Communications in Applied Mathematics and Computational Science

ON THE CONVERGENCE OF SPECTRAL DEFERRED CORRECTION METHODS

MATHEW F. CAUSLEY AND DAVID C. SEAL

vol. 14 no. 1 2019

msp

COMM. APP. MATH. AND COMP. SCI.
Vol. 14, No. 1, 2019
dx.doi.org/10.2140/camcos.2019.14.33

ON THE CONVERGENCE OF SPECTRAL DEFERRED CORRECTION METHODS

MATHEW F. CAUSLEY AND DAVID C. SEAL

In this work we analyze the convergence properties of the spectral deferred correction (SDC) method originally proposed by Dutt et al. (BIT **40** (2000), no. 2, 241–266). The framework for this high-order ordinary differential equation (ODE) solver is typically described as a low-order approximation (such as forward or backward Euler) lifted to higher-order accuracy by applying the *same* low-order method to an error equation and then adding in the resulting defect to correct the solution. Our focus is not on solving the error equation to increase the order of accuracy, but on rewriting the solver as an iterative Picard integral equation solver. In doing so, our chief finding is that it is not the low-order solver that picks up the order of accuracy with each correction, but it is the underlying quadrature rule of the right-hand-side function that is solely responsible for picking up additional orders of accuracy. Our proofs point to a total of three sources of errors that SDC methods carry: the error at the current time point, the error from the previous iterate, and the numerical integration error that comes from the total number of quadrature nodes used for integration. The second of these two sources of errors is what separates SDC methods from Picard integral equation methods; our findings indicate that as long as the difference between the current and previous iterates always gets multiplied by at least a constant multiple of the time step size, then high-order accuracy can be found even if the underlying ODE "solver" is inconsistent. From this vantage, we solidify the prospects of extending spectral deferred correction methods to a larger class of solvers, of which we present some examples.

1. Introduction

The spectral deferred correction (SDC) method defines a large class of ordinary differential equation (ODE) solvers that were originally introduced in 2000 by Dutt, Greengard, and Rokhlin [12]. These types of methods are typically introduced by

We would like to thank the anonymous referees for their thoughtful comments, suggestions that improved the quality of this manuscript, and recommendations for future research. The work of Seal was supported by the Naval Academy Research Council.
MSC2010: 65L05, 65L20.
Keywords: initial-value problems, spectral deferred correction, Picard integral, semi-implicit methods.

defining an *error equation*, and then repeatedly applying the same low-order solver to the error equation and adding the solution back into the current approximation in order to pick up an order of accuracy. This idea can be traced back to the work of Zadunaisky in 1976 [40], who sought out high-order solvers in order to reduce numerical roundoff errors for astronomical applications. Before introducing the classical SDC methods defined in [12], we stop here to point out some of the recent work that has been happening over the past two decades including [30; 28; 6; 26; 7; 4; 22]. We refer the interested reader to [32] for a nice list of references for the first of these last two decades. Here, we provide a sampling of some of the current topics of interest to the community.

Many variations of the original SDC method are being studied as part of an effort to expedite the convergence of the solver. The chief goal here is to reduce the total number of iterations required to obtain the same high-order accuracy of the original method. These methods include the option of using Krylov deferred correction methods [20; 21] as well as the *multilevel* SDC methods [27; 36]. The multilevel approach starts with a lower-order interpolant and then successively increases the degree of the interpolant with each future sweep of the method. This has the primary advantage of decreasing the overall number of function evaluations that need to be conducted, but introduces additional complications involving the need to evaluate interpolating polynomials. In the same vein, higher-order embedded integrators have been explored within the so-called integral deferred correction (IDC) framework [6; 7], where a moderate order solver (such as second- or fourth-order Runge–Kutta method) is embedded inside a very high-order SDC solver. With this framework, each successive correction increases the order by the same amount as that of the base solver. In addition, parallel in time solvers [9; 31; 5; 13] are being investigated as a mechanism to address the needs of modern high-performance computing architectures, and adaptive time stepping options have been more recently investigated in [10]. This work is based upon the nice property that SDC methods naturally embed a lower-order solver inside a higher-order solver.

In addition to the above mentioned extensions, various semi-implicit formulations have been, and are currently being, explored. While the original solver was meant for classical nonlinear ODEs, semi-implicit formulations have been derived as early as 2003 [30] and are still an ongoing topic of research [29; 4]. The effect of the choice of correctors including second-order semi-implicit solvers for the error equation has been researched in [25], and an investigation into the efficiency of semi-implicit and multi-implicit spectral deferred correction methods for problems with varying temporal scales has been conducted in [26]. Related high-order operator splitting methods have been proposed in [15; 3; 8], where the focus is not on an implicit-explicit splitting, but rather on splitting the right-hand side of the ODE into smaller systems that can be more readily inverted with each sweep of the solver.

Very recent work includes applications of the SDC framework to generate exponential integrators of arbitrary orders [2], exploring interesting LU decompositions of the implicit Butcher tableau on nonequispaced grids [38], further investigation into high-order operator splitting [11], a comparison of essentially nonoscillatory (ENO) versus piecewise parabolic methods (PPM) coupled with SDC time integrators [23], and additional implicit-explicit (IMEX) splittings for fast-wave slow-wave splitting constructed from within the SDC framework [35].

It is not our aim to conduct a comprehensive review and comparison of all of these methods; rather it is our goal to present rigorous analysis of the original method that can be extended to these more complicated solvers. With that in mind, we now turn to a brief introduction of the spectral deferred correction framework, and in the process of doing so, we seek to directly compare this method with that of the Picard integral formulation of a numerical ODE solver.

1A. *Picard iteration and the SDC framework.*

We begin by giving a brief description of classical SDC methods. In doing so, we explain the differences between SDC and Picard iteration, which defines the cornerstone of the present work.

Classical SDC solvers are designed to solve initial value problems of the form

$$y' = \frac{dy}{dt} = f(y), \quad t > 0, \; y(0) = y_0, \tag{1}$$

where y can be taken to be a vector of unknowns. The solution $y(t)$ can be expressed as an integral through formal integration:

$$y(t) = y_0 + \int_0^t f(y(s))\,ds, \quad t > 0. \tag{2}$$

In this work, we assume that f is Lipshitz continuous. That is, we assume

$$|f(z) - f(w)| \le L|z - w|, \tag{3}$$

for some constant $L \ge 0$ and all $z, w \in \mathbb{R}$. This is sufficient to guarantee existence and uniqueness for solutions of IVP (1), and produce rigorous numerical error bounds for SDC methods.

Consider a set of M quadrature points $0 \le \xi_1 < \cdots < \xi_M \le 1$ that partition the unit interval into a total of N disjoint subintervals, defined by

$$N = \begin{cases} M - 1 & \text{if both endpoints are used,} \\ M & \text{if only one endpoint is used,} \\ M + 1 & \text{if neither endpoint is used.} \end{cases}$$

We make this choice because a given quadrature rule may or may not include the endpoints of the interval, and this convention allows us to study Gaussian quadrature rules, uniformly spaced quadrature rules, Radau II quadrature rules, and others all within the same context. With that in mind, we define the right endpoints ξ_n^R, for $n = 0, 1, \ldots N-1$, of each of the N subintervals as

$$\xi_n^R = \begin{cases} \xi_{n+1} & \text{if the left endpoint is included,} \\ \xi_n & \text{if the left endpoint not included,} \end{cases}$$

and $\xi_0^R = 0$ and $\xi_N^R = 1$ for the two boundary edge cases. Next, we define quadrature weights by

$$w_{n,m} = \int_{\xi_{n-1}^R}^{\xi_n^R} \ell_m(x)\,dx, \quad n = 1, 2, \ldots N,\ m = 1, 2, \ldots M, \tag{4}$$

where $\ell_m(x)$ is the Lagrange interpolating polynomial of degree at most $M-1$ corresponding to the quadrature point ξ_m:

$$\ell_m(x) = \frac{1}{c_m} \prod_{k=1,\,k\neq m}^{M} (x - \xi_k), \qquad c_m = \prod_{k=1,\,k\neq m}^{M} (\xi_m - \xi_k). \tag{5}$$

Once these weights are obtained, approximate integral solutions, say $\eta_m \approx y(\xi_m^R h)$ for $h > 0$ and $m = 0, 1, \ldots, N$, can be formed via

$$\text{(fully implicit collocation)} \quad \eta_n = \eta_{n-1} + h \sum_{m=1}^{M} w_{n,m} f(\eta_m), \quad n = 1, 2, \ldots, N, \tag{6}$$

whereas the exact solution $y_m := y(\xi_m^R h)$ satisfies the exact integral

$$y_n = y_{n-1} + \int_{t_{n-1}}^{t_n} f(y(t))\,dt, \quad t_n = \xi_n^R h,\ n = 1, 2, \ldots, N. \tag{7}$$

By convention, $\eta_0 := y_0$ is known to high order (because it comes from the previous time step), and $\eta_N \approx y(h)$ constitutes one “full” time step. Since each substep uses information from all substeps to construct the right-hand side, the solution is higher order, but also requires the solution of a nonlinear system of M unknowns (one for each quadrature point) at each time step. Although this integrator has some very nice properties (e.g., it can be made to be symplectic and L-stable for suitably chosen quadrature points), it is not typically used in practice given the additional storage requirements and the larger matrices that need to be inverted for each time step. This is particularly relevant when it is used as the base solver for a partial differential equation, but even these bounds are being explored as a viable option for PDE solvers such as the discontinuous Galerkin method [33].

In place of the fully implicit collocation method, Picard iteration (with numerical quadrature) defines a solver by iterating on a current solution $\eta_n^{[p]}$, $p \in \mathbb{Z}_{\geq 0}$, and then creates a better approximation through

$$\text{(Picard iteration)} \qquad \eta_n^{[p+1]} = \eta_{n-1}^{[p+1]} + h \sum_{m=1}^{M} w_{n,m} f\big(\eta_m^{[p]}\big), \quad n = 1, 2, \ldots N. \tag{8}$$

Note that the current value $\eta_0^{[p+1]} := \eta_0 \approx y_0$ is a known value that is equal to the exact solution up to high order. While this solver picks up a single order of accuracy with each correction, it has the unfortunate consequence of having a finite region of absolute stability.

The explicit spectral deferred correction framework is

$$\text{(explicit SDC)}\ \eta_n^{[p+1]} = \eta_{n-1}^{[p+1]} + h_n\big[f\big(\eta_{n-1}^{[p+1]}\big) - f\big(\eta_{n-1}^{[p]}\big)\big] + h\sum_{m=1}^{M} w_{n,m} f\big(\eta_m^{[p]}\big), \tag{9}$$

where $h_n = (\xi_n^R - \xi_{n-1}^R)h$ is the length of the n-th subinterval. This solver also has a finite region of absolute stability.

Remark. Although traditional SDC methods were originally cast as a method that corrects a provisional solution by solving an error equation, some modern descriptions of the same solver identify (9) as the base solver, which has the added benefit of pointing out a solid link between SDC methods and iterative Picard integral equation solvers.

In order to construct methods that have more favorable regions of absolute stability for stiff problems, the implicit SDC framework exacts multiple backward Euler time steps through each iteration with

$$\text{(implicit SDC)}\ \eta_n^{[p+1]} = \eta_{n-1}^{[p+1]} + h_n\big[f\big(\eta_n^{[p+1]}\big) - f\big(\eta_n^{[p]}\big)\big] + h\sum_{m=1}^{M} w_{n,m} f\big(\eta_m^{[p]}\big). \tag{10}$$

Note that this framework allows for implicit and high-order solutions to be constructed with greater computational efficiency when compared to the fully implicit collocation solver defined in (6) because smaller systems need to be inverted in order to take a single time step.

Remark. It has been noted that the scaling in front of the h_n term does not have impact on the order of accuracy [39]. It is our aim with this work to solidify that claim with rigorous numerical bounds, which we do for both the explicit and implicit solvers.

Before doing so, we point out an aside that is in common with all SDC solvers.

Remark. If $\lim_{p\to\infty} \eta_n^{[p]} = \eta_n$ converges, then solutions to (8), (9), and (10) converge to that of the fully implicit collocation method defined in (6).

While some SDC methods work with a fixed number of iterates in order to obtain a desired order of accuracy, there are many examples in the literature where convergence of the SDC iterations to the fully implicit scheme is considered. For example, the work in [38] is wholly concerned with this convergence, and not the accuracy of the underlying method for fixed iterations. Moreover, for stiff problems, it is well understood that using a fixed number of iterations can lead to order reduction which negates the advantage of using SDC in the first place. In addition, the multilevel spectral deferred correction (MLSDC) methods also are typically iterated to a residual tolerance, since one cannot be sure that coarse level sweeps will provide enough increase in accuracy (or decrease in the residual) [36].

One key advantage of iterating an SDC method to convergence is that when this is done, the method inherits well known and desirable properties that the fully implicit collocation method enjoys. For example, Kuntzmann [24] and Butcher [1] separately point out that if a total of M Gaussian quadrature points are used, then the fully implicit collocation method will have superconvergence order $\mathcal{O}(h^{2M})$. (For more details, we refer the interested reader to the excellent tomes of Hairer, Wanner et al. [16; 17; 18]. For example, see [16, §II.7], [17, Theorem 5.2], or [18, Theorem 1.5].) In general, the maximum order of accuracy for the underlying solver with M quadrature points is $\mathcal{O}(h^{2M})$ if they are Gauss–Legendre points, $\mathcal{O}(h^{2M-1})$ for the RadauIIA points, and $\mathcal{O}(h^{2M-2})$ for Gauss–Lobatto points. (The local truncation error is one order higher.) Uniform points have $\mathcal{O}(h^{M})$ order of convergence if M is even, and $\mathcal{O}(h^{M+1})$ if M is odd. The extra pickup in the order of accuracy is due to symmetry of the quadrature rule. (For example, $M = 1$ points reproduces the so-called "midpoint" rule, $M = 3$ reproduces Simpson's rule, and $M = 5$ yields Boole's rule, each of which pick up an extra order of accuracy.)

1B. ***An outline of the present work.*** Despite the increasing popularity of spectral deferred correction solvers, very little work has been performed on convergence results for this large class of methods. The results that are currently in the literature [15; 6; 7; 19; 37] typically proceed via induction on the current order of the approximate solution, and they all hinge on solving the *error equation*, wherein the same low-order solver is applied and then a defect, or correction, is added back into the current solution in order to increase its overall order of accuracy. In other recent work [34], the authors consider SDC methods as fixed-point iterations on a Neumann series expansion. There, the low-order method is viewed as an efficient preconditioner (in numerical linear algebra language), and the SDC iterations are thought of as simplified Newton iterations. Additionally, the work in [38] makes use of linear algebra techniques in order to optimize coefficients so that the method converges faster to the collocation solution for stiff problems.

In this work, we do not require the use of the error equation, nor do we work with any sort of defect such as that defined in [40]; rather we instead focus on the

Picard integral underpinnings inherent to all SDC methods. Our work solely uses fundamental numerical analysis tools: error estimates for numerical interpolation and integration. While these tools do rely on quadrature rules, our proofs are generic enough to accommodate any set of quadrature points, which are an ongoing discussion in terms of how to construct base solvers.

In this work, we prove rigorous error bounds for both implicit and explicit SDC methods, and in doing so, we expect the reader will find that these methods can be thought of as being built upon classical Picard iteration. Our results are applicable for general quadrature rules, but unlike the findings found in [37], where convergence is proven using the error equation, our work relies on the fundamental mechanics behind why the solver works. That is, we point out that the primary contributor to the order of accuracy of the solver lies within the integral of the residual, and not necessarily the application of any base solver to an error equation.

Indeed, our proofs follow in a manner similar to the proof of the Picard–Lindelöf theorem, but our proofs take into account numerical quadrature errors and do not rely on exact integration of the right-hand-side function $f(y)$. The primary differences between our proofs and that of the Picard–Lindelöf theorem are the following:

- Spectral deferred correction methods require the use of *numerical quadrature* to approximate the integrals presented in the Picard–Lindelöf theorem. Our error estimates take into account any errors resulting from quadrature rules.
- Each correction step in the *implicit scheme* defined in (10) requires a nonlinear inversion, whereas the Picard–Lindelöf theorem is typically proven using exact integration.

There are two main results in this work, one for explicit SDC methods and one for implicit SDC methods. These are both found as corollaries to a single theorem on semi-implicit SDC. In each case, we produce rigorous error bounds that are applicable for generic quadrature rules. Furthermore, we find that there are a total of three sources of error that SDC methods carry: the error from the previous time step, the error from the previous iterate, and the error from the quadrature rule being used.

The outline of this paper is as follows. In Section 2, we present some necessary lemmas concerning error estimates for integrals of interpolants as well as some error estimates for sequences of inequalities that show up in our proofs. In Section 3 we present a convergence proof for the more general case of a semi-implicit SDC solver, and then immediately point out two corollaries that prove implicit and explicit SDC methods converge. In Section 4, we present results for an SDC method that makes use of a higher-order base solver, the trapezoidal rule. In Section 5 we present some numerical results, where we compare explicit SDC methods with Picard iterative methods, we investigate modified implicit SDC methods, and we experiment with different semi-implicit formulations of SDC methods. Error estimates for all of these variants come from direct extensions of the proofs found in this work. Finally, some conclusions and suggestions for future work are drawn up in Section 6.

2. Preliminaries

We now point out a couple of important tools that we use to show that SDC solvers converge. Our aim is to focus on a single time step. Without loss of generality, from here on out we will focus on constructing a solution over the interval $[0, h]$, where h is the time step size and we will assume that $\eta_0 \approx y_0$ is a high-order approximation to the exact solution.

2A. ***Error estimates for integrals of interpolants.*** If $\boldsymbol{\eta} = (\eta_1, \eta_2, \dots, \eta_M)$ is a set of discrete values and $t \in [0, h]$ is a time interval we are interested in studying, we define the *interpolation operator I* to be the projection onto the space of polynomials of degree at most $M-1$ via

$$I[f(\boldsymbol{\eta})](t) := \sum_{m=1}^{M} f(\eta_m) l_m(t/h), \quad f(\boldsymbol{\eta}) := (f(\eta_1), f(\eta_2), \dots, f(\eta_M)). \tag{11}$$

Note that this produces the integration identity

$$\int_{t_{n-1}}^{t_n} I[f(\boldsymbol{\eta})](t)\, dt = h \sum_{m=1}^{M} w_{n,m} f(\eta_m) \tag{12}$$

after integrating (11) over a subinterval $[t_{n-1}, t_n] := [h\xi^R_{n-1}, h\xi^R_n]$, and the weights are defined as in (4).

Convergence results for both the explicit and the implicit SDC method (as well as Picard iteration) require the use of the following lemma.

Lemma 2.1. *Suppose that $f \circ y \in C^M([0, h])$, $\left\| \frac{d^M}{dt^M}(f \circ y) \right\|_\infty \le F$, and f is Lipschitz continuous with Lipshitz constant L. Then we have the estimate*

$$\left| \int_{t_{n-1}}^{t_n} I[f(\boldsymbol{\eta})](t) - f(y(t))\, dt \right| \le h \|\boldsymbol{\eta} - \boldsymbol{y}\| W_n L + \frac{F}{M!} h^{M+1}, \tag{13}$$

where the discrete norm is defined by

$$\|\boldsymbol{e}\| := \max_{1 \le n \le M} |e_n|, \quad \boldsymbol{e} = (e_1, e_2, \dots, e_M), \tag{14}$$

and the constant W_n is defined by

$$W_n := \sum_{m=1}^{M} \int_{\xi^R_{n-1}}^{\xi^R_n} |l_m(\xi)|\, d\xi. \tag{15}$$

For a fixed quadrature rule, this constant is finite and independent of the function.

Proof. Add and subtract the Lagrange interpolant $I[f(\boldsymbol{y})](t)$ for $f \circ y$ inside the left-hand side of (13) and apply the triangle inequality:

$$\left|\int_{t_{n-1}}^{t_n} I[f(\boldsymbol{\eta})](t) - f(y(t))\,dt\right| \le \left|\int_{t_{n-1}}^{t_n} I[f(\boldsymbol{\eta})](t) - I[f(\boldsymbol{y})](t)\,dt\right| + \left|\int_{t_{n-1}}^{t_n} I[f(\boldsymbol{y})](t) - f(y(t))\,dt\right|. \quad (16)$$

An estimate for the first of these two terms follows by linearity of the interpolation operator:

$$\begin{aligned}\left|\int_{t_{n-1}}^{t_n} I[f(\boldsymbol{\eta})](t) - I[f(\boldsymbol{y})](t)\,dt\right| &= \left|h\sum_{m=1}^{M}\omega_{n,m}(f(\eta_m) - f(y_m))\right| \\ &\le h\sum_{m=1}^{M}|\omega_{n,m}||f(\eta_m) - f(y_m)| \\ &\le hL\sum_{m=1}^{M}|\omega_{n,m}||\eta_m - y_m| \\ &\le hL\|\boldsymbol{\eta} - \boldsymbol{y}\|\sum_{m=1}^{M}|\omega_{n,m}|. \end{aligned} \quad (17)$$

The quadrature weights in this estimate are bounded above by

$$|\omega_{n,m}| \le \int_{\xi_{n-1}^R}^{\xi_n^R} |\ell_m(\xi)|\,d\xi$$

and then summed over all m to produce the constant W_n.

The second of the two integrals in (16) is a function solely of the smoothness of f and the choice of the quadrature rule. That is, classical interpolation error estimates result in a bound on the M-th derivative of $f \circ y$ through a single point $z(t) \in [0, h]$ that yields

$$|I[f(\boldsymbol{y})](t) - f(y(t))| = \left|\frac{(f \circ y)^{(M)}(z(t))}{M!}\prod_{m=1}^{M}(t - t_m)\,dt\right| \le \frac{F}{M!}\prod_{m=1}^{M}|t - t_m|. \quad (18)$$

Because $|t - t_m| \le h$ for each m, the result follows after integration. □

We stop to point out that due to the Runge phenomenon, the coefficient W_n defined in (15) can become quite large if a large number of quadrature points are chosen for constructing the polynomial interpolants required for the SDC method. In Table 1, we demonstrate a few sample values when uniform, Chebyshev, Gauss–Legendre, Gauss–Radau, and Gauss–Lobatto quadrature nodes are used to construct

	type of quadrature points				
M	uniform	Chebyshev	Legendre	Gauss–Radau	Gauss–Lobatto
2	1.000	1.207	1.366	1.500	1.000
3	1.000	1.244	1.479	1.558	1.000
4	1.056	1.257	1.527	1.578	1.000
5	1.152	1.263	1.551	1.586	1.000
6	1.257	1.266	1.566	1.591	1.000
7	1.362	1.268	1.575	1.594	1.000
8	1.663	1.269	1.581	1.596	1.000
9	2.550	1.270	1.585	1.597	1.000
10	4.028	1.271	1.588	1.598	1.000
11	6.506	1.271	1.590	1.599	1.000
12	10.963	1.271	1.592	1.599	1.000
13	18.340	1.272	1.594	1.600	1.000
14	32.060	1.272	1.595	1.600	1.000
15	54.998	1.272	1.596	1.600	1.000
16	98.531	1.272	1.596	1.600	1.000
17	172.176	1.272	1.597	1.601	1.000
18	313.675	1.272	1.597	1.601	1.000
19	556.491	1.273	1.598	1.601	1.000
20	1026.313	1.273	1.598	1.601	1.000
30	496210.554	1.273	1.600	1.602	1.000
50	208948162475.383	1.273	1.601	1.602	1.000

Table 1. Maximum size of the Lagrange polynomials $\max_{1\le n\le M} \max_{\xi\in[0,1]} |\ell_n(\xi)|$ for different quadrature points. The Gauss–Legendre, Gauss–Radau, and Gauss–Lobatto quadrature rules with M points have degrees of precision $2M+1, 2M$, and $2M-1$, respectively.

the polynomial interpolants. Uniform quadrature points tend to start performing quite poorly in the teens; however, even a small amount of points, say five or six, produces a high-order numerical method compared to other ODE solvers because in this regime the error constant is reasonable. The selection of quadrature points that minimizes this portion of the error constant is the Gauss–Lobatto nodes, but because convergence is found through refinement in h rather than p, any of these points will produce a method that converges, provided the exact solution has a suitable degree of regularity.

2B. ***Error estimates for sequences of inequalities.*** Finally, we require a second lemma as well as a simple corollary. Both of these are stated in [14], and their proofs are elementary.

Lemma 2.2. *If* $\{a_n\}_{n\in\mathbb{Z}_{\geq 0}}$ *is a sequence that satisfies* $|a_n| \leq A|a_{n-1}| + B$ *with* $A \neq 1$, *then*

$$|a_n| \leq A^n|a_0| + \frac{A^n - 1}{A - 1}B. \tag{19}$$

Proof. Recursively apply the inequality, and sum the remaining finite geometric series. □

Corollary 2.3. *If* $A > 1$ *and* $\{a_n\}_{n\in\mathbb{Z}}$ *is a sequence that satisfies* $|a_n| \leq A|a_{n-1}| + B$, *then*

$$|a_n| \leq A^n|a_0| + nA^{n-1}B \tag{20}$$

for every n.

Proof. By Lemma 2.2, the sequence satisfies (19). We estimate the (finite) geometric series by

$$\frac{A^n - 1}{A - 1} = 1 + A + \cdots + A^{n-1} \leq nA^{n-1} \tag{21}$$

because there are a total of n terms and each $A^l \leq A^{n-1}$ for $l = 0, 1, \dots, n-1$. □

With these preliminaries out of the way, we are now ready to state and prove our main result.

3. Convergence results

In place of separately proving explicit and implicit results for (1), we instead consider an umbrella class of ODEs, defined through a semi-implicit formulation:

$$y' = f(y), \quad f(y) = f_I(y) + f_E(y), \; y(0) = y_0, \tag{22}$$

where f_I is to be treated implicitly and f_E is to be treated explicitly. We assume that both f_I and f_E have Lipshitz constants L_I and L_E, respectively. In turn, this implies that f has a Lipshitz constant of $L := L_I + L_E$. In the case where $f_I \equiv 0$, we set $L_I = 0$, and in the case where $f_E \equiv 0$, we set $L_E = 0$.

The classical semi-implicit SDC (SISDC) method for (22) begins with a *provisional solution*, or initial guess $\eta_n^{[0]} \approx y(\xi_n h)$, that is typically defined with

$$\eta_n^{[0]} = \eta_{n-1}^{[0]} + h_n f_I\big(\eta_n^{[0]}\big) + h_n f_E\big(\eta_{n-1}^{[0]}\big), \quad n = 1, 2, \dots, N, \tag{23}$$

where $h_n = (\xi_n^R - \xi_{n-1}^R)h$. This yields a first-order implicit-explicit (IMEX) predictor for the solution based upon a forward-backward Euler method. Our numerical (and analytical) results indicate the "predictor" step has little bearing on the overall order of accuracy of the solver. For example, it is possible to hold the solution constant for the initial iteration and still obtain high-order accuracy, albeit with one additional iteration.

The classical SISDC method [30] iterates on the provisional solution through

$$\text{(SISDC)}\quad \eta_n^{[p+1]} = \eta_{n-1}^{[p+1]} + h_n\big[f_I\big(\eta_n^{[p+1]}\big) - f_I\big(\eta_n^{[p]}\big)\big] + h_n\big[f_E\big(\eta_{n-1}^{[p+1]}\big) - f_E\big(\eta_{n-1}^{[p]}\big)\big] + h\sum_{m=1}^{M} w_{n,m} f\big(\eta_m^{[p]}\big), \tag{24}$$

where $\eta_0^{[p+1]} = \eta_0$ is a known quantity. This value is typically taken to be the result from the previous time step, and we assume that it is known to high-order accuracy. Our focus is on the local truncation error, to which end we assume that the error at time zero is nonzero. That is, we assume $e_0 = \eta_0 - y_0 \neq 0$. Once the single step error is established, a global error can be directly found using textbook techniques. In the event where $f_I \equiv 0$, we end up with the explicit SDC method defined in (9), and when $f_E \equiv 0$, we end up the implicit SDC defined in (10).

We repeat that the collocation method defined in (6) requires simultaneously solving for each η_n and is clearly more expensive than multiple applications of the backward Euler method found in (24), either on part of or the entire right-hand side. In the event where $f_I \equiv 0$, then the method should be less expensive to run for a single time step, but the regions of absolute stability suffer [28; 29]. We also repeat that, provided $\boldsymbol{\eta}^{[p]}$ converges as $p \to \infty$, then (24) defines a solution to (6). Proving which initial guesses converge to the fully implicit solver is beyond the scope of this work. Currently, our aim is to show that each correction step in the SDC framework picks up at least a single order of accuracy to the order predetermined by the quadrature rule.

3A. *Statement of the main result.*

Theorem 3.1. *The errors for a single step of the semi-implicit SDC method satisfy*

$$\big|e_n^{[p+1]}\big| \le e^{Nh(2L_I+L_E)}|e_0| + C_1 h\big\|\boldsymbol{e}^{[p]}\big\| + C_2 h^{M+1}, \tag{25}$$

provided $hL_I < \frac{1}{2}$, where N is the number of intervals under consideration, $L := L_I + L_E$ is the Lipschitz constant of f, and

$$C_1 = 2Ne^{Nh(2L_I+L_E)}W \quad \textit{and} \quad C_2 = 2Ne^{Nh(2L_I+L_E)}\frac{F}{M!}$$

are constants that depend only on f, the exact solution y, and the selection of quadrature points.

In Section 3C we point out two corollaries to this result, one for implicit and one for explicit SDC, but before proving this theorem, we stop to point out an important observation that is applicable to any of the aforementioned methods.

Remark. The statement of this theorem highlights that there are a total of three sources of error that SDC methods admit, which are ordered by appearance in the right-hand side of (25):

(1) the error at the current time step $e_0 = \eta_0 - y_0$,

(2) the error from the previous iterate (or predictor) $\boldsymbol{e}^{[p]} = \boldsymbol{\eta}^{[p]} - \boldsymbol{y}$, and

(3) the number of quadrature points M.

The most important takeaway is that *because the error from the previous iterate, $\|\boldsymbol{e}^{[p]}\|$, gets multiplied by a factor of h, the error gets improved by one order of accuracy with each correction.* Of course this order reaches a maximum order based upon the number of the quadrature points chosen, which can be seen in the third source of error. This can be improved by selecting quadrature points with superconvergence properties such as the Gaussian or Gauss–Lobatto quadrature points. Finally, please note that we make no comment about how the "previous" function values were found. This is intentional because we would like to focus our attention on the impact of what a single correction does to the solution. Doing so permits the analysis to apply to parallel implementations of SDC methods where synchronizations between different correctors (threads) are seldom seen [9; 13].

3B. *Proof of the main result.*

Proof. We subtract the exact equation (7) from (24) and find that the discrete error evolution equation is

$$e_n^{[p+1]} = e_{n-1}^{[p+1]} + h_n\big[f_I\big(\eta_n^{[p+1]}\big) - f_I\big(\eta_n^{[p]}\big)\big] + h_n\big[f_E\big(\eta_{n-1}^{[p+1]}\big) - f_E\big(\eta_{n-1}^{[p]}\big)\big] \\ + \int_{t_{n-1}}^{t_n} I\big[f\big(\boldsymbol{\eta}^{[p]}\big)\big](t) - f(y(t))\,dt. \quad (26)$$

The last term in this summand can be estimated by appealing to Lemma 2.1 and observing

$$|I_n| := \left|\int_{t_{n-1}}^{t_n} I\big[f\big(\boldsymbol{\eta}^{[p]}\big)\big](t) - f(y(t))\,dt\right| \le h\big\|\boldsymbol{e}^{[p]}\big\| W_n L + \frac{F}{M!}h^{M+1}. \quad (27)$$

We estimate the other terms by making use of their respective Lipshitz constants:

$$\big|e_n^{[p+1]}\big| \le \big|e_{n-1}^{[p+1]}\big| + h_n\big|f_I\big(\eta_n^{[p+1]}\big) - f_I\big(\eta_n^{[p]}\big)\big| + h_n\big|f_E\big(\eta_{n-1}^{[p+1]}\big) - f_E\big(\eta_{n-1}^{[p]}\big)\big| + |I_n| \\ \le \big|e_{n-1}^{[p+1]}\big| + hL_I\big(\big|e_n^{[p+1]}\big| + \big|e_n^{[p]}\big|\big) + hL_E\big(\big|e_{n-1}^{[p+1]}\big| + \big|e_{n-1}^{[p]}\big|\big) + |I_n|. \quad (28)$$

The second line follows from the first by adding and subtracting $f_I(y_n)$ and $f_E(y_{n-1})$ to the inside of each of the absolute values containing $\big|f\big(\eta_n^{[p+1]}\big) - f\big(\eta_n^{[p]}\big)\big|$ and $\big|f\big(\eta_{n-1}^{[p+1]}\big) - f\big(\eta_{n-1}^{[p]}\big)\big|$, respectively. Note that we also make use of the fact that $h_n \le h$, although this too can be relaxed.

We continue by subtracting $hL_I\big|e_n^{[p+1]}\big|$ from both sides, dividing by $1-hL_I > 0$, recognizing that $h_n < h$, and collecting the remaining terms involving the "explicit" portions:

$$\begin{aligned}\left|e_n^{[p+1]}\right| &\le \frac{1}{1-hL_I}\left[(1+hL_E)\left|e_{n-1}^{[p+1]}\right| + hL_I\left|e_n^{[p]}\right| + hL_E\left|e_{n-1}^{[p]}\right| + |I_n|\right]\\ &\le \frac{1}{1-hL_I}\left[(1+hL_E)\left|e_{n-1}^{[p+1]}\right| + hL\left\|\boldsymbol{e}^{[p]}\right\| + |I_n|\right]\\ &\le \frac{1}{1-hL_I}\left[(1+hL_E)\left|e_{n-1}^{[p+1]}\right| + hL(1+W_n)\left\|\boldsymbol{e}^{[p]}\right\| + \frac{F}{M!}h^{M+1}\right]\\ &\le \frac{1+hL_E}{1-hL_I}\left|e_{n-1}^{[p+1]}\right| + \frac{1}{1-hL_I}\left[hW\left\|e^{[p]}\right\| + \frac{F}{M!}h^{M+1}\right], \qquad (29)\end{aligned}$$

where we define $W := \max_{1\le n\le N}(1+W_nL)$.

We make use of two separate estimates for $1/(1-hL_I)$ to estimate the two terms found in the right-hand side of (29). For the first term, we expand the geometric series and keep the first two terms:

$$\frac{1}{1-hL_I} = 1 + (hL_I) + (hL_I)^2 + \cdots = 1 + (hL_I) + (hL_I)^2\frac{1}{1-hL_I}. \qquad (30)$$

This is valid because $hL_I < 1$. Additionally, $hL_I < \frac{1}{2}$, and therefore,

$$hL_I < 1 - hL_I \quad \Longrightarrow \quad \frac{(hL)^2}{1-hL_I} < hL_I. \qquad (31)$$

Together, these estimates imply that the first term can be estimated with

$$\frac{1}{1-hL_I} \le 1 + 2hL_I \le e^{2hL_I}. \qquad (32)$$

For the second term, we have $1/(1-hL_I) \le 2$ for all $hL_I \in \left[0, \frac{1}{2}\right]$. This leads us to observe that

$$\left|e_n^{[p+1]}\right| \le e^{2hL_I}(1+hL_E)\left|e_{n-1}^{[p+1]}\right| + 2\left(hW\left\|\boldsymbol{e}^{[p]}\right\| + \frac{F}{M!}h^{M+1}\right). \qquad (33)$$

Next, we appeal to Corollary 2.3 and make use of $A = e^{2hL_I}(1+hL_E) > 1$ and $B = 2\left(hW\left\|\boldsymbol{e}^{[p]}\right\| + (F/M!)h^{M+1}\right)$ to conclude that

$$\begin{aligned}\left|e_n^{[p+1]}\right| &\le e^{2hnL_I}(1+hL_E)^n|e_0|\\ &\qquad + ne^{2h(n-1)L_I}(1+hL_E)^{n-1}2\left(hW\left\|\boldsymbol{e}^{[p]}\right\| + \frac{F}{M!}h^{M+1}\right). \qquad (34)\end{aligned}$$

Since $1 + hL_E \le e^{hL_E}$ and $n \le N$, we have the desired result. $\square$

3C. ***Corollaries of main result: implicit and explicit error estimates.*** With the general case proven in Theorem 3.1, we find results for both implicit as well as explicit SDC solvers. An immediate corollary to Theorem 3.1 can be found by setting $f_E \equiv 0$, in which case L_I becomes the Lipshitz constant for f, and the SISDC solver reduces to classical SDC with backward Euler defined in (10).

Corollary 3.2. *The errors for a single step of the implicit SDC method defined in (10) satisfy*

$$\left|e_n^{[p+1]}\right| \le e^{2NhL}|e_0| + C_1 h \left\| \boldsymbol{e}^{[p]} \right\| + C_2 h^{M+1} \tag{35}$$

provided $h < 1/(2L)$. The constants C_1 and C_2 depend only on the smoothness of f, the exact solution y, and the choice of quadrature points.

It is worth noting that the error estimate provided here is an *asymptotic* error estimate. That is, one key assumption that we have to make is that $h < 1/(2L)$, which we do not have to make for the explicit case. Unfortunately, one key benefit of implicit solvers is that large time steps can be taken, in which case it is certainly possible that the solver does not obey this assumption. For these cases, a rigorous error estimate and analysis when $h > 1/(2L)$ would make for an interesting result, which would be especially important for multiscale problems that contain large time scale separations. This observation is beyond the scope of the present work.

A related corollary for explicit solvers with tighter error bounds can be found. The result is the following:

Corollary 3.3. *The errors for a single step of the explicit SDC method defined in (9) satisfy*

$$\left|e_n^{[p+1]}\right| \le e^{NhL}|e_0| + C_1 h \left\| \boldsymbol{e}^{[p]} \right\| + C_2 h^{M+1}, \tag{36}$$

where N is the number of intervals under consideration, L is the Lipschitz constant of f for the ODE $y' = f(y)$, and C_1 and C_2 are constants that depend only on f, the exact solution y, and the selection of quadrature points.

Proof. Revisit the proof of Theorem 3.1, and replace the error estimate for $1/(1 - hL_I) \le 2$ with 1 instead of 2. □

4. Convergence proofs for higher-order base solvers

We now consider the spectral deferred correction method with the implicit trapezoidal rule as its base solver:

$$\eta_n^{[p+1]} = \eta_{n-1}^{[p+1]} + \frac{h_n}{2}\left[f\left(\eta_n^{[p+1]}\right) + f\left(\eta_{n-1}^{[p+1]}\right) - f\left(\eta_n^{[p]}\right) - f\left(\eta_{n-1}^{[p]}\right)\right] + h \sum_{m=1}^{M} w_{n,m} f\left(\eta_m^{[p]}\right). \tag{37}$$

What makes this method interesting is that it picks up a total of two orders of accuracy with each correction. Note again that, in the absence of the terms that the factor $h_n/2$ multiplies, this method reduces to explicit Picard iteration, which picks up a single additional order of accuracy with each correction.

The result we focus on is the impact of each correction step, in which case any provisional solution may be used. In order to retain large regions of absolute stability reasonable methods include low-order implicit solvers such as backward Euler, or the second-order implicit trapezoidal (Crank–Nicholson) rule

$$\text{(trapezoidal rule)}\quad \eta_n^{[0]} = \eta_{n-1}^{[0]} + \frac{h_n}{2}\left(f\left(\eta_{n-1}^{[0]}\right) + f\left(\eta_n^{[0]}\right)\right), \quad n = 1, 2, \dots, N. \tag{38}$$

In this section, we examine the interplay between the integral over the entire time interval, and the addition of extra integral terms that allows this solver to pick up additional orders of accuracy.

Let us define the exact value of the right-hand-side function as $f_n = f(y(t_n))$, the approximate value of the right-hand-side function as $f_n^{[p]} = f\left(\eta_n^{[p]}\right)$, and the local and global quadrature rules for integration over the subinterval $[t_{n-1}, t_n]$ as

$$T_n = \frac{h_n}{2}[f_{n-1} + f_n], \qquad T_n^{[p]} = \frac{h_n}{2}\left[f_{n-1}^{[p]} + f_n^{[p]}\right],$$

$$H_n = h\sum_{m=1}^{M} \omega_{n,m} f_m, \qquad H_n^{[p]} = h\sum_{m=1}^{M} \omega_{n,m} f_m^{[p]}.$$

These definitions allow us to compactly write the SDC method with the trapezoidal rule defined in (37) to read

$$\eta_n^{[p+1]} = \eta_{n-1}^{[p+1]} + \left(T_n^{[p+1]} - T_n^{[p]}\right) + H_n^{[p]}. \tag{39}$$

Recall that the exact solution satisfies the integral (7), which we repeat:

$$y_n = y_{n-1} + \int_{t_{n-1}}^{t_n} f(y(t))\, dt.$$

Theorem 4.1. *When coupled with the implicit trapezoidal rule, the errors for a single step of the spectral deferred correction method satisfy*

$$\left|e_n^{[p+1]}\right| \le e^{2NhL}|e_0| + 2Ne^{2(N-1)hL}\left(\frac{1}{12}\left\|\frac{d^2E^{[p]}}{dt^2}(\,\cdot\,)\right\|h^3 + \frac{F}{M!}h^{M+1}\right), \tag{40}$$

provided $hL < 1$, where N is the number of intervals under consideration, M is the number of points involved, L is the Lipschitz constant of f, F is an upper bound for the M-th derivative of f, and the function $E^{[p]}(t)$ is the polynomial interpolant for the error in the approximation of the right-hand-side function during the p-th *iterate defined by*

$$E^{[p]}(t) := I\big[f\big(\boldsymbol{\eta}^{[p]}\big)\big](t) - I[f(\boldsymbol{y})](t) = \sum_{m=1}^{M} \Delta f_m^{[p]} \ell_m(t/h), \tag{41}$$

where $\Delta f_m^{[p]} := f_m^{[p]} - f_m$ *for each* $m = 1, 2, \ldots, M$. *The norm defined in (40) is the maximum absolute value of the second derivative of* $E^{[p]}$:

$$\left\| \frac{d^2 E^{[p]}}{dt^2}(\,\cdot\,) \right\| := \max_{t\in[0,h]} \big|\big(E^{[p]}\big)''(t)\big|. \tag{42}$$

Proof. We subtract the exact solution defined in (7) from the SDC method based upon the trapezoidal rule defined in (39) to end up with

$$\begin{aligned} e_n^{[p+1]} &= e_{n-1}^{[p+1]} + \big(T_n^{[p+1]} - T_n^{[p]}\big) + H_n^{[p]} - \int_{t_{n-1}}^{t_n} f(y(t))\,dt \\ &= e_{n-1}^{[p+1]} + T_n^{[p+1]} - T_n + T_n - T_n^{[p]} + H_n^{[p]} - H_n + H_n - \int_{t_{n-1}}^{t_n} f(y(t))\,dt \\ &= e_{n-1}^{[p+1]} + \underbrace{\big(T_n^{[p+1]} - T_n\big)}_{\text{I}} + \underbrace{\big(H_n^{[p]} - T_n^{[p]} + T_n - H_n\big)}_{\text{II}} + \underbrace{I_n}_{\text{III}}, \end{aligned} \tag{43}$$

where $I_n := H_n - \int_{t_{n-1}}^{t_n} f(y(t))\,dt$ is the difference between the high-order (discrete) integral and the exact integral of the right-hand side.

We now estimate each of the three terms to the right of $e_{n-1}^{[p+1]}$ in (43) separately, starting with the first term:

$$|\text{I}| = \frac{h_n}{2}\big|f_{n-1}^{[p+1]} + f_n^{[p+1]} - f_{n-1} - f_n\big| \le \frac{Lh_n}{2}\big(\big|e_n^{[p+1]}\big| + \big|e_{n-1}^{[p+1]}\big|\big), \tag{44}$$

which follows from the Lipshitz continuity of f. The third term can be estimated by first recognizing that

$$H_n := h\sum_{m=1}^{M} \omega_{n,m} f_m = \int_{t_{n-1}}^{t_n} I[f(\boldsymbol{y})](t)\,dt,$$

and then using (18) (which requires assuming that $f \circ y \in C^M$) in order to yield

$$|\text{III}| = \left| H_n - \int_{t_{n-1}}^{t_n} f(y(t))\,dt \right| = |H_n - I_n| \le \frac{F}{M!} h^{M+1}, \tag{45}$$

where F is any number that satisfies $\left\| \frac{d^M}{dt^M}(f \circ y) \right\|_\infty \le F$.

Finally, we address the second, and most interesting, term on the right-hand side of (43). The key observation comes from recognizing this term as the difference between a low-order (local) quadrature, T_n, and a high-order (global) quadrature H_n.

Note that the exact integral of the polynomial interpolant $E^{[p]}$ over the subinterval $[t_{n-1}, t_n]$ is

$$\int_{t_{n-1}}^{t_n} E^{[p]}(t)\,dt = H_n^{[p]} - H_n, \tag{46}$$

and that

$$T_n^{[p]} - T_n = \frac{h_n}{2}\big(E^{[p]}(t_n) + E^{[p]}(t_{n-1})\big) \tag{47}$$

is a low-order approximation to this integral. By textbook results, we have

$$\mathrm{II} = \big(H_n^{[p]} - H_n - T_n^{[p]} + T_n\big) = -\frac{h_n^3}{12}\frac{d^2 E^{[p]}}{dt^2}(\xi_n), \tag{48}$$

where ξ_n is some number between t_{n-1} and t_n. Together, this implies

$$|\mathrm{II}| \le \frac{h_n^3}{12}\left\|\frac{d^2 E^{[p]}}{dt^2}(\,\cdot\,)\right\|. \tag{49}$$

All together, inserting (44), (49), and (45) into (43), we have

$$\begin{aligned}\big|e_n^{[p+1]}\big| &\le \big|e_{n-1}^{[p+1]}\big| + |\mathrm{I}| + |\mathrm{II}| + |\mathrm{III}| \\ &\le \left(1 + \frac{Lh_n}{2}\right)e_{n-1}^{[p+1]} + \frac{Lh_n}{2}e_n^{[p+1]} + \frac{h_n^3}{12}\left\|\frac{d^2E^{[p]}}{dt^2}(\,\cdot\,)\right\| + \frac{F}{M!}h^{M+1}.\end{aligned} \tag{50}$$

After replacing each $h_n \le h$, rearranging, and assuming that $hL/2 < 1$, we have

$$\big|e_n^{[p+1]}\big| \le \underbrace{\frac{1+hL/2}{1-hL/2}}_{\le e^{2hL}} e_{n-1}^{[p+1]} + \underbrace{\frac{1}{1-hL/2}}_{\le 2}\left(\frac{h^3}{12}\left\|\frac{d^2E^{[p]}}{dt^2}(\cdot)\right\| + \frac{F}{M!}h^{M+1}\right). \tag{51}$$

We now verify the two underscored inequalities involving the $1 \pm hL/2$ terms in (51). Identical to (32), we have

$$\frac{1}{1-hL/2} \le 1 + 2(hL/2) = 1 + hL,$$

after expanding the rational expression in terms of a geometric series, and assuming that $hL/2 < 1$ in order to retain convergence. Because $1 + hL/2 \le 1 + hL$, we have

$$\frac{1+hL/2}{1-hL/2} \le (1+hL)^2 \le e^{2hL},$$

which verifies the first of the two underscored inequalities. For the second one, we need only assume that $hL < 1$, which yields $1/(1 - hL/2) < 2$.

All together, we have

$$\big|e_n^{[p+1]}\big| \le e^{2hL}e_{n-1}^{[p+1]} + 2\left(\frac{h^3}{12}\left\|\frac{d^2E^{[p]}}{dt^2}(\,\cdot\,)\right\| + \frac{F}{M!}h^{M+1}\right), \tag{52}$$

which yields the desired result after appealing to Corollary 2.3 and using $n \le N$. □

Remark. The same decomposition for the error can be used for SDC coupled with forward (or backward) Euler. That is, identical to the decomposition found in (43), we can decompose the error as

$$\begin{aligned}
e_n^{[p+1]} &= e_{n-1}^{[p+1]} + L_n^{[p+1]} - L_n^{[p]} + H_n^{[p]} - \int_{t_{n-1}}^{t_n} f(y(t))\,dt \\
&= e_{n-1}^{[p+1]} + L_n^{[p+1]} - L_n + L_n - L_n^{[p]} + H_n^{[p]} - H_n + H_n - \int_{t_{n-1}}^{t_n} f(y(t))\,dt \\
&= e_{n-1}^{[p+1]} + \underbrace{\big(L_n^{[p+1]} - L_n\big)}_{\mathrm{I}} + \underbrace{\big(H_n^{[p]} - L_n^{[p]} + L_n - H_n\big)}_{\mathrm{II}} + \underbrace{I_n}_{\mathrm{III}}, \qquad (53)
\end{aligned}$$

where L_n and $L_n^{[p]}$ denote a "low-order" integral of the right-hand side, but this time we use

$$L_n := h_n f_{n-1}, \qquad L_n^{[p]} := h_n f_{n-1}^{[p]}, \tag{54}$$

for forward Euler, or instead

$$L_n := h_n f_n, \qquad L_n^{[p]} := h_n f_n^{[p]} \tag{55}$$

for backward Euler. The third term III is again $\mathbb{O}(h^{M+1})$, and the first term I can be bounded by a constant times $\big|e_n^{[p]}\big| + \big|e_{n-1}^{[p+1]}\big|$. The lack of additional order pickup can be found by observing that the second source of error instead satisfies

$$\mathrm{II} = \big(H_n^{[p]} - H_n - L_n^{[p]} + L_n\big) = -\frac{h_n^2}{2}\frac{dE^{[p]}}{dt}(\xi_n), \tag{56}$$

where ξ_n is some number between t_{n-1} and t_n. In the following examples, we compare this term to that found from the trapezoidal rule.

4A. ***Examples.*** To illustrate the results of the theorem presented in this section, we consider the linear test case

$$y'(t) = y(t), \quad t > 0, \; y(0) = 1, \tag{57}$$

and examine the errors produced by the SDC method when coupled with a higher-order base solver. The order pickup for the SDC method can be found by examining the size of the second source of error, defined in (48) and (56), given by $h_n^3/12\big(E^{[p]}\big)''(\xi_n)$ for the trapezoidal rule and $h_n^2/2\big(E^{[p]}\big)'(\xi_n)$ for the forward (or backward) Euler method. Note that the form and size of this error is identical for either the forward or backward Euler base solver.

In the following examples, we work out the size of this term for a few different case studies. Taylor expansions are found by making use of the Maple software package.

	trapezoidal rule		forward Euler	
interval	error	$p=0$	error	$p=0$
$[0, \frac{h}{2}]$	$\frac{h}{24}\big(e_1^{[p]} - 2e_2^{[p]} + e_3^{[p]}\big)$	$\mathcal{O}(h^4)$	$\frac{h}{24}\big(7e_1^{[p]} - 8e_2^{[p]} + e_3^{[p]}\big)$	$\mathcal{O}(h^3)$
$[\frac{h}{2}, h]$	$\frac{h}{24}\big(e_1^{[p]} - 2e_2^{[p]} + e_3^{[p]}\big)$	$\mathcal{O}(h^4)$	$\frac{h}{24}\big(e_1^{[p]} + 4e_2^{[p]} - 5e_3^{[p]}\big)$	$\mathcal{O}(h^3)$

Table 2. Correction errors with $M = 3$ uniformly spaced points.

	trapezoidal rule		forward Euler	
interval	error	$p=0$	error	$p=0$
$[0, \frac{h}{3}]$	$\frac{h}{72}\big(3e_1^{[p]}-7e_2^{[p]}+5e_3^{[p]}-e_4^{[p]}\big)$	$\mathcal{O}(h^4)$	$\frac{h}{72}\big(15e_1^{[p]}-19e_2^{[p]}+5e_3^{[p]}-e_4^{[p]}\big)$	$\mathcal{O}(h^3)$
$[\frac{h}{3}, \frac{2h}{3}]$	$\frac{h}{72}\big(e_1^{[p]}-e_2^{[p]}-e_3^{[p]}+e_4^{[p]}\big)$	$\mathcal{O}(h^4)$	$\frac{h}{72}\big(e_1^{[p]}+11e_2^{[p]}-13e_3^{[p]}+e_4^{[p]}\big)$	$\mathcal{O}(h^3)$
$[\frac{2h}{3}, h]$	$\frac{h}{72}\big(e_1^{[p]}-5e_2^{[p]}+7e_3^{[p]}-3e_4^{[p]}\big)$	$\mathcal{O}(h^4)$	$\frac{h}{72}\big(e_1^{[p]}-5e_2^{[p]}-5e_3^{[p]}+9e_4^{[p]}\big)$	$\mathcal{O}(h^3)$

Table 3. Correction errors with $M = 4$ uniformly spaced points.

$M = 3$ *uniformly spaced points.* We first consider the results of using a total of $M = 3$ equispaced quadrature points and look at the local error over the subinterval $[t_{n-1}, t_n]$ produced by the second term II defined in (48) and (56). In the second set of columns in Table 2 look at the size of this term by writing out Taylor expansions for the first $p = 0$ SDC iteration of a solver constructed by taking a forward Euler provisional solution for the first time step. Note that in this case, each point satisfies $e_n^{[0]} = \mathcal{O}(h^2)$ for each n, because the local truncation error (LTE) for Euler's method is second-order accurate. Therefore, the jump from $\mathcal{O}(h^2)$ to $\mathcal{O}(h^4)$ indicates that the trapezoidal correction picks up an additional two orders of accuracy, whereas the jump from $\mathcal{O}(h^2)$ to $\mathcal{O}(h^3)$ picks up a single additional order of accuracy for the Euler base solver, which is consistent with the theory.

$M = 4$ *uniformly spaced points.* We next consider the same problem with a total of $M = 4$ equispaced quadrature points. Again, we look at the errors for the trapezoidal method compared to the forward Euler method after first constructing a provisional solution with the forward Euler method. We again observe that the trapezoidal rule improves the order of accuracy of the provisional solution by two factors, whereas forward Euler (or likewise backward Euler) only improves the order by one. Results for these quadrature points are presented in Table 3.

$M = 4$ *nonequispaced spaced points.* Finally, we consider a case with a total of $M = 4$ nonequispaced points. As an illustrative example, we consider the quadrature points $\xi_1 = 0$, $\xi_2 = \frac{1}{3}$, $\xi_3 = \frac{1}{2}$, and $\xi_4 = 1$. We find that the trapezoidal error only increases the order of the solver by one degree, which is consistent with the findings in [6], where the authors show that when the second-order Runge–Kutta

	trapezoidal rule	
interval	error	$p=0$
$[0, \frac{h}{3}]$	$\frac{h}{162}(8e_1^{[p]} - 27e_2^{[p]} + 20e_3^{[p]} - e_4^{[p]})$	$\mathcal{O}(h^3)$
$[\frac{h}{3}, \frac{h}{2}]$	$\frac{h}{5184}(14e_1^{[p]} - 27e_2^{[p]} + 8e_3^{[p]} + 5e_4^{[p]})$	$\mathcal{O}(h^3)$
$[\frac{h}{2}, h]$	$\frac{h}{192}(10e_1^{[p]} - 81e_2^{[p]} + 88e_3^{[p]} - 17e_4^{[p]})$	$\mathcal{O}(h^3)$

	forward Euler	
interval	error	$p=0$
$[0, \frac{h}{3}]$	$\frac{h}{162}(35e_1^{[p]} - 54e_2^{[p]} + 20e_3^{[p]} - e_4^{[p]})$	$\mathcal{O}(h^3)$
$[\frac{h}{3}, \frac{h}{2}]$	$\frac{h}{5184}(14e_1^{[p]} + 405e_2^{[p]} - 424e_3^{[p]} + 5e_4^{[p]})$	$\mathcal{O}(h^3)$
$[\frac{h}{2}, h]$	$\frac{h}{192}(10e_1^{[p]} - 81e_2^{[p]} + 40e_3^{[p]} + 31e_4^{[p]})$	$\mathcal{O}(h^3)$

Table 4. Correction errors with $M = 4$ nonequispaced points. In this case, we find that both methods only pick up a single additional order of accuracy.

method is used as a corrector, then the solver does not always pick up two orders of accuracy with each correction loop. Results for this problem are presented in Table 4.

5. Numerical results

The primary contribution of this work is to construct rigorous error estimates for classical SDC methods, and therefore, we only include a couple of numerical results. An abundance of SDC examples applied to ordinary and partial differential equations can be found in the literature. One of our key goals here is to promulgate the fact that the primary source of high-order accuracy inherent in all SDC methods comes from its underlying Picard integral formulation, and not necessarily the "base solver", and therefore, we focus our results on nearby variations of classical SDC methods and demonstrate how classical SDC methods can be extended to produce related high-order solvers.

First we introduce a comparison of errors (and stability regions) for explicit SDC versus Picard iteration, second we explore modifications of the constant in front of an implicit SDC method, and finally we compare semi-implicit SDC and modified semi-implicit SDC solvers. For the sake of brevity the proposed modifications to SDC methods are not formally analyzed but straightforward extensions of the theorems presented in this work can be constructed to present formal error bounds for these methods. The numerical evidence presented here supports this claim.

5A. ***A comparison of explicit SDC and Picard iteration.*** In this numerical example, we compare the errors and stability regions by applying the Picard iterative

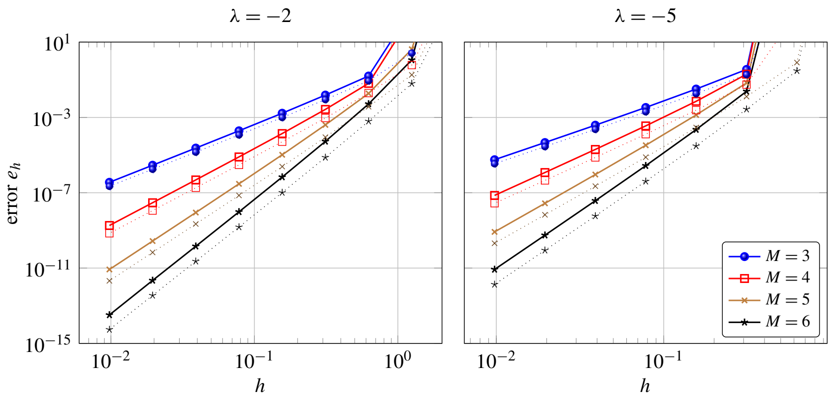

Figure 1. Linear test case. Here, we compare explicit SDC (solid lines) with Picard iteration (dashed lines) of various orders. Each method attains the desired order of accuracy with the minimum number of corrections. In each case, Picard iteration has slightly smaller errors when compared to the equivalent explicit SDC method of the same order and same quadrature rule.

method defined in (8) to that of the explicit SDC defined in (9). In order to present an equal comparison of these two solvers, we consider identical initial guesses, or provisional solutions $\boldsymbol{\eta}^{[0]}$ based upon forward Euler time stepping and we work with uniform quadrature points for all of our test cases.

Errors for a linear test case. In Figure 1, we compare errors for the linear equation

$$y' = \lambda y, \qquad y(0) = 1, \tag{58}$$

at a final time of $T = 10$ with $\lambda = -2$ and $\lambda = -5$. Other orders and values of λ show similar results where we observe slightly smaller error constants when using the Picard iterative method compared to the equivalent SDC method. This is consistent with the findings of Corollary 3.3, because the first error estimate

$$\left|e_n^{[p+1]}\right| \le \left|e_{n-1}^{[p+1]}\right| + h_n \left| f\left(\eta_{n-1}^{[p+1]}\right) - f\left(\eta_{n-1}^{[p]}\right)\right| + |I_n|$$

could be tightened up to read

$$\left|e_n^{[p+1]}\right| \le \left|e_{n-1}^{[p+1]}\right| + |I_n|,$$

which produces a smaller (provable) overall error for the Picard method when compared to the SDC method.

A comparison of regions of absolute stability for explicit methods. Next, we seek to compare regions of absolute stability for explicit SDC methods and their Picard iterative cousins. Here we observe that the stability regions are slightly improved when the “Euler term” in the time stepping is dropped from the SDC method. That

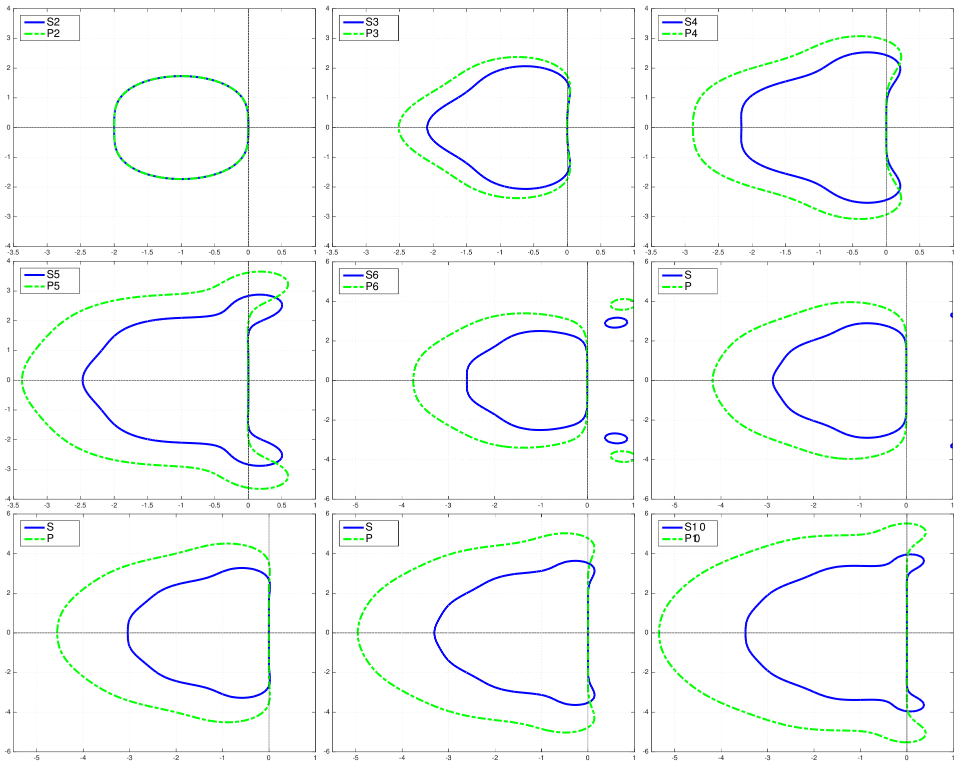

Figure 2. Stability regions for explicit methods. Here, we compare SDC methods to that of Picard iterative methods where the explicit "Euler term" is dropped from the iterative process. In each case save one, we observe that the Picard iterative methods have slightly larger regions of absolute stability. The second-order SDC and Picard methods that use two quadrature points are identical because the forward Euler time step vanishes in the SDC method. The scaling on the axes for the methods of orders 6 through 10 is different than the scaling for the methods of orders two through five.

is to say, we find that the Picard iterative methods generally have larger regions of absolute stability when compared to their SDC counterparts.

In order to demonstrate this, in Figure 2, we include a comparison of plots of the regions of absolute stability, defined by

$$\mathbb{D} := \{z \in \mathbb{C} : |\rho(z)| < 1\} \tag{59}$$

where $\rho(z)$ is the amplification factor for various quadrature rules for both of these methods, $z := \lambda h$, and λ is defined as in (58). There, we compare methods of orders two through ten, all based on equispaced quadrature points, forward Euler time stepping for the provisional solution, and the minimum number of corrections required to reach the desired order of accuracy. (For example, the third-order method uses two corrections and the fifth-order method uses four corrections.) Similar to most explicit Runge–Kutta methods, we find that the regions of absolute stability increase as the order is increased, but there are also more function evaluations per time step.

5B. ***Implicit SDC methods with modified backward Euler time steps.*** Here, we consider implicit SDC methods with a variable constant in front of the vanishing term:

$$\eta_n^{[p+1]} = \eta_{n-1}^{[p+1]} + \theta h\big[f\big(\eta_n^{[p+1]}\big) - f\big(\eta_n^{[p]}\big)\big] + h\sum_{m=1}^{M} w_{n,m} f\big(\eta_m^{[p]}\big), \qquad n = 1, 2, \dots, N. \tag{60}$$

This same scaling has already been explored in [39] for SDC methods, but there the authors only consider the case where $\frac{1}{2} \le \theta \le 1$. With $\theta = 0$, we have (explicit) Picard iteration (provided the provisional solution is modified), with $\theta = 1$, we have the classical implicit SDC method, and with negative values of θ we have backward Euler solves on negative time steps; none of the these changes affect the overall order of accuracy, only the size of the error constant and the regions of absolute stability. Following the proof of the main theorem in this work, we find the following result under a modified estimate for the time step size.

Theorem 5.1. *The errors for a single step of the modified implicit SDC method defined in (60) satisfy*

$$\big|e_n^{[p+1]}\big| \le e^{2N|\theta|hL}|e_0| + C_1 h \big\|\boldsymbol{e}^{[p]}\big\| + C_2 h^{M+1} \tag{61}$$

provided $|\theta|h < 1/(2L)$. *The constants*

$$C_1 = 2Ne^{2N|\theta|hL}\Big(|\theta| + L\max_n W_n\Big) \quad \textit{and} \quad C_2 = 2Ne^{2N|\theta|hL}\frac{F}{M!}$$

again depend only on the smoothness of f, *the exact solution* y, *and the choice of quadrature points, but this time they also depend on* θ.

Note that the value of $\theta = 0$ minimizes the size of these constants, but it also produces poor regions of absolute stability. With $\theta \gg 1$ we have a method that is heavy handed on multiple backward Euler solves, and therefore, it has a very large region of absolute stability; however, these methods unfortunately introduce larger error constants. Small values of θ decrease these error constants, but they modify the regions of absolute stability to the point where they become finite and therefore undesirable as these are implicit methods.

A verification of high-order accuracy: the nonlinear pendulum problem. As a verification of the high-order accuracy of the solvers, we consider the equations of motion for a nonlinear pendulum:

$$x''(t) + \sin x(t) = 0 \tag{62}$$

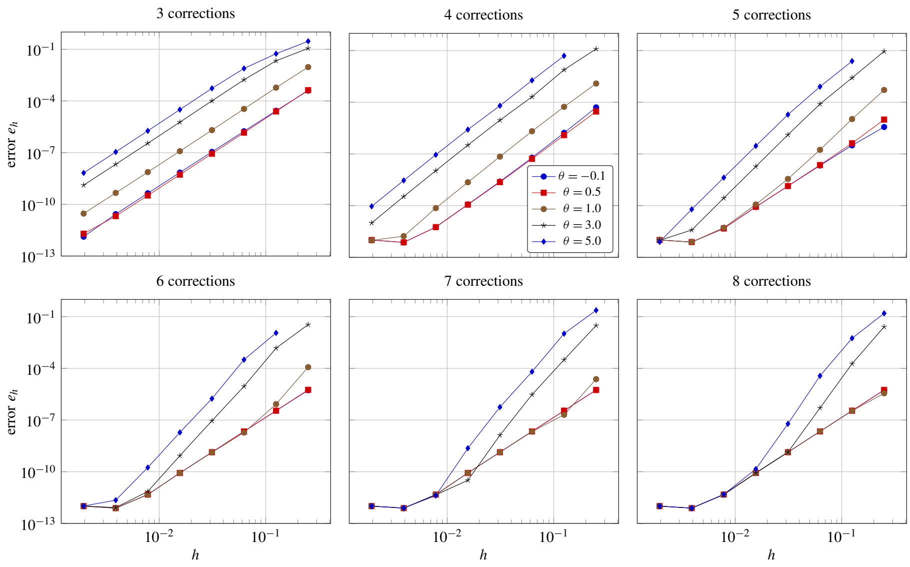

Figure 3. Nonlinear pendulum problem. Here, we compare SDC methods with different scalings on the backward Euler term as defined in (60). The case with $\theta = 1$ is classical implicit SDC. We observe the expected result that all methods have high-order accuracy and that $\theta > 1$ produces larger error constants. The case with $\theta = -0.1$ is not a useful method because it has a finite region of absolute stability. While all methods here are fourth-order accurate after three corrections, the methods with large values of θ stand to gain the most through additional corrections. This can be attributed to the large error constants found in the backward Euler term that vanish at the number of iterations increase (provided the iterates converge).

with appropriate initial conditions. If we perform the change of variables $y_1(t) = x(t)$ and $y_2(t) = x'(t)$, we end up with the following first-order nonlinear system of equations that is equivalent to (62):

$$(y_1, y_2)' = (y_2, -\sin y_1). \tag{63}$$

We consider initial conditions defined by $(y_1(0), y_2(0)) = (0, 1)$, and we integrate this problem to a final time of $T = 10$. To compute a reference solution, we use MATLAB's built-in ode45 with a relative tolerance of 10^{-12} and an absolute tolerance of 10^{-14}.

We present convergence results for this problem in Figure 3. These results indicate that each method is indeed high-order independent of the value of θ. For the sake of brevity, we only report results for methods with a total of $M = 4$ equispaced quadrature points, but we also compare results for different number of corrections. (This underlying quadrature rule is also known as Simpson's $\frac{3}{8}$ rule, which has a smaller error constant than Simpson's rule that uses $M = 3$ equispaced points, but it comes at the cost of an additional function evaluation.) We not only

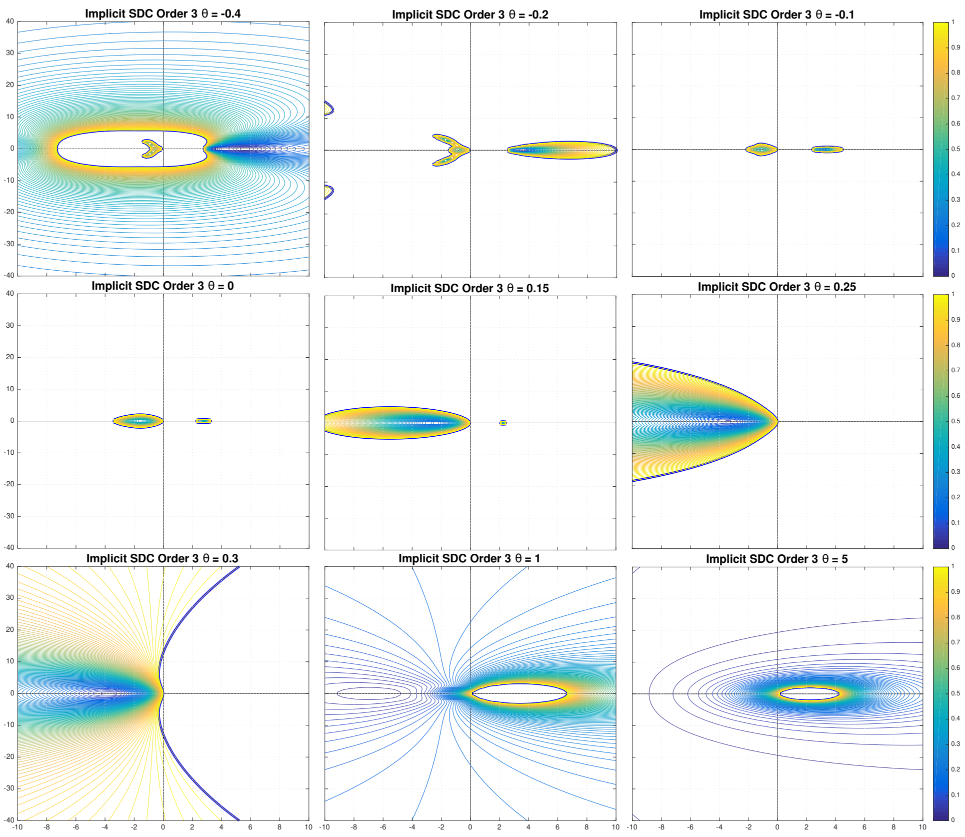

Figure 4. Stability regions for modified implicit methods. Here, we perform a parameter study to the modified implicit SDC method introduced in (60); the case with $\theta = 0$ is explicit for each of the correction steps because there is no backward Euler time step to be solved for; however, the stability region is different than that of the same-order Picard method because the initial guess (provisional solution) is computed using backward Euler time steps, whereas the Picard method makes use of forward Euler steps for its initial guess. Somewhere in the interval $\theta \in [0, 1]$ there is a transition between finite and infinite regions of absolute stability. Infinite and large regions are typically desirable when performing implicit solves. Large values of θ increase the regions of absolute stability but at the cost of stiff inversions and larger error constants.

find that smaller values of θ produce smaller errors, which the theory supports, but we also demonstrate that more corrections for the methods with large θ values can help to decrease the errors.

A parameter study of regions of absolute stability for implicit methods. Given the results of the previous section, it would be tempting to set $\theta = 0$ in order to reduce the total error. What is missing from this observation is an understanding of the regions of absolute stability. We now address this question. We find that small values of θ produce finite regions of absolute stability, and that large values of θ

increase the regions of absolute stability (when compared to classical SDC methods) but they also increase the stiffness of each implicit solve. With that being said, larger time steps should be able to be taken, but as is pointed out in the previous section, larger errors are introduced. This reproduces the usual tradeoff between being able to take large time steps with large errors or being forced to take smaller time steps but at an increased computational cost.

In Figure 4 we present results for a third-order method with various values of θ. There we plot contour plots of the modulus of the amplification factor $|\rho(z)|$ in place of the boundary defined by $|\rho(z)| = 1$ because if we were to plot the boundary, then it would not be clear what parts are stable. In this sequence of images, we present results for various values of $\theta \in [-0.4, 5]$. Larger values of θ such as $\theta = 100$ look very similar to $\theta = 5$. Tests on methods of other orders produce similar results involving transitions between finite and infinite regions of absolute stability as θ increases from 0. The tradeoff between the size and shape of the stability regions for various quadrature rules is left for future work.

Before continuing, we point out that diagonally implicit Runge–Kutta methods (on nonequispaced points) can be constructed from this very same framework. The point here is that because the term involving the difference $f\left(\eta_n^{[p+1]}\right) - f\left(\eta_n^{[p]}\right)$ in (60) does not contribute to the overall order of accuracy, the scaling in front of this term can be modified so that each implicit solve uses the exact same time step, which would result in a singly diagonally implicit Runge–Kutta (SDIRK) method. This could be advantageous for easing the implementation of SDC methods in large-scale code bases. In such a case the provisional solution would have to be modified in order to retain constant time steps for each stage in the solver. This would mean an extra correction or a (low-order) polynomial interpolation step would be necessary to not lose the starting accuracy found in the provisional solution, which would again modify the regions of absolute stability for solvers of various orders.

Similar modifications have recently been explored on nonequispaced points from a linear algebra perspective. In [38], the author makes use of this vantage and optimizes their solvers by modifying the coefficients in the fixed-point iteration matrices. It is pointed out that there are a number of items that could be optimized, such as the spectral radius of the solver (in order to optimize the convergence rate of the sweeps), the matrix norm (for the purpose of reducing the error under the assumption of a small number of sweeps), the error at the final time, the average reduction factor in each sweep block (for the purpose of adaptively choosing the number of sweeps, which could include flexible or greedy sweeps), and so on. Even though SDC methods are a subset of Runge–Kutta methods, and all of these options can be found by looking at this more general class of methods, one key advantage SDC methods enjoy is they do not typically sacrifice the difficult order conditions that a more generic RK method would have to address. At the same time, SDC methods still have ample levers to tune for optimization purposes.

5C. ***Modified semi-implicit SDC methods.*** Recall the semi-implicit SDC (SISDC) method begins with a partition of the right-hand side into two functions f_I and f_E via

$$y' = f(y), \quad f(y) = f_I(y) + f_E(y), \; y(0) = y_0, \tag{64}$$

and then they apply a forward Euler/backward Euler (FE/BE) pair to the right-hand side defined in (24):

$$\eta_n^{[p+1]} = \eta_{n-1}^{[p+1]} + h_n\big[f_I\big(\eta_n^{[p+1]}\big) - f_I\big(\eta_n^{[p]}\big)\big] \\ + h_n\big[f_E\big(\eta_{n-1}^{[p+1]}\big) - f_E\big(\eta_{n-1}^{[p]}\big)\big] + h\sum_{m=1}^{M} w_{n,m} f\big(\eta_m^{[p]}\big). \tag{65}$$

With a straightforward extension the results from the present work point out that high-order accuracy can be achieved where there are no forward Euler time steps on the explicit term $f_E(y)$. That is, we propose examining the modified SISDC method

$$\eta_n^{[p+1]} = \eta_{n-1}^{[p+1]} + h_n\big[f_I\big(\eta_n^{[p+1]}\big) - f_I\big(\eta_n^{[p]}\big)\big] + h\sum_{m=1}^{M} w_{n,m} f\big(\eta_m^{[p]}\big). \tag{66}$$

Note that *the "base solver" for this method is not even consistent with the underlying ODE*. This serves as another example of how the findings from this work permit modifications to classical SDC methods in order to produce nearby variations. As an additional benefit, this reduces the computational coding complexity by asking the user to only define f and f_I as opposed to f, f_I, and f_E. This type of modification can be readily found after understanding the source of high-order accuracy inherent to the SDC framework.

Given that the iterative Picard methods demonstrate smaller errors by dropping the forward Euler terms in the right-hand side of the iterations from the SDC solver, one might expect that this method has better accuracy than its SISDC parent. In the event when $f_I = 0$ (or is small), this would be true because in that case we would be comparing explicit SDC to Picard iteration, and we have already shown that those errors are smaller for some problems. However, we will shortly see that this is not necessarily the case. Even though this modification does not affect the overall order of the solver, we will show that for the following test case it does not improve the total overall error of the solver. With that being said, we believe it is still important to understand the source of the overall order of the SDC solvers, because only then can new methods be developed from the existing framework.

mesh	SISDC	order	modified SISDC	order
4	2.24×10^{-02}		6.45×10^{-02}	
8	6.06×10^{-04}	5.21	2.84×10^{-03}	4.51
16	4.11×10^{-05}	3.88	1.91×10^{-04}	3.89
32	3.44×10^{-06}	3.58	1.46×10^{-05}	3.71
64	2.56×10^{-07}	3.75	1.01×10^{-06}	3.85
128	1.78×10^{-08}	3.85	6.68×10^{-08}	3.92
256	1.17×10^{-09}	3.93	4.29×10^{-09}	3.96
512	7.26×10^{-11}	4.01	2.69×10^{-10}	3.99

Table 5. Van der Pol oscillator. Here we present numerical results where we compare the implicit classical method defined in (24) as well as the modified semi-implicit SDC method defined in (66) against each other. Despite the fact that theory can show that the errors could be smaller for the modified method that relies solely on backward Euler time stepping embedded within Picard iteration, in this case the classical SISDC method based forward/backward Euler time stepping outperforms the other solver with its smaller error constants.

Van der Pol's equation. As a prototypical IMEX example, we include results for Van der Pol's equation

$$x''(t) = -x(t) + \mu(1 - x(t))^2 x'(t) \tag{67}$$

with appropriate initial conditions. After the usual transformation of $y_1(t) = x(t)$ and $y_2(t) = \mu x'(t)$, and rescaling time through $t \to t/\mu$, we have the system of differential equations [30; 25]

$$y_1' = y_2, \qquad y_2' = \frac{-y_1 + (1 - y_1^2)y_2}{\epsilon}, \qquad \epsilon = \frac{1}{\mu^2}. \tag{68}$$

In an IMEX setting, this problem is typically split into $f_E(y) = (y_2, 0)$ and $f_I = (0, (-y_1 + (1 - y_1^2)y_2)/\epsilon)$ as an effort to account for the stiffness as $\epsilon \to 0$. For this problem we only seek to verify the high-order accuracy of the classical semi-implicit SDC method defined in (24), denoted by SISDC, as well as the modified solver defined in (66), denoted by "modified SISDC". With this aim in mind, we set $\epsilon = 1$ so that the equations remain nonstiff, and we integrate to a long final time of $T = 4$. The initial conditions are the same as those found in an example in [25], which are $y_1(0) = 2$ and $y_2(0) = -0.666666654321$. In Table 5, we compare a convergence study for the fourth-order versions of these two methods where we use a total of four equispaced quadrature points, a provisional solution defined by the split forward/backward Euler method, as well as three corrections in the solver. For this problem, we find that the classical SISDC method has slightly smaller errors, despite what theory might otherwise predict we could observe. For problems where f_I is negligible or small, the modified method should outperform the SISDC solver.

Other values for ϵ produce similar findings, where the usual order reduction can be found as ϵ approaches zero. In other cases, the two solvers have similar behavior, and both show high-order accuracy for large values of ϵ. For brevity, these other results are omitted.

6. Conclusions

In this work we present rigorous error bounds for both explicit and implicit spectral deferred correction methods. Unlike most presentations that introduce SDC methods as methods that iteratively correct provisional solutions by solving an error equation, our work hinges on the fact that the basic solver can be recast as a variation on Picard iteration. This observation allows new SDC methods to be developed through modifications of the (forward or backward) Euler part of the iterative procedure. In addition, we present some analysis for SDC methods constructed with higher-order base solvers. In the numerical results section we present some sample variations that serve to indicate that the choice of the base solver need not be consistent with the underlying ODE in order to obtain a method that converges. That is, our findings indicate that it is not important to use the same low-order solver for each correction step because the desired high-order accuracy can be found in the integral of the residual. However, the choice of the base solver certainly has an impact on the overall scheme. For example, up to the degree of precision of the underlying quadrature rule, the choice of the forward or backward Euler method or even an inconsistent base solver leads to a single pickup on the order of accuracy of the solver with each correction step, whereas the choice of a trapezoidal rule for a base solver yields two orders of pickup with each correction step. For stiff problems, an implicit method constructed with the backward Euler method (or some variation of it) is certainly preferable due to the larger regions of absolute stability. Future work involves further analysis of embedded high-order base solvers as well as exploring further modifications of the solver to modify regions of absolute stability of existing solvers for explicit, implicit, and semi-implicit SDC methods.

References

[1] J. C. Butcher, *Implicit Runge–Kutta processes*, Math. Comp. **18** (1964), 50–64. MR Zbl

[2] T. Buvoli, *A class of exponential integrators based on spectral deferred correction*, preprint, 2015. arXiv

[3] A. Christlieb, W. Guo, M. Morton, and J.-M. Qiu, *A high order time splitting method based on integral deferred correction for semi-Lagrangian Vlasov simulations*, J. Comput. Phys. **267** (2014), 7–27. MR Zbl

[4] A. Christlieb, M. Morton, B. Ong, and J.-M. Qiu, *Semi-implicit integral deferred correction constructed with additive Runge–Kutta methods*, Commun. Math. Sci. **9** (2011), no. 3, 879–902. MR Zbl

[5] A. Christlieb and B. Ong, *Implicit parallel time integrators*, J. Sci. Comput. **49** (2011), no. 2, 167–179. MR Zbl

[6] A. Christlieb, B. Ong, and J.-M. Qiu, *Comments on high-order integrators embedded within integral deferred correction methods*, Commun. Appl. Math. Comput. Sci. **4** (2009), 27–56. MR Zbl

[7] ———, *Integral deferred correction methods constructed with high order Runge–Kutta integrators*, Math. Comp. **79** (2010), no. 270, 761–783. MR Zbl

[8] A. J. Christlieb, Y. Liu, and Z. Xu, *High order operator splitting methods based on an integral deferred correction framework*, J. Comput. Phys. **294** (2015), 224–242. MR Zbl

[9] A. J. Christlieb, C. B. Macdonald, and B. W. Ong, *Parallel high-order integrators*, SIAM J. Sci. Comput. **32** (2010), no. 2, 818–835. MR Zbl

[10] A. J. Christlieb, C. B. Macdonald, B. W. Ong, and R. J. Spiteri, *Revisionist integral deferred correction with adaptive step-size control*, Commun. Appl. Math. Comput. Sci. **10** (2015), no. 1, 1–25. MR Zbl

[11] M. Duarte and M. Emmett, *High order schemes based on operator splitting and deferred corrections for stiff time dependent PDEs*, preprint, 2014. arXiv

[12] A. Dutt, L. Greengard, and V. Rokhlin, *Spectral deferred correction methods for ordinary differential equations*, BIT **40** (2000), no. 2, 241–266. MR Zbl

[13] M. Emmett and M. L. Minion, *Toward an efficient parallel in time method for partial differential equations*, Commun. Appl. Math. Comput. Sci. **7** (2012), no. 1, 105–132. MR Zbl

[14] I. Faragó, *Note on the convergence of the implicit Euler method*, Numerical analysis and its applications (I. Dimov, I. Faragó, and L. Vulkov, eds.), Lecture Notes in Comput. Sci., no. 8236, Springer, 2013, pp. 1–11. MR Zbl

[15] T. Hagstrom and R. Zhou, *On the spectral deferred correction of splitting methods for initial value problems*, Commun. Appl. Math. Comput. Sci. **1** (2006), 169–205. MR Zbl

[16] E. Hairer, S. P. Nørsett, and G. Wanner, *Solving ordinary differential equations, I: Nonstiff problems*, 2nd ed., Springer Series in Computational Mathematics, no. 8, Springer, 1993. MR Zbl

[17] E. Hairer and G. Wanner, *Solving ordinary differential equations, II: Stiff and differential-algebraic problems*, 2nd ed., Springer Series in Computational Mathematics, no. 14, Springer, 1996. MR Zbl

[18] E. Hairer, C. Lubich, and G. Wanner, *Geometric numerical integration: structure-preserving algorithms for ordinary differential equations*, 2nd ed., Springer Series in Computational Mathematics, no. 31, Springer, 2006. MR Zbl

[19] A. C. Hansen and J. Strain, *On the order of deferred correction*, Appl. Numer. Math. **61** (2011), no. 8, 961–973. MR Zbl

[20] J. Huang, J. Jia, and M. Minion, *Accelerating the convergence of spectral deferred correction methods*, J. Comput. Phys. **214** (2006), no. 2, 633–656. MR Zbl

[21] ———, *Arbitrary order Krylov deferred correction methods for differential algebraic equations*, J. Comput. Phys. **221** (2007), no. 2, 739–760. MR Zbl

[22] J. Jia, J. C. Hill, K. J. Evans, G. I. Fann, and M. A. Taylor, *A spectral deferred correction method applied to the shallow water equations on a sphere*, Mon. Weather Rev. **141** (2013), 3435–3449.

[23] S. Y. Kadioglu and V. Colak, *An essentially non-oscillatory spectral deferred correction method for conservation laws*, Int. J. Comput. Methods **13** (2016), no. 5, art. id. 1650027. MR Zbl

[24] J. Kuntzmann, *Neuere Entwicklungen der Methode von Runge und Kutta*, Z. Angew. Math. Mech. **41** (1961), no. S1, T28–T31. Zbl

[25] A. T. Layton, *On the choice of correctors for semi-implicit Picard deferred correction methods*, Appl. Numer. Math. **58** (2008), no. 6, 845–858. MR Zbl

[26] ———, *On the efficiency of spectral deferred correction methods for time-dependent partial differential equations*, Appl. Numer. Math. **59** (2009), no. 7, 1629–1643. MR Zbl

[27] A. T. Layton and M. L. Minion, *Conservative multi-implicit spectral deferred correction methods for reacting gas dynamics*, J. Comput. Phys. **194** (2004), no. 2, 697–715. MR Zbl

[28] ———, *Implications of the choice of quadrature nodes for Picard integral deferred corrections methods for ordinary differential equations*, BIT **45** (2005), no. 2, 341–373. MR Zbl

[29] ———, *Implications of the choice of predictors for semi-implicit Picard integral deferred correction methods*, Commun. Appl. Math. Comput. Sci. **2** (2007), 1–34. MR Zbl

[30] M. L. Minion, *Semi-implicit spectral deferred correction methods for ordinary differential equations*, Commun. Math. Sci. **1** (2003), no. 3, 471–500. MR Zbl

[31] ———, *A hybrid parareal spectral deferred corrections method*, Commun. Appl. Math. Comput. Sci. **5** (2010), no. 2, 265–301. MR Zbl

[32] M. M. Morton, *Integral Deferred Correction methods for scientific computing*, Ph.D. thesis, Michigan State University, 2010. MR

[33] W. Pazner and P.-O. Persson, *Stage-parallel fully implicit Runge–Kutta solvers for discontinuous Galerkin fluid simulations*, J. Comput. Phys. **335** (2017), 700–717. MR Zbl

[34] W. Qu, N. Brandon, D. Chen, J. Huang, and T. Kress, *A numerical framework for integrating deferred correction methods to solve high order collocation formulations of ODEs*, J. Sci. Comput. **68** (2016), no. 2, 484–520. MR Zbl

[35] D. Ruprecht and R. Speck, *Spectral deferred corrections with fast-wave slow-wave splitting*, SIAM J. Sci. Comput. **38** (2016), no. 4, A2535–A2557. MR Zbl

[36] R. Speck, D. Ruprecht, M. Emmett, M. Minion, M. Bolten, and R. Krause, *A multi-level spectral deferred correction method*, BIT **55** (2015), no. 3, 843–867. MR Zbl

[37] T. Tang, H. Xie, and X. Yin, *High-order convergence of spectral deferred correction methods on general quadrature nodes*, J. Sci. Comput. **56** (2013), no. 1, 1–13. MR Zbl

[38] M. Weiser, *Faster SDC convergence on non-equidistant grids by DIRK sweeps*, BIT **55** (2015), no. 4, 1219–1241. MR Zbl

[39] Y. Xia, Y. Xu, and C.-W. Shu, *Efficient time discretization for local discontinuous Galerkin methods*, Discrete Contin. Dyn. Syst. Ser. B **8** (2007), no. 3, 677–693. MR Zbl

[40] P. E. Zadunaisky, *On the estimation of errors propagated in the numerical integration of ordinary differential equations*, Numer. Math. **27** (1976), no. 1, 21–39. MR Zbl

Received June 19, 2017. Revised November 7, 2018.

MATHEW F. CAUSLEY: mcausley@kettering.edu
Mathematics Department, Kettering University, Flint, MI, United States

DAVID C. SEAL: seal@usna.edu
Department of Mathematics, United States Naval Academy, Annapolis, MD, United States