}(\tau_N)^{i_N} \frac{\partial f}{\partial x_{i_1} \cdots \partial x_{i_N}}.$$

where $\tau = \{\tau_1 \cdots \tau_N\}$ with $\tau_1 \cdots \tau_N$ not distinct, and $F_{\Delta t f}(\bullet) := \Delta t f$.

Several examples of the elementary differentials are

$$F_{\Delta t f}(\bullet) = (\Delta t)^2 f' f, \quad F_{\Delta t f}(\begin{smallmatrix} \bullet \\ | \\ \bullet \end{smallmatrix}) = (\Delta t)^3 f' f' f, \quad F_{\Delta t f}(\begin{smallmatrix} \bullet & \bullet \\ \diagdown & \diagup \\ \bullet \end{smallmatrix}) = (\Delta t)^3 f''(f, f),$$

where f' is the first derivative of f (represented by the Jacobian) and f'' is the second derivative of f (recall that f'' is a bilinear map $\mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}^d$). We are now able to represent the Taylor expansion in (11) using formal sums indexed by rooted trees [36, 45–47],

$$\begin{aligned} u_{n+1} &= u_n + \sum_{\tau \in T} \frac{\alpha(\tau)}{\sigma(\tau)} F_{\Delta t f}(\tau)(u_n), \\ u(t_{n+1}) &= u_n + \sum_{\tau \in T} \frac{1}{\tau! \sigma(\tau)} F_{\Delta t f}(\tau)(u_n), \end{aligned} \tag{12}$$

where $\alpha : T \rightarrow \mathbb{R}$ is a functional uniquely defined by the Butcher tableau of the Runge-Kutta method, for example

$$\alpha(\bullet) = \sum_{i=1}^s b_i, \quad \alpha(\begin{smallmatrix} \bullet \\ | \\ \bullet \end{smallmatrix}) = \sum_{i,j=1}^s b_i a_{ij}, \quad \alpha(\begin{smallmatrix} \bullet & \bullet \\ \diagdown & \diagup \\ \bullet \end{smallmatrix}) = \sum_{i,j,k,l=1}^s b_i a_{ij} a_{ik} a_{kl}.$$

The formal sums in (12) are called Butcher series. They give us explicit expressions for order conditions of Runge-Kutta methods of any order p . A Runge-Kutta method is of order p if

$$\alpha(\tau) = \frac{1}{\tau!} \text{ for all } |\tau| \leq p.$$

Definition 3 (Butcher series). *Let $\alpha : T \rightarrow \mathbb{R}$ be a coefficient map over trees, Δt a timestep, and $f : \mathbb{R}^d \rightarrow \mathbb{R}^d$ a vector field. Then,*

$$B_{\Delta t f}(\alpha) = \sum_{\tau \in T} \frac{\alpha(\tau)}{\sigma(\tau)} F_{\Delta t f}(\tau)$$

is a formal sum of vector fields and called a Butcher series.

We note that both u_{n+1} and $u_n^{[i]}$ of a Runge-Kutta method of the form (2a),(2b) can be written using Butcher series as

$$\begin{aligned} u_{n+1} &= u_n + B_{\Delta t f}(\beta)(u_n) \\ u_n^{[i]} &= u_n + B_{\Delta t f}(\beta^{[i]})(u_n), \quad i = 1, \dots, s. \end{aligned}$$

The coefficient maps β and $\beta^{[i]}$ are defined recursively by

$$\begin{aligned} \beta(\{\tau_1 \cdots \tau_n\}) &= \sum_{j=1}^s b_j \beta^{[j]}(\tau_1) \cdots \beta^{[j]}(\tau_n), \\ \beta^{[i]}(\{\tau_1 \cdots \tau_n\}) &= \sum_{j=1}^s a_{ij} \beta^{[j]}(\tau_1) \cdots \beta^{[j]}(\tau_n), \end{aligned} \tag{13}$$

with $\beta(\bullet) = \sum_{j=1}^s b_j$ and $\beta^{[i]}(\bullet) = \sum_{j=1}^s a_{ij}$. In the context of SDC methods defined by (6), (7), let $\alpha^{[K]}$ and $\alpha^{[K,i]}$ denote the final and internal stages of SDC ($\mathbf{A}_\Delta^0, \dots, \mathbf{A}_\Delta^K, \mathbf{A}, \mathbf{b}$) obtained after K iterations, precisely,

$$\begin{aligned} u_n^{[K,i]} &= u_n + B_{\Delta t f}(\alpha^{[K,i]})(u_n), \\ u_{n+1} &= u_n + \sum_{i=1}^s b_i f(u_n^{[K,i]}) = u_n + B_{\Delta t f}(\alpha^{[K]})(u_n). \end{aligned}$$

Combining (13) with the form of the Butcher tableau of SDC in (9), we have the following property of $\alpha^{[K]}$ and $\alpha^{[k,i]}$ with $\mathbf{A}^k = (\tilde{a}_{ij}^k)$ for $k = 1, \dots, K$,

$$\begin{aligned} \alpha^{[K]}(\{\tau_1 \cdots \tau_n\}) &= \sum_{i=1}^s b_i \alpha^{[K,i]}(\tau_1) \cdots \alpha^{[K,i]}(\tau_n), \\ \alpha^{[k,i]}(\{\tau_1 \cdots \tau_n\}) &= \sum_{j=1}^s (a_{ij} - \tilde{a}_{ij}^k) \alpha^{[k-1,j]}(\tau_1) \cdots \alpha^{[k-1,j]}(\tau_n) \\ &\quad + \sum_{j=1}^s \tilde{a}_{ij}^k \alpha^{[k,j]}(\tau_1) \cdots \alpha^{[k,j]}(\tau_n), \end{aligned} \tag{14}$$

as well as $\alpha^{[0,i]}(\tau) = 0$ for all $\tau \in T$, $\alpha^{[K]}(\bullet) = \beta(\bullet)$, and $\alpha^{[k,i]}(\bullet) = \beta^{[i]}(\bullet)$. This property is used extensively in the proofs that follow.

3 Convergence analysis of SDC methods

Consider an SDC method $(\mathbf{A}_\Delta^0, \dots, \mathbf{A}_\Delta^K, \mathbf{A}, \mathbf{b})$. Since SDC approximates the internal stages of the Runge-Kutta method $(\mathbf{A}, \mathbf{b})$, it is not meant to achieve a higher order of accuracy for the exact solution than the RKM. Therefore, we introduce the term *iteration order* of an SDC method $(\mathbf{A}_\Delta^0, \dots, \mathbf{A}_\Delta^K, \mathbf{A}, \mathbf{b})$ as being determined by the local

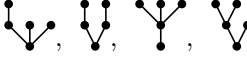
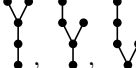
	size 2	size 3	size 4	size 5
height 2				
height 3				
height 4				
height 5				

Table 1 Comparison of the size and height of trees.

error of final stage (7) compared to the Runge-Kutta method $(\mathbf{A}, \mathbf{b})$ which motivates the following definition.

Definition 4 (SDC order). *An SDC method $(\mathbf{A}_\Delta^0, \dots, \mathbf{A}_\Delta^K, \mathbf{A}, \mathbf{b})$ is of iteration order p_K if it coincides with the Runge-Kutta method $(\mathbf{A}, \mathbf{b})$ after one step, up to terms of order $\mathcal{O}(\Delta t^{p_K+1})$, i.e.,*

$$\alpha^{[K]}(\tau) = \beta(\tau), \quad \text{for } |\tau| \leq p_K.$$

In addition, internal stages of an SDC method are of iteration order p_K if

$$\alpha^{[K,i]}(\tau) = \beta^{[i]}(\tau), \quad \text{for } i = 1, \dots, s \quad \text{and } |\tau| \leq p_K. \quad (15)$$

The iteration order should not be confused with the order of the Runge-Kutta method (2a), (2b) or the Runge-Kutta method with the Butcher tableau (9) which, by contrast, is based on the local error compared to the exact solution,

$$\alpha^{[K,i]}(\tau) = \frac{1}{\tau!}, \quad \text{for } i = 1, \dots, s \quad \text{and } |\tau| \leq p_K.$$

Now consider the height of a tree and the corresponding concept of height order of SDC.

Definition 5 (tree height). *The height of a tree $\tau = \{\tau_1 \dots \tau_n\}$ with trees $\tau_1, \dots, \tau_n \in T$ is defined as the number of vertices in the longest path from the root to a leaf, that is, $ht(\tau) := 1 + \max_{i=1}^n ht(\tau_i)$ with $ht(\bullet) := 1$.*

We note that for any tree $\tau \in T$, the size of the tree τ is greater than its height, that is, $|\tau| \geq ht(\tau)$, see Table 1.

Definition 6 (height order). An SDC method $(\mathbf{A}_\Delta^0, \dots, \mathbf{A}_\Delta^K, \mathbf{A}, \mathbf{b}, \mathbf{c})$ has height order h_K if

$$\alpha^{[K]}(\tau) = \beta(\tau), \quad \text{for } i = 1, \dots, s, \quad ht(\tau) \leq h_K.$$

In addition, internal stages of an SDC method are of height order h_K if

$$\alpha^{[K,i]}(\tau) = \beta^{[i]}(\tau), \quad \text{for } i = 1, \dots, s \quad \text{and } ht(\tau) \leq h_K. \quad (16)$$

Since $|\tau| \geq ht(\tau)$ for any given tree τ , meaning its size is greater than its height, the set of all trees of height up to h_k includes the set of all trees up to size h_k . Therefore, an SDC method with height order h_k is (at least) of iteration order h_k . We also note that there is an infinite number of trees of a given height $ht(\tau) \geq 2$, so that the B-series of an SDC method with height order of at least 2 coincides with the B-series of the Runge-Kutta method $(\mathbf{A}, \mathbf{b})$ for an infinite number of trees.

3.1 Convergence order of SDC

The following theorem shows that an SDC iteration with an arbitrary EED increases the height order of SDC by one.

Theorem 1. Let the internal stages of an SDC method $(\mathbf{A}_\Delta^0, \dots, \mathbf{A}_\Delta^K, \mathbf{A}, \mathbf{b})$ be of height order $h_K - 1$. It then holds that

1. the method has height order h_K ,
2. the internal stages of $(\mathbf{A}_\Delta^0, \dots, \mathbf{A}_\Delta^K, \mathbf{A}_\Delta^{K+1}, \mathbf{A}, \mathbf{b}, \mathbf{c})$ are of height order h_K for any $\mathbf{A}_\Delta^{K+1}$.

Proof. Let us start by proving the first statement. We show that $\alpha^{[K]}(\tau) = \beta(\tau)$ for all trees τ of height up to h_K . Let $\tau = \{\tau_1 \cdots \tau_n\}$ be a tree of height up to h_K , $ht(\tau) \leq h_K$, with τ_i being trees of height up to $h_K - 1$, then, use (14),

$$\alpha^{[K]}(\tau) = \sum_{i=1}^s b_i \alpha^{[K,i]}(\tau_1) \cdots \alpha^{[K,i]}(\tau_n)$$

apply the hypothesis $\alpha^{[K,i]}(\tau_j) = \beta^{[i]}(\tau_j)$ for $j = 1, \dots, n$,

$$= \sum_{i=1}^s b_i \beta^{[i]}(\tau_1) \cdots \beta^{[i]}(\tau_n) = \beta(\tau),$$

and the first statement is proved. For the second statement, we apply induction. We use (14) again and note that $\alpha^{[K+1,i]}(\bullet) = \beta^{[i]}(\bullet)$. Let $\tau = \{\tau_1 \cdots \tau_n\}$ be a tree of height up to h_K , $ht(\tau) \leq h_K$, with τ_i being trees of height up to $h_K - 1$, then,

$$\begin{aligned} \alpha^{[K+1,i]}(\tau) &= \sum_{j=1}^s (a_{ij} - \tilde{a}_{ij}^{K+1}) \alpha^{[K,j]}(\tau_1) \cdots \alpha^{[K,j]}(\tau_n) \\ &\quad + \sum_{j=1}^s \tilde{a}_{ij}^{K+1} \alpha^{[K+1,j]}(\tau_1) \cdots \alpha^{[K+1,j]}(\tau_n) \end{aligned}$$

apply the hypothesis $\alpha^{[K,j]}(\tau_j) = \beta^{[i]}(\tau_j)$ and induction hypothesis $\alpha^{[K+1,j]}(\tau_j) = \beta^{[i]}(\tau_j)$ for $j = 1, \dots, n$,

$$= \sum_{j=1}^s a_{ij} \beta^{[j]}(\tau_1) \cdots \beta^{[j]}(\tau_n) = \beta^{[i]}(\tau),$$

and the second statement is proved. $\square$

Note that Theorem 1 remains valid if we replace height order h_K by iteration order p_K .

Corollary 1. Consider an SDC method $(\mathbf{A}_\Delta^0, \mathbf{A}_\Delta^1, \dots, \mathbf{A}_\Delta^K, \mathbf{A}, \mathbf{b})$ with K iterations, then it has height order $h_K \geq K$ and iteration order $p_K \geq K$.

Remark 6. An immediate consequence of Corollary 1 is that we obtain the SDC iteration order $p_K \geq h_K \geq K$ for the SDC method $(\mathbf{A}_\Delta^0, \mathbf{A}_\Delta^1, \dots, \mathbf{A}_\Delta^K, \mathbf{A}, \mathbf{b})$ if we use any of the EEDs introduced in [18]. Choosing

$$\mathbf{A}_\Delta^k = \text{diag}\left(\frac{\mathbf{c}}{s}\right), \quad \mathbf{A}_\Delta^k = \text{diag}\left(\frac{\mathbf{c}}{k}\right) \quad \text{or} \quad \mathbf{A}_\Delta^k = \underset{\mathbf{A}_\Delta^k = \text{diag}(\tilde{\mathbf{c}})}{\text{argmin}} \lambda_{\max}(\mathbf{K}_S^k), \quad (17)$$

generates their MIN-SR-NS, MIN-SR-FLEX and MIN-SR-S variant, respectively.

Remark 7. If an EED based on the trapezoidal method is used by averaging the first two EEDs in (10), the iteration directly after an order jump will fail to increase the order of the SDC method. This can be explained by the fact that the order jumps do not increase the height order of all internal stages. Therefore, this phenomenon does not contradict Theorem 1.

Let us illustrate Theorem 1 with an example of an SDC method with $K = 2$ iterations based on Runge-Kutta method with $s = 2$ stages.

Example 1. Consider an SDC method $(\mathbf{A}_\Delta^0, \mathbf{A}_\Delta^1, \mathbf{A}_\Delta^2, \mathbf{A}, \mathbf{b})$ with $K = 2$ with lower triangular EEDs where $(\mathbf{A}, \mathbf{b})$ is a 2-stage Runge-Kutta method. The Butcher tableau (9) of the SDC method has the form

c_1	$\tilde{a}_{11}^0$				
c_2	$\tilde{a}_{21}^0$	$a_{22}^{0\Delta}$			
c_1	$a_{11} - \tilde{a}_{11}^1$	a_{12}	$\tilde{a}_{11}^1$		
c_2	$a_{21} - \tilde{a}_{21}^1$	$a_{22} - \tilde{a}_{22}^1$	$\tilde{a}_{21}^1$	$\tilde{a}_{22}^1$	
c_1			$a_{11} - \tilde{a}_{11}^2$	a_{12}	$\tilde{a}_{11}^2$
c_2			$a_{21} - \tilde{a}_{21}^2$	$a_{22} - \tilde{a}_{22}^2$	$\tilde{a}_{21}^2 \quad \tilde{a}_{22}^2$
					$b_1 \quad b_2$

Expanding the SDC method in B-series,

$$u_{n+1} = u_n + B_{hf}(\alpha^{[2]})(u_n),$$

and comparing with the B -series of the Runge-Kutta method $(\mathbf{A}, \mathbf{b})$, we obtain,

$$\begin{aligned}
\alpha^{[2]}(\bullet) &= b_1 + b_2 = \beta(\bullet), \\
\alpha^{[2]}(\textcircled{\bullet}) &= b_1(a_{11} - \tilde{a}_{11}^2 + a_{12} + \tilde{a}_{11}^2) + b_2(a_{21} - \tilde{a}_{21}^2 + a_{22} - \tilde{a}_{22}^2 + \tilde{a}_{21}^2 + \tilde{a}_{22}^2) \\
&= b_1(a_{11} + a_{12}) + b_2(a_{21} + a_{22}) = \beta(\textcircled{\bullet}), \\
\alpha^{[2]}(\textcircled{\bullet\bullet}) &= b_1(a_{11} - \tilde{a}_{11}^2 + a_{12} + \tilde{a}_{11}^2)^2 + b_2(a_{21} - \tilde{a}_{21}^2 + a_{22} - \tilde{a}_{22}^2 + \tilde{a}_{21}^2 + \tilde{a}_{22}^2)^2 \\
&= b_1(a_{11} + a_{12})^2 + b_2(a_{21} + a_{22})^2 = \beta(\textcircled{\bullet\bullet}), \\
\alpha^{[2]}(\textcircled{\bullet\bullet\bullet}) &= b_1((a_{11} - \tilde{a}_{11}^2)(a_{11} - \tilde{a}_{11}^1 + a_{12} + \tilde{a}_{11}^1) \\
&\quad + a_{12}(a_{21} - \tilde{a}_{21}^1 + a_{22} - \tilde{a}_{22}^1 + \tilde{a}_{21}^1 + \tilde{a}_{22}^1) \\
&\quad + \tilde{a}_{11}^2(a_{11} - \tilde{a}_{11}^2 + a_{12} + \tilde{a}_{11}^2)) \\
&\quad + b_2((a_{21} - \tilde{a}_{21}^2)(a_{11} - \tilde{a}_{11}^1 + a_{12} + \tilde{a}_{11}^1) \\
&\quad + (a_{22} - \tilde{a}_{22}^2)(a_{21} - \tilde{a}_{21}^1 + a_{22} - \tilde{a}_{22}^1 + \tilde{a}_{21}^1 + \tilde{a}_{22}^1) \\
&\quad + \tilde{a}_{21}^2(a_{11} - \tilde{a}_{11}^2 + a_{12} + \tilde{a}_{11}^2)) \\
&\quad + \tilde{a}_{22}^2(a_{21} - \tilde{a}_{21}^2 + a_{22} - \tilde{a}_{22}^2 + \tilde{a}_{21}^2 + \tilde{a}_{22}^2)) \\
&= b_1((a_{11} - \tilde{a}_{11}^2)(a_{11} + a_{12}) + a_{12}(a_{21} + a_{22}) + \tilde{a}_{11}^2(a_{11} + a_{12})) \\
&\quad + b_2((a_{21} - \tilde{a}_{21}^2)(a_{11} + a_{12}) + (a_{22} - \tilde{a}_{22}^2)(a_{21} + a_{22}) + \tilde{a}_{21}^2(a_{11} + a_{12})) + \tilde{a}_{22}^2(a_{21} + a_{22})) \\
&= b_1(a_{11}(a_{11} + a_{12}) + a_{12}(a_{21} + a_{22})) + b_2(a_{21}(a_{11} + a_{12}) + a_{22}(a_{21} + a_{22})) = \beta(\textcircled{\bullet\bullet\bullet}).
\end{aligned}$$

We see that the equalities are true independent of the entries $\tilde{a}_{ij}^k$ of the EEDs $\mathbf{A}_\Delta^0, \mathbf{A}_\Delta^1, \mathbf{A}_\Delta^2$.

3.2 Analysis of order jumps

In the following, we analyse the phenomenon when the order of accuracy with which the value u_{n+1} is approximated increases by more than one in a single iteration. Such a phenomenon is called an *order jump*.

Recall that the Runge-Kutta coefficients of collocation methods of order p using Gauss, Radau or Lobatto nodes satisfy the following simplifying assumptions [37, 45], see Table 2, called $B(p), C(\eta), D(\zeta)$:

$$\begin{aligned}
B(p) : \quad & \sum_{i=1}^s b_i c_i^{q-1} = \frac{1}{q}, \quad \text{for } q = 1, \dots, p; \\
C(\eta) : \quad & \sum_{j=1}^s a_{ij} c_j^{q-1} = \frac{c_i^q}{q}, \quad \text{for } i = 1, \dots, s, \quad q = 1, \dots, \eta; \\
D(\zeta) : \quad & \sum_{i=1}^s b_i c_i^{q-1} a_{ij} = \frac{b_j}{q} (1 - c_j^q), \quad \text{for } j = 1, \dots, s, \quad q = 1, \dots, \zeta;
\end{aligned} \tag{18}$$

Method	Simplifying assumptions			order
Gauss	$B(2s)$	$C(s)$	$D(s)$	$2s$
Radau IA	$B(2s-1)$	$C(s-1)$	$D(s)$	$2s-1$
Radau IIA	$B(2s-1)$	$C(s)$	$D(s-1)$	$2s-1$
Lobatto IIIA	$B(2s-2)$	$C(s)$	$D(s-2)$	$2s-2$
Lobatto IIIB	$B(2s-2)$	$C(s-2)$	$D(s)$	$2s-2$
Lobatto IIIC	$B(2s-2)$	$C(s-1)$	$D(s-1)$	$2s-2$

Table 2 Simplifying assumptions and corresponding order of collocation and discontinuous collocation methods with s stages.

with $p \leq \eta + \zeta + 1$ and $p \leq 2\eta + 2$. Recall that collocation Runge–Kutta methods satisfy $C(s)$ by definition, while discontinuous collocation Runge–Kutta methods satisfy $C(\eta)$ with $\eta < s$, see [35, Section II.1.2]. In terms of trees, simplifying assumption $B(p)$ is equivalent to the functional β of the collocation or discontinuous collocation method satisfying the order conditions on trees of height 2 up to size p , that is,

$$\beta(\text{tree with root and two children}) = \frac{1}{\text{tree with root and two children}}.$$

Simplifying assumption $C(\eta)$ is equivalent to the functional β being invariant with respect to the transformation on trees that replaces a branch with $q-1$ leaves by q leaves for $1 \leq q \leq \eta$ and divides by q , that is,

$$\beta(\text{tree with root and } q \text{ children}) = \frac{1}{q} \beta(\text{tree with root and } q-1 \text{ children}),$$

where $\square$ is a placeholder for the rest of the tree. Simplifying assumption $D(\zeta)$ is equivalent to the functional β being invariant with respect to the following transformation on trees with $q-1$ leaves at the root,

$$\beta(\text{tree with root and } q \text{ children}) = \frac{1}{q} \beta(\text{tree with root and } q-1 \text{ children}),$$

where $\square$ is a placeholder for one or multiple branches. See [37, 45] for details.

Given an EED $\mathbf{A}_\Delta = (\tilde{a}_{ij})_{i,j=1}^s$, a constant $Y \in \mathbb{R}$, and a set of constants $W = (W_q)_{q=1,\dots,\eta}$, let us introduce assumptions on $\mathbf{A}_\Delta$ defined as

$$C_W(\eta) : \quad \sum_{j=1}^s \tilde{a}_{ij} c_j^{q-1} = c_i^q W_q, \quad \text{for } i = 1, \dots, s, \quad q = 1, \dots, \eta;$$

$$D_Y(\zeta) : \quad \sum_{i=1}^s b_i c_i^{q-1} \tilde{a}_{ij} = \sum_{j=1}^s b_j c_j^q Y, \quad \text{for } j = 1, \dots, s, \quad q = 1, \dots, \zeta.$$

We note that $C(\eta)$ is a particular case of $C_W(\eta)$ with $W_q = \frac{1}{q}$, however, $D(\zeta)$ is not a particular case of $D_Y(\zeta)$. The main result of this section is stated in Theorem 2 which

is proved in Section 3.2.2 and which, given an SDC method $(\mathbf{A}_\Delta^0, \dots, \mathbf{A}_\Delta^{K+1}, \mathbf{A}, \mathbf{b})$, provides conditions on $\mathbf{A}_\Delta^k$ for $k = 1, \dots, K+1$ which guarantee an order jump.

Theorem 2. *Consider SDC with $K+1$ iterations which approximates a Runge–Kutta method of order p satisfying $B(p), C(\eta), D(\zeta)$ with $p = \eta + \zeta$. Let the K^{th} iteration of SDC be of iteration order p_K . Assume there exists a sequence of natural numbers $\eta_k \in \mathbb{N}$ and constants $W_k \in \mathbb{R}^{\eta_k}$ for $k = 1, \dots, K+1$ such that $\mathbf{A}_\Delta^k$ satisfies $C_{\eta_k}(W_k)$ with*

$$W_{K+1, p_K+1} = \frac{1}{p_K + 1}, \quad \text{and} \quad 1 \leq \eta_k \leq \eta, \quad \eta_{k+1} = \eta_k + 1.$$

We have $p_{K+1} = p_K + 2$ if one of the following conditions is satisfied:

1. $p_K < \eta_{K+1}$,
2. $\eta_{K+1} \leq p_K < \eta_{K+1} + \zeta_{K+1}$ and $\mathbf{A}_\Delta^k$ satisfies $D_{Y_k}(\zeta_k)$ for $k = 1, \dots, K+1$ with

$$Y_{K+1} = \frac{1}{p_K + 1}, \quad \text{and} \quad 1 \leq \zeta_k \leq \zeta, \quad \zeta_k \leq \eta_k + 1.$$

The proof of Theorem 2 is presented in Section 3.2.2. Note that, in the statement of Theorem 2, constants $W_{k,q}$ and Y_k for which no assumptions are placed can be arbitrary. We can then systematically build families of EEDs that benefit from repeated order jumps.

Corollary 2. *A diagonal EED $\mathbf{A}_\Delta^k$ satisfies $C_{W_k}(\eta_k)$ and $D_{Y_k}(\zeta_k)$ for any η_k and ζ_k if and only if $\mathbf{A}_\Delta^k = \text{diag}(\mathbf{c})M_k$ for some constant $M_k \in \mathbb{R}$ with*

$$M_k := Y_k = W_{k,1} = \dots = W_{k,\eta_k}.$$

In particular, there exists a unique diagonal EED $\mathbf{A}_\Delta^k = \text{diag}(\mathbf{c})/2k$ for all $k = 1, \dots, K+1$ such that the iteration order of SDC increases by 2 per iteration.

Proof. Let the EED $\mathbf{A}_\Delta^k$ be diagonal for some k . If $\mathbf{A}_\Delta^k = (\tilde{a}_{ij})_{i,j=1}^s$ satisfies $C_{W_k}(\eta_k)$ for any $\eta_k > 1$, then $\tilde{a}_{ii} = c_i W_{k,q}$ for all $q = 1, \dots, \eta_k$. Similarly, if $\mathbf{A}_\Delta^k$ satisfies $D_{Y_k}(\zeta_k)$ for any $\zeta_k > 1$, then, $\tilde{a}_{ii} = c_i Y_k$ with $Y_k = W_{k,1} = \dots = W_{k,\eta_k}$. Therefore, $\mathbf{A}_\Delta^k = \text{diag}(\mathbf{c})M_k$ with $M_k := Y_k = W_{k,1} = \dots = W_{k,\eta_k}$.

Note that to perform an order jump at iteration k with a diagonal EED, we need to choose $M_k = \frac{1}{p_{k-1}+1}$. Assuming we get an order jump at each iteration, we have $p_k = 2k + 1$, which, in the case when all EEDs are diagonal, implies $M_k = 1/2k$. $\square$

Note that the uniqueness of the diagonal EED which results in an order jump follows also from Corollary 4 where a linear problem is considered.

Corollary 3. *Setting $\mathbf{A}_\Delta^k = \text{diag}(\mathbf{c})/s$ for all k , corresponding to the MIN-SR-NS EED introduced by [18], results in an SDC method that performs an order jump on $(s-1)^{\text{th}}$ iteration.*

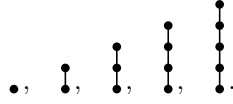
Proof. Note that following the notation of Theorem 2, $\mathbf{A}_\Delta^k$ is a diagonal EED with $M_k = 1/s$ and performs an order jump at k^{th} iteration if $s = p_{k-1} + 1$. Since the order of SDC is increased by 1 per iteration with $p_0 = 1$, we have $p_{k-1} = k$, and the order jump is performed when $k = s - 1$. $\square$

Remark 8. Note that Theorem 2 applies only if $\mathbf{A}$ is a collocation or discontinuous collocation method. However, numerical experiments not reported here suggest that many Runge–Kutta methods can serve as the underlying method. A possible explanation is that most high-order Runge–Kutta methods are constructed using simplifying assumptions analogous to $B(p), C(\eta), D(\zeta)$.

3.2.1 Order jumps for linear problems

In this subsection, we focus on understanding the phenomenon of order jumps of SDC approximating a general Runge–Kutta method. We prove Proposition 3 which introduces a necessary and sufficient condition (19) on the EED $\mathbf{A}_\Delta$ which results in an order jump when the problem is linear, that is, $f : \mathbb{R}^d \rightarrow \mathbb{R}^d$ of (1) is a degree one polynomial. For a general problem, the condition (19) is necessary, but not sufficient.

A bamboo tree is a tree which does not have any branches. It is the only tree whose size is equal to its height see Table 1 and the only tree which appears in the B-series of Runge–Kutta methods and the exact solution when the problem is linear. A list of all bamboo trees up to size (and height) 5 can be found below,



We use bamboo trees to derive the condition on EED $\mathbf{A}_\Delta$ for the order to be increased by 2 per iteration. Let the bamboo tree of height h be denoted by



Let us define the coefficient error of the i^{th} stage of SDC with K iterations to be

$$\epsilon^{[K,i]} := \beta^{[i]} - \alpha^{[K,i]}.$$

We use (14) to prove Proposition 3 which derives a condition on the EED $\mathbf{A}_\Delta$ that results in an order jump. Note that for linear problems, order and height order agree.

Proposition 3. Let the internal stages of SDC with K iterations be of iteration order h_K . An EED $\mathbf{A}_\Delta^{K+1} = (\tilde{a}_{ij})_{i,j=1}^s$ which results in SDC with $K+1$ iterations and internal stages of iteration order h_K+2 satisfies the following condition

$$\sum_{j=1}^s \tilde{a}_{ij} \epsilon^{[K,j]} \left(\begin{array}{c} \bullet \\ \vdots \\ \bullet \end{array} \right)_{h_K+1} = \sum_{j=1}^s a_{ij} \epsilon^{[K,j]} \left(\begin{array}{c} \bullet \\ \vdots \\ \bullet \end{array} \right)_{h_K+1}. \quad (19)$$

For a linear problem, condition (19) is sufficient to guarantee an order jump.

Proof. Assume iteration $K + 1$ performs an order jump, then, the coefficient map of the resulting SDC method must satisfy

$$\alpha^{[K+1,i]}(\overset{\bullet}{\underset{\bullet}{\vdots}}_{h_K+2}) = \beta^{[i]}(\overset{\bullet}{\underset{\bullet}{\vdots}}_{h_K+2}).$$

We use (14) to derive the condition on $\tilde{a}_{ij}$,

$$\sum_{j=1}^s (a_{ij} - \tilde{a}_{ij}) \alpha^{[K,j]}(\overset{\bullet}{\underset{\bullet}{\vdots}}_{h_K+1}) + \sum_{j=1}^s \tilde{a}_{ij} \alpha^{[K+1,j]}(\overset{\bullet}{\underset{\bullet}{\vdots}}_{h_K+1}) = \beta^{[i]}(\overset{\bullet}{\underset{\bullet}{\vdots}}_{h_K+2}),$$

We recall Theorem 1 and group the $\tilde{a}_{ij}$ terms which gives,

$$\sum_{j=1}^s a_{ij} \alpha^{[K,j]}(\overset{\bullet}{\underset{\bullet}{\vdots}}_{h_K+1}) + \sum_{j=1}^s \tilde{a}_{ij} \epsilon^{[K,j]}(\overset{\bullet}{\underset{\bullet}{\vdots}}_{h_K+1}) = \beta^{[i]}(\overset{\bullet}{\underset{\bullet}{\vdots}}_{h_K+2}),$$

We use (13) to obtain

$$\sum_{j=1}^s \tilde{a}_{ij} \epsilon^{[K,j]}(\overset{\bullet}{\underset{\bullet}{\vdots}}_{h_K+1}) = \sum_{j=1}^s a_{ij} \epsilon^{[K,j]}(\overset{\bullet}{\underset{\bullet}{\vdots}}_{h_K+1}),$$

and the statement is proved. $\square$

Corollary 4. *If the EED $\mathbf{A}_\Delta$ is diagonal, the only EED that satisfies (19) is*

$$\tilde{a}_{ii} = \sum_{j=1}^s a_{ij} \frac{\epsilon^{[K,j]}}{\epsilon^{[K,i]}}(\overset{\bullet}{\underset{\bullet}{\vdots}}_{h_K+1}).$$

Note that we cannot guarantee that the unique diagonal EED which results in an order 2 jump also produces an order 3 jump, see Example 2. Therefore, for general SDC methods, an order jump by two is the maximum that can be guaranteed by a diagonal EED. General EEDs have additional degrees of freedom which might permit the construction of EEDs resulting in high order jumps.

Example 2. *Suppose an SDC method with K iterations, internal stages of iteration order h_K , and diagonal EEDs $\mathbf{A}_\Delta^k = (\tilde{a}_{ij}^k)_{i,j=1}^s$ such that $\tilde{a}_{ii}^k = l(k,i)c_i$ for some constants $l(k,i) \in \mathbb{R}$ and $\tilde{a}_{ij}^k = 0$ for $i \neq j$. We require*

$$\alpha^{[K+1,i]}(\overset{\bullet}{\underset{\bullet}{\vdots}}_{h_K+2}) = \beta^{[i]}(\overset{\bullet}{\underset{\bullet}{\vdots}}_{h_K+2})$$

to increase the iteration order of the internal stages by 2. From (14) we obtain

$$\alpha^{[k+1,i]}(\overset{\bullet}{\underset{\bullet}{\vdots}}_{h_K+2}) = \sum_{j=1}^s a_{ij} \alpha^{[k,j]}(\overset{\bullet}{\underset{\bullet}{\vdots}}_{h_K+1}) - \tilde{a}_{ii}^{k+1} \alpha^{[k,i]}(\overset{\bullet}{\underset{\bullet}{\vdots}}_{h_K+1}) + \tilde{a}_{ii}^{k+1} \alpha^{[k+1,i]}(\overset{\bullet}{\underset{\bullet}{\vdots}}_{h_K+1}).$$

Let's begin with the first iteration, given a copied initial state, i.e. $u_n^{[0,i]} = u_n$. We then require

$$\alpha^{[0+1,i]}(\mathbf{I}) = \beta^{[i]}(\mathbf{I}).$$

From above we have

$$\alpha^{[0+1,i]}(\mathbf{I}) = \sum_{j=1}^s a_{ij} \alpha^{[0,j]}(\bullet) - \tilde{a}_{ii}^1 \alpha^{[0,i]}(\bullet) + \tilde{a}_{ii}^1 \alpha^{[1,i]}(\bullet).$$

Since $\alpha^{[0,j]}(\bullet) = 0$ and $\alpha^{[1,i]}(\bullet) = \beta^{[i]}(\bullet) = c_i$ we obtain $\alpha^{[0+1,i]}(\mathbf{I}) = l(1, i) c_i^2$, which is equivalent to

$$l(1, i) = \frac{\sum_{j=1}^s a_{ij} c_j}{c_i^2} = \frac{1}{2} \frac{c_i^2}{c_i^2} = \frac{1}{2} = l(1),$$

where the second equality uses the assumption that the Runge–Kutta method $(\mathbf{A}, \mathbf{b})$ satisfies $C(2)$. Hence to increase the order by 2 in the first iteration we require $\tilde{a}_{ii}^1 = c_i/2$. Let us check that we cannot perform an order 3 jump,

$$\alpha^{[1,i]}(\mathbf{I}) = l(1, i) c_i \alpha^{[1,i]}(\mathbf{I}) = c_i^3 l(1, i)^2 = \frac{c_i^3}{4} \neq \beta^{[i]}(\mathbf{I}),$$

if we assume the Runge–Kutta method $(\mathbf{A}, \mathbf{b})$ satisfies $C(3)$. Next, consider an SDC method with iteration order 1 after the first iteration and the same assumptions as above. We require

$$\alpha^{[1+1,i]}(\mathbf{I}) = \beta^{[i]}(\mathbf{I}). \quad (20)$$

We compute the left-hand side by using (14) as follows,

$$\begin{aligned} \alpha^{[1+1,i]}(\mathbf{I}) &= \sum_{j=1}^s a_{ij} \alpha^{[1,j]}(\mathbf{I}) - \tilde{a}_{ii}^2 \alpha^{[1,i]}(\mathbf{I}) + \tilde{a}_{ii}^2 \alpha^{[2,i]}(\mathbf{I}) \\ &= \sum_{j=1}^s a_{ij} l(1) c_j^2 - l(2, i) l(1) c_i^3 + \frac{1}{2} l(2, i) c_i^3 \end{aligned}$$

where we used Theorem 1 and assumption $\beta^{[i]}(\mathbf{I}) = c_i^2/2$ implied by $C(2)$. Assuming the Runge–Kutta method $(\mathbf{A}, \mathbf{b})$ satisfies $C(3)$, identity (20) becomes,

$$l(1) \frac{c_i^3}{3} + l(2, i) \left(\frac{c_i^3}{2} - l(1) c_i^3 \right) = \frac{c_i^3}{6}.$$

There are two cases to be taken care of. The first one is $l(1) = 1/2$. Then the equation above holds true and $l(2, i)$ vanishes, that is, the SDC method is of iteration order 3 due to the order jump in the first iteration regardless of the EED in the second iteration. In the second case $l(1) \neq 1/2$. Solving for $l(2, i)$ yields

$$l(2, i) = \frac{1}{3}.$$

We note that the nodes c_i and the constant $l(1)$ cancel, and hence, $l(2)$ is independent of constants previously chosen. We finish this example by describing how to go from second to fourth order in the second iteration using a diagonal EED. We require

$$\alpha^{[2,i]}(\text{tree}) = \sum_{j=1}^s a_{ij} \alpha^{[1,j]}(\text{tree}) - \tilde{a}_{ii}^2 \alpha^{[1,i]}(\text{tree}) + \tilde{a}_{ii}^2 \alpha^{[2,i]}(\text{tree}) \quad (21)$$

Since $\alpha^{[2,i]}(\text{tree}) = c_i^3/6$ due to Theorem 1, we only need to find

$$\begin{aligned} \alpha^{[1,i]}(\text{tree}) &= \sum_{j=1}^s a_{ij} \alpha^{[0,j]}(\text{tree}) - \tilde{a}_{ii}^1 \alpha^{[0,i]}(\text{tree}) + \tilde{a}_{ii}^1 \alpha^{[1,i]}(\text{tree}) \\ &= \tilde{a}_{ii}^1 \frac{c_i^2}{2} = \frac{c_i^3}{4}, \end{aligned}$$

where in the last equality we used $\tilde{a}_{ii}^1 = c_i/2$. Hence (21) can be simplified to

$$\begin{aligned} \alpha^{[2,i]}(\text{tree}) &= \sum_{j=1}^s a_{ij} \frac{c_j^3}{4} - \tilde{a}_{ii}^2 \frac{c_i^3}{4} + \tilde{a}_{ii}^2 \frac{c_i^3}{6} \\ &= \frac{c_i^4}{16} + \tilde{a}_{ii}^2 \left(\frac{c_i^3}{6} - \frac{c_i^3}{4} \right) \end{aligned}$$

Once again letting $\tilde{a}_{ii}^2 = l(2, i)c_i$ and requiring $\alpha^{[2,i]}(\text{tree}) = c_i^4/24$, yields

$$l(2, i) = l(2) = \frac{1}{4}.$$

A pattern seems to be emerging that requires $l(k) = 1/(2k)$ in order to obtain the jumps in each iteration. Continuing in this manner reveals an emerging pattern that requires $l(k) = 1/(2k - v)$ in order to increase the order by 2 in the next iteration, where v is number of iterations without order jumps.

For non-linear problems, the condition (19) is sufficient to perform an order jump only once. This is detailed in Proposition 4, see also Corollary 1.

Proposition 4. *SDC with K iterations and internal stages of height order h_k , performing an iteration with an EED satisfying (19), results in SDC with $K+1$ iterations and internal stages of height order $h_{K+1} = h_K + 1$ and iteration order $p_{K+1} \geq h_k + 2$.*

Proof. We use Theorem 1 and the relation between height order and iteration order to see that $p_{K+1} \geq h_{K+1} = h_K + 1$. This implies that, $\alpha^{[K+1,i]}(\tau) = \beta^{[i]}(\tau)$ for all $ht(\tau) \leq h_K + 1$. We note that the only tree of size $h_K + 2$ whose height is greater than $h_K + 1$ is the bamboo tree. Condition (19) ensures that $\alpha^{[K+1,i]}(\tau) = \beta^{[i]}(\tau)$ for τ being a bamboo tree of size $h_K + 2$ which implies iteration order $p_{K+1} \geq h_K + 2$. $\square$

3.2.2 Order jumps for non-linear problems

In this subsection, we derive the necessary and sufficient conditions for an EED $\mathbf{A}_\Delta^k$ to produce an order jump using the simplifying assumptions $B(p), C(\eta), D(\zeta)$ of Gauss, Lobatto, and Radau collocation and discontinuous collocation methods described in the beginning of Section 3.2.

Lemma 1. *Consider an SDC with K iterations approximating a Runge–Kutta method satisfying $C(\eta)$. Given $\eta_k \in \mathbb{R}$ and $W_k \in \mathbb{R}^{\eta_k}$, assume $\mathbf{A}_\Delta^k$ satisfies $C_{W_k}(\eta_k)$ with*

$$1 \leq \eta_k \leq \eta, \quad \eta_{k+1} \leq \eta_k + 1, \quad \text{for } k = 1, \dots, K.$$

Then, given a tree τ with $|\tau| \leq \eta_K$, we have,

$$\alpha^{[K,i]}(\tau) = c_i^{|\tau|} \omega_K(\tau), \quad (22)$$

for some constant $\omega_K(\tau)$ which depends on the tree τ and W_k for $k = 1, \dots, K$.

Proof. Let us prove the statement for $K = 1$ taking into account that $\mathbf{A}_\Delta^0 = \mathbf{0}$ ($\alpha^{[0,i]} = 0$), and, therefore, given a tree $\tau = \{\tau_1 \dots \tau_n\}$ such that $1 < |\tau| \leq \eta_1$, we use (14) to obtain

$$\alpha^{[1,i]}(\tau) = \sum_{j=1}^s \tilde{a}_{ij}^1 \alpha^{[1,i]}(\tau_1) \dots \alpha^{[1,i]}(\tau_n) = c_i^{|\tau|} \prod_{l=1}^{|\tau|} W_{1,l},$$

where we used induction on the size of the tree and $C_{W_1}(\eta_1)$ to prove the statement for $K = 1$. We note that for any K , $\alpha^{[K,i]}(\bullet) = c_i$. Let $\tau = \{\tau_1 \dots \tau_n\}$ such that $|\tau| \leq \eta_K$, we use (14) to write

$$\alpha^{[K,i]}(\tau) = \sum_{j=1}^s (a_{ij} - \tilde{a}_{ij}^K) \alpha^{[K-1,j]}(\tau_1) \dots \alpha^{[K-1,j]}(\tau_n) + \sum_{j=1}^s \tilde{a}_{ij}^K \alpha^{[K,j]}(\tau_1) \dots \alpha^{[K,j]}(\tau_n)$$

by induction on the size of the tree and K , we get

$$= \sum_{j=1}^s (a_{ij} - \tilde{a}_{ij}^K) c_j^{|\tau|-1} \omega_{K-1}(\tau_1) \dots \omega_{K-1}(\tau_n) + \sum_{j=1}^s \tilde{a}_{ij}^K c_j^{|\tau|-1} \omega_K(\tau_1) \dots \omega_K(\tau_n)$$

using $C(\eta)$ and $C_{W_K}(\eta_K)$, we get

$$= c_i^{|\tau|} \left(\left(\frac{1}{|\tau|} - W_{K,|\tau|} \right) \omega_{K-1}(\tau_1) \dots \omega_{K-1}(\tau_n) + W_{K,|\tau|} \omega_K(\tau_1) \dots \omega_K(\tau_n) \right).$$

We obtain the statement (22) with

$$\omega_K(\tau) := \left(\frac{1}{|\tau|} - W_{K,|\tau|} \right) \omega_{K-1}(\tau_1) \dots \omega_{K-1}(\tau_n) + W_{K,|\tau|} \omega_K(\tau_1) \dots \omega_K(\tau_n),$$

and the proof is finished. $\square$

Lemma 2. Consider SDC with K iterations which approximates a Runge–Kutta method of order p satisfying $B(p), C(\eta), D(\zeta)$ with $p = \eta + \zeta$. Assume there exists a sequence of natural numbers $\eta_k \in \mathbb{N}$ and constants $W_k \in \mathbb{R}^{\eta_k}$ for $k = 1, \dots, K$ such that $\mathbf{A}_\Delta^k$ satisfies $C_{\eta_k}(W_k)$ with

$$1 \leq \eta_k \leq \eta, \quad \eta_{k+1} = \eta_k + 1.$$

Then, given $\tau = \{\tau_1 \cdots \tau_n\}$, we have,

$$\alpha^{[K]}(\tau) = \frac{1}{|\tau|} \omega_K(\tau_1) \cdots \omega_K(\tau_n), \quad (23)$$

if one of the following conditions is satisfied:

1. $|\tau| \leq \eta_K + 1$,
2. $\eta_K + 1 < |\tau| \leq \eta_K + \zeta_K + 1$ and $\mathbf{A}_\Delta^k$ satisfies $D_{Y_k}(\zeta_k)$ with $Y_k \in \mathbb{R}$ where,

$$1 \leq \zeta_k \leq \zeta, \quad \zeta_k \leq \eta_k + 1, \quad \text{and} \quad k = 1, \dots, K.$$

Proof. For a tree $\tau = \{\tau_1 \cdots \tau_n\}$ of size $|\tau| \leq \eta_K + 1$, we use Lemma 1 and $B(p)$ to obtain,

$$\begin{aligned} \alpha^{[K]}(\tau) &= \sum_{i=1}^s b_i \alpha^{[K,i]}(\tau_1) \cdots \alpha^{[K,i]}(\tau_n) \\ &= \sum_{i=1}^s b_i c_i^{|\tau|-1} \omega_K(\tau_1) \cdots \omega_K(\tau_n) = \frac{1}{|\tau|} \omega_K(\tau_1) \cdots \omega_K(\tau_n), \end{aligned} \quad (24)$$

which proves the statement. For any tree $\tau = \{\tau_1 \cdots \tau_n\}$ of size $\eta_K + 1 < |\tau| \leq \eta_K + \zeta_K + 1$, if all τ_i have sizes less or equal to η_K , then (24) proves the statement.

Assume $\tau = \{\tau_1 \cdots \tau_n\}$ of size $\eta_K + 1 < |\tau| \leq \eta_K + \zeta_K + 1$ and assume there exists a tree $\gamma = \tau_j$ for some j such that $|\gamma| \geq \eta_K + 1$, then, $|\tau_1 \cdots \tau_{j-1} \tau_{j+1} \cdots \tau_n| \leq \zeta_K - 1 \leq \eta_K$, which implies that γ is the only tree τ_i such that $|\gamma| \geq \eta_K + 1$. Let $q := |\tau_1 \cdots \tau_{j-1} \tau_{j+1} \cdots \tau_n|$. We use Lemma 1 to obtain

$$\begin{aligned} \alpha^{[K]}(\tau) &= \sum_{i=1}^s b_i \alpha^{[K,i]}(\tau_1) \cdots \alpha^{[K,i]}(\tau_n) \\ &= \sum_{i=1}^s b_i c_i^q \alpha^{[K,i]}(\gamma) \omega_K(\tau_1) \cdots \omega_K(\tau_{j-1}) \omega_K(\tau_{j+1}) \cdots \omega_K(\tau_n) \\ &= \alpha^{[K]}(\{\bullet^q \gamma\}) \omega_K(\tau_1) \cdots \omega_K(\tau_{j-1}) \omega_K(\tau_{j+1}) \cdots \omega_K(\tau_n). \end{aligned}$$

This implies that it is enough to prove the statement for trees of the form $\{\bullet^q \gamma\}$. We have $q < \zeta_K \leq \zeta$, therefore, we can apply $D_{Y_K}(\zeta_K)$ and $D(\zeta)$. Let $\gamma = \{\gamma_1 \cdots \gamma_m\}$,

then, using (14),

$$\begin{aligned}\alpha^{[K]}(\{\bullet^q \gamma\}) &= \sum_{i,j=1}^s b_i c_i^q a_{ij} \alpha^{[K-1,j]}(\gamma_1) \cdots \alpha^{[K-1,j]}(\gamma_m) \\ &\quad - \sum_{i,j=1}^s b_i c_i^q \tilde{a}_{ij}^K \alpha^{[K-1,j]}(\gamma_1) \cdots \alpha^{[K-1,j]}(\gamma_m) \\ &\quad + \sum_{i,j=1}^s b_i c_i^q \tilde{a}_{ij}^K \alpha^{[K,j]}(\gamma_1) \cdots \alpha^{[K,j]}(\gamma_m)\end{aligned}$$

using $D(\zeta)$ and $D_{Y_K}(\zeta_K)$, we get

$$\begin{aligned}\alpha^{[K]}(\{\bullet^q \gamma\}) &= \frac{1}{q+1} \sum_{i=1}^s b_i (1 - c_i^{q+1}) \alpha^{[K-1,i]}(\gamma_1) \cdots \alpha^{[K-1,i]}(\gamma_m) \\ &\quad - \sum_{i=1}^s b_i c_i^{q+1} Y_K \alpha^{[K-1,i]}(\gamma_1) \cdots \alpha^{[K-1,i]}(\gamma_m) \\ &\quad + \sum_{i=1}^s b_i c_i^{q+1} Y_K \alpha^{[K,i]}(\gamma_1) \cdots \alpha^{[K,i]}(\gamma_m)\end{aligned}$$

the definition of $\alpha^{[K-1]}$ and $\alpha^{[K]}$ gives

$$\begin{aligned}\alpha^{[K]}(\{\bullet^q \gamma\}) &= \frac{1}{q+1} (\alpha^{[K-1]}(\gamma) - \alpha^{[K-1]}(\{\bullet^{q+1} \gamma_1 \cdots \gamma_m\})) \\ &\quad - Y_K \alpha^{[K-1]}(\{\bullet^{q+1} \gamma_1 \cdots \gamma_m\}) + Y_K \alpha^{[K]}(\{\bullet^{q+1} \gamma_1 \cdots \gamma_m\})\end{aligned}$$

we use induction on the size of $\gamma_1 \cdots \gamma_m$ and the fact that $\omega_k(\bullet) = 1$ to get

$$\begin{aligned}\alpha^{[K]}(\{\bullet^q \gamma\}) &= \left(\frac{1}{(q+1)|\gamma|} - \frac{1}{(q+1)(|\gamma|+q+1)} \right. \\ &\quad \left. - \frac{1}{|\gamma|+q+1} Y_K \omega_{K-1}(\gamma_1) \cdots \omega_{K-1}(\gamma_m) \right. \\ &\quad \left. + \frac{1}{|\gamma|+q+1} Y_K \omega_K(\gamma_1) \cdots \omega_K(\gamma_m) \right) \\ &= \frac{1}{|\gamma|+q+1} \left(\left(\frac{|\gamma|+q+1}{(q+1)|\gamma|} - \frac{1}{q+1} - Y_K \right) \omega_{K-1}(\gamma_1) \cdots \omega_{K-1}(\gamma_m) \right. \\ &\quad \left. + Y_K \omega_K(\gamma_1) \cdots \omega_K(\gamma_m) \right)\end{aligned}$$

which reduces to

$$\begin{aligned}\alpha^{[K]}(\{\bullet^q \gamma\}) &= \frac{1}{|\tau|} \left(\left(\frac{1}{|\gamma|} - Y_K \right) \omega_{K-1}(\gamma_1) \cdots \omega_{K-1}(\gamma_m) + Y_K \omega_K(\gamma_1) \cdots \omega_K(\gamma_m) \right) \\ &= \frac{1}{|\tau|} \omega_K(\gamma).\end{aligned}$$

where

$$\omega_K(\gamma) = \left(\frac{1}{|\gamma|} - Y_K\right)\omega_{K-1}(\gamma_1) \cdots \omega_{K-1}(\gamma_m) + Y_K\omega_K(\gamma_1) \cdots \omega_K(\gamma_m),$$

which proves the statement of the lemma. $\square$

We can now prove the main Theorem 2 which characterizes EEDs that produce order jumps.

Proof of Theorem 2. To show that $p_{K+1} = p_K + 2$ assuming $p_K + 2 \leq p$, we need to prove for all $\tau \in T$ with $|\tau| \leq p_K + 2$ that

$$\alpha^{[K+1]}(\tau) = \beta(\tau), \quad \text{where } \beta(\tau) = 1/\tau!. \quad (25)$$

This follows for all trees $\tau \in T$ with $|\tau| \leq p_K + 1$ due to Theorem 1. It remains to show that the same is true for all trees τ of size $|\tau| = p_K + 2$. Given $\tau = \{\tau_1 \cdots \tau_n\}$ of size $p_K + 2$, we apply Lemma 2,

$$\alpha^{[K+1]}(\tau) = \frac{1}{|\tau|}\omega_{K+1}(\tau_1) \cdots \omega_{K+1}(\tau_n),$$

and it remains to show that $\omega_{K+1}(\tau_i) = 1/\tau_i!$ for $i = 1, \dots, n$. If $n > 1$, then all trees τ_i have size less or equal to p_K and, thus, following Lemma 2,

$$\omega_{K+1}(\tau_i) = (|\tau_i| + 1)\alpha^{[K+1]}(\{\tau_i\}) = \frac{|\tau_i| + 1}{\{\tau_i\}!} = \frac{1}{\tau_i!},$$

and (25) is proved. Assume $n = 1$, then let τ_1 be denoted by $\gamma = \{\gamma_1 \cdots \gamma_m\}$. We note that $|\gamma| = p_K + 1$. If $p_K < \eta_{K+1}$, then, from the proof of Lemma 1, we have

$$\begin{aligned} \omega_{K+1}(\gamma) &= \left(\frac{1}{|\gamma|} - W_{K,p_K+1}\right)\omega_K(\gamma_1) \cdots \omega_K(\gamma_m) \\ &\quad + W_{K,p_K+1}\omega_{K+1}(\gamma_1) \cdots \omega_{K+1}(\gamma_m) = \frac{1}{|\gamma|}\omega_{K+1}(\gamma_1) \cdots \omega_{K+1}(\gamma_m), \end{aligned}$$

If $\eta_{K+1} \leq p_K < \eta_{K+1} + \zeta_{K+1}$, then, according to the proof of Lemma 2 and depending on the structure of γ , we either apply Lemma 1 as in the previous argument, or we obtain,

$$\begin{aligned} \omega_{K+1}(\gamma) &= \left(\frac{1}{|\gamma|} - Y_{K+1}\right)\omega_K(\gamma_1) \cdots \omega_K(\gamma_m) + Y_{K+1}\omega_{K+1}(\gamma_1) \cdots \omega_{K+1}(\gamma_m) \\ &= \frac{1}{|\gamma|}\omega_{K+1}(\gamma_1) \cdots \omega_{K+1}(\gamma_m). \end{aligned}$$

Since $|\gamma_i| \leq p_K$ for all $i = 1, \dots, m$, we have $\omega_{K+1}(\gamma_i) = 1/\gamma_i!$ and we obtain

$$\alpha^{[K+1]}(\tau) = \frac{1}{|\tau|} \frac{1}{|\gamma|} \frac{1}{\gamma_1! \cdots \gamma_m!} = \frac{1}{\tau!},$$

and (25) is proved. $\square$

4 Stability and convergence order

This section addresses numerically the stability properties of the new SDC methods and the convergence of the SDC iteration towards the underlying RKM as $k \rightarrow \infty$ for finite values of $\Delta t > 0$. We also confirm numerically the order jumps of the SDC iterations as analysed in Section 3.2 by considering the linear Dahlquist test equation

$$u' = \lambda u, \quad u(t_n) = u_n, \quad t \in [t_n, t_{n+1}], \quad (26)$$

with $\lambda \in \mathbb{C}$ and $\text{Re}(\lambda) \leq 0$ and the Euler's nonlinear rigid body dynamics [48]

$$\frac{1}{D_1 D_2 D_3} \nabla C \times \nabla H = \begin{pmatrix} \dot{Y}_1 \\ \dot{Y}_2 \\ \dot{Y}_3 \end{pmatrix} = \begin{pmatrix} Y_2 Y_3 \\ Y_1 Y_3 \\ -Y_1 Y_2 \end{pmatrix}, \quad \mathbf{Y}_0 = \begin{pmatrix} 1/\sqrt{3} \\ 1 \\ 0 \end{pmatrix}, \quad t \in [t_0 = 0, t_{\text{end}} = 10], \quad (27)$$

where $D_1 = N_2 - N_3$, $D_2 = N_3 - N_1$, $D_3 = N_2 - N_1$, $C = 0.5(N_1 D_1 Y_1^2 + N_2 D_2 Y_2^2 + N_3 D_3 Y_3^2)$ is the Casimir, $H = 0.5(D_1 Y_1^2 + D_2 Y_2^2 + D_3 Y_3^2)$ the Hamiltonian, $Y_i = \sqrt{N_i/D_i} \tilde{Q} A_i$ a normalised angular acceleration and N_n are constants satisfying $N_1 < N_3 < N_2$ and $\tilde{Q} = \sqrt{D_1 D_2 D_3 / (N_1 N_2 N_3)}$. We set $N_1 = 1$, $N_2 = 3$ and $N_3 = 2$. The initial conditions in the non-normalised amplitudes A_i are given by $(1, 1, 0)^\top$.

4.1 Numerical confirmation of convergence order

Finally, we confirm numerically that the SDC method with order jump in every sweep reaches the theoretically expected convergence order. Figure 3 shows the absolute error at the end of the simulation in the infinity norm versus the timestep size for the linear Dahlquist equation (26) (left) and the nonlinear Euler rigid body problem (27) (right). As our theory predicts, the method gains two orders with every sweep in both the linear and nonlinear case.

4.2 Stability and convergence of the SDC iteration

To analyse convergence of SDC towards the collocation solution outside the asymptotic limit we establish some preliminaries. Consider an SDC method $(\mathbf{A}_\Delta^0, \dots, \mathbf{A}_\Delta^K, \mathbf{A}, \mathbf{b}, \mathbf{c})$ applied to (26). Let $\mathbf{u}$ denote the stages of the collocation method (2b) and $\mathbf{u}^k$ the SDC stages after k iterations. The error $\mathbf{e}^k := \mathbf{u}^k - \mathbf{u}$ is governed by

$$\mathbf{e}^{k+1} = \mathbf{B}^k(z) \mathbf{e}^k = \mathbf{B}^k \mathbf{e}^k = \mathbf{B}^k \dots \mathbf{B}^1 \mathbf{e}^1 \quad (28)$$

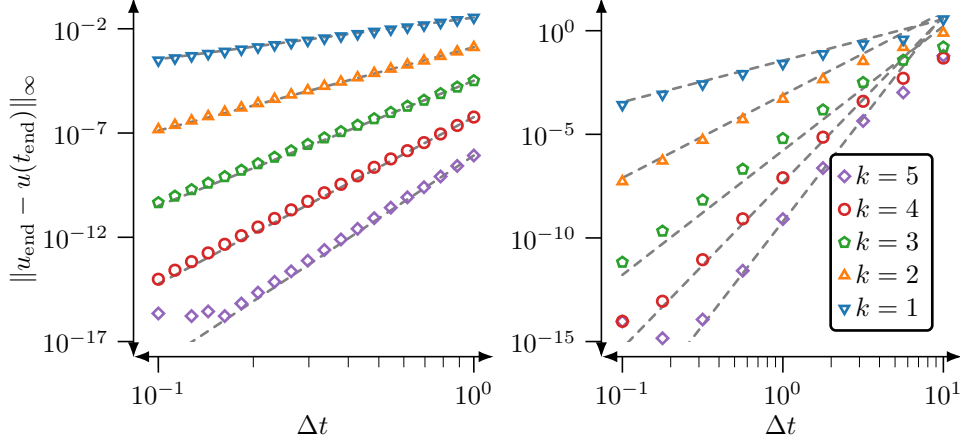


Fig. 3 Convergence of SDC with $M = 6$ Radau IIA nodes and the EED introduced in Theorem 2 that guarantee an order jump in every iteration. The last node was used as the solution in each timestep. The grey dashed lines serve as a guide to the eye with slopes 2, 4, 6, 8 and 10. The method shows the expected 2nd, 4th, 6th, 8th, and 10th order convergence for $k = 1, 2, 3, 4, 5$ iterations. **Left:** Dahlquist test equation with $t_0 = 0$, $t_{\text{end}} = 1$, $\lambda = -1$ and $u_0 = 1$. The data point of $k = 5$ close to $\Delta t = 10^{-1}$ that appears to be missing is below the shown scale. **Right:** Euler rigid body equations. Because we use SciPy `fsolve` to solve the arising nonlinear systems, the error saturates at around $\sim 10^{-14}$.

with

$$\mathbf{B}^k(z) = z(\mathbf{I} - z\mathbf{A}_\Delta^k)^{-1}(\mathbf{A} - \mathbf{A}_\Delta^k) \quad (29)$$

and $z := \lambda\Delta t$. $\mathbf{B}^k(z)$ is called the iteration matrix. For a constant EED, the iteration matrix remains the same for all iterations, i.e. $\mathbf{B}^k(z) = \mathbf{B}(z)$. If, for a given z , its spectral radius is < 1 , the method $(\mathbf{A}_\Delta, \mathbf{A}, \mathbf{b}, \mathbf{c})$ converges towards the collocation solution of $(\mathbf{A}, \mathbf{b}, \mathbf{c})$ as $k \rightarrow \infty$. Note that while the collocation method is A-stable and thus stable for any $z \in \mathbb{C}^-$, the SDC iteration will not necessarily converge on the full negative complex half-plane. Therefore, if the spectral radius of $\mathbf{B}$ is smaller than unity for a given z , we only know that SDC will become stable if the number of iterations K is large enough. But the precise interplay between *convergence domain* and *stability domain* of SDC is complex and not yet very well studied [49].

For iterations that use a different EED in every sweep, $\mathbf{B}^k(z)$ will not be constant and we require a different concept to assess convergence. First, let $\Sigma = \{\mathbf{B}^i(z) : i = 1, 2, \dots\}$ be a sequence of iteration matrices (29) computed from a sequence of EEDs $\{\mathbf{A}_\Delta^1, \mathbf{A}_\Delta^2, \dots\}$. Then let $\Sigma_K := \{\mathbf{B}^1(z), \dots, \mathbf{B}^K(z)\}$ be the iteration matrices for the EEDs used in an SDC method with a finite number of K sweeps. Set $\rho_K(\Sigma, \|\cdot\|) := \max\{\|\mathbf{B}^1(z)\|^{\frac{1}{K}}, \dots, \|\mathbf{B}^K(z)\|^{\frac{1}{K}}\}$ and define the joint spectral radius [50] as

$$\rho(\Sigma) := \lim_{K \rightarrow \infty} \rho_K(\Sigma, \|\cdot\|).$$

Iteration (28) will converge for a selection of EEDs Σ if $\rho(\Sigma) < 1$ [51, Theorem 1].

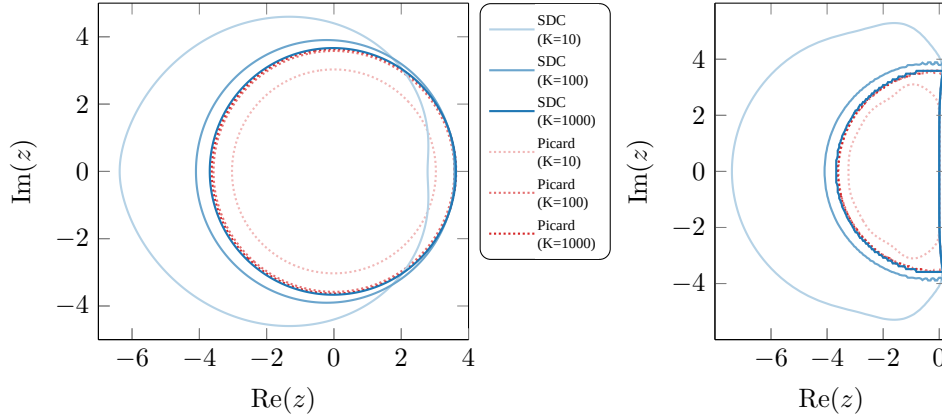


Fig. 4 $M = 3$ Radau IIA nodes were used for the figure. **Left:** Domains where $\rho_K < 1$ for $K = 10^1, 10^2, 10^3$, approximating convergence domain where the joint spectral radius is smaller than unity, for the unique EED guaranteeing order jumps in each iteration (blue solid lines) and Picard iterations (dotted red lines). **Right:** Stability domain for SDC with order jump in each iteration (blue solid lines) and Picard iteration (red dotted lines) after $k = 10^1, 10^2, 10^3$ iterations. In both plots the contours of the SDC method with order jumps approach the contours of Picard iterations as the number of sweeps is increased.

We analyse convergence of the SDC method $(\mathbf{A}_\Delta^1, \dots, \mathbf{A}_\Delta^K, \mathbf{A}, \mathbf{b}, \mathbf{c})$ where $\mathbf{A}_\Delta^k = \text{diag}(\mathbf{c})/2k$ are the unique EEDs that generate an order jump each iteration characterized in Theorem 2. The iteration matrix in iteration k reads

$$\mathbf{B}^k = z \left(\mathbf{I} - \frac{z}{2k} \text{diag}(\mathbf{c}) \right)^{-1} \left(\mathbf{A} - \frac{1}{2k} \text{diag}(\mathbf{c}) \right)$$

so that

$$\lim_{k \rightarrow \infty} \mathbf{B}^k = z\mathbf{A}.$$

Hence, for a fixed z , the iteration matrix $\mathbf{B}^k$ approaches the iteration matrix of Picard iterations in the limit $k \rightarrow \infty$. Since Picard iterations define an explicit RKM, we expect the method build with EEDs that guarantee an order jump each iteration to be stable for $|z| \gg 1$ and large k . Figure 4 shows convergence domains where the joint spectral radius $\bar{\rho}_k$ is smaller than unity and stability domains of the SDC method with order jumps and Picard iterations. As the number of sweeps K increases, the stability domains of SDC with order jumps shrinks down to that of Picard iterations. This is in contrast to implicit-explicit SDC for which stability regions increase with more iterations [3].

Therefore, maximizing only for order by exclusively using EEDs with order jumps will result in an SDC method suited only for non-stiff problems. We can, however, mix different EEDs to balance order and stability. Numerical experiments not documented here suggest that combinations of the MIN-SR-FLEX with order jump EEDs could deliver useful methods. However, a detailed investigation and the development of new SDC variants that efficiently combine EEDs for stability with EEDs for order is left for future work.

5 Summary

We provide a new derivation of spectral deferred corrections (SDC) which allows to formulate a wide range of different variants of SDC methods as Runge-Kutta methods (RKM) represented by Butcher tables. By using the theory of B-series and the methodology of simplifying assumptions for order conditions, we can provide a comprehensive analysis of SDC's convergence order. A general proof is given that any error equation discretization (EED), also called "sweeper" in SDC parlance, increases the overall order of the method by at least one, independent of the underlying nodes. Importantly, the proof covers non-stationary EEDs where the sweeper changes in every iteration which is required for optimal convergence in some parallelizable SDC variants. Our proof also covers EEDs that are algebraically motivated and which are therefore not consistent discretizations of the error equation. We provide a rigorous explanation of when parallel SDC methods produce order jumps, that is, increase the order by two orders in a single iteration, a phenomenon that was observed numerically before but not yet explained theoretically. While SDC's main area of application are stiff problems, our analysis tackles the non-stiff convergence order. We emphasize that order reduction for stiff problems has been observed for SDC [18] with strategies for mitigation in [4, 8]. The proposed examples of EEDs with repeated order jumps do not preserve the A or L-stability properties of the underlying Runge-Kutta methods, but we believe it paves the way toward new SDC methods combining both order jumps and favourable stability properties.

Acknowledgements. JF and DR thankfully acknowledge funding from the German Federal Ministry for Research, Technology and Aeronautics (BMFTR) under grant 16ME0679K. Supported by the European Union - NextGenerationEU. EB and GV thankfully acknowledge the support of the Swiss National Science Foundation, projects No 200020_214819, and No. 200020_192129 and No. 10009199. EB thankfully acknowledges the support of the Knut and Alice Wallenberg Foundation (grant number KAW 2023.0433).

References

- [1] Dutt, A., Greengard, L., Rokhlin, V.: Spectral Deferred Correction methods for ordinary differential equations. *BIT Numerical Mathematics* **40**(2), 241–266 (2000) <https://doi.org/10.1023/A:1022338906936>
- [2] Minion, M.L.: Semi-implicit spectral deferred correction methods for ordinary differential equations. *Communications in Mathematical Sciences* **1**(3), 471–500 (2003) <https://doi.org/10.4310/CMS.2003.v1.n3.a6>
- [3] Ruprecht, D., Speck, R.: Spectral Deferred Corrections with fast-wave slow-wave splitting. *SIAM Journal on Scientific Computing* **38**(4), 2535–2557 (2016) <https://doi.org/10.1137/16M1060078>
- [4] Huang, J., Jia, J., Minion, M.: Accelerating the convergence of spectral deferred correction methods. *Journal of Computational Physics* **214**(2), 633–656 (2006)
- [5] Speck, R.: Parallelizing Spectral Deferred Corrections across the Method. *arXiv* (2017). <https://doi.org/10.48550/ARXIV.1703.08079>
- [6] Crockatt, M.M., Christlieb, A.J.: Low-Storage Integral Deferred Correction Methods for Scientific Computing. *SIAM Journal on Scientific Computing* **40**(5), 2883–2904 (2018) <https://doi.org/10.1137/18M117368X>
- [7] Xia, Y., Xu, Y., Shu, C.-W.: Efficient time discretization for local discontinuous Galerkin methods. *Discrete and Continuous Dynamical Systems – Series B* **8**(3), 677–693 (2007)
- [8] Weiser, M.: Faster SDC convergence on non-equidistant grids by DIRK sweeps. *BIT Numerical Mathematics* **55**(4), 1219–1241 (2014) <https://doi.org/10.1007/s10543-014-0540-y>
- [9] Winkel, M., Speck, R., Ruprecht, D.: A high-order Boris integrator. *Journal of Computational Physics* **295**, 456–474 (2015) <https://doi.org/10.1016/j.jcp.2015.04.022>
- [10] Christlieb, A., Ong, B., Qiu, J.-M.: Comments on high-order integrators embedded within integral deferred correction methods. *Communications in Applied Mathematics and Computational Science* **4**(1), 27–56 (2009) <https://doi.org/10.2140/camcos.2009.4.27>
- [11] Ong, B.W., Spiteri, R.J.: Deferred correction methods for ordinary differential equations. *Journal of Scientific Computing* **83**(60) (2020) <https://doi.org/10.1007/s10915-020-01235-8>
- [12] Ketcheson, D.I., Waheed, U.: A comparison of high-order explicit Runge-Kutta, extrapolation, and deferred correction methods in serial and parallel. *Communications in Applied Mathematics and Computational Science* **9**, 175–200

(2014)

- [13] Naumann, A., Ruprecht, D., Wensch, J.: Toward transient finite element simulation of thermal deformation of machine tools in real-time. *Computational Mechanics* **62**(5), 929–942 (2018) <https://doi.org/10.1007/s00466-018-1540-6>
- [14] Speck, R.: Parallelizing spectral deferred corrections across the method. *Computing and Visualization in Science* (2018) <https://doi.org/10.1007/s00791-018-0298-x>
- [15] Tretiak, K., Buchanan, J., Akers, R., Ruprecht, D.: Performance of the BGSDC integrator for computing fast ion trajectories in nuclear fusion reactors. *Computer Physics Communications* **264**, 107876 (2021) <https://doi.org/10.1016/j.cpc.2021.107876>
- [16] Brown, A., Fregin, J., Bendall, T., Melvin, T., Ruprecht, D., Shipton, J.: Fast-wave slow-wave spectral deferred correction methods applied to the compressible Euler equations (2025). <https://arxiv.org/abs/2505.15985>
- [17] Guesmi, M., Grotteschi, M., Stiller, J.: Assessment of high-order imex methods for incompressible flow. *International Journal for Numerical Methods in Fluids* **95**(6), 954–978 (2023)
- [18] Čaklović, G., Lunet, T., Götschel, S., Ruprecht, D.: Improving efficiency of parallel across the method Spectral Deferred Corrections. *SIAM Journal on Scientific Computing* **47**(1), 430–453 (2025) <https://doi.org/10.1137/24M1649800>
- [19] Hairer, E.: On the order of Iterated Defect Correction. *Numerische Mathematik* **29**(4), 409–424 (1978) <https://doi.org/10.1007/BF01432878>
- [20] Gottlieb, S., Ketcheson, D.I., Shu, C.-W.: High order strong stability preserving time discretizations. *Journal of Scientific Computing* **38**(3), 251–289 (2009)
- [21] Gottlieb, S., Ketcheson, D., Shu, C.-W.: Strong Stability Preserving Runge-Kutta and Multistep Time Discretizations. WORLD SCIENTIFIC, Singapore (2011). <https://doi.org/10.1142/7498>
- [22] Houwen, P.J., Sommeijer, B.P.: Iterated Runge–Kutta methods on parallel computers. *SIAM Journal on Scientific and Statistical Computing* **12**(5), 1000–1028 (1991) <https://doi.org/10.1137/0912054>
- [23] Houwen, P.J.V.D., Sommeijer, B.P., Couzy, W.: Embedded Diagonally Implicit Runge-Kutta Algorithms on Parallel Computers. *Mathematics of Computation* **58**(197), 135 (1992) <https://doi.org/10.2307/2153025>
- [24] Houwen, P.J., Sommeijer, B.P.: Analysis of parallel diagonally implicit iteration of Runge-Kutta methods. *Applied Numerical Mathematics* **11**(1–3), 169–188 (1993)

[https://doi.org/10.1016/0168-9274\(93\)90047-u](https://doi.org/10.1016/0168-9274(93)90047-u)

- [25] Boscarino, S., Qiu, J.-M., Russo, G.: Implicit-explicit Integral Deferred Correction methods for stiff problems. *SIAM Journal on Scientific Computing* **40**(2), 787–816 (2018) <https://doi.org/10.1137/16M1105232>
- [26] Hansen, A.C., Strain, J.: Convergence theory for spectral deferred correction. University of California at Berkeley (2005)
- [27] Hansen, A.C., Strain, J.: On the order of deferred correction. *Applied Numerical Mathematics* **61**, 961–973 (2011) <https://doi.org/10.1016/j.apnum.2011.04.001>
- [28] Causley, M.F., Seal, D.C.: On the convergence of spectral deferred correction methods. *Communications in Applied Mathematics and Computational Science* **14**, 33–64 (2019) <https://doi.org/10.2140/camcos.2019.14.33>
- [29] Christlieb, A., Ong, B.W., Qiu, J.-M.: Integral deferred correction methods constructed with high order Runge-Kutta integrators. *Mathematics of Computation* **79**, 761–783 (2010) <https://doi.org/10.1090/S0025-5718-09-02276-5>
- [30] Hagstrom, T., Zhou, R.: On the spectral deferred correction of splitting methods for initial value problems. *Communications in Applied Mathematics and Computational Science* **1**(1), 169–205 (2006) <https://doi.org/10.2140/camcos.2006.1.169>
- [31] Tang, T., Xie, H., Yin, X.: High-order convergence of Spectral Deferred Correction methods on general quadrature nodes. *Journal of Scientific Computing* **56**(1), 1–13 (2013)
- [32] Jackson, K., Kværnø, A., Nørsett, S.: An analysis of the order of runge-kutta methods that use an iterative scheme to compute their internal stage values. *BIT Numerical Mathematics* **36**(4), 713–765 (1996)
- [33] Bolten, M., Wimmer, L.: On the analysis of spectral deferred corrections for differential-algebraic equations of index one: M. bolten, l. wimmer. *Journal of Scientific Computing* **108**(3), 67 (2026)
- [34] Ketcheson, D.I., Ranocha, H.: Computing with B-series. *ACM Trans. Math. Softw.* **49**(2) (2023) <https://doi.org/10.1145/3573384>
- [35] Hairer, E., Lubich, C., Wanner, G.: Geometric Numerical Integration: Structure-preserving Algorithms for Ordinary Differential Equations. Springer, Heidelberg (2002). <https://doi.org/10.1007/3-540-30666-8>
- [36] Butcher, J.C.: On Runge-Kutta processes of high order. *Journal of the Australian Mathematical Society* **4**(2), 179–194 (1964) <https://doi.org/10.1017/s1446788700023387>

- [37] Hairer, E., Wanner, G.: Solving Ordinary Differential Equations II: Stiff Problems. Springer, Heidelberg (1996). <https://doi.org/10.1007/978-3-642-05221-7>
- [38] Ehle, B.L.: On Padé approximations to the exponential function and A-stable methods for the numerical solution of initial value problems. PhD thesis, University of Waterloo Waterloo, Ontario (1969)
- [39] Prothero, A., Robinson, A.: On the stability and accuracy of one-step methods for solving stiff systems of ordinary differential equations. *Mathematics of Computation* **28**(125), 145–162 (1974) <https://doi.org/10.1090/s0025-5718-1974-0331793-2>
- [40] Ascher, U.M., Ruuth, S.J., Spiteri, R.J.: Implicit-explicit Runge-Kutta methods for time-dependent partial differential equations. *Applied Numerical Mathematics* **25**(2-3), 151–167 (1997) [https://doi.org/10.1016/s0168-9274\(97\)00056-1](https://doi.org/10.1016/s0168-9274(97)00056-1)
- [41] Layton, A.T., Minion, M.L.: Implications of the choice of quadrature nodes for Picard Integral Deferred Corrections methods for ordinary differential equations. *BIT Numerical Mathematics* **45**(2), 341–373 (2005)
- [42] Fregin, J.: Analysis of spectral deferred corrections as Runge-Kutta methods and their application to numerical weather prediction. PhD thesis, Hamburg University of Technology, Hamburg, Germany (March 2026). In preparation
- [43] Cayley, A.: On the theory of the analytical forms called trees. *Philosophical Magazine* **13**, 172–176 (1857) <https://doi.org/10.1080/14786445708642866>
- [44] Hairer, E., Wanner, G.: On the Butcher group and general multi-value methods. *Computing* **13**(1), 1–15 (1974) <https://doi.org/10.1007/BF02268387>
- [45] Hairer, E., Nørsett, S.P., Wanner, G.: Solving Ordinary Differential Equations I: Nonstiff Problems, 2nd edn. Springer, Heidelberg (1993). <https://doi.org/10.1007/978-3-540-78862-1>
- [46] McLachlan, R., Modin, K., Munthe-Kaas, H., Verdier, O.: What are Butcher series, really? the story of rooted trees and numerical methods for evolution equations. *Asia Pacific Mathematics Newsletter* (2015)
- [47] Sanz-Serna, J.M., Murua, A.: Formal series and numerical integrators: some history and some new techniques. In: *Proceedings of the 8th International Congress on Industrial and Applied Mathematics*, pp. 311–331. Higher Ed. Press, Beijing (2015)
- [48] Vilmart, G.: In: Engquist, B. (ed.) *Rigid Body Dynamics in Encyclopedia of Applied and Computational Mathematics*, pp. 1268–1276. Springer, Berlin, Heidelberg (2015). https://doi.org/10.1007/978-3-540-70529-1_142

- [49] Akramov, I., Götschel, S., Minion, M., Ruprecht, D., Speck, R.: Spectral deferred correction methods for second-order problems. *SIAM Journal on Scientific Computing* **46**(3), 1690–1713 (2024) <https://doi.org/10.1137/23M1592596>
- [50] Rota, G.-C., Strang, G.: A note on the joint spectral radius. *Indag. Math* **22**(4), 379–381 (1960)
- [51] Berger, M.A., Wang, Y.: Bounded semigroups of matrices. *Linear Algebra and its Applications* **166**, 21–27 (1992)