Spectral Deferred Corrections in the framework of Runge-Kutta methods

Eugen Bronasco[1*†], Joscha Fregin[2*†], Daniel Ruprecht[2] and Gilles Vilmart[3]

[1*]Department of Mathematical Sciences, Chalmers University of Technology and University of Gothenburg, Chalmersplatsen 4, Gothenburg, SE-412 96, Gothenburg, Sweden.
[2]Chair 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

We interpret a wide range of flavors of Spectral Deferred Corrections (SDC) as Runge-Kutta methods (RKM). Using Butcher series, we show that the considered class of SDC methods achieve at least order $\boldsymbol{p}$ after $\boldsymbol{p}$ iterations compared to the underlying RKM, independently of the error discretisation chosen and the choice of nodes. For all collocation RKM, we analyse the phenomenon of order jumps in SDC iterations, where the order is increased by two at each iteration. We prove that it can be obtained by using appropriate inconsistent, implicit, parallelisable error discretisations. We also investigate the stability properties of the new SDC methods which can in general reduce to that of explicit RKM, but it can be improved by suitable combinations of error discretisations. We confirm the convergence analysis with numerical experiments and we apply relaxation RKM to derive SDC variants that conserve quadratic invariants.

Keywords: Spectral Deferred Corrections, Runge-Kutta, B-Series

1 Introduction

Spectral deferred corrections (SDC) have received significant amounts of attention since their introduction by Dutt et al. in [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].

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 discretisation of order at least one) 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 change in each iteration [8, 12].
4. The last step of SDC is to calculate the solution at the endpoint from the stage values. Extrapolation is proposed in [1] where the interpolating polynomial defined by the stage values is evaluated at the end point of the time step. An alternative is to use collocation nodes that 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 usual in RKM.

Some links between SDC and RKM have already been established. A relation between IDC and RKM is shown in [10]. Butcher series were used in [13] to study the convergence of deferred correction schemes. For select SDC methods, a Shu-Osher form has been described in [14, 15]. The structure of Butcher tableaus that emerges from this approach has first been described in [16–18] which predates the original introduction of SDC in [1] by almost a decade. Butcher tableaus for IMEX-SDC have been derived in [19].

The present paper incorporates all those design choices 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 before [12] 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. As one additional example how our RKM interpretation can be useful, we build an S-conservative SDC method based on relaxation approaches for RKM which conserves quadratic invariants and favourable long term integration properties for periodic, time reversible systems [20–22].

Related work on the theory of SDC.

Most convergence results for SDC consider either equidistant nodes and consistent EED [10, 13, 23–26] or arbitrary nodes and inconsistent EED [4, 7, 16, 27]. Few works address the combination of arbitrary nodes and consistent EEDs, one example being [3]. Consistent, high order EED are also discussed in [10, 13, 23, 24, 26, 28].

SDC can be initialised using a low order method [1] or by copying the initial state [3, 17]. [7] prove that 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. [23, 24] analyse SDC for arbitrary order one-step and multi-step methods for equidistant nodes and 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 [12].

[27] 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]. [25] 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. [26] discretise the error equation using arbitrary order RKM to construct consistent EED when using equidistant nodes.

[10] expand the IDC methods introduced in [26] 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 node a high order ($p \geq 2$) EED does not necessarily increase the order of SDC by more than one per iteration. [13] interprets deferred correction methods with n equidistant nodes as n steps of an integrator which can be expaned as a Butcher series and uses the theory of Butcher series to prove the order increase per iteration.

[3] prove that SDC gains one order per iteration when using implicit-explicit methods as EED, but the prove works for standard SDC as well. Their proof breaks down for inconsistent EED because their bound on the supremums norm of the matrix containing the collocation weights is no longer guaranteed to be true. Finally, [16] 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 discussed already in [10, 23, 24, 26]. 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 in [10] (Example 3.13) and observed numerically for parallel SDC methods in [12]. To our knowledge, there is no theory available that predicts order jumps for non-equidistant nodes for general EEDs.

We use the Julia package described in [29] and translate SDC into Runge-Kutta methods to calculate the order of SDC methods based on their Butcher tableaux and analyse the behaviour of order jumps further. Tables A1–A6 show the orders of SDC methods with Gauss, Radau IIA, and Lobatto nodes using the trapezoidal rule EED considered in [10], an EED optimised for non stiff problems considered in [12], and a novel EED which we introduce in Theorem 5, Section 3.2. Tables A1 and A2 show that considering an EED derived from an order 2 method is not enough to obtain an order jump when the nodes are non-equidistant. Moreover, Table A1 illustrates that an iteration of SDC fails to increase the order by 1 after an order jump occurs which agrees with the observations from [10]. This is discussed in more detail in Remark 5, Section 3.1. Tables A3, A4, and A5 show that the SDC methods with Gauss, Radau IIA, and Lobatto nodes with the EED optimised for non stiff problems perform an order jump when the iteration count is one below the number of nodes used. This phenomenon is explained in Corollary 2, Section 3.2.2. Table A6 demonstrates that the novel, diagonal EED introduced in Theorem 5 performs an order jump on each iteration using a non equidistant node set.

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 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 method is used as the underlying method. 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. Finally, we use the theory of relaxation RKM to force conservation of the Hamiltonian of Eulers rigid body equations when solved using SDC to further motivate SDC in the RKM framework. The resulting methods can achieve favourable long term integration properties compared to standard SDC.

2 Spectral deferred corrections

Consider the system of ordinary differential equations (ODE),

$$\frac{\mathrm{d}u(t)}{\mathrm{d}t} = f(u(t)), \tag{1}$$

with vector field[1] $f : \mathbb{R}^d \to \mathbb{R}^d$ assumed sufficiently differentiable and with initial condition $u(0) = u_0$. In this paper we are interested in the implementaion of the classical implicit Runge-Kutta methods (RKM) of the form,

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

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

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

$$\begin{array}{c|c} \boldsymbol{c} & \boldsymbol{A} \\ \hline & \boldsymbol{b}^\intercal \end{array},$$

where $\boldsymbol{c} = (c_i)_{i=1}^s$, $\boldsymbol{b} = (b_i)_{i=1}^s$, and $\boldsymbol{A} = (a_{ij})_{i,j=1}^s$ [31]. In this paper, we shall also focus on the case of *collocation* Runge-Kutta methods [30, Section II.1.2] with arbitrary collocation nodes c_i, $i = 1, \ldots, s$, and where we recall that Runge-Kutta coefficients are given by,

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

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 $c_i, i = 1, \ldots, s$. 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 $2s-2$ with the favourable A-stability property which is desirable for stiff problems.

Consider the Dahlquist's test equation $\frac{\mathrm{d}u(t)}{\mathrm{d}t} = \lambda u(t)$ with $\lambda \in \mathbb{C}$ and $Re(\lambda) \leq 0$. 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 [32] (Definition 2.1) we introduce the set $S = \{z \in \mathbb{C} : |R(z)| \leq 1\}$ as the stability domain of a given method. If $z \in S$, then the numerical solution $\{u_n\}$ is bounded and is called stable. A method is called A(α) stable if $\{z : |\arg(-z)| \leq \alpha\} \in S$ with $\alpha \in [0, \pi/2]$ ([32] Definition 3.9, [33]). A method is called A-stable if it is A(α)-stable with $\alpha = \pi/2$ [34] [32] (Definition 3.3). A method is called L-stable if it is

[1] For simplicity of the presentation, we assume the ODE to be autonomous. Note that this is without loss of generality by considering the augmented system with time equation $\frac{ds(t)}{dt} = 1$, see [30, Section III.1.1].

A-stable and additionally satisfies [35]

$$\lim_{z\to\infty} R(z) = 0. \tag{3}$$

A method is L(α)-stable if it is A(α)-stable and its stability function fulfills the limit above. Finally, for a method that satisfies $a_{sj} = b_j$, $j = 1, \dots, s$, Equation 3 holds true [32] (Proposition 3.8). These methods are called stiffly accurate [36]. For instance, Radau IIA and Lobatto IIIA are L-stable and stiffly accurate RKM.

Proposition 1. *Let* $(\tilde{\boldsymbol{A}}, \boldsymbol{b}, \boldsymbol{c}) = (\boldsymbol{A}_\Delta^0, \dots, \boldsymbol{A}_\Delta^K, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c})$ *define an implicit SDC method. If* $\tilde{\boldsymbol{A}}$ *is not singular and satisfies* $a_{si} = b_i$*, the method is stiffly accurate, then* $\lim_{z\to\infty} R(z) = 0$.

Proof. See Proposition 3.8 in [32] and the proof therein. □

This result is mentioned in [16] and is independent of the EED's chosen. We restate it here since the equivalence of parallel RKM and SDC wasn't recognised until recently [12].

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 formulations of SDC method. Here we consider the following derivation based on an integral formulation of (1). Consider the partitioned problem

$$\frac{\mathrm{d}\tilde{u}(t)}{\mathrm{d}t} = \tilde{f}(t, \tilde{u}, \tilde{\delta}) \tag{4}$$

$$\frac{\mathrm{d}\tilde{\delta}(t)}{\mathrm{d}t} = \tilde{f}_\delta(t, \tilde{u}, \tilde{\delta}) \tag{5}$$

and let $u(t) = \tilde{u}(t) + \tilde{\delta}(t)$, where $\tilde{u}(t)$ is an approximate solution and $\tilde{\delta}(t)$ is the error where we assume that $\tilde{\delta}(t_n) = 0$. Furthermore, let $\tilde{f} := f$ and $\tilde{f}_\delta(t, \tilde{u}, \tilde{\delta}) := f(t, \tilde{u}+\tilde{\delta}) - f(t, \tilde{u})$. Based on this partitioning, (1) can then be written in integral form

$$u(t) = u(t_n) + \int_{t_n}^{t} f(u(s))\,\mathrm{d}s = u(t_n) + \int_{t_n}^{t} f(\tilde{u}(s))\,\mathrm{d}s + \int_{t_n}^{t} (f(u(s)) - f(\tilde{u}(s)))\,\mathrm{d}s\,, \tag{6}$$

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 $u(t)$,

$$u^{k+1}(t) = u_n + \int_{t_n}^{t} f(u^k(s))\,\mathrm{d}s + \int_{t_n}^{t} (f(u^{k+1}(s)) - f(u^k(s)))\,\mathrm{d}s\,, \quad k = 0, 1, 2, \dots\,, \tag{7}$$

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,

$$u_n^{[k+1,i]} = u_n + \underbrace{\Delta t \sum_{j=1}^{s} a_{ij} f(u_n^{[k,j]})}_{\text{collocation term}} + \underbrace{\Delta t \sum_{j=1}^{s} \tilde{a}_{ij}^k (f(u_n^{[k+1,j]}) - f(u_n^{[k,j]}))}_{\text{correction term}}, \tag{8}$$

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 $\boldsymbol{A}_\Delta^k = (\tilde{a}_{ij}^k)_{i,j=1}^s$ correspond to the so called *error equation discretization* (EED) where the corresponding term in (8) can be interpreted as a correction term in the fixed point Picard iterations. The output of the SDC method is then given by

$$u_{n+1} = u_n + \Delta t \sum_{i=1}^{s} b_i f(u_n^{[K,i]}). \tag{9}$$

The iterative method (8) and (9) defines the class of SDC methods $(\boldsymbol{A}_\Delta^0, \dots, \boldsymbol{A}_\Delta^K, \boldsymbol{A}, \boldsymbol{b})$ with $\boldsymbol{A}_\Delta^k = (\tilde{a}_{ij}^k)_{i,j=1}^s$, that we shall analyse in this paper.

We emphasize that the SDC method (8) and (9) is itself a Runge-Kutta method with the following augmented Butcher tableau with $(K+1)s$ internal stages,

$$\begin{array}{c|cccc} \tilde{\boldsymbol{c}} & \boldsymbol{A}_\Delta^0 & & & \\ \boldsymbol{c} & \boldsymbol{A} - \boldsymbol{A}_\Delta^1 & \boldsymbol{A}_\Delta^1 & & \\ \vdots & & \ddots & \ddots & \\ \boldsymbol{c} & & & \boldsymbol{A} - \boldsymbol{A}_\Delta^K & \boldsymbol{A}_\Delta^K \\ \hline & & & & \boldsymbol{b}^\intercal \end{array}, \tag{10}$$

where $\boldsymbol{A}$ is the coefficients of the original Runge-Kutta method (2b) and $\tilde{\boldsymbol{c}} = (\tilde{c}_i)_{i=1}^s$ with $\tilde{c}_i = \sum_{j=1}^s \tilde{a}_{ij}^0$. Natural choices for choosing EED $\boldsymbol{A}_\Delta^k$ are based on implicit or explicit Euler methods or diagonal form, which is equivalent to setting, respectively,

$$\boldsymbol{A}_\Delta^k = \begin{pmatrix} \Delta\tau_1 & & \\ \vdots & \ddots & \\ \Delta\tau_1 & \dots & \Delta\tau_s \end{pmatrix}, \quad \boldsymbol{A}_\Delta^k = \begin{pmatrix} 0 & & & \\ \Delta\tau_2 & 0 & & \\ \vdots & \ddots & \ddots & \\ \Delta\tau_2 & \dots & \Delta\tau_s & 0 \end{pmatrix}, \quad \boldsymbol{A}_\Delta^k = \alpha_k \operatorname{diag}(\boldsymbol{c}), \tag{11}$$

with $\Delta\tau_i = c_i - c_{i-1}$ and $\Delta\tau_1 = c_1$, for some constants $\alpha_k \in \mathbb{R}$. Note that the above diagonal form is suitable for a parallel implementation and reveals to be a key choice for faster convergence of the SDC iterations (order jumps, see Theorem 5 in Section 3.2).

Remark 1. *For the initialization of the SDC method* (8), (9) *we consider for simplicity the choice that* $u_n^{[0,i]} = u_n$, $i = 1, \dots, s$, *of a constant initial guess. We*

emphasize, however, that other choices are possible as considered in [37] in the context of implicit-explicit RKM,

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

for a choice of coefficients $\tilde{q}_{ij}$ yielding a higher order of approximation of u_{n+1} and satisfying $\boldsymbol{c} = \tilde{\boldsymbol{c}}$ to approximate the internal stages. Note that this is included in the considered class of SDC methods by considering an appropriate choice for the initial EED, in particular $\boldsymbol{A}_\Delta^0 = (\tilde{q}_{ij})_{i,j=1}^s$. Indeed, using (8), we have,

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

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

Remark 2. *We emphasize that the assumption $\sum_{j=1}^s \tilde{a}_{ij}^k = c_i$ used in the Remark 1 for $k = 0$ is not needed for defining EED $\boldsymbol{A}_\Delta^k$ with $k \geq 1$. For instance, the second and third choices in (11) do not satisfy this assumption that the EED is consistent with the nodes $\boldsymbol{c}$ of the original Runge-Kutta method.*

Remark 3. *There are different possible choices for the output of the SDC method compared to the choice (9) considered for simplicity. For instance, following [1, Eq. 4.2] one can consider the extrapolation formula $u_{n+1} = \sum_{i=1}^s \ell_i(1) u_n^{[K,i]}$, with ℓ_i being the Lagrange polynomials to the nodes c_i. For stiffly accurate Runge-Kutta methods where $a_{sj} = b_j$ like Radau IIA or Lobatto IIIA methods, the choice $u_{n+1} = u_n^{[K,s]}$ can produce $L(\alpha)$-stable SDC methods [38, Theorem 3.1]. The choice (9) can be advantageous compared to $u_{n+1} = u_n^{[K,s]}$, because it permits to gain one order of accuracy without extra cost since $f(u_n^{[K,i]})$ is available from the last iteration of the SDC method (8). Note, that all outlined choices to obtain u_{n+1} require different coefficients $\boldsymbol{b}$ [39]. The coefficients $\boldsymbol{A}, \boldsymbol{A}_\Delta^k, \boldsymbol{c}$ 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 they are a natural choice for the analysis of SDC methods which can be themselves interpreted as Runge-Kutta methods as emphasized in Section 2. Expanding in Taylor series as $h \to 0$ the exact solution and numerical solution after one step, we obtain assuming $u(t_n) = u_n$,

$$\begin{aligned} u_{n+1} &= u_n + \Delta t \sum_{i=1}^{s} b_i f(u_n) + \Delta t^2 \sum_{i,j=1}^{s} b_i a_{ij} f'(u_n) f(u_n) + \mathcal{O}(\Delta t^3), \\ u(t_{n+1}) &= u_n + \Delta t f(u_n) + \Delta t^2 \frac{1}{2} f'(u_n) f(u_n) + \mathcal{O}(\Delta t^3), \end{aligned} \tag{12}$$

and 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 (12) that Runge-Kutta method with coefficients $(\boldsymbol{A}, \boldsymbol{b})$ 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

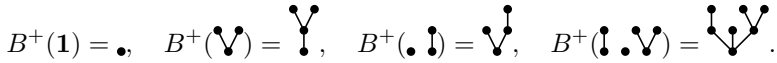
$$\sum_{i=1}^{s} b_i = 1, \quad \sum_{i,j=1}^{s} b_i a_{ij} = \frac{1}{2}.$$

Rooted trees [40] 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 as introduced in [41] based on the seminal work of J. Butcher [31]. We follow here the presentation in [30, 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 the roots at the bottom are

Monomials of trees are called *forests* and denoted by $S(T)$. The empty monomial is written as $\mathbf{1}$. We use the map $B^+ : S(T) \to T$ to construct a tree by attaching all roots of a forest to a new vertex which then becomes a new root. Examples would be



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 $\tau = B^+(\tau_1^{m_1} \cdots \tau_k^{m_k})$ with distinct trees $\tau_1, \ldots, \tau_k \in T$ with multiplicities $m_1, \ldots, m_k \in \mathbb{N}$ 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, $\gamma(\tau) = |\tau| \prod_{i=1}^{k} \gamma(\tau_i)^{m_i}$ with $\gamma(\bullet) = 1$,*
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 *is defined as*

$$F_{\Delta t f}(\tau) := \Delta t f^{(N)}\left(F_{\Delta t f}(\tau_1), \ldots, F_{\Delta t f}(\tau_N)\right),$$

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

An alternative representation of an elementary differential $F_{\Delta t f}(\tau) : \mathbb{R}^d \to \mathbb{R}^d$ with $\tau = B^+(\tau_1 \cdots \tau_N)$ where $\tau, \tau_1, \ldots, \tau_N \in T$ is given by

$$F_{\Delta t f}(\tau) = \Delta t \sum_{i_1, \ldots, i_N = 1}^{d} F_{\Delta t f}(\tau_1)^{i_1} \cdots F_{\Delta t f}(\tau_N)^{i_N} \frac{\partial f}{\partial x_{i_1} \cdots \partial x_{i_N}}. \tag{13}$$

We are now able to represent the Taylor expansion in (12) using formal sums indexed by rooted trees [31, 42–44],

$$u_{n+1} = u_n + \sum_{\tau \in T} \frac{\alpha(\tau)}{\sigma(\tau)} F_{\Delta t f}(\tau)(u_n),$$
$$u(t_{n+1}) = u_n + \sum_{\tau \in T} \frac{1}{\gamma(\tau)\sigma(\tau)} F_{\Delta t f}(\tau)(u_n), \tag{14}$$

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

$$\alpha(\bullet) = \sum_{i=1}^{s} b_i, \quad \alpha(\;\vdots\;) = \sum_{i,j=1}^{s} b_i a_{ij}, \quad \alpha(\;\bigvee\;) = \sum_{i,j,k,l=1}^{s} b_i a_{ij} a_{ik} a_{kl}.$$

The formal sums in (14) 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

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

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

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

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

$$\beta(B^+(\tau_1 \cdots \tau_n)) = \sum_{j=1}^{s} b_j \beta^{[j]}(\tau_1 \cdots \tau_n),$$
$$\beta^{[i]}(B^+(\tau_1 \cdots \tau_n)) = \sum_{j=1}^{s} a_{ij} \beta^{[j]}(\tau_1 \cdots \tau_n), \tag{15}$$

where $\beta^{[j]}$ is extended to $S(T)$ using the multiplicative property,

$$\beta^{[j]}(\tau_1 \cdots \tau_n) = \beta^{[j]}(\tau_1) \cdots \beta^{[j]}(\tau_n).$$

In the context of SDC methods defined by (8), (9), let $\alpha^{[K]}$ and $\alpha^{[K,i]}$ denote the final and internal stages of SDC $(\boldsymbol{A}_\Delta^0, \ldots, \boldsymbol{A}_\Delta^K, \boldsymbol{A}, \boldsymbol{b})$ obtained after K iterations, precisely,

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

Combining (15) with the form of the Butcher tableau of SDC in (10), we have the following property of $\alpha^{[K]}$ and $\alpha^{[K,i]}$,

$$\begin{aligned}
\alpha^{[K]}(B^+(\tau_1 \cdots \tau_n)) &= \sum_{i=1}^{s} b_i \alpha^{[K,i]}(\tau_1 \cdots \tau_n), \\
\alpha^{[K,i]}(B^+(\tau_1 \cdots \tau_n)) &= \sum_{j=1}^{s} \left(a_{ij} - a_{ij}^\Delta\right) \alpha^{[K-1,j]}(\tau_1 \cdots \tau_n) + \sum_{j=1}^{s} a_{ij}^\Delta \alpha^{[K,j]}(\tau_1 \cdots \tau_n),
\end{aligned} \tag{16}$$

that we use extensively in the proofs that follow.

3 Convergence analysis of SDC methods

Consider an SDC method $(\boldsymbol{A}_\Delta^0, \ldots, \boldsymbol{A}_\Delta^K, \boldsymbol{A}, \boldsymbol{b})$. Since SDC approximates the internal stages of the Runge-Kutta method $(\boldsymbol{A}, \boldsymbol{b})$, it is not meant to achieve a higher order of accuracy for the exact solution than the RKM. The order of an SDC method $(\boldsymbol{A}_\Delta^0, \ldots, \boldsymbol{A}_\Delta^K, \boldsymbol{A}, \boldsymbol{b})$ is therefore determined by the local error of final stage (9) compared to the Runge-Kutta method $(\boldsymbol{A}, \boldsymbol{b})$ which motivates the following definition.

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

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

The SDC order should not be confused with the order of the Runge-Kutta method (2a), (2b) or the Runge-Kutta method with the Butcher tableau (10) which in contrast is based on the local error compared to the exact solution. 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 $\tau = B^+(\tau_1 \cdots \tau_n)$ with trees $\tau_1, \ldots, \tau_n \in T$ 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 Figure 1.

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

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

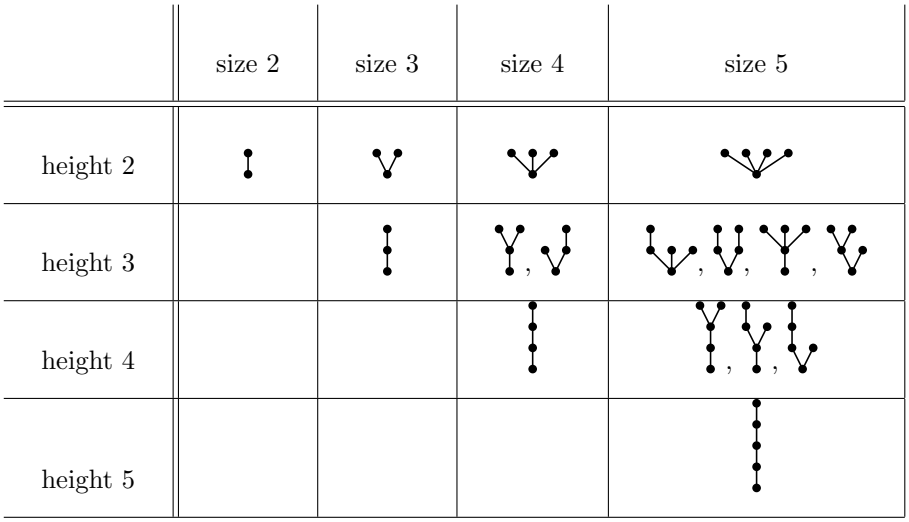


Fig. 1 Comparison of the size and height of trees.

The internal stages of an SDC method are of order $\tilde{p}_K$ (which means it has height order $\tilde{h}_K$) if

$$\alpha^{[K,i]}(\tau) = \beta^{[i]}(\tau), \quad \text{for } i = 1, \dots, s \quad \text{and } |\tau| \leq \tilde{p}_K \,(\text{or } ht(\tau) \leq \tilde{h}_k). \tag{17}$$

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 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 $(\boldsymbol{A}, \boldsymbol{b})$ 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 1.

Theorem 2. *Let the internal stages of an SDC method* $(\boldsymbol{A}_\Delta^0, \dots, \boldsymbol{A}_\Delta^K, \boldsymbol{A}, \boldsymbol{b})$ *be of height order* $h_K - 1$*. It then holds that*

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

Proof. Recall the property (16) and note that $\alpha^{[K+1,i]}(\bullet) = \beta(\bullet)$. Let $\tau = B^+(\tau_1 \cdots \tau_n)$ be a tree of height up to h_K, $ht(\tau) \leq h_K$, then,

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

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

where $\alpha^{[K+1,i]}(\tau_1\cdots\tau_n) = \beta^{[i]}(\tau_1\cdots\tau_n)$ by induction on height. □

Note that Theorem 2 remains valid if we replace height order h_K by order p_K.

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

Remark 4. *An immediate consequence of Corollary 1 is that we obtain the SDC order $p_K \geq h_K \geq K$ for the SDC method $(\boldsymbol{A}_\Delta^0, \boldsymbol{A}_\Delta^1, \dots, \boldsymbol{A}_\Delta^K, \boldsymbol{A}, \boldsymbol{b})$ which uses any of the EEDs introduced in [12] and analysed using a linear algebra perspective,*

$$\boldsymbol{A}^k_{\Delta_{\texttt{MIN-SR-NS}}} = \operatorname{diag}(\frac{\boldsymbol{c}}{s}), \quad \boldsymbol{A}^k_{\Delta_{\texttt{MIN-SR-FLEX}}} = \operatorname{diag}(\frac{\boldsymbol{c}}{k}), \quad \boldsymbol{A}^k_{\Delta_{\texttt{MIN-SR-S}}} = \underset{\boldsymbol{A}^k_\Delta = \operatorname{diag}(\hat{\boldsymbol{c}})}{argmin}\ \lambda_{\max}(\boldsymbol{K}_S^k), \tag{18}$$

where in the third case we minimise the spetral radius $\lambda_{\max}$ of the matrix $\boldsymbol{K}_S^k := \boldsymbol{I} - (\boldsymbol{A}_\Delta^k)^{-1}\boldsymbol{A}$.

Remark 5. *Table A1 shows the convergence order of SDC method with Lobatto nodes and $\boldsymbol{Q}_{\Delta_T}$ EED derived from the trapezoidal method. The table demonstrates that the iterations performed after an order jump do not increase the order of the SDC method. This can be explained by the fact the the order jumps do not increase the height order of all internal stages and, therefore, this phenomenon does not contradict Theorem 2.*

Let us illustrate Theorem 2 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 $(\boldsymbol{A}_\Delta^0, \boldsymbol{A}_\Delta^1, \boldsymbol{A}_\Delta^2, \boldsymbol{A}, \boldsymbol{b})$ with $K = 2$ with lower triangular EEDs where $(\boldsymbol{A}, \boldsymbol{b})$ is a 2 stage Runge-Kutta method. The Butcher tableau (10) of the SDC method has the form*

c_1	$\tilde{a}^0_{11}$					
c_2	$\tilde{a}^0_{21}$	$a^{0\Delta}_{22}$				
c_1	$a_{11}-\tilde{a}^1_{11}$	a_{12}	$\tilde{a}^1_{11}$			
c_2	$a_{21}-\tilde{a}^1_{21}$	$a_{22}-\tilde{a}^1_{22}$	$\tilde{a}^1_{21}$	$\tilde{a}^1_{22}$		
c_1			$a_{11}-\tilde{a}^2_{11}$	a_{12}	$\tilde{a}^2_{11}$	
c_2			$a_{21}-\tilde{a}^2_{21}$	$a_{22}-\tilde{a}^2_{22}$	$\tilde{a}^2_{21}$	$\tilde{a}^2_{22}$
					b_1	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 $(\boldsymbol{A}, \boldsymbol{b})$*, we obtain,*

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

We see that the equalities are true for any $\boldsymbol{A}^0_\Delta, \boldsymbol{A}^1_\Delta, \boldsymbol{A}^2_\Delta$*.*

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 phenonmenon is called an *order jump*.

Recall that the Runge-Kutta coefficients of collocation methods of order p using Gauss, Radau or Lobatto nodes were chosen to satisfy the following simplifying assumptions [32, 42], see Table 2, called $B(p), C(\eta), D(\zeta)$:

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

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	$B(2s-2)$	$C(s-1)$	$D(s-1)$	$2s-2$

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

with $p \leq \eta + \zeta + 1$ and $p \leq 2\eta + 2$. In terms of trees, simplifying assumption $B(p)$ is equivalent to the functional β of the collocation method satisfying the order conditions on trees of height 2 up to size p, that is,

$$\beta(\;\;) = \frac{1}{\gamma(\;\;)}.$$

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,

$$\beta(\;\;) = \frac{1}{q}\beta(\;\;),$$

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,

$$\beta(\;\;) = \frac{1}{q}\beta(\;\; - \;\;),$$

where $\square$ is a placeholder for one or multiple branches. See [32, 42] for details.

Given an EED $\boldsymbol{A}_\Delta = (\tilde{a}_{ij})_{i,j=1}^{s}$ and two sets of constants $W = (W_q)_{q\in\mathbb{N}}$ and $Y = (Y_q)_{q\in\mathbb{N}}$, let us introduce assumptions on $\boldsymbol{A}_\Delta$ defined as

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

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

We note that $C(\eta)$ is a particular case of $C_W(\eta)$, however, $D(\zeta)$ is not a particular case of $D_Y(\zeta)$. The main result of this section is stated in Theorem 5 which is proved in Section 3.2.2.

Theorem 5. *Consider SDC with $K+1$ iterations which approximates a collocation method. Let K^{th} iteration of SDC be of order p_K. We have $p_{K+1} = \min\{p_K + 2, p\}$ if one of the following assumptions is satisfied:*

1. *$p_K < \eta_{K+1}$ and $\boldsymbol{A}_\Delta^k$ satisfies $C_{W_k}(\eta_k)$ with*

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

   where $W_{K+1,p_K+1} = \frac{1}{p_K+1}$.
2. *$\eta_{K+1} \leq p_K < \eta_{K+1} + \zeta_{K+1}$ and $\boldsymbol{A}_\Delta^k$ satisfies $C_{W_k}(\eta_k)$ and $D_{Y_k}(\zeta_k)$ with*

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

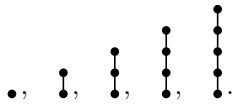
   where $W_{K+1,p_K+1} = \frac{1}{p_K+1}$ and $Y_{K+1,q} = \frac{1}{p_K+1}$ for all $q = 1, \ldots, \zeta_{K+1}$.

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

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 $\boldsymbol{A}_\Delta$ which results in an order jump when the problem is linear, that is, $f : \mathbb{R}^d \to \mathbb{R}^d$ of (1) is a degree one polynomial. We note that 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 Fig. 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 $\boldsymbol{A}_\Delta$ for the order to be increased by 2 per iteration. Let the bamboo tree of hight 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 (16) to prove Proposition 3 which derives a condition on the EED $\boldsymbol{A}_\Delta$ 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 order $\tilde{h}_K$. An EED $\boldsymbol{A}_\Delta = (\tilde{a}_{ij})_{i,j=1}^s$ which results in SDC with $K+1$ iterations and internal stages of order $\tilde{h}_K + 2$ satisfies the following condition*

$$\sum_{j=1}^{s} \tilde{a}_{ij} \epsilon^{[K,j]}(\cdot^{\tilde{h}_K+1}) = \sum_{j=1}^{s} a_{ij} \epsilon^{[K,j]}(\cdot^{\tilde{h}_K+1}). \tag{19}$$

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

$$\alpha^{[K+1,i]}(\cdot^{\tilde{h}_K+2}) = \beta^{[i]}(\cdot^{\tilde{h}_K+2}).$$

We use the property (16) to derive the condition on $\tilde{a}_{ij}$,

$$\sum_{j=1}^{s} \big(a_{ij} - \tilde{a}_{ij}\big)\alpha^{[K,j]}(\cdot^{\tilde{h}_K+1}) + \sum_{j=1}^{s} \tilde{a}_{ij} \alpha^{[K+1,j]}(\cdot^{\tilde{h}_K+1}) = \beta^{[i]}(\cdot^{\tilde{h}_K+2}),$$

We recall Theorem 2 and group the $\tilde{a}_{ij}$ terms which gives,

$$\sum_{j=1}^{s} a_{ij} \alpha^{[K,j]}(\cdot^{\tilde{h}_K+1}) + \sum_{j=1}^{s} \tilde{a}_{ij} \epsilon^{[K,j]}(\cdot^{\tilde{h}_K+1}) = \beta^{[i]}(\cdot^{\tilde{h}_K+2}),$$

We use the property (15) to obtain

$$\sum_{j=1}^{s} \tilde{a}_{ij} \epsilon^{[K,j]}(\cdot^{\tilde{h}_K+1}) = \sum_{j=1}^{s} a_{ij} \epsilon^{[K,j]}(\cdot^{\tilde{h}_K+1}),$$

and the statement is proved. □

Remark 6. *If the EED $\boldsymbol{A}_\Delta$ is diagonal, then, the only EED satisfying (19) is*

$$\tilde{a}_{ii} = \sum_{j=1}^{s} a_{ij} \frac{\epsilon^{[K,j]}}{\epsilon^{[K,i]}}(\cdot^{\tilde{h}_K+1}).$$

Since there exists a unique diagonal EED which results in an order 2 jump, we cannot guarantee that the same EED results in an order 3 jump, see Example 2.

Example 2. *Suppose an SDC method with K iterations, internal stages of order $\tilde{h}_K$, and $\tilde{a}_{ii}^k = l(k,i)c_i$ for some constants $l(k,i) \in \mathbb{R}$ and $\tilde{a}_{ij}^k = 0$ for $i \neq j$. By Proposition*

3, we require $\alpha^{[K+1,i]}($$) = \beta^{[i]}($$)$ to increase the order of the method by 2. From the Butcher tableau we obtain

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

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

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

From above we have

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

Since $\alpha^{[0,j]}(\text{}) = 0$ and $c_i = \sum_{j=1}^{s} a_{ij} = \beta^{[i]}(\text{})$ we obtain

$$\alpha^{[0+1,i]}(\text{}) = l(1,i)c_i\alpha^{[1,i]}(\text{}) = l(1,i)\sum_{j=1}^{s} a_{ij}\sum_{j=1}^{s}(a_{ij} - \tilde{a}_{ij}^{1} + \tilde{a}_{ij}^{1}) = l(1,i)\beta^{[i]}(\text{}),$$

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 internal stages are of order at least 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,

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

if we assume the internal stages of the collocation method to be of order at least 3.

Next, consider an SDC method with order 1 after the first iteration and the same assumptions as above. We require

$$\alpha^{[1+1,i]}(\text{}) = \beta^{[i]}(\text{}),$$

which by looking at the Butcher tableau can be written as

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

$$
\begin{aligned}
&= \sum_{j=1}^{s} a_{ij} l(1) \beta^{[i]}(\bullet\ \bullet) - l(2,i)\alpha^{[1,i]}(\bullet\!\!-\!\!\bullet\ \bullet) + l(2,i)\alpha^{[2,i]}(\bullet\!\!-\!\!\bullet\ \bullet)\\
&= \sum_{j=1}^{s} a_{ij} l(1) \beta^{[j]}(\bullet\ \bullet) - l(2,i)l(1)\beta^{[i]}(\bullet\ \bullet\ \bullet) + l(2,i)\alpha^{[2,i]}(\bullet\!\!-\!\!\bullet\ \bullet)\\
&= \sum_{j=1}^{s} a_{ij} l(1) \beta^{[j]}(\bullet\ \bullet) - l(2,i)l(1)\beta^{[i]}(\bullet\ \bullet\ \bullet) + l(2,i)\beta^{[i]}(\bullet\!\!-\!\!\bullet\ \bullet)\\
&= l(1)\beta^{[i]}(\bullet\!\!<\!\!\bullet) - l(2,i)l(1)\beta^{[i]}(\bullet\ \bullet\ \bullet) + l(2,i)\beta^{[i]}(\bullet\!\!-\!\!\bullet\ \bullet)\\
&= \beta^{[i]}(\bullet\!\!-\!\!\bullet\!\!-\!\!\bullet).
\end{aligned}
$$

Substituting the corresponding summations yields

$$
\begin{aligned}
\sum_{j,k=1}^{s} a_{ij}a_{jk}c_k &= l(1)\sum_{j=1} a_{ij}c_j^2 - l(2,i)l(1)c_i^3 + l(2,i)c_i \sum_{j=1}^{s} a_{ij}c_j\\
&= l(1)\frac{c_i^3}{3} + l(2,i)\left(\frac{c_i^3}{2} - l(1)c_i^3\right).
\end{aligned}
$$

Note that the left hand side can be reduced to $\sum_{j,k=1}^{s} a_{ij}a_{jk}c_k = c_i^3/6$ *by assuming order at least* 3 *for the internal stages. There are two cases to be taken care of. The first one is* $l(1) = 0.5$*. Then the equation above holds true and* $l(2,i)$ *vanishes due to independence of* $\boldsymbol{A}_\Delta$*. In the second case* $l(1) \neq 1/2$*. Solving for* $l(2)$ *yields*

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

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*

$$
\alpha^{[2,i]}(\bullet\!\!-\!\!\bullet\!\!-\!\!\bullet) = \sum_{j=1}^{s} a_{ij}\alpha^{[1,j]}(\bullet\!\!-\!\!\bullet\!\!-\!\!\bullet) - \tilde{a}_{ii}^2\alpha^{[1,i]}(\bullet\!\!-\!\!\bullet\!\!-\!\!\bullet) + \tilde{a}_{ii}^2\alpha^{[2,i]}(\bullet\!\!-\!\!\bullet\!\!-\!\!\bullet) \tag{20}
$$

Since the tree in the last term is once again equal to that of the underlying collocation rule we only need to find

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

where in the last equality we used $\tilde{a}_{ii}^1 = c_i/2$. *Hence* (20) *can be simplified to*

$$\begin{aligned}\frac{c_i^4}{24} &= \sum_{j=1}^{s} a_{ij}\frac{c_j^3}{4} - \tilde{a}_{ii}^2\frac{c_i^3}{4} + \tilde{a}_{ii}^2\frac{c_i^3}{6} \\ &= \frac{c_i^4}{16} + \tilde{a}_{ii}^2(\frac{c_i^3}{6} - \frac{c_i^3}{4})\end{aligned}$$

Once again letting $\tilde{a}_{ii}^2 = l(2,i)c_i$, *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. *Consider SDC with* K *iterations and internal stages of height order* $\tilde{h}_k$, *performing an iteration with an EED satisfying* (19) *results in SDC with* $K+1$ *iterations and internal stages of height order* $\tilde{h}_{K+1} = \tilde{h}_K + 1$ *and order* $\tilde{p}_{K+1} \geq \tilde{h}_k + 2$.

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

3.2.2 Order jumps for non-linear problems

In this subsection, we derive the necessary and sufficient conditions on $\boldsymbol{A}_\Delta$ for an order jump using the simplifying assumptions $B(p), C(\eta), D(\zeta)$ of Gauss, Lobatto, and Radau collocation methods described in the beginning of Section 3.2. From now on, assume the SDC method $(\boldsymbol{A}_\Delta^0, \ldots, \boldsymbol{A}_\Delta^K, \boldsymbol{A}, \boldsymbol{b})$ is used to approximate a collocation Runge-Kutta method $(\boldsymbol{A}, \boldsymbol{b})$ of order p satisfying $B(p)$, $C(\eta)$, and $D(\zeta)$.

Lemma 1. *Consider an SDC with* K *iterations such that* $\boldsymbol{A}_\Delta^k$ *satisfies* $C_{W_k}(\eta_k)$ *with*

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

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

$$\alpha^{[K,i]}(\tau) = c_i^{|\tau|}\omega_K(\tau). \tag{21}$$

In the case, when $\boldsymbol{A}_\Delta^K$ is a diagonal matrix, it must be equal to $\boldsymbol{A}_\Delta^K = \text{diag}(\mathbf{c})W_K$ for some constant W_K to satisfy $C_{W_K}(\eta_K)$ for $\eta_K \geq 1$ where $W_{K,q} = W_K$ for all $q \in \mathbb{N}$. Note that the statement (21) can be extended multiplicatively from trees to forests

with the functional ω_K respecting the concatenation product, that is, $\omega_K(\pi_1 \cdot \pi_2) = \omega_K(\pi_1)\omega_K(\pi_2)$.

Proof. Let us prove the statement for $K = 1$ taking into account that $\boldsymbol{A}^0_\Delta = \mathbf{0}$ ($\alpha^{[0,i]} = 0$), and, therefore, given a tree $\tau = B^+(\pi)$ such that $1 < |\tau| \leq \eta_1$, we use (16) to obtain

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

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 $\tau = B^+(\pi)$ such that $|\tau| \leq \eta_K$, we use (16) to write

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

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

$$\alpha^{[K,i]}(\tau) = \sum_{j=1}^{s} a_{ij} c_j^{|\pi|} \omega_{K-1}(\pi) - \sum_{j=1}^{s} \tilde{a}^K_{ij} c_j^{|\pi|} \omega_{K-1}(\pi) + \sum_{j=1}^{s} \tilde{a}^K_{ij} c_j^{|\pi|} \omega_K(\pi),$$

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

$$\begin{aligned} \alpha^{[K,i]}(\tau) &= \frac{1}{|\tau|} c_i^{|\tau|} \omega_{K-1}(\pi) - c_i^{|\tau|} W_{K,|\tau|} \omega_{K-1}(\pi) + c_i^{|\tau|} W_{K,|\tau|} \omega_K(\pi), \\ &= c_i^{|\tau|} \big(\frac{1}{|\tau|} \omega_{K-1}(\pi) - W_{K,|\tau|} \omega_{K-1}(\pi) + W_{K,|\tau|} \omega_K(\pi) \big). \end{aligned}$$

We obtain the statement (21) with

$$\omega_K(\tau) := \frac{1}{|\tau|} \omega_{K-1}(\pi) - W_{K,|\tau|} \omega_{K-1}(\pi) + W_{K,|\tau|} \omega_K(\pi),$$

and the proof is finished. □

Lemma 2. *Consider an SDC with K iterations and a tree $\tau = B^+(\pi)$, we have*

$$\alpha^{[K]}(\tau) = \frac{1}{|\tau|} \omega_K(\pi), \tag{22}$$

if one of the following assumptions is satisfied:

1. *$|\tau| \leq \eta_K + 1$ and $\boldsymbol{A}^k_\Delta$ satisfies $C_{W_k}(\eta_k)$ with*

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

2. *$\eta_K + 1 < |\tau| \leq \eta_K + \zeta_K + 1$ and $\boldsymbol{A}^k_\Delta$ satisfies $C_{W_k}(\eta_k)$ and $D_{Y_k}(\zeta_k)$ with*

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

Proof. For a tree $\tau = B^+(\pi)$ of size $|\tau| \leq \eta_K + 1$, we use Lemma 1 and $B(p)$ to obtain,

$$\alpha^{[K]}(\tau) = \sum_{i=1}^{s} b_i \alpha^{[K,i]}(\pi) = \sum_{i=1}^{s} b_i c_i^{|\pi|} \omega_K(\pi) = \frac{1}{|\tau|} \omega_K(\pi), \tag{23}$$

which proves the statement. For any tree $\tau = B^+(\pi)$ of size $\eta_K + 1 < |\tau| \leq \eta_K + \zeta_K + 1$, if all trees in π have size less or equal to η_K, then (23) proves the statement.

Assume $\tau = B^+(\pi)$ of size $\eta_K + 1 < |\tau| \leq \eta_K + \zeta_K + 1$ and assume there exists a tree γ in π such that $|\gamma| \geq \eta_K + 1$, then, $|\pi \setminus \gamma| \leq \zeta_K - 1 \leq \eta_K$, which implies that γ is the only tree in π such that $|\gamma| \geq \eta_K + 1$. Let $q := |\pi \setminus \gamma|$. We use Lemma 1 to obtain

$$\alpha^{[K]}(\tau) = \sum_{i=1}^{s} b_i \alpha^{[K,i]}(\pi) = \sum_{i=1}^{s} b_i c_i^q \alpha^{[K,i]}(\gamma) \omega_K(\pi \setminus \gamma) = \alpha^{[K]}\big(B^+(\bullet^q \gamma)\big) \omega_K(\pi \setminus \gamma).$$

This implies that it is enough to prove the statement for trees of the form $B^+(\bullet^q \gamma)$. We have $q < \zeta_K \leq \zeta$, therefore, we can apply $D_{Y_K}(\zeta_K)$ and $D(\zeta)$. Let $\gamma = B^+(\pi_\gamma)$, then, using (16),

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

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

$$\begin{aligned}
\alpha^{[K]}\big(B^+(\bullet^q \gamma)\big) = &\frac{1}{q+1} \sum_{i=1}^{s} b_i (1 - c_i^{q+1}) \alpha^{[K-1,i]}(\pi_\gamma) - \sum_{i=1}^{s} b_i c_i^{q+1} Y_{K,q+1} \alpha^{[K-1,i]}(\pi_\gamma) \\
&+ \sum_{i=1}^{s} b_i c_i^{q+1} Y_{K,q+1} \alpha^{[K,i]}(\pi_\gamma)
\end{aligned}$$

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

$$\begin{aligned}
\alpha^{[K]}\big(B^+(\bullet^q \gamma)\big) = &\frac{1}{q+1} \big(\alpha^{[K-1]}(\gamma) - \alpha^{[K-1]}(B^+(\bullet^{q+1} \pi_\gamma)\big) \\
&- Y_{K,q+1} \alpha^{[K-1]}\big(B^+(\bullet^{q+1} \pi_\gamma)\big) + Y_{K,q+1} \alpha^{[K]}\big(B^+(\bullet^{q+1} \pi_\gamma)\big)
\end{aligned}$$

we use induction on the size of π_γ to get

$$\begin{aligned}
\alpha^{[K]}\big(B^+(\bullet^q \gamma)\big) = &\frac{1}{(q+1)|\gamma|} \omega_{K-1}(\pi_\gamma) - \frac{1}{(q+1)(|\gamma| + q + 1)} \omega_{K-1}(\bullet^{q+1} \pi_\gamma) \\
&- \frac{1}{|\gamma| + q + 1} Y_{K,q+1} \omega_{K-1}(\bullet^{q+1} \pi_\gamma) + \frac{1}{|\gamma| + q + 1} Y_{K,q+1} \omega_K(\bullet^{q+1} \pi_\gamma)
\end{aligned}$$

we use the property $\omega_K(\bullet\pi) = \omega_K(\pi)$ for any π to obtain

$$\alpha^{[K]}\big(B^+(\bullet^q \gamma)\big) = \frac{1}{|\gamma| + q + 1} \Big(\frac{|\gamma| + q + 1}{(q+1)|\gamma|} \omega_{K-1}(\pi_\gamma) - \frac{1}{q+1} \omega_{K-1}(\pi_\gamma)$$

$$- Y_{K,q+1}\omega_{K-1}(\pi_\gamma) + Y_{K,q+1}\omega_K(\pi_\gamma)\big)$$

which reduces to

$$\alpha^{[K]}\big(B^+(\bullet^q\gamma)\big) = \frac{1}{|\gamma|+q+1}\big(\frac{1}{|\gamma|}\omega_{K-1}(\pi_\gamma) - Y_{K,q+1}\omega_{K-1}(\pi_\gamma) + Y_{K,q+1}\omega_K(\pi_\gamma)\big)$$

We obtain (22) with

$$\omega_K(\bullet^q\gamma) = \omega_K(\gamma) = \frac{1}{|\gamma|}\omega_{K-1}(\pi_\gamma) - Y_{K,q+1}\omega_{K-1}(\pi_\gamma) + Y_{K,q+1}\omega_K(\pi_\gamma), \quad \text{and}$$
$$\omega_K(\pi) = \omega_K(\gamma)\cdot\omega_K(\pi\setminus\gamma),$$

which proves the statement of the lemma. □

We can now prove the following theorem.

Theorem 5. *Consider SDC with $K+1$ iterations. Let K^{th} iteration of SDC be of order p_K. We have $p_{K+1} = \min\{p_K+2, p\}$ if one of the following assumptions is satisfied:*

1. *$p_K < \eta_{K+1}$ and $\boldsymbol{A}^k_\Delta$ satisfies $C_{W_k}(\eta_k)$ with*

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

   where $W_{K+1,p_K+1} = \frac{1}{p_K+1}$.
2. *$\eta_{K+1} \le p_K < \eta_{K+1} + \zeta_{K+1}$ and $\boldsymbol{A}^k_\Delta$ satisfies $C_{W_k}(\eta_k)$ and $D_{Y_k}(\zeta_k)$ with*

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

   where $W_{K+1,p_K+1} = \frac{1}{p_K+1}$ and $Y_{K+1,q} = \frac{1}{p_K+1}$ for all $q = 1,\dots,\zeta_{K+1}$.

In particular, there exists a unique diagonal EED $\boldsymbol{A}^k_\Delta = \frac{\mathrm{diag}(\mathbf{c})}{2k} =: \boldsymbol{A}^k_{\Delta_J}$ for all iterations k such that the order of SDC increases by 2 per iteration.

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

$$\alpha^{[K+1]}(\tau) = \frac{1}{\gamma(\tau)}.$$

This follows for all trees $\tau \in T$ with $|\tau| \le p_K + 1$ due to Theorem 2. It remains to show that the same is true for all trees τ of size $|\tau| = p_K + 2$. We recall from Lemma 2 that, given $\tau = B^+(\pi)$ of size $p_K + 2$, we have,

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

and it remains to show that $\omega_{K+1}(\pi) = \frac{1}{\gamma(\pi)}$. For $\pi = \gamma_1\cdots\gamma_n$, we have,

$$\omega_{K+1}(\pi) = \omega_{K+1}(\gamma_1)\cdots\omega_{K+1}(\gamma_n).$$

If $n > 1$, then all trees γ_i have size less or equal to p_K and the statement is proved. Assume $n = 1$, then let γ_1 be denoted by γ. We note that $|\gamma| = p_K + 1$. If $p_K < \eta_{K+1}$, then, from the proof of Lemma 1, we have

$$\omega_K(\gamma) := \frac{1}{|\gamma|}\omega_{K-1}(\pi_\gamma) - W_{K,p_K+1}\omega_{K-1}(\pi_\gamma) + W_{K,p_K+1}\omega_K(\pi_\gamma) = \frac{1}{|\gamma|}\omega_{K+1}(\pi_\gamma),$$

where $\gamma = B^+(\pi_\gamma)$. 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, or there exists some $q \in 1, \ldots, \zeta_{K+1}$ such that

$$\omega_{K+1}(\gamma) = \frac{1}{|\gamma|}\omega_K(\pi_\gamma) - Y_{K+1,q}\omega_K(\pi_\gamma) + Y_{K+1,q}\omega_{K+1}(\pi_\gamma) = \frac{1}{|\gamma|}\omega_{K+1}(\pi_\gamma).$$

Since $|\pi_\gamma| = p_K$, $\omega_{K+1}(\pi_\gamma) = \frac{1}{\gamma(\pi_\gamma)}$ and we obtain

$$\alpha^{[K+1]}(\tau) = \frac{1}{|\tau|}\frac{1}{|\gamma|}\frac{1}{\gamma(\pi_\gamma)} = \frac{1}{\gamma(\tau)},$$

and the statement is proved. □

Corollary 2. *Choosing the* $\boldsymbol{A}^k_{\Delta_{\text{MIN-SR-NS}}} = \text{diag}(c_i/s)$ *introduced in [12] for all* k *results in an SDC method that performs an order jump on* $(s-1)^{th}$ *iteration.*

Remark 7. *Note that the prove of Theorem 5 only holds if* $\boldsymbol{A}$ *is a 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* $B(p), C(\eta), D(\zeta)$.

4 Stability and numerical experiments

This section addresses numerically the stability properties of the new SDC methods and the convergence of the iterations themselves towards underlying RKM as $k \to \infty$, outside the asymptotic limit $\Delta t \to 0$. We also confirm numerically the order jumps of the SDC iterations as analysed in Section 3.2 by considering the linear Dahlquist's test equation

$$u' = \lambda u, \qquad u(t_n) = u_n, \qquad t \in [t_n, t_{n+1}], \tag{24}$$

with $\lambda \in \mathbb{C}$ and $Re(\lambda) \leq 0$ and by considering the nonlinear Euler's rigid body dynamics [45]

$$\frac{1}{D_1 D_2 D_3}\nabla C \times \nabla H = \begin{pmatrix}\dot{Y}_1\\ \dot{Y}_2\\ \dot{Y}_3\end{pmatrix} = \begin{pmatrix}Y_2Y_3\\ Y_1Y_3\\ -Y_1Y_2\end{pmatrix}, \boldsymbol{Y}_0 = \begin{pmatrix}1/\sqrt{3}\\ 1\\ 0\end{pmatrix}, t \in [t_0 = 0, t_{end} = 10], \tag{25}$$

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 $\tilde{Q} = \sqrt{D_1D_2D_3/(N_1N_2N_3)}$. 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 $(1, 1, 0)^\intercal$.

4.1 Stability and convergence in the limit $k \to \infty$

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

$$\boldsymbol{e}^{k+1} = \boldsymbol{B}^k(z)\boldsymbol{e}^k = \boldsymbol{B}^k\boldsymbol{e}^k = \boldsymbol{B}^k \cdots \boldsymbol{B}^1\boldsymbol{e}^1 \tag{26}$$

with $\boldsymbol{B}^k(z) = z(\boldsymbol{I} - z\boldsymbol{A}^k_\Delta)^{-1}(\boldsymbol{A} - \boldsymbol{A}^k_\Delta)$. $\boldsymbol{B}^k(z)$ is called the iteration matrix. If it is stationary, i.e. $\boldsymbol{B}^k(z) = \boldsymbol{B}(z)$ and its spectral radius is < 1 the method $(\boldsymbol{A}_\Delta, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c})$ converges towards the collocation solution of $(\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c})$. In the non stationary case we need another measure of convergence.

Let $\Sigma_K := \left\{\boldsymbol{B}^K(z) \cdots \boldsymbol{B}^1(z) : \boldsymbol{B}^i(z) \in \Sigma\right\}$, where Σ is the set of all possible iteration matrices, and let $\hat{\rho}_K(\Sigma, \|\cdot\|) := \sup\{\|\tilde{\boldsymbol{B}}\|^{\frac{1}{K}} : \tilde{\boldsymbol{B}} \in \Sigma_K\}$. Define the joint spectral radius [46]

$$\hat{\rho}(\Sigma) := \lim_{k\to\infty} \hat{\rho}_k(\Sigma, \|\cdot\|).$$

By Theorem 1 in [47] (26) will converge for an arbitrary $\tilde{\boldsymbol{B}}$ if and only if $\hat{\rho}(\Sigma) < 1$. Due to the supremum in $\hat{\rho}_k$ it is the worst case growth rate of (26). However, in most cases we know $\tilde{\boldsymbol{B}}$ beforehand and, thus, we can relax the convergence criterion as follows. Let $\bar{\rho}_k(\tilde{\boldsymbol{B}}, \|\cdot\|) := \{\|\tilde{\boldsymbol{B}}\|^{\frac{1}{k}} : \tilde{\boldsymbol{B}} \in \Sigma_k\}$ and define the growth rate to be

$$\bar{\rho}(\tilde{\boldsymbol{B}}) := \lim_{k\to\infty} \bar{\rho}_k(\tilde{\boldsymbol{B}}, \|\cdot\|).$$

Hence, (26) converges if $\bar{\rho}(\tilde{\boldsymbol{B}}) < 1$ or if $\bar{\rho}_k(\tilde{\boldsymbol{B}}, \|\cdot\|) = 0$ for $k \leq K$. Note, the similarity of $\bar{\rho}_k(\tilde{\boldsymbol{B}}, \|\cdot\|)$ and Equation 4.4 in [8].

We analyse convergence of the SDC method $(\boldsymbol{A}^1_{\Delta_\mathrm{J}}, \dots, \boldsymbol{A}^K_{\Delta_\mathrm{J}}, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c})$ outside the asymptotic limit. The iteration matrix in iteration k reads

$$\boldsymbol{B}^k = z\left(\boldsymbol{I} - \frac{z}{2k}\boldsymbol{A}^{\mathrm{IP}}_\Delta\right)^{-1}\left(\boldsymbol{A} - \frac{1}{2k}\boldsymbol{A}^{\mathrm{IP}}_\Delta\right)$$

and

$$\lim_{k\to\infty} \boldsymbol{B}^k = z\boldsymbol{A}.$$

Hence, for a fixed z, the iteration matrix $\boldsymbol{B}^k$ approaches the iteration matrix of Picard iterations in the limit $k \to \infty$. Since the Picard iterations define an explicit RKM , we can not expect the method $(\boldsymbol{A}^1_{\Delta_\mathrm{J}}, \dots, \boldsymbol{A}^K_{\Delta_\mathrm{J}}, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c})$ to converge for $|z| \gg 1$ and large K. Fig. 3 compares $\bar{\rho}_k$ and stability domains of SDC methods using either $\boldsymbol{A}^k_{\Delta_\mathrm{J}}$ or $\boldsymbol{A}^k_{\Delta_\mathrm{P}}$. The stability domain of $\boldsymbol{A}^k_{\Delta_\mathrm{J}}$ shrinks with increasing k. This is in contrast to implicit-explicit SDC for which stability regions increase with more iterations [3].

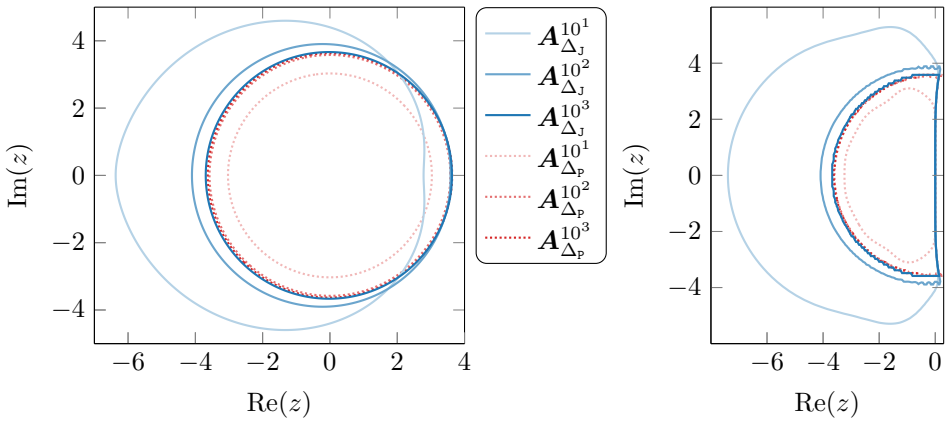


Fig. 3 $M = 3$ Radau IIA nodes were used for the figure. **Left**: $\bar{\rho}_k(\tilde{\boldsymbol{B}}, \|\cdot\|_\infty) \leq 1$ inside enclosed contour lines for $k = 10^1, 10^2, 10^3$ and $(\boldsymbol{A}^k_{\Delta_J}, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c})$ (blue solid lines) as well as Picard iterations $(\boldsymbol{A}^k_{\Delta_P}, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c})$ (dotted red lines). **Right**: Stability domain for $(\boldsymbol{A}^k_{\Delta_J}, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c})$ (blue solid lines) and $(\boldsymbol{A}^k_{\Delta_P}, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c})$ (red dotted lines) after $k = 10^1, 10^2, 10^3$ iterations. Stability domain is inside enclosed contour lines. For both plots the contours of $(\boldsymbol{A}^k_{\Delta_J}, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c})$ approach the contours of $(\boldsymbol{A}^k_{\Delta_P}, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c})$ with increasing iteration count k.

In the stiff limit we find $\lim_{z\to\infty} \boldsymbol{e}^{k+1} = \boldsymbol{B}^k_S \boldsymbol{e}^k$ with

$$\lim_{z\to\infty} \boldsymbol{B}(z) = \lim_{z\to\infty} \frac{z}{z}\left(\left(\frac{1}{z}\boldsymbol{I} - \boldsymbol{A}^k_\Delta\right)^{-1}\right)(\boldsymbol{A} - \boldsymbol{A}^k_\Delta) = \boldsymbol{I} - (\boldsymbol{A}^k_\Delta)^{-1}\boldsymbol{A} -: \boldsymbol{B}^k_S.$$

[12] find that using M iterations of $\boldsymbol{A}_{\Delta_{\text{MIN-SR-FLEX}}}$ in the stiff limit results in a nilpotent iteration matrix and, hence, convergence towards the collocation solution is ensured. Note, that their proof of Theorem 2.12 in [12], does not depend on the order of $\boldsymbol{A}^k_\Delta$. This allows us to ensure convergence for $\boldsymbol{A}_{\Delta_J}$ (and other EED's) as follows.

Proposition 6. *Let* $A_\Delta = \{\boldsymbol{A}^0_\Delta, \ldots, \boldsymbol{A}^K_\Delta : \boldsymbol{A}^i_\Delta \in \operatorname{diag}(\frac{\boldsymbol{c}}{c})\}$ *with* $c \in \mathbb{R} \setminus \{0\}$ *and let* $A_{\Delta_{\text{MIN-SR-FLEX}}} = \{\operatorname{diag}(\frac{\boldsymbol{c}}{1}), \ldots, \operatorname{diag}(\frac{\boldsymbol{c}}{M})\}$ *such that* $A_{\Delta_{\text{MIN-SR-FLEX}}} \subseteq A_\Delta$. *Consider the SDC method* $(A_\Delta, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c})$ *then* $\lim_{z\to\infty} \boldsymbol{e}^{K+1} = 0$.

Proof. In the stiff limit we find $\lim_{z\to\infty} \boldsymbol{e}^{K+1} = \boldsymbol{B}^K_S \cdots \boldsymbol{B}^1_S \boldsymbol{e}^1 = \tilde{\boldsymbol{B}}_S \boldsymbol{e}^1$. Hence, we require $\tilde{\boldsymbol{B}}_S = 0$ if $\boldsymbol{e}^1 \neq 0$. The proof that indeed $\tilde{\boldsymbol{B}}_S = 0$ works analogous to the proof of Theorem 2.12. in [12]. □

Corollary 3. *For the method* $(A_\Delta, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c})$ *it holds that* $\bar{\rho}_k(\tilde{\boldsymbol{B}}_S, \|\cdot\|) = 0$.

Note, that if the assumptions in proposition 6 hold $\bar{\rho}(\tilde{\boldsymbol{B}}_S) = 0$ in the limit $k \to \infty$. Hence, despite the equality of $\boldsymbol{A}^k_{\Delta_J}$ and $\boldsymbol{A}^k_{\Delta_P}$ in the stiff limit, we can, using the approach above, ensure convergence towards the collocation solution in the stiff limit (see Fig. 4).

4.2 Stability and convergence for a fixed number k of iterations

Fig. 5 confirms that $\boldsymbol{A}^k_{\Delta_J}$ reaches the expected convergence orders for up to 5 iterations (10th order) for linear and non-linear problems.

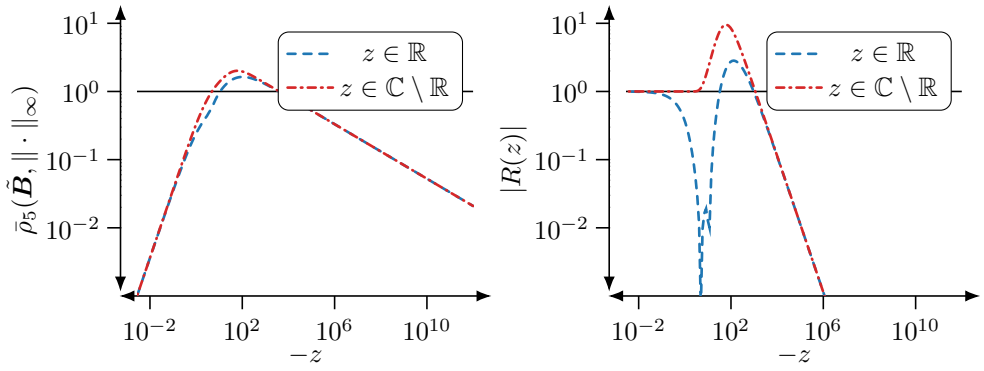


Fig. 4 SDC method $(\boldsymbol{A}_{\Delta_\mathrm{J}}, \boldsymbol{A}_{\Delta_\mathrm{J}}, \boldsymbol{A}_{\Delta_\mathrm{J}}, \mathrm{diag}(\boldsymbol{c}), \mathrm{diag}(\boldsymbol{c}/3), \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c})$ with $M = 5$ and RadauIIA nodes. **Left**: $\bar{\rho}_5(\tilde{\boldsymbol{B}}, \|\cdot\|_\infty)$ along the real (dashed, blue) and imaginary (dash dotted, red) axes. **Right**: $|R(z)|$ along the real (dashed, blue) and imaginary (dash dotted, red) axes. For both plots the graphs approach zero with increasing z, which is to be expected by proposition 6.

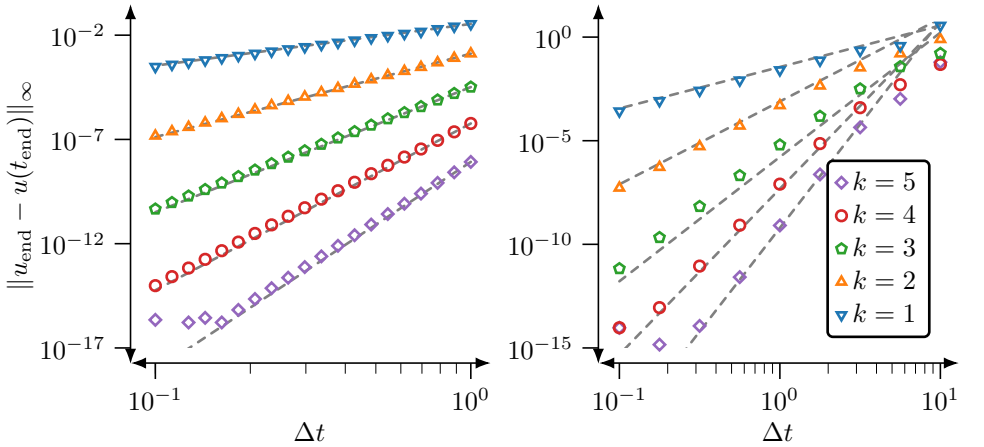


Fig. 5 Numerical convergence of SDC with $M = 6$ Radau IIA nodes and the EED introduced in Theorem 5. The last node was used as the solution in each timestep. The grey dashed lines serve as a guide to the eye with slopes of 2, 4, 6, 8 and 10. The method shows the expected 2nd, 4th, 6th, 8th, and 10th order convergence for $k = 1, 2, 3, 4, 5$ iterations (see 5). **Left**: Dahlquist test equation with $t_0 = 0$, $t_\mathrm{end} = 1$, $\lambda = -1$ and $u_0 = 1$. The data point of $k = 5$ close to $\Delta t = 10^{-1}$ that appears to be missing is below the scale shown. Within this area the error of the method with $k = 5$ is close to machine precision, which is why slope appears to be 0. **Right**: Euler rigid body equations. The implementation using SciPy `fsolve` limits the precision to $\sim 10^{-14}$. See Fig. A6.

Figure 6 shows stability regions for a selection of parallel EEDs using 5 and 3 Radau IIA nodes. Exclusively using $\boldsymbol{A}_{\Delta_\mathrm{J}}$ yields methods that are only suited to non-stiff problems. However, even for non-stiff problems the methods will be expensive due to the implicit solves. To overcome these limitations we need to mix different EEDs. If we include $\boldsymbol{A}^1_\Delta = \mathrm{diag}(\boldsymbol{c})$ and then proceed with $\boldsymbol{A}^k_\Delta = \mathrm{diag}(\boldsymbol{c})/(2k-1)$ we can see that the stability regions are larger compared to the previous case. However, the stability regions get smaller with increasing k. Along the imaginary axis the method won't be

stable for $k \geq 3$ and, hence, is not suited to integrate hyperbolic problems. The best results we could find for parallel SDC use s iterations of $\boldsymbol{A}_{\Delta_{\text{MIN-SR-FLEX}}}$, which results in a nilpotent iteration matrix in the stiff limit, and then proceed with $\boldsymbol{A}_\Delta = \text{diag}\,\boldsymbol{c}/(2k-s)$ as in the third example on the right hand side of Fig. 6. This approach often results in L-stable methods for more than just 2 subsequent iterations using $\boldsymbol{A}_{\Delta_\text{J}}$, nonetheless, this is not always the case and requires further studying. For serial SDC, using $\boldsymbol{A}_{\Delta_\text{J}}$, followed by $\boldsymbol{A}_{\Delta_\text{LU}}$ allows even more order jumps without sacrificing stability. These results show that combining different EEDs can result in useful methods that include order jumps and, hence, reduce computational work.

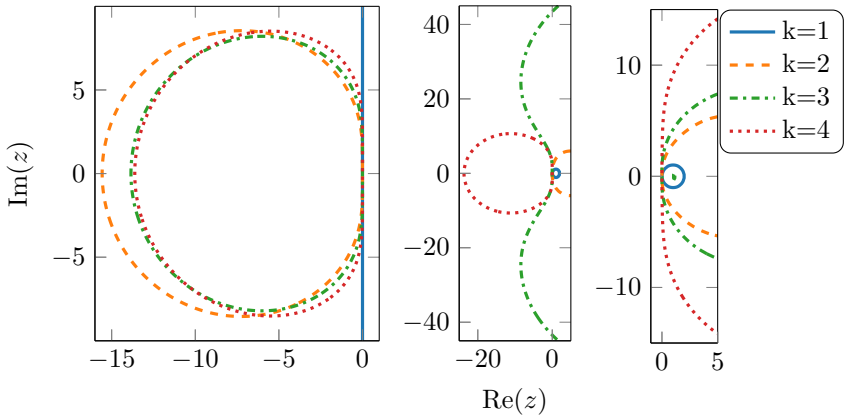


Fig. 6 Stability regions for different SDC methods using $s = 5$ (left, middle) and $s = 3$ (right) Radau IIA nodes. A method is either stable within its corresponding enclosed contour or left of a non enclosed contour. **Left**: SDC using $\boldsymbol{A}_\Delta^k = \text{diag}(\boldsymbol{c})/(2k)$. $k = 1$ shows the stability region of the trapezoidal rule. Despite the implicit nature of the methods, the stability regions for $k > 1$ are comparable to that of explicit methods due to the closed contour lines in the left half of the complex plane. The convergence order after $k = 1, 2, 3, 4$ is 2, 4, 6, 8, respectively. **Middle**: SDC using$\boldsymbol{A}_\Delta^1 = \text{diag}(\boldsymbol{c})$ and $\boldsymbol{A}_\Delta^k = \text{diag}(\boldsymbol{c})/(2k-1)$ for $k \in \{2, 3, 4\}$. $k = 1$ shows the contour of the backward Euler method. $k = 2$ is A- and L-stable, $k = 3$ is L($\alpha \approx 67.57°$)-stable (less suited to hyperbolic problems), $k = 4$ has a larger stability region compared to methods with $k > 1$ on the left hand side but looks like an explicit method due to the closed contour in the left half of the complex plane. Note that $k = 1, 2$ are barely visible as the contours open to the right half of the complex plane. The convergence order after $k = 1, 2, 3, 4$ is 1, 3, 5, 7, respectively. **Right**: SDC using $\boldsymbol{A}_{\Delta_{\text{MIN-SR-FLEX}}}^k$ for $k \leq s$ and $\boldsymbol{A}_\Delta^k = \text{diag}(\boldsymbol{c})/(2k-3)$ for $k = 4$. $k = 1$ shows the contour of the backward Euler method. All iterations are L-stable. The convergence order after $k = 1, 2, 3, 4$ is 1, 2, 3, 5, respectively. The three plots do not share the same axis.

Note, that if an EED that does order jumps is built such that it is a lower triangular matrix, there are many more degrees of freedom which may allow for fine tuning the convergence properties discussed above. While parallel EEDs have a unique set of coefficients on the diagonal, non-parallel EEDs allow to freely choose $n-1$ coefficients for the n^{th} node. Exploring these types of lower triangular EEDs is left for future work.

4.3 An S-conservative SDC method

In [48] an approach to conserve quadratic invariants within arbitrary RKM of order ≥ 2 is outlined. We will use their approach to enforce conservation of a single quadratic quantity in arbitrary SDC methods. Note that while the underlying collocation method can possess properties like symmetry or even symplecticity that ensure conservation of certain properties, SDC typically does not inherit these properties [9]. More precisely we want $u_{n+1}^\intercal S u_{n+1} = u_n^\intercal S u_n$, where $S \in \mathbb{R}^{q\times q}$ is a symmetric Matrix. This property is called S-conservative and requires $m_{ij} = b_i a_{ij} + b_j a_{ji} - b_i b_j = 0$ which a RKM in general violates. To make SDC S-conservative, we change (2b) to

$$u_{n+1} = u_n + \Delta t \gamma_n \sum_{i=1}^{s} b_i f(t_n + c_i \Delta t, u_n^{[i]}), \tag{27}$$

with

$$\gamma_n = \frac{2\sum_{ij}^{s} b_i a_{ij} \langle S f_i, f_j\rangle}{\sum_{ij}^{s} b_i b_j \langle S f_i, f_j\rangle}. \tag{28}$$

where $\langle\cdot,\cdot\rangle$ defines some inner product [48–50]. If the SDC method is of order p the solution of the relaxation RKM will be of order $p-1$ if interpreted as a numerical approximation of $u(t_n+\Delta t)$ and of order p if interpreted as a numerical approximation of $u(t_n+\gamma_n\Delta t)$ [49, Thm 3 and Cor 5]). In order to conserve the quadratic Hamiltonian H of (25), we set

$$S = \frac{1}{2}\begin{pmatrix} D_1 & 0 & 0 \\ 0 & D_2 & 0 \\ 0 & 0 & D_3 \end{pmatrix}$$

and calculate γ_n according to (28). Fig 7 (left) shows that SDC with relaxation conserves the H up to machine precision. Fig 7 (right) shows the global error over time for standard SDC and SDC with relaxation. For standard SDC the error grows linearly at first and then at around $t = 40$ starts to grow proportional to t^2, in line with the analysis by e.g. [20, 22]. By contrast, for SDC with relaxation the error growth only linearly with t and remains small even for simulations over very long times. Projection methods [50] could be used to build SDC methods that also conserve non-quadratic quantities but this is left for future work.

5 Summary

We provide a new derivation of spectral deferred corrections using the perspective of a Runge-Kutta method applied to a partitioned initial value problem. This allows to formulate a wide range of different flavours of SDC methods as RKM represented by Butcher tables and to analyze their convergence properties using the theory of B-series and the methodology of simplifying assumptions for order conditions. A general proof is given that any error equation discretization or "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 with every iteration which is required for optimal convergence in some parallelizable SDC

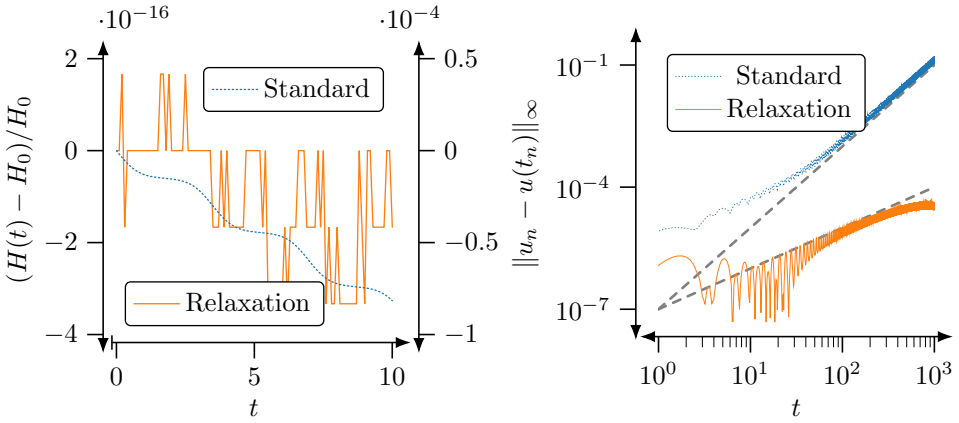


Fig. 7 SDC with $s = 3$ Gauss Legendre nodes with $K = 2$ iterations using an explicit Euler EED $\boldsymbol{Q}_{\Delta_E}$. **Left**: The blue dotted line shows the normalised Hamiltonian without relaxation and corresponds to the axis on the right. The orange line shows the normalised Hamiltonian with relaxation and corresponds to the axis on the left. **Right**: Global error of the method with (orange) and without (blue, dotted) relaxation with $t_{end} = 10^3$.

variants. Our framework also extends to EEDs that are algebraically motivated and are not consistent discretizations of the error equation. We provide a rigorous explanation of when parallel SDC methods can benefit from order jumps and deliver an increase of two orders in one iteration, a phenomenon that was observed numerically before but not yet explained theoretically. We also investigated the stability properties of the new methods and showed how the interpretation of SDC as RKM allows to construct variants that conserve quadratic invariants. It would be interesting to investigate in future work if the favourable stability properties of the underlying RKM could be preserved by the iterations of the SDC combined with order jumps in SDC iterations.

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. EB thankfully acknowledges the support of the Knut and Alice Wallenberg Foundation (grant number KAW 2023.0433).

Appendix A Order of various SDC methods

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

Fig. A1 Convergence order of SDC using $\boldsymbol{Q}_\Delta^T$, s Lobatto nodes (lines) 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	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
2	**3**	4	4	4	4	4	4	4	4	4	4	4	4	4	4
3	**3**	4	5	6	6	6	6	6	6	6	6	6	6	6	6
4	**3**	4	5	6	7	8	8	8	8	8	8	8	8	8	8
5	**3**	4	5	6	7	8	9	10	10	10	10	10	10	10	10
6	**3**	4	5	6	7	8	9	10	**12**	12	12	12	12	12	12
7	**3**	4	5	6	7	8	**10**	11	12	13	14	14	14	14	14
8	**3**	4	5	6	7	**9**	10	**12**	13	14	15	16	16	16	16

Fig. A2 Convergence order of SDC using $\boldsymbol{Q}_\Delta^T$, s Gauss nodes (lines) 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	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
2	**3**	4	4	4	4	4	4	4	4	4	4	4	4	4	4
3	2	**4**	5	6	6	6	6	6	6	6	6	6	6	6	6
4	2	3	**5**	6	7	8	8	8	8	8	8	8	8	8	8
5	2	3	4	**6**	7	8	9	10	10	10	10	10	10	10	10
6	2	3	4	5	**7**	8	9	10	11	12	12	12	12	12	12
7	2	3	4	5	6	**8**	9	10	11	12	13	14	14	14	14
8	2	3	4	5	6	7	**9**	10	11	12	13	14	15	16	16

Fig. A3 Convergence order of SDC using $\boldsymbol{A}^k_{\Delta_{\texttt{MIN-SR-NS}}}$, s Gauss nodes (lines) 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	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

Fig. A4 Convergence order of SDC using $\boldsymbol{A}^k_{\Delta_{\texttt{MIN-SR-NS}}}$, s Radau IIA nodes (lines) 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
2	**2**	2	2	2	2	2	2	2	2	2	2	2	2	2	2
3	1	**3**	4	4	4	4	4	4	4	4	4	4	4	4	4
4	1	2	**4**	5	6	6	6	6	6	6	6	6	6	6	6
5	1	2	3	**5**	6	7	8	8	8	8	8	8	8	8	8
6	1	2	3	4	**6**	7	8	9	10	10	10	10	10	10	10
7	1	2	3	4	5	**7**	8	9	10	11	12	12	12	12	12
8	1	2	3	4	5	6	**8**	9	10	11	12	13	14	14	14

Fig. A5 Convergence order of SDC using $\boldsymbol{A}^k_{\Delta_{\texttt{MIN-SR-NS}}}$, s Lobatto nodes (lines) 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

Fig. A6 Convergence order of SDC using $\boldsymbol{A}^k_{\Delta_{\texttt{Jumper}}}$, s Radau IIA nodes (lines) and k iterations (columns). Thick numbers indicate order jumps.

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] 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

[13] Hairer, E.: On the order of Iterated Defect Correction. Numerische Mathematik **29**(4), 409–424 (1978) https://doi.org/10.1007/BF01432878

[14] Gottlieb, S., Ketcheson, D.I., Shu, C.-W.: High order strong stability preserving time discretizations. Journal of Scientific Computing **38**(3), 251–289 (2009)

[15] Gottlieb, S., Ketcheson, D., Shu, C.-W.: Strong Stability Preserving Runge-Kutta and Multistep Time Discretizations. WORLD SCIENTIFIC, ??? (2011). https://doi.org/10.1142/7498

[16] 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

[17] 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

[18] 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

[19] 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

[20] Calvo, M., Laburta, M.P., Montijano, J.I., Rández, L.: Error growth in the numerical integration of periodic orbits. Mathematics and Computers in Simulation **81**(12), 2646–2661 (2011) https://doi.org/10.1016/j.matcom.2011.05.007

[21] Cano, B., Sanz-Serna, J.M.: Error growth in the numerical integration of periodic orbits, with application to hamiltonian and reversible systems. SIAM Journal on Numerical Analysis **34**(4), 1391–1417 (1997) https://doi.org/10.1137/S0036142995281152 https://doi.org/10.1137/S0036142995281152

[22] Ranocha, H., Ketcheson, D.I.: Relaxation Runge–Kutta methods for Hamiltonian problems. Journal of Scientific Computing **84**(1) (2020) https://doi.org/10.1007/s10915-020-01277-y

[23] Hansen, A.C., Strain, J.: Convergence theory for spectral deferred correction. University of California at Berkeley (2005)

[24] 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

[25] 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

[26] 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

[27] 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

[28] 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)

[29] Ketcheson, D.I., Ranocha, H.: Computing with B-series. ACM Trans. Math. Softw. **49**(2) (2023) https://doi.org/10.1145/3573384

[30] Hairer, E., Lubich, C., Wanner, G.: Geometric Numerical Integration: Structure-preserving Algorithms for Ordinary Differential Equations. Springer, ??? (2002). https://doi.org/10.1007/3-540-30666-8 . http://dx.doi.org/10.1007/3-540-30666-8

[31] 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

[32] Hairer, E., Wanner, G.: Solving Ordinary Differential Equations II: Stiff Problems. Springer, ??? (1996). https://doi.org/10.1007/978-3-642-05221-7 . http://dx.doi.org/10.1007/978-3-642-05221-7

[33] Widlund, O.B.: A note on unconditionally stable linear multistep methods. BIT **7**(1), 65–70 (1967) https://doi.org/10.1007/bf01934126

[34] Dahlquist, G.G.: A special stability problem for linear multistep methods. BIT **3**(1), 27–43 (1963) https://doi.org/10.1007/BF01963532

[35] 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)

[36] 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

[37] 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

[38] 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)

[39] 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

[40] Cayley, A.: On the theory of the analytical forms called trees. Philosophical Magazine **13**, 172–176 (1857) https://doi.org/10.1080/14786445708642866

[41] 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

[42] Hairer, E., Nørsett, S.P., Wanner, G.: Solving Ordinary Differential Equations I: Nonstiff Problems, 2nd edn. Springer, ??? (1993). https://doi.org/10.1007/978-3-540-78862-1 . http://dx.doi.org/10.1007/978-3-540-78862-1

[43] 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)

[44] 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 I Ndustrial and Applied Mathematics, pp. 311–331. Higher Ed. Press, Beijing, ??? (2015)

[45] 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 . https://doi.org/10.1007/978-3-540-70529-1_142

[46] Rota, G.-C., Strang, G.: A note on the joint spectral radius. Indag. Math **22**(4), 379–381 (1960)

[47] Berger, M.A., Wang, Y.: Bounded semigroups of matrices. Linear Algebra and its Applications **166**, 21–27 (1992)

[48] Del Buono, N., Mastroserio, C.: Explicit methods based on a class of four stage fourth order Runge–Kutta methods for preserving quadratic laws. Journal of Computational and Applied Mathematics **140**(1–2), 231–243 (2002) https://doi.org/10.1016/s0377-0427(01)00398-3

[49] Ketcheson, D.I.: Relaxation Runge-Kutta Methods: Conservation and stability for Inner-Product Norms. arXiv (2019). https://doi.org/10.48550/ARXIV.1905.09847

[50] Ranocha, H., Sayyari, M., Dalcin, L., Parsani, M., Ketcheson, D.I.: Relaxation Runge–Kutta methods: Fully discrete explicit entropy-stable schemes for the compressible Euler and Navier–Stokes equations. SIAM Journal on Scientific Computing **42**(2), 612–638 (2020) https://doi.org/10.1137/19m1263480