Journal of Computational Physics 294 (2015) 224–242

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

High order operator splitting methods based on an integral deferred correction framework

Andrew J. Christlieb [a,b], Yuan Liu [a,*], Zhengfu Xu [c]

[a] Department of Mathematics, Michigan State University, East Lansing, MI 48824, USA
[b] Department of Electrical and Computer Engineering, Michigan State University, East Lansing, MI 48824, USA
[c] Department of Mathematical Science, Michigan Technological University, Houghton, MI 49931, USA

ARTICLE INFO

Article history:
Received 2 July 2014
Accepted 18 March 2015
Available online 31 March 2015

Keywords:
Integral deferred correction
Initial–boundary value problem
High-order accuracy
Operator splitting

ABSTRACT

Integral deferred correction (IDC) methods have been shown to be an efficient way to achieve arbitrary high order accuracy and possess good stability properties. In this paper, we construct high order operator splitting schemes using the IDC procedure to solve initial value problems (IVPs). We present analysis to show that the IDC methods can correct for both the splitting and numerical errors, lifting the order of accuracy by r with each correction, where r is the order of accuracy of the method used to solve the correction equation. We further apply this framework to solve partial differential equations (PDEs). Numerical examples in two dimensions of linear and nonlinear initial–boundary value problems are presented to demonstrate the performance of the proposed IDC approach.

1. Introduction

In this paper we present high order operator splitting methods based on the integral deferred correction (IDC) mechanism. The methods are designed to leverage recent progress on parallel time stepping and offer a great deal of flexibility for computing the ordinary differential equations (ODEs). We focus on extending IDC theory to the case of splitting schemes on the IVP

$$u_t = f(t,u) = \sum_{\nu=1}^{\Lambda} f_\nu(t,u), \qquad u(0) = u_0, \qquad t \in [0,T], \tag{1.1}$$

and discuss the application in parabolic PDEs. Here, $u \in \mathbb{R}^n$ and $f(t,u) : \mathbb{R}^+ \times \mathbb{R}^n \to \mathbb{R}^n$.

In the case that (1.1) arises from a method of lines discretization of time dependent PDEs which describe multi-physics problems, we encounter high dimensional computation. For these problems, the splitting methods can be applied to decouple the problems into simpler sub-problems. Therefore, the main advantages of operator splitting methods are problem simplification, dimension reduction, and lower computational cost. Two broad categories can classify many splitting methods: differential operator splitting [1,25,32,33] and algebraic splitting with the prominent example of the alternating direction implicit (ADI) method, which was first introduced in [9,7,30] for solving two dimensional heat equations. The main barrier in designing high order numerical methods based on the idea of splitting is the operator splitting error. To obtain high order accuracy via low order splitting method generally adds complexity to designing a scheme and stability

* Corresponding author.
E-mail addresses: christli@msu.edu (A.J. Christlieb), yliu7@math.msu.edu (Y. Liu), zhengfux@mtu.edu (Z. Xu).

http://dx.doi.org/10.1016/j.jcp.2015.03.032
0021-9991/

analysis [14,27,18,26,34,12]. A recent work in [3] utilizes the spectral deferred correction (SDC) procedure to the advection–diffusion–reaction system in one dimension in order to enhance the overall order of accuracy. However, this work does not contain a proof that the corrections raise the order of the method.

In [10], an SDC method is first proposed as a new variation on the classical deferred correction methods [2]. The key idea is to recast the error equation such that the residual appears in the error equation in integral form instead of differential form, which greatly stabilizes the method. It is proposed as a framework to generate arbitrarily high order methods. This family of methods use Gaussian quadrature nodes in the correction to the defect or error, hence the method can achieve a maximal order of $2(M-1)$ on M grid points with $2(M-2)$ corrections. This main feature of the SDC method made it popular and extensive investigation can be found in [10,28,21,23,22,15,16,24]. Following this line of approach, the IDC methods are introduced in [6,5,4]. High order explicit and implicit Runge–Kutta (RK) integrators in both the prediction and correction steps (IDC-RK) are developed by utilizing uniform quadrature nodes for computing the residual. In [6,5], it is established that using explicit RK methods of order r in the correction step results in r higher degrees of accuracy with each successive correction step, but only if uniform nodes are used instead of the Gaussian quadrature nodes of SDC. It is shown in [5] that the new methods produced by the IDC procedure are yet again RK methods. It is also demonstrated that, for the same order, IDC-RK methods possess better stability properties than the equivalent SDC methods. Furthermore, for explicit methods, each correction of IDC or SDC increases the region of absolute stability. Similar results are generalized to arbitrary order implicit and additive RK methods in [4]. Generally, for *implicit* methods based on IDC and SDC, the stability region becomes smaller when more correction steps are employed. It is believed that this is due to the numerical approximation of the residual integral. The primary purpose of this work is to apply the IDC methods to the low order operator splitting methods in order to obtain higher order accuracy.

The paper is organized as follows. In Section 2, we briefly review several classical operator splitting methods and show how these methods can be cast as additive RK (ARK) methods. In Section 3 we formulate the IDC methodology for application to operator splitting schemes. In Section 4, we prove that IDC methods can correct for both the splitting and numerical errors of ODEs, giving r higher degrees of accuracy with each correction, where r is the order of the method used in the correction steps. In Section 5, as an interesting example, we will show how to use integral deferred correction for operator splitting (IDC-OS) schemes as a temporal discretization when solving PDEs. In Section 6 we carry out numerical simulations based on IDC methods for both linear and non-linear parabolic equations, and demonstrate that the new framework can achieve high order accuracy in time. In Section 7 we conclude the paper and discuss future work. We note that both the parallel time stepping version of IDC and the work presented in this paper are likely to benefit from the work in [20], and will be the subject of further investigation.

2. Operator splitting schemes for ODEs

In this section, we review several splitting methods which will serve as the base solver in the IDC framework. For differential operator splitting, such as Lie–Trotter splitting and Strang splitting, which happens at continuous level, we will apply appropriate numerical methods to the sub-problems and refer the whole approach as the discrete form of differential splitting. For both the differential splitting and algebraic splitting, we will show that each of the numerical schemes can be written as an ARK method. This insight is the first step required to apply the IDC methodology [4] to operator splitting schemes, which is the primary purpose of the present work.

2.1. Review of ARK methods

For IVP (1.1), when different p-stage RK integrators are applied to each operator L_ν, the entire numerical method is called an ARK method. If we define the numerical solution after n time steps as v^n, which is an approximation to the exact solution $u(t_n)$, then one step of a p-stage ARK method is given by

$$v^{n+1} = v^n + \Delta t \sum_{\nu=1}^{\Lambda} \sum_{i=1}^{p} b_i^{[\nu]} f_\nu(t_n + c_i^{[\nu]} \Delta t, \tilde{v}_i), \tag{2.1}$$

with

$$\tilde{v}_i = v^n + \Delta t \sum_{\nu=1}^{\Lambda} \sum_{j=1}^{p} a_{ij}^{[\nu]} f_\nu(t_n + c_j^{[\nu]} \Delta t, \tilde{v}_j) \tag{2.2}$$

and $\Delta t = t_{n+1} - t_n$. An ARK method is succinctly identified by its Butcher tableau, as is demonstrated in Table 2.1.

In the following sections, we will explicitly write out the Butcher tableau for each operator splitting scheme and conclude that each of the operator splitting schemes considered in this work is indeed a form of ARK method.

2.2. Lie–Trotter splitting

We describe Lie–Trotter splitting for (1.1) in the case of $\Lambda = 2$ in the right hand side functions. We consider a single interval $[t_n, t_{n+1}]$. With first order Lie–Trotter splitting, (1.1) can be solved by two sub-problems:

Table 2.1
Butcher tableau for a p-stage ARK method.

$c_1^{[1]}$	$\cdots$	$c_1^{[\Lambda]}$	$a_{11}^{[1]}$	$a_{12}^{[1]}$	$\cdots$	$a_{1p}^{[1]}$	$\cdots$	$a_{11}^{[\Lambda]}$	$a_{12}^{[\Lambda]}$	$\cdots$	$a_{1p}^{[\Lambda]}$
$c_2^{[1]}$	$\cdots$	$c_2^{[\Lambda]}$	$a_{21}^{[1]}$	$a_{22}^{[1]}$	$\cdots$	$a_{2p}^{[1]}$	$\cdots$	$a_{21}^{[\Lambda]}$	$a_{22}^{[\Lambda]}$	$\cdots$	$a_{2p}^{[\Lambda]}$
$\vdots$	$\cdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\cdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
$c_p^{[1]}$	$\cdots$	$c_p^{[\Lambda]}$	$a_{p1}^{[1]}$	$a_{p2}^{[1]}$	$\cdots$	$a_{pp}^{[1]}$	$\cdots$	$a_{p1}^{[\Lambda]}$	$a_{p2}^{[\Lambda]}$	$\cdots$	$a_{pp}^{[\Lambda]}$
			$b_1^{[1]}$	$b_2^{[1]}$	$\cdots$	$b_p^{[1]}$	$\cdots$	$b_1^{[\Lambda]}$	$b_2^{[\Lambda]}$	$\cdots$	$b_p^{[\Lambda]}$

Table 2.2
Butcher tableau for Lie–Trotter splitting.

0	0	0	0	0	0	0
1	0	1	0	0	0	0
1	0	1	0	0	0	1
	0	1	0	0	0	1

$$\begin{cases} u_t = f_1(t,u), & \text{on } [t_n, t_{n+1}], \\ u_t = f_2(t,u), & \text{on } [t_n, t_{n+1}]. \end{cases} \tag{2.3}$$

The solution calculated from the first equation is used as the initial value of the second equation. Note that this splitting occurs on the continuous level. In order to define a discrete solver for (1.1), we need to choose a numerical scheme for solving each sub-problem. For example, if we use the backward Euler scheme to solve both equations, we obtain a scheme of the form

$$\begin{cases} \dfrac{\widetilde{v} - v^n}{\Delta t} = f_1(t_{n+1}, \widetilde{v}), \\ \dfrac{v^{n+1} - \widetilde{v}}{\Delta t} = f_2(t_{n+1}, v^{n+1}), \end{cases} \tag{2.4}$$

where v^n denotes the numerical approximation for u at time level t_n. However, this approach only produces a first order approximation.

In order to make use of IDC methodology [4] to lift the order of accuracy of (2.4), we write a Butcher tableau for (2.4) in Table 2.2. Comparing the Butcher tableau for the Lie–Trotter splitting with the general form of the Butcher tableau of an ARK method, we can view the discrete form of Lie–Trotter splitting (2.4) as a 2-stage ARK method. This can be extended to the case of Λ operators, where the resulting Butcher tableau would be a Λ-stage ARK method.

2.3. Strang splitting

In this section, we consider the second order Strang splitting for the case of three operators to demonstrate how to construct Butcher tableaus for general differential operator splitting schemes. The case of $\Lambda = 3$ operators can arise when splitting a stiff ODE into three sub-problems while maintaining second order accuracy in time.

We also focus on a single time step, $[t_n, t_{n+1}]$. Second order Strang splitting for (1.1) reads as

$$\begin{cases} u_t = f_1(t,u), & t \in [t_n, t_{n+\frac{1}{2}}], \\ u_t = f_2(t,u), & t \in [t_{n+\frac{1}{2}}, t_{n+1}], \\ u_t = f_3(t,u), & t \in [t_n, t_{n+1}], \\ u_t = f_2(t,u), & t \in [t_n, t_{n+\frac{1}{2}}], \\ u_t = f_1(t,u), & t \in [t_{n+\frac{1}{2}}, t_{n+1}]. \end{cases} \tag{2.5}$$

Note that this splitting occurs on the continuous level, i.e. the temporal derivative for each sub-problem in (2.5) has yet to be discretized. If we discretize Eqs. (2.5) with trapezoidal rule, we obtain an update of the form,

Table 2.3
Butcher tableau for Strang splitting with $\Lambda = 3$.

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
$\frac{1}{2}$	$\frac{1}{2}$	0	$\frac{1}{4}$	$\frac{1}{4}$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	$\frac{1}{4}$	$\frac{1}{4}$	0	0	0	0	0	$\frac{1}{4}$	$\frac{1}{4}$	0	0	0	0	0	0	0	0	0	0
0	0	1	$\frac{1}{4}$	$\frac{1}{4}$	0	0	0	0	0	$\frac{1}{4}$	$\frac{1}{4}$	0	0	0	0	0	$\frac{1}{2}$	$\frac{1}{2}$	0	0	
$\frac{1}{2}$	$\frac{1}{2}$	0	$\frac{1}{4}$	$\frac{1}{4}$	0	0	0	0	0	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	0	0	0	$\frac{1}{2}$	$\frac{1}{2}$	0	0	
1	0	0	$\frac{1}{4}$	$\frac{1}{4}$	0	0	$\frac{1}{4}$	$\frac{1}{4}$	0	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	0	0	0	$\frac{1}{2}$	$\frac{1}{2}$	0	0	
			$\frac{1}{4}$	$\frac{1}{4}$	0	0	$\frac{1}{4}$	$\frac{1}{4}$	0	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	0	0	0	$\frac{1}{2}$	$\frac{1}{2}$	0	0	

Table 2.4
Butcher tableau for Strang splitting when $L_3 = 0$.

0	0	0	0	0	0	0	0	0	0	0	0	0	0
$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{4}$	0	0	0	0	0	0	0	0	0	0
0	1	$\frac{1}{4}$	$\frac{1}{4}$	0	0	0	0	0	$\frac{1}{4}$	$\frac{1}{4}$	0	0	0
$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{4}$	0	0	0	0	0	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	0
1	0	$\frac{1}{4}$	$\frac{1}{4}$	0	0	$\frac{1}{4}$	$\frac{1}{4}$	0	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	0
		$\frac{1}{4}$	$\frac{1}{4}$	0	$\frac{1}{4}$	$\frac{1}{4}$	0	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	0	0

Table 2.5
Butcher tableau for ADI scheme.

0	0	0	0	0	0	0
$\frac{1}{2}$	0	$\frac{1}{2}$	0	$\frac{1}{2}$	0	0
1	0	1	0	$\frac{1}{2}$	0	$\frac{1}{2}$
	0	1	0	$\frac{1}{2}$	0	$\frac{1}{2}$

$$\begin{cases} \dfrac{\tilde{v}_1 - v^n}{\frac{1}{2}\Delta t} = \dfrac{1}{2}(f_1(t_n, v^n) + f_1(t_{n+\frac{1}{2}}, \tilde{v}_1)), \\ \dfrac{\tilde{v}_2 - \tilde{v}_1}{\frac{1}{2}\Delta t} = \dfrac{1}{2}(f_2(t_{n+\frac{1}{2}}, \tilde{v}_1) + f_2(t_{n+1}, \tilde{v}_2)), \\ \dfrac{\tilde{v}_3 - \tilde{v}_2}{\Delta t} = \dfrac{1}{2}(f_3(t_n, \tilde{v}_2) + f_3(t_{n+1}, \tilde{v}_3)), \\ \dfrac{\tilde{v}_4 - \tilde{v}_3}{\frac{1}{2}\Delta t} = \dfrac{1}{2}(f_2(t_n, \tilde{v}_3) + f_2(t_{n+\frac{1}{2}}, \tilde{v}_4)), \\ \dfrac{v^{n+1} - \tilde{v}_4}{\frac{1}{2}\Delta t} = \dfrac{1}{2}(f_1(t_{n+\frac{1}{2}}, \tilde{v}_4) + f_1(t_{n+1}, v^{n+1})), \end{cases} \tag{2.6}$$

where $t_{n+\frac{1}{2}} = t_n + \frac{1}{2}\Delta t$. In Tables 2.3 and 2.4, we write this scheme in the Butcher tableau. Comparing this with the Butcher tableau of the ARK methods, again we see that we can view the Strang splitting as a 5-stage ARK scheme.

2.4. ADI splitting

The ADI method is a predictor–corrector scheme as a typical example of algebraic splitting, which happens after the discretization of equations. Here we are considering the discretized ODE version of the Peaceman–Rachford scheme [30] for (1.1). When $\Lambda = 2$, the ADI scheme takes the form

$$\begin{cases} \dfrac{\tilde{v} - v^n}{\frac{1}{2}\Delta t} = f_1(t_{n+\frac{1}{2}}, \tilde{v}) + f_2(t_n, v^n), \\ \dfrac{v^{n+1} - \tilde{v}}{\frac{1}{2}\Delta t} = f_1(t_{n+\frac{1}{2}}, \tilde{v}) + f_2(t_{n+1}, v^{n+1}). \end{cases} \tag{2.7}$$

The Butcher tableau for the scheme (2.7) is shown in Table 2.5, and clearly, we see that we can view the ADI splitting scheme as a 2-stage ARK method.

3. Formulation of IDC-OS schemes

In this section, we review the formulation of IDC-OS presented in [4]. The authors [4] considered IDC methods for implicit–explicit (IMEX) schemes, where the non-stiff part of the problem was treated explicitly, and the stiff part of the problem was treated implicitly. At present, our focus is on entirely implicit schemes.

We begin with some preliminary definitions. The starting point is to partition the time interval $[0, T]$ into intervals $[t_n, t_{n+1}]$, $n = 0, 1, \ldots, N-1$, that satisfy

$$0 = t_0 < t_1 < t_2 < \cdots < t_n < \cdots < t_N = T. \tag{3.1}$$

"Macro"-time steps are defined by $H_n = t_{n+1} - t_n$, and we permit them to vary with n. Next, each interval $[t_n, t_{n+1}]$ is further partitioned into M sub-intervals $[t_{n,m}, t_{n,m+1}]$, $m = 0, 1, \ldots, M-1$,

$$t_n = t_{n,0} < t_{n,1} < t_{n,2} < \cdots < t_{n,m} < \cdots < t_{n,M} = t_{n+1} \tag{3.2}$$

with time step size $h_{n,m} = t_{n,m} - t_{n,m-1}$. If Gaussian quadrature nodes are selected, as was originally done with the SDC method [10], $h_{n,m}$ varies with m. Here, we only consider the case of uniform quadrature nodes, i.e. with $h_{n,m} = \frac{H_n}{M}$ for $m = 1, 2, \ldots, M$. Thus, without any ambiguity, we will drop the subscript m on $h_{n,m}$. Note that in what follows we will use superscript $[i]$ to denote the i-th correction at a discrete set of time points and superscript (i) to denote the continuous approximation given by passing an M-th order polynomial through the discrete approximation. For simplicity, we drop the n subscript for the description of the IDC procedure on "macro"-time interval $[t_n, t_{n+1}]$. The whole iterative prediction–correction procedure is completed before moving on to the next time interval $[t_{n+1}, t_{n+2}]$. The numerical solution at t_{n+1} serves as the initial condition for the following interval $[t_{n+1}, t_{n+2}]$.

Prediction step. Use an r_0-th order numerical method to obtain a preliminary solution to IVP (1.1)

$$\upsilon^{[0]} = (\upsilon_0^{[0]}, \upsilon_1^{[0]}, \ldots, \upsilon_m^{[0]}, \ldots, \upsilon_M^{[0]}), \tag{3.3}$$

which is an r_0-th order approximation to the exact solution

$$u = (u_0, u_1, \ldots, u_m, \ldots, u_M), \tag{3.4}$$

where $u_m = u(t_m)$ is the exact solution at t_m for $m = 0, 1, 2, \ldots, M$.

Correction step. Use the error function to improve the accuracy of the scheme at each iteration. For $k = 1$ to c_s (c_s is the number of correction steps):

(1) Denote the error function from the previous step as

$$e^{(k-1)}(t) = u(t) - \upsilon^{(k-1)}(t), \tag{3.5}$$

where $u(t)$ is the exact solution and $\upsilon^{(k-1)}(t)$ is an M-th degree polynomial interpolating $\upsilon^{[k-1]}$. Note that the error function, $e^{(k-1)}(t)$ is not a polynomial in general.

(2) Denote the residual function as

$$\epsilon^{(k-1)}(t) \equiv (\upsilon^{(k-1)})'(t) - f(t, \upsilon^{(k-1)}), \tag{3.6}$$

and compute the integral of the residual. For example,

$$\int_{t_0}^{t_{m+1}} \epsilon^{(k-1)}(\tau) d\tau \approx \upsilon_{m+1}^{[k-1]} - u_0 - (t_{m+1} - t_0) \sum_{j=0}^{M} \gamma_{m,j} f(t_j, \upsilon_j^{[k-1]}), \tag{3.7}$$

where $\gamma_{m,j}$ are the coefficients that result from approximation of the integral by quadrature formulas and $\upsilon_j^{[k-1]} = \upsilon^{(k-1)}(t_j)$.

(3) Use an r_k-th order numerical method to obtain an approximation to error vector

$$e^{[k-1]} = (e_0^{[k-1]}, \ldots, e_m^{[k-1]}, \ldots, e_M^{[k-1]}), \tag{3.8}$$

where $e_m^{[k-1]} = e^{(k-1)}(t_m)$ is the value of the exact error function (3.5) at time t_m and we denote it as

$$\delta^{[k]} = (\delta_0^{[k]}, \ldots, \delta_m^{[k]}, \ldots, \delta_M^{[k]}). \tag{3.9}$$

To compute $\delta^{[k]}$ by an operator splitting method consistent with the base method, we first express the error equation in a form consistent with original problem we are solving. We start by differentiating the error (3.5), together with (1.1)

$$
\begin{aligned}
(e^{(k-1)})'(t) &= u'(t) - (\upsilon^{(k-1)})'(t) \\
&= f(t, u(t)) - f(t, \upsilon^{(k-1)}(t)) - \epsilon^{(k-1)}(t) \\
&= f(t, \upsilon^{(k-1)}(t) + e^{(k-1)}(t)) - f(t, \upsilon^{(k-1)}(t)) - \epsilon^{(k-1)}(t).
\end{aligned} \tag{3.10}
$$

Bring the residual to the left hand side, we have

$$
(e^{(k-1)}(t) + \int_{t_0}^{t} \epsilon^{(k-1)}(\tau)d\tau)' = f(t, \upsilon^{(k-1)}(t) + e^{(k-1)}(t)) - f(t, \upsilon^{(k-1)}(t)). \tag{3.11}
$$

We now make the following change of variable,

$$
\begin{aligned}
&Q^{(k-1)}(t) = e^{(k-1)}(t) + \int_{t_0}^{t} \epsilon^{(k-1)}(\tau)d\tau, \\
&G^{(k-1)}(t, Q^{(k-1)}(t)) = f(t, \upsilon^{(k-1)}(t) + Q^{(k-1)}(t) - \int_{t_0}^{t} \epsilon^{(k-1)}(\tau)d\tau) - f(t, \upsilon^{(k-1)}(t)).
\end{aligned} \tag{3.12}
$$

With this change of variable, we see that the error equation can be expressed as an IVP of the form,

$$
\begin{cases}
(Q^{(k-1)})'(t) = G^{(k-1)}(t, Q^{(k-1)}(t)), & t \in [t_0, t_M], \\
Q^{(k-1)}(t_0) = 0.
\end{cases} \tag{3.13}
$$

This is now in the form of (1.1) and we can apply the same operator splitting scheme to (3.13) that we applied to (1.1) and obtain the numerical approximation to $\vartheta_m^{[k-1]} = Q^{(k-1)}(t_m)$. Recovering δ given ϑ is a simple procedure.

(4) Update the numerical solution as $\upsilon^{[k]} = \upsilon^{[k-1]} + \delta^{[k]}$.

Remark 1 *(The prediction step).* For example, if we apply the discrete form of first order Lie–Trotter splitting (2.4) to (1.1) with $\Lambda = 2$, we have for $m = 0, 1, 2, \ldots, M-1$,

$$
\begin{cases}
\dfrac{\widetilde{\upsilon} - \upsilon_m^{[0]}}{h_n} = f_1(t_{m+1}, \widetilde{\upsilon}), \\
\dfrac{\upsilon_{m+1}^{[0]} - \widetilde{\upsilon}}{h_n} = f_2(t_{m+1}, \upsilon_{m+1}^{[0]}).
\end{cases} \tag{3.14}
$$

Remark 2 *(The correction step).* As an example, if we use ADI splitting in the correction step, we will solve (3.13) with $\Lambda = 2$, we have for $m = 0, 1, 2, \ldots, M-1$,

$$
\begin{cases}
\dfrac{\widetilde{\vartheta} - \vartheta_m^{[k]}}{\frac{h_n}{2}} = G_1^{(k-1)}(t_m + \dfrac{h_n}{2}, \widetilde{\vartheta}) + G_2^{(k-1)}(t_m, \vartheta_m^{[k]}), \\
\dfrac{\vartheta_{m+1}^{[k]} - \widetilde{\vartheta}}{\frac{h_n}{2}} = G_2^{(k-1)}(t_{m+1}, \vartheta_{m+1}^{[k]}) + G_1^{(k-1)}(t_m + \dfrac{h_n}{2}, \widetilde{\vartheta}),
\end{cases} \tag{3.15}
$$

where

$$
G_\nu^{(k-1)}(t, Q^{(k-1)}(t)) = f_\nu(t, \upsilon^{(k-1)}(t) + Q^{(k-1)}(t) - \int_{t_0}^{t} \epsilon^{(k-1)}(\tau)d\tau) - f_\nu(t, \upsilon^{(k-1)}(t)) \tag{3.16}
$$

for $\nu = 1, 2$. Moreover, we note that we split the residual term equally for each operator in implementation.

4. Analysis of IDC-OS methods

In this section, we will discuss the error estimate for IDC-OS methods. Our analysis is similar to previous work of IDC-RK and IDC-ARK [6,5,4].

In Section 4.1, we will establish that the IDC procedure can successfully reduce the splitting error for differential operator splitting methods where each sub-problem is solved exactly. In Section 4.2, we continue by leveraging the ideas from the work in [4], and prove that the overall accuracy for the fully discrete methods is increased, as expected, with each successive correction. The second set of arguments apply to the discrete form of the differential operator splitting methods as well as

the algebraic operator splitting methods. We present results for the stability regions of IDC-OS schemes in Section 4.3. We remark that throughout this section, superscripts with a curly bracket $\{k\}$ denote the analytical functions related to solutions through differential splitting methods.

4.1. Splitting error: exact solutions to sub-problems

Differential operator splitting introduces a splitting error. If each sub-problem is solved exactly, the overall method only contains splitting error. Our starting point is to prove that IDC framework can reduce this splitting error. The primary result from this subsection is given by the following theorem.

Theorem 4.1. *Assume $u(t)$ is the exact solution to IVP (1.1). Consider one time interval of an IDC method with $t \in [0, h]$. Suppose Lie–Trotter splitting (2.3) is used in the prediction step and the successive c_s correction steps, and the sub-problems in each step are solved exactly. If $u(t)$ and f_ν are at least $(c_s + 3)$ differentiable, then the splitting error is of order $\mathcal{O}(h^{c_s+2})$ after c_s correction steps.*

The proof of Theorem 4.1 follows by induction from the following two lemmas: Lemma 4.2 for the prediction step and Lemma 4.3 for the correction steps respectively.

Lemma 4.2 *(Prediction step). Consider IVP (1.1) on the interval $t \in [0, h]$. If $u(t)$ and f_ν satisfy the smoothness requirements in Theorem 4.1, and $u^{\{0\}}(t)$ is the solution obtained by applying Lie–Trotter splitting (2.3) to (1.1), and the followed sub-problems are solved exactly, then the splitting error scales as*

$$\|e^{(0)}\| = \|u(h) - u^{\{0\}}(h)\| \sim \mathcal{O}(h^2), \qquad t \in [0, h].$$

The conclusion of Lemma 4.2 is simply a restatement of what the local error of splitting methods measures. The splitting error is $\mathcal{O}(h^2)$ for first order Lie–Trotter splitting [26].

Lemma 4.3 *(Correction step). Assume $u(t)$ is the solution to IVP (1.1) on the interval $t \in [0, h]$. Let $u(t)$, and f_ν satisfy the smoothness requirements in Theorem 4.1. For $k \le c_s$, let $u^{\{k\}}(t)$ be the solution after the prediction step and k-th correction step via Lie–Trotter splitting method in Theorem 4.1. If $\|e^{(k-1)}\| \sim \mathcal{O}(h^{k+1})$, then $\|e^{(k)}\| \sim \mathcal{O}(h^{k+2})$ after k correction steps.*

Proof. We show the proof with the simple case $\Lambda = 2$. We have the error equation (3.13) after prediction and $(k-1)$ correction steps. Use the Lie–Trotter splitting method (2.3) to solve (3.13), we have

$$\begin{cases} (Q_1^{\{k-1\}}(t))' = G_1^{(k-1)}(t, Q_1^{\{k-1\}}(t)), & t \in [0, h], \\ Q_1^{\{k-1\}}(0) = 0, \end{cases} \tag{4.1}$$

and

$$\begin{cases} (Q_2^{\{k-1\}}(t))' = G_2^{(k-1)}(t, Q_2^{\{k-1\}}(t)), & t \in [0, h], \\ Q_2^{\{k-1\}}(0) = Q_1^{\{k-1\}}(h), \end{cases} \tag{4.2}$$

with $G_\nu^{(k-1)}(t, Q^{(k-1)}(t))$ defined in (3.16). Hence $Q_2^{\{k-1\}}(h)$ is the approximation of $Q^{(k-1)}(h)$ solved by the Lie–Trotter splitting method. It's easy to see that

$$e^{(k)}(h) = e^{(k-1)}(h) - e^{\{k-1\}}(h) = Q^{(k-1)}(h) - Q_2^{\{k-1\}}(h), \tag{4.3}$$

for $t \in [0, h]$. To prove $Q^{(k-1)}(h) - Q_2^{\{k-1\}}(h) \sim \mathcal{O}(h^{k+2})$, we examine the scaled variant

$$\bar{Q}^{(k-1)}(t) = \frac{1}{h^k} Q^{(k-1)}(t). \tag{4.4}$$

With this new notation, IVP (3.13) can be equivalently written as

$$\begin{cases} (\bar{Q}^{(k-1)}(t))' = \bar{G}^{(k-1)}(t, \bar{Q}^{(k-1)}(t)), & t \in [0, h], \\ \bar{Q}^{(k-1)}(0) = 0, \end{cases} \tag{4.5}$$

with

$$\bar{G}^{(k-1)}(t, \bar{Q}^{(k-1)}(t)) = \frac{1}{h^k} G^{(k-1)}(t, h^k \bar{Q}^{(k-1)}(t)). \tag{4.6}$$

Using the Lie–Trotter splitting method to solve IVP (4.5) will give us

$$\begin{cases} (\bar{Q}_1^{\{k-1\}}(t))' = \bar{G}_1^{(k-1)}(t, \bar{Q}_1^{\{k-1\}}(t)), & t \in [0, h], \\ \bar{Q}_1^{\{k-1\}}(0) = 0, \end{cases} \tag{4.7}$$

and

$$\begin{cases} (\bar{Q}_2^{\{k-1\}}(t))' = \bar{G}_2^{(k-1)}(t, \bar{Q}_2^{\{k-1\}}(t)), & t \in [0,h], \\ \bar{Q}_2^{\{k-1\}}(0) = \bar{Q}_1^{\{k-1\}}(h), \end{cases} \tag{4.8}$$

with

$$\bar{G}_\nu^{(k-1)}(t, \bar{Q}^{(k-1)}(t)) = \frac{1}{h^k} G_\nu^{(k-1)}(t, h^k \bar{Q}^{(k-1)}(t)), \qquad \nu = 1,2. \tag{4.9}$$

$\bar{Q}_2^{\{k-1\}}(h)$ is the approximation to $\bar{Q}^{(k-1)}(h)$ through Lie–Trotter splitting. If $e^{(k-1)} \sim \mathcal{O}(h^{k+1})$, it is easy to verify that $Q^{(k-1)}(t) \sim \mathcal{O}(h^{k+1})$ and $G^{(k-1)}(t, Q^{(k-1)}(t)) \sim \mathcal{O}(h^{k+1})$. Similar as the work of IDC-RK in [6], one can further check that $\frac{d}{dt}\bar{Q}^{\{k-1\}}(t) \sim \mathcal{O}(1)$ and $\bar{G}^{(k-1)}(t, \bar{Q}^{\{k-1\}}(t)) \sim \mathcal{O}(1)$. Therefore,

$$\|\bar{Q}^{(k-1)}(h) - \bar{Q}_2^{\{k-1\}}(h)\| \sim \mathcal{O}(h^2). \tag{4.10}$$

Notice that IVP (4.1) and (4.7) are both first order ODEs, and $h^k \bar{G}_1^{(k-1)}(t, \bar{Q}_1^{\{k-1\}}(t)) = G_1^{(k-1)}(t, h^k \bar{Q}_1^{\{k-1\}}(t))$. Since $\bar{Q}_1^{\{k-1\}}(t)$ is the solution to (4.7), $h^k \bar{Q}_1^{\{k-1\}}(t)$ is a solution to (4.1). Through the uniqueness of the solution for IVP, one can conclude that

$$\bar{Q}_1^{\{k-1\}}(h) = \frac{1}{h^k} Q_1^{\{k-1\}}(h). \tag{4.11}$$

Similarly, from IVP (4.2) and (4.8), one can further conclude

$$\bar{Q}_2^{\{k-1\}}(h) = \frac{1}{h^k} Q_2^{\{k-1\}}(h). \tag{4.12}$$

Thus (4.10) is equivalent to

$$\|\frac{1}{h^k} Q^{(k-1)}(h) - \frac{1}{h^k} Q_2^{(k-1)}(h)\| \sim \mathcal{O}(h^2), \tag{4.13}$$

i.e.

$$\|e^{(k)}\| = \|Q^{(k-1)}(h) - Q_2^{\{k-1\}}(h)\| \sim \mathcal{O}(h^{k+2}). \tag{4.14}$$

We now complete the proof of Lemma 4.3 for the case of Lie–Trotter splitting. □

The conclusion in Theorem 4.1 also holds for Strang splitting method (2.5) and the proof is essentially the same as Lie–Trotter splitting. We have now demonstrated that IDC can lift the order of accuracy when each sub-problem is solved exactly, however, in practice, we usually do not have access to analytical solutions for these sub-problems. We will consider the fully discrete scheme in the next section.

4.2. Local truncation error: discrete solutions to sub-problems

A fully discrete solution introduces additional error beyond the splitting error. In this section, we turn to analyzing fully discrete IDC-OS schemes and begin with some preliminary definitions [6].

Definition 4.4 *(Discrete differentiation).* Consider the discrete data set, $(\vec{t}, \vec{\psi}) = \{(t_0, \psi_0), \ldots, (t_M, \psi_M)\}$, with $\{t_m\}_{m=0}^M$ defined as uniform quadrature nodes in (3.2). We denote L^M as the M-th degree Lagrangian interpolant of (t, ψ):

$$L^M(t,\psi) = \sum_{m=0}^{M} c_m(t)\psi_m, \qquad c_m(t) = \prod_{n \neq m} \frac{t - t_n}{t_m - t_n}. \tag{4.15}$$

An s-th degree discrete differentiation is a linear mapping that maps $\vec{\psi}$ to $\overrightarrow{\hat{d}_s\psi}$, where

$$(\hat{d}_s\psi)_m = \frac{\partial^s}{\partial t^s} L^M(t,\psi)\,|_{t=t_m}\,. \tag{4.16}$$

This linear mapping can be represented by a matrix multiplication $\overrightarrow{\hat{d}_s\psi} = \hat{D}_s \cdot \vec{\psi}$, where $\hat{D}_s \in \Re^{(M+1)\times(M+1)}$ and $(\hat{D})_{mn} = \frac{\partial^s}{\partial t^s} c_n(t)\,|_{t=t_m}$, $m, n = 0, \ldots, M$.

Definition 4.5. The $(\hat{S},\infty)$ Sobolev norm of the discrete data set $(\vec{t},\vec{\psi})$ is defined as

$$\|\vec{\psi}\|_{\hat{S},\infty} \doteq \sum_{s=0}^{\hat{S}} \|\overrightarrow{\hat{d}_s\psi}\|_\infty = \sum_{s=0}^{\hat{S}} \|\hat{D}_s\cdot\vec{\psi}\|_\infty, \tag{4.17}$$

where $\overrightarrow{\hat{d}_s\psi} = Id\cdot\hat{\psi}$ is the identity matrix operating on $\hat{\psi}$.

Definition 4.6 *(Smoothness of a discrete data set).* A discrete data set $(\vec{t},\vec{\psi}) = \{(t_0,\psi_0),\dots,(t_M,\psi_M)\}$ possesses $\hat{S}$ $(\hat{S}\le M)$ degrees of smoothness if $\|\vec{\psi}\|_{\hat{S},\infty}$ is bounded as $h\to 0$, with h defined as the step size in the sub-interval (t_m,t_{m+1}) where $m=0,1,\cdots,M-1$.

As discussed in Section 2, all the listed operator splitting schemes are a form of ARK methods. Therefore, we can use the framework of the IDC-ARK schemes in [4] to enhance the order of the discretized scheme. Hence, we shall describe only what is needed for clarity when extending the results of the work in [4] to the fully implicit case under consideration here. For further details, we refer the reader to [5,4]. The theorems below apply to lifting the order of algebraic splitting as well as the discrete form of differential splitting. The splitting error discussed in Theorem 4.1 is directly related to the local truncation error. We note that the results in the following theorem can be generalized to all IDC-OS schemes which can be written as a form of ARK method and the proof is quite similar.

Theorem 4.7. *Let $u(t)$ be the solution to IVP (1.1). Assume $u(t)$, $f(t,u)$ and $f_v(t,u)$ are at least σ differentiable with respect to each argument, where $\sigma\ge M+2$. Consider one time interval of an IDC method with $t\in[0,H]$ and $M+1$ uniformly distributed quadrature points. Suppose an r_0-th order ARK method (2.1) is used in the prediction step and $(r_1,r_2,\dots,r_{c_s})$-th order ARK methods are used in the successive c_s correction steps. Let $s_k=\sum_{j=0}^{k} r_j$. If $s_{c_s}\le M+1$, then the local truncation error is of order $\mathcal{O}(h^{s_{c_s}+1})$ after c_s correction steps.*

The proof of Theorem 4.7 follows by induction from the following lemmas for the prediction and correction steps. For clarity, similar as [4], we will sketch a proof for Lie–Trotter splitting.

Lemma 4.8 *(Prediction step). Consider an r_0-th order ARK method for (1.1) on $[0,H]$, with $(M+1)$ uniformly distributed quadrature points. $u(t)$ and f_v satisfy the smoothness requirement in Theorem 4.7 and let $v^{[0]}=(v_0^{[0]},v_1^{[0]},\dots v_m^{[0]},\dots,v_M^{[0]})$ be the numerical solution. Then,*

(1) *The error vector $e^{[0]}=u-v^{[0]}$ satisfies $\|e^{[0]}\|_\infty\sim\mathcal{O}(h^{r_0+1})$.*

(2) *The rescaled error vector $\bar{e}^{[0]}=\dfrac{1}{h^{r_0}}e^{[0]}$ has $\min(\sigma-r_0,M)$ degrees of smoothness in the discrete sense.*

Proof. (1) is obvious. We will prove (2) next. We drop the superscript [0] as there is no ambiguity. Applying the discrete form of the Lie–Trotter splitting (2.4) to IVP (1.1) with $\Lambda=2$, we have

$$\begin{cases} \dfrac{\widetilde{v}-v_m}{h} = f_1(t_{m+1},\widetilde{v}),\\ \dfrac{v_{m+1}-\widetilde{v}}{h} = f_2(t_{m+1},v_{m+1}), \end{cases} \tag{4.18}$$

i.e.

$$v_{m+1} = v_m + hf_1(t_{m+1},\widetilde{v}) + hf_2(t_{m+1},v_{m+1}). \tag{4.19}$$

Performing Taylor expansion of $f_1(t_{m+1},\widetilde{v})$ at $t=t_m$, we get

$$v_{m+1} = v_m + hf_1(t_m,v_m) + hf_2(t_{m+1},v_{m+1}) + \sum_{i=1}^{\sigma-2}\frac{h^{i+1}}{i!}\frac{d^i f_1}{dt^i}(t_m,v_m) + \mathcal{O}(h^\sigma), \tag{4.20}$$

on the other hand, the exact solution satisfies

$$\begin{aligned} u_{m+1} &= u_m + \int_{t_m}^{t_{m+1}} f_1(\tau,u(\tau))d\tau + \int_{t_m}^{t_{m+1}} f_2(\tau,u(\tau))d\tau\\ &= u_m + hf_1(t_m,u_m) + \sum_{i=1}^{\sigma-2}\frac{h^{i+1}}{(i+1)!}\frac{d^i f_1}{dt^i}(t_m,u_m)\\ &\quad + hf_2(t_{m+1},u_{m+1}) + \sum_{i=1}^{\sigma-2}\frac{(-1)^{i+1}h^{i+1}}{(i+1)!}\frac{d^i f_2}{dt^i}(t_{m+1},u_{m+1}) + \mathcal{O}(h^\sigma). \end{aligned} \tag{4.21}$$

Subtracting (4.20) from (4.21) gives

$$e_{m+1} = e_m + h(f_1(t_m, u_m) - f_1(t_m, \upsilon_m)) + h(f_2(t_{m+1}, u_{m+1}) - f_2(t_{m+1}, \upsilon_{m+1})) + \sum_{i=1}^{\sigma-2} \frac{h^{i+1}}{(i+1)!} \frac{d^i f_1}{dt^i}(t_m, u_m) + \sum_{i=1}^{\sigma-2} \frac{(-1)^{i+1} h^{i+1}}{(i+1)!} \frac{d^i f_2}{dt^i}(t_{m+1}, u_{m+1}) - \sum_{i=1}^{\sigma-2} \frac{h^{i+1}}{i!} \frac{d^i f_1}{dt^i}(t_m, \upsilon_m) + \mathcal{O}(h^\sigma),$$

where $e_{m+1} = u_{m+1} - \upsilon_{m+1}$ is the error at t_{m+1}. Denote

$$l_m = (f_1(t_m, u_m) - f_1(t_m, \upsilon_m)) + (f_2(t_{m+1}, u_{m+1}) - f_2(t_{m+1}, \upsilon_{m+1})) \tag{4.22}$$

and

$$r_m = \sum_{i=1}^{\sigma-2} \frac{h^{i+1}}{(i+1)!} \frac{d^i f_1}{dt^i}(t_m, u_m) + \sum_{i=1}^{\sigma-2} \frac{(-1)^{i+1} h^{i+1}}{(i+1)!} \frac{d^i f_2}{dt^i}(t_{m+1}, u_{m+1}) - \sum_{i=1}^{\sigma-2} \frac{h^{i+1}}{i!} \frac{d^i f_1}{dt^i}(t_m, \upsilon_m). \tag{4.23}$$

We will use an inductive approach with respect to the degree of the smoothness s to investigate the smoothness of the rescaled error vector $\bar{e} = \frac{e}{h}$, and

$$(d_1 \bar{e})_m = \frac{\bar{e}_{m+1} - \bar{e}_m}{h} = \frac{l_m}{h} + \frac{r_m}{h^2} + \mathcal{O}(h^{\sigma-2}). \tag{4.24}$$

First of all, $\bar{e}$ has at least zero degrees of smoothness in the discrete sense since $\|\bar{e}\| \sim \mathcal{O}(h)$. Assume $\bar{e}$ has $s \le M-1$ degrees of smoothness, we will show $d_1\bar{e}$ has s degrees of smoothness, from which we can conclude $\bar{e}$ has $(s+1)$ degrees of smoothness.

$$\begin{aligned} l_m &= (f_1(t_m, u_m) - f_1(t_m, \upsilon_m)) + (f_2(t_{m+1}, u_{m-1}) - f_2(t_{m+1}\upsilon_{m+1})) \\ &= \sum_{i=1}^{\sigma-2} \frac{1}{i!} (e_m)^i \frac{\partial^i f_1}{\partial u^i}(t_m, u_m) + \sum_{i=1}^{\sigma-2} \frac{1}{i!} (e_{m+1})^i \frac{\partial^i f_2}{\partial u^i}(t_{m+1}, u_{m+1}) + \mathcal{O}((e_m)^{\sigma-1}) + \mathcal{O}((e_{m+1})^{\sigma-1}) \\ &= \sum_{i=1}^{\sigma-2} \frac{h^i}{i!} (\bar{e}_m)^i \frac{\partial^i f_1}{\partial u^i}(t_m, u_m) + \sum_{i=1}^{\sigma-2} \frac{h^i}{i!} (\bar{e}_{m+1})^i \frac{\partial^i f_2}{\partial u^i}(t_{m+1}, u_{m+1}) + \mathcal{O}((h\bar{e}_m)^{\sigma-1}) + \mathcal{O}((h\bar{e}_{m+1})^{\sigma-1}). \end{aligned} \tag{4.25}$$

By assuming that f_1 and f_2 have at least σ degrees of smoothness, we can conclude $\frac{\partial^i f_1}{\partial u^i}$ and $\frac{\partial^i f_2}{\partial u^i}$ have at least $\sigma - i - 1$ degrees of smoothness, which implies $h^{i-1}\frac{\partial^i f_1}{\partial u^i}$ and $h^{i-1}\frac{\partial^i f_2}{\partial u^i}$ have at least $\sigma - 2$ degrees of smoothness. Therefore $\frac{l_m}{h}$ will have $\min(\sigma-2, s)$ degrees of smoothness. Also, $\frac{r_m}{h^2}$ will have at least s degrees of smoothness. Therefore, $d_1\bar{e}$ has s degrees of smoothness. Therefore, $\bar{e}$ has $(s+1)$ degrees of smoothness. Notice that $\sigma \ge M+2$, we complete the inductive approach and conclude $\bar{e}$ has M degrees of smoothness. □

Before investigating the correction step for IDC-OS schemes, we describe some details for the error equations first. Notice that the error equation after $(k-1)$ correction steps has the form of (3.11), with the notation $Q^{(k-1)}(t)$, we actually implement the problem (3.13) on time interval $[t_m, t_{m+1}]$ via discrete Lie–Trotter splitting as follows

$$\begin{cases} \dfrac{\widetilde{\vartheta} - \vartheta_m^{[k]}}{h} = G_1^{(k-1)}(t_{m+1}, \widetilde{\vartheta}), \\ \dfrac{\vartheta_{m+1}^{[k]} - \widetilde{\vartheta}}{h} = G_2^{(k-1)}(t_{m+1}, \vartheta_{m+1}^{[k]}), \end{cases} \tag{4.26}$$

through which $\vartheta_{m+1}^{[k]}$ is updated. Furthermore, we can update $\delta_{m+1}^{[k]}$ by (3.12) and (3.7). Similarly, if we apply Lie–Trotter splitting to the scaled error equation (4.5) over the time interval $[t_m, t_{m+1}]$, we have

$$\begin{cases} \dfrac{\widetilde{\bar{\vartheta}} - \bar{\vartheta}_m^{[k]}}{h} = \bar{G}_1^{(k-1)}(t_{m+1}, \widetilde{\bar{\vartheta}}), \\ \dfrac{\bar{\vartheta}_{m+1}^{[k]} - \widetilde{\bar{\vartheta}}}{h} = \bar{G}_2^{(k-1)}(t_{m+1}, \bar{\vartheta}_{m+1}^{[k]}), \end{cases} \tag{4.27}$$

from which we obtain $\bar{\vartheta}_{m+1}^{[k]}$ and further $\bar{\delta}_{m+1}^{[k]}$.

Lemma 4.9 *(Correction step). Let $u(t)$ and L_v satisfy the smoothness requirements in Theorem 4.7. Suppose $e^{[k-1]} \sim \mathcal{O}(h^{s_{k-1}+1})$ and $\bar{e}^{[k-1]} = \dfrac{1}{h^{s_{k-1}}} e^{[k-1]}$ has $(M+1-s_{k-1})$ degrees of smoothness in the discrete sense after the $(k-1)$-th correction step. Then, after the k-th correction step using an r_k-th order ARK method and $k \le k_{c_s}$,*

(1) $\|e^{[k]}\|_\infty \sim \mathcal{O}(h^{s_k+1})$.

(2) *The rescaled error vector* $\bar{e}^{[k]} = \dfrac{1}{h^{s_k}} e^{[k]}$ *has* $M+1-s_k$ *degrees of smoothness in the discrete sense.*

Proof. The proof of Lemma 4.9 is similar to that of Lemma 4.8, but more tedious. Similar as in [4], we outline the proof here and present the difference between the proof of IDC-OS and IDC-RK, IDC-ARK in Proposition 4.10, we refer the reader to [5] for details.

1. Subtract the numerical error vector from the integrated error equation
$$e_{m+1}^{[k]} = e_{m+1}^{[k-1]} - \delta_{m+1}^{[k]} \tag{4.28}$$
and make necessary substitution and expansion via the rescaled equations.
2. Bound the error $e^{[k]}$ by an inductive approach.

The following proposition is about the equivalence of the rescaled error vector and unscaled error vectors for Lie–Trotter splitting. We remark that the proof of this proposition shows the difference of the proof between IDC-OS and IDC-RK in [5], IDC-ARK in [4].

Proposition 4.10. *Consider a single step of an IDC scheme constructed with the Lie–Trotter splitting scheme for the error equation, assume the exact solution* $u(t)$*, and* L_v *satisfies the smoothness requirement in Theorem 4.7, then for a sufficiently smooth error function* $e^{(k-1)}(t)$*, the difference between the Taylor series for the exact error* $e^{(k-1)}(t_{m+1})$ *and the numerical error* $\delta_{m+1}^{[k]}$ *is* $\mathcal{O}(h^{k+2})$ *after k correction steps.*

Proof. Notice that the left and right hand side terms of the rescaled error equation (4.5) is $\mathcal{O}(1)$, applying the discrete form of Lie–Trotter splitting scheme (2.4) to (4.5) will result in
$$\bar{Q}_{m+1}^{[k-1]} - \bar{\vartheta}_{m+1}^{[k]} \sim \mathcal{O}(h^2). \tag{4.29}$$
Since
$$\bar{Q}_{m+1}^{[k-1]} = \frac{1}{h^k} Q_{m+1}^{[k-1]} = \frac{1}{h^k}\left(e_{m+1}^{[k-1]} - \int_{t_m}^{t_{m+1}} \varepsilon^{(k-1)}(\tau)d\tau\right). \tag{4.30}$$
The proof is complete if the following argument holds.
$$h^k \bar{\delta}_m^{[k]} = \delta_m^{[k]} + \mathcal{O}(h^\sigma), \qquad m = 0, 1, 2, \ldots, M, \tag{4.31}$$
which is also equivalent to
$$h^k \bar{\vartheta}_m^{[k]} = \vartheta_m^{[k]} + \mathcal{O}(h^\sigma), \qquad m = 0, 1, 2, \ldots, M. \tag{4.32}$$
We will prove (4.32) by induction. (4.32) holds for $m = 0$ since the initial condition for the error equation is set as 0. Assume (4.31) holds for m, then
$$\begin{aligned}
\widetilde{\bar{\vartheta}} &= \bar{\vartheta}_m^{[k]} + h\bar{G}_1^{(k-1)}(t_{m+1}, \widetilde{\bar{\vartheta}}) \\
&= \bar{\vartheta}_m^{[k]} + h\sum_{i=0}^{\sigma-1} \frac{h^i}{i!}\frac{d^i}{dt^i}\bar{G}_1^{(k-1)}(t_m, \bar{\vartheta}_m^{[k]}) + \mathcal{O}(h^\sigma) \\
&= \bar{\vartheta}_m^{[k]} + h\sum_{i=0}^{\sigma-1} \frac{h^i}{i!}\frac{d^i}{dt^i}\left(\frac{1}{h^k} G_1^{(k-1)}(t_m, h^k\bar{\vartheta}_m^{[k]})\right) + \mathcal{O}(h^\sigma) \\
&= \frac{1}{h^k}\left(\vartheta_m^{[k]} + h\sum_{i=0}^{\sigma-1} \frac{h^i}{i!}\frac{d^i}{dt^i} G_1^{(k-1)}(t_m, h^k\bar{\vartheta}_m^{[k]})\right) + \mathcal{O}(h^\sigma).
\end{aligned} \tag{4.33}$$
On the other hand, Taylor expanding $\widetilde{\vartheta}$ at t_m will give us
$$\begin{aligned}
\widetilde{\vartheta} &= \vartheta_m^{[k]} + hG_1^{(k-1)}(t_{m+1}, \widetilde{\vartheta}) \\
&= \vartheta_m^{[k]} + h\sum_{i=0}^{\sigma-1} \frac{h^i}{i!}\frac{d^i}{dt^i} G_1^{(k-1)}(t_m, h^k\bar{\vartheta}_m^{[k]}) + \mathcal{O}(h^\sigma).
\end{aligned} \tag{4.34}$$

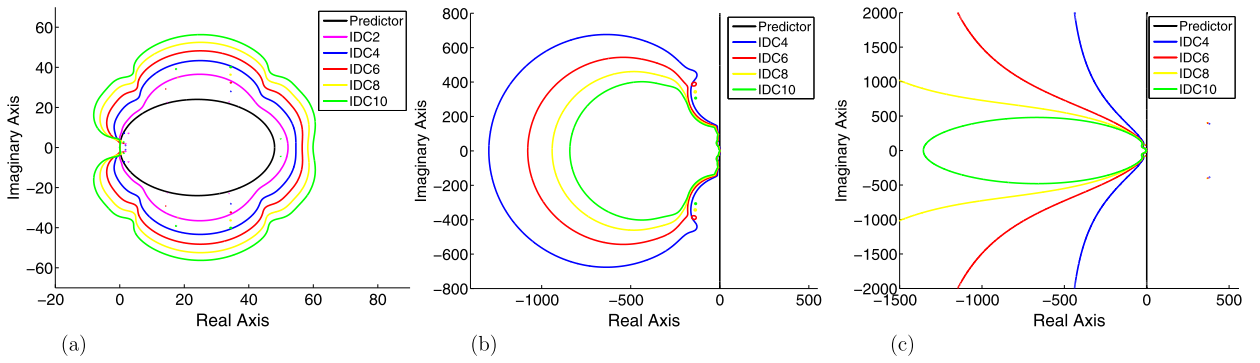

Fig. 4.1. Stability region for IDC-OS schemes with different number of corrections. (a) Lie–Trotter splitting; (b) Strang splitting; (c) ADI splitting.

Compare (4.33) and (4.34), we can conclude

$$\widetilde{\vartheta} = h^k \widetilde{\vartheta} + \mathcal{O}(h^\sigma). \tag{4.35}$$

Similar approach to the second equation in (4.26) and (4.27) will result in

$$\vartheta_{m+1}^{[k]} = h^k \bar{\vartheta}_{m+1}^{[k]} + \mathcal{O}(h^\sigma), \tag{4.36}$$

which completes the inductive proof for (4.32). □

4.3. Stability

In this subsection, we study the linear stability of the proposed IDC-OS numerical schemes. As is common practice [13], we consider the test problem

$$u_t = \lambda u, \tag{4.37}$$

and observe how the numerical scheme behaves for different complex values of λ. Without loss of generality, we will assume that $u(0) = 1$, and we'll consider a single time step of length $\Delta t = 1$. The stability region of a numerical method is then defined as

$$\mathbb{D} := \{\lambda \in \mathbb{C} : |u(1)| \leq 1\}. \tag{4.38}$$

An additional complication comes from the fact that an operator splitting scheme requires a splitting of the right hand side of (4.37) into Λ parts. For simplicity, we'll consider the special case of $\Lambda = 2$ with $\lambda = \lambda_1 + \lambda_2$, and we further assume that $\lambda_1 = \lambda_2$ for simplicity.

In Fig. 4.1 we present stability regions for IDC-OS methods based on three separate base solvers: Lie–Trotter splitting, Strang splitting and ADI splitting. The stability region of Lie–Trotter splitting with IDC procedure is everywhere outside the curves, and the stability regions for Strang splitting and ADI is the finite region inside the curves. The number in the legend denotes the order of the method. For example, "IDC4" represents fourth order methods achieved by the IDC-OS schemes; for Lie–Trotter splitting, we require three correctors to attain fourth-order accuracy, whereas Strang and ADI splitting only require a single correction. Our first observation is that of the three base solvers, Lie–Trotter splitting is the only solver that retains an infinite region of absolute stability, whereas Strang splitting and ADI reduce to finite regions of absolute stability.

Consistent with other *implicit* IDC methods, the stability regions for our implicit IDC-OS methods decreases as the number of correction steps increases. We have observed that larger stability regions can be found if we include more quadrature nodes for evaluating the integral of the residual.[1] This leads us to conjecture that a more accurate numerical approximation of the residual integral is important in finding larger stability regions.

5. Application of IDC-OS schemes to parabolic PDEs

In this section, we will discuss how to apply the IDC-OS framework to the parabolic problem of the form

$$\begin{cases} u_t = \nabla \cdot (a(x,y)\nabla u) + s(t,u), & (x,y) \in \Omega \\ u(0,x,y) = u_0(x,y), \\ u = g, & (x,y) \in \partial\Omega. \end{cases} \tag{5.1}$$

[1] For all simulations used in this work, we use 13 uniformly distributed interior nodes for evaluating the residual integral in the error equation.

The methods can be generalized to a high dimensional setting, but in this work we restrict our attention to two dimensions. For differential splitting methods, it is quite straightforward to apply IDC-OS schemes if we solve (5.1) via method of lines. One can obtain semi-discrete ODE systems which have the same form as (1.1) after spatial discretization. It is natural to assume one operator, say L_1 is related to the terms in x-direction, while L_2 is related to the terms in y-direction. As for algebraic splitting, one major difficulty for applying the IDC-OS framework to PDEs is how to handle the boundary and initial conditions for the error equation. In the following context, we will introduce one ADI formulation which can effectively deal with those issues. For simplicity, we only discuss the case when there is no nonlinear source in (5.1), i.e. $s(t,u)=0$.

Classical ADI starts by applying second-order Crank–Nicholson time discretization to the continuous PDE (5.1), this process produces a semi-discrete scheme

$$\frac{u^{n+1}-u^n}{\Delta t}=\frac{a}{2}(u_{xx}^{n+1}+u_{xx}^n)+\frac{a_x}{2}(u_x^{n+1}+u_x^n)+\frac{a}{2}(u_{yy}^{n+1}+u_{yy}^n)+\frac{a_y}{2}(u_y^{n+1}+u_y^n), \tag{5.2}$$

where $\Delta t=t^{n+1}-t^n$ is the time step, $a=a(x,y)$, $a_x=a_x(x,y)$ and $a_y=a_y(x,y)$. On a two dimensional structured mesh, we choose to use the central difference approximation (of orders 2, 4 or 6) for approximating the spatial operators $\frac{\partial}{\partial x^2}$, $\frac{\partial}{\partial x}$, $\frac{\partial}{\partial y^2}$ and $\frac{\partial}{\partial y}$, and we denote them by A_x, B_x, A_y, B_y, respectively. If the spatial discretization is performed on an $N_x\times N_y$ grid, there are $N_x\times N_y$ equations in the form of (5.2). We denote Υ as the unknowns in vector form, then we can write these $N_x\times N_y$ equations into matrix multiplication where boundary conditions are also incorporated,

$$\begin{aligned}\frac{\Upsilon^{n+1}-\Upsilon^n}{\Delta t}=&\frac{a}{2}(A_x\Upsilon^{n+1}+A_x\Upsilon^n)+\frac{a_x}{2}(B_x\Upsilon^{n+1}+B_x\Upsilon^n)+\frac{a}{2}(A_y\Upsilon^{n+1}+A_y\Upsilon^n)+\frac{a_y}{2}(B_y\Upsilon^{n+1}+B_y\Upsilon^n)\\&+\frac{a}{2}(g_{A_x}^{n+1}+g_{A_x}^n)+\frac{a_x}{2}(g_{B_x}^{n+1}+g_{B_x}^n)+\frac{a}{2}(g_{A_y}^{n+1}+g_{A_y}^n)+\frac{a_y}{2}(g_{B_y}^{n+1}+g_{B_y}^n)\,,\end{aligned} \tag{5.3}$$

where g_{A_x}, g_{B_x}, g_{A_y}, and g_{B_y} are the boundary terms. Notice that, different from [8], we enforce the boundary conditions strictly in the scheme. It is easy to verify that the method given in (5.3) is second order accurate in time. Specifically, if we use six order central difference for spatial derivatives such as in the numerical simulations, the local truncation error of (5.3) is $\mathcal{O}(\Delta t\Delta x^6+\Delta t^3)$. Denoting

$$\begin{aligned}J_1&=\frac{\Delta t}{2}(aA_x+a_xB_x),\\J_2&=\frac{\Delta t}{2}(aA_y+a_yB_y),\\S&=\frac{a}{2}(g_{A_x}^{n+1}+g_{A_x}^n)+\frac{a_x}{2}(g_{B_x}^{n+1}+g_{B_x}^n)+\frac{a}{2}(g_{A_y}^{n+1}+g_{A_y}^n)+\frac{a_y}{2}(g_{B_y}^{n+1}+g_{B_y}^n)\,,\end{aligned} \tag{5.4}$$

(5.3) is equivalent to

$$(I-J_1-J_2)\Upsilon^{n+1}=(I+J_1+J_2)\Upsilon^n+\Delta tS. \tag{5.5}$$

To set up an ADI scheme, we follow [8] by adding one term $J_1J_2\Upsilon^{n+1}$ to both sides of (5.5), which results in

$$(I-J_1-J_2+J_1J_2)\Upsilon^{n+1}=(I+J_1+J_2+J_1J_2)\Upsilon^n+J_1J_2(\Upsilon^{n+1}-\Upsilon^n)+\Delta tS. \tag{5.6}$$

Then it is straightforward to factor (5.6) as

$$(I-J_1)(I-J_2)\Upsilon^{n+1}=(I+J_1)(I+J_2)\Upsilon^n+J_1J_2(\Upsilon^{n+1}-\Upsilon^n)+\Delta tS. \tag{5.7}$$

Let us consider the second term on the right hand side of (5.7). Observe that

$$\Upsilon^{n+1}=\Upsilon^n+\mathcal{O}(\Delta t), \tag{5.8}$$

and that J_1 and J_2 both carry a Δt in them, we see that the term $J_1J_2(\Upsilon^{n+1}-\Upsilon^n)\sim\mathcal{O}(\Delta t^3)$. Hence, the second term on the right hand side of (5.7) is the same order as the truncation error, thus can be dropped. Therefore, the scheme reduces to

$$(I-J_1)(I-J_2)\Upsilon^{n+1}=(I+J_1)(I+J_2)\Upsilon^n+\Delta tS. \tag{5.9}$$

To solve (5.9), a two-step method was proposed in [9,30],

$$\begin{cases}(I-J_1)\tilde{\Upsilon}^{n+\frac{1}{2}}=(I+J_2)\Upsilon^n+\frac{\Delta t}{2}S, & \text{x-sweep},\\(I-J_2)\Upsilon^{n+1}=(1+J_1)\tilde{\Upsilon}^{n+\frac{1}{2}}+\frac{\Delta t}{2}S, & \text{y-sweep}.\end{cases} \tag{5.10}$$

However, to be symbolically consistent, symmetric and suited for IDC method, we choose to split the boundary values S in the following way,

$$\begin{cases} (I - J_1)\tilde{\Upsilon}^{n+\frac{1}{2}} = (I + J_2)\Upsilon^n + S_1, & \text{x-sweep,} \\ (I - J_2)\Upsilon^{n+1} = (1 + J_1)\tilde{\Upsilon}^{n+\frac{1}{2}} + S_2, & \text{y-sweep,} \end{cases} \tag{5.11}$$

with boundary terms defined as

$$\begin{cases} S_1 = \frac{\Delta t}{2}(a g_{A_x}^{n+1} + a_x g_{B_x}^{n+1} + a g_{A_y}^{n} + a_y g_{B_y}^{n}), \\ S_2 = \frac{\Delta t}{2}(a g_{A_x}^{n} + a_x g_{B_x}^{n} + a g_{A_y}^{n+1} + a_y g_{B_y}^{n+1}). \end{cases} \tag{5.12}$$

It should also be pointed out that the boundary values S are associated with boundary functions g at time $t = t_n$ and $t = t_{n+1}$, instead of $t_{n+\frac{1}{2}}$, therefore there is no error introduced from intermediate values $\tilde{\Upsilon}$ on the boundary. This is important for setting the boundary conditions when solving the error equation of IDC when we combine the ADI scheme with the IDC methodology. Because the Dirichlet boundary conditions of (5.1) are exact and accounted for in the formulation of the prediction, therefore, the boundary terms will not show up in the correction steps.

6. Numerical examples

In this section, we present numerical results for the proposed implicit IDC-OS schemes on a variety of examples of the parabolic initial–boundary value problem (5.1), where our aim is to demonstrate the efficiency of the proposed time-stepping methods. We begin with two linear examples of (5.1), and then present an example of the heat equation with a nonlinear forcing term. Our final two examples come from mathematical biology: the Fitzhugh–Nagumo reaction–diffusion model and the Schnakenberg model.

Our present work is in two dimensions, and every result is performed on a square domain with a Cartesian grid. We solve (5.1) using 6-th order central difference for the spatial discretization in order that the temporal error is dominant in the measured numerical error.

Example 1 *(Linear example: Dirichlet boundary conditions).* We solve initial boundary value problem (5.1) with constant coefficient $a(x, y) = 1$ in the domain $[-1, 1] \times [-1, 1]$. Initial condition is taken as $u_0(x, y) = (1 - y)e^x$ and time dependent boundary conditions are $g(x, y, t) = (1 - y)e^{t+x}$. Therefore, (5.1) has the exact solution $u(x, y, t) = (1 - y)e^{t+x}$. $N_{x,y} = N_x = N_y$ represents the number of spatial grids in x- and y-direction. N_t is the time steps used in the time interval $[0, T]$ where T is end time. c_s is the number of correction steps. u is the exact solution and υ as the numerical solution. We solve Example 1 by first order Lie–Trotter splitting (2.4), second order Strang splitting (2.6) and ADI splitting (5.11) and all the splitting performed via dimensional fashion, the numerical errors are shown in Table 6.1, Table 6.2 and Table 6.3 respectively. We can clearly conclude that the schemes achieve the designed order with IDC methodology, i.e. with one more correction step, the order of the scheme increases by 1 for Lie–Trotter splitting, while the order of the scheme increases by 2 for Strang splitting and ADI splitting.

Example 2 *(Linear example: periodic boundary conditions).* In this example, we will solve (5.1) when $a(x, y) = 2 + 0.5\sin(\pi(4x + y))$, and the initial condition $u_0(x, y) = \sin(2\pi(x + y))$. We compute errors using the difference between two successive refinements:

$$\text{error} = \|\upsilon_{N_t} - \upsilon_{\frac{N_t}{2}}\|_\infty, \tag{6.1}$$

where N_t describes the number of time steps.

Table 6.1
Linear example with Dirichlet boundary conditions. Errors $\|u - \upsilon_{N_t}\|_\infty$ for Lie–Trotter splitting method, $T = 0.025$.

$N_{x,y} = 45$	Number of time steps N_t							
Correction	$N_t = 60$	Order	$N_t = 80$	Order	$N_t = 100$	Order	$N_t = 120$	Order
$c_s = 0$	1.53e−5	–	1.15e−5	0.99	9.24e−6	0.98	7.70e−6	1.00
$c_s = 1$	1.90e−7	–	1.09e−7	1.93	7.08e−8	1.93	4.94e−8	1.97
$c_s = 2$	3.10e−9	–	1.47e−9	2.59	8.06e−10	2.69	4.87e−10	2.76

Table 6.2
Linear example with Dirichlet boundary conditions. Errors $\|u - \upsilon_{N_t}\|_\infty$ for Strang splitting method, $T = 0.025$.

$N_{x,y} = 45$	Number of time steps N_t							
Correction	$N_t = 60$	Order	$N_t = 80$	Order	$N_t = 100$	Order	$N_t = 120$	Order
$c_s = 0$	3.02e−5	–	1.69e−5	2.02	1.08e−5	2.01	7.55e−6	1.96
$c_s = 1$	7.15e−7	–	2.45e−7	3.72	1.04e−7	3.84	5.20e−8	3.80
$c_s = 2$	3.64e−10	–	8.06e−11	5.24	2.28e−11	5.66	7.16e−12	6.35

Table 6.3
Linear example with Dirichlet boundary conditions. Errors $\|u - v_{N_t}\|_\infty$ for IDC-OS based on ADI splitting, $T = 0.025$.

$N_{x,y} = 150$	Number of time steps N_t							
Correction	$N_t = 60$	Order	$N_t = 80$	Order	$N_t = 100$	Order	$N_t = 120$	Order
$c_s = 0$	3.68e−5	–	2.07e−5	2.00	1.32e−5	2.00	9.20e−6	2.00
$c_s = 1$	4.49e−7	–	2.08e−7	2.67	1.03e−7	3.13	5.07e−8	3.91
$c_s = 2$	1.18e−7	–	1.69e−8	6.74	4.74e−9	5.71	1.59e−9	5.98

Table 6.4
Linear example with periodic boundary conditions. Errors $\|v_{N_t} - v_{\frac{N_t}{2}}\|_\infty$ for Lie–Trotter splitting method, $T = 0.025$.

$N_{x,y} = 45$	Number of time steps N_t							
Correction	$N_t = 40$	Order	$N_t = 80$	Order	$N_t = 160$	Order	$N_t = 320$	Order
$c_s = 0$	4.65e−3	–	2.35e−3	0.98	1.18e−3	0.99	5.94e−4	0.99
$c_s = 1$	1.85e−4	–	5.68e−5	1.70	1.63e−5	1.80	4.44e−6	1.88
$c_s = 2$	3.47e−6	–	6.55e−7	2.41	1.19e−7	2.46	1.88e−8	2.66

Table 6.5
Linear example with periodic boundary conditions. Errors $\|v_{N_t} - v_{\frac{N_t}{2}}\|_\infty$ for Strang splitting method, $T = 0.025$.

$N_{x,y} = 45$	Number of time steps N_t							
Correction	$N_t = 40$	Order	$N_t = 80$	Order	$N_t = 160$	Order	$N_t = 320$	Order
$c_s = 0$	5.24e−5	–	1.31e−5	2.00	3.29e−6	1.99	8.22e−7	2.00
$c_s = 1$	3.30e−9	–	2.06e−10	4.00	1.29e−11	4.00	8.04e−13	4.00
$c_s = 2$	5.80e−12	–	4.90e−14	6.89	7.77e−16	5.98	1.11e−16	2.81

Table 6.6
Linear example with periodic boundary conditions. Errors $\|v_{N_t} - v_{\frac{N_t}{2}}\|_\infty$ for ADI splitting method, $T = 0.05$.

$N_{x,y} = 200$	Number of time steps N_t							
Correction	$N_t = 40$	Order	$N_t = 80$	Order	$N_t = 160$	Order	$N_t = 320$	Order
$c_s = 0$	7.77e−5	–	1.94e−5	2.00	4.85e−6	2.00	1.21e−6	2.00
$c_s = 1$	1.93e−8	–	1.20e−9	4.00	7.52e−11	4.00	4.70e−12	4.00
$c_s = 2$	1.43e−11	–	2.23e−13	6.01	3.56e−15	5.96	1.04e−16	5.10

Again, we present results using three splitting options: Lie–Trotter, Strang and ADI splitting. Convergence studies are presented in Tables 6.4, 6.5 and 6.6, and we can also observe that the schemes achieve the designed order. Note that the numerical error for the two correctors when $N_t = 320$ is not reliable in the cases of Strang and ADI splitting because of precision limitation.

Example 3 *(Nonlinear equation with Dirichlet boundary conditions).* We now test the proposed IDC-OS methods on a nonlinear example of (5.1) with a known exact solution.

$$\begin{cases} u_t = u_{xx} + u_{yy} - u^2 + e^{-2t}\cos^2(\pi x)\cos^2(\pi y) + (2\pi^2 - 1)e^{-t}\cos(\pi x)\cos(\pi y), \\ u(0, x, y) = \cos(\pi x)\cos(\pi y), \end{cases} \tag{6.2}$$

on the domain $(x, y) \in [-1, 1] \times [-1, 1]$. The exact solution to this problem is $u(x, y, t) = e^{-t}\cos(\pi x)\cos(\pi y)$. Given that we have an exact solution, all our numerical tests use exact boundary conditions from this solution.

An IDC-OS solver for (6.2) requires a definition for how the splitting will be performed. Here, we choose to split the problem into three pieces: L_1 and L_2 are the same as linear case, while L_3 contains the remaining non-linear terms,

$$L_3(t, u) = -u^2 + e^{-2t}\cos^2(\pi x)\cos^2(\pi y) + (2\pi^2 - 1)e^{-t}\cos(\pi x)\cos(\pi y). \tag{6.3}$$

We use Newton iteration to solve the discretized version of $u_t = L_3(t, u)$.

In Tables 6.7 and 6.8 we present results from applying the IDC-OS method with Lie–Trotter and Strang splitting as the base solvers. In each case, we can see the successful increase of order after each correction: one in the case of Lie–Trotter splitting, and two in the case of Strang splitting.

In this work, we do not present results for IDC-OS methods based on ADI splitting for non-linear problems due to their computational complexity. The high-order differential operator splitting methods discussed here are much simpler than what would arise from using even low-order ADI splitting.

Table 6.7
Nonlinear example with Dirichlet boundary conditions. Errors $\|u - v_{N_t}\|_\infty$ for IDC-OS based on Lie–Trotter splitting, $T = 0.025$.

$N_{x,y} = 45$	Number of time steps N_t							
Correction	$N_t = 60$	Order	$N_t = 80$	Order	$N_t = 100$	Order	$N_t = 120$	Order
$c_s = 0$	6.88e−3	–	5.16e−3	1.00	4.13e−3	1.00	3.44e−3	1.00
$c_s = 1$	7.31e−4	–	4.33e−4	1.82	2.87e−4	1.84	2.03e−4	1.90
$c_s = 2$	1.60e−5	–	6.95e−6	2.90	3.59e−6	2.96	2.06e−6	3.05

Table 6.8
Nonlinear example with Dirichlet boundary conditions. Errors $\|u - v_{N_t}\|_\infty$ for IDC-OS based on Strang splitting, $T = 0.01$.

$N_{x,y} = 100$	Number of time steps N_t							
Correction	$N_t = 60$	Order	$N_t = 80$	Order	$N_t = 100$	Order	$N_t = 120$	Order
$c_s = 0$	9.21e−5	–	5.20e−5	1.99	3.34e−5	1.98	2.33e−5	1.99
$c_s = 1$	3.04e−6	–	8.96e−7	4.25	3.38e−7	4.37	1.56e−7	4.24
$c_s = 2$	1.87e−8	–	3.22e−9	6.11	8.33e−10	6.06	2.84e−10	5.90

Example 4 *(Fitzhugh–Nagumo reaction–diffusion model).* A simple mathematical model of an excitable medium is Fitzhugh–Nagumo (FHN) equations [11]. FHN equations with diffusion can be written as

$$\begin{cases} \dfrac{\partial u}{\partial t} = D_u \nabla^2 u + \dfrac{1}{\delta} h(u, v), \\ \dfrac{\partial v}{\partial t} = D_v \nabla^2 v + g(u, v), \end{cases} \tag{6.4}$$

where D_u, D_v are the diffusion coefficients for activator u and inhibitor v respectively, and δ is a real parameter. We consider the classical cubic FHN local dynamics [19,29]

$$\begin{cases} h(u, v) = Cu(1 - u)(u - a) - v, \\ g(u, v) = u - dv, \end{cases} \tag{6.5}$$

where C, a and d are dimensionless parameters. We perform the numerical experiment for (6.4) and (6.5) on the domain $[-20, 20] \times [-20, 20]$ with periodic boundary conditions. The parameters are chosen as follows, $D_u = 1$, $D_v = 0$, $a = 0.1$, $C = 1$, $d = 0.5$, and $\delta = 0.005$. The initial condition is

$$u(x, y, 0) = \begin{cases} 0, & \text{if } \{x < 0\} \bigcup \{y > 5\}; \\ \dfrac{1}{(1 + e^{4(|x|-5)})^2} - \dfrac{1}{(1 + e^{4(|x|-1)})^2}, & \text{otherwise.} \end{cases} \tag{6.6}$$

$$v(x, y, 0) = \begin{cases} 0.15, & \text{if } \{x < 1\} \bigcap \{y > -10\}; \\ 0, & \text{otherwise.} \end{cases} \tag{6.7}$$

The domain is partitioned with a 200×200 grid. Fig. 6.1 shows the numerical solution to the concentration of the activator u solving by Lie–Trotter splitting scheme, with three operators similar as Example 3. We observe the spiral waves at $T = 2, 5, 10$, which show a good agreement with the reference solutions. The computational step size is $\Delta t = 0.005$ in all cases. We remark that, because the lower order schemes suffer more from the numerical error of diffusion than higher order ones, the pattern for $T = 10$ looks more consistent if we take a smaller computational time step size or a more refined mesh. Similar patterns can also be obtained by IDC-OS scheme based on Strang splitting.

Remark. Fig. 6.2 shows the order of accuracy for Fitzhugh–Nagumo reaction–diffusion model (6.4) at $t = 0.025$, in which we clearly observe the order increase of IDC-OS scheme with successive correction steps. However, for a more stiff parameter such as $\delta = 10^{-10}$, order reduction phenomena is observed in the convergence study. Similar observation is made in Example 5 on Schnakenberg model. How to approximate the residual integral and design a robust solver for stiff ODEs is an open question. Recently in [20], the authors proposed a highly accurate solver based on an approximation of the integral of the residual as a linear combination of exponentials on uniform quadrature nodes. Their method is shown to do a good job of attaining high order of accuracy with correction steps and preserving the stability region of the original implicit time integrator which is used as the base scheme.

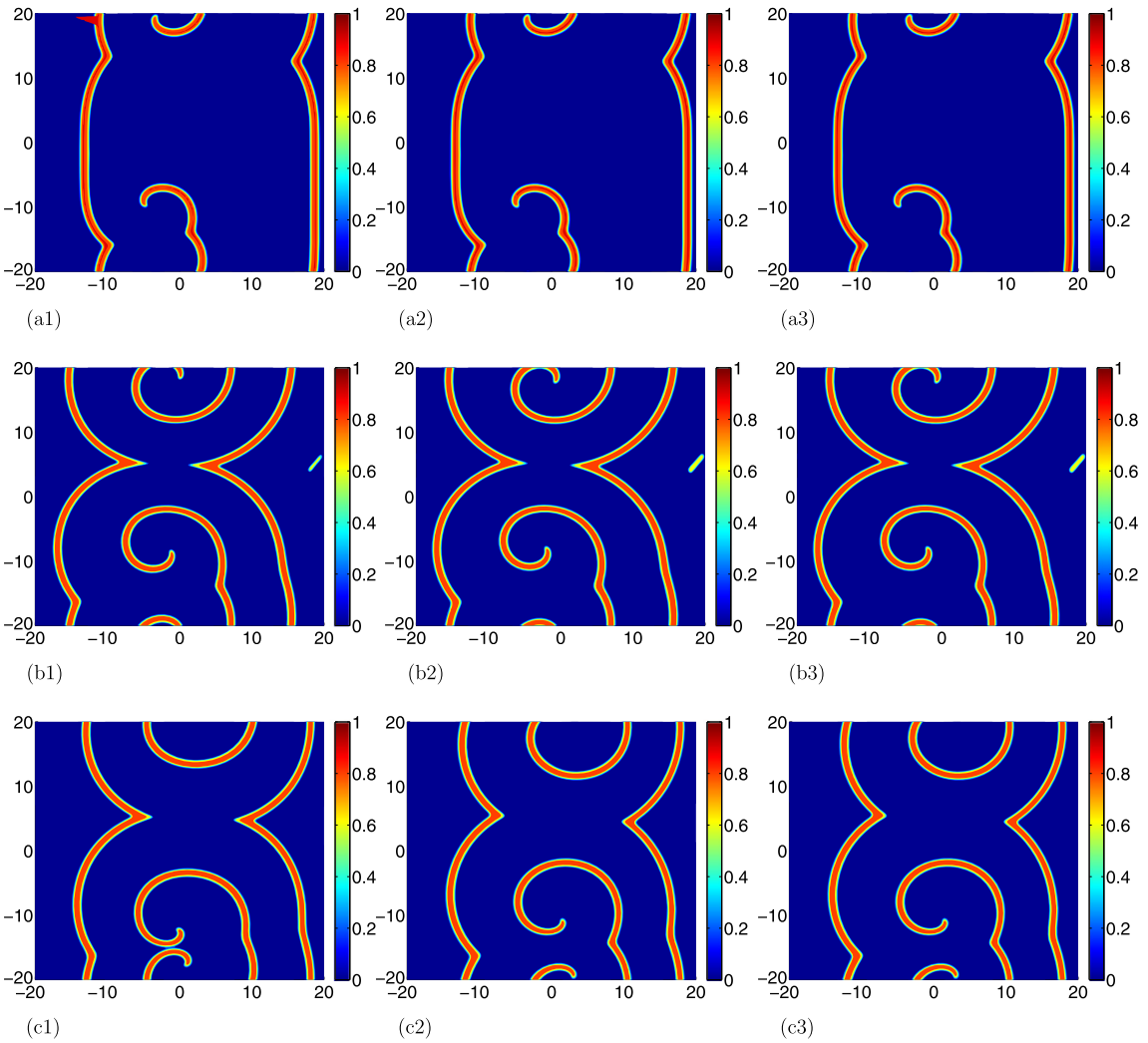

Fig. 6.1. Numerical simulations of the concentration of activator u for Fitzhugh–Nagumo reaction–diffusion model at different times. (a1)–(a3) $t = 2$; (b1)–(b3) $t = 5$; (c1)–(c3) $t = 10$. (a1)–(c1) Lie–Trotter without corrector; (a2)–(c2) Lie–Trotter with two correctors; (a3)–(c3) Lie–Trotter with three correctors.

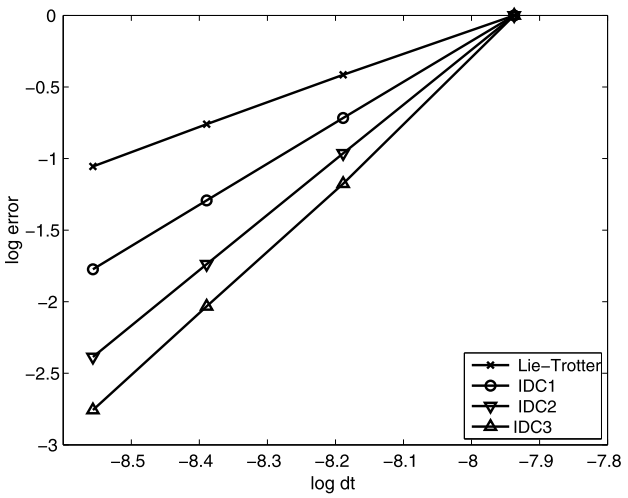

Fig. 6.2. Accuracy study for Fitzhugh–Nagumo reaction–diffusion model. $t = 0.025$.

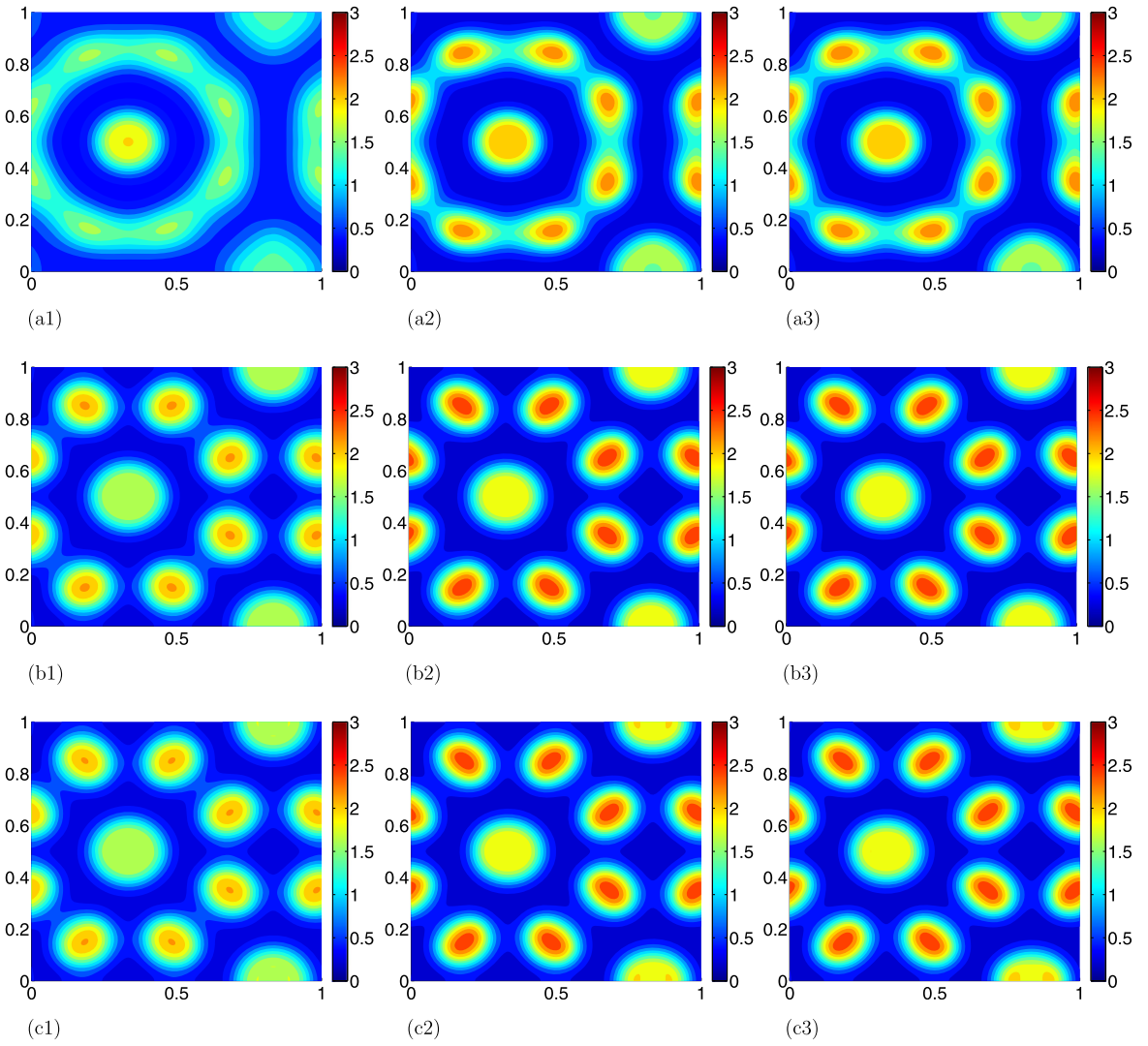

Fig. 6.3. Numerical simulations of the concentration of activator C_a for Schnakenberg reaction–diffusion model at different times. (a1)–(a3) $t = 0.5$; (b1)–(b3) $t = 1$; (c1)–(c3) $t = 1.5$. (a1)–(c1) Lie–Trotter without corrector; (a2)–(c2) Lie–Trotter with one corrector; (a3)–(c3) Lie–Trotter with two correctors.

Example 5 *(Schnakenberg model).* The Schnakenberg system [31] has been used to model the spatial distribution of a morphogen. It has the following form

$$\begin{cases} \dfrac{\partial C_a}{\partial t} = D_1 \nabla^2 C_a + \kappa(a - C_a + C_a^2 C_i), \\ \dfrac{\partial C_i}{\partial t} = D_2 \nabla^2 C_i + \kappa(b - C_a^2 C_i), \end{cases} \tag{6.8}$$

where C_a and C_i represent the concentrations of activator and inhibitor, with D_1 and D_a as the diffusion coefficients respectively. κ, a and b are rate constants of biochemical reactions. Following the setup in [17], we take the initial conditions as

$$C_a(x, y, 0) = a + b + 10^{-3} e^{-100((x-\frac{1}{3})^2 + (y-\frac{1}{2})^2)}, \tag{6.9}$$

$$C_i(x, y, 0) = \frac{b}{(a+b)^2}, \tag{6.10}$$

and the boundary conditions are periodic. The parameters are $\kappa = 100$, $a = 0.1305$, $b = 0.7695$, $D_1 = 0.05$ and $D_2 = 1$. The computational domain is $[0, 1] \times [0, 1]$. The numerical simulations with Lie–Trotter splitting is performed on a 200×200 spatial grid and the numerical dynamical process of the concentration of the activator C_a at different times are shown in Fig. 6.3, we can observe that the initial data are amplified and spreads, leading to thew formation of spot pattern. The

computational time step size is chosen as $\Delta t = 0.001$. We also note that similar patterns can be obtained by IDC-OS scheme based on Strang splitting.

7. Conclusion

In this paper, we have provided a general temporal framework for the construction of high order operator splitting methods based on the integral deferred correction procedure. The method can achieve arbitrary high order via solving correction equation whereas reduce the computational cost by taking the advantage of operator splitting. Error analysis and numerical examples for IDC-OS methods are performed to show that the proposed IDC framework successfully enhances the order of accuracy in time. A study on order reduction for very stiff problems will be part of our future work.

Acknowledgements

AJC is supported in part by AFOSR grants FA9550-11-1-0281, FA9550-12-1-0343 and FA9550-12-1-0455, NSF grant DMS-1115709, and MSU Foundation grant SPG-RG100059. ZX is supported by NSF grant DMS-1316662. Additionally, the authors would like to thank Prof. William Hitchon and Dr. David Seal for helpful comments on this work.

References

[1] K. Bagrinovskii, S. Godunov, Difference schemes for multidimensional problems, Dokl. Akad. Nauk 115 (1957) 431–433 (in Russian).
[2] K. Bohmer, H. Stetter, Defect Correction Methods. Theory and Applications, Springer-Verlag, Wien, 1984.
[3] A. Bourlioux, A. Layton, M. Minion, High-order multi-implicit spectral deferred correction methods for problems of reactive flow, J. Comput. Phys. 189 (2) (2003) 651–675.
[4] A. Christlieb, M. Morton, B. Ong, J.-M. Qiu, Semi-implicit integral deferred correction constructed with additive Runge–Kutta methods, Commun. Math. Sci. 9 (2011) 879–902.
[5] A. Christlieb, B. Ong, J.-M. Qiu, Comments on high order integrators embedded within integral deferred correction methods, Commun. Appl. Math. Comput. Sci. 4 (1) (2009) 27–56.
[6] A. Christlieb, B. Ong, J.-M. Qiu, Integral deferred correction methods constructed with high order Runge–Kutta integrators, Math. Comput. 79 (270) (2010) 761–783.
[7] J. Douglas, On the numerical integration of $\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = \frac{\partial u}{\partial t}$ by implicit methods, J. Soc. Ind. Appl. Math. 3 (1) (1955) 42–65.
[8] J. Douglas, S. Kim, On accuracy of alternating direction implicit methods for parabolic equations, Preprint, 1999.
[9] J. Douglas, D. Peaceman, Numerical solution of two-dimensional heat-flow problems, AIChE J. 1 (4) (1955) 505–512.
[10] A. Dutt, L. Greengard, V. Rokhlin, Spectral deferred correction methods for ordinary differential equations, BIT Numer. Math. 40 (2) (2000) 241–266.
[11] P.C. Fife, et al., Mathematical Aspects of Reacting and Diffusing Systems, Springer-Verlag, 1979.
[12] J. Geiser, Higher-order difference and higher-order splitting methods for 2d parabolic problems with mixed derivatives, Int. Math. Forum 2 (67) (2007) 3339–3350.
[13] E. Hairer, S. Nørsett, G. Wanner, Solving Ordinary Differential Equations, vol. 2, Springer, 1991.
[14] E. Hansen, A. Ostermann, High order splitting methods for analytic semigroups exist, BIT Numer. Math. 49 (3) (2009) 527–542.
[15] J. Huang, J. Jia, M. Minion, Accelerating the convergence of spectral deferred correction methods, J. Comput. Phys. 214 (2) (2006) 633–656.
[16] J. Huang, J. Jia, M. Minion, Arbitrary order Krylov deferred correction methods for differential algebraic equations, J. Comput. Phys. 221 (2) (2007) 739–760.
[17] W. Hundsdorfer, J. Verwer, Numerical Solution of Time-Dependent Advection–Diffusion–Reaction Equations, vol. 33, Springer, 2003.
[18] H. Jia, K. Li, A third accurate operator splitting method, Math. Comput. Model. 53 (1) (2011) 387–396.
[19] J. Keener, J. Sneyd, Mathematical Physiology I: Cellular Physiology, vol. 1, Springer, 2010.
[20] D. Kushnir, V. Rokhlin, A highly accurate solver for stiff ordinary differential equations, SIAM J. Sci. Comput. 34 (3) (2012) A1296–A1315.
[21] A. Layton, On the choice of correctors for semi-implicit Picard deferred correction methods, Appl. Numer. Math. 58 (6) (2008) 845–858.
[22] A. Layton, M. Minion, Implications of the choice of quadrature nodes for Picard integral deferred corrections methods for ordinary differential equations, BIT Numer. Math. 45 (2) (2005) 341–373.
[23] A. Layton, M. Minion, Implications of the choice of predictors for semi-implicit Picard integral deferred correction methods, Commun. Appl. Math. Comput. Sci. 2 (1) (2007) 1–34.
[24] Y. Liu, C.-W. Shu, M. Zhang, Strong stability preserving property of the deferred correction time discretization, J. Comput. Math. 26 (5) (2008) 633–656.
[25] G. Marchuk, Some application of splitting-up methods to the solution of mathematical physics problems, Apl. Mat. 13 (2) (1968) 103–132.
[26] R. McLachlan, G.R. Quispel, Splitting methods, Acta Numer. 11 (2002) 341–434.
[27] R. McLachlan, G.R. Quispel, Geometric integrators for odes, J. Phys. A, Math. Gen. 39 (19) (2006) 5251.
[28] M. Minion, Semi-implicit spectral deferred correction methods for ordinary differential equations, Commun. Math. Sci. 1 (3) (2003) 471–500.
[29] D. Olmos, B. Shizgal, Pseudospectral method of solution of the Fitzhugh–Nagumo equation, Math. Comput. Simul. 79 (7) (2009) 2258–2278.
[30] D.W. Peaceman, H.H. Rachford Jr., The numerical solution of parabolic and elliptic differential equations, J. Soc. Ind. Appl. Math. 3 (1) (1955) 28–41.
[31] J. Schnakenberg, Simple chemical reaction systems with limit cycle behaviour, J. Theor. Biol. 81 (3) (1979) 389–400.
[32] G. Strang, On the construction and comparison of difference schemes, SIAM J. Numer. Anal. 5 (3) (1968) 506–517.
[33] G. Strang, Approximating semigroups and the consistency of difference schemes, Proc. Am. Math. Soc. 20 (1) (1969) 1–7.
[34] M. Thalhammer, High-order exponential operator splitting methods for time-dependent Schrödinger equations, SIAM J. Numer. Anal. 46 (4) (2008) 2022–2038.