DISCRETE AND CONTINUOUS
DYNAMICAL SYSTEMS SERIES B
Volume **24**, Number **12**, December **2019**

doi:10.3934/dcdsb.2019144

pp. **6387–6417**

A HYBRID METHOD FOR STIFF REACTION–DIFFUSION EQUATIONS

Yuchi Qiu

Department of Mathematics, University of California, Irvine
Irvine, CA 92697, USA

Weitao Chen

Department of Mathematics, University of California, Riverside
Riverside, CA 92507, USA

Qing Nie*

Department of Mathematics, Department of Developmental and Cell Biology
University of California, Irvine
Irvine, CA 92697, USA

(Communicated by Lei Zhang)

Abstract. The second-order implicit integration factor method (IIF2) is effective at solving stiff reaction–diffusion equations owing to its nice stability condition. IIF has previously been applied primarily to systems in which the reaction contained no explicitly time-dependent terms and the boundary conditions were homogeneous. If applied to a system with explicitly time-dependent reaction terms, we find that IIF2 requires prohibitively small time-steps, that are relative to the square of spatial grid sizes, to attain its theoretical second-order temporal accuracy. Although the second-order implicit exponential time differencing (iETD2) method can accurately handle explicitly time-dependent reactions, it is more computationally expensive than IIF2. In this paper, we develop a hybrid approach that combines the advantages of both methods, applying IIF2 to reaction terms that are not explicitly time-dependent and applying iETD2 to those which are. The second-order hybrid IIF-ETD method (hIFE2) inherits the lower complexity of IIF2 and the ability to remain second-order accurate in time for large time-steps from iETD2. Also, it inherits the unconditional stability from IIF2 and iETD2 methods for dealing with the stiffness in reaction–diffusion systems. Through a transformation, hIFE2 can handle nonhomogeneous boundary conditions accurately and efficiently. In addition, this approach can be naturally combined with the compact and array representations of IIF and ETD for systems in higher spatial dimensions. Various numerical simulations containing linear and nonlinear reactions are presented to demonstrate the superior stability, accuracy, and efficiency of the new hIFE method.

2010 *Mathematics Subject Classification.* Primary: 65M06, 35K57; Secondary: 65M12.

Key words and phrases. Implicit integration factor methods, exponential time differencing methods, reaction–diffusion equations, explicitly time-dependent reaction, nonhomogeneous boundary conditions.

* Corresponding author: Qing Nie.

1. **Introduction.** Consider a reaction–diffusion system

$$\mathbf{u}_t = D\Delta\mathbf{u} + \mathbf{f}(\mathbf{u}, \mathbf{x}, t), \quad \mathbf{x} \in \Omega \subset \mathbb{R}^k, \ t \in [0, T], \tag{1}$$

where $\mathbf{u} = \mathbf{u}(\mathbf{x}, t) \in \mathbb{R}^m$, $D \in \mathbb{R}^{m\times m}$ is the diffusion coefficient matrix, and $\mathbf{f}(\mathbf{u}, \mathbf{x}, t)$ describes the reactions. In biology, reaction–diffusion equations have been used to model predator–prey interactions [35, 10, 11], the formation of Turing patterns in organs or tissues [42, 12], stochastic dynamics in gene networks [37], and fetal and adult dermal wound healing [5]. In ecology, they have been applied to study population dynamics [17, 36, 1]. In finance, the estimation of option prices under several risk factors can be represented by reaction–diffusion systems as well [53]. While the reaction terms in these applications are often autonomous, i.e. $\mathbf{f}(\mathbf{u}, \mathbf{x}, t) = \mathbf{h}(\mathbf{u})$, in morphogen gradients systems in biology [8, 49, 46, 24, 26, 50], the reactions contain explicitly time-dependent terms.

Many numerical methods have been developed to solve (1). Typically, finite difference or finite element methods are used to approximate the equation in space, equipped with some time integration method. To ensure stability, classic explicit methods require a time step $\Delta t \sim \Delta x^2$ [27, 32, 41]. To relax this severe stability restriction, many other schemes have been developed, such as exponential time differencing (ETD) and semi-implicit integration factor (IIF) methods. In both ETD and IIF methods, (1) is written into a system of ordinary differential equations (ODEs) by applying the spatial discretization,

$$U_t = AU + F(U, t), \tag{2}$$

where $U = U(t)$ is the spatially discretized form of $\mathbf{u}$, and AU is the finite difference approximation of the diffusion term $D\Delta\mathbf{u}$. By using the integration factor e^{-At} to integrate (2) from t_n to t_{n+1} exactly, i.e.,

$$\begin{aligned} U(t_{n+1}) &= e^{A\Delta t}U(t_n) + e^{A\Delta t}\int_0^{\Delta t} e^{-A\tau}F\big(U(t_n+\tau), t_n+\tau\big)\,d\tau, \\ &\triangleq e^{A\Delta t}U(t_n) + \tilde{F}_n, \end{aligned} \tag{3}$$

the stability constraint due to the diffusion is removed, and the problem becomes one of approximating the integral $\tilde{F}_n$.

In ETD, $\tilde{F}_n$ is approximated by integrating the product of $e^{-A\tau}$ and the interpolated $F(U, t_n + \tau)$ [2, 16]. All explicit ETD (eETD) methods are not unconditionally linearly stable; they require prohibitively small time steps to solve stiff systems [25]. To help improve stability, Runge–Kutta-type methods are sometimes employed [4, 14, 15, 23], and several methods have been developed through splitting of the linear diffusion operator [6, 7, 22, 52]. An unconditionally linearly stable (A-stable) method is the implicit second-order ETD method (iETD2). Although iETD2 is A-stable, it has the drawback of high computational cost to solve the implicit equations at each step [34].

In IIF, instead of only interpolating $F(U, t_n + \tau)$, the approximation of $\tilde{F}_n$ is obtained by interpolating $e^{-A\tau}F(U, t_n + \tau)$ [34]. The second-order IIF method (IIF2) is A-stable like iETD2, but it has the advantage that the nonlinear equations at each step are much cheaper to solve. Owing to its good stability and reduced computational complexity compared with iETD2, IIF2 is more suitable for solving stiff reaction–diffusion equations. For high-dimensional systems, compact and array representations of IIF [33, 45] are effective in reducing the storage and computational cost associated with the exponential matrix, along with the flexibility

to handle non-constant diffusion coefficients or cross-derivatives. Furthermore, by incorporating the sparse grids technique [44, 29], the IIF method can be applied to multi-dimensional systems with better efficiency. IIF has also been extended to treat fourth-order parabolic equations [21, 31], reaction–diffusion–advection equations [51], and stochastic differential equations [40], and it has been combined with adaptive meshes [28].

In both ETD and IIF methods, the high cost of computing exponential matrices is challenging. To speed up the computation, the discrete fast Fourier transformation (FFT)-based algorithms were adopted in both ETD and IIF methods [43, 48, 22]. For non-constant diffusion coefficients, Krylov-ETD and Krylov-IIF methods [39, 3, 19, 20, 30] were developed to reduce the computational cost and storage associated with exponential matrices by utilizing Krylov subspace approximation [38, 9, 13].

Although the IIF2 method is theoretically second order in time [34], when it is applied to explicitly time-dependent reactions, extremely small time steps compared to spatial grid sizes, $\Delta t \leq \mathcal{O}(\Delta x^2)$, are required to observe second-order temporal error. Above some critical threshold for Δt, the observed error in IIF2 is only first order. In contrast, iETD2 remains second order in time for larger Δt, especially for finer spatial discretization. When applied to nonhomogeneous boundary conditions, IIF2 also requires small Δt to retain second-order accuracy because the nonhomogeneous boundary conditions can be interpreted as introducing a large time-dependent reaction to the semi-discrete form in (2) [22]. In an attempt to construct a method that can deal with both explicitly time-dependent reactions and nonhomogeneous boundary conditions efficiently, the fast explicit integration factor (fEIF) method [22] can retain its theoretical order of accuracy under large Δt like iETD2 and has low computational cost like IIF2. One of the drawbacks of fEIF is that, as a conditionally stable method, it has strict time step constraint when reactions are stiff.

In this paper, we introduce a new hybrid method which combines the IIF and iETD methods. The hybrid IIF-ETD method (hIFE) is composed in such a way as to inherit the advantages of both methods simultaneously: that is, to retain second-order accuracy for large time steps with time-dependent reactions like iETD and to reduce computational complexity in each iteration like IIF. In addition, in contrast to fEIF2, the second-order hIFE method (hIFE2) inherits A-stability from its constituents. We also introduce a procedure to more easily accommodate nonhomogeneous boundary conditions by transforming the system into one with homogeneous boundary conditions, using hIFE on the transformed system. Combining the transformation with hIFE2 provides a framework for an A-stable, efficient numerical method that remains effectively second-order accurate in time in the presence of time-dependent terms and nonhomogeneous boundary conditions.

The paper is organized as follows. In Section 2, we introduce the IIF2, iETD2, and new hIFE2 methods and explore and compare the order of accuracy of hIFE2 with that of IIF2 and iETD2 for explicitly time-dependent, autonomous, and mixed reactions in both scalar and semi-discrete form. In Section 3, we apply hIFE2 to systems with nonhomogeneous boundary conditions and introduce a transformation to better treat these boundary conditions. In Section 4, we demonstrate an extension of the hIFE2 method to high-dimensional problems. In Section 5, we provide multiple numerical tests to demonstrate the accuracy, efficiency, and stability of hIFE. Finally, in Section 6, we give some concluding remarks. In Appendix A, we prove that IIF2 displays first order when the time step is large. In Appendix B, we

provide the complexity analysis. In Appendix C, we introduce the exponential-like matrices formation.

2. **Temporal error analysis.** The defining feature of each of the methods discussed in this paper is how the integral $\tilde{F}_n$ in (3) is approximated. Both ETD and IIF approximate $\tilde{F}_n$ using Lagrange interpolation [18]. In ETD, only the reaction $F(U, t_n+\tau)$ is interpolated, yielding a polynomial $p(\tau)$. Then the product $e^{-A\tau}p(\tau)$ is integrated exactly [2, 16]. The approximation for iETD2 is thus given by

$$\tilde{F}_n \approx \frac{I + (-I + A\Delta t)e^{A\Delta t}}{A^2\Delta t} F_n + \frac{(-I - A\Delta t) + e^{A\Delta t}}{A^2\Delta t} F_{n+1}, \tag{4}$$

where $F_n \triangleq F(U(t_n), t_n)$. In IIF, instead of only interpolating $F(U, t_n+\tau)$, the approximation of $\tilde{F}_n$ is obtained by interpolating the entire integrand, $e^{-A\tau}F(U, t_n + \tau) \approx q(\tau)$, and integrating $q(\tau)$ exactly [34]. IIF2 approximates

$$\tilde{F}_n \approx \frac{\Delta t}{2}\big(e^{A\Delta t}F_n + F_{n+1}\big). \tag{5}$$

We show in Section 2.1 that, when applied to time-dependent reactions, IIF2 requires extremely small Δt to exhibit second-order behavior while iETD2 remains second order for large time steps. Motivated by this analysis, we define for our hIFE method a splitting of the reaction term

$$F(U, t) = [F(U, t) - F(0, t)] + [F(0, t)] \triangleq F_1(U, t) + F_2(t) \tag{6}$$

in (3) and apply IIF on $F_1(U, t)$ and iETD on $F_2(t)$. The second-order hybrid method (**hIFE2**) thus approximates

$$\begin{aligned}\tilde{F}_n &= e^{A\Delta t}\int_0^{\Delta t} e^{-A\tau}F_1\big(U(t_n+\tau), t_n+\tau\big)\,d\tau + e^{A\Delta t}\int_0^{\Delta t} e^{-A\tau}F_2\big(t_n+\tau\big)\,d\tau\\ &\approx \frac{\Delta t}{2}\big(e^{A\Delta t}(F_1)_n + (F_1)_{n+1}\big) + \frac{I + (-I + A\Delta t)e^{A\Delta t}}{A^2\Delta t}(F_2)_n\\ &\quad + \frac{(-I - A\Delta t) + e^{A\Delta t}}{A^2\Delta t}(F_2)_{n+1}.\end{aligned} \tag{7}$$

In Section 2.1, we perform an analysis of the temporal error in using each method to solve (2) with the operator A replaced by a scalar α for explicitly time-dependent, autonomous, and mixed reactions. Then in Section 2.2, we investigate the differences between the scalar and semi-discrete form, showing that iETD2 and hIFE2 remain effectively second order for large Δt while IIF2 does not.

2.1. **Accuracy in scalar form.** To compare the error associated with each method above, we first consider the scalar form of (2), where A is replaced by the scalar α:

$$\begin{cases} u_t = \alpha u + f(u, t), \quad t \in [0, T],\\ u(0) = v. \end{cases} \tag{8}$$

By making use of the Taylor expansions

$$\frac{1 + (-1 + \alpha\Delta t)e^{\alpha\Delta t}}{\alpha^2\Delta t} = \Delta t\sum_{k=0}^{\infty}\frac{1}{(k+2)k!}(\alpha\Delta t)^k, \quad f(t_{n+1}) = \sum_{k=0}^{\infty}\frac{1}{k!}\Delta t^k\frac{d^k}{dt^k}f_n,$$

$$\frac{(-1-\alpha\Delta t)+e^{\alpha\Delta t}}{\alpha^2\Delta t}=\Delta t\sum_{k=0}^{\infty}\frac{1}{(k+2)!}(\alpha\Delta t)^k,\quad u(t_{n+1})=\sum_{k=0}^{\infty}\frac{1}{k!}\Delta t^k\frac{d^k}{dt^k}u_n, \tag{9}$$

and, from (8),

$$\frac{d^k}{dt^k}u=\alpha^k u+\sum_{j=0}^{k-1}\alpha^{k-1-j}\frac{d^j}{dt^j}f, \tag{10}$$

we derive the local truncation errors, T_n, of iETD2 and IIF2 for a general $f(u,t)$ to be

$$\begin{aligned}
&T_n^{iETD2}\\
&=\frac{1}{\Delta t}\left(u(t_{n+1})-e^{\alpha\Delta t}u_n-\frac{1+(-1+\alpha\Delta t)e^{\alpha\Delta t}}{\alpha^2\Delta t}f_n-\frac{(-1-\alpha\Delta t)+e^{\alpha\Delta t}}{\alpha^2\Delta t}f_{n+1}\right)\\
&=\sum_{j=2}^{\infty}\Delta t^j\frac{d^j}{dt^j}f_n\left[\sum_{k=0}^{\infty}\left(\frac{1}{(k+j+1)!}-\frac{1}{j!(k+2)!}\right)(\alpha\Delta t)^k\right]\\
&=\sum_{j=2}^{\infty}\Delta t^j\frac{d^j}{dt^j}f_n\left(Q_{j+1}(\alpha\Delta t)-\frac{1}{j!}Q_2(\alpha\Delta t)\right),
\end{aligned} \tag{11}$$

$$\begin{aligned}
T_n^{IIF2}&=\frac{1}{\Delta t}\left(u(t_{n+1})-e^{\alpha\Delta t}u_n-\frac{\Delta t}{2}\left(e^{\alpha\Delta t}f_n+f_{n+1}\right)\right)\\
&=\alpha^2\Delta t^2 f_n\sum_{k=0}^{\infty}\left(\frac{1}{(k+3)!}-\frac{1}{2(k+2)!}\right)(\alpha\Delta t)^k\\
&\qquad+\sum_{j=1}^{\infty}\Delta t^j\frac{d^j}{dt^j}f_n\left(\sum_{k=0}^{\infty}\frac{1}{(k+j+1)!}(\alpha\Delta t)^k-\frac{1}{2j!}\right)\\
&=\alpha^2\Delta t^2 f_n\left(Q_3(\alpha\Delta t)-\frac{1}{2}Q_2(\alpha\Delta t)\right)+\alpha\Delta t^2\frac{d}{dt}f_nQ_3(\alpha\Delta t)\\
&\qquad+\sum_{j=2}^{\infty}\Delta t^j\frac{d^j}{dt^j}f_n\left(Q_{j+1}(\alpha\Delta t)-\frac{1}{2j!}\right),
\end{aligned} \tag{12}$$

where we have made use of the function

$$Q_j(x)=\frac{e^x-\sum_{k=0}^{j-1}\frac{1}{k!}x^k}{x^j}=\sum_{k=0}^{\infty}\frac{1}{(k+j)!}x^k \tag{13}$$

from [2] to simplify the expressions.

The truncation error for hIFE is a more complicated expression, and its general form is omitted in favor of more specific cases below. We now investigate the properties of each of these for three different possible $f(u,t)$: (I) explicitly time-dependent reactions, $f(u,t)=g(t)$; (II) autonomous reactions, $f(u,t)=h(u)$; and (III) mixed reactions of the form $f(u,t)=h(u)+g(t)$. The first few terms of each of these expressions is included in explicit form up to $\mathcal{O}(\Delta t^4)$ for each of these cases in Table 1.

2.1.1. *Case I: Explicitly time-dependent reactions.* Here we consider explicitly time-dependent reactions of the form $f(u,t)=g(t)$. We assume $g\in C^\infty$ and g and its derivatives of any order are bounded and $\mathcal{O}(1)$. We only compare IIF2 and iETD2 here, since hIFE2 is equivalent to iETD2 in this case ($F_1(u,t)\equiv 0$ in (6)). Since

$f(u,t) = g(t)$ is dependent only on t, the time derivatives $\frac{d^j}{dt^j} f$ in (11) and (12) are just the single-variable derivatives $g^{(j)}(t)$, so the truncation error takes the same general form as those expressions in this case.

Although both methods have second-order truncation errors $\mathcal{O}(\Delta t^2)$ and thus should exhibit second-order behavior in the limit $\Delta t \to 0$, for a fixed nonzero Δt, we are only guaranteed to observe second-order behavior if the higher-order terms are much smaller in magnitude than the terms involving Δt^2. By comparing the Δt^2 and Δt^3 terms (given explicitly in Table 1), we see in both IIF2 and iETD2 that if $\Delta t > \mathcal{O}(1/|\alpha|)$, the Δt^3 terms have a larger magnitude than the Δt^2 terms. If we want to observe second-order behavior, then it must satisfy $\Delta t < \mathcal{O}(1/|\alpha|)$; the behavior of the error above this threshold is unpredictable. We demonstrate this claim by means of an example, taking $g(t) = t^2$. The numerical errors from applying IIF2 and iETD2 (and, thus, hIFE2) to (8) with $f(u,t) = t^2$ are plotted in Figure 1A as a function of Δt for various $-\alpha$. We see that for a fixed $-\alpha$, the thresholds at which both IIF2 and iETD2 "switch" to second order are similar to each other, and both are near $\Delta t \approx 1/|\alpha|$, consistent with our claim.

On the other hand, another interesting feature of the plot is that for a fixed Δt, the error in IIF2 *increases* as $-\alpha$ increases while that of iETD2/hIFE2 *decreases*. Indeed, to demonstrate why the error for each method behaves differently as $-\alpha$ increases for general $g(t)$, we explore the magnitudes of truncation errors when $\alpha \to 0$ and $\alpha \to -\infty$. We first note that for a fixed Δt, as $\alpha \to 0$, the truncation errors in (11) and (12) for both methods take the form of the Crank–Nicholson method, with truncation error

$$T_n = \sum_{j=2}^{\infty} \Delta t^j g_n^{(j)} \left(\frac{1}{(j+1)!} - \frac{1}{2j!} \right) = \mathcal{O}(\Delta t^2).$$

Then as $\alpha \to -\infty$, the truncation error for iETD2 in (11) tends to zero while that for IIF2 in (12) tends to $-\frac{1}{2} g_{n+1}$, which is $\mathcal{O}(1)$. When $-\alpha$ varies from 0 to ∞, the truncation errors of iETD2 are generally decreasing, changing from $\mathcal{O}(\Delta t^2)$ to 0. In contrast, the truncation errors of IIF2 are generally increasing, changing from $\mathcal{O}(\Delta t^2)$ to $\mathcal{O}(1)$. Thus, we expect, for a fixed Δt, the global error for iETD2 to decrease for larger $-\alpha$ while the global error in IIF2 should increase.

Remark. We have shown in the scalar case that the threshold at which IIF2 and iETD2 "switch" to second-order temporal error for time-dependent reactions is similar in both methods, and this seems to invalidate our entire motivation for developing hIFE2 in the first place. It is, however, the preceding observation concerning the behavior of the errors for large $-\alpha$ that will prove crucial to why iETD2 remains second order for larger Δt than IIF2 when applied to a semi-discrete *system* with time-dependent reactions. We examine that form in Section 2.2.

Remark. In Figure 1A, for Δt above the threshold, the temporal error of IIF2 has *first*-order behavior instead of second. For some remarks on why that might be the case, see Appendix A.

2.1.2. *Case II: Autonomous reactions.* We now consider reactions of the form $f(u,t) = h(u)$ satisfying $h(0) = 0$, which do not have explicitly time-dependent terms. In this case, hIFE2 is equivalent to IIF2 since $F_2(t) \equiv 0$ in (6), so again we only compare IIF2 and iETD2 here. Further, we only consider $h(u) = ru$ ($r \in \mathbb{R}$) since this form is easily extendable to the semi-discrete case.

Here, the total time derivatives in (11) and (12) take the form

$$\frac{d^j}{dt^j} f_n = r(\alpha + r)^j u_n = r(\alpha + r)^j e^{(\alpha+r)t_n} u_0, \tag{14}$$

where we have made use of the exact solution $u_n = u_0 e^{(\alpha+r)t_n}$. The full truncation error up to $\mathcal{O}(\Delta t^4)$ is given for both methods in Table 1. Similar to Case I, we note that the Δt^3 coefficients are $\mathcal{O}(\alpha)$ larger than the coefficients of Δt^2 in both iETD2 and IIF2/hIFE2 so that it must also satisfy $\Delta t < \mathcal{O}(1/|\alpha|)$ to observe second-order behavior of the error.

The most important difference between this case and Case I, though, is that the error in both IIF2 and iETD2 now includes, through the derivatives, a factor of $u_n = u_0 e^{(\alpha+r)t_n}$ in every term, and thus as $-\alpha$ increases, u_n exponentially suppresses the truncation error to zero in *both* methods, not just in iETD2.

Again, we demonstrate the validity of our claims with an example. The error in applying iETD2 and IIF2/hIFE2 to (8) with $f(u,t) = -u$ (so $r = -1$) is shown as a function of Δt for various $-\alpha$ in Figure 1B. We note that, consistent with the above analysis, the error decreases dramatically with $-\alpha$ for both iETD2 and IIF2/hIFE, and both demonstrate second-order accuracy for $\Delta t < \mathcal{O}(1/|\alpha|)$ while the error (particularly in iETD2) remains unpredictable above this threshold.

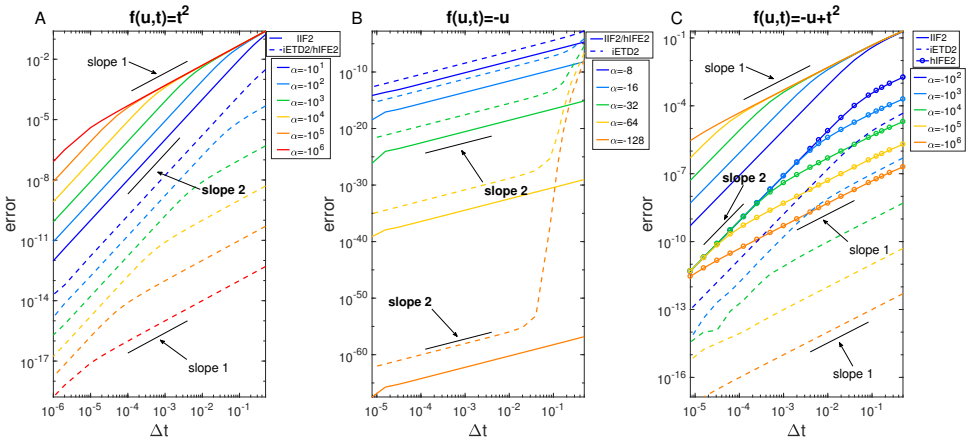

FIGURE 1. Plots of the numerical error at $T = 1$ after applying IIF2, iETD2, and hIFE2 to the scalar equation in (8) with $u(0) = 1$ for various Δt. Plots are shown for (A) $f(u,t) = t^2$ with $\alpha = -10^1, -10^2, -10^3, -10^4, -10^5$, and -10^6; (B) $f(u,t) = -u$ with $\alpha = -8, -16, -32, -64$, and -128; and (C) $f(u,t) = -u + t^2$ with $\alpha = -10^2, -10^3, -10^4, -10^5$, and -10^6. The curves for iETD2 and hIFE2 are identical in (A), and those for IIF2 and hIFE2 are identical in (B). We see that for the time-dependent reactions (A,C), the error in IIF2 increases as $-\alpha$ increases while the error in iETD2 and hIFE2 decreases.

2.1.3. *Case III: Mixed reactions.* Finally, we consider mixed reactions of the form $f(u,t) = ru + g(t)$, where, as in Case I, we assume $g \in C^\infty$ and g and all its derivatives are bounded and $\mathcal{O}(1)$. Again, we only consider ru for the u-dependent

term so that the extension of the analysis to system case in the next section is straightforward. In this case, the derivatives in (11) and (12) are given by

$$\frac{d^j}{dt^j} f_n = r(\alpha + r)^j u_n + g_n^{(j)} + r \sum_{l=0}^{j-1} (\alpha + r)^{j-1-l} g_n^{(l)}. \tag{15}$$

The expressions for the truncation errors of IIF2, iETD2, and hIFE2 up to $\mathcal{O}(\Delta t^4)$ are given in Table 1. This is the first (and only) case we examine where the hIFE2 method differs from both IIF2 and iETD2.

As in Cases I and II, we observe that the Δt^3 terms are $\mathcal{O}(\alpha)$ times larger than the Δt^2 terms, so the threshold $\Delta t < \mathcal{O}(1/|\alpha|)$ also applies here. The main feature we point out for the errors in this case is that, owing to the form of the derivatives in (15), after substituting this expression into the truncation errors in (11) and (12) and multiplying everything out, the truncation error in this case will contain terms that involve just the derivatives $g_n^{(j)}$, exactly replicating the error in Case I. Then, regardless of how the other terms behave, there will be a part of the error that behaves in the same manner as the time-dependent Case I. That is, as $-\alpha$ grows, some part of the error will increase under IIF2 whereas that same part of the error will decrease under iETD2. The error should also decrease under hIFE2 since this method treats those time-dependent terms with iETD2. Thus, in this case we expect to observe similar behavior as in Case I: increasing error in IIF2 and decreasing error in iETD2 and hIFE2 as $-\alpha$ increases.

We demonstrate this claim with yet another example, this time taking $f(u,t) = -u + t^2$, the results of which are shown in Figure 1C as a function of Δt. We see that indeed the error for all three methods behaves similarly to Case I, increasing as $-\alpha$ increases under IIF2 while decreasing under iETD2 and hIFE2. The plot also verifies our prediction that the threshold at which second-order behavior is seen is around $\Delta t \approx 1/|\alpha|$.

2.2. **Accuracy in semi-discrete form.** Having examined each of the three cases for the scalar equation (8), we now turn our attention to the differences between the scalar form and the semi-discrete form,

$$U_t = AU + F(U,t). \tag{16}$$

A motivating one-dimensional problem that can be written in the form (16) is

$$\begin{cases} u_t = du_{xx} + f(u,x,t), \quad x \in [a,b],\ t \in [0,T], \\ u_x|_{x=a} = u|_{x=b} = 0, \\ u(x,0) = v(x), \end{cases} \tag{17}$$

where $u = u(x,t)$, $d > 0$, and we require $v(x)$ to satisfy the given boundary conditions. Note that the boundary conditions here are mixed homogeneous; we discuss different types of boundary conditions in Section 3.

We can put the partial differential equation (PDE) (17) into the form (16) by using finite difference discretization of the operator $d\frac{\partial^2}{\partial x^2}$. Let $\Delta x = (b-a)/N$ be the mesh size and N be the number of grid points in the spatial discretization. We can write A as the diagonalizable $N \times N$ matrix,

Method	Reaction f	Truncation error
IIF2	$g(t)$	$-\frac{1}{12}\Delta t^2(\alpha^2 g_n - 2\alpha g_n' + g_n'')$ $-\frac{1}{24}\Delta t^3(\alpha^3 g_n - \alpha^2 g_n' - \alpha g_n'' + g_n''') + \mathcal{O}(\Delta t^4)$
	ru	$-\frac{1}{12}\Delta t^2 r^3 u_n$ $-\frac{1}{24}\Delta t^3(2\alpha r^3 + r^4)u_n + \mathcal{O}(\Delta t^4)$
	$ru + g(t)$	$-\frac{1}{12}\Delta t^2\big[\alpha^2 g_n + \alpha(-rg_n - 2g_n') + (r^3 u_n + r^2 g_n + rg_n' + g_n'')\big]$ $-\frac{1}{24}\Delta t^3\big[\alpha^3 g_n + \alpha^2(-rg_n - g_n') + \alpha(2r^3 u_n + r^2 g_n - g_n'')$ $+(r^4 u_n + r^3 g_n + r^2 g_n' + rg_n'' + g_n''')\big] + \mathcal{O}(\Delta t^4)$
iETD2	$g(t)$	$-\frac{1}{12}\Delta t^2 g_n''$ $-\frac{1}{24}\Delta t^3(\alpha g_n'' + g_n''') + \mathcal{O}(\Delta t^4)$
	ru	$-\frac{1}{12}\Delta t^2(\alpha^2 r + 2\alpha r^2 + r^3)u_n$ $-\frac{1}{24}\Delta t^3(2\alpha^3 r + 5\alpha^2 r^2 + 4\alpha r^3 + r^4)u_n + \mathcal{O}(\Delta t^4)$
	$ru + g(t)$	$-\frac{1}{12}\Delta t^2\big[\alpha^2 ru_n + \alpha(2r^2 u_n + rg_n) + (r^3 u_n + r^2 g_n + rg_n' + g_n'')\big]$ $-\frac{1}{24}\Delta t^3\big[2\alpha^3 ru_n - \alpha^2(5r^2 u_n + 2rg_n) + \alpha(4r^3 u_n + 3r^2 g_n + 2rg_n' + g_n'')$ $+(r^4 u_n + r^3 g_n + r^2 g_n' + rg_n'' + g_n''')\big] + \mathcal{O}(\Delta t^4)$
hIFE2	$g(t)$	equivalent to iETD2
	ru	equivalent to IIF2
	$ru + g(t)$	$-\frac{1}{12}\Delta t^2\big[-\alpha rg_n + (r^3 u_n + r^2 g_n + rg_n' + g_n'')\big]$ $-\frac{1}{24}\Delta t^3\big[-\alpha^2 rg_n + \alpha(2r^3 u_n + r^2 g_n + g_n'')$ $+(r^4 u_n + r^3 g_n + r^2 g_n' + rg_n'' + g_n''')\big] + \mathcal{O}(\Delta t^4)$

TABLE 1. The truncation errors of IIF2, iETD2, and hIFE2 when applied to (8) with different reactions.

$$A = \frac{d}{\Delta x^2}\begin{bmatrix} -2 & 2 & & & & \\ 1 & -2 & 1 & & & \\ & 1 & -2 & 1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & 1 & -2 & 1 \\ & & & & 1 & -2 \end{bmatrix}_{N\times N}, \tag{18}$$

and we form the vectors

$$\begin{aligned} U(t) &= \big(u^0(t), u^1(t), \ldots, u^{N-1}(t)\big)^T \\ F(U,t) &= \big(f(u^0(t), x^0, t), f(u^1(t), x^1, t), \ldots, f(u^{N-1}(t), x^{N-1}, t)\big)^T \end{aligned} \tag{19}$$

where $x^i = a + i\Delta x$, $i = 0, \ldots, N-1$, are the grid points in the discretization (we identify $x_N = b$), and $u^i(t) \triangleq u(x^i, t)$. This spatial discretization completely removes x-dependence from the equation and introduces an error from the exact solution of $\mathcal{O}(\Delta x^2)$; we assume this level of spatial error in the remaining calculations and only discuss temporal error for the remainder of the analysis.

The matrix A has N distinct eigenvalues, $\lambda_j \triangleq d\sigma_j/\Delta x^2$, $j = 1, \ldots, N$, where σ_j are listed in descending order:

$$\sigma_j = -2 + 2\cos\frac{(2j-1)\pi}{2N}, \quad j = 1, \ldots, N. \tag{20}$$

The eigenvalues are all negative, and in the limit $N \to \infty$,

$$\lambda_1 \to -d\left(\frac{\pi/2}{b-a}\right)^2 = \mathcal{O}(d), \quad \lambda_N \approx -\frac{4d}{\Delta x^2} \to -\infty. \tag{21}$$

For convenience, we henceforth set $a = 0$, $b = \pi/2$ so that $\lambda_1 \to -d$. The least negative possible eigenvalue occurs at $N = j = 1$, where $\lambda_1 = -\frac{8d}{\pi^2}$. For small Δx (i.e. large N), the range of eigenvalues is large, as listed in Table 2. Though we have only discussed mixed boundary conditions here, the eigenvalues of the matrices corresponding to Neumann and Dirichlet conditions have similar properties, which are all negative and differ in a large range [41].

j \ N	32	64	128	256	512	1024
1	−1.00	−1.00	−1.00	−1.00	−1.00	−1.00
5	−7.97e+1	−8.07e+1	−8.09e+1	−8.09e+1	−8.10e+1	−8.10e+1
$N/2$	−7.89e+2	−3.23e+03	−1.31e+04	−5.28e+04	−2.12e+05	−8.49e+05
N	−1.66e+03	−6.64e+03	−2.66e+04	−1.06e+05	−4.25e+05	−1.70e+06

TABLE 2. Eigenvalues of A, λ_j, under different spatial resolutions, where $d = 1$, $a = 0$, $b = \pi/2$, $j = 1,\ 5,\ N/2,\ N$.

2.2.1. *Relating the error in semi-discrete form to that of the scalar form.* In each of the three cases in Section 2.1, we have considered reactions that were easily extendable to the semi-discrete form. Since the matrix A in (18) is diagonalizable, we thus set $A = H\Lambda H^{-1}$, where Λ is the diagonal matrix consisting of the eigenvalues λ_j, $j = 1, \ldots, N$. Then for an explicitly time-dependent reaction $F(U,t) = G(t)$, we can multiply (16) on the left by H^{-1} to obtain

$$\frac{d}{dt}\left(H^{-1}U\right) = \Lambda\left(H^{-1}U\right) + H^{-1}G(t), \tag{22}$$

where $H^{-1}U$ and $H^{-1}G(t)$ are N-vectors. Then since Λ is diagonal, the jth component of this system takes the form

$$\frac{d}{dt}\left(H^{-1}U\right)_j = \lambda_j\left(H^{-1}U\right)_j + \left(H^{-1}G\right)_j(t), \tag{23}$$

which matches the scalar form (8) with the new variable $u = (H^{-1}U)_j$, $\alpha = \lambda_j$, and $f(u,t) = \left(H^{-1}G\right)_j(t)$, independent of u. We can thus solve this system by solving each of the components individually.

Similarly, if $F(U,t) = rU$, we can also diagonalize the system, writing

$$\frac{d}{dt}\left(H^{-1}U\right) = \Lambda\left(H^{-1}U\right) + r\left(H^{-1}U\right), \tag{24}$$

so that we obtain a system of scalar equations with $\alpha = \lambda_j$, $f(u,t) = r(H^{-1}U)_j$ (cf. $u_t = \alpha u + ru$). For the mixed reaction, $F(U,t) = rU + G(t)$, a similar form can be obtained. For general nonlinear reactions, the conclusions in scalar form cannot be directly extended to semi-discrete form since the system may not be diagonalizable. We do not analyze the nonlinear reactions in this paper, only numerical tests are carried out in Section 5.

From the above, we see that solving an equation in the semi-discrete form is equivalent to solving N separate scalar equations, each with different α. We showed in Sections 2.1.1 and 2.1.3 that for scalar equations with time-dependent or mixed reactions, as $-\alpha$ increases, the error in IIF2 increases while the error in iETD2 and hIFE2 decreases. Thus, if we apply each method to the entire system (22) and measure error with the maximum norm, the component we expect to have the largest error under IIF2 is the one with the largest (i.e. most negative) eigenvalue, λ_N, whereas we expect the component with the largest error under iETD2 and hIFE2 to be the one with the smallest eigenvalue, λ_1. Since each scalar equation

requires $\Delta t < \mathcal{O}(1/|\alpha|) = \mathcal{O}(1/|\lambda_j|)$ to observe second-order temporal error (Section 2.1.1), we expect iETD2 and hIFE2 to display second-order behavior for any $\Delta t < \mathcal{O}(1/|\lambda_1|) = \mathcal{O}(1/d)$ while IIF2 requires $\Delta t < \mathcal{O}(1/|\lambda_N|) = \mathcal{O}(\Delta x^2/d)$. For large N, the magnitudes of λ_1 and λ_N differ in large range as shown in (21), then IIF2 requires a much smaller Δt to exhibit second-order temporal error than iETD2 and hIFE2. Meanwhile, for autonomous reactions, we showed in Section 2.1.2 that the error for the scalar equation in all three methods decreases with larger $-\alpha$. Thus, in this case, the error in all three methods is dictated by the smallest eigenvalue, λ_1, so that all three methods exhibit second-order temporal error for large $\Delta t < \mathcal{O}(1/d)$.

We demonstrate the above claims in Figure 2 by comparing the behavior of IIF2, iETD2, and hIFE2 on the three reactions $F(U,t) = t^2$, $F(U,t) = -U$, and $F(U,t) = -U + t^2$ with various N as a function of Δt. We see that indeed for IIF2, as N increases, progressively smaller Δt are required to observe second-order temporal error. In contrast, iETD2 and hIFE2 always maintain second-order temporal accuracy for different N, consistent with our analysis.

Remark. When the explicitly time-dependent terms appear in the reactions, the difference in performance of the IIF2 and iETD2 methods makes intuitive sense. In IIF2, the product of the exponential integration factor and the reaction is interpolated together, whereas in iETD2, only the reaction is interpolated, not the exponential. For large $-\alpha$, IIF2 has to interpolate an exponentially increasing function, $e^{-\alpha t}g(t)$, with a polynomial, thus likely introducing a lot of error. In contrast, iETD2 interpolates only the reaction, $g(t)$, which is independent of $-\alpha$ and likely varies less over time. Therefore, the approximation by iETD2 is likely to be more accurate than IIF2 in this case.

Remark. In previous work, the absolute stability analysis was carried out on ETD [2] and IIF [34] methods. Both iETD2 and IIF2 are unconditionally linearly stable (A-stable), while eETD2 is conditionally linearly stable. The absolute stability analysis only considers equations with autonomous reactions and homogeneous boundary conditions. The hIFE2 has the same absolute stability with IIF2. The fEIF2 method [22] has the same absolute stability with eETD2. Therefore, the hIFE2 method is A-stable, and fEIF2 is conditionally linearly stable. For solving stiff systems, a large time step size is allowed if the method is A-stable. We show the advantage of hIFE2 over fEIF2 in stability in Section 5.2.

3. The hybrid method hIFE for systems with nonhomogeneous boundary conditions.

We showed that hIFE2 is more accurate than IIF2 in Section 2 and has lower computational cost than iETD2 in Appendix B. In this section, we discuss the extension of hIFE2 to equations with nonhomogeneous boundary conditions through a transformation.

3.1. Direct treatment of nonhomogeneous boundary conditions.

In all of the analysis in Section 2, we assumed that the explicitly time-dependent terms were $\mathcal{O}(1)$. When considering a one-dimensional reaction–diffusion equation of the form

$$u_t = du_{xx} + f(u,x,t). \tag{25}$$

with nonhomogeneous boundary conditions, however, the system can be written in the modified semi-discrete form (cf. (16)),

$$U_t = AU + B(t) + F(U,t), \tag{26}$$

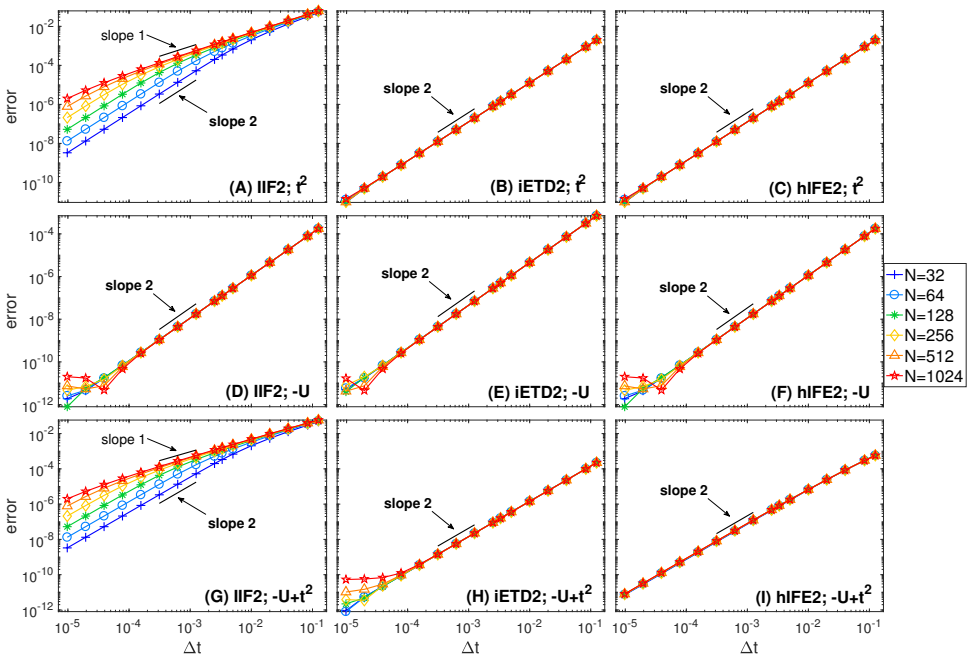

FIGURE 2. The temporal errors at $T = 1$ in the maximum norm when solving the semi-discrete form (16) of (27) for different reactions with the IIF, iETD2, and hIFE2 methods. In all simulations, the reaction coefficient $d = 1$. (A) IIF2 for $F(U,t) = t^2$; (B) iETD2 for $F(U,t) = t^2$; (C) hIFE2 for $F(U,t) = t^2$; (D) IIF2 for $F(U,t) = -U$; (E) iETD2 for $F(U,t) = -U$; (F) hIFE2 for $F(U,t) = -U$; (G) IIF2 for $F(U,t) = -U + t^2$; (H) iETD2 for $F(U,t) = -U + t^2$; (I) hIFE2 for $F(U,t) = -U + t^2$. Different colors represent the number of points, N, in the spatial discretization, where $N = 32,\ 64,\ 128,\ 256,\ 512$, and 1024. Subfigures in same row share the same y-axis while subfigures in same column share the same x-axis. Panels (B) and (C) are identical because hIFE2 treats time-dependent terms with iETD2, and panels (D) and (F) are identical since hIFE2 treats autonomous terms with IIF2.

where $B(t)$ is a time-dependent term of $\mathcal{O}(1/\Delta x)$ or $\mathcal{O}(1/\Delta x^2)$ for Neumann- or Dirichlet-type conditions, respectively [41]. Then hIFE2 may not remain second-order under large Δt.

For example, we consider a specific one-dimensional reaction–diffusion equation,

$$\begin{cases} u_t = u_{xx} - u, \quad x \in \left[0, \dfrac{\pi}{2}\right], \\ u(x,0) = \sin(x + \dfrac{\pi}{6}), \end{cases} \tag{27}$$

a solution of which is

$$u(x,t) = e^{-2t} \sin(x + \frac{\pi}{6}). \tag{28}$$

We now consider three different variations of this problem with different non-homogeneous boundary conditions dictated by the exact solution (28): (a) the

Neumann boundary conditions; (b) the Dirichlet boundary conditions; (c) mixed boundary conditions, Neumann at $x = 0$ and Dirichlet at $x = \frac{\pi}{2}$. We list the boundary conditions and the corresponding A and $B(t)$ in the semi-discrete form (26) in Table 3. Note that Neumann conditions cause $B(t)$ to be $\mathcal{O}(1/\Delta x)$, and Dirichlet and mixed conditions cause $B(t)$ to be $\mathcal{O}(1/\Delta x^2)$.

	Neumann	Dirichlet	Mixed
BCs	$u_x\|_{x=0} = e^{-2t}\cos\frac{\pi}{6}$ $u_x\|_{x=\frac{\pi}{2}} = e^{-2t}\cos\frac{2\pi}{3}$	$u\|_{x=0} = e^{-2t}\sin\frac{\pi}{6}$, $u\|_{x=\frac{\pi}{2}} = e^{-2t}\sin\frac{2\pi}{3}$	$u_x\|_{x=0} = e^{-2t}\cos\frac{\pi}{6}$ $u\|_{x=\frac{\pi}{2}} = e^{-2t}\sin\frac{2\pi}{3}$
$B(t)$	$e^{-2t}\begin{bmatrix} -\frac{2\cos\frac{\pi}{6}}{\Delta x} \\ 0 \\ \vdots \\ 0 \\ \frac{2\cos\frac{2\pi}{3}}{\Delta x} \end{bmatrix}_{N+1}$	$e^{-2t}\begin{bmatrix} \frac{\sin\frac{\pi}{6}}{\Delta x^2} \\ 0 \\ \vdots \\ 0 \\ \frac{\sin\frac{2\pi}{3}}{\Delta x^2} \end{bmatrix}_{N-1}$	$e^{-2t}\begin{bmatrix} -\frac{2\cos\frac{\pi}{6}}{\Delta x} \\ 0 \\ \vdots \\ 0 \\ \frac{\sin\frac{2\pi}{3}}{\Delta x^2} \end{bmatrix}_{N}$
A	$\frac{1}{\Delta x^2}\begin{bmatrix} -2 & 2 & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ & & & 2 & -2 \end{bmatrix}_{(N+1)^2}$	$\frac{1}{\Delta x^2}\begin{bmatrix} -2 & 1 & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ & & & 1 & -2 \end{bmatrix}_{(N-1)^2}$	$\frac{1}{\Delta x^2}\begin{bmatrix} -2 & 2 & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ & & & 1 & -2 \end{bmatrix}_{N^2}$

TABLE 3. Different boundary conditions in (27), and their corresponding A and $B(t)$ in the semi-discrete form (26).

We directly apply hIFE2 to (27) with these three kinds of boundary conditions and plot the error at $T = 1$ in Figure 3A–C. In hIFE2, the explicitly time-dependent term $B(t)$ is treated by iETD2, and the autonomous term $-U$ is treated by IIF2. In each of the cases, we refine Δt until the error is dominated by spatial error introduced in the discretization. We see that hIFE2 exhibits second-order accuracy only for the $\mathcal{O}(1/\Delta x)$ Neumann boundary conditions. The hIFE2 method does not retain second-order accuracy for large Δt with the $\mathcal{O}(1/\Delta x^2)$ Dirichlet or mixed boundary conditions, presumably because $B(t)$ is too large.

3.2. **A transformation for nonhomogeneous boundary conditions.** To observe second-order behavior for hIFE2 under large time step when dealing with nonhomogeneous boundary conditions, we construct an auxiliary function u_B that satisfies the same boundary conditions as u and form a new system in terms of the variable $\tilde{u} = u - u_B$ that satisfies *homogeneous* boundary conditions [47]. We then solve the new system using hIFE2 for $\tilde{u}$ and add the auxiliary function u_B back into the obtained solution to recover u. Since $\tilde{u}$ satisfies homogeneous boundary conditions, hIFE2 can obtain a solution with second-order accuracy even when u itself satisfies Dirichlet conditions.

3.2.1. *One-dimensional system.* Let $u(x,t)$ satisfy (25) with the following general nonhomogeneous boundary conditions on a one-dimensional domain $[x_1, x_2]$:

$$\begin{cases} \mathcal{B}_1 u(x_1,t) = (\alpha_1 u + \beta_1 u_x)|_{x=x_1} = f_1(t), \\ \mathcal{B}_2 u(x_2,t) = (\alpha_2 u + \beta_2 u_x)|_{x=x_2} = f_2(t). \end{cases} \tag{29}$$

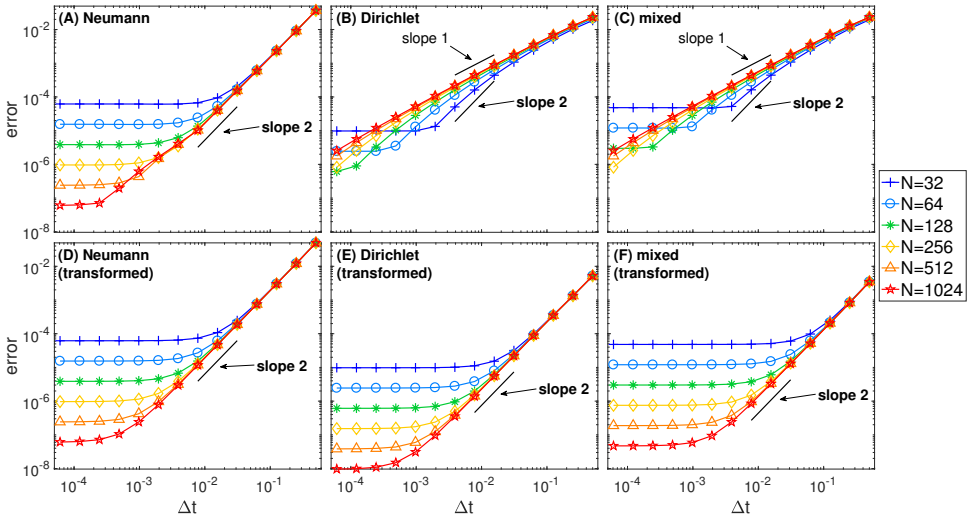

FIGURE 3. Plots of the numerical error at $T = 1$ in maximum norm after applying hIFE2 to (27) with Neumann, Dirichlet, and mixed boundary conditions for various Δt and fixed N. The hIFE2 is applied to both original and transformed (Section 3.2) equations. Plots are shown for hIFE2 on: (A) the original equation with Neumann boundary; (B) the original equation with Dirichlet boundary; (C) the original equation with mixed boundary; (D) the transformed equation with Neumann boundary; (E) the transformed equation with Dirichlet boundary; (F) the transformed equation with mixed boundary. Different colors represent different spatial mesh sizes N, where $N = 32,\ 64,\ 128,\ 256,\ 512$, and 1024.

To construct the auxiliary function u_B, we define two basis functions,

$$\begin{aligned} C_1(x) &= \frac{(x-x_2)^2}{\alpha_1(x_1-x_2)^2 + 2\beta_1(x_1-x_2)}, \\ C_2(x) &= \frac{(x-x_1)^2}{\alpha_2(x_2-x_1)^2 + 2\beta_2(x_2-x_1)}, \end{aligned} \tag{30}$$

so that

$$\begin{aligned} \mathcal{B}_1 C_1(x_1) = 1,\ \mathcal{B}_1 C_2(x_1) = 0, \\ \mathcal{B}_2 C_1(x_2) = 0,\ \mathcal{B}_2 C_2(x_2) = 1. \end{aligned}$$

Then we construct u_B as a linear combination of the basis functions

$$u_B(x,t) = C_1(x) f_1(t) + C_2(x) f_2(t), \tag{31}$$

so that u_B satisfies the same nonhomogeneous boundary conditions (29) as u. Hence, $\tilde{u} = u - u_B$ will satisfy homogeneous boundary conditions in the modified equation

$$\tilde{u}_t = d\tilde{u}_{xx} + f(\tilde{u} + u_B, x, t) - (u_B)_t + d(u_B)_{xx}. \tag{32}$$

This modified equation can thus be solved by hIFE2 and the original solution u of (25) can be found by adding u_B back into the numerically obtained solution $\tilde{u}$ of (32). Note that the function u_B is known, so u_B and its derivatives can be

computed analytically. The modified equation includes nonzero explicitly time-dependent terms with magnitude $\mathcal{O}(1)$. Indeed, hIFE2 can exhibit second order in time for large Δt while IIF2 cannot exhibit that. The effect of the transformation can be thought of as "spreading out" the two large nonzero components in the untransformed $B(t)$ over the entire domain to dampen their magnitude.

We include plots of the numerical results of applying hIFE2 to the transformed versions of the model problem (27) from the previous section in Figure 3D–F, noting that indeed hIFE2 remains second order for the transformed equations when it failed to do so for the untransformed ones.

3.2.2. *Higher-dimensional systems.* Suppose a solution $u(x, y, t)$ is desired on the two-dimensional domain $[x_1, x_2] \times [y_1, y_2]$, with nonhomogeneous boundary conditions,

$$\begin{cases} \mathcal{B}_{11}u(x_1, y, t) = (\alpha_{11}u + \beta_{11}u_x)|_{x=x_1} = f_1(y, t), \\ \mathcal{B}_{12}u(x_2, y, t) = (\alpha_{12}u + \beta_{12}u_x)|_{x=x_2} = f_2(y, t), \\ \mathcal{B}_{21}u(x, y_1, t) = (\alpha_{21}u + \beta_{21}u_y)|_{y=y_1} = g_1(x, t), \\ \mathcal{B}_{22}u(x, y_2, t) = (\alpha_{22}u + \beta_{22}u_y)|_{y=y_2} = g_2(x, t). \end{cases} \tag{33}$$

Using a similar technique as in the one-dimensional case (30), we define the basis functions $C_{11}(x)$, $C_{12}(x)$, $C_{21}(y)$, $C_{22}(y)$ satisfying

$$\begin{aligned} \mathcal{B}_{11}C_{11}(x_1) = 1,\ \mathcal{B}_{11}C_{12}(x_1) = 0, \\ \mathcal{B}_{12}C_{11}(x_2) = 0,\ \mathcal{B}_{12}C_{12}(x_2) = 1, \\ \mathcal{B}_{21}C_{21}(y_1) = 1,\ \mathcal{B}_{21}C_{22}(y_1) = 0, \\ \mathcal{B}_{22}C_{21}(y_2) = 0,\ \mathcal{B}_{22}C_{22}(y_2) = 1. \end{aligned}$$

To construct the auxiliary function u_B, we first transform the boundary conditions at $x = x_1, x_2$, letting

$$u_{B_1}(x, y, t) = C_{11}(x)f_1(y, t) + C_{12}(x)f_2(y, t). \tag{34}$$

Then $u - u_{B_1}$ satisfies the boundary conditions

$$\begin{aligned} &\mathcal{B}_{11}(u - u_{B_1})(x_1, y, t) = 0, \\ &\mathcal{B}_{12}(u - u_{B_1})(x_2, y, t) = 0, \\ &\mathcal{B}_{21}(u - u_{B_1})(x, y_1, t) = g_1(x, t) - [C_{11}(x)\mathcal{B}_{21}f_1(y_1, t) + C_{12}(x)\mathcal{B}_{21}f_2(y_1, t)] \triangleq \tilde{g}_1(x, t), \\ &\mathcal{B}_{22}(u - u_{B_1})(x, y_2, t) = g_2(x, t) - [C_{11}(x)\mathcal{B}_{22}f_1(y_2, t) + C_{12}(x)\mathcal{B}_{22}f_2(y_2, t)] \triangleq \tilde{g}_2(x, t), \end{aligned}$$

which is homogeneous with respect to x. Then we transform the boundary conditions at $y = y_1, y_2$, letting

$$u_{B_2}(x, y, t) = C_{21}(y)\tilde{g}_1(x, t) + C_{22}(y)\tilde{g}_2(x, t).$$

Then $u_B = u_{B_1} + u_{B_2}$ satisfies the same boundary conditions (33) as u so that $\tilde{u} = u - u_{B_1} - u_{B_2} = u - u_B$ satisfies homogeneous boundary conditions. Altogether, then, we have

$$\begin{aligned} u_B = {} & C_{11}(x)f_1(y, t) + C_{12}(x)f_2(y, t) + C_{21}(y)g_1(x, t) + C_{22}(y)g_2(x, t) \\ & - C_{11}(x)C_{21}(y)\mathcal{B}_{21}f_1(y_1, t) - C_{12}(x)C_{21}(y)\mathcal{B}_{21}f_2(y_1, t) \\ & - C_{11}(x)C_{22}(y)\mathcal{B}_{22}f_1(y_2, t) - C_{12}(x)C_{22}(y)\mathcal{B}_{22}f_2(y_2, t). \end{aligned} \tag{35}$$

Remark. Such an approach of constructing u_B can be extended to higher dimensions in a straightforward manner.

4. **The hIFE method in higher dimensions.** We now introduce a higher-dimensional version of the hIFE method that utilizes the compact versions of iETD [22] and IIF [33] to reduce the size of the system compared with a naïve discretization of the Laplacian operator. We only introduce the three-dimensional version since the procedure generalizes trivially to other dimensions.

The main drawback of higher-dimensional IIF and iETD methods is the large size of the system. For example, in three dimensions, if N_x, N_y, and N_z correspond to the number of grid points in the x, y, and z, directions, respectively, the matrix A in the semi-discrete form (2) has dimension $N_xN_yN_Z \times N_xN_yN_z$ (or $N^3 \times N^3$ if $N_x = N_y = N_z = N$). Since the number of computations per iteration for IIF is proportional to the square of the size of the system, each iteration requires $\mathcal{O}(N_x^2N_y^2N_z^2)$ operations (or $\mathcal{O}(N^6)$ if all are equal). The number of operations for iETD is proportional to the *cube* of the size of the system—$\mathcal{O}(N^9)$ operations per iteration for equal spacing in all directions.

In contrast, the compact representation reduces the storage requirement on A to $\mathcal{O}(N_x^2 + N_y^2 + N_z^2) = \mathcal{O}(3N^2)$ and the number of operations in each iteration to $\mathcal{O}(N_x^2N_yN_z + N_xN_y^2N_z + N_xN_yN_z^2) = \mathcal{O}(3N^4)$. When applied to hIFE, this approach leads to a significant improvement in its efficiency for higher-dimensional systems.

To illustrate the compact representation approach, we consider (1) in three dimensions. As in the one-dimensional hIFE, we split the reaction term $f(u,x,y,z,t) = f_1(u,x,y,z,t) + f_2(x,y,z,t)$, where $f_1(u,x,y,z,t) \triangleq f(u,x,y,z,t) - f(0,x,y,z,t)$ and $f_2(x,y,z,t) \triangleq f(0,x,y,z,t)$. Let N_x, N_y, and N_z be the number of grid points in each dimension, and h_x, h_y, and h_z be the corresponding mesh sizes. The spatial discretization is $\{(x_i,y_j,z_k) : 1 \le i \le N_x, 1 \le j \le N_y, 1 \le k \le N_z\}$. Denote the discretized u by $U_{N_x\times N_y\times N_z} = (U_{i,j,k})_{N_x\times N_y\times N_z}$. Here $F_1(U,t)$ and $F_2(t)$ are the discretized forms of f_1 and f_2, respectively. We define the linear operators ⓧ, ⓨ, and ⓩ as

$$\begin{aligned}(A\circledX U)_{i,j,k} &= \sum_{l=1}^{N_x} A_{i,l}U_{l,j,k}\\ (B\circledY U)_{i,j,k} &= \sum_{l=1}^{N_y} B_{j,l}U_{i,l,k}\\ (C\circledZ U)_{i,j,k} &= \sum_{l=1}^{N_z} C_{k,l}U_{i,j,l},\end{aligned} \tag{36}$$

and define the matrices $\mathcal{L}_x, \mathcal{L}_y, \mathcal{L}_z$ to approximate the one-dimensional diffusion operators $D\frac{\partial^2}{\partial x^2}, D\frac{\partial^2}{\partial y^2}, D\frac{\partial^2}{\partial z^2}$ by the same second-order central difference in the one-dimensional method. The compact representation for the diffusion approximation is

$$U_t = \mathcal{L}_x\circledX U + \mathcal{L}_y\circledY U + \mathcal{L}_z\circledZ U + F_1(U,t) + F_2(t). \tag{37}$$

The matrices $\mathcal{L}_x$, $\mathcal{L}_y$, and $\mathcal{L}_z$ are diagonalizable, i.e.,

$$\mathcal{L}_x = P_x\Lambda_xP_x^{-1}, \quad \mathcal{L}_y = P_y\Lambda_yP_y^{-1}, \quad \mathcal{L}_z = P_z\Lambda_zP_z^{-1},$$

where Λ_x, Λ_y, and Λ_z are diagonal matrices,

$$\begin{aligned}\Lambda_x &= \operatorname{diag}[\alpha_1, \alpha_2, \ldots, \alpha_{N_x}],\\ \Lambda_y &= \operatorname{diag}[\beta_1, \beta_2, \ldots, \beta_{N_y}],\\ \Lambda_z &= \operatorname{diag}[\gamma_1, \gamma_2, \ldots, \gamma_{N_z}].\end{aligned}$$

We define $H = (h_{i,j,k})_{N_x \times N_y \times N_z}$ such that

$$h_{i,j,k} = \alpha_i + \beta_j + \gamma_k,$$

the operator (e^*) as taking exponential of an array element by element,

$$(e^*)^H = (e^{h_{i,j,k}})_{N_x \times N_y \times N_z},$$

and the operator $\odot$ as componentwise matrix multiplication,

$$A \odot B = (a_{i,j,k} b_{i,j,k})_{N_x \times N_y \times N_z}.$$

Then we define the operator $\mathcal{L}(t)$ applied to U as

$$\begin{aligned}\mathcal{L}(t)U &= e^{-\mathcal{L}_z t} \textcircled{z} e^{-\mathcal{L}_y t} \textcircled{y} e^{-\mathcal{L}_x t} \textcircled{x} U\\ &= (P_z \textcircled{z} P_y \textcircled{y} P_x \textcircled{x}) e^{-\Lambda_z t} \textcircled{z} e^{-\Lambda_y t} \textcircled{y} e^{-\Lambda_x t} \textcircled{x} (P_z^{-1} \textcircled{z} P_y^{-1} \textcircled{y} P_x^{-1} \textcircled{x} U)\\ &= (P_z \textcircled{z} P_y \textcircled{y} P_x \textcircled{x})(e^*)^{-Ht} \odot (P_z^{-1} \textcircled{z} P_y^{-1} \textcircled{y} P_x^{-1} \textcircled{x} U).\end{aligned}$$

Using $\mathcal{L}(t)$ as an integration factor in (37) and integrating over $[t_n,\ t_{n+1}]$, we obtain

$$\begin{aligned}U_{n+1} &= \mathcal{L}(-\Delta t)U_n + \mathcal{L}(-\Delta t)\int_0^{\Delta t} \mathcal{L}(\tau) F_1(U(t_n+\tau), t_n+\tau)\, d\tau\\ &\qquad + \mathcal{L}(-\Delta t)\int_0^{\Delta t} \mathcal{L}(\tau) F_2(t_n+\tau)\, d\tau\\ &\triangleq \mathcal{L}(-\Delta t)U_n + (\tilde{F}_1)_n + (\tilde{F}_2)_n,\end{aligned} \tag{38}$$

which takes a similar form to (3) in the one-dimensional method. Motivated by the one-dimensional method, then, in the compact hIFE we approximate $(\tilde{F}_1)_n$ using the compact IIF and $(\tilde{F}_2)_n$ using the compact iETD. As in one dimension, the compact iETD approximates $F_2 \approx p(\tau)$ using Lagrange interpolation, and $\mathcal{L}(\tau)p(\tau)$ is integrated exactly. The compact IIF interpolates $\mathcal{L}(\tau)F_1 \approx q(\tau)$, and $q(\tau)$ is integrated exactly. The compact hIFE2 thus approximates

$$(\tilde{F}_1)_n \approx \frac{\Delta t}{2}\Big(F_1(U_{n+1}) + e^{\mathcal{L}_z \Delta t} \textcircled{z} e^{\mathcal{L}_y \Delta t} \textcircled{y} e^{\mathcal{L}_x \Delta t} \textcircled{x} F_1(U_n)\Big), \tag{39}$$

$$\begin{aligned}(\tilde{F}_2)_n \approx P_z \textcircled{z} P_y \textcircled{y} P_x \textcircled{x} \Bigg(&\Big(\frac{1 + (-1 + h_{i,j,k}\Delta t)e^{h_{i,j,k}\Delta t}}{h_{i,j,k}^2 \Delta t}\Big)_{i,j,k} \odot (\mathbf{F}_2)_n\\ &+ \Big(\frac{(-1 - h_{i,j,k}\Delta t) + e^{h_{i,j,k}\Delta t}}{h_{i,j,k}^2 \Delta t}\Big)_{i,j,k} \odot (\mathbf{F}_2)_{n+1}\Bigg).\end{aligned} \tag{40}$$

Note that these expressions are very similar to expression (7) for hIFE2 in one dimension.

5. **Application of hIFE to more complex systems.** In this section, we present several numerical simulations, demonstrating the advantages of hIFE2 in accuracy, complexity, and stability compared with the other methods. In Section 5.1, we apply IIF2, iETD2, fEIF2, and hIFE2 to a nonlinear reaction to demonstrate the advantages of hIFE2 over IIF2 and iETD2 in accuracy and complexity. Then, in Section 5.2, we apply all of the methods to a stiff system of coupled PDEs to demonstrate the stability advantage of hIFE2 over fEIF2. In both examples we choose systems with nonhomogeneous boundary conditions to highlight how hIFE2 can handle these (with the aid of the transformation from Section 3.2). In Section 5.3, we provide examples justifying the choice of F_1, F_2 in (6) compared with the more "obvious" choice. Finally, in Section 5.4, we provide an example of the compact hIFE2 in three dimensions.

5.1. **Reaction–diffusion equation with a nonlinear reaction term.** All the reactions we have considered in the analysis so far have been linear in u. We now consider an equation with a nonlinear reaction term and, just for good measure, a space-dependent term as well. We compare the obtained solutions using IIF2, iETD2, fEIF2, and hIFE2 in terms of accuracy and complexity. We start with the equation

$$\begin{cases} u_t = du_{xx} + u^2 - e^{-2dt}\sin^2 x, & 0 \le x \le \dfrac{\pi}{2}, \\ u_x|_{x=0} = e^{-dt}, \quad u|_{x=\frac{\pi}{2}} = e^{-dt}, \\ u(x,0) = \sin x, \end{cases} \tag{41}$$

which has the exact solution

$$u = e^{-dt}\sin x. \tag{42}$$

Since the boundary conditions are nonhomogeneous, we apply the transformation introduced in Section 3.2 to the equation, setting

$$u_B = e^{-dt}\left(\frac{-(x-\frac{\pi}{2})^2}{\pi} + \frac{4x^2}{\pi^2}\right)$$

so that we obtain a transformed equation for $\tilde{u} = u - u_B$,

$$\tilde{u}_t = d\tilde{u}_{xx} + \tilde{u}^2 + 2u_B\tilde{u} + u_B^2 + du_B + de^{-dt}\left(-\frac{2}{\pi} + \frac{8}{\pi^2}\right) - e^{-2dt}\sin^2 x, \tag{43}$$

with homogeneous boundary conditions. Since fEIF2 is supposed to work without transforming the boundary conditions, we apply it to the untransformed equation (41), and we apply IIF2, iETD2, and hIFE2 to the transformed equation (43). In both cases, we choose a uniform spatial grid with $\Delta x = \pi/2N$ and set the time step to vary proportional to the spatial grid, $\Delta t = 0.1\Delta x$. We further set the diffusion constant $d = 2$ and approximate the solution through time $T = 1$. Since fEIF2 is a second-order explicit method, we require knowledge of the discretized U at two time steps to begin the approximation when we are only given the initial condition. To determine the second time step, $U_1 = U(\Delta t)$, we modify fEIF2 to use the first-order eETD1 on the reaction terms and iETD2 on the boundary terms. Then we proceed with the usual fEIF2 for later time steps. For solving the nonlinear equations in the implicit methods, IIF2, iETD2, and hIFE2, we use Newton's method [18] with a tolerance of 10^{-8} and a maximum of 15 iterations.

We compare the results of the four methods for various N in terms of the L^∞ error at $T = 1$ and CPU time for the computation in Table 4. We include both the CPU time ("CPU time 1") for forming the exponential-like matrices (Appendix C) and

("CPU time 2") for the actual iterations. The total CPU time is listed in a separate column. We see that, consistent with Section 2, IIF2 does not attain second-order accuracy, especially for large N, while the other methods do. In addition, consistent with Appendix B, the ratio of CPU time among IIF2:hIFE2:fEIF2 is around 1:3:5, and the computation time of iETD2 is much higher than the others.

In this example, hIFE2 and fEIF2 perform similarly and exhibit a major advantage in accuracy compared with IIF2 and computational speed compared with iETD2. Despite being implicit, hIFE2 even has a minor advantage in computational time over fEIF2.

5.2. **Stiff system of coupled reaction–diffusion equations.** In the previous example, hIFE2 and fEIF2 performed similarly in terms of accuracy and complexity. We note, however, that the IIF2, iETD2, and hIFE2 methods are A-stable whereas fEIF2 is not. This advantage in stability will be significant for systems with stiff reactions, which we now investigate. In the system of PDEs,

$$\begin{cases} u_t = du_{xx} - au + v, \\ v_t = dv_{xx} - bv, \\ u_x|_{x=0} = e^{-(a+d)t} + e^{-(b+d)t}, \quad v_x|_{x=0} = (a-b)e^{-(b+d)t}, \\ u|_{x=\frac{\pi}{2}} = e^{-(a+d)t} + e^{-(b+d)t}, \quad v|_{x=\frac{\pi}{2}} = (a-b)e^{-(b+d)t}, \\ u(x,0) = 2\sin x, \\ v(x,0) = (a-b)\sin x, \end{cases} \tag{44}$$

whose exact solution is

$$\begin{aligned} u(x,t) &= (e^{-(a+d)t} + e^{-(b+d)t})\sin(x), \\ v(x,t) &= (a-b)e^{-(b+d)t}\sin(x), \end{aligned} \tag{45}$$

if a and b have very different magnitudes, the resulting system will be stiff. As such, we compare the performance of IIF2, iETD2, fEIF2, and hIFE2 on this system of PDEs.

We transform the equation to make the boundary conditions homogeneous and apply IIF2, iETD2, and hIFE2 to the transformed system whereas we apply fEIF2 to the untransformed system. The corresponding auxiliary functions are

$$\begin{cases} u_B = (e^{-(a+d)t} + e^{-(b+d)t})\left(\dfrac{-(x-\frac{\pi}{2})^2}{\pi} + \dfrac{4x^2}{\pi^2}\right), \\ v_B = (a-b)e^{-(b+d)t}\left(\dfrac{-(x-\frac{\pi}{2})^2}{\pi} + \dfrac{4x^2}{\pi^2}\right). \end{cases}$$

In each of the simulations, we fix the spatial mesh size $\Delta x = \pi/2N$ with $N = 1024$ and run each method through K time steps to a final time T (so $\Delta t = T/K$). We set $a = 500$, $b = -2$, $d = 0.1$, $T = 1$, and include the results for various K in Table 5. As the magnitude of the solutions is large, we also include the relative error in this table.

In this example, hIFE2 is still more accurate than IIF2 and faster than iETD2. The fEIF2 method, however, suffers from stability issues. The numerical solution obtained with fEIF2 blows up when the time step size is not small enough. Meanwhile, the other three methods never suffer from this blow-up issue since they are all A-stable. When $K = 320$ (so $\Delta t = 1/320 = 3.125 \times 10^{-3}$), the error in the numerical solution of fEIF2 is still huge while the relative error of hIFE2 is already

	N	L^∞ error	Order	CPU time (s)	CPU time 1 (s)	CPU time 2 (s)
IIF2	8	0.00228	-	0.09	0.05	0.04
	16	0.000591	1.95	0.04	0.02	0.02
	32	0.000198	1.58	0.07	0.03	0.04
	64	7.81e-05	1.34	0.13	0.04	0.09
	128	0.000108	−0.46	0.54	0.07	0.47
	256	5.18e-05	1.06	1.26	0.23	1.03
	512	1.83e-05	1.50	4.00	1.39	2.61
	1024	2.07e-05	−0.18	28.30	7.75	20.55
	2048	1.07e-05	0.96	168.12	42.10	126.02
	4096	5.35e-06	1.00	1148.42	265.35	883.07
	N	L^∞ error	Order	CPU time (s)	CPU time 1 (s)	CPU time 2 (s)
iETD2	8	0.00216	-	0.07	0.04	0.03
	16	0.000539	2.00	0.07	0.04	0.03
	32	0.000135	2.00	0.12	0.06	0.06
	64	3.37e-05	2.00	0.80	0.07	0.73
	128	8.41e-06	2.00	3.78	0.16	3.62
	256	2.1e-06	2.00	22.99	0.54	22.45
	512	5.26e-07	2.00	289.66	2.70	286.96
	1024	1.32e-07	2.00	2841.66	14.65	2827.01
	2048	3.31e-08	1.99	35348.32	91.84	35256.48
	4096	-	-	too long	-	-
	N	L^∞ error	Order	CPU time (s)	CPU time 1 (s)	CPU time 2 (s)
hIFE2	8	0.00217	-	0.12	0.09	0.03
	16	0.000544	1.99	0.06	0.04	0.02
	32	0.000137	1.99	0.08	0.05	0.03
	64	3.42e-05	2.00	0.16	0.08	0.08
	128	8.75e-06	1.97	0.76	0.17	0.59
	256	2.21e-06	1.99	1.85	0.54	1.31
	512	5.53e-07	2.00	9.17	2.61	6.56
	1024	1.49e-07	1.89	61.82	14.20	47.62
	2048	3.93e-08	1.93	419.24	89.49	329.75
	4096	1.12e-08	1.81	3096.23	603.04	2493.19
	N	L^∞ error	Order	CPU time (s)	CPU time 1 (s)	CPU time 2 (s)
fEIF2	8	0.00216	-	0.37	0.37	0.00
	16	0.00054	2.00	0.04	0.04	0.00
	32	0.000135	2.00	0.07	0.07	0.00
	64	3.38e-05	2.00	0.09	0.08	0.01
	128	8.44e-06	2.00	0.54	0.18	0.36
	256	2.11e-06	2.00	1.41	0.69	0.72
	512	5.28e-07	2.00	11.62	3.01	8.61
	1024	1.32e-07	2.00	84.11	16.11	68.00
	2048	3.31e-08	1.99	613.91	101.12	512.79
	4096	8.89e-09	1.90	4700.11	707.64	3992.47

TABLE 4. Numerical errors in terms of the maximum norm and CPU time for the various methods on the example in Section 5.1 at $T = 1$ with diffusion coefficient $d = 2$. Here N is the number of grid points in the spatial discretization ($\Delta x = \pi/2N$), and the time step $\Delta t = 0.1\Delta x$. "CPU time 1" is the CPU time for initializing the matrices (Appendix C), "CPU time 2" is the CPU time for the iterations, and "CPU time" is the sum of the two.

very small. A-stability in solving stiff equations is thus a significant advantage of hIFE2 over fEIF2, cementing its position as the most versatile of the four methods examined in this paper.

5.3. **Justification of the chosen splitting of the reaction terms.** Recall that the motivation for the hIFE2 method was that IIF2 could not handle time-dependent reaction terms with second-order temporal accuracy while iETD2 could. For the hIFE2 method, we then defined a splitting of the reaction term in (6) that might at first glance seem more complicated than necessary. At the level of the

	K	L^∞ error	Relative error	Order	CPU time (s)	CPU time 1 (s)	CPU time 2 (s)
IIF2	20	10	0.00381	-	5.32	5.25	0.07
	40	4.81	0.00182	1.06	5.07	4.91	0.16
	80	2.32	0.000881	1.05	5.09	4.78	0.31
	160	1.12	0.000425	1.05	5.07	4.44	0.63
	320	0.534	0.000203	1.07	5.24	3.90	1.34
	640	0.251	9.51e-05	1.09	5.92	3.40	2.52
	1280	0.115	4.34e-05	1.13	7.90	2.92	4.98
	2560	0.0503	1.91e-05	1.19	12.84	2.55	10.29
	K	L^∞ error	Relative error	Order	CPU time (s)	CPU time 1 (s)	CPU time 2 (s)
iETD2	20	3.99	0.00151	-	19.63	10.88	8.75
	40	0.994	0.000377	2.00	28.60	10.80	17.80
	80	0.248	9.41e-05	2.00	46.92	10.76	36.16
	160	0.0617	2.34e-05	2.01	80.10	10.41	69.69
	320	0.0152	5.76e-06	2.02	148.60	9.80	138.80
	640	0.00366	1.39e-06	2.05	285.20	9.27	275.93
	1280	0.000872	3.31e-07	2.07	567.11	8.94	558.17
	2560	0.000227	8.61e-08	1.94	1140.59	8.49	1132.10
	K	L^∞ error	Relative error	Order	CPU time (s)	CPU time 1 (s)	CPU time 2 (s)
hIFE2	20	4.19	0.000397	-	11.11	0.00	0.28
	40	1.05	0.000397	2.00	11.96	11.39	0.57
	80	0.261	9.91e-05	2.00	11.61	10.70	0.91
	160	0.0652	2.47e-05	2.00	12.36	10.37	1.99
	320	0.0162	6.14e-06	2.01	13.90	9.84	4.06
	640	0.00397	1.51e-06	2.03	17.65	9.43	8.22
	1280	0.000971	3.68e-07	2.03	25.08	8.88	16.20
	2560	0.000256	9.72e-08	1.92	40.83	8.45	32.38
	K	L^∞ error	Relative error	Order	CPU time (s)	CPU time 1 (s)	CPU time 2 (s)
fEIF2	20	1.49e+29	4.43e+25	-	12.42	11.96	0.46
	40	2.9e+48	8.65e+44	−64.08	12.61	11.78	0.83
	80	6.04e+73	1.8e+70	−84.11	13.07	11.46	1.61
	160	2.27e+96	6.77e+92	−74.99	14.43	11.20	3.23
	320	1.92e+79	5.71e+75	56.72	17.20	10.59	6.61
	640	0.251	7.48e-05	265.37	23.47	9.93	13.54
	1280	0.119	3.54e-05	1.08	35.96	9.57	26.39
	2560	0.0603	1.8e-05	0.98	62.05	9.08	52.97

TABLE 5. Numerical errors and CPU time for the test in Section 5.2 at time $T = 1$. We set the diffusion coefficient $d = 0.1$ and the coefficients of the reactions $a = 500$ and $b = -2$. For each simulation, we fix the number of grid points $N = 1024$ ($\Delta x = \pi/2N$), and run the simulation for K time steps ($\Delta t = T/K$). The error e is measured in the maximum norm, and the relative error is defined by $e/\max\{\|U_K\|_\infty, \|V_K\|_\infty\}$, where U_K and V_K are the numerical solutions after K time steps. "CPU time 1" is the CPU time for initialization (Appendix C), "CPU time 2" is the CPU time for the iterations, and "CPU time" is the sum of the two.

undiscretized PDEs, the method splits the reaction $f(u,x,t) = f_1(u,x,t) + f_2(x,t)$, where

$$f_1(u,x,t) = f(u,x,t) - f(0,x,t), \quad f_2(x,t) = f(0,x,t). \tag{46}$$

In this section we demonstrate by means of two examples why we suggest this decomposition over what one may consider a more "obvious" one.

Consider a general reaction–diffusion equation in one dimension with homogeneous boundary conditions,

$$\begin{cases} u_t = u_{xx} + f(u,x,t), & 0 \le x \le \dfrac{\pi}{2}, \\ u_x|_{x=0} = 0, \quad u|_{x=\frac{\pi}{2}} = 0, \\ u(x,0) = \cos x. \end{cases}$$

5.3.1. *Straightforward decomposition.* First consider the reaction

$$f(u,x,t) = \cos u + t. \tag{47}$$

Since hIFE2 aims to treat time-dependent terms with iETD2 and autonomous terms with IIF2, one might be led to choose the straightforward decomposition of the reaction

$$f_1(u,x,t) = \cos(u), \quad f_2(x,t) = t. \tag{48}$$

The decomposition in (46), however, would lead to the splitting

$$f_1(u,x,t) = \cos(u) - 1, \quad f_2(x,t) = 1 + t. \tag{49}$$

We include the numerical results of each of the decompositions (48) and (49) for various spatial mesh sizes, N, fixing $\Delta t = 0.1\Delta x$ in Table 6(A). We see that indeed the decomposition (49) outperforms the naïve decomposition (48) for each value of N since it remains second order while the naïve decomposition does not.

Numerical results aside, however, there are other problems with naïvely splitting the reaction into time-dependent and autonomous terms. Namely, what does one do with a term such as ut, $\cos(t)u^2$, or e^{tu}? Should they be included in f_1 or f_2? The decomposition (46) provides a framework for even these more complicated reactions and so is more desirable than the naïve decomposition, especially since it seems to give more accurate results anyway.

5.3.2. *More complicated reactions.* In this section, we provide an example of a more complicated reaction. We construct a reaction that contains x and t, and is unable to be fully decomposed into an autonomous term and an explicitly time-dependent term:

$$f(u,x,t) = (t+1)\cos(xu) + xe^t. \tag{50}$$

The naïve decomposition is

$$f_1(u,x,t) = (t+1)\cos(xu), \quad f_2(x,t) = xe^t, \tag{51}$$

and the decomposition followed by (46) is

$$f_1(u,x,t) = (t+1)\cos(xu) - (t+1), \quad f_2(x,t) = t + 1 + xe^t. \tag{52}$$

The numerical errors for various N, similar to the previous example, are shown in Table 6(B). Again, we see that the decomposition (52) outperforms the naïve decomposition (51) for each value of N since it remains second order while the naïve decomposition does not. The hIFE2 method is thus able to accurately handle even complicated equations with temporal and spatial variables present in both of the decomposed terms.

Remark. Despite the appearance of explicitly time-dependent term in f_1, the hIFE2 method can achieve second–order accuracy with large time step as long as $f_1(0,x,t) = 0$ is held. This example shows that hIFE2 is also able to handle complicated reactions efficiently.

5.4. **Reaction–diffusion system with nonhomogeneous boundary conditions in three dimensions.** Finally, we include an example of the compact hIFE2 applied to a three-dimensional system with nonhomogeneous boundary conditions,

$$\begin{cases} u_t = d_1 u_{xx} + d_2 u_{yy} + d_3 u_{zz} + ru, & (x,y,z) \in \left[0, \dfrac{\pi}{2}\right]^3 \\ u_x|_{x=0} = u|_{x=\frac{\pi}{2}} = E\sin y \sin z, \\ u_y|_{y=0} = u|_{y=\frac{\pi}{2}} = E\sin x \sin z, \\ u_z|_{z=0} = u|_{z=\frac{\pi}{2}} = E\sin x \sin y, \end{cases}$$

(A)

	Decomposition (48)		Decomposition (49)	
N	A priori error	Order	A priori error	Order
16	0.00103	-	0.00102	-
32	0.000532	0.95	0.000255	2.00
64	0.000328	0.70	6.37e-05	2.00
128	8.21e-05	2.00	1.59e-05	2.00
256	0.000196	−1.25	3.98e-06	2.00
512	0.000106	0.88	9.95e-07	2.00
1024	8.69e-06	3.61	2.49e-07	2.00
2048	3.37e-05	−1.96	6.19e-08	2.01
4096	1.75e-05	0.95	1.42e-08	2.12

(B)

	Decomposition (51)		Decomposition (52)	
N	A priori error	Order	A priori error	Order
16	0.00979	-	0.00977	-
32	0.00245	2.00	0.00245	1.99
64	6.67e-04	1.88	0.000614	2.00
128	1.67e-04	2.00	0.000153	2.00
256	3.94e-04	-1.24	3.83e-05	2.00
512	2.13e-04	0.89	9.59e-06	2.00
1024	1.76e-05	3.60	2.4e-06	2.00
2048	6.75e-05	-1.94	5.99e-07	2.00
4096	3.50e-05	0.95	1.48e-07	2.02

TABLE 6. Numerical a priori error in applying hIFE2 to a one-dimensional reaction–diffusion system with (A) $f(u,x,t) = \cos u + t$ for the decomposition (48) and (49) and (B) $f(u,x,t) = (t+1)\cos(xu) + xe^t$ for the decomposition (51) and (52). The a priori error is defined by $\|u^N - u^{N/2}\|_\infty$, where N is the number of grid points in the spatial discretization. The simulations are run through time $T = 1$ with $\Delta x = \frac{\pi}{2N}$ and $\Delta t = 0.1\Delta x$.

where $E = e^{(-d_1-d_2-d_3+r)t}$. The exact solution is

$$u(x,y,z,t) = e^{(-d_1-d_2-d_3+r)t}\sin x \sin y \sin z.$$

As usual, we transform the boundary conditions to be homogeneous, yielding the auxiliary function

$$\begin{aligned} u_B =& E\left[C(x)\sin y\sin z + C(y)\sin x\sin z + C(z)\sin x\sin y\right. \\ &- C(x)C(y)\sin z - C(x)C(z)\sin y - C(y)C(z)\sin x \\ &\left. + C(x)C(y)C(z)\right], \end{aligned}$$

where $C(x) = \frac{-(x-\frac{\pi}{2})^2}{\pi} + \frac{4x^2}{\pi^2}$. We use a uniform mesh with $N_x = N_y = N_z = N$, and set $\Delta x = \pi/2N$ and time step $\Delta t = 0.1\Delta x$. We choose parameters $d_1 = d_2 = d_3 = 1$, $r = -1$, and $T = 1$. Applying compact hIFE2 to the transformed equation for $\tilde{u} = u - u_B$ with homogeneous boundary conditions, the numerical errors are listed in Table 7 for various N. We verify that indeed the method attains second-order temporal error as expected.

$N \times N \times N$	L^∞ error	Order
$4 \times 4 \times 4$	1.33e-03	-
$8 \times 8 \times 8$	3.28e-04	2.02
$16 \times 16 \times 16$	8.17e-05	2.01
$32 \times 32 \times 32$	2.04e-05	2.00
$64 \times 64 \times 64$	5.10e-05	2.00
$128 \times 128 \times 128$	1.27e-06	2.00

TABLE 7. Numerical errors in the maximum norm for hIFE2 applied to the example in Section 5.4. The spatial resolution is $\Delta x = \frac{\pi}{2N}$ in all three dimensions, the time step is $\Delta t = 0.1\Delta x$, the ending time is $T = 1$, and the coefficients are $d_1 = d_2 = d_3 = 1$ and $r = -1$.

6. **Conclusions and discussion.** IIF and iETD methods are two existing methods designed for dealing with stiffness in reaction–diffusion equations. When solving systems that have explicitly time-dependent reactions, these two methods behave differently in accuracy and efficiency. In particular, IIF2 requires extremely small time steps to exhibit the theoretical second-order temporal accuracy in practice, whereas the iETD2 method maintains the second order with relatively large time steps. On the other hand, the IIF2 method has the advantage of being more efficient than iETD2 when solving systems with nonlinear reactions owing to the lower computational cost per time step. The hybrid (hIFE) method intends to take advantage of the strength in both methods. In hIFE method, the key step is to split the reaction term into two parts by using (6), and to treat F_1 by IIF and F_2 by iETD. For complicated reactions, as long as the condition $F_1|_{t=0} = 0$ is held (i.e. a more general form than autonomous), the hIFE method exhibits theoretical order of accuracy with large time step sizes compared to spatial grid size. We have applied this hybrid method to reaction–diffusion systems containing explicitly time-dependent reactions, as well as systems with nonhomogeneous boundary conditions through a transformation. To reduce the cost associated with both the storage and computation of large matrices in high dimensions, we have incorporated the compact representation previously developed for IIF and ETD methods into the high-dimensional hIFE method. Based on various numerical simulations and comparisons with other schemes, the hIFE method is found to be more advantageous with respect to stability, efficiency, and accuracy in solving reaction–diffusion systems with more complicated reactions or nonhomogeneous boundary conditions. The stability and restriction on the time step Δt to exhibit second order for all methods presented in this paper are provided in Table 8.

Method	A-stability	Δt to exhibit second-order accuracy	
		Time-dependent reactions	Nonhomogeneous BCs
IIF2	Yes	$\mathcal{O}(\Delta x^2)$	$\leq \mathcal{O}(\Delta x^2)$
iETD2	Yes	$\mathcal{O}(1)$	-
fEIF2	No	$\mathcal{O}(1)$	$\mathcal{O}(1)$
hIFE2	Yes	$\mathcal{O}(1)$	$< \mathcal{O}(1)$
hIFE2 (transformed)	Yes	$\mathcal{O}(1)$	$\mathcal{O}(1)$

TABLE 8. A summary of the four methods: for their A-stability, and the restriction on Δt to exhibit second order, with explicitly time-dependent reactions or nonhomogeneous boundary conditions.

The computational complexity (Appendix B) of our method is based on a simple implementation in this work. There are other techniques to reduce the computational cost. For example, the discrete fast Fourier transformation [43, 48, 22] and Krylov method [39, 3, 19, 20, 30] are two effective ways to reduce the computational cost for matrix-vector multiplications. While the Newton's method has been used to solve the nonlinear equations in hIFE, the fixed–point method could be used to reduce the cost in each iteration whereas the number of iterations may increase. Because IIF2 and hIFE2 do not involve solving large nonlinear systems, its associated cost for the Newton's methods is significantly lower than the iETD2 method [34]. Together, hIFE2 will cost less computationally than iETD2 but slightly more than IIF2, as long as similar matrix-vector multiplication techniques and nonlinear solvers are used.

In high dimensions, the current compact hIFE method can only deal with systems on fixed rectangular domains and equally distributed meshes in spatial discretization. For more complicated geometry or unstructured meshes, hIFE method needs to be further improved, potentially coupled with Krylov method [3, 30], to reduce the computational cost. For systems in higher dimension (> 3), other techniques, such as the sparse grid technique, can be incorporated into this hybrid scheme to further improve the efficiency. In addition, the hybrid approach can potentially be applied to convection–reaction–diffusion equations with explicitly time-dependent reactions, nonhomogeneous boundary conditions or domains moving with time. It will be also interesting to generalize this hybrid method to reaction–diffusion equations with anisotropic diffusion or to PDEs with high-order spatial differential operators, such as Cahn–Hilliard equations with dynamic boundary conditions.

Acknowledgments. This work is supported by the NIH grants U01AR073159, R01GM107264, and R01NS095355; a grant from the Simons Foundation (594598, QN), and the NSF grant DMS1763272, DMS1562176 and DMS1762063. The authors would like to thank Floyd Maseda for proof-reading of the draft of this paper.

Appendix A. **The first-order exhibition of IIF2.** In Section 2, when dealing with explicitly time-dependent reactions, we conclude that IIF2 requires $\Delta t < \mathcal{O}(1/|\alpha|)$ in the scalar equation (8) or $\Delta t < \mathcal{O}(\Delta x^2/d)$ in semi-discrete form (16) to ensure second-order temporal accuracy, but we did not address the accuracy of IIF2 for Δt above this threshold. In Figures 1A and 2(A), however, we observe IIF2 exhibits first-order behavior when Δt is large. Motivated by this observation, we attempt to provide an explanation for the first-order behavior in this appendix.

First, we investigate the scalar equation (8), only considering explicitly time-dependent reactions, $f(u,t) = g(t)$, with $g \in C^\infty$ having an upper bound $M > 0$. We consider the global error at time $0 < T < \infty$, with a fixed $\Delta t = T/K$ (so K time steps) and various $\alpha < 0$ satisfying $\Delta t > \mathcal{O}(1/|\alpha|)$.

The one-step iteration of IIF2 at arbitrary time step t_n only depends on the value of u_{n-1} multiplied by an exponential factor $e^{\alpha\Delta t}$. Then since $\Delta t > \mathcal{O}(1/|\alpha|) \implies |\alpha\Delta t| > \mathcal{O}(1) \implies e^{\alpha\Delta t} \ll 1$. Indeed even for $\alpha\Delta t = -100$, $e^{\alpha\Delta t} \approx 10^{-44}$. Then for large $-\alpha$, the approximation u_n becomes essentially independent of previous iterates,

$$u_n = e^{\alpha\Delta t}u_{n-1} + \frac{\Delta t}{2}e^{\alpha\Delta t}g_{n-1} + \frac{\Delta t}{2}g_n \longrightarrow \frac{\Delta t}{2}g_n \quad (\alpha \to -\infty), \tag{53}$$

so that, in particular,

$$u_K \to \frac{\Delta t}{2} g_K = \frac{\Delta t}{2} g(T) \quad (\alpha \to -\infty). \tag{54}$$

Now the exact solution $u(T)$ can be written in integral form, which we can bound using the assumption $|g(t)| \le M$:

$$|u(T)| = \left| e^{\alpha T} u(0) + e^{\alpha T} \int_0^T e^{-\alpha t} g(t)\, dt \right| \le e^{\alpha T} |u(0)| + \left| \frac{e^{\alpha T} - 1}{\alpha} \right| M. \tag{55}$$

Then as $\alpha \to -\infty$, $|u(T)| \to 0$. Indeed, even for finite but large $-\alpha$, we have $|u(T)| \lesssim M/|\alpha|$. We thus have as a rough estimate of the global error $e(T, K) \triangleq |u(T) - u_K|$ in IIF2 for large $-\alpha$,

$$e(T, K) \lesssim \frac{\Delta t}{2} |g(T)| + \frac{1}{|\alpha|} M. \tag{56}$$

Since we are working under the assumption $\Delta t > \mathcal{O}(1/|\alpha|)$, $\frac{1}{|\alpha|} \ll \frac{\Delta t}{2}$, so that we have $e(T, K) = \mathcal{O}(\Delta t)$, first-order temporal error.

The same argument applies to IIF2 on the semi-discrete equation (16) with an explicitly time-dependent reaction. As demonstrated in Section 2.2, if we apply IIF2 to the diagonal system (22) and measure error with the maximum norm, the component that has the largest error is that with the most negative eigenvalue. This then requires $\Delta t < \mathcal{O}(1/|\lambda_N|) = \mathcal{O}(\Delta x^2/d)$ to observe second-order temporal error. Repeating the above procedure with $\Delta t > \mathcal{O}(\Delta x^2/d)$, the error of IIF2 in that component will be first order, causing the overall error to be first order as well.

Appendix B. **Complexity.** In this appendix, we compare the computational cost per iteration of iETD2, IIF2, hIFE2, and fEIF2, showing hIFE2 requires far fewer operations than iETD2 and a number of operations similar to that of the less-costly IIF2 method.

We remind the reader of the form of the iteration in each of the methods below for convenience:

$$\begin{aligned} U_{n+1}^{iETD2} &= e^{A\Delta t} U_n + \frac{I + (-I + A\Delta t) e^{A\Delta t}}{A^2 \Delta t} F_n + \frac{(-I - A\Delta t) + e^{A\Delta t}}{A^2 \Delta t} F_{n+1} \\ &\triangleq e^{A\Delta t} U_n + \Delta t \Big[L_1(A\Delta t) F_n + L_2(A\Delta t) F_{n+1} \Big], \end{aligned} \tag{57}$$

$$U_{n+1}^{IIF2} = e^{A\Delta t} U_n + \frac{\Delta t}{2} \big(e^{A\Delta t} F_n + F_{n+1} \big), \tag{58}$$

$$\begin{aligned} U_{n+1}^{hIFE2} &= e^{A\Delta t} U_n + \frac{\Delta t}{2} \big(e^{A\Delta t} (F_1)_n + (F_1)_{n+1} \big) \\ &\quad + \Delta t \Big[L_1(A\Delta t)(F_2)_n + L_2(A\Delta t)(F_2)_{n+1} \Big], \end{aligned} \tag{59}$$

$$\begin{aligned} U_{n+1}^{fEIF2} &= e^{A\Delta t} U_n \\ &\quad + \Delta t \Big[L_3(A\Delta t) F_{n-1} + L_4(A\Delta t) F_n + L_1(A\Delta t) B_n + L_2(A\Delta t) B_{n+1} \Big], \end{aligned} \tag{60}$$

where $F_1(U, t)$ and $F_2(t)$ were defined in (6), $L_3(A\Delta t)$ and $L_4(A\Delta t)$ are $N \times N$ matrices of similar form to L_1 and L_2 (i.e., including factors of $e^{A\Delta t}$) whose exact forms are tangential to this discussion but given in full in [22], and B is a vector related to nonhomogeneous boundary conditions that does not depend on U (see Section 3 for details). The formation of the exponential-like matrices, $L_i(A\Delta t)$, $i =$

1, 2, 3, 4, can be found in Appendix C. Assuming the exponential-like matrices have already been formed, the two main sources of computational cost are matrix–vector multiplication and solving implicit equations.

At the beginning of each iteration in iETD2, the vectors U_n and F_n are known from the previous iteration, and the matrices $e^{A\Delta t}$, $L_1(A\Delta t)$, and $L_2(A\Delta t)$ are assumed to have already been computed and stored. Then after performing two matrix–vector multiplications, requiring $\mathcal{O}(2N^2)$ operations, the system takes the form

$$U_{n+1} = \Delta t L_2(A\Delta t)F(U_{n+1}, t_{n+1}) + \text{known} \triangleq \mathcal{F}_1(U_{n+1}).$$

This is a nonlinear system of N equations for the components of U_{n+1} that can be solved, e.g., by Newton's method. During each step of Newton's method, however, the $N \times N$ Jacobian matrix $J_{\mathcal{F}_1}$ of $\mathcal{F}_1$ needs to be inverted, requiring $\mathcal{O}(N^3)$ operations since the matrix L_2 is not in general diagonal. This computation then dominates the cost of the method, pushing it up to $\mathcal{O}(kN^3)$ operations if Newton's method converges in k steps.

In IIF2, the vectors U_n and F_n are known, and we can form the vector $U_n + \frac{\Delta t}{2}F_n$ in just $\mathcal{O}(N)$ operations, so that only one matrix–vector multiplication is required for a total of $\mathcal{O}(N^2)$ operations. Then the system takes the form

$$U_{n+1} = \frac{\Delta t}{2}F(U_{n+1}, t_{n+1}) + \text{known} \triangleq, \mathcal{F}_2(U_{n+1})$$

which is again a nonlinear system. Since $\frac{\Delta t}{2}$ is just a number, though, we can decouple this system into N individual scalar equations, one for each component, eliminating the need to invert the Jacobian so that the total number of computations required to find a solution is reduced to $\mathcal{O}(kN)$ for convergence in k iterations. The initial multiplication then dominates the cost of this method, making its total complexity $\mathcal{O}(N^2)$, an entire order of magnitude smaller than iETD2.

Recall that the hIFE2 method splits $F(U, t)$ into two parts: $F_1(U, t)$ and $F_2(t)$. Since F_2 is only time-dependent, it is easy to calculate at each step, so in addition to the pre-determined vectors U_n, $(F_1)_n$, and $(F_2)_n$, $(F_2)_{n+1}$ can be calculated in $\mathcal{O}(N)$ operations at the beginning of the iteration. Similar to IIF2, we can form the vector $U_n + \frac{\Delta t}{2}(F_1)_n$ in only $\mathcal{O}(N)$ operations, so we only require three matrix–vector multiplications, a total of $\mathcal{O}(3N^2)$ operations. Since only F_2 is treated with iETD2 while F_1 is treated with IIF2, the nonlinear system obtained after these multiplications takes the same form as IIF2 and so can also be decoupled into N separate scalar equations and solved in $\mathcal{O}(kN)$ operations. Thus, the total complexity of this method is $\mathcal{O}(3N^2)$, slightly more than IIF2 but still an order of magnitude better than iETD2.

Finally, fEIF2 is a completely explicit method, so all vectors are known at the beginning of the iteration, and no nonlinear equations need to be solved. Each term involves a different matrix, and there are five in total, so this method requires $\mathcal{O}(5N^2)$ operations. Even though fEIF2 is an explicit method, the implicit hIFE2 requires fewer operations per iteration due to the ease with which the nonlinear equations are solved.

We summarize the complexity of these four methods in Table 9. The iETD2 method has the largest computational complexity of all the examined methods, a full order of magnitude larger than any of the others. The fEIF2 method has a higher computational complexity than hIFE2, and hIFE2 has a higher computational complexity than IIF2, which scores the best among the methods. We

conclude that since both hIFE2 and iETD2 were shown to remain second-order accurate in time (Section 2) for larger Δt than IIF2, since hIFE2 requires fewer computations per iteration than iETD2, it is the more desirable method, especially for large systems.

	Operations per iteration	Total complexity (ratio)
IIF2	$\mathcal{O}(N^2)$	1
iETD2	$\mathcal{O}(kN^3)$	$\mathcal{O}(kN)$
hIFE2	$\mathcal{O}(3N^2)$	3
fEIF2	$\mathcal{O}(5N^2)$	5

TABLE 9. A comparison of the computational complexity between the IIF2, iETD2, hIFE2, and fEIF2 methods.

Appendix C. **Exponential-like matrices formation.** When iETD2, IIF2, and hIFE2 are applied, those matrices involved in an exponential matrix only need to be formed once if the time step Δt is fixed. In our paper, the matrix $e^{A\Delta t}$ in (3) and (5) is approximated by the `expm()` function in MATLAB, and the two exponential-like matrices, $L_1(A\Delta t) = \frac{I+(-I+A\Delta t)e^{A\Delta t}}{A^2\Delta t^2}$ and $L_2(A\Delta t) = \frac{(-I-A\Delta t)+e^{A\Delta t}}{A^2\Delta t^2}$, in the iETD2 iteration (4) are obtained via a cutoff of Taylor series [4] to avoid cancellation errors related to inverting the matrix A.

We see that since A is diagonalizable, so are $L_1(A\Delta t)$ and $L_2(A\Delta t)$. We can thus build the matrices by operating on the eigenvalues one by one rather than performing many matrix multiplications. We only need to evaluate $L_1(\lambda\Delta t)$ and $L_2(\lambda\Delta t)$ for arbitrary eigenvalue λ of A. If $|\lambda\Delta t|$ is small, $L_1(\lambda\Delta t)$ and $L_2(\lambda\Delta t)$ are "$\frac{0}{0}$"-type fractions in their analytic forms, we will encounter a large cancellation error. We set a threshold $bd = 10$, where if $|\lambda_j\Delta t| < bd$, we use a cutoff of the Taylor series

$$\begin{aligned} L_1(\lambda\Delta t) &= \sum_{n=0}^{\infty} \frac{1}{(n+2)n!}(\lambda\Delta t)^n \\ L_2(\lambda\Delta t) &= \Delta t \sum_{n=0}^{\infty} \frac{1}{(n+2)!}(\lambda\Delta t)^n \end{aligned} \tag{61}$$

up to a chosen tolerance $TOL = 10^{-8}$, and otherwise use the direct expression in the fractional form. We provide the algorithm of $L_1(A\Delta t)$ for an instance, the formation of $L_2(A\Delta t)$ follows the same procedure.

REFERENCES

[1] D. Alonso, F. Bartumeus and J. Catalan, Mutual interference between predators can give rise to turing spatial patterns, *Ecology*, **83** (2002), 28–34.

[2] G. Beylkin, J. M. Keiser and L. Vozovoi, A new class of time discretization schemes for the solution of nonlinear PDEs, *Journal of Computational Physics*, **147** (1998), 362–387.

[3] S. Chen and Y.-T. Zhang, Krylov implicit integration factor methods for spatial discretization on high dimensional unstructured meshes: application to discontinuous Galerkin methods, *Journal of Computational Physics*, **230** (2011), 4336–4352.

[4] S. M. Cox and P. C. Matthews, Exponential time differencing for stiff systems, *Journal of Computational Physics*, **176** (2002), 430–455.

[5] P. D. Dale, J. A. Sherratt and P. K. Maini, Role of fibroblast migration in collagen fiber formation during fetal and adult dermal wound healing, *Bulletin of mathematical biology*, **59** (1997), 1077–1100.

Algorithm 1 Generate $L_1(A\Delta t)$

procedure
  $\Lambda = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_N) \leftarrow$ *diagonalization* : $[V, \Lambda] = eig(A)$ where $A = V\Lambda V^{-1}$
  for $j = 1 : N$ **do**
    if $|\lambda_j \Delta t| < bd$ **then**
      $m = 1,\ n = 0,\ \mu_j = 0,$
      while $m < TOL$ **do**
        $m = \frac{1}{(n+2)n!}(\lambda_j \Delta t)^n$ ▷ Use Taylor expansion
        $n \leftarrow n + 1$
        $\mu_j \leftarrow \mu_j + m$
      end while
    else
      $\mu_j = \frac{1+(-1+\lambda_j \Delta t)e^{\lambda_j \Delta t}}{(\lambda_j \Delta t)^2}$ ▷ Direct implementation
    end if
  end for
  $L_1(A\Delta t) \leftarrow V\mathrm{diag}(\mu_1, \mu_2, \ldots, \mu_N)V^{-1}$
end procedure

[6] Q. Du and W. Zhu, Stability analysis and application of the exponential time differencing schemes, *Journal of Computational Mathematics*, **22** (2014), 200–209.

[7] Q. Du and W. Zhu, Analysis and applications of the exponential time differencing schemes and their contour integration modifications, *BIT Numerical Mathematics*, **45** (2005), 307–328.

[8] A. Eldar, R. Dorfman, D. Weiss, H. Ashe, B.-Z. Shilo and N. Barkai, Robustness of the bmp morphogen gradient in drosophila embryonic patterning, *Nature*, **419** (2002), 304–308.

[9] E. Gallopoulos and Y. Saad, Efficient solution of parabolic equations by Krylov approximation methods, *SIAM Journal on Scientific and Statistical Computing*, **13** (1992), 1236–1264.

[10] M. Garvie and C. Trenchea, Analysis of two generic spatially extended predator-prey models, *Nonlinear Anal. Real World Appl.*

[11] M. R. Garvie, Finite-difference schemes for reaction–diffusion equations modeling predator–prey interactions in matlab, *Bulletin of mathematical biology*, **69** (2007), 931–956.

[12] A. Gierer and H. Meinhardt, A theory of biological pattern formation, *Biological Cybernetics*, **12** (1972), 30–39.

[13] M. Hochbruck and C. Lubich, On Krylov subspace approximations to the matrix exponential operator, *SIAM Journal on Numerical Analysis*, **34** (1997), 1911–1925.

[14] M. Hochbruck and A. Ostermann, Explicit exponential Runge–Kutta methods for semilinear parabolic problems, *SIAM Journal on Numerical Analysis*, **43** (2005), 1069–1090.

[15] M. Hochbruck and A. Ostermann, Exponential Runge–Kutta methods for parabolic problems, *Applied Numerical Mathematics*, **53** (2005), 323–339.

[16] M. Hochbruck and A. Ostermann, Exponential integrators, *Acta Numerica*, **19** (2010), 209–286.

[17] E. E. Holmes, M. A. Lewis, J. Banks and R. Veit, Partial differential equations in ecology: spatial interactions and population dynamics, *Ecology*, **75** (1994), 17–29.

[18] E. Isaacson and H. B. Keller, *Analysis of Numerical Methods*, John Wiley & Sons, Inc., New York-London-Sydney, 1966.

[19] T. Jiang and Y.-T. Zhang, Krylov implicit integration factor WENO methods for semilinear and fully nonlinear advection–diffusion–reaction equations, *Journal of Computational Physics*, **253** (2013), 368–388.

[20] T. Jiang and Y.-T. Zhang, Krylov single-step implicit integration factor WENO methods for advection–diffusion–reaction equations, *Journal of Computational Physics*, **311** (2016), 22–44.

[21] L. Ju, X. Liu and W. Leng, Compact implicit integration factor methods for a family of semilinear fourth-order parabolic equations, *Discrete & Continuous Dynamical Systems-Series B*, **19** (2014), 1667–1687.

[22] L. Ju, J. Zhang, L. Zhu and Q. Du, Fast explicit integration factor methods for semilinear parabolic equations, *Journal of Scientific Computing*, **62** (2015), 431–455.
[23] A.-K. Kassam and L. N. Trefethen, Fourth-order time-stepping for stiff PDEs, *SIAM Journal on Scientific Computing*, **26** (2005), 1214–1233.
[24] A. Kicheva, P. Pantazis, T. Bollenbach, Y. Kalaidzidis, T. Bittig, F. Jülicher and M. Gonzalez-Gaitan, Kinetics of morphogen gradient formation, *Science*, **315** (2007), 521–525.
[25] J. D. Lambert, *Numerical Methods for Ordinary Differential Systems: The Initial Value Problem*, John Wiley & Sons, Inc., 1991.
[26] A. D. Lander, Q. Nie and F. Y. Wan, Do morphogen gradients arise by diffusion?, *Developmental cell*, **2** (2002), 785–796.
[27] S. Larsson and V. Thomée, *Partial Differential Equations with Numerical Methods*, vol. 45, Springer-Verlag, Berlin, 2003.
[28] X. Liu and Q. Nie, Compact integration factor methods for complex domains and adaptive mesh refinement, *Journal of Computational Physics*, **229** (2010), 5692–5706.
[29] D. Lu and Y.-T. Zhang, Krylov integration factor method on sparse grids for high spatial dimension convection–diffusion equations, *Journal of Scientific Computing*, **69** (2016), 736–763.
[30] D. Lu and Y.-T. Zhang, Computational complexity study on Krylov integration factor WENO method for high spatial dimension convection–diffusion problems, *Journal of Scientific Computing*, **73** (2017), 980–1027.
[31] M. Machen and Y.-T. Zhang, Krylov implicit integration factor methods for semilinear fourth-order equations, *Mathematics*, **5** (2017), 63.
[32] R. E. Mickens, Nonstandard finite difference schemes for reaction-diffusion equations, *Numerical Methods for Partial Differential Equations*, **15** (1999), 201–214.
[33] Q. Nie, F. Y. Wan, Y.-T. Zhang and X. Liu, Compact integration factor methods in high spatial dimensions, *Journal of Computational Physics*, **227** (2008), 5238–5255.
[34] Q. Nie, Y.-T. Zhang and R. Zhao, Efficient semi-implicit schemes for stiff systems, *Journal of Computational Physics*, **214** (2006), 521–537.
[35] S. V. Petrovskii and H. Malchow, A minimal model of pattern formation in a prey-predator system, *Mathematical and Computer Modelling*, **29** (1999), 49–63.
[36] S. V. Petrovskii and H. Malchow, Wave of chaos: new mechanism of pattern formation in spatio-temporal population dynamics, *Theoretical population biology*, **59** (2001), 157–174.
[37] H. Risken, *The Fokker-Planck Equation. Methods of Solution and Applications*, Springer Series in Synergetics, 18. Springer-Verlag, Berlin, 1984.
[38] Y. Saad, Analysis of some Krylov subspace approximations to the matrix exponential operator, *SIAM Journal on Numerical Analysis*, **29** (1992), 209–228.
[39] J. C. Schulze, P. J. Schmid and J. L. Sesterhenn, Exponential time integration using Krylov subspaces, *International Journal for Numerical Methods in Fluids*, **60** (2009), 591–609.
[40] C. Ta, D. Wang and Q. Nie, An integration factor method for stochastic and stiff reaction–diffusion systems, *Journal of Computational Physics*, **295** (2015), 505–522.
[41] J. W. Thomas, *Numerical Partial Differential Equations: Finite Difference Methods*, Texts in Applied Mathematics, 33. Springer-Verlag, New York, 1999.
[42] A. M. Turing, The chemical basis of morphogenesis, *Bulletin of mathematical biology*, **52** (1990), 153–197.
[43] C. Van Loan, *Computational Frameworks for the Fast FOurier Transform*, vol. 10, SIAM, 1992.
[44] D. Wang, W. Chen and Q. Nie, Semi-implicit integration factor methods on sparse grids for high-dimensional systems, *Journal of Computational Physics*, **292** (2015), 43–55.
[45] D. Wang, L. Zhang and Q. Nie, Array-representation integration factor method for high-dimensional systems, *Journal of Computational Physics*, **258** (2014), 585–600.
[46] O. Wartlick, A. Kicheva and M. González-Gaitán, Morphogen gradient formation, *Cold Spring Harbor perspectives in biology*, **1** (2009), a001255.
[47] A.-M. Wazwaz, *Partial Differential Equations*, CRC Press, 2002.
[48] A. Wiegmann, Fast Poisson, fast Helmholtz and fast linear elastostatic solvers on rectangular parallelepipeds, *Lawrence Berkeley National Laboratory*.
[49] S. R. Yu, M. Burkhardt, M. Nowak, J. Ries, Z. Petrášek, S. Scholpp, P. Schwille and M. Brand, Fgf8 morphogen gradient forms by a source-sink mechanism with freely diffusing molecules, *Nature*, **461** (2009), 533–536.

[50] L. Zhang, A. D. Lander and Q. Nie, A reaction–diffusion mechanism influences cell lineage progression as a basis for formation, regeneration, and stability of intestinal crypts, *BMC Systems Biology*, **6** (2012), 93.

[51] S. Zhao, J. Ovadia, X. Liu, Y.-T. Zhang and Q. Nie, Operator splitting implicit integration factor methods for stiff reaction–diffusion–advection systems, *Journal of Computational Physics*, **230** (2011), 5996–6009.

[52] L. Zhu, L. Ju and W. Zhao, Fast high-order compact exponential time differencing runge–kutta methods for second-order semilinear parabolic equations, *Journal of Scientific Computing*, **67** (2016), 1043–1065.

[53] Y.-L. Zhu, X. Wu, I.-L. Chern and Z.-Z. Sun, *Derivative Securities and Difference Methods*, Springer, 2004.

Received October 2018; revised January 2019.

E-mail address: yuchiq@uci.edu
E-mail address: weitaoc@ucr.edu
E-mail address: qnie@math.uci.edu