Semi-Implicit Spectral Deferred Correction Methods for Ordinary Differential Equations

Michael L. Minion[1]
Department of Mathematics, Phillips Hall, CB 3250
University of North Carolina
Chapel Hill, NC 27599
E-mail: minion@amath.unc.edu

Key Words: Stiff equations, Fractional Step Methods, Method of Lines

Received May 00, 0000; revised February 0, 0000; accepted February 0, 0000

A semi-implicit formulation of the method of spectral deferred corrections (SISDC) for ordinary differential equations with both stiff and non-stiff terms is presented. Several modifications and variations to the original spectral deferred corrections method by Dutt, Greengard, and Rokhlin concerning the choice of integration points and the form of the correction iteration are presented. The stability and accuracy of the resulting ODE methods are explored analytically and numerically. The SISDC methods are intended to be combined with the method of lines approach to yield a flexible framework for creating higher-order semi-implicit methods for partial differential equations. A discussion and numerical examples of the SISDC method applied to advection-diffusion type equations are included. The results suggest that higher-order SISDC methods are more efficient than semi-implicit Runge-Kutta methods for moderately stiff problems in terms of accuracy per function evaluation.

1. INTRODUCTION

The question of how to construct stable and accurate numerical methods for the solution of initial value problems determined by ordinary differential equations (ODEs) has been studied extensively and with a great deal of success in the last thirty years. In particular, for non-stiff ODEs, explicit high-order methods such as Runge-Kutta, multi-step, or predictor corrector methods are well understood, and

[1]This work was supported by the Director, DOE Office of Science, Office of Advanced Scientific Computing Research, Office of Mathematics, Information, and Computational Sciences, Applied Mathematical Sciences Program, under contract DE-AC03-76SF00098 and by the National Science Foundation Grant DMS-9973290.

are readily available, e.g. [16, 8, 3]. For stiff systems, where efficient methods are implicit, the issues can be more complicated, but still many good methods have been developed, e.g. [17]. Nevertheless, Dutt, Greengard, and Rokhlin recently presented a new variation of the classical method of deferred corrections, the spectral deferred correction method (SDC) [12]. Implicit versions of this method are shown to have good stability and accuracy properties for stiff equations even for versions with very high-order accuracy (up to thirtieth order in [12]).

Traditional explicit or implicit methods for temporal integration are often inefficient when the equation being considered possesses two or more widely varying time scales. The primary examples of such equations that motivate the current work result from the temporal discretization of partial differential equations (PDEs) which model physical systems with two or more disparate time scales. Two well-known examples are advection-diffusion-reaction problems and systems containing fluid-membrane interactions. A common strategy for producing higher-order methods for PDEs is the so called *method of lines* approach (hereafter MOL). In MOL, a PDE is discretized in space only, which results in a set of coupled ODEs, one for each discretization variable. These ODEs, which often contain both stiff and non-stiff terms, can then be solved with any appropriate integration method. When stiff terms are present, it is often expensive to implement fully implicit ODE methods because spatial nonlinearities in the PDEs would require the solution of large coupled nonlinear equations. It is therefore desirable to use *semi-implicit* ODE methods which allow stiff terms to be treated implicitly and non-stiff terms explicitly.

Many semi-implicit methods for ODEs have indeed appeared in recent years, and there are disadvantages to each, particularly when very accurate solutions are required. In this paper, a family of semi-implicit SDC methods will be introduced which are designed to overcome the disadvantages of existing methods. The main advantage of SDC methods is that one can use a simple numerical method (even a first-order method) to compute a solution with higher-order accuracy. This is accomplished by using the numerical method to solve a series of correction equations during each time step, each of which increases the order of accuracy of the solution. The flexibility in the choice of the method used in the deferred correction iterations makes SDC methods particularly attractive to problems possessing disparate time scales since a lower-order accurate semi-implicit or time-split approach can be used during each iteration without limiting the overall solution to lower-order accuracy. In this work, a simple first-order, semi-implicit method is used in the context of SDC to construct higher-order semi-implicit SDC methods (hereafter SISDC methods).

The outline of the paper is as follows. After a short description of the SDC method in Sect. 2, a semi-implicit version for a general class of ODEs is introduced in Sect. 3, and a discussion of the advantages of SISDC methods over existing approaches is presented. The stability and accuracy of SISDC methods are investigated in Sect. 4. Techniques for reducing the number of function evaluations necessary for a given order of accuracy are also discussed. In Sect. 5, numerical examples using an initial value ODE and two advection-diffusion type PDEs demonstrate the accuracy and efficiency of the methods.

2. THE METHOD OF SPECTRAL DEFERRED CORRECTIONS

In this section, the original SDC method appearing in [12] will first be reviewed. A semi-implicit version suitable for equations with both stiff and non-stiff terms is presented in the following section along with a discussion of the advantages of SISDC methods over existing methods.

2.1. Mathematical Preliminaries

The initial value ODE takes the form

$$\phi'(t) = F(t, \phi(t)) \qquad t \in [a, b] \tag{1}$$
$$\phi(a) = \phi_a. \tag{2}$$

Here, the solution $\phi(t)$ and initial value ϕ_a are in $\mathbb{C}^N$ and $F : \mathbb{R} \times \mathbb{C}^N \to \mathbb{C}^N$. It is assumed that F is smooth so that the discussion of higher-order methods is appropriate. This assumption is more stringent than the Lipschitz continuity necessary to assure that solutions exist (for some time) and are unique. (See any standard text on ODE for the associated proofs.)

The SDC method of Dutt, Greengard, and Rokhlin is based on the following observations concerning the integral form of the solution to Eqs. (1)-(2)

$$\phi(t) = \phi_a + \int_a^t F(\tau, \phi(\tau)) d\tau. \tag{3}$$

Given an approximation to the solution $\tilde{\phi}(t)$, the goal of all deferred corrections strategies is to formulate an equation for the correction $\delta(t) = \phi(t) - \tilde{\phi}(t)$. Following [12], note that a measure of the error in $\tilde{\phi}(t)$ based on Eq. (3) can be written

$$E(t, \tilde{\phi}) = \phi_a + \int_a^t F(\tau, \tilde{\phi}(\tau)) d\tau - \tilde{\phi}(t). \tag{4}$$

Since $\phi(t) = \tilde{\phi}(t) + \delta(t)$, Eq. (3) yields

$$\tilde{\phi}(t) + \delta(t) = \phi_a + \int_a^t F(\tau, \tilde{\phi}(\tau) + \delta(\tau)) d\tau, \tag{5}$$

which implies

$$\delta(t) = \int_a^t F(\tau, \tilde{\phi}(\tau) + \delta(\tau)) - F(\tau, \tilde{\phi}(\tau)) d\tau + E(t, \tilde{\phi}). \tag{6}$$

This equation will be referred to as the *correction equation*.

Eq. (6) also gives an indication of how well $E(t, \tilde{\phi})$ approximates $\delta(t)$. In a numerical method for ODEs, the interval $[a, b]$ generally spans a single time step $[t_n, t_{n+1}] = [t_n, t_n + \Delta t]$. If F is Lipschitz continuous in ϕ and $||\tilde{\phi} - \phi|| = O(\Delta t^r)$ on $[t_n, t_{n+1}]$, then $||\delta(t) - E(t, \tilde{\phi})|| = O(\Delta t^{r+1})$. Note also that if F is a function of t alone, then $\delta(t) = E(t, \tilde{\phi})$.

The general strategy of the SDC method is to use a simple numerical method to compute a provisional solution ϕ on the interval $[t_n, t_{n+1}]$, and then to solve a series of correction equations based on Eq. (6), each of which improves the accuracy of the provisional solution. In [12], a forward or backward Euler type method is used

for computing both the provisional solutions and the corrections. This procedure is reviewed in the next section and a semi-implicit version is then presented in Sect. 3.

2.2. SDC based on Euler Methods

Given Eqs. (1)-(2) and some interval $[t_n, t_{n+1}]$ on which the solution is desired, the SDC method proceeds by first dividing $[t_n, t_{n+1}]$ into p subintervals by choosing points t_m for $m = 0 \ldots p$, with $t_n = t_0 < t_1 < \ldots < t_p = t_{n+1}$. In the following, the interval $[t_n, t_{n+1}]$ will be referred to as a *time step* while a subinterval $[t_m, t_{m+1}] = [t_m, t_m + \Delta t_m]$ will be referred to as a *substep*. An approximate solution $\phi^0(t_m)$ is computed for $m = 0 \ldots p$ using the standard forward Euler method for non-stiff equations or the backward Euler method for stiff problems.

Next, a sequence of corrections $\delta^k(t_m)$ is computed by approximating Eq. (6) which provides an increasingly accurate approximation to the solution $\phi^{k+1} = \phi^k + \delta^k$. In order to do this, the function $E(t, \phi^k(t))$ must be approximated using numerical quadrature. For this reason the points t_m are chosen in [12] to correspond to the nodes for Gaussian quadrature. The choice of nodes is discussed further at the end of Sect. 3.

Eq. (6) is also approximated by using forward or backward Euler. For instance, using the notation $\delta_m^k = \delta^k(t_m)$ (and likewise for ϕ_m^k and $E_m(\phi^k)$), the backward Euler approximation to the correction equation is

$$\begin{aligned} \delta_{m+1}^k &= \delta_m^k + \Delta t_m [F(t_{m+1}, \phi_{m+1}^k + \delta_{m+1}^k) - F(t_{m+1}, \phi_{m+1}^k)] \\ &+ E_{m+1}(\phi^k) - E_m(\phi^k). \end{aligned} \tag{7}$$

By rearranging terms, a direct equation for ϕ^{k+1} can be derived. Let $I_m^{m+1}(\phi^k)$ denote the numerical quadrature approximation to

$$\int_{t_m}^{t_{m+1}} F(\tau, \phi^k(\tau)) d\tau,$$

then using Eq. (4),

$$E_{m+1}(\phi^k) - E_m(\phi^k) = I_m^{m+1}(\phi^k) - \phi_{m+1}^k + \phi_m^k.$$

Hence, Eq. (7) is equivalent to

$$\phi_{m+1}^{k+1} = \phi_m^{k+1} + \Delta t_m [F(t_{m+1}, \phi_{m+1}^{k+1}) - F(t_{m+1}, \phi_{m+1}^k)] + I_m^{m+1}(\phi^k). \tag{8}$$

When SISDC methods are used with MOL, this equation allows one to consider boundary conditions for ϕ^{k+1} directly. In the numerical implementation however, the incremental form (7) is used to avoid a loss of precision.

As long as the integral terms $I_m^{m+1}(\phi^k)$ are computed to $O(\Delta t^{k+1})$ (which in this setting implies some smoothness in F), each iteration of the correction equation will increase the order of accuracy of the solution by one order. Hence for k iterations of the correction equation, the above procedure will produce an approximate solution with truncation error of size $O(\Delta t^{k+2})$ and global accuracy $O(\Delta t^{k+1})$. (See e.g. [28] for analysis of deferred correction methods.) If $F(t, \phi)$ is Lipschitz continuous in ϕ, it is straightforward to show that the numerical method is also. Hence, since

the entire SDC procedure is a single-step method, convergence is assured (see e.g. [3]).

Time step selection and error estimation for SDC methods is facilitated by the fact that approximations to the correction are being directly computed. The size of these corrections can be monitored as part of a time step selection procedure. A number of possibilities are discussed in [12], and all these procedures are applicable for the methods introduced here. Note that despite the number of function evaluations needed in the SDC iterations, these methods are shown in [12] to be competitive with existing methods in terms of accuracy per function evaluation.

3. SEMI-IMPLICIT SPECTRAL DEFERRED CORRECTIONS

The key advantage that SDC methods provide in the design of semi-implicit methods for ODEs is that the numerical method used in each deferred correction can be very simple. It seems natural therefore to use simple semi-implicit methods in the context of SDC to construct higher-order semi-implicit methods. The details of such schemes are presented in this section followed by a discussion of alternative methods.

3.1. Details of The Semi-Implicit Methods

Consider the ODE

$$\begin{aligned} \phi'(t) &= F(t,\phi(t)) = F_E(t,\phi(t)) + F_I(t,\phi(t)) \\ \phi(a) &= \phi_a, \end{aligned} \tag{9}$$

where $t \in [a,b]$, F_E is a non-stiff term and F_I is a stiff term. The subscripts refer to the desire to treat the non-stiff terms explicitly and the stiff terms implicitly. This can be accomplished with a few straightforward modifications of the SDC procedure described above.

A first-order numerical method for finding $\phi^0(t_m)$ is simply

$$\phi^0_{m+1} = \phi^0_m + \Delta t_m [F_E(t_m,\phi^0_m) + F_I(t_{m+1},\phi^0_{m+1})]. \tag{10}$$

When F_I is linear, this type of method has also been referred to in the literature as the W-Euler method. (See [23] for an analysis of W-methods). Likewise, a semi-implicit version of Eq. (7) is

$$\begin{aligned} \delta^k_{m+1} &= \delta^k_m + \Delta t_m [F_E(t_m,\phi^k_m + \delta^k_m) - F_E(t_m,\phi^k_m) \\ &+ F_I(t_{m+1},\phi^k_{m+1} + \delta^k_{m+1}) - F_I(t_{m+1},\phi^k_{m+1})] + E_{m+1}(\phi^k) - E_m(\phi^k). \end{aligned} \tag{11}$$

As in Eq. (8), this equation can be rearranged to yield a direct update for ϕ^{k+1}_{m+1}

$$\begin{aligned} \phi^{k+1}_{m+1} &= \phi^{k+1}_m + \Delta t_m [F_E(t_m,\phi^{k+1}_m) - F_E(t_m,\phi^k_m) \\ &+ F_I(t_{m+1},\phi^{k+1}_{m+1}) - F_I(t_{m+1},\phi^k_{m+1})] + I^{m+1}_m(\phi^k). \end{aligned} \tag{12}$$

Eqs. (10) and (12) can be used to construct SISDC methods of arbitrarily high order of accuracy.

A few comments on the form of the quadrature for $I^{m+1}_m(\phi^k)$ are required. In [12], the points t_m are chosen to be the standard Gaussian quadrature nodes, and the solution is only integrated at these nodes on the interior of the interval. This requires

the use of extrapolation to yield the solution at the endpoint of the interval. Here, the nodes from Gauss-Lobatto quadrature are used instead, hence no extrapolation is necessary. Since the quadrature must be done for each subinterval $[t_m, t_{m+1}]$, there are actually p quadrature rules

$$I_m^{m+1}(\phi^k) = \sum_{l=0}^{p} q_m^l F(t_l, \phi_l^k) \tag{13}$$

for $m = 0 \dots p-1$. The coefficients q_m^l can be precomputed, and the quadrature is reduced to a simple matrix-vector multiplication.

Since the function $F(t_m, \phi_m^k)$ is known at $p+1$ Gauss-Lobatto quadrature nodes, the approximation to the integral of F over the entire time step, i.e.

$$\int_{t_n}^{t_{n+1}} F(\tau, \phi^k(\tau)) d\tau,$$

can be computed with error $O(\Delta t^{2p+1})$. The integrals in Eq. (13), however, are computed with error $O(\Delta t^{p+2})$ since they are simply computed as the integral of the interpolating polynomial over the subinterval. Since a truncation error $O(\Delta t^{K+1})$ is required for a Kth-order method, choosing a value of $p = K - 1$ provides sufficient accuracy in the quadratures. This choice is also sufficient to provide an $O(\Delta t^K)$ approximation of the solution for any desired point through polynomial interpolation at the quadrature nodes.

For ease of identification, the SISDC method using P Gauss-Lobatto nodes (i.e. $P-1$ substeps) and K total iterations (or $K-1$ iterations of the correction equation) will be denoted $SISDC_P^K$. This is slightly different than the notation in [12], but has the benefit that an $SISDC_P^K$ method has global order of accuracy $\min(K, P)$. In the remainder of the paper, the methods considered will have $P = K$.

Note that the sum of the quadratures in Eq. (13) is the Gauss-Lobatto quadrature rule, hence the sum of the quadrature errors on the subintervals is again $O(\Delta t^{2p+1})$. Therefore, in the time marching of the correction equation the individual errors cancel to an extent, and it is reasonable to suggest that $SISDC_{K-1}^K$ methods should perform similarly to $SISDC_K^K$ methods while requiring fewer function evaluations per step. This has been observed in numerical tests, although the observed differences were slight in terms of accuracy per function evaluation. Lastly, rather than using Gauss-Lobatto quadrature nodes, it is certainly possible to choose the nodes in a way which results in the quadrature rules on each subinterval being more accurate at the cost of reducing the accuracy of the quadrature over the entire integral. However, it is not clear *a priori* whether this would improve the overall accuracy of the method, and this idea is not pursued further here.

3.2. Comparison with Existing Methods

Like fully implicit or explicit SDC methods, SISDC methods can in principle be constructed with an arbitrarily high order of accuracy. In general, the smaller the tolerance for error, the more attractive a higher-order method becomes, and as is shown in the next section, higher-order SISDC methods retain the good stability properties of lower-order methods. In this section, multi-step methods, Runge-

Kutta methods and operator splitting methods are all compared to SISDC methods[2].

High-order semi-implicit linear multi-step methods have been well documented (see e.g. [13]) and have also been studied in the MOL context (see e.g. [5, 13, 2]). These methods have the disadvantage that they are not self-starting, they are cumbersome when variable time-stepping is required, and higher-order versions typically suffer from severe stability restrictions.

Because of the disadvantages mentioned above, Runge-Kutta methods have become the most popular methods for the numerical solution of ODEs in the context of MOL. Semi-implicit Runge-Kutta methods (also called IMEX or additive Runge-Kutta methods) have been developed which are similar in style to fully implicit SDIRK methods and are also closely related to W-methods [4, 24, 23, 19]. The recent paper by Kennedy and Carpenter [19] presents detailed numerical experiments using semi-implicit Runge-Kutta methods which have good stability properties and are efficient in terms of accuracy per function evaluation for orders up to five (methods with order higher than five appear to have unfavorable stability properties). Similar methods have also been used with MOL for PDEs with stiff and non-stiff components (see e.g. [27, 4, 19, 9]).

In all of the Runge-Kutta methods mentioned above, the value of the solution computed during a time step is given by a particular linear combination of the right hand side of the ODE evaluated at intermediate or stage values, some of which have truncation errors of lower-order accuracy than the final value. The linear combination is chosen so that the lower-order truncation errors in the stage values cancel, and hence the exact form of these errors must be known. This makes the generation of Runge-Kutta methods for more than two disparate time scales very difficult. It also limits the flexibility of how different terms in the equation are calculated.

In SDC methods, lower-order intermediate solutions are used only to calculate forcing terms in the subsequent correction equation, and hence do not contribute directly to the final value. This allows for a straightforward extension to problems with three or even more time scales, and also allows different time steps for different terms in the equation (see [6]). Two different time steps could be used for SISDC methods as well, but in the examples presented here, the cost of evaluating the explicit piece is much less expensive than that of computing the implicit piece, so there is little point in having a larger explicit time step. For some methods for PDEs (e.g. those for hyperbolic conservation laws involving the solution of Riemann problems), the cost of computing the explicit piece may not be negligible.

Operator splitting is another approach to integrating an equation of the form of Eq. (9) that allows flexibility in the way different terms in the equation are updated. In fact, the predictor step in SISDC can be thought of as a simple Strang splitting using forward Euler on the explicit piece and backward Euler on the implicit piece. The correction equation can further be thought of as correcting both the integration error and splitting error simultaneously.

[2]A complete review of the literature will not be attempted. The citations given represent only a selection of recent papers. See the bibliographies in the cited works for more complete citations.

It has been well documented that when combined with MOL, Runge-Kutta methods typically suffer from a reduction in the order of accuracy when time dependent boundary conditions are prescribed unless special care is taken when imposing intermediate boundary conditions [15, 20, 26, 11, 1, 25]. In particular, the exact boundary conditions imposed for the PDE cannot be used for intermediate boundary conditions without a degradation occurring in the order of accuracy. The loss of accuracy appears as a boundary layer because the error at the boundary for intermediate stage solutions is forced to be zero, while the error in the interior of the domain is of the size of the stage order of the method. The loss of accuracy occurs when spatial derivatives of this boundary layer are computed since spatial derivatives are not Lipshitz continuous. Suggestions for restoring full accuracy in certain cases have been proposed for explicit Runge-Kutta methods (e.g. [1, 25, 10]), but at present, no general strategy for semi-implicit Runge-Kutta methods has been developed.

A similar problem occurs as well with SISDC methods, but because intermediate solutions do not contribute directly to the final solution, there is more flexibility in the imposition of boundary conditions. Accuracy at the boundary can be iteratively improved with the deferred corrections along with error in the interior. A paper addressing the imposition of boundary conditions for SISDC methods applied to PDEs where diffusion is treated implicitly is in preparation [22].

4. STABILITY AND ACCURACY ANALYSIS

A complete stability analysis of numerical methods for nonlinear PDEs with multiple time scales is possible only in the most specialized instances. With MOL, one can instead perform a linearized analysis to yield insight into how the stability of the method depends on the time step. Therefore a complete understanding of the stability of the underlying ODE method is essential.

The stability of a given ODE method is traditionally studied by considering the model problem

$$\begin{aligned} \phi'(t) &= \lambda\phi \\ \phi(0) &= 1, \end{aligned}$$

where λ is some complex constant. For a given method, let $\phi_1(\lambda)$ denote the solution computed using one step of the method and $\Delta t = 1$. The stability region for the method can then be defined by the set of λ for which $|\phi_1(\lambda)| < 1$. Alternatively, the stability region can be interpreted as the region in which the product $\lambda\Delta t$ must lie for a given λ so that the numerical solution is bounded as the number of time steps goes to infinity. Hence $\phi_1(\lambda)$ is called the amplification factor.

To examine the stability of SISDC methods, let $\lambda = \alpha + \beta i$ for real constants α and β. The model problem can then be split into

$$\begin{aligned} \phi'(t) &= \beta i\phi + \alpha\phi \\ \phi(0) &= 1, \end{aligned}$$

where the imaginary piece of the equation is dealt with explicitly and the real piece implicitly. This is the relevant model problem resulting from a linearized stability

analysis of methods for advection-diffusion type PDEs. The stability diagrams for $SISDC_K^K$ methods are computed on the model problem. The stability diagrams for methods of third through seventh order are shown in Fig. 1. The plots suggest that each method is $A(\alpha)$-stable for roughly the same α.

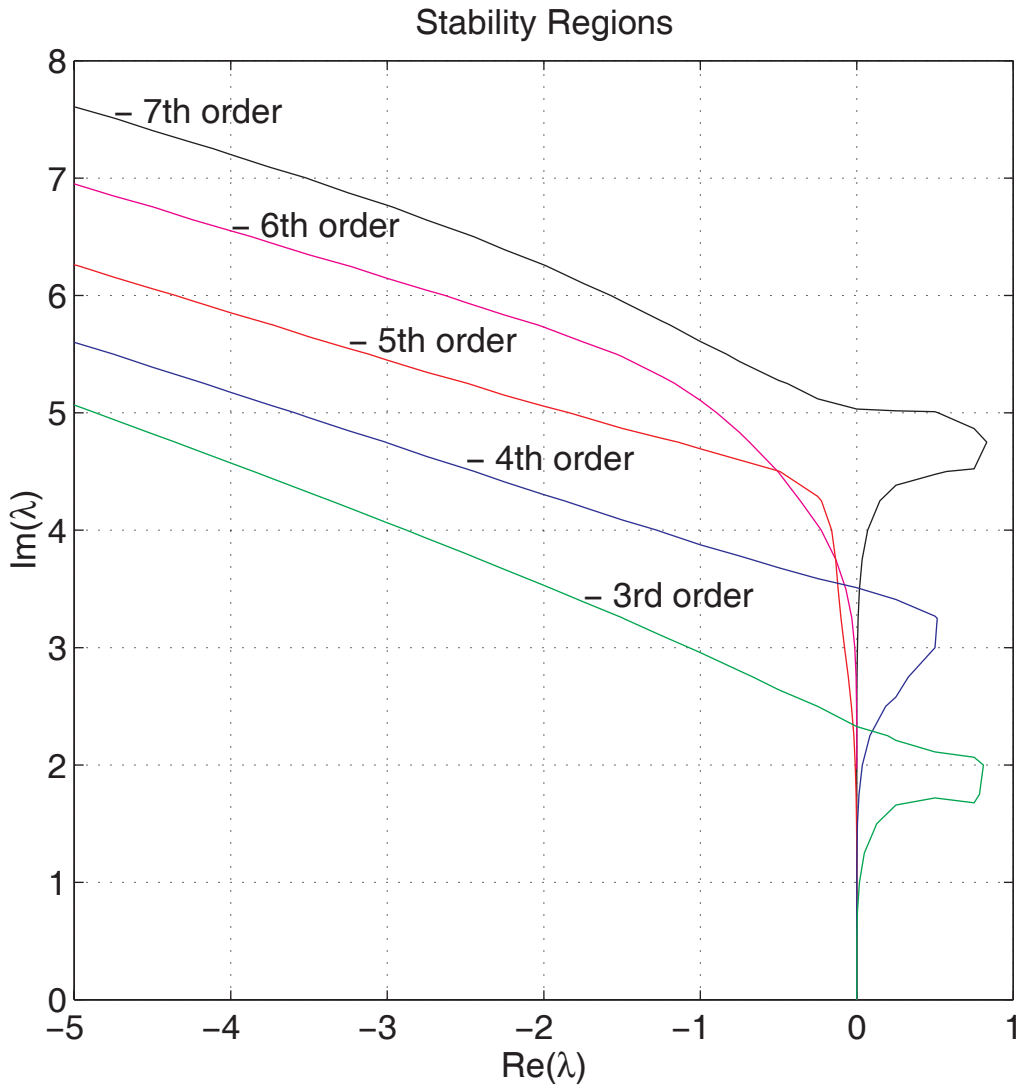

FIG. 1. Stability regions for the $SISDC_K^K$ methods with K ranging from three through seven.

Since the amount of work being done for each step of an SISDC method increases quadratically with the order, it is informative to scale the stability diagrams proportionally. The stability diagrams scaled by the number of implicit function evaluations required per time step, which is $K(K-1)$, are displayed in Fig. 2. It is tempting to draw the conclusion from this figure that the lower-order methods are more efficient since relatively larger time steps can be taken. This is only true however if one ignores the issue of accuracy. As will be discussed in the next sec-

tion, the higher-order methods possess greater relative accuracy. In other words, higher-order methods allow one to use a larger time step for a given error tolerance than lower-order methods.

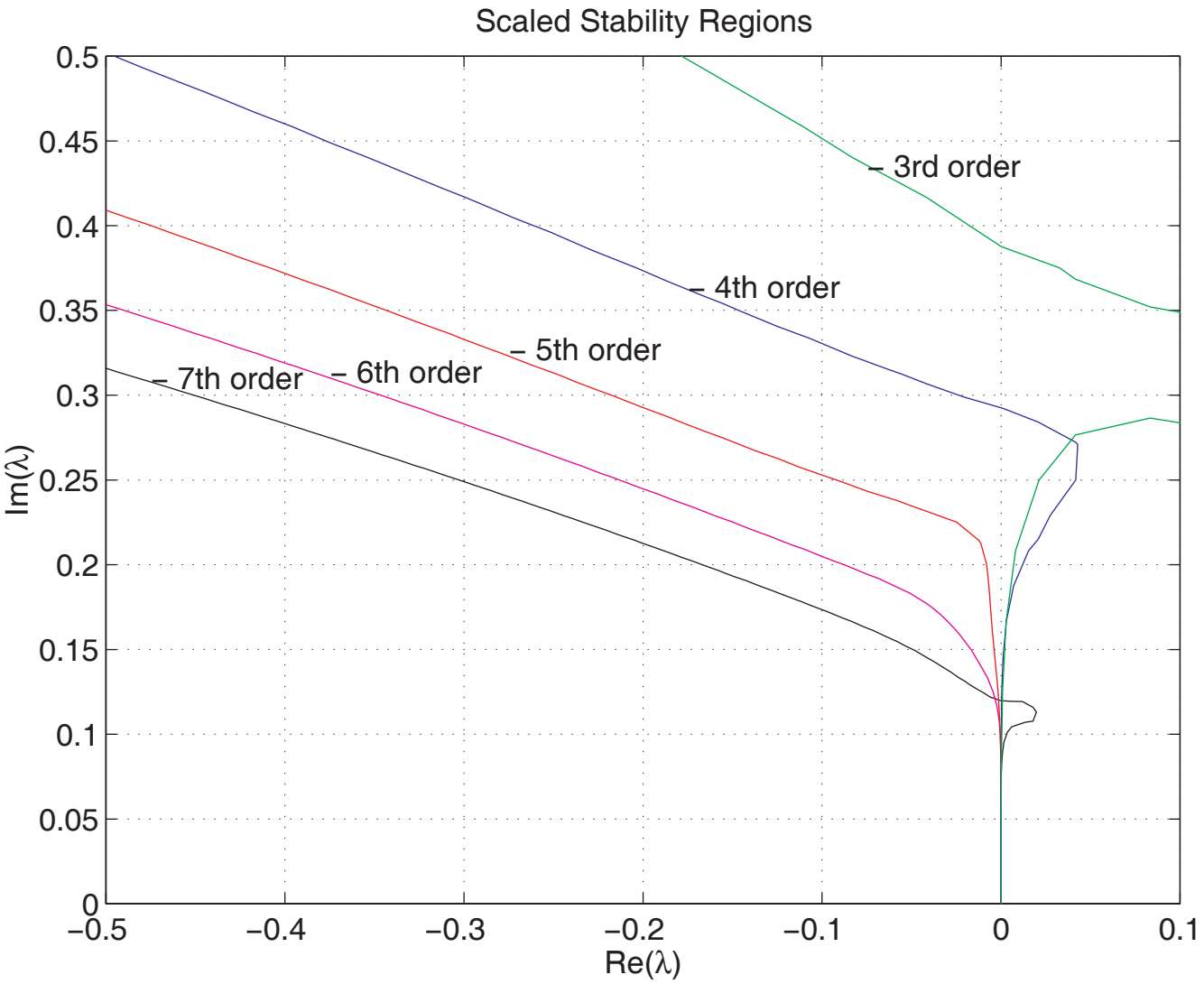

FIG. 2. Scaled stability regions for the $SISDC_K^K$ methods with K ranging from three through seven.

Although the SISDC methods above appear to be $A(\alpha)$-stable, none are L-stable. As mentioned in [12], for the fully implicit SDC methods, $\phi_1(\lambda)$ is simply a rational function of the complex number λ, hence

$$\lim_{|\lambda|\to\infty} \phi_1(\lambda) \tag{14}$$

is independent of how λ actually approaches infinity. Therefore, the implicit SDC method are L-stable if

$$\lim_{\alpha \to -\infty} \phi_1(\lambda) = 0, \tag{15}$$

where again $\lambda = \alpha + \beta i$. None of the implicit SDC methods based on backward Euler are L-stable, but the above limit exists, is easily computed, and is less than one. Hence, as is pointed out in [12], a linear combination of the results from two different SDC methods can be used to construct an L-stable method.

For the SISDC methods, matters are more subtle. The amplification factor is not a rational function of λ, but instead a ratio of polynomials in α and β. Therefore the limit in Eq. (14) depends on how the limit is taken. For fixed β however, the limit in Eq. (15) exists and does not depend on β. None of the SISDC methods studied here have the property that this limit is zero, hence none could possibly be considered L-stable in any generalized sense. It is possible, however, to construct a scheme for which this limit does vanish.

Following [12], let $\mu(k,p)$ be the limit in Eq. (15) for the $SISDC_p^k$ method. As in the fully implicit case, $\mu(k,p)$ can be easily approximated numerically. If k_1, k_2, p_1, and p_2 are such that $\mu(k_1,p_1) \neq \mu(k_2,p_2)$, then the method defined by the linear combination

$$\frac{\mu(k_2,p_2)SISDC_{p_1}^{k_1} - \mu(k_1,p_1)SISDC_{p_2}^{k_2}}{\mu(k_2,p_2) - \mu(k_1,p_1)}$$

would have the property that the limit in Eq. (15) vanishes. However, it is certainly not obvious under what conditions this procedure would be advantageous, and the subject is not pursued further in this paper.

4.1. Order Versus Accuracy

Although informative, the stability diagrams presented in the last section give no information concerning the accuracy of computed solutions. In the context of higher-order methods, a more important measure of the efficiency of a method is the relative accuracy region. Whereas the stability diagram for a method gives an indication of how large the time step can be so that the solution does not contain catastrophic error, accuracy diagrams give an indication of how large a time step can be so that certain error criteria are met. In this section, the accuracy diagrams of SISDC methods of different orders will be presented and compared to those of a particular additive Runge-Kutta method.

For a given accuracy ϵ, the accuracy region is the set of $\lambda \in \mathbb{C}$ for which $|\phi_1(\lambda) - e^{\lambda}| < \epsilon$. The accuracy regions for the SISDC methods for $\epsilon = 10^{-4}$ are shown in Fig. 3. As one would expect, the size of the accuracy region increases with the order of the method. As with stability diagrams, it is more relevant to consider stability diagrams which are scaled by the number of function evaluations required for the method. The scaled accuracy regions are shown in Fig. 4, and unlike the stability diagrams, the higher-order methods still have larger accuracy regions after scaling. The scaled regions for $\epsilon = 10^{-8}$, where the differences are more pronounced, are displayed in Fig. 5. This implies that the greater the accuracy required of the solution, the more cost-effective higher-order methods become.

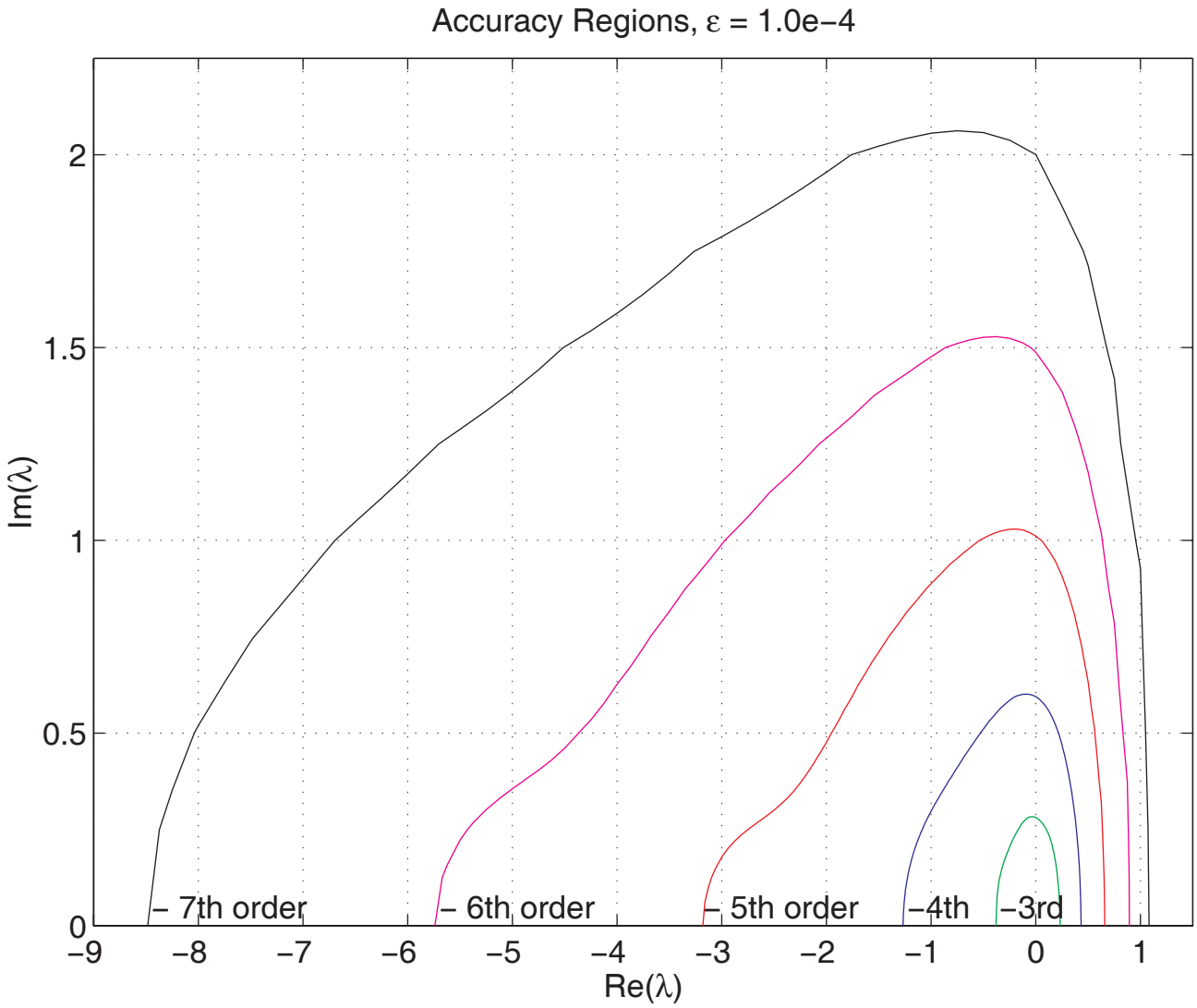

FIG. 3. Accuracy regions for the $SISDC_K^K$ methods with K ranging from three through seven for error tolerance 10^{-4}.

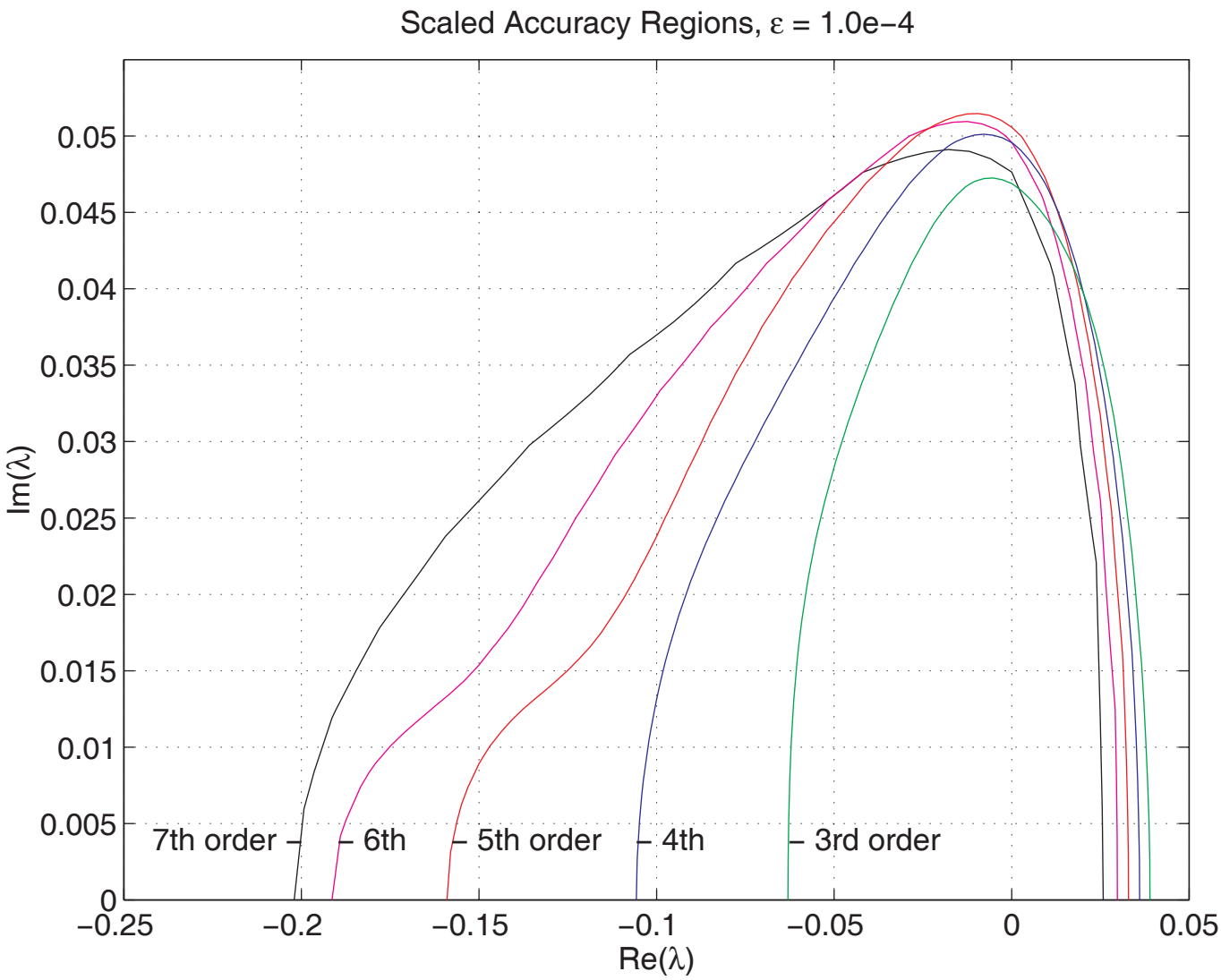

FIG. 4. Scaled accuracy regions for the $SISDC_K^K$ methods with K ranging from three through seven for error tolerance 10^{-4}.

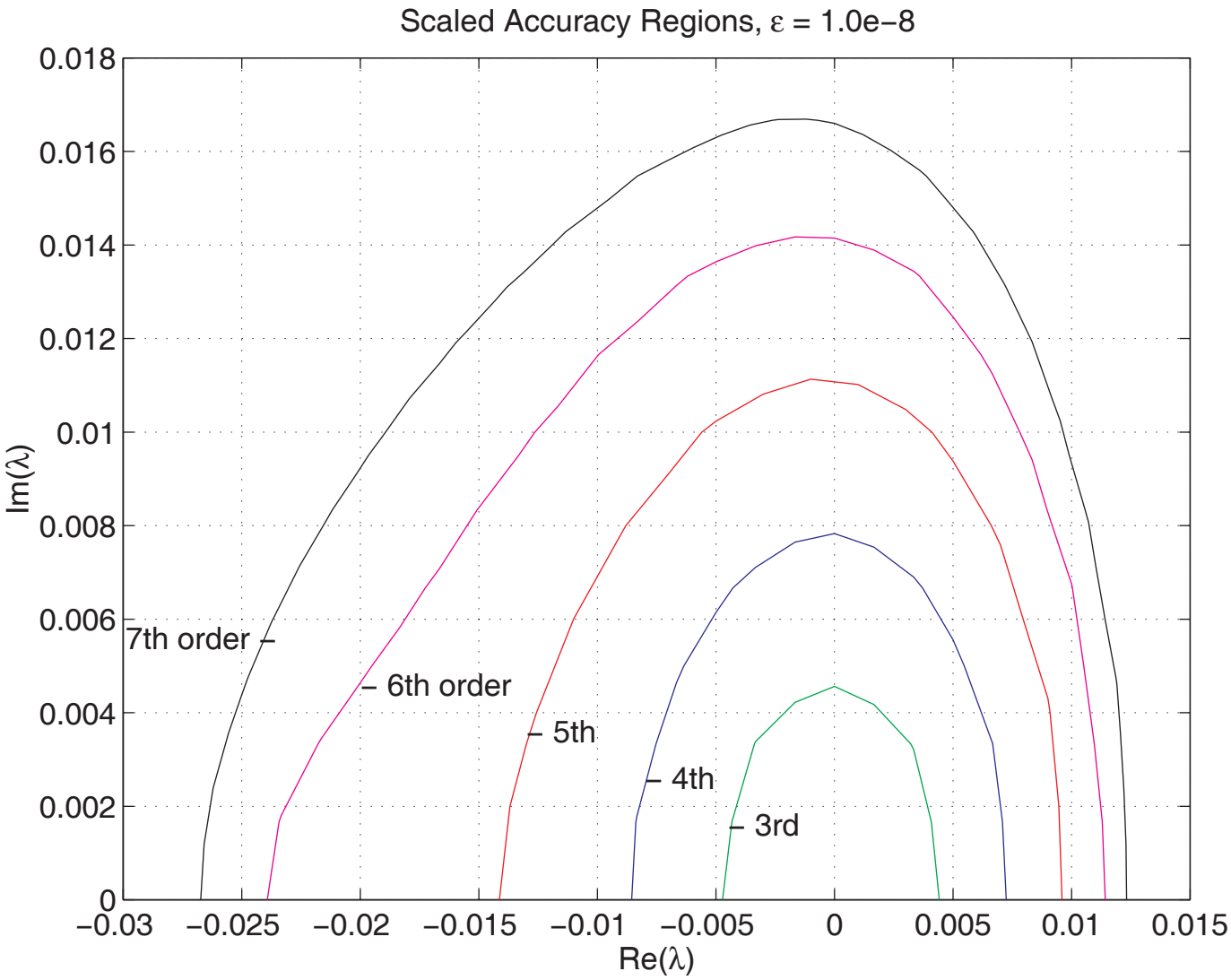

FIG. 5. Scaled accuracy regions for the $SISDC_K^K$ methods with K ranging from three through seven for error tolerance 10^{-8}.

As a comparison, the accuracy diagram of an SISDC method is compared to a semi-implicit additive Runge-Kutta method from [19]. Fig. 6 shows the scaled accuracy diagrams for the fourth-order ARK4(3)6L[2]SA method and the $SISDC_6^7$ method for $\epsilon = 10^{-8}$. At this level of precision, the two methods are of comparable efficiency in terms of accuracy per function evaluation. Since the SISDC method is of higher order, the size of the corresponding accuracy region will shrink more slowly than that of the ARK method as ϵ is further decreased. This is confirmed in Fig. 6 which shows the scaled accuracy diagrams for $\epsilon = 10^{-10}$.

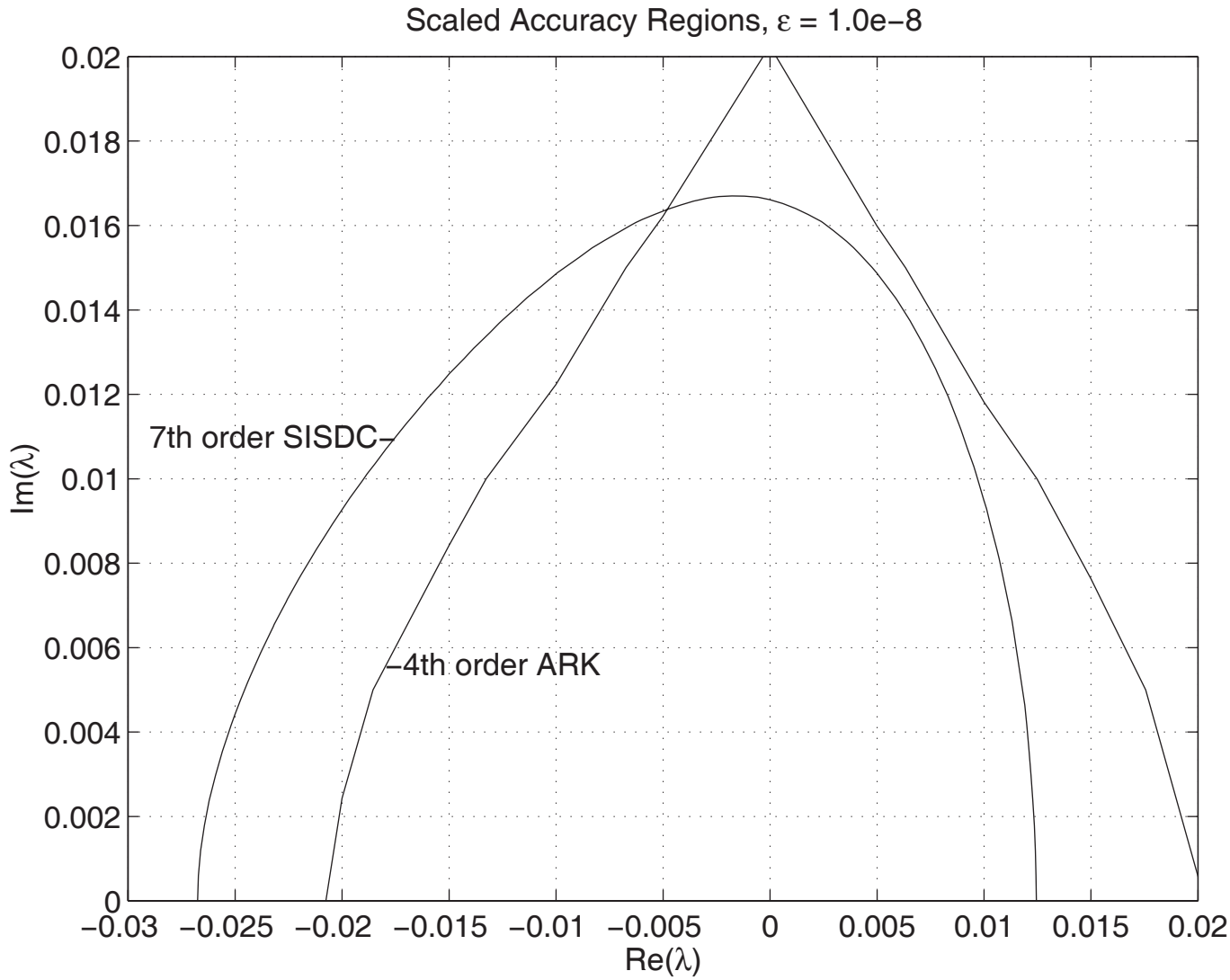

FIG. 6. Scaled accuracy regions for error tolerance 10^{-8} for the $SISDC_7^7$ method and the fourth-order (semi-implicit) Additive Runge-Kutta method of Kennedy and Carpenter.

The pertinent point is that higher-order SISDC methods are more efficient than lower-order SISDC or additive Runge-Kutta methods when the solution is required

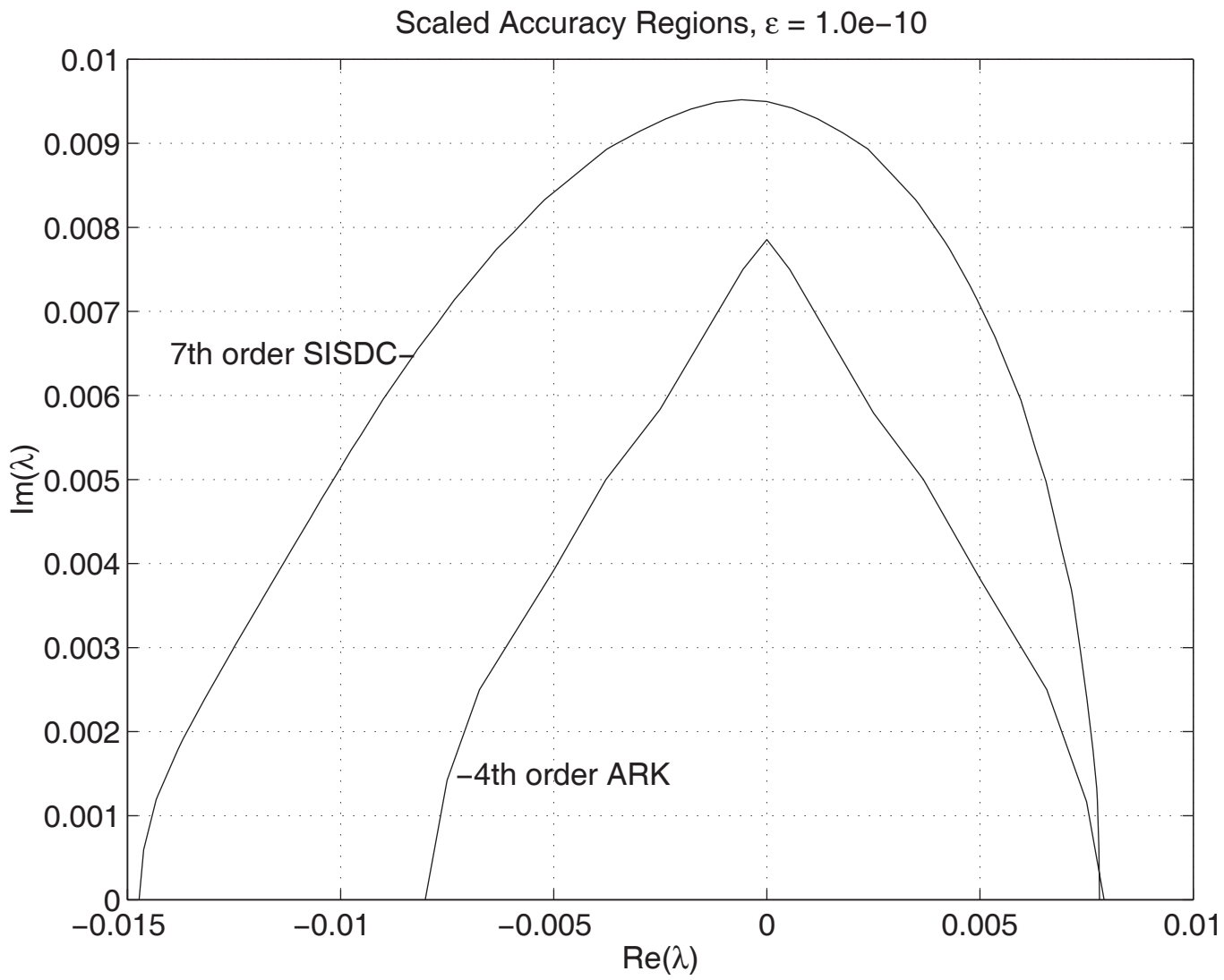

FIG. 7. Scaled accuracy regions for error tolerance 10^{-10} for the $SISDC_7^7$ method and the fourth-order (semi-implicit) Additive Runge-Kutta method of Kennedy and Carpenter.

to be computed very accurately. The threshold at which higher-order methods become more efficient is, of course, problem dependent. For example, in the stiff Van der Pol's equation test problem presented in Sect. 5, the ARK method is more efficient in terms of accuracy per function evaluation than the SISDC method for the size of time steps considered, although one could reasonably conclude from the error plots that the opposite would be true if greater accuracy were required.

4.2. Ladder Methods

It is possible to reduce the computational cost of a single step of the SISDC methods presented above without reducing the *order* of the overall method. One strategy for doing this is based on the observation that the kth correction equation only requires that the solution up to that point have truncation error of size $O(\Delta t^{k+2})$. Therefore, it is possible to reduce the number of substeps used to compute the solution when k is small. In the following discussion, iterations corresponding to k small will be referred to generically as the *lower-order iterations*. The stability and accuracy of the resultant method will of course be affected. In the context of using SISDC methods for PDEs, it may be the case that the solution is well resolved in time (and presumably the time-step is restricted by spatial resolution). In this case, reducing the temporal resolution of the lower-order iterations is justified.

As an example, consider the fourth-order $SISDC_4^4$ method. Rather than using three substeps per iteration, it is sufficient to use only one for the initial iteration, then two for the following two iterations, and three for the last, yielding a total of eight rather than twelve total substeps. As is shown in Fig. 8, the fourth-order ladder method has a scaled accuracy region of similar size as that of $SISDC_4^4$.

The scaled accuracy regions for a of variety of possible ladder methods of various order have been compared to those of the corresponding $SISDC_K^K$ methods. No variation has been found that produces a substantially more efficient method although the possibility certainly exists. It is possible that in the context of PDEs, more efficient methods could be developed by reducing both the spatial and temporal resolution for lower-order iterations. This will be addressed in future work.

5. NUMERICAL EXAMPLES

To evaluate the performance of the SISDC methods, versions varying in order from three through seven are tested on two different sets of problems. In the first set of tests, the formal order of accuracy as well as the accuracy for stiff problems is demonstrated using Van der Pol's equation. This common test problem is chosen so that a comparison in terms of accuracy per function evaluation between SISDC methods and other existing methods can be made. In the second set of tests, advection-diffusion type PDEs are approximated using the SISDC methods and MOL.

5.1. Van der Pol's Equation

Van der Pol's equation is a popular nonlinear test problem for methods for stiff ODEs. The equation prescribes the motion of a particle $x(t)$ by

$$x''(t) + \mu(1 - x(t)^2)x'(t) + x(t) = 0.$$

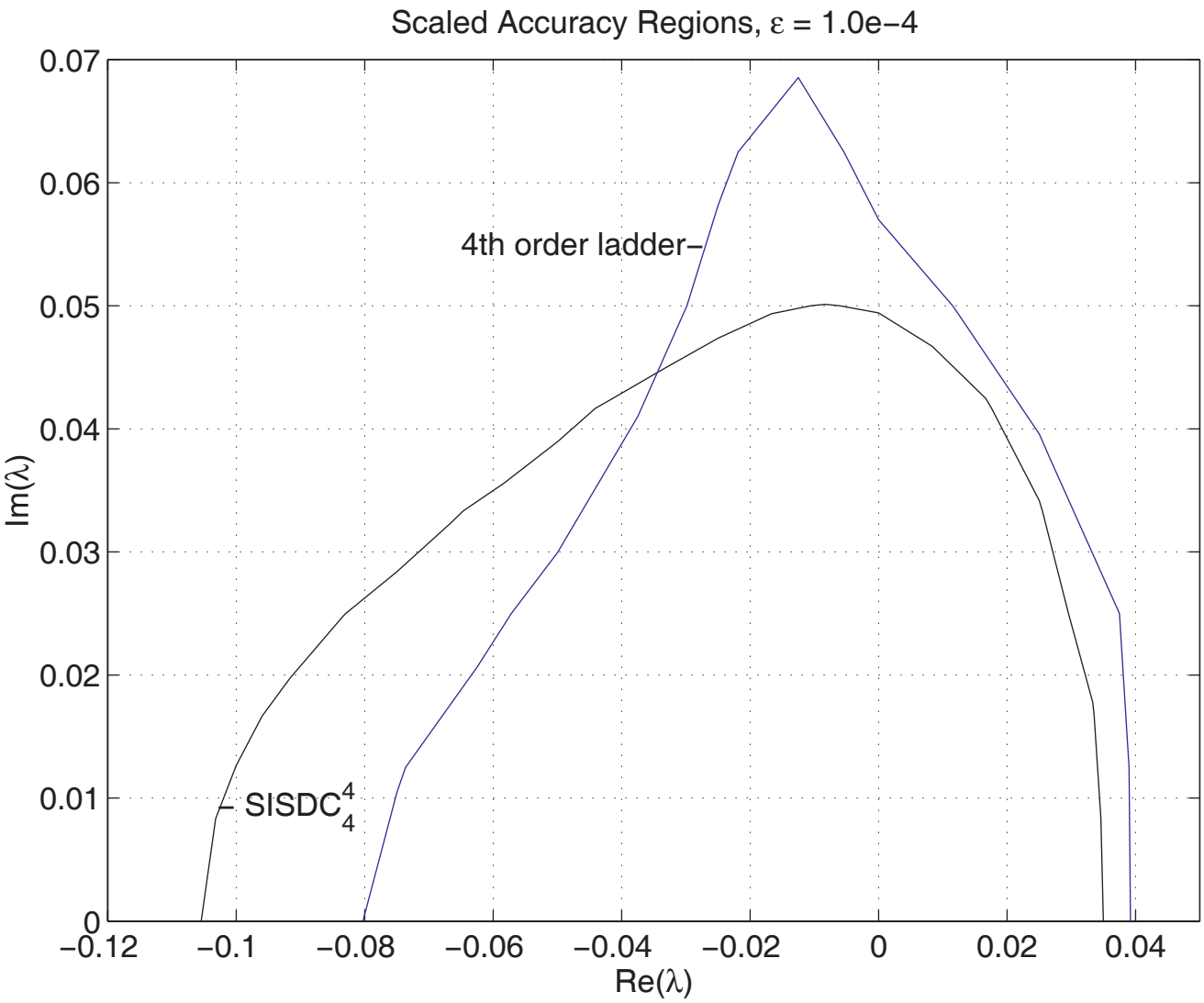

FIG. 8. Scaled accuracy regions for error tolerance 10^{-4} for the $SISDC_4^4$ method and a fourth-order ladder method requiring only eight function evaluations.

Making the usual transformation, $y_1(t) = x(t)$, $y_2(t) = \mu x'(t)$, and $t = t/\mu$ yields the system

$$\begin{aligned} y_1' &= y_2 \\ y_2' &= (-y_1 + (1 - y_1^2)y_2)/\epsilon, \end{aligned}$$

where $\epsilon = 1/\mu^2$. As ϵ approaches zero, these equations become increasingly stiff. For the SISDC methods, the first equation is treated explicitly, and the second implicitly.

The first example is designed to demonstrate that the SISDC methods do exhibit the correct convergence behavior on a nontrivial problem. The SISDC methods of order three through seven are tested on Van der Pol's equation with $\epsilon = 1$. In this case, the equation is not stiff, and the solution is smooth. Initial conditions for each case are given by $y_1(0) = 2$ and $y_2(0) = 2/3$ (i.e. $y_2'(0) = 0$), and the solution is computed to time $t = 4$, with a fixed time step.

To compute numerical errors for this problem, a reference solution is first computed using $SISDC_7^7$ and 1024 time steps. Errors in a given solution are then computed by interpolating values of the reference solution. Unlike many Runge-Kutta methods, intermediate values in the SISDC method have full order accuracy, hence the value of the reference solution at every *substep* is used in the interpolation. An approximation to the L_1 error in time is computed by integrating the absolute difference between the reference solution and each particular numerical solution. Again intermediate values (which are conveniently located at the Gauss-Lobatto integration nodes) are used in the integration.

The error in the second components of the solution is displayed in log coordinates in Fig. 9 as a function of the number of function evaluations (which is $K(K-1)$ times the number of times steps for the Kth order method). The corresponding plot for the first component of the solution is virtually identical, and is hence omitted. For each order, a line corresponding to the expected convergence rate is superimposed. In each case, the convergence of the numerical solution approaches the expected value as Δt approaches zero. Note as well, that except for larger values of Δt, the number of function evaluations required to attain a given accuracy decreases as the order of the method increases. In other words, when high precision is desired, higher-order versions of the method are more efficient than those of lower order.

5.2. Stiff Equations and Order Reduction

Since the SISDC methods were motivated by the need to solve problems with both stiff and non-stiff parts, the above experiment is of limited significance. Therefore, the above experiment is repeated for a mildly stiff case with $\epsilon = 10^{-1}$ and a stiff case $\epsilon = 10^{-3}$. To enable a direct comparison with the ARK methods in [19], the initial conditions are set to $y(1) = 2$, $y(2) = -0.6666654321121172$, and solutions are computed only to time $t = 0.5$. Reference solutions are computed using a 7th order fully implicit SDC integrator, and errors are computed by simply comparing the absolute error at $t = 0.5$.

The results for $\epsilon = 10^{-1}$ are shown in Figs. 10 and 11. As in the non-stiff case, the convergence rate of each version approaches the expected value as Δt approaches zero, although in this case, the observed convergence rate decreases if only larger

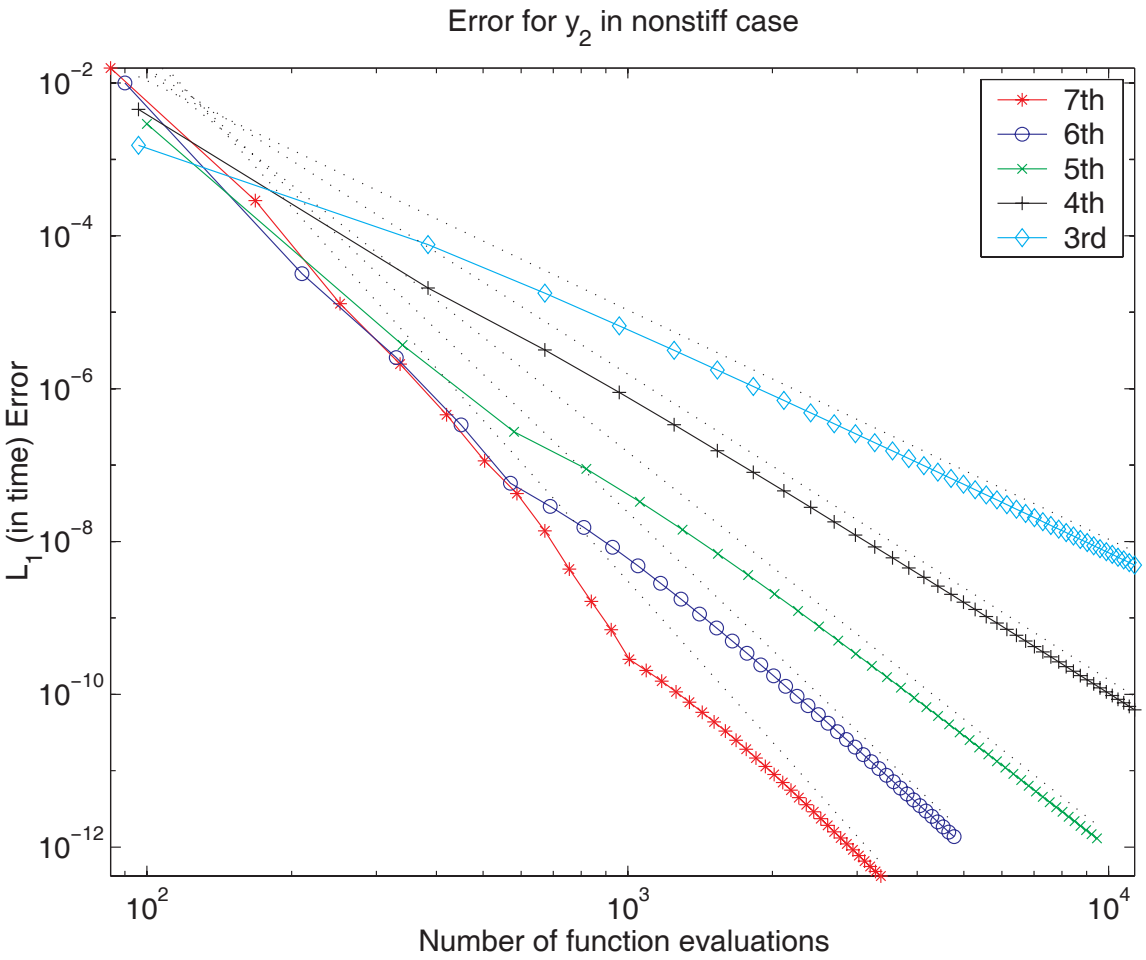

FIG. 9. Errors for the Van der Pol equation with $\epsilon = 1$ computed using $SISDC_K^K$ methods for K ranging from 3 through 7.

values of Δt are considered. The dramatic dips in the graphs correspond to certain values at which the error at $t = 0.5$ as a function of Δt changes sign. These dips are therefore not a true indication of increased accuracy in the method. In this example, higher order methods are again more efficient when higher precision is required. In terms of accuracy per function evaluation, the higher order methods compare favorably with the methods in [19].

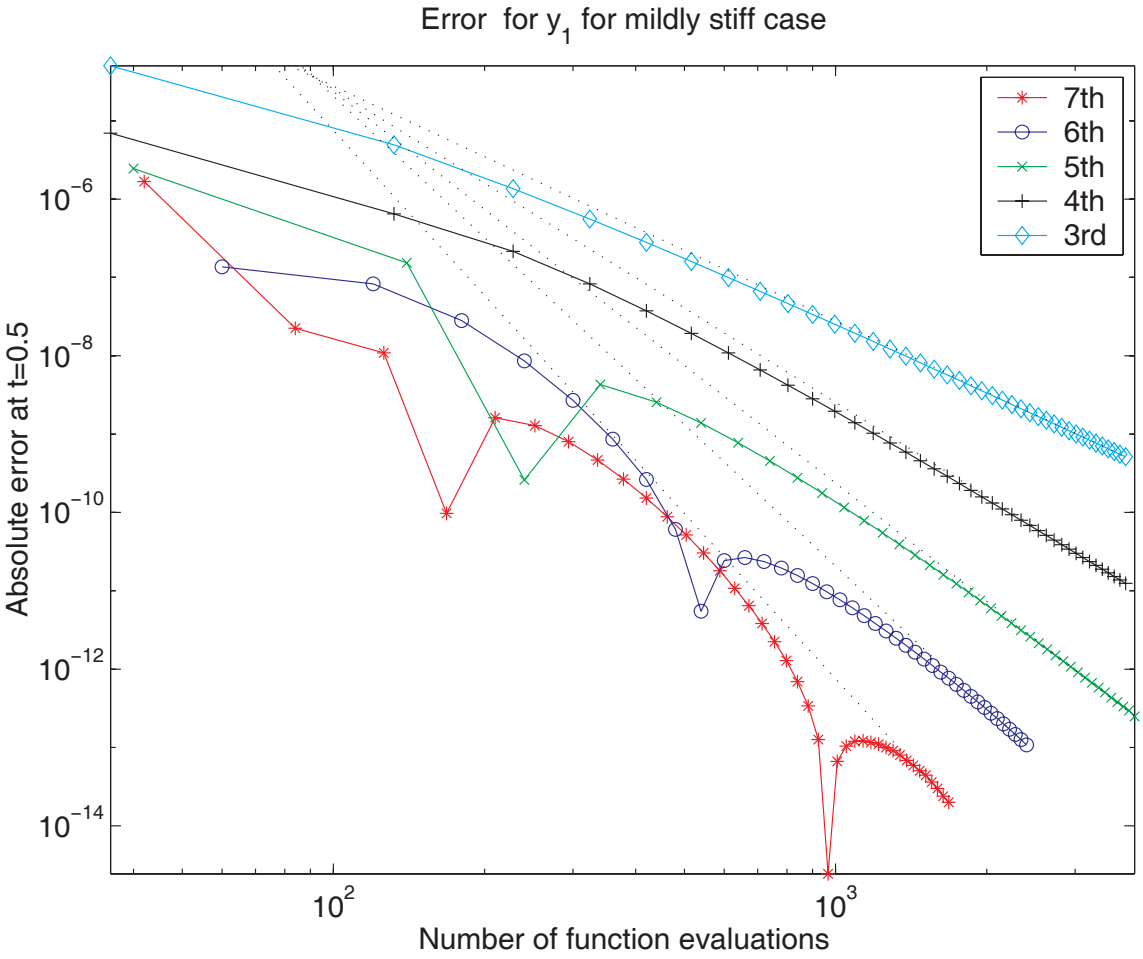

FIG. 10. Errors for the first component of the Van der Pol equation with $\epsilon = 10^{-1}$ computed using $SISDC_K^K$ methods for K ranging from 3 through 7.

The results for $\epsilon = 10^{-3}$ are shown in Figs. 12 and 13. In this case, the convergence behavior of the methods in the range of Δt displayed differs considerably from the formal order. This phenomenon, known as *order reduction*, occurs in Runge-Kutta and other methods as well [17, 24, 19]. Also in this range, higher-order versions are not significantly more efficient than those with lower order. It is reasonable to assume that the correct asymptotic convergence rates would be observed given sufficiently small Δt; however, this is not the relevant issue in most applications.

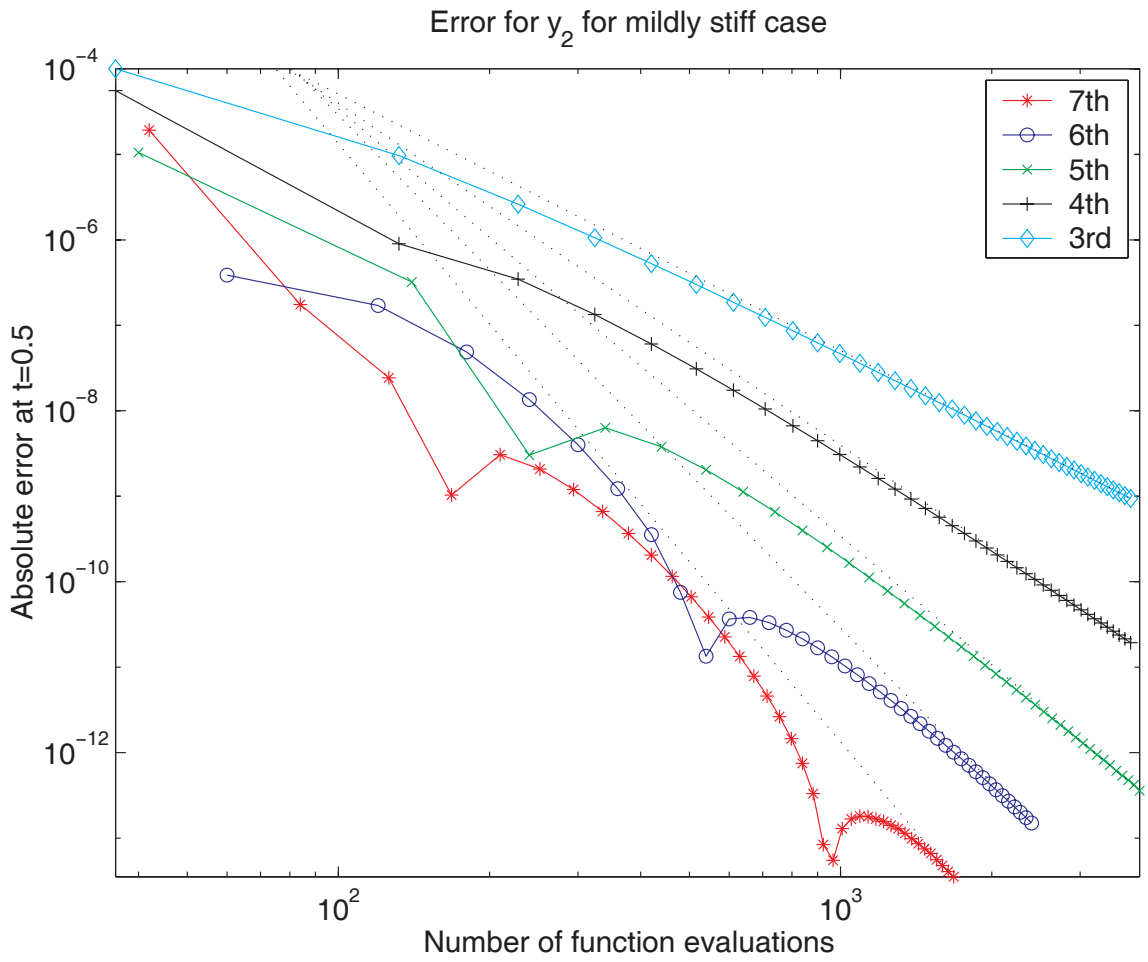

FIG. 11. Errors for the second component of the Van der Pol equation with $\epsilon = 10^{-1}$ computed using $SISDC_K^K$ methods for K ranging from 3 through 7.

Comparing these results with those from the additive Runge-Kutta methods in [19], suggests that the higher-order SISDC methods are more efficient for mildly stiff problems when high precision is desired. In very stiff cases, the comparison is not so clear. In particular, the SISDC methods do not achieve the same absolute accuracy in the first component of the solution as the ARK methods in the range of the figures in [19].

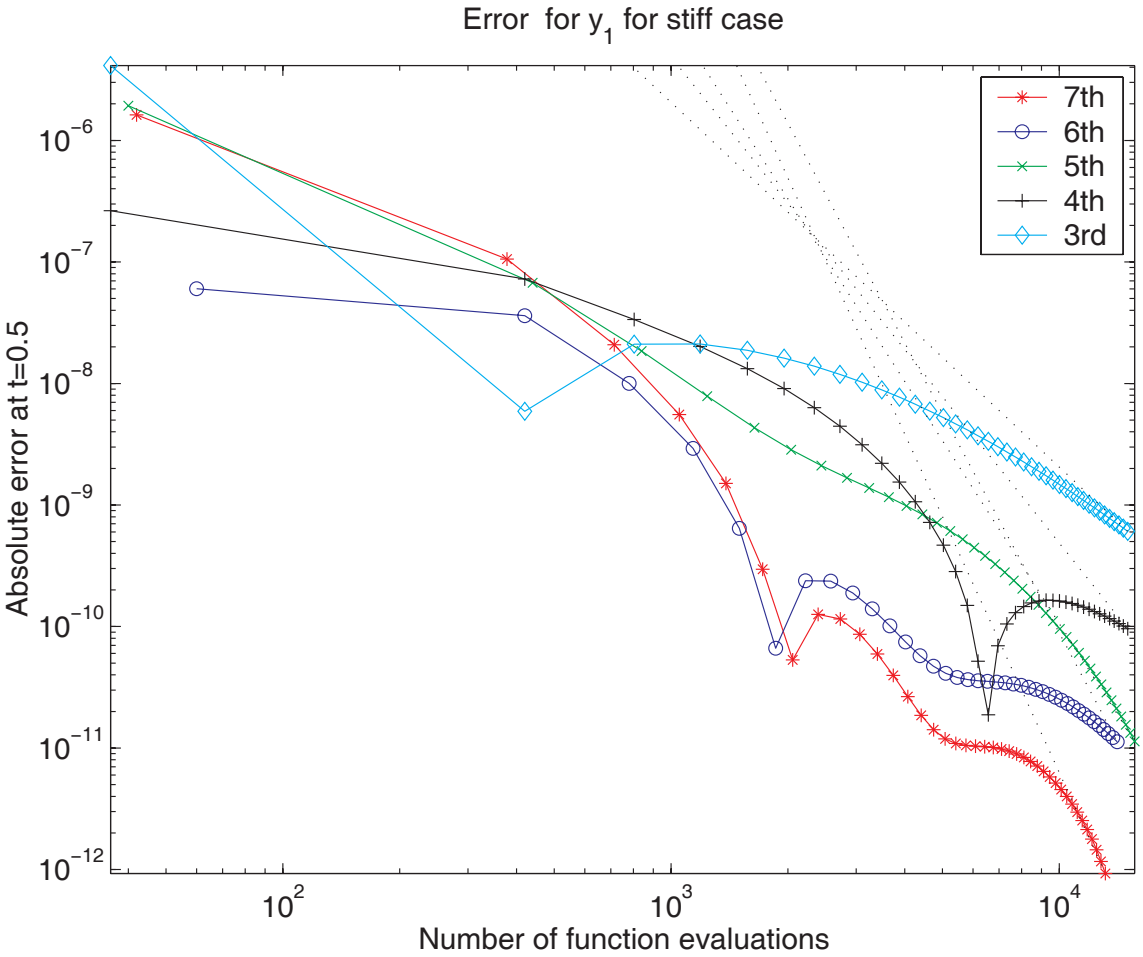

FIG. 12. Errors for the first component of the Van der Pol equation with $\epsilon = 10^{-3}$ computed using $SISDC_K^K$ methods for K ranging from 3 through 7.

Ideally for very stiff problems, one would like to integrate the stiff part of the equation using a smaller time step than the non-stiff part. This approach is possible in the SISDC context and is discussed in [6].

5.3. Method of Lines with SISDC

The motivation behind the development of SISDC methods is for the solution of PDEs with both stiff and non-stiff terms. Therefore in this section, numerical examples of simple advection diffusion equations are presented. Given the equation

$$u_t = F_E(u) + F_I(u), \tag{16}$$

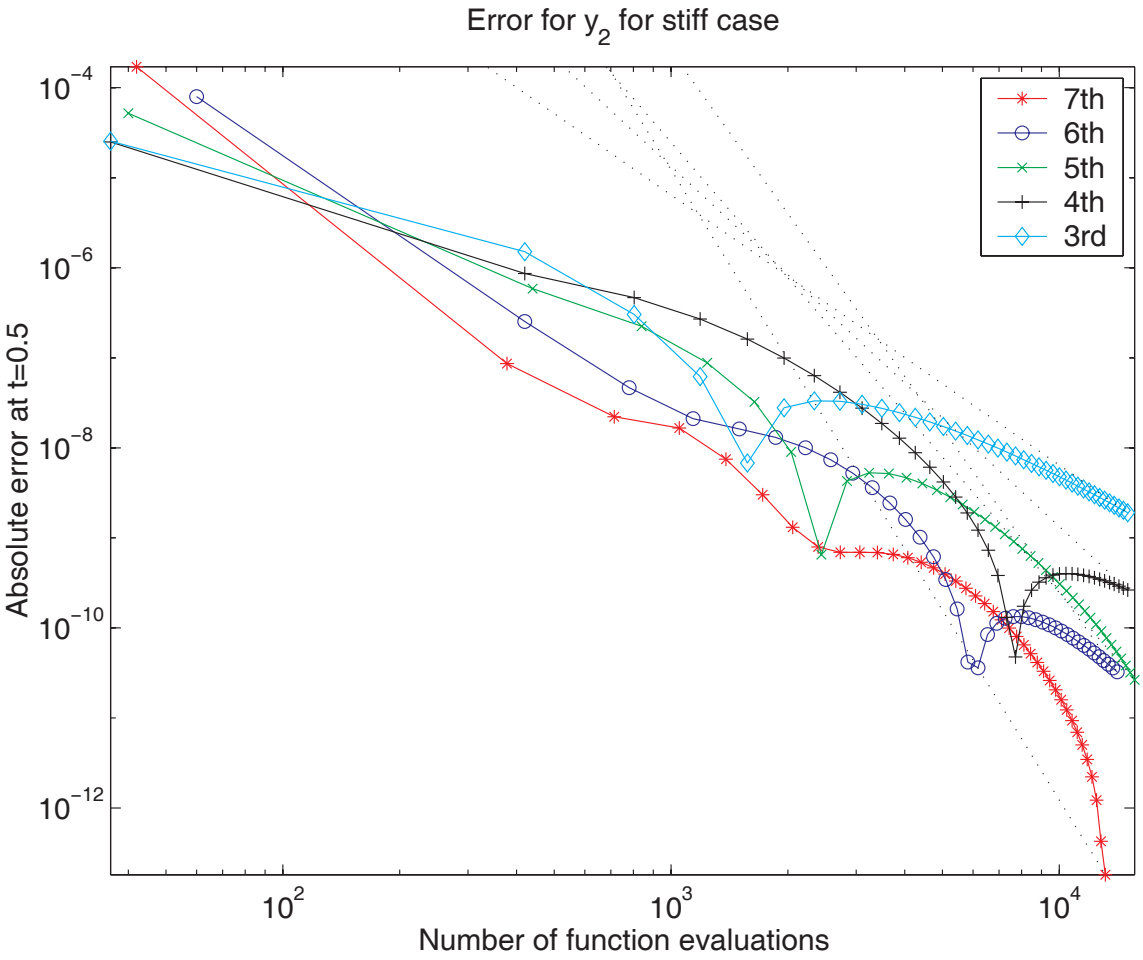

FIG. 13. Errors for the second component of the Van der Pol equation with $\epsilon = 10^{-3}$ computed using $SISDC_K^K$ methods for K ranging from 3 through 7.

where F_E and F_I are functions of u and its spatial derivatives, the SISDC method can be used with the MOL to yield higher-order, semi-implicit methods.

To illustrate this technique, consider the advection diffusion equation

$$u_t = a(t)u_x + d(t)u_{xx}. \tag{17}$$

For simplicity, let the spatial domain be the unit line $[0, 1]$ with periodic boundary conditions.

For a given grid spacing Δx let $x_i = i\Delta x$, and let U_i^n denote the numerical approximation to the solution $u(x_i, t_n)$. Following the usual custom, indices are omitted when the value is obvious. If DU_i is a finite-difference approximation to $u_x(x_i)$, and LU_i an approximation to $u_{xx}(x_i)$, a simple first order method for Eq. (17) is

$$U^{m+1} = U^m + \Delta t_m(a(t_m)DU^m + d(t_{m+1})LU^{m+1}). \tag{18}$$

Eq. (18) is an implicit equation for U^{m+1} which requires the inversion of the linear equation

$$(I - d(t_{m+1})\Delta t_m L)U^{m+1} = U^m + \Delta t_m a(t_m)DU^m. \tag{19}$$

Techniques for solving this equation, as well as many variations of it, are well established. In particular, efficient integral equation methods with up to eighth-order spatial accuracy have recently been developed [18, 14].

In the context of SISDC, a similar equation for the correction must be solved. Specifically, the direct form of Eq. (11) becomes

$$\begin{aligned}(I - d(t_{m+1})\Delta t_m L)U^{m+1,k+1} &= U^{m,k+1} + \Delta t_m[a(t_m)(DU^{m,k+1} - DU^{m,k}) \\ &- d(t_{m+1})LU^{m+1,k}] - I_m^{m+1}(U_i^k).\end{aligned} \tag{20}$$

Here, the last term of the equation is an approximation to the time integral of the right hand side of Eq. (17) given by the appropriate quadrature rule (see Eq. (13))

$$I_m^{m+1}(U_i^k) = \sum_{l=0}^{p} q_m^l(a(t_l)DU_i^l + d(t_l)LU_i^l).$$

In all of the above discussion, the issue of prescribing the boundary conditions for Eqs. (19) and (20) has been ignored. When Dirichlet or Neumann boundary conditions are imposed at the domain boundary, care must be taken when determining the boundary conditions imposed during the SISDC process. In particular, the exact prescribed boundary conditions cannot be imposed when solving the correction equation (except for the final correction) without a resulting reduction in the order of accuracy. This phenomenon is similar to that which occurs with Runge-Kutta methods [15, 20, 26, 11, 1, 25]. An important difference in the SISDC methods is that the intermediate solutions are only used to construct the next correction equation. This is unlike Runge-Kutta methods where the final solution is typically a linear combination of the intermediate function values. A detailed discussion of boundary conditions for the MOL as well as a general strategy for avoiding the loss of accuracy for SISDC methods is presented in [22].

The SISDC method is combined with the MOL approach and applied to Eq. (17) with $a(t) = 1+\cos(5\pi t)$, $d(t) = \nu(3-\sin(7\pi t))/4$. The initial condition is $u(x,0) = \cos(2\pi x)$, and periodic boundary conditions are enforced, hence the exact solution is

$$u(x,t) = e^{-\pi^2\nu(3t+\cos(7\pi t)/7\pi)}\cos(2\pi(x-t-\sin(5\pi)/5\pi)).$$

The functions $a(t)$ and $d(t)$ are chosen to oscillate rapidly in time in order to emphasize the temporal error in the numerical solution.

Two different values of ν are considered, the mildly stiff case $\nu = 0.01$ and the stiff case $\nu = 0.25$. For each case the operators D and L are sixth-order centered-difference operators, and the implicit equation is solved via the FFT.

The $SISDC_K^K$ method is used for the numerical tests with $K = 3$ to 5, and the solution is computed to time $t = 1.0$ using a time step $\Delta t = 4\Delta x$. Errors in the L_∞ norm are plotted versus grid spacing in Figs. 14 and 15 along with lines with slopes corresponding to 3rd, 4th, and 5th order convergence. In the mildly stiff case, the convergence rates are extremely close to the expected values. For the stiff case, the rates approach the expected values as Δt is reduced. It is important to remember that the real part of the eigenvalues of the linear system of ODEs resulting from the discretization scales like $-\nu/\Delta x^2$, so as Δt approaches zero, the equation becomes increasingly stiff. For the finest run, the ratio $\nu\Delta t/\Delta x^2$ is greater than 350.

A similar test is performed using a non-linear equation, specifically Burgers equation

$$u_t + uu_x = \nu u_{xx}. \tag{21}$$

The initial conditions are $u(x,0) = 1 + 0.5\cos(2\pi x)$, $\nu = 0.02$, and periodic boundary conditions are again used. A solution computed using $SISDC_7^7$ and 1024 grid points is used as a reference solution, and the reported errors are calculated by comparison with this solution at time $t = 1.0$. The L_∞ errors are displayed in Fig. 16 for grid sizes ranging $\Delta x = 1/64$ to $1/352$ and $\Delta t = 4\Delta x$. Lines with slopes corresponding to 3rd, 4th, and 5th order convergence are superimposed and confirm the expected convergence.

6. CONCLUSION

In this paper, a semi-implicit version of the method of spectral deferred corrections for ordinary differential equations is presented. The methods are motivated by the desire to design higher-order methods for partial differential equations with both stiff and non-stiff terms. Numerical tests suggest that in cases that are not extremely stiff, higher-order SISDC methods are as efficient or more efficient than recent semi-implicit Runge-Kutta methods. For equations with vastly varying time scales more elaborate time-splitting methods using different sized time-steps may prove more effective. For example, in [6], it is shown that using smaller time steps for the reaction piece in a multi-implicit SDC method for advection-diffusion-reaction equations with very stiff reaction terms increases the overall efficiency of the method.

The SISDC methods presented here have already been combined with the method of lines approach for PDEs. In [21], fourth-order projection methods for the incom-

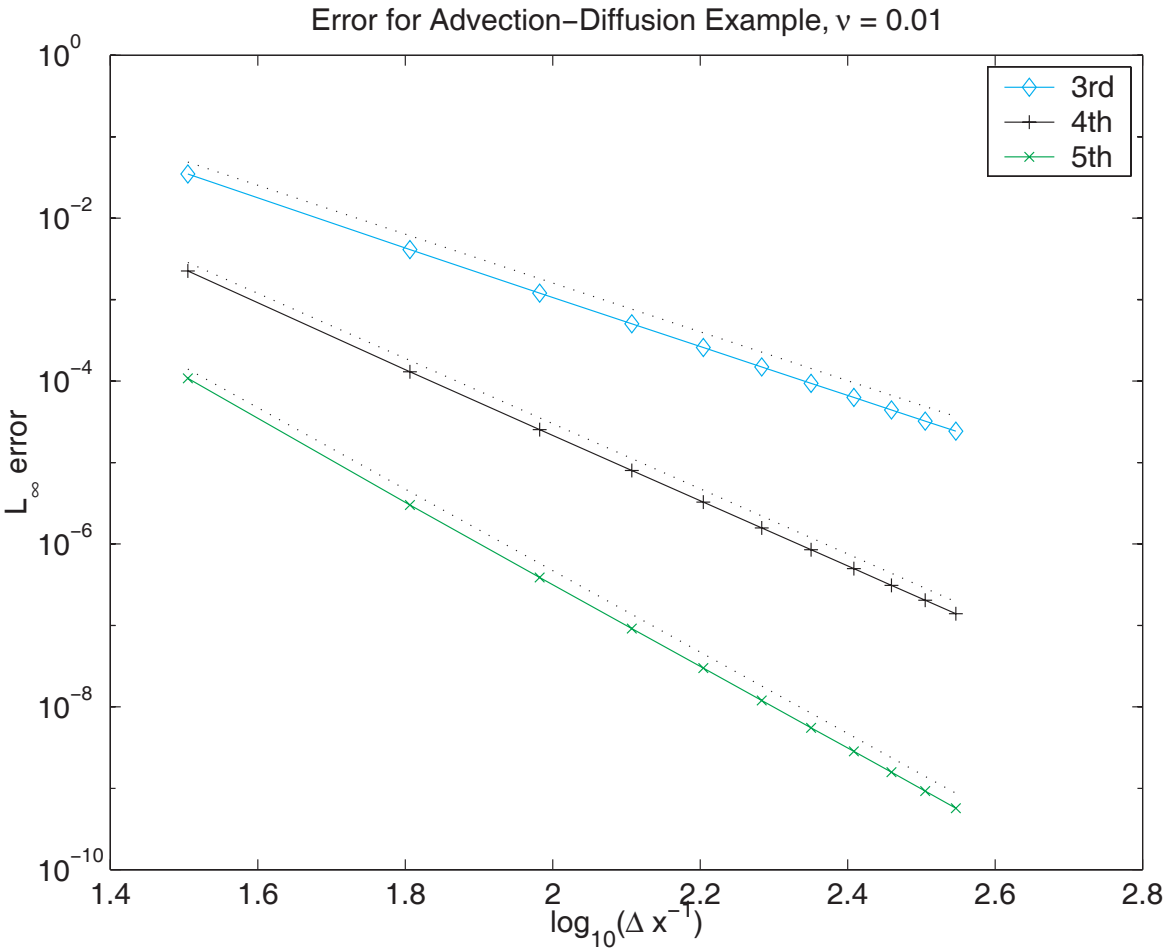

FIG. 14. Errors for the advection diffusion equation with $\nu = 0.01$ using $SISDC_K^K$ methods for K ranging from 3 through 5.

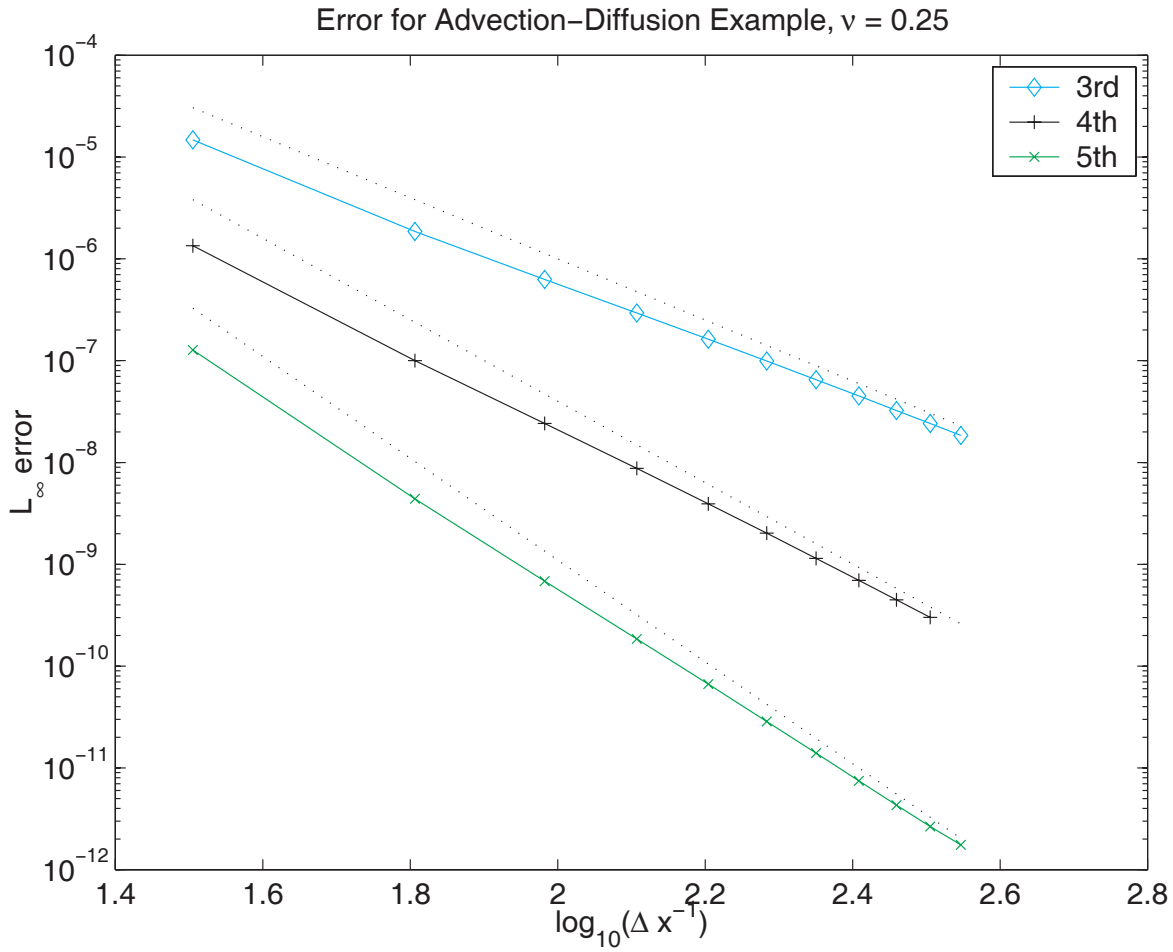

FIG. 15. Errors for the advection diffusion equation with $\nu = 0.25$ using $SISDC_K^K$ methods for K ranging from 3 through 5.

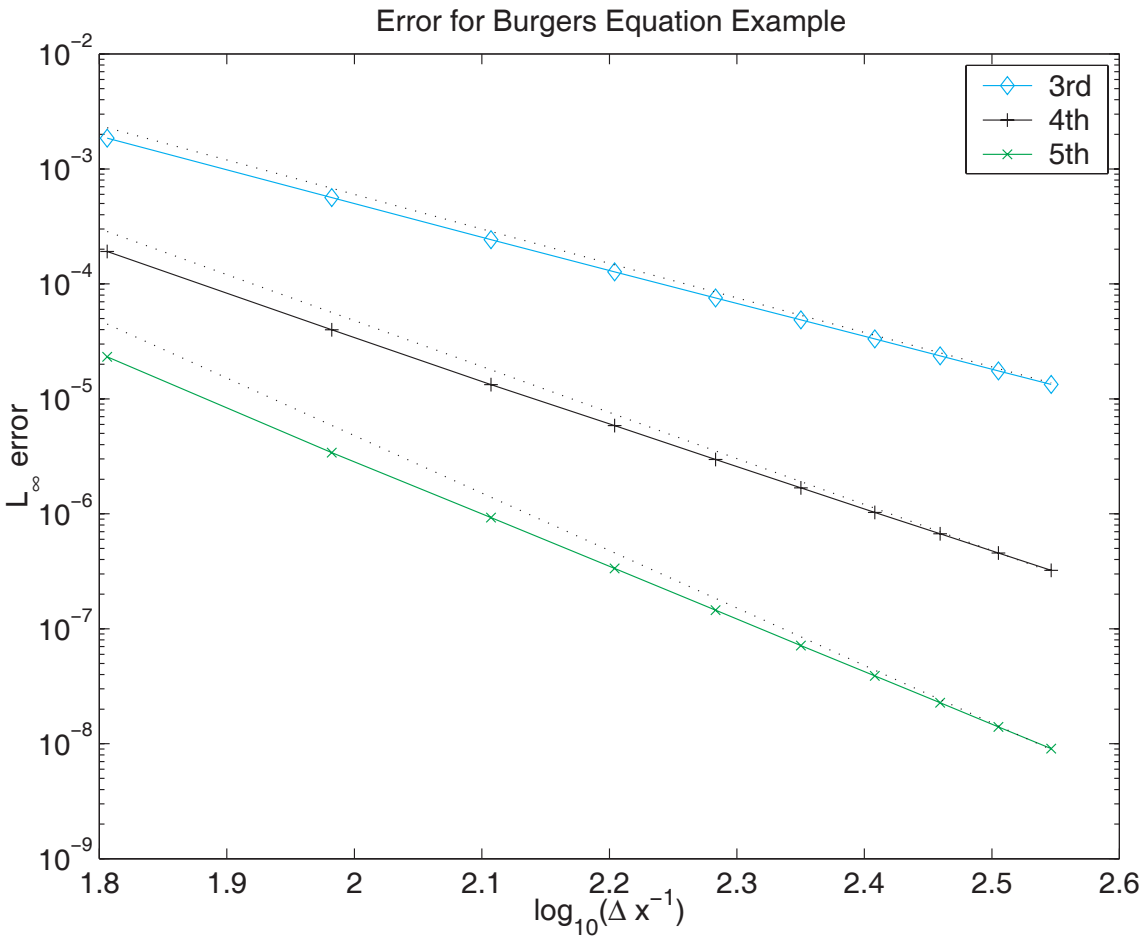

FIG. 16. Errors for the Burgers equation example using $SISDC_K^K$ methods for K ranging from 3 through 5.

pressible Navier-Stokes equations are presented, and numerical examples set in two-dimensional periodic domain are included there. The imposition of boundary conditions for semi-implicit projection methods of any variety has been a controversial subject virtually since projection methods were introduced. The relationship between boundary conditions and the accuracy of the pressure is examined in [7], and a project incorporating this analysis with SISDC methods to create higher-order semi-implicit projection method for flows with material boundaries is underway. Other applications involving the modeling of immersed boundaries in incompressible flows, advection-diffusion-reaction equations, and low-Mach number combustion are also being pursued.

REFERENCES

1. Saul Abarbanel, David Gottlieb, and Mark H. Carpenter. On the removal of boundary errors caused by Runge-Kutta integration of nonlinear partial differential equations. *SIAM J. Sci. Comput.*, 17:777–782, 1996.
2. Georgios Akrivis, Michel Crouzeix, and Charalambos Makridakis. Implicit-explicit multistep methods for quasilinear parabolic equations. *Num. Math.*, 82:521–541, 1999.
3. Uri M. Ascher and Linda R. Petzold. *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations.* SIAM, Philadelphia, PA, 2000.
4. Uri M. Ascher, Steven J. Ruuth, and Raymond J. Spiteri. Implicit-explicit Runge-Kutta methods for time-dependent partial differential equations. *Appl. Numer. Math.*, 25:151–167, 1997.
5. Uri M. Ascher, Steven J. Ruuth, and Brian T. R. Wetton. Implicit-explicit methods for time-dependent PDE's. *SIAM J. Numer. Anal.*, 32:797–823, 1995.
6. A. Bourlioux, A. T. Layton, and M. Minion. Higher-order multi-implicit spectral deferred correction methods for problems of reacting flow. *J. Comput. Phys.*, 2002. submitted.
7. D. L. Brown, R. Cortez, and M. Minion. Accurate projection methods for the incompressible Navier-Stokes equations. *J. Comput. Phys.*, 168, Apr. 2001.
8. J. C. Butcher. *The Numerical Analysis of Ordinary Differential Equations: Runge-Kutta and General Linear Methods.* John Wiley and Sons, Chichester, 1987.
9. M. P. Calvo, J. de Frutos, and J. Novo. Linearly implicit Runge-Kutta methods for advection-reaction-diffusion equations. *Appl. Numer. Math.*, 37:535–549, 2001.
10. M. P. Calvo and C. Palencia. Avoiding the order reduction of Runge-Kutta methods for linear initial boundary value problems. *Math. Comp.*, 2001. to appear.
11. Mark H. Carpenter, David Gottlieb, Saul Abarbanel, and Wai-Sun Don. The theoretical accuracy of Runge-Kutta time discretization for the initial boundary value problem: A study of the boundary error. *SIAM J. Sci. Comput.*, 16:1241–1252, 1995.
12. Alok Dutt, Leslie Greengard, and Vladimir Rokhlin. Spectral deferred correction methods for ordinary differential equations. *BIT*, 40(2):241–266, 2000.
13. J. Frank, W. H. Hundsdorder, and J. G. Verwer. Stability of implicit-explicit linear multistep methods. *Appl. Numer. Math.*, 25:193–205, 1997.
14. L. Greengard and J. F. Huang. A new version of the fast multipole method for screened Coulomb interactions in three dimensions. *J. Comput. Phys.*, 2002. submitted.
15. Bertil Gustafsson. The convergence rate for difference approximations to mixed initial boundary value problems. *Math. Comp.*, 29:396–406, 1975.
16. Ernst Hairer, Syvert Paul Norsett, and Gerhard Wanner. *Solving Ordinary Differential Equations I, Nonstiff Problems.* Springer-Verlag, Berlin, 1987.
17. Ernst Hairer and Gerhard Wanner. *Solving Ordinary Differential Equations II, Stiff and Differential-Algebraic Problems.* Springer-Verlag, Berlin, 1991.
18. J. Huang, L. Greengard, and F. Ethridge. A new fast-multipole accelerated Yukawa solver in two dimensions. 2001. In preparation.

19. Christopher A. Kennedy and Mark H. Carpenter. Additive Runge-Kutta schemes for convection-diffusion-reaction equations. *Appl. Numer. Math.*, 2001. Submitted.

20. R. J. LeVeque and J. Oliger. Numerical methods based on additive splittings for hyperbolic partial differential equations. *J. Comput. Phys.*, 40(162):469–497, 1983.

21. M. L. Minion. Higher-order semi-implicit projection methods. In *Numerical Simulations of Incompressible Flows: Proceedings of a conference held at Half Moon Bay, CA, June 18-20, 2001*, June 2001. LLNL unclassified report UCRL-JC-145295.

22. M. L. Minion. Higher-order semi-implicit methods for initial boundary value partial differential equations. 2002. In preparation.

23. Alexander Ostermann. Stability of W-methods with applications to operator splitting and to geometric theory. *Appl. Numer. Math.*, 2001. To appear.

24. Lorenzo Pareshi and Giovanni Russo. *Implicit-Explicit Runge-Kutta schemes for stiff systems of differential equations*, volume 3, pages 269–287. Nova Science, 2000.

25. D. Pathria. The correct formulation of intermediate boundary condidtions for Runge-Kutta time integration of intitial boundary value problems. *SIAM J. Sci. Comput.*, 18(5):1255–1266, 1997.

26. J. M. Sans-Serna, J. G. Verwer, and W. H. Hundsdorfer. Convergence and order reduction of Runge-Kutta schemes applied to evolutionary problems in partial differential equations. *Num. Math.*, 50:405–418, 1986.

27. J. W. Shen and X. Zhong. Semi-implicit Runge-Kutta schemes for the non-autonomous differential equations in reactive flow computations. In *Proceedings of the 27th AIAA Fluid Dynamics Conference*. AIAA, June 1996.

28. R. D. Skeel. A theoretical framework for proving accuracy results for deferred corrections. *SIAM J. Numer. Anal.*, 19(1):171–196, 1981.