

A first-order computational algorithm for reaction-diffusion type equations via primal-dual hybrid gradient method

Shu Liu^{*}, Siting Liu^{*}, Stanley Osher^{*}, Wuchen Li[†]

^{*} Department of Mathematics, University of California, Los Angeles

{shuliu, siting6, sjo}@math.ucla.edu

[†] Department of Mathematics, University of South Carolina

wuchen@mailbox.sc.edu

Abstract

We propose an easy-to-implement iterative method for resolving the implicit (or semi-implicit) schemes arising in solving reaction-diffusion (RD) type equations. We formulate the nonlinear time implicit scheme as a min-max saddle point problem and then apply the primal-dual hybrid gradient (PDHG) method. Suitable precondition matrices are applied to the PDHG method to accelerate the convergence of algorithms under different circumstances. Furthermore, our method is applicable to various discrete numerical schemes with high flexibility. From various numerical examples tested in this paper, the proposed method converges properly and can efficiently produce numerical solutions with sufficient accuracy.

Keywords— Primal-dual hybrid gradient algorithm; Reaction-diffusion equations; First order optimization algorithm; Implicit finite difference schemes; Preconditioners.

1 Introduction

Reaction-diffusion (RD) equations (systems) have broad applications in many scientific and engineering areas. In material science, the phase-field model is described by typical RD-type equations known as Allen-Cahn [1] or Cahn-Hilliard equations [4]. They are used to model the development of microstructures in a mixture of two or more materials or phases over time; In chemistry, RD systems are used to depict the reaction and diffusion phenomena of chemical species in which a variety of patterns are produced [37, 39]; RD systems are also ubiquitous tools in biology: They are widely used for modeling morphogenesis [11], as well as the evolution of species distribution in ecology system [34]. In recent years, researchers also found that RD equations are useful in modeling and predicting crimes [44].

Reaction-diffusion (RD) equations are nonlinear parabolic partial differential equations possessing the following general form

$$\frac{\partial u(x, t)}{\partial t} = \mathcal{L}u(x, t) + \mathcal{R}u(x, t) \quad \text{on } \Omega \subset \mathbb{R}^d, \quad (1)$$

with initial condition $u(\cdot, 0) = u_0$,

where \mathcal{L} is a certain non-positive definite differential operator associated with the diffusion process. For example, \mathcal{L} can be taken as the Laplace operator Δ or negative biharmonic operator $-\Delta^2$, or

more general operators with variable coefficients; \mathcal{R} is a nonlinear operator depicting the reaction process. The RD equation is usually equipped with either the Neumann boundary condition if Ω is an ordinary region in \mathbb{R}^d , or the periodic boundary condition if Ω is a periodic region \mathbb{T}^d . We can also extend the RD equation (1) from the 1D function u to multiple dimensional vector function \mathbf{U} :

$$\frac{\partial \mathbf{U}(x, t)}{\partial t} = \mathcal{L}\mathbf{U}(x, t) + \mathcal{R}\mathbf{U}(x, t) \quad \text{on } \Omega \subset \mathbb{R}^d, \quad (2)$$

with initial condition $\mathbf{U}(\cdot, 0) = \mathbf{U}_0$.

Equation (2) can also be equipped with either Neumann or periodic boundary conditions. We also call the equation (2) reaction-diffusion system.

In recent decades, numerical methods, including finite difference methods [32, 17, 24, 27, 7, 12, 42, 40, 41, 28, 29, 30] and finite element methods [24, 49, 19], have been developed for computing the reaction-diffusion type equations (systems). Several benchmark problems [26, 13] have also been introduced to verify the proposed methods' effectiveness.

In order to get rid of the restriction of the Courant–Friedrichs–Lewy (CFL) condition [15] on small time steps, most of the popular numerical schemes designed for solving reaction-diffusion equations (systems) in the aforementioned works of literature are implicit or semi-implicit. As one uses implicit or semi-implicit schemes for solving RD equations (systems) with nonlinear terms, Newton's method [2] is usually needed for solving the series of nonlinear equations arising from time discretization. However, Newton's method encounters several drawbacks that may affect the performance of the proposed numerical scheme, namely,

- Newton's method requires the initial guess position to be close enough to the exact solution of the nonlinear equation. Otherwise, Newton's method may diverge.
- When solving RD equations on mesh grids by Newton's method, in each iteration, one has to solve a large-scale linear equation involving the Jacobian matrix of a certain nonlinear function. Solving this large-scale linear equation for multiple Newton iterations could be challenging and time-costing.

In this paper, we introduce a method based on the Primal-Dual Hybrid Gradient method (PDHG) [50, 8, 47, 14, 25] for solving the nonlinear updates arising in the time discretization schemes of RD equations (systems) with satisfying speed and accuracy.

We sketch the proposed method as follows. We briefly illustrate the main idea by considering the following fully implicit, semi-discrete scheme of the RD equation at the t -th time step:

$$\frac{u^{t+1} - u^t}{h_t} = \mathcal{L}u^{t+1} + \mathcal{R}u^{t+1}. \quad (3)$$

In this case, u^t is given, and $h_t > 0$ is a stepsize. We need to solve for u^{t+1} . Consider the following function \mathcal{F}

$$\mathcal{F}(u) = u - h_t(\mathcal{L}u + \mathcal{R}u) - u^t. \quad (4)$$

The goal is to solve $\mathcal{F}(u) = 0$. If we consider the indicator function ι defined as

$$\iota(u) = \begin{cases} 0 & u = 0 \\ +\infty & u \neq 0. \end{cases}$$

Then the nonlinear functional equation $\mathcal{F}(u) = 0$ is equivalent to the minimization problem

$$\min_{u \in X} \iota(\mathcal{F}(u)),$$

where X is a certain linear functional space for u . Now since ι can be treated as the Legendre transform of the constant function 0, i.e., $\iota(u) = \sup_{p \in X^*} \{(p, u)\}$ (here X^* denotes the dual space of X), we can recast the above minimization problem as a min-max saddle point problem as follows

$$\min_{u \in X} \max_{p \in X^*} \{(p, \mathcal{F}(u))\}. \quad (5)$$

We denote $L(u, p) = (p, \mathcal{F}(u))$. To deal with the saddle problem (5), by leveraging the ideas proposed in the PDHG method, we evolve p, u via the following proximal algorithms with extrapolation on the dual variable p .

$$p_{n+1} = \operatorname{argmin}_{p \in X^*} \left\{ \frac{\|p - p_n\|_2^2}{2\tau_p} - L(u_n, p) \right\} = p_n + \tau_p \mathcal{F}(u_n), \quad (6)$$

$$\tilde{p}_{n+1} = p_{n+1} + \omega(p_{n+1} - p_n), \quad (7)$$

$$u_{n+1} = \operatorname{argmin}_{u \in X} \left\{ \frac{\|u - u_n\|_2^2}{2\tau_u} + L(u, \tilde{p}_{n+1}) \right\} = (\operatorname{Id} + \tau_u(\tilde{p}_{n+1}, \partial_u \mathcal{F}))^{-1}(u_n). \quad (8)$$

Here the extrapolation coefficient $\omega > 0$, τ_p, τ_u are time steps used to evolve u, p . We should remind the reader to distinguish the PDHG time steps τ_p, τ_u from the time step h_t of the reaction-diffusion equation. Recall u^{t+1} as the solution to $\mathcal{F}(u) = 0$, if we further assume that $\partial_u \mathcal{F}(u)$, as a linear map from X to X^* , is injective. Then it is not hard to verify that $u = u^{t+1}, p = 0$ is the equilibrium of the above dynamic (6) - (8). Thus, we may anticipate that, by evolving (6), (7), (8), u_k, p_k could converge to the desired equilibrium point $u^{t+1}, 0$.

Furthermore, since \mathcal{F} is usually nonlinear, the inversion in (8) cannot be directly evaluated. To mitigate this, we replace $L(u, \tilde{p}_{k+1})$ by its linearization $\hat{L}(u, \tilde{p}_{k+1}) = L(u_k, \tilde{p}_{k+1}) + (\partial_u L(u_k, \tilde{p}_{k+1}), u - u_k)$ at $u = u_k$. Thus the update of u_{k+1} will have an explicit form

$$u_{k+1} = u_k - \tau_u(\tilde{p}_{k+1}, \partial_u \mathcal{F}(u_k)). \quad (9)$$

As a result, we can evolve the discrete-time dynamic (6), (7), (9) for approximating the solution u^{t+1} of the nonlinear equation $\mathcal{F}(u) = 0$, and the explicit updating rules will enable us to deal with large-scale computational problems conveniently and efficiently. This work will mainly focus on applying such a PDHG algorithm to solve various types of reaction-diffusion equations (systems) up to satisfying accuracy and efficiency. Based on the discussion and presentation in this paper, our method may serve as a potential alternative to the widely used Newton-type algorithms for time-implicit updates of reaction-diffusion equations for time-implicit schemes.

It is worth mentioning that instead of designing and analyzing new discretization schemes for RD equations, our paper is mainly devoted to a strategy that can efficiently resolve the ready-made scheme. Thus, in our paper, we will omit most of the discussions on the properties of the numerical scheme but focus more on the implementing details and the effectiveness of the proposed PDHG method.

We clarify that the method is inspired by [31], in which the authors design a similar algorithm for solving multiple types of PDEs accompanied by 1-D examples. This paper will be more specific and focus on computing various 2-D RD equations (systems) with different boundary conditions.

It is also worth mentioning that introducing damping terms into wave equations to achieve faster stabilization [18] shares great similarity with the limiting stepsize version of applying the PDHG method to PDE-solving algorithms. On the other hand, PDHG methods are also utilized in [47] to solve nonlinear equations with theoretical convergence guarantee under different circumstances.

Furthermore, people have applied PDHG or first-order methods to compute time-implicit updates of Wasserstein gradient flows [5] and reaction-diffusions [6, 19]. Compared to them, the proposed approach work for general non-gradient flow reaction-diffusion equations. In recent research [10], the authors deal with the nonlinear saddle point problems via the transformed PDHG method, with the follow-up research [9] aiming at solving the nonlinear equations associated with a class of monotone operators. In recent works [48, 3], the authors utilize the weak forms of PDEs and deep learning techniques to compute high-dimensional PDEs. The algorithm in [48] can be formulated as a min-max saddle point problem and is directly solved by alternative stochastic gradient descent and ascent method. Although the proposed method shares similarities with the aforementioned research. It differs from them in the saddle point problem formulation and the computational scheme.

This paper is organized as follows. In section 2, we provide a brief introduction to the Primal-Dual Hybrid Gradients method; then we present the details of how we implement the PDHG method to update the finite difference schemes for RD equations (systems). We then provide some existing theoretical results on the convergence of the PDHG method. We demonstrate our numerical examples in section 3. Our numerical experiments cover well-known Allen-Cahn and Cahn-Hilliard equations; higher-order gradient flow that emerges from functionalized polymer research; RD system known as the Schnakenberg model, which originates from the study of steady chemical patterns; and RD systems involving nonlocal terms depicting the species evolution of wolves and deer. We conclude the work in section 4. Some of the future research directions will also be discussed in section 4.

2 PDHG method for reaction-diffusion equation

In recent years, The Primal-Dual Hybrid Gradient (PDHG) method [50, 49, 8] proves to be an efficient algorithm for solving saddle point problems emerging from imaging. This method is iterative and each of its iterations consists of alternative proximal steps together with a suitable extrapolation. We refer the readers to [8] for further details (both theoretical and experimental) of the method.

2.1 PDHG method for updating implicit finite difference schemes

To clearly convey our proposed idea, let us first consider the following reaction-diffusion equation as an illustrative example on 2D periodic region $\Omega = \mathbb{T}^2$. We assume Ω is square shaped and denote its side length as L .

$$\frac{\partial u(x, t)}{\partial t} = \lambda \Delta u(x, t) + f(u(x, t)), \quad u(x, 0) = u_0(x). \quad (10)$$

Here we assume λ is a positive constant coefficient; $f : \mathbb{R} \rightarrow \mathbb{R}$ is the nonlinear function depicting the reaction term. Since we assume Ω to be the periodic region, we use periodic boundary conditions (BC) for equation (10).

Although there are numerous pieces of research on designing numerical schemes for RD equations, to demonstrate how our method works, let us narrow down and focus on the implicit one-step finite difference (FD) scheme. Once we have demonstrated how to implement the method to this implicit scheme, such a method can be easily extended to general numerical schemes.

We discretize the time interval $[0, T]$ into N_t equal subintervals with length $h_t = T/N_t$. Suppose we discretize each side of the region Ω into N_x subintervals with space stepsize $h_x = L/N_x$, we choose the central difference scheme to discretize the Laplace operator Δ . We denote the discrete Laplace operator with the periodic boundary condition as $\text{Lap}_{h_x}^P$ which is an $N_x^2 \times N_x^2$ block-circulant matrix possessing the following form

$$\text{Lap}_{h_x}^P = \frac{1}{h_x^2} \begin{bmatrix} L & I & & I \\ I & L & I & \\ & \ddots & \ddots & \ddots \\ & & I & L & I \\ I & & & I & L \end{bmatrix}_{N_x \times N_x \text{ blocks}} \quad L = \begin{bmatrix} -4 & 1 & & 1 \\ 1 & -4 & 1 & \\ & \ddots & \ddots & \ddots \\ & & 1 & -4 & 1 \\ 1 & & & 1 & -4 \end{bmatrix}_{N_x \times N_x}.$$

Here I is the N_x by N_x identity matrix. We denote $U^k \in \mathbb{R}^{N_x^2}$ as the numerical solution of (10) at the k th time step. We vectorize the $N_x \times N_x$ square array along the column to form the 1D vector U^k . That is, for $l = i \cdot N_x + j$ with $1 \leq i \leq N_x$ and $1 \leq j \leq N_x$, U_l^k is the numerical approximation of $u((j-1)h_x, (i-1)h_x, kh_t)$.

Now the implicit one-step FD scheme for (10) is cast as

$$U^{k+1} - U^k = h_t(\lambda \text{Lap}_{h_x}^P U^{k+1} + f(U^{k+1})), \quad U^0 = U_0. \quad (11)$$

Here U_0 denotes the initial condition on mesh grid points. When solving for the numerical solution of (10), one has to sequentially solve a series of nonlinear equations as shown in (11). This is the place in which we should apply the PDHG method. Let us denote the function $F : \mathbb{R}^{N_x^2} \rightarrow \mathbb{R}^{N_x^2}$ as

$$F(U) = U - U^k - h_t(\lambda \text{Lap}_{h_x}^P U + f(U)). \quad (12)$$

We want to solve $F(U) = 0$. As discussed in the introduction, this is equivalent to minimizing $\iota(F(U))$, which can further be cast as the following min-max saddle point problem

$$\min_{U \in \mathbb{R}^{N_x^2}} \max_{P \in \mathbb{R}^{N_x^2}} \{L(U, P)\}, \quad (13)$$

where L is defined as $L(U, P) = P^\top F(U)$. As a result, solving the equation $F(U) = 0$ finally boils down to the min-max problem (13).

Now the PDHG method suggests the following gradient ascent-descent dynamic for solving (13).

$$P_{n+1} = \underset{P \in \mathbb{R}^{N_x^2}}{\text{argmin}} \left\{ \frac{\|P - P_n\|_2^2}{2\tau_p} + L(U_n, P) \right\} = P_n + \tau_p F(U_n); \quad (14)$$

$$\tilde{P}_{n+1} = P_{n+1} + \omega(P_{n+1} - P_n); \quad (15)$$

$$U_{n+1} = \underset{U \in \mathbb{R}^{N_x^2}}{\text{argmin}} \left\{ \frac{\|U - U_n\|_2^2}{2\tau_u} + L(U, \tilde{P}_{n+1}) \right\} = (\text{Id} + \tau_u \nabla_U F(\cdot)^\top \tilde{P}_{n+1})^{-1} U_n. \quad (16)$$

Similar to our discussion in the introduction, the third line above involves a nonlinear equation that cannot be directly solved. We thus can replace the term $L(U, \tilde{P}_{n+1})$ in (16) by the linearization $\hat{L}(U, P) = L(U_n, P) + \nabla_U L(U_n, P)(U - U_n)$, then (16) can be explicitly computed as

$$U_{n+1} = \underset{U \in \mathbb{R}^{N_x^2}}{\text{argmin}} \left\{ \frac{\|U - U_n\|_2^2}{2\tau_u} + \hat{L}(U, \tilde{P}_{n+1}) \right\} = U_n - \tau_u \nabla_U F(U_n)^\top \tilde{P}_{n+1}. \quad (17)$$

Let us denote U_* as the solution to $F(U) = 0$. It is not hard to tell that $(U_*, 0)$ is a critical point of the functional $L(U, P)$. Furthermore, $(U_*, 0)$ is the equilibrium point of the time-discrete dynamic (14), (15), (17).

To analyze the convergence speed to the equilibrium state $(U_*, 0)$, we first consider the affine case in which $F(U) = AU - b$ with A as an $N_x^2 \times N_x^2$ symmetric matrix. We have the following result, similar to the analysis carried out in [31].

Theorem 1 (Convergence speed in Linear, symmetric case). *We fix the extrapolation coefficient $\omega = 2$. Suppose we obtain the sequence $\{(U_n, P_n)\}_{n \geq 0}$ by evolving the PDHG dynamic (14), (15), (17) with initial condition (U_0, P_0) . Suppose $F(U) = AU$ with A symmetric and non-singular. Denote λ_{\max} as the maximum eigenvalue (in absolute value) of A , and denote κ as the condition number of A . Then $\{(U_n, P_n)\}$ will converge to $(U_*, 0)$ linearly if $\tau_u \tau_p \leq \frac{4}{3\lambda_{\max}^2}$. Then the maximum convergence speed is achieved when $\tau_u \tau_p = \frac{\eta_*}{\lambda_{\max}^2}$, with the optimal convergence rate $\gamma_* = \sqrt{1 - \frac{\eta_*}{\kappa^2}}$, i.e., we have for any $n \geq 1$, $\|(U_n, P_n) - (U_*, 0)\|_2 \leq \gamma_*^n \|(U_0, P_0) - (U_*, 0)\|_2$. Here $\eta_* = \eta_*(\kappa)$ is a function of κ . The range of η_* belongs to $[1, \frac{3}{4})$.*

The proof of the theorem is provided in Appendix A. The explicit form of $\eta_*(\kappa)$ and γ_* are given in remark 2 of Appendix A.

The optimal convergence rate γ_* will be very close to 1 if the condition number κ is large. Furthermore, one will require $O(\kappa^2)$ iterations for U_n to converge. This can be very expensive

when A is a large-scale matrix with a large condition number. For example, we consider the heat equation

$$\partial_t u(x, t) = \Delta u(x, t)$$

with periodic boundary conditions. We apply the one-step implicit scheme to this equation, i.e., we consider solving $(I - \lambda h_t \text{Lap}_{h_x}^P)U^{n+1} = U^n$ at each time step n . Thus $A = I - h_t \lambda \text{Lap}_{h_x}^P$. One can tell that the eigenvalues of A equal $1 + 4h_t \lambda N^2 \sin^2(\frac{\pi k}{N})$ for $1 \leq k \leq N$. When N is even, the condition number κ of A equals $1 + 4\lambda N^2 h_t$. Since we can get rid of the CFL condition by using the implicit scheme, we can pick $h_t \gg \frac{1}{N^2}$, which leads to $\kappa(A) \gg 1$. The convergence of the primitive PDHG method could be very slow, even for the heat equation.

As discussed in remark 2, γ_* approaches 0 if the condition number κ drops to 1. Hence, we need to control the condition number of the matrix A to achieve faster convergence speed. This motivates us to introduce the preconditioning technique to the PDHG method. As suggested in both [25] and [31], we replace the l^2 norm used in either $\|U - U_n\|_2$ or $\|P - P_n\|_2$ by the G -norm $\|\cdot\|_G$ which is defined as

$$\|v\|_G = \sqrt{v^\top G v},$$

with G as a symmetric, positive definite matrix. In this work, we mainly focus on substituting the norm $\|P - P_n\|_2$ with $\|P - P_n\|_G$. The PDHG method involving G -norm in its proximal step is sometimes named G -prox PDHG [25]. The dynamic obtained via such G -prox PDHG is shown below.

$$P_{n+1} = P_n + \tau_p G^{-1} F(U_n); \quad (18)$$

$$\tilde{P}_{n+1} = P_{n+1} + \omega(P_{n+1} - P_n); \quad (19)$$

$$U_{n+1} = U_n - \tau_u \nabla_U F(U_n)^\top \tilde{P}_{n+1}. \quad (20)$$

We should pause here to emphasize to the reader that the above three-line dynamic (18), (19), (20) will be the core gadget for our RD equation solver throughout the remaining part of the paper. Before we move on to further details on solving the RD equation (10) via G -prox PDHG dynamic, let us provide a little more explanation on how (18) - (20) improve the convergence speed γ_* . Analogous to Theorem 1, we have the following corollary for affine function F .

Corollary 1.1 (Convergence for G -prox dynamic). *Suppose we keep all the assumptions in Theorem 1, if we further assume that G commutes with A , i.e., $GA = AG$, then all the conclusions in Theorem 1 still hold except λ_{\max} now denotes the largest (in absolute value) eigenvalue of $A^\top G^{-1} A$, and κ^2 now is the condition number of $A^\top G^{-1} A$.*

It is now clear that if we can find a matrix G that approximates AA^\top well, then $A^\top G^{-1} A$ will be reasonably close to the identity matrix I . Thus the condition number of $A^\top G^{-1} A$ will hopefully remain close to 1. In such cases, by properly choosing the step size τ_u, τ_p such that $\tau_u \tau_p$ is close to 1, we can obtain a rather fast convergence rate γ_* .

Up to this stage, although most of our intuition and analysis on G -prox PDHG dynamic comes from the case when F is affine, it is natural to extend our treatment to the nonlinear $F(U)$ defined in (12). We may still anticipate the effectiveness of our method since (12) can be recast as

$$F(U) = (I - \lambda h_t \text{Lap}_{h_x}^P)U - U^k - h_t f(U),$$

which can be treated as an affine function with a nonlinear perturbation $h_t f(U)$ carrying the small h_t coefficient. We now discuss several details in applying the G -prox PDHG dynamic (18)-(20) to the above $F(U)$. We aim at evolving the following dynamic in order to update the implicit one-step scheme (11).

$$P_{n+1} = P_n + \tau_p G^{-1} (U_n - \lambda h_t \text{Lap}_{h_x}^P U_n - h_t f(U_n) - U^k); \quad (21)$$

$$\tilde{P}_{n+1} = P_{n+1} + \omega(P_{n+1} - P_n); \quad (22)$$

$$U_{n+1} = U_n - \tau_u (\tilde{P}_{n+1} - \lambda h_t \text{Lap}_{h_x}^P \tilde{P}_{n+1} - h_t f'(U_n) \odot \tilde{P}_{n+1}). \quad (23)$$

Initial guess It is natural to choose the initial value U_0 of the dynamic (21) - (23) as the computed result from the last time step k , i.e., we set $U_0 = U^k$; And we will simply set $P_0 = 0$. A more sophisticated choice for U_0 could be the numerical solution at time $k + 1$ obtained by a forward Euler scheme or an IMEX scheme [36, 24].

Choosing the matrix G The function $F(U)$ defined in (12) is dominated by the affine term $(I - \lambda h_t \text{Lap}_{h_x}^P)U - U^k$. It is then natural to choose $G = (I - \lambda h_t \text{Lap}_{h_x}^P)^2$ as the preconditioner matrix. If the nonlinear term $f(U)$ is a highly stiff term, we may also consider absorbing its Jacobian $\nabla_U f(U) = \text{diag}(f(U))$ into G . So, G can also be chosen as $G = (I - \lambda h_t \text{Lap}_{h_x}^P - \text{diag}(f(U)))^2$ in such case. However, there might be a trade-off in doing so: if $\text{diag}(f(U))$ does not have equal diagonal entries, $I - \lambda h_t \text{Lap}_{h_x}^P - \text{diag}(f(U))$ cannot be efficiently inverted by Fast Fourier Transform (FFT) or Discrete Cosine Transform (DCT) method. Nevertheless, in most of our experiments, we discover that choosing $G = (I - \lambda h_t \text{Lap}_{h_x}^P)^2$ is adequate for achieving satisfying convergence speed. For more general reaction-diffusion equations, we discover that the nonlinear function $F(U)$ is usually decomposed as the sum of the linear term AU and the nonlinear term $h_t f(U)$. The linear term AU can be treated as the dominating term of $F(U)$. It is then reasonable to choose $G = AA^\top$ or at least close to AA^\top as a decent preconditioner of our method. This strategy works properly on general RD equations such as the Cahn-Hilliard equation or higher-order equations arising in polymer science. We refer the reader to examples in section 3 for details.

Application of FFT for fast computation Making use of the Fast Fourier Transform (FFT) method, or more precisely, 2-dimensional FFT [22] to accelerate our computation is crucial in our method. There are two places where we should apply FFT. The first is where we compute $G^{-1}u$; the second is where we compute $\text{Lap}_{h_x}^P u$. We refer the readers to chapter 4.8 of [22] and the references therein for details on implementing FFT. We also refer the reader to the examples in section 3 for applying FFT to general RD equations.

Choose suitable stepsize For the affine case discussed in Corollary 1.1, if G is close to AA^\top , then $\lambda_{\max}(A^\top G^{-1}A)$ should be a close to 1, then $\tau_u \tau_p \leq \frac{4}{3\lambda_{\max}^2}$ indicates that we could choose stepsizes τ_u, τ_p rather large. In our practice, starting at $\tau_u = \tau_p = 0.8$ should be a reasonable choice. One can increase or decrease the stepsize based on the actual performance of the method.

Stopping criterion During our computation, we will set up a threshold δ for our method. After each iteration of the PDHG dynamic, we evaluate the l^2 norm of the residual $\text{Res}(U_n) = \frac{U_n - U^k}{h_t} - (\lambda \text{Lap}_{h_x}^P U_n + f(U_n))$, we terminate the PDHG iteration iff $\|\text{Res}(U_n)\|_2 \leq \delta$.

Remark 1 (Neumann boundary condition and DCT). *It is worth providing some further discussions on our treatment of the Neumann boundary condition (BC), i.e., for a particular rectangular region $\Omega \subset \mathbb{R}^2$, $\frac{\partial u}{\partial \bar{n}} = 0$ on $\partial\Omega$. We discretize both sides of Ω into $N_x - 1$ subintervals. Thus the space stepsize $h_x = \frac{L}{N_x - 1}$. Such discretization will lead to N_x^2 mesh grid points. For point $(ih_x, 0)$ on the vertical boundary of Ω , we apply the central difference scheme to the Neumann boundary condition at the midpoint $(ih_x, -\frac{1}{2}h_x)$, which leads to $\frac{U_{i,-1} - U_{i,0}}{h_x} = 0$, thus $U_{i,-1} = U_{i,0}$. Similar treatments are applied to the other boundaries of Ω . Suppose we consider using the central difference scheme to discretize the Laplacian Δ , let us denote the discretized Laplacian w.r.t. Neumann boundary condition as $\text{Lap}_{h_x}^N$, then $\text{Lap}_{h_x}^N$ takes the following form.*

$$\text{Lap}_{h_x}^N = \frac{1}{h_x^2} \begin{bmatrix} L_1 & I & & & & \\ I & L_2 & I & & & \\ & & \ddots & \ddots & \ddots & \\ & & & I & L_2 & I \\ & & & & I & L_1 \end{bmatrix}_{N_x \times N_x \text{ blocks}}. \quad (24)$$

Here the block matrices L_1, L_2 are

$$L_1 = \begin{bmatrix} -2 & 1 & & & & \\ 1 & -3 & 1 & & & \\ & \ddots & \ddots & \ddots & & \\ & & & 1 & -3 & 1 \\ & & & & 1 & -2 \end{bmatrix}_{N_x \times N_x}, \quad L_2 = \begin{bmatrix} -3 & 1 & & & & \\ 1 & -4 & 1 & & & \\ & \ddots & \ddots & \ddots & & \\ & & & 1 & -4 & 1 \\ & & & & 1 & -3 \end{bmatrix}_{N_x \times N_x}$$

Similar to using FFT for the computation involving $\text{Lap}_{h_x}^P$, we can use the Discrete Cosine Transform (DCT) [46] [22] to efficiently evaluate matrix-vector multiplication or solve linear equations involving the matrix $\text{Lap}_{h_x}^N$. To be more specific, we use the DCT-2 transform introduced in [46] in the 2-dimensional scenario which enjoys the $O(N_x^2 \log N_x)$ computational complexity.

We summarize our method in the following algorithm. It is not hard to tell that the total complexity of each inner PDHG method is $O(\#\{\text{PDHG iter}\} \cdot N_x^2 \log N_x)$.

Algorithm 1 PDHG method for updating implicit one-step FD scheme of RD equation

- 1: **Input:** Initial condition u_0 , terminal time T , number of time subintervals N_t ; region size L , number of space subintervals N_x ;
 - 2: Initialize $h_t = T/N_t$, $h_x = L/N_x$, $\{U_{ij}^0\} = \{u_0(ih_x, jh_x)\}$.
 - 3: **for** $0 \leq k \leq N_t - 1$ **do**
 - 4: Set initial condition: $U_0 = U^k$ (or \hat{U}^{k+1} obtained via explicit Euler or IMEX scheme).
 - 5: $n = 0$.
 - 6: **while** $\|\text{Res}(U_n)\|_2 \geq \delta$ **do**
 - 7: Evolve the G -prox PDHG dynamic (21) - (23) with help of 2D FFT(DCT):
 - 8: We use FFT for periodic BC and DCT for Neumann BC.
 - 9: Compute $V_n = \text{Lap}_{h_x}^P U_n$ via 2D FFT(DCT);
 - 10: Compute $W_n = U_n - \lambda h_t V_n - h_t f(U_n) - U^k$;
 - 11: Solve $GY_n = W_n$ via 2D FFT(DCT);
 - 12: Update $P_{n+1} = P_n + \tau_p Y_n$; $\tilde{P}_{n+1} = P_{n+1} + \omega(P_{n+1} - P_n)$;
 - 13: Compute $Q_{n+1} = \text{Lap}_{h_x}^P \tilde{P}_{n+1}$ via 2D FFT(DCT);
 - 14: Update $U_{n+1} = U_n - \tau_u(\tilde{P}_{n+1} - \lambda h_t Q_{n+1} - h_t f'(U_n) \odot \tilde{P}_{n+1})$;
 - 15: $n = n + 1$;
 - 16: **end while**
 - 17: Set $U^{k+1} = U_n$;
 - 18: **end for**
 - 19: **Output:** The numerical solution U^0, U^1, \dots, U^{N_t} .
-

In this section, we mainly focus on the one-step implicit finite difference (FD) scheme to illustrate how we apply the PDHG iterations to update the given FD scheme. But we should emphasize that our method is not restricted to such a scheme. One can extend the PDHG method to various types of numerical schemes by formulating the scheme at a certain time step k as a nonlinear equation $F^k(U) = 0$, and construct the functional $L^k(U, P) = P^\top F^k(U)$. Then one can apply the dynamic (18) - (20) to update the numerical solution from U^k to U^{k+1} . In addition, our method is applicable to more general reaction-diffusion equations (systems). Further discussions and details are supplied in section 3.

2.2 Discussion on convergence criteria and adaptive h_t method

Theorem 1 suggests that under the linear case, the convergence of PDHG method relies on condition number κ , which is directly related to h_x, h_t of our discrete scheme. In practice, we fix h_x and τ_u, τ_p in the algorithm. At every time step n we discover that when time stepsize h_t gets larger than a certain threshold value h_t^* which depends on h_x, τ_u, τ_p and n , PDHG method will hardly converge. The method works well when h_t is slightly smaller than the threshold. The theoretical study on how h_t^* guarantees the convergence of our method will be an important future research direction.

Adaptive time stepsize As discussed above, we cannot guarantee the convergence of the PDHG iteration (21) - (23) for any time stepsize h_t . Since the aforementioned threshold h_t^* may vary at different time stages, and how h_t^* varies depends on the nature of the equation as well as the discretization scheme. Given the potential difficulty of choosing the suitable h_t that guarantees both the numerical accuracy as well as the convergence of the PDHG iterations, we come up with the strategy of using adaptive stepsize h_t throughout the computation. We choose a time stepsize h_t^0 that guarantees the numerical accuracy and will serve as the upper bound of all h_t throughout our method, we also set up two integers $N^* > N_* > 0$ as the thresholding integers for enlarging or shrinking the time stepsize h_t . We also pick a rescaling coefficient $\eta \in (0, 1)$. During each time step n of the algorithm, we record the total number of PDHG iterations M_{PDHG} , and reset the stepsize h_t for the next time step based on the following rules:

- If $M_{\text{PDHG}} > N^*$, we shrink h_t by rate η , $h_t = \eta h_t$;
- If $N^* \geq M_{\text{PDHG}} \geq N_*$, we remain h_t unchanged;
- If $M_{\text{PDHG}} < N_*$, we enlarge h_t by rate $\frac{1}{\eta}$, $h_t = \frac{h_t}{\eta}$, if $\frac{h_t}{\eta} \leq h_t^0$; we remain h_t unchanged if $\frac{h_t}{\eta} > h_t^0$.

It is also reasonable to fix h_x, h_t but to only shrink the PDHG stepsizes τ_u, τ_p when we encounter difficulties in converging. However, according to our experience, shrinking τ_u, τ_p usually requires much more PDHG iterations for convergence, which may cause the algorithm less efficient compared with the aforementioned adaptive h_t strategy.

3 Numerical examples

In this section, we demonstrate some numerical results computed by the proposed method. Throughout our experiments, we always use the extrapolation coefficient $\omega = 1$, and set the initial condition U_0 as the numerical solution U^k computed from the last time step t_k . One can also try other values of ω or a more sophisticated initial guess of U_0 . Our experiences show that confining ω around 1 will probably provide the best performance of the PDHG method. Choosing U_0 as \hat{U}^{k+1} obtained by a specific explicit or IMEX scheme may slightly shorten the convergence time of the PDHG iteration. However, it is worth mentioning that when we are dealing with stiff equations, such treatment may introduce instability to the PDHG dynamic which may lead to the blow-up of the method.

Furthermore, as we have emphasized before, the mission of this paper is to verify the correctness and effectiveness of the proposed PDHG method in resolving the equation $F(U) = 0$ at each time step. Thus, in this work, we will mainly focus on the straightforward one-step implicit scheme (11) in all numerical examples by omitting further discussions and experiments on more sophisticated numerical schemes.

We use fixed time stepsize h_t in our numerical examples unless we emphasize that the adaptive h_t method is applied in the experiments.

Our numerical examples are computed in MATLAB on a laptop with 11th Gen Intel(R) Core(TM) i5-1135G7 @ 2.40GHz 2.42 GHz CPU.

3.1 Allen-Cahn equations

The Allen-Cahn equation [1] is a typical reaction-diffusion equation taking the following form

$$\frac{\partial u(x, t)}{\partial t} = a\Delta u(x, t) - bW'(u(x, t)), \text{ on } \Omega \subset \mathbb{R}^2, \quad u(\cdot, 0) = u_0. \quad (25)$$

Here $a, b > 0$ are positive coefficients,

$$W(u) = \frac{(u^2 - 1)^2}{4} \quad (26)$$

is a double-well potential function with its derivative $W'(u) = u^3 - u$. We will always assume periodic boundary conditions in our discussion.

The Allen-Cahn equation can be treated as the L^2 -gradient flow of the following functional $\mathcal{E}(u)$.

$$\mathcal{E}(u) = \int_{\Omega} \frac{1}{2}a|\nabla u|^2 + bW(u) \, dx. \quad (27)$$

3.1.1 Examples with shrinking level set curve

In the first example, we consider $\Omega = [-L, L]^2$ with $L = 0.25$. We consider taking $a = \epsilon$, $b = \frac{1}{\epsilon}$ with $\epsilon = 0.01$. Let us consider $u_0 = 2\chi_B - 1$. Here χ_E denotes the indicator function of measurable set E , i.e., $\chi_E(x) = 1$ if $x \in E$, and $\chi_E(x) = 0$ otherwise. We denote B as the disk centered at O with a radius equal to 0.2. Suppose we use periodic boundary conditions for (25). It is well-known that the zero-level-set curve of the solution $u(x, t)$ to Allen-Cahn equation behaves similarly to the mean curvature flow as time t increases [38, 33, 32]. In this case, we can treat the initial level set curve as the circle centered at the origin with radius $r(0) = 0.2$. As t increases, the circle radius will shrink at the rate of ϵ times circle curvature, i.e., $\dot{r}(t) = -\epsilon\kappa(t) = -\frac{\epsilon}{r(t)}$. Solving this equation leads to $r(t) = \sqrt{r(0)^2 - 2\epsilon t}$. Thus the level set circle will vanish at finite time $t = \frac{r(0)^2}{2\epsilon} = 2$.

As suggested in section 4.4 of [33], it is important to choose the spatial stepsize h_x small enough so that h_x no larger than $O(\epsilon)$ to capture the shrinkage of level set curve, otherwise, the numerical solution may get stuck at some intermediate stage. In this example, we solve the equation on time interval $[0, 3]$. We choose $N_t = 3000$, thus $h_t = 1/1000$; $N_x = 100$ with $h_x = L/N_x = 1/200$. Recall $h_x < \epsilon$. We choose $\tau_u = \tau_p = 0.5$ as the stepsize for the PDHG iteration. Some computed results are shown in Figure 1. Plots of the radial position as well as the moving speed of the front (zero level set circle) of the numerical solution are presented in Figure 2.

We apply the PDHG method described in Algorithm 1 to this problem. Although equation (1) contains a nonlinear term with a significant coefficient $\frac{1}{\epsilon}$, our proposed PDHG method still solves the nonlinear $F(U) = 0$ efficiently. To be more specific, we set the PDHG-threshold $\delta = 10^{-7}$. Most PDHG iterations will terminate in less than 200 steps for each discrete time step. The right plot of Figure 2 indicates the linear convergence of the proposed method.

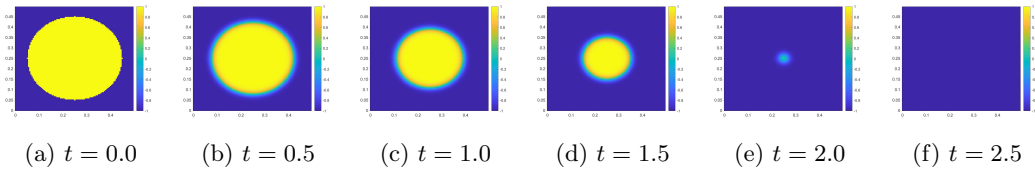
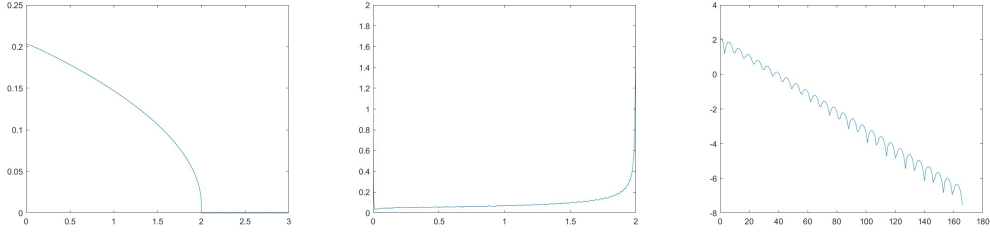


Figure 1: Numerical solution of (1) at different times with initial condition $u_0 = 2\chi_B - 1$.

We also solve equation (1) on $\Omega = [0, 0.5]^2$ within the time interval $[0, 0.5]$. We still set $a = \epsilon$, $b = \frac{1}{\epsilon}$ with $\epsilon = 0.01$. We pick $N_x = 100$, $N_t = 500$, thus $h_x = 1/200$, $h_t = 1/1000$. We consider



(a) Plot of the front position in radial direction versus time (b) Plot of the front speed (calculated from linear interpolation of the numerical solution) versus time (c) Plot of the front speed (calculated from finite difference) versus time

Figure 2: Plots of front position and speed (Left & Middle); Plots of $\log_{10} \text{Res}(U_n)$ versus PDHG iterations at time stage $t = 1.0$ (Right).

the initial condition $u_0 = 2\chi_E - 1$ with the region $E = (B_1 \setminus B_2) \cup (B_2 \setminus B_1)$, where B_1, B_2 are disks centered at $(0.2, 0.25)$, $(0.3, 0.25)$ with radius both equal to 0.1. We apply the PDHG method with $\tau_u = \tau_p = 0.5$ and obtain the numerical results in Figure 3. In this example, the PDHG method takes no more than 200 iterations for each time step update.

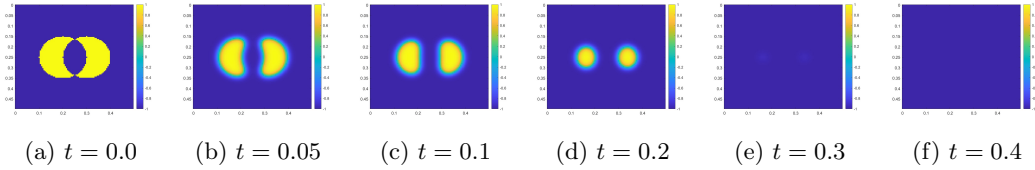


Figure 3: Numerical solution at different times with initial condition $u_0 = 2\chi_E - 1$. Notice that in the last plot, we have almost converged to the equilibrium solution $u = -1$.

3.2 Cahn-Hilliard equations

We now switch to another well-known reaction-diffusion equation known as the Cahn-Hilliard equation [4], which takes the following form.

$$\frac{\partial u(x, t)}{\partial t} = -a\Delta\Delta u(x, t) + b\Delta W'(u(x, t)), \quad \text{on } \Omega \subset \mathbb{R}^2, \quad u(\cdot, 0) = u_0. \quad (28)$$

Here we assume $a, b > 0$, and $W(u)$ defined the same as in the Allen-Cahn equation. In this section, we will restrict our discussion to periodic boundary conditions. Similar to the Allen-Cahn equation, the Cahn-Hilliard equation can be treated as the H^{-1} -gradient flow of the functional $\mathcal{E}(u)$ defined in (27). Due to this reason, compared with the Allen-Cahn equation, the Cahn-Hilliard equation involves one extra operator $-\Delta$ on the right-hand side of the equation. This difference leads to several slight modifications to our original algorithm.

The functional $F(U)$ introduced in (12) is now

$$F(U) = (I - \lambda h_t \text{Lap}_{h_x}^P \text{Lap}_{h_x}^P)U - U^k - h_t \text{Lap}_{h_x}^P f(U). \quad (29)$$

Thus $L(U, P) = P^\top ((I - \lambda h_t \text{Lap}_{h_x}^P \text{Lap}_{h_x}^P)U - U^k) - h_t P^\top \text{Lap}_{h_x}^P f(U)$. It is then natural to choose precondition matrix G as $(I - \lambda h_t \text{Lap}_{h_x}^P \text{Lap}_{h_x}^P)^2$. The three-step PDHG update for Cahn-Hilliard

i	1	2	3	4	5	6	7
x_i	$\pi/2$	$\pi/4$	$\pi/2$	π	$3\pi/2$	π	$3\pi/2$
y_i	$\pi/2$	$3\pi/4$	$5\pi/4$	$\pi/4$	$\pi/4$	π	$3\pi/2$
r_i	$\pi/5$	$2\pi/15$	$\pi/15$	$\pi/10$	$\pi/10$	$\pi/4$	$\pi/4$

Table 1: data 7 circles

equation can thus be formulated as

$$P_{n+1} = P_n + \tau_p G^{-1}(U_n - \lambda h_t \text{Lap}_{h_x}^P \text{Lap}_{h_x}^P U_n - h_t \text{Lap}_{h_x}^P f(U_n) - U^k); \quad (30)$$

$$\tilde{P}_{n+1} = P_{n+1} + \omega(P_{n+1} - P_n); \quad (31)$$

$$U_{n+1} = U_n - \tau_u(\tilde{P}_{n+1} - \lambda h_t \text{Lap}_{h_x}^P \text{Lap}_{h_x}^P \tilde{P}_{n+1} - h_t f'(U_n) \odot \text{Lap}_{h_x}^P \tilde{P}_{n+1}). \quad (32)$$

By carefully investigating the steps among (30) - (32), one can tell that both the linear equation involving G and the matrix-vector multiplication involving $\text{Lap}_{h_x}^P$ can be computed via FFT, which indicates the effectiveness of the computational scheme when applied to Cahn-Hilliard equations.

We demonstrate several numerical examples below.

3.2.1 Example with seven circles

Inspired by the second example introduced in [13], we consider Cahn-Hilliard equation (28) on periodic domain $\Omega = [0, 2\pi]^2$ with $a = 0.1^2$ and $b = 1$. We set the initial condition u_0 as

$$u_0(x, y) = -1 + \sum_{i=1}^7 \varphi(\sqrt{(x - x_i)^2 + (y - y_i)^2} - r_i),$$

where the mollifier function φ is defined as

$$\varphi(s) = \begin{cases} 2e^{-\frac{\epsilon^2}{s^2}} & s < 0; \\ 0 & s \geq 0 \end{cases}, \quad \text{with } \epsilon = 0.1.$$

One can think of $u_0(x, y)$ as an indicator function whose value equals +1 if (x, y) falls into any of the seven circles; and equals -1 otherwise. Furthermore, we set the centers and radii of the seven circles as in Table 1.

We will solve equation (28) on the time interval $[0, 30]$. In our numerical implementation, we set $N_x = 128$, $h_x = \pi/64$; $N_t = 6000$, $h_t = 1/200$. For the PDHG iteration, we set $\tau_u = \tau_p = 0.5$. The numerical solution to this equation is demonstrated in Figure 4. The plots of log residuals at different time stages are also presented in Figure 4, which exhibit the linear convergence of the PDHG algorithm. The small circles will gradually fade out, leaving the largest circle in the center of the domain till the end. By analyzing our numerical solution, the time T_1 at which the value of our numerical solution is evaluated at $(\pi/2, \pi/2)$ passes 0 is located in the interval $[6.340, 6.345]$; while the time T_2 at which our numerical value evaluated at $(3\pi/2, 3\pi/2)$ passes 0 is located in the interval $[26.015, 26.020]$. Both times meet the accuracy proposed in [13].

3.2.2 Example with sinusoidal initial condition

In this section, we follow example 4 proposed in [13] to compute (28) on $\Omega = [0, 2\pi]^2$. We set $a = \frac{\pi^2}{25000}$, $b = 1$. The initial condition is set as

$$u_0(x, y) = 0.05(\cos(3x)\cos(4y) + (\cos(4x)\cos(3y))^2 + \cos(x - 5y)\cos(2x - y)).$$

We solve the equation on the time interval $[0, 8]$. We set $N_x = 256$, $h_x = \pi/128$; $N_t = 24000$, $h_t = 1/3000$. For the PDHG part, we choose $\tau_u = \tau_p = 0.5$. The PDHG iteration is working efficiently at every time stepsize. Some numerical plots are shown in Figure 5.

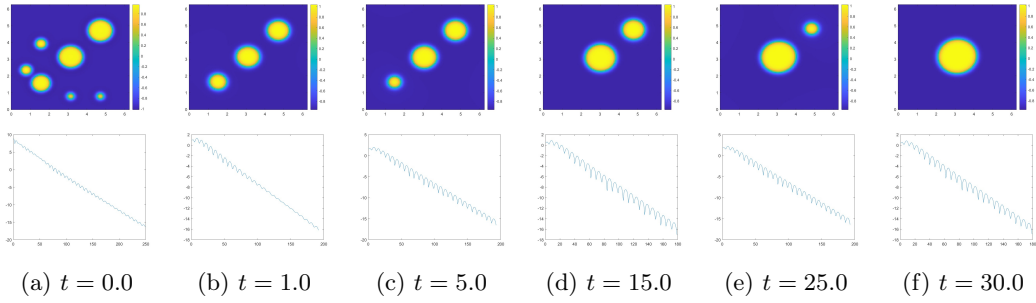


Figure 4: Numerical solution and $\log_{10} \text{Res}(U_n)$ plot at different time stages for the seven circle example. The residual plots indicate the linear convergence of PDHG method for the nonlinear objective functions used in this example.

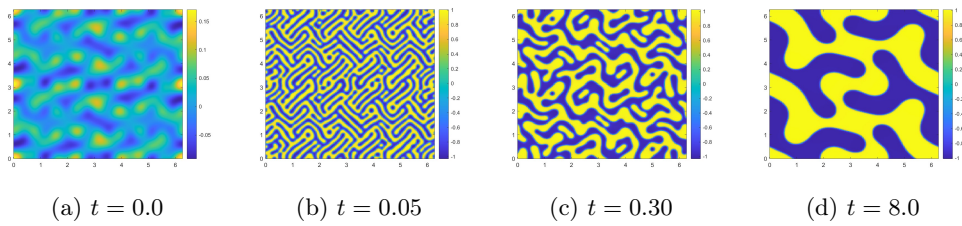


Figure 5: Numerical solution at different time stages with sinusoidal initial condition.

3.2.3 Example with random initial condition

One can also consider the Cahn-Hilliard equation (28) with random initial condition. This may impose more challenges to our computation since the randomness will remove the regularity of u_0 and make the numerical computation unstable. We will solve the equation (28) on $[0, 1]^2$ in this example. We let the periodic domain $\Omega = [0, 1]^2$. Then we choose $N_x = 128$, $h_x = 1/128$; $N_t = 100000$ with $h_t = 1/100000$. We choose a rather small time step size in this example in order to guarantee the accuracy of our numerical solution. For the initial condition, we choose u_0 as a random scalar field that takes i.i.d. values uniformly distributed on $[-0.05, 0.05]$. We evolve the PDHG dynamic with stepsize $\tau_u = \tau_p = 0.75$. We plot the numerical solutions at certain time stages in Figure 6. The reaction-diffusion system reaches the equilibrium state at $t = 1$. The residual plots of $\text{Res}(U)$ at time $t = 0.01$ and $t = 1$ are provided in Figure 7.

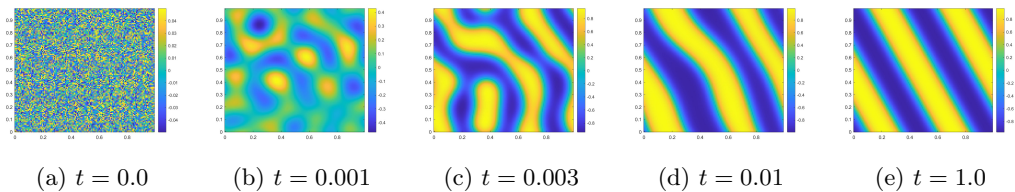
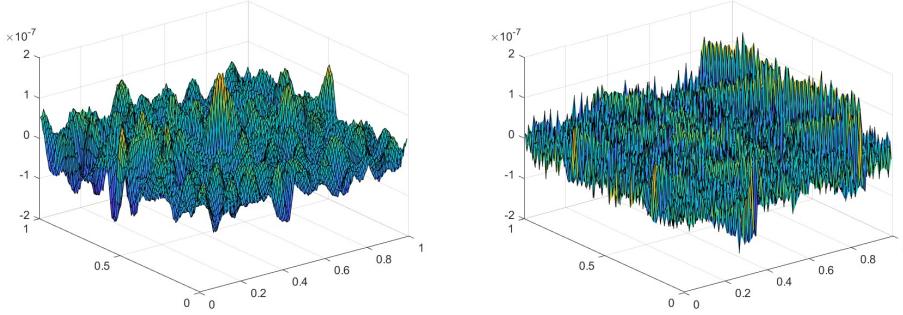


Figure 6: Numerical solution at different time stages with random initial condition.



(a) Plot of the residual $\text{Res}(U)$ at $t = 0.01$ (b) Plot of the residual $\text{Res}(U)$ at $t = 1.0$

Figure 7: Plots of residual at $t = 0.01$ and $t = 1.0$.

3.3 Higher-order Reaction-Diffusion Equations

In addition to the Allen-Cahn and Cahn-Hilliard equations, we test the method on the following 6th-order Cahn-Hilliard-type equation.

$$\frac{\partial u(x, t)}{\partial t} = \Delta(\epsilon^2 \Delta - W''(u) + \epsilon^2)(\epsilon^2 \Delta u - W'(u)) \text{ On } \Omega, \quad u(\cdot, 0) = u_0. \quad (33)$$

The above equation was first proposed in [21] which depicts the pore formation in functionalized polymers. This equation was later studied in the numerical examples of [12]. In this example, we set $\Omega = [0, 2\pi]^2$. We choose parameter $\epsilon = 0.18$. The potential function $W(u)$ is the same as defined in (26). Thus $W'(u) = u^3 - u$, $W''(u) = 3u^2$. Similar to Allen-Cahn or Cahn-Hilliard equations, equation (33) can also be treated as a flow that dissipates the energy $\mathcal{E}(u)$ with $a = \epsilon^2, b = 1$.

In our numerical implementation of the PDHG method, the functional $F(U)$ is now

$$F(U) = U - h_t \text{Lap}_{h_x}^P (\epsilon^2 \text{Lap}_{h_x}^P - \text{diag}(W''(U)) + \epsilon^2 I) (\epsilon^2 \text{Lap}_{h_x}^P U - W'(U)) - U^k.$$

Similar to Allen-Cahn and Cahn-Hilliard equations, we pick the preconditioner G as the square of the matrix in the dominating linear part of $F(U)$. However, if we directly keep the diagonal matrix $\text{diag}(W''(U))$, we will not be able to invert G efficiently by using FFT. Since the value of $W''(u)$ close to the equilibrium states ± 1 is approximately 2, we follow a similar idea in [12] to replace such matrix with $2I$. Thus, in this problem, we set $G = (I - h_t \epsilon^2 \text{Lap}_{h_x}^P (\epsilon^2 \text{Lap}_{h_x}^P - (2 - \epsilon^2)I) \text{Lap}_{h_x}^P)^2$ which can be inverted via FFT algorithm. Now the 3-line PDHG update is formulated as

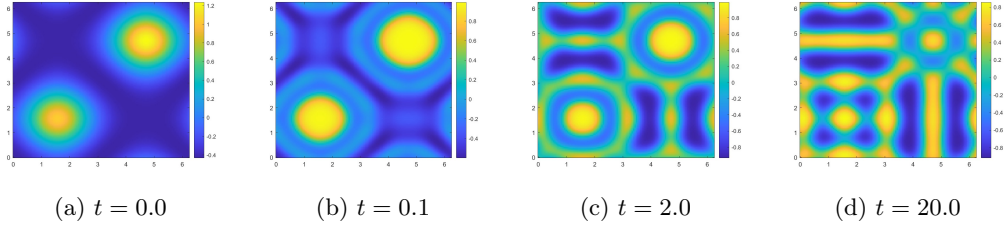
$$\begin{aligned} P_{n+1} &= P_n + \tau_p G^{-1} (U_n - h_t \text{Lap}_{h_x}^P \widetilde{\text{Lap}}_{h_x}(\epsilon, U_n)) (\epsilon^2 \text{Lap}_{h_x}^P U_n - W'(U_n)) - U^k; \\ \tilde{P}_{n+1} &= P_{n+1} + \omega (P_{n+1} - P_n); \\ U_{n+1} &= U_n - \tau_u (\tilde{P}_{n+1} - h_t (\epsilon^2 \text{Lap}_{h_x}^P - \text{diag}(W''(U))) \widetilde{\text{Lap}}_{h_x}(\epsilon, U_n) \text{Lap}_{h_x}^P \tilde{P}_{n+1} \\ &\quad + h_t (\epsilon^2 \text{Lap}_{h_x}^P U_n - W'(U_n)) \odot W'''(U_n) \odot \text{Lap}_{h_x}^P \tilde{P}_{n+1}). \end{aligned}$$

Here we denote $\widetilde{\text{Lap}}_{h_x}(\epsilon, U) = \epsilon^2 \text{Lap}_{h_x}^P - \text{diag}(W''(U)) + \epsilon^2 I$. If the size of U is N_x^2 , one can verify that all calculations among the PDHG iteration can be computed with complexity $O(N_x^2 \log(N_x))$ via the FFT method.

Similar to [12], we choose initial condition

$$u_0(x, y) = 2e^{\sin x + \sin y - 2} + 2.2e^{-\sin x - \sin y - 2} - 1. \quad (34)$$

In the numerical implementation, we solve the equation (33) from $t = 0$ to $t = 20$. We choose $N_x = 128$, $h_x = \pi/64$; $N_t = 20000$, $h_t = 1/1000$. We choose the PDHG stepsizes $\tau_u = \tau_p = 0.58$. We choose the threshold for terminating the iteration as $\delta = 0.5 \times 10^{-5}$. We present some of the results in Figure 8 and Figure 9.



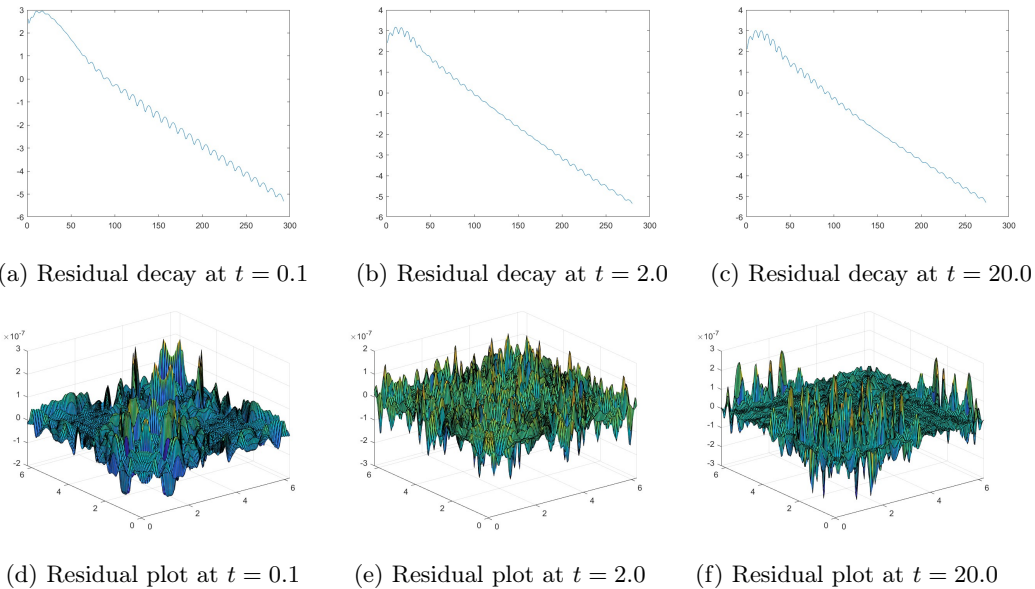
(a) $t = 0.0$

(b) $t = 0.1$

(c) $t = 2.0$

(d) $t = 20.0$

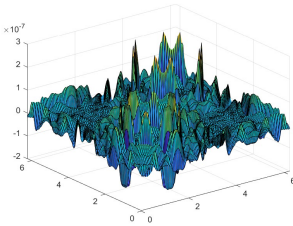
Figure 8: Numerical solution at different time stages with initial condition (34).



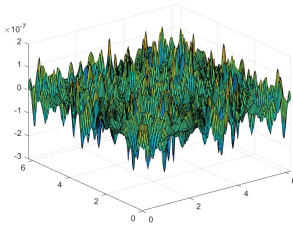
(a) Residual decay at $t = 0.1$

(b) Residual decay at $t = 2.0$

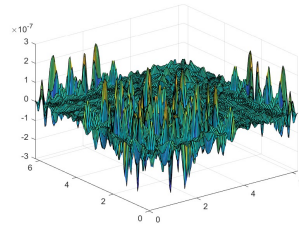
(c) Residual decay at $t = 20.0$



(d) Residual plot at $t = 0.1$



(e) Residual plot at $t = 2.0$



(f) Residual plot at $t = 20.0$

Figure 9: $\log -$ residual decay & plots of residual functional $\text{Res}(U^n)$ at different time stages $t = 0.1, 2.0, 20.0$.

3.4 Reaction-diffusion systems

We have already shown some reaction-diffusion equation examples in the previous sections. We now apply the method to compute reaction-diffusion systems.

3.4.1 Schnakenberg Model

The Schnakenberg model is first considered in [39] to model the limit-cycle behavior in a two-component chemical reaction system. In the discussion, we consider the following reaction-diffusion

PDE system defined on unit square $\Omega = [0, 1]^2$ where u, v represent the density concentration of two chemicals. Such a PDE system is also investigated in references [24, 49].

$$\frac{\partial u(x, y, t)}{\partial t} = D_1 \Delta u(x, y, t) + \kappa(a - u + u^2 v), \quad (35)$$

$$\frac{\partial v(x, y, t)}{\partial t} = D_2 \Delta v(x, y, t) + \kappa(b - u^2 v). \quad (36)$$

The initial condition of the system is

$$u(x, y, 0) = a + b + 10^{-3} * e^{-100((x-\frac{1}{3})^2 + (y-\frac{1}{2})^2)}, \quad v(x, y, 0) = \frac{b}{(a+b)^2}. \quad (37)$$

Here we set $\kappa = 100, a = 0.1305, b = 0.7695, D_1 = 0.05, D_2 = 1$. One can understand the initial data as exerting a tiny perturbation to the equilibrium solution $(a+b, \frac{b}{(a+b)^2})$ of the Schnakenberg system (35), (36). Such equilibrium state is unstable, the small perturbation will lead to the formation of certain dotted patterns in both components u and v .

We assume the Neumann boundary condition $\frac{\partial u}{\partial \mathbf{n}} = 0$ on $\partial\Omega$ where $\frac{\partial}{\partial \mathbf{n}}$ denotes the directional derivative w.r.t. the outer pointing normal direction \mathbf{n} .

Suppose we apply the one-step implicit scheme to solve this problem, recall the discrete Laplacian with Neumann BC introduced in (24), at the k -th time step, we consider

$$\begin{aligned} F_u(U, V) &= U - U^k - h_t(D_1 \text{Lap}_{h_x}^N U + \kappa(a\mathbf{1} - U + U^2 \odot V)); \\ F_v(U, V) &= V - V^k - h_t(D_2 \text{Lap}_{h_x}^N V + \kappa(b\mathbf{1} - U^2 \odot V)). \end{aligned}$$

At each time step, our purpose is to solve $F_u(U, V) = 0, F_v(U, V) = 0$ for updating U^k, V^k . By treating $\tilde{U} = (U, V) \in \mathbb{R}^{2N_x^2}$ as an entity; and by denoting $\tilde{F} : \mathbb{R}^{2N_x^2} \rightarrow \mathbb{R}^{2N_x^2}, \tilde{U} \mapsto (F_u(U, V)^\top, F_v(U, V)^\top)^\top$, the problem of solving $\tilde{F}(\tilde{U}) = 0$ reduces to the scenario of solving single $F(U) = 0$ discussed before. Hence, it is natural to introduce the dual variable $\tilde{P} = (P, Q) \in \mathbb{R}^{2N_x^2}$; The stiff Laplacian terms can be treated as dominating linear terms of both functions F, G , thus we set our preconditioner matrix $\tilde{G} = \begin{pmatrix} G_u & \\ & G_v \end{pmatrix}$ with $G_u = (I - h_t D_1 \text{Lap}_{h_x}^N)^2$ and $G_v = (I - h_t D_2 \text{Lap}_{h_x}^N)^2$. The corresponding PDHG iteration for solving $\tilde{F}(\tilde{U}) = 0$ is formulated as follows.

$$\begin{aligned} P_{n+1} &= P_n + \tau_p G_u^{-1}(U_n - U^k - h_t(D_1 \text{Lap}_{h_x}^N U_n + \kappa(a\mathbf{1} - U_n + U_n^2 \odot V_n))); \\ Q_{n+1} &= Q_n + \tau_p G_v^{-1}(V_n - V^k - h_t(D_2 \text{Lap}_{h_x}^N V_n + \kappa(b\mathbf{1} - U_n^2 \odot V_n))); \\ \tilde{P}_{n+1} &= P_{n+1} + \omega(P_{n+1} - P_n); \quad \tilde{Q}_{n+1} = Q_{n+1} + \omega(Q_{n+1} - Q_n); \\ U_{n+1} &= U_n - \tau_u(\tilde{P}_{n+1} - h_t(D_1 \text{Lap}_{h_x}^N \tilde{P}_{n+1} + \kappa(-\tilde{P}_{n+1} + 2U_n \odot V_n \odot (\tilde{P}_{n+1} - \tilde{Q}_{n+1}))); \\ V_{n+1} &= V_n - \tau_u(\tilde{Q}_{n+1} - h_t(D_2 \text{Lap}_{h_x}^N \tilde{Q}_{n+1} + \kappa(U_n^2 \odot (\tilde{P}_{n+1} - \tilde{Q}_{n+1}))). \end{aligned}$$

We recall that the Discrete Cosine Transform mentioned in Remark 1 can be used to compute matrix-vector multiplication involving $\text{Lap}_{h_x}^N$ as well as inverting the preconditioners G_u, G_v within $O(N_x^2 \log N_x)$ complexity. Thus every step of the above PDHG iterations can be computed efficiently.

In our numerical implementation, we solve this PDE system, on time interval $[0, 2]$. We choose $N_x = 128, h_x = 1/128$; and $N_t = 10000, h_t = 1/5000$. We then choose $\tau_u = \tau_p = 0.9$. We terminate the PDHG iteration when $\|\text{Res}(U_n)\|_2 + \|\text{Res}(V_n)\|_2 < \delta$, where we pick threshold $\delta = 10^{-7}$. Our numerical solutions are presented in the following Figure 10.

In this example, the performance of the PDHG method is stable and the method terminates in around 30 iterations for all 10000 time steps. We plot the loss as well as the residual term $\text{Res}(U)$ at different time stages in Figure 11.

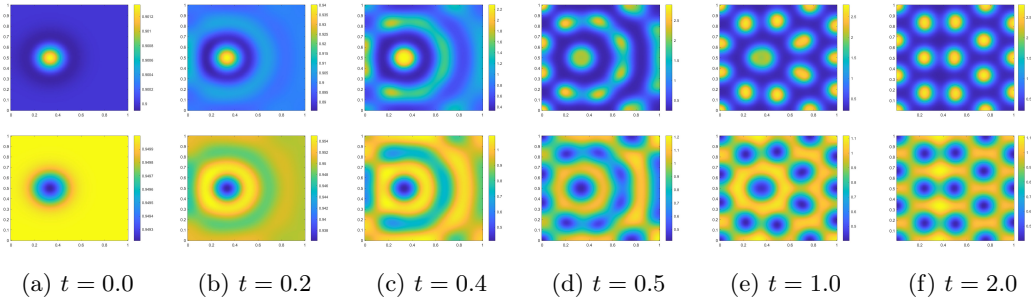


Figure 10: Numerical solution of u (upper row), and v (lower row) at different time stages with initial condition (37).

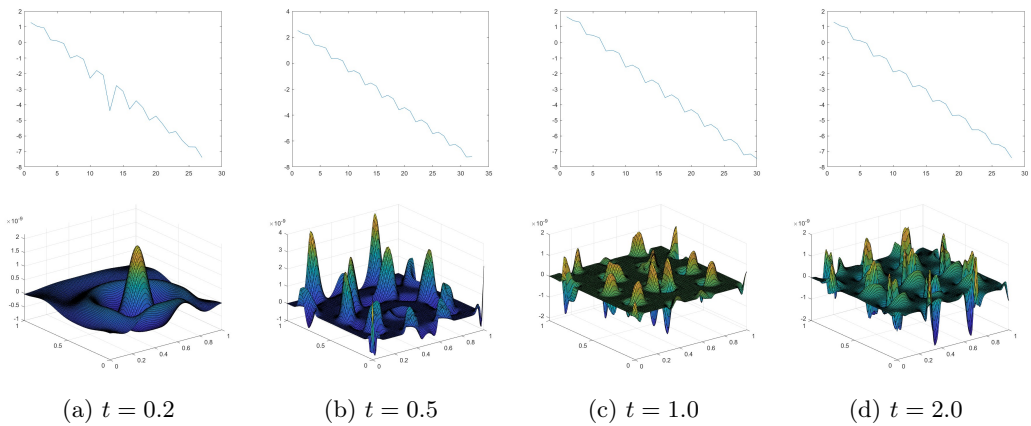


Figure 11: \log -residual decay of U & plots of residual $\text{Res}(U)$ at different time stages $t = 0.2, 0.5, 1.0, 2.0$.

We compare the computational speed of our PHDG method with the commonly used Newton-SOR method [43, 32]. We fix all the parameters the same for both methods, typically, we set the termination threshold for both methods to be $\delta = 10^{-7}$. We solve the PDE system on $[0, 1]$ with $N_t = 5000$ and $N_x = 128$. The time cost for the Newton-SOR method is 8122.81s, while the time cost for the PDHG method is 1121.81s.

3.4.2 Wolf-deer model

At last, let us consider an equation system describing the evolution of predator (wolves) and prey (deer) distributions in an ecology system [34, 20]. The PDE system is defined on the region $\Omega = [-L, L]^2$ and takes the following form,

$$\frac{\partial \rho_1}{\partial t} = D\Delta\rho_1 + \nabla \cdot (\rho_1 \nabla \mathcal{V}_1(\rho_1, \rho_2)) + A\rho_1(1 - \rho_1) - B \frac{\rho_1 \rho_2}{1 + \rho_1}; \quad (38)$$

$$\frac{\partial \rho_2}{\partial t} = D\Delta\rho_2 + \nabla \cdot (\rho_2 \nabla \mathcal{V}_2(\rho_1, \rho_2)) + B \frac{\rho_1 \rho_2}{1 + \rho_1} - C\rho_2. \quad (39)$$

Here we set $D = \frac{1}{2}$, $A = 5$, $B = 35$, $C = \frac{5}{2}$. We also define the interacting potentials $\mathcal{V}_1, \mathcal{V}_2$ as

$$\mathcal{V}_1(\rho_a, \rho_b)(\cdot) = V * \rho_1 - V * \rho_2; \quad \mathcal{V}_2(\rho_a, \rho_b)(\cdot) = V * \rho_1 + V * \rho_2.$$

Here the convolution is defined as $V * \rho(x, y) = \iint_{\Omega \times \Omega} V((x, y) - (x', y')) \rho(x', y') dx' dy'$ with potential $V(x, y) = \frac{x^2 + y^2}{2}$.

We choose Neumann boundary condition for both ρ_1 and ρ_2 , and set the initial condition

$$\rho_i(x, 0) = \frac{1}{\pi} \left(\frac{\pi}{2} + \arctan \left(\frac{R^2 - |X - \bar{\mu}_i|^2}{\epsilon} \right) \right), \quad i = 1, 2, \quad (40)$$

where $\bar{\mu}_1 = (\frac{3}{2}, \frac{3}{2})$, $\bar{\mu}_2 = (-\frac{3}{2}, -\frac{3}{2})$, $R = 1$ and $\epsilon = 0.1$.

In system (38), (39), ρ_1 represents the distribution of deer, and ρ_2 stands for the distribution of wolf. In addition to the diffusion and reaction terms affecting ρ_a, ρ_2 , the PDE system (38), (39) contain non-local drift terms $\nabla \mathcal{V}_1(\rho_a, \rho_2), \nabla \mathcal{V}_2(\rho_1, \rho_2)$ that depict the interactions among the individuals of wolves and deer: The deer are attracting each other to dodge wolves' predation, while the wolves are gathering together to chase the flock of deer.

Suppose we discretize each side of Ω into $N_x - 1$ equal subintervals, we denote the numerical solutions at the n -th time step as $\rho_1^n, \rho_2^n \in \mathbb{R}^{N_x^2}$. We consider the following implicit, central-difference scheme,

$$\begin{aligned} \frac{\rho_1^{n+1} - \rho_1^n}{h_t} - DLap_{h_x}^N \rho_1^{n+1} - D_x^\top (\overline{\rho_1^{n+1}}^x \odot D_x (K \rho_1^{n+1} - K \rho_2^{n+1})) \\ - D_y^\top (\overline{\rho_1^{n+1}}^y \odot D_y (K \rho_1^{n+1} - K \rho_2^{n+1})) + R_1(\rho_1^{n+1}, \rho_2^{n+1}) = 0; \\ \frac{\rho_2^{n+1} - \rho_2^n}{h_t} - DLap_{h_x}^N \rho_2^{n+1} - D_x^\top (\overline{\rho_2^{n+1}}^x \odot D_x (K \rho_1^{n+1} + K \rho_2^{n+1})) \\ - D_y^\top (\overline{\rho_2^{n+1}}^y \odot D_y (K \rho_1^{n+1} + K \rho_2^{n+1})) + R_2(\rho_1^{n+1}, \rho_2^{n+1}) = 0. \end{aligned}$$

Here D_x is an $(N_x + 1)N_x \times N_x^2$ matrix which can be treated as the discrete gradient with respect to x , i.e., for any $u \in \mathbb{R}^{N_x^2}$, $(D_x u)_{(i, j + \frac{1}{2})}$ equals $\frac{u_{i, j + \frac{1}{2}} - u_{i, j}}{h_x}$ for $1 \leq i \leq N_x$, $1 \leq j \leq N_x - 1$; for $j = 0, N_x$, we define $(D_x u)_{(i, \frac{1}{2})} = \frac{u_{i, 2} - u_{i, 1}}{h_x}$ and $(D_x u)_{(i, N_x + \frac{1}{2})} = \frac{u_{i, N_x} - u_{i, N_x - 1}}{h_x}$. D_y can also be defined in a similar way.

The notation $\overline{\rho_1^{n+1}}^x \in \mathbb{R}^{(N_x + 1)N_x}$ denotes the average value of ρ_1^{n+1} at midpoints, i.e., $(\overline{\rho_1^{n+1}}^x)_{(i, j + \frac{1}{2})} = \frac{\rho_{1, (i, j)}^{n+1} + \rho_{1, (i, j + 1)}^{n+1}}{2}$ for $1 \leq i \leq N_x$, $1 \leq j \leq N_x - 1$; for $j = 0, N_x$, we define $\overline{\rho_1^{n+1}}^x_{(i, \frac{1}{2})} = \rho_{1, i, 1}^{n+1}$ and $\overline{\rho_1^{n+1}}^x_{(i, N_x + \frac{1}{2})} = \rho_{1, i, N_x}^{n+1}$. $\overline{\rho_1^{n+1}}^y, \overline{\rho_2^{n+1}}^x, \overline{\rho_2^{n+1}}^y$ can be defined in the similar way.

K is an $N_x^2 \times N_x^2$ matrix used for approximating the convolution $V * \rho_1, V * \rho_2$, to precisely, for any $u \in \mathbb{R}^{N_x^2}$ defined on the mesh grid of Ω , Ku is defined as

$$(Ku)_{(i, j)} = \sum_{1 \leq k, l \leq N_x} h_x^2 V(v_{i, j} - v_{k, l}) u_{k, l} = \sum_{1 \leq k, l \leq N_x} \frac{h_x^4}{2} ((i - k)^2 + (j - l)^2) u_{k, l}.$$

The above discrete convolution can be reduced to Toeplitz matrix-vector multiplication computation, which can be efficiently computed by FFT algorithm [45].

Furthermore, for $\rho_1, \rho_2 \in \mathbb{R}^{N_x^2}$ the reaction terms are defined as $R_1(\rho_1, \rho_2) = A \rho_1 \odot (1 - \rho_1) - B \frac{\rho_1}{1 + \rho_1} \odot \rho_2$; $R_2(\rho_1, \rho_2) = B \frac{\rho_1}{1 + \rho_1} \odot \rho_2 - C \rho_2$.

Given the above discrete scheme of the PDE system, similar to the discussion made in the Schnakenberg model, our purpose is to solve two nonlinear equations $F_{\rho_1}(\rho_1, \rho_2) = 0$, $F_{\rho_2}(\rho_1, \rho_2) = 0$ at each time step n . We apply two dual variables P, Q , and compose the corresponding PDHG dynamic for solving the two equations. According to the previous discussion, one can verify that each PDHG step can be computed within $O(N_x^2 \log N_x)$ complexity. This guarantees the efficiency of the computation. To keep the discussion concise, we omit the exact formulas for the PDHG dynamic here.

In our implementation, we set $L = 3$. We solve the equation system (38), (39) on time interval $[0, 1]$. We set $N_x = 128$, $h_x = 3/64$. We practice the method of adaptive time stepsize h_t in

this example. We set both our initial time stepsize h_t and maximum stepsize h_t^0 equal to $1/500$ with shrinkage/enlarge coefficient η as 0.75 . The thresholding iteration numbers of shrinking and enlarging h_t are set to be $N^* = 100$, and $N_* = 20$. For the PDHG iteration, we set stepsize $\tau_u = \tau_p = 0.95$, and pick the threshold $\delta = 5 \times 10^{-6}$. We present the numerical results in Figure 12.

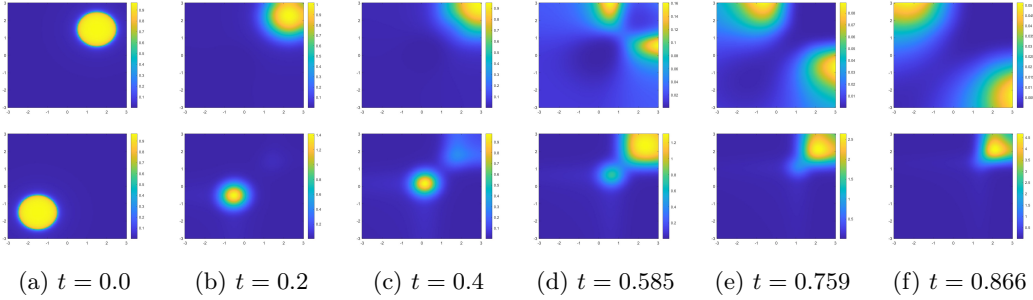


Figure 12: Numerical solution of ρ_1 (upper row), and ρ_2 (lower row) at different time stages with initial condition (40).

The linear convergence of the residual term $\|\text{Res}(U)\|_2$ is reflected from the residual decay plots in Figure 13.

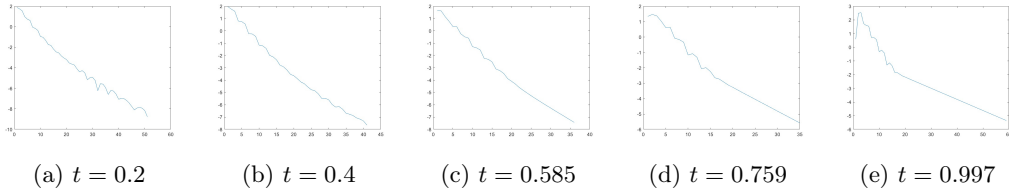
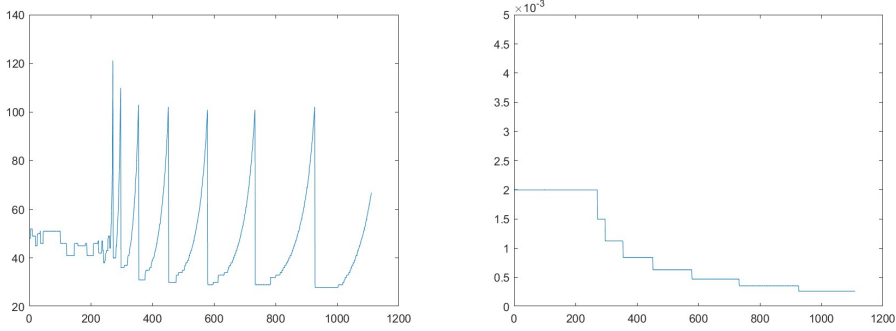


Figure 13: $\log_{10} \text{Res}(U_n)$ vs PDHG iteration number n at different time stages.

The changes in PDHG iterations at each time stepsize as well as the changes in time step size h_t are demonstrated via Figure 14. As reflected from the plots, in this example, we are gradually shrinking the time stepsize h_t as the accumulated time increases to guarantee the computational efficiency of the PDHG method. Our method takes a total of 1106 time to complete the computation. the algorithm experiences 7 stepsize shrinkage among our computation. The initial h_t is set as 0.002 while when we finish the computation, $h_t = 0.00027$.

4 Conclusion and Future Study

This research proposes an iterative method as a convenient but efficient gadget for solving the implicit (or semi-implicit) numerical schemes arising in time-evolution PDEs, especially the reaction-diffusion type equations. Our method recasts the nonlinear equation from the discrete numerical scheme at each time step as a min-max saddle point problem and applies the Primal-Dual Hybrid Gradient method. The algorithm can flexibly fit into various numerical schemes, such as semi-implicit and fully implicit schemes, etc. Furthermore, the method is easy to implement since it gets rid of the computation of large-scale linear systems involving Jacobian matrices, which are usually required by Newton's methods. The performance of our method on accuracy and efficiency is satisfying and is comparable to the commonly used Newton-type methods. This has been verified by the numerical examples presented in this paper.



(a) Number of PDHG iterations at each time step. (b) Time stepsize h_t used at each time step, initial $h_t = 0.002$, terminal $h_t \approx 0.00027$.

Figure 14: Changes in number of PDHG iterations & time stepsize h_t .

There are three main future research directions of our work. We summarize them below:

- Conduct theoretical analysis on the convergence of the PDHG method for nonlinear RD equations. We are interested in the necessary condition on h_t, h_x, τ_u, τ_p that can guarantee the convergence of our method for certain types of RD equations.
- Generalize the method to nonlinear time-evolution equations, especially the advection-reaction-diffusion dynamics from GENERIC (General Equation for Non-Equilibrium Reversible-Irreversible Coupling) [16, 23, 35].
- Apply the method to high-dimensional time-evolution PDEs by leveraging deep learning techniques and PDHG algorithms.

Acknowledgement: S. Liu and S. Osher's work was partly supported by AFOSR MURI FP 9550-18-1-502 and ONR grants: N00014-20-1-2093 and N00014-20-1-2787. W. Li's work was supported by AFOSR MURI FP 9550-18-1-502, AFOSR YIP award No. FA9550-23-1-0087, and NSF RTG: 2038080.

A Proof of Theorem 1

Proof. It is not hard to verify that the dynamic (14), (15) and (17) can be formulated as

$$\begin{bmatrix} U_{n+1} \\ P_{n+1} \end{bmatrix} = \begin{bmatrix} I - 2\tau_u\tau_p A^\top A & -\tau_u A^\top \\ \tau_p A & I \end{bmatrix} \begin{bmatrix} U_n \\ P_n \end{bmatrix} + \begin{bmatrix} 2\tau_u\tau_p A^\top b \\ -\tau_p b \end{bmatrix}, \quad n \geq 0 \quad (41)$$

This equation admits a unique fixed point $X_* = (U_*, P_*) = (A^{-1}b, 0)$. We denote $X_n = [U_n^\top, P_n^\top]^\top$ and the above recurrence equation as $X_{n+1} = MX_n + y$ (or equivalently, $X_{n+1} - X_* = M(X_n - X_*)$) for shorthand. Suppose A has spectral decomposition $A = Q\Lambda Q^\top$, then M is decomposed as

$$M = \begin{bmatrix} Q & \\ & Q \end{bmatrix} \begin{bmatrix} I - 2\tau_u\tau_p\Lambda^2 & -\tau_u\Lambda \\ \tau_p\Lambda & I \end{bmatrix} \begin{bmatrix} Q & \\ & Q \end{bmatrix}^\top$$

We denote N_A as the size of A . The middle matrix is composed of four $N_A \times N_A$ diagonal matrices, by rearranging the rows and columns of it, one can show that it is orthogonally equivalent to the block diagonal matrix $\Sigma = \text{diag}(D_1, D_2, \dots, D_{N_A})$ where each $D_k = \begin{bmatrix} 1 - 2\tau_u\tau_p\lambda_k^2 & -\tau_u\lambda_k \\ \tau_p\lambda_k & 1 \end{bmatrix}$. We

now analyze the spectral radius $\rho(M)$ of M , which equals $\rho(\Sigma) = \max_{1 \leq k \leq N_A} \{\rho(D_k)\}$. We can calculate

$$\rho(D_k) = \max\{|\tau_u \tau_p \lambda_k^2 - 1 \pm \sqrt{(\tau_u \tau_p)^2 \lambda_k^4 - \tau_u \tau_p \lambda_k^2}|\}.$$

We denote $f(t) = \max\{|t - 1 + \sqrt{t^2 - t}|, |t - 1 - \sqrt{t^2 - t}|\}$, with $t > 0$. One can directly compute $f(t) = \begin{cases} \sqrt{1-t} & 0 < t \leq 1 \\ t - 1 + \sqrt{t^2 - t} & t > 1 \end{cases}$. Thus we have $\rho(M) = \max_{1 \leq k \leq N_A} \{f(\tau_u \tau_p \lambda_k^2)\}$. Then f is decreasing on $[0, 1]$ and increasing on $[1, \infty)$ with $f(0) = f(\frac{4}{3}) = 1$. We know that the convergence of (41) is guaranteed if and only if $\rho(M) < 1$. This is equivalent to $\tau_u \tau_p \lambda_{\max}^2 \leq \frac{4}{3}$, which yields $\tau_u \tau_p \leq \frac{4}{3\lambda_{\max}^2}$.

Furthermore, $\rho(M)$ is the convergence rate of the dynamic, i.e.,

$$\|X_n - X_*\|_2 \leq \rho(M)^n \|X_0 - X_*\|_2$$

To evaluate the optimal convergence rate, we compute the minimum value of $\rho(M)$ w.r.t. stepsizes τ_u, τ_p . Suppose we require $\tau_u \tau_p \leq \frac{4}{3\lambda_{\max}^2}$ to guarantee convergence, if we denote $\eta = \tau_u \tau_p \lambda_{\max}^2$, then $\rho(M) = \max_{1 \leq k \leq N_A} \left\{ f\left(\frac{\lambda_k^2}{\lambda_{\max}^2} \eta\right) \right\}$ for $\eta \in (0, \frac{4}{3})$. The minimum value of $\rho(M)$ will be attained at a unique $\eta = \eta_* \in [1, \frac{4}{3})$ such that $f(\frac{\eta}{\kappa^2}) = f(\eta)$. I.e., η_* is the solution of

$$\sqrt{1 - \frac{\eta}{\kappa^2}} = \eta - 1 + \sqrt{\eta^2 - \eta}, \quad \text{on } [1, \frac{4}{3}). \quad (42)$$

Thus, the optimal convergence rate $\gamma_* = \sqrt{1 - \frac{\eta_*}{\kappa^2}}$ and it is achieved when τ_u, τ_p satisfy $\tau_u \tau_p = \frac{\eta_*}{\lambda_{\max}^2}$. \square

Remark 2. The equation (42) can be reduced to a quadratic equation. And it admits a unique solution η_* on $[1, \frac{4}{3})$, η_* takes the following form

$$\eta_* = \frac{2\kappa^2}{\left(\frac{3}{4}\kappa^2 + \frac{3}{2} - \frac{1}{4\kappa^2}\right) + \frac{\kappa-1}{2\kappa} \sqrt{(\kappa-1)(3\kappa+1)} \sqrt{\frac{3}{4}\kappa^2 + \frac{3}{2} + 2\kappa - \frac{1}{4\kappa^2}}}.$$

The optimal convergence rate $\gamma_* = \sqrt{1 - \frac{\eta_*}{\kappa^2}}$ takes the explicit form

$$\gamma_* = \left(1 - \frac{2}{\left(\frac{3}{4}\kappa^2 + \frac{3}{2} - \frac{1}{4\kappa^2}\right) + \frac{\kappa-1}{2\kappa} \sqrt{(\kappa-1)(3\kappa+1)} \sqrt{\frac{3}{4}\kappa^2 + \frac{3}{2} + 2\kappa - \frac{1}{4\kappa^2}}} \right)^{\frac{1}{2}}.$$

Notice that $\gamma_* \approx (1 - \frac{4}{3\kappa^2})^{\frac{1}{2}}$ when the conditional number κ is very large; and γ_* will approach 0 as condition number κ approaches 1. This motivates the preconditioning technique of our method.

References

- [1] Samuel M Allen and John W Cahn. A microscopic theory for antiphase boundary motion and its application to antiphase domain coarsening. *Acta metallurgica*, 27(6):1085–1095, 1979.
- [2] Kendall Atkinson. *An introduction to numerical analysis*. John Wiley & sons, 1991.
- [3] Gang Bao, Xiaojing Ye, Yaohua Zang, and Haomin Zhou. Numerical solution of inverse problems by weak adversarial networks. *Inverse Problems*, 36(11):115003, 2020.
- [4] John W Cahn. On spinodal decomposition. *Acta metallurgica*, 9(9):795–801, 1961.
- [5] José A. Carrillo, Katy Craig, Li Wang, and Chaozhen Wei. Primal Dual Methods for Wasserstein Gradient Flows. *Foundations of Computational Mathematics*, 22(2):389–443, 2022.

- [6] Jose A. Carrillo, Li Wang, and Chaozhen Wei. Structure preserving primal dual methods for gradient flows with nonlinear mobility transport distances, 2023.
- [7] Hector D Ceniceros and Carlos J García-Cervera. A new approach for the numerical solution of diffusion equations with variable and degenerate mobility. *Journal of Computational Physics*, 246:1–10, 2013.
- [8] Antonin Chambolle and Thomas Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of mathematical imaging and vision*, 40:120–145, 2011.
- [9] Long Chen and Jingrong Wei. Accelerated gradient and skew-symmetric splitting methods for a class of monotone operator equations. *arXiv preprint arXiv:2303.09009*, 2023.
- [10] Long Chen and Jingrong Wei. Transformed primal-dual methods for nonlinear saddle point systems. *Journal of Numerical Mathematics*, 2023.
- [11] Ching-Shan Chou, Yong-Tao Zhang, Rui Zhao, and Qing Nie. Numerical methods for stiff reaction-diffusion systems. *Discrete and continuous dynamical systems, series B*, 7(3):515, 2007.
- [12] Andrew Christlieb, Jaylan Jones, Keith Promislow, Brian Wetton, and Mark Willoughby. High accuracy solutions to energy gradient flows from material science models. *Journal of computational physics*, 257:193–215, 2014.
- [13] Jon M Church, Zhenlin Guo, Peter K Jimack, Anotida Madzvamuse, Keith Promislow, Brian Wetton, Steven M Wise, and Fengwei Yang. High accuracy benchmark problems for Allen-Cahn and Cahn-Hilliard dynamics. *Communications in computational physics*, 26(4), 2019.
- [14] Christian Clason and Tuomo Valkonen. Primal-dual extragradient methods for nonlinear nonsmooth PDE-constrained optimization. *SIAM Journal on Optimization*, 27(3):1314–1339, 2017.
- [15] Richard Courant, Kurt Friedrichs, and Hans Lewy. Über die partiellen differenzgleichungen der mathematischen physik. *Mathematische annalen*, 100(1):32–74, 1928.
- [16] Manh Hong Duong and Michela Ottobre. Non-reversible processes: GENERIC, hypocoercivity and fluctuations. *Nonlinearity*, 36(3):1617, 2023.
- [17] David J Eyre. An unconditionally stable one-step scheme for gradient systems. *Unpublished article*, 6, 1998.
- [18] Fariba Fahroo and Kazufumi Ito. Optimum damping design for an abstract wave equation. *Kybernetika*, 32(6):557–574, 1996.
- [19] Guosheng Fu, Stanley Osher, and Wuchen Li. High order spatial discretization for variational time implicit schemes: Wasserstein gradient flows and reaction-diffusion systems. *arXiv preprint arXiv:2303.08950*, 2023.
- [20] Thomas Gallouët, Maxime Laborde, and Leonard Monsaingeon. An unbalanced optimal transport splitting scheme for general advection-reaction-diffusion problems. *ESAIM: Control, Optimisation and Calculus of Variations*, 25:8, 2019.
- [21] Nir Gavish, Jaylan Jones, Zhengfu Xu, Andrew Christlieb, and Keith Promislow. Variational models of network formation and ion transport: Applications to perfluorosulfonate ionomer membranes. *Polymers*, 4(1):630–655, 2012.
- [22] Gene H Golub and Charles F Van Loan. *Matrix computations*. JHU press, 2013.
- [23] Miroslav Grmela and Hans Christian Öttinger. Dynamics and thermodynamics of complex fluids. i. development of a general formalism. *Physical Review E*, 56(6):6620, 1997.
- [24] Willem H Hundsdorfer, Jan G Verwer, and WH Hundsdorfer. *Numerical solution of time-dependent advection-diffusion-reaction equations*, volume 33. Springer, 2003.
- [25] Matt Jacobs, Flavien Léger, Wuchen Li, and Stanley Osher. Solving large-scale optimization problems with a convergence rate independent of grid size. *SIAM Journal on Numerical Analysis*, 57(3):1100–1123, 2019.

- [26] Andrea M Jokisaari, PW Voorhees, Jonathan E Guyer, James Warren, and OG Heinonen. Benchmark problems for numerical implementations of phase field models. *Computational Materials Science*, 126:139–151, 2017.
- [27] Yibao Li, Hyun Geun Lee, Darae Jeong, and Junseok Kim. An unconditionally stable hybrid numerical method for solving the Allen-Cahn equation. *Computers & Mathematics with Applications*, 60(6):1591–1606, 2010.
- [28] Chun Liu, Cheng Wang, and Yiwei Wang. A structure-preserving, operator splitting scheme for reaction-diffusion equations with detailed balance. *Journal of Computational Physics*, 436:110253, 2021.
- [29] Chun Liu, Cheng Wang, and Yiwei Wang. A second-order accurate, operator splitting scheme for reaction-diffusion systems in an energetic variational formulation. *SIAM Journal on Scientific Computing*, 44(4):A2276–A2301, 2022.
- [30] Chun Liu, Cheng Wang, Yiwei Wang, and Steven M Wise. Convergence analysis of the variational operator splitting scheme for a reaction-diffusion system with detailed balance. *SIAM Journal on Numerical Analysis*, 60(2):781–803, 2022.
- [31] Siting Liu, Stanley Osher, Wuchen Li, and Chi-Wang Shu. A primal-dual approach for solving conservation laws with implicit in time approximations. *Journal of Computational Physics*, 472, 2023.
- [32] Barry Merriman, James K Bence, and Stanley J Osher. Motion of multiple junctions: A level set approach. *Journal of computational physics*, 112(2):334–363, 1994.
- [33] Barry Merriman, James Kenyard Bence, and Stanley Osher. *Diffusion generated motion by mean curvature*. Department of Mathematics, University of California, Los Angeles, 1992.
- [34] James D Murray. *Mathematical biology II: Spatial models and biomedical applications*, volume 3. Springer New York, 2001.
- [35] Hans Christian Öttinger and Miroslav Grmela. Dynamics and thermodynamics of complex fluids. ii. illustrations of a general formalism. *Physical Review E*, 56(6):6633, 1997.
- [36] Lorenzo Pareschi and Giovanni Russo. Implicit-explicit runge-kutta schemes for stiff systems. *Recent trends in numerical analysis*, 3:269, 2000.
- [37] John E Pearson. Complex patterns in a simple system. *Science*, 261(5118):189–192, 1993.
- [38] Jacob Rubinstein, Peter Sternberg, and Joseph B. Keller. Fast Reaction, Slow Diffusion, and Curve Shortening. *SIAM Journal on Applied Mathematics*, 49(1):116–133, 1989.
- [39] J Schnakenberg. Simple chemical reaction systems with limit cycle behaviour. *Journal of theoretical biology*, 81(3):389–400, 1979.
- [40] Jie Shen, Jie Xu, and Jiang Yang. The scalar auxiliary variable (SAV) approach for gradient flows. *Journal of Computational Physics*, 353:407–416, 2018.
- [41] Jie Shen, Jie Xu, and Jiang Yang. A new class of efficient and robust energy stable schemes for gradient flows. *SIAM Review*, 61(3):474–506, 2019.
- [42] Jie Shen and Xiaofeng Yang. Numerical approximations of Allen-Cahn and Cahn-Hilliard Equations. *Discrete Contin. Dyn. Syst*, 28(4):1669–1691, 2010.
- [43] Andrew H Sherman. On newton-iterative methods for the solution of systems of nonlinear equations. *SIAM Journal on Numerical Analysis*, 15(4):755–771, 1978.
- [44] Martin B. Short, P. Jeffrey Brantingham, Andrea L. Bertozzi, and George E. Tita. Dissipation and displacement of hotspots in reaction-diffusion models of crime. *Proceedings of the National Academy of Sciences*, 107(9):3961–3965, 2010.
- [45] Gilbert Strang. A proposal for Toeplitz matrix calculations. *Studies in Applied Mathematics*, 74(2):171–176, 1986.

- [46] Gilbert Strang. The Discrete Cosine Transform. *SIAM Review*, 41(1):135–147, 1999.
- [47] Tuomo Valkonen. A primal-dual hybrid gradient method for nonlinear operators with applications to MRI. *Inverse Problems*, 30(5):055012, 2014.
- [48] Yaohua Zang, Gang Bao, Xiaojing Ye, and Haomin Zhou. Weak adversarial networks for high-dimensional partial differential equations. *Journal of Computational Physics*, 411:109409, 2020.
- [49] Jianfeng Zhu, Yong-Tao Zhang, Stuart A Newman, and Mark Alber. Application of discontinuous Galerkin methods for reaction-diffusion systems in developmental biology. *Journal of Scientific Computing*, 40:391–418, 2009.
- [50] Mingqiang Zhu and Tony Chan. An efficient primal-dual hybrid gradient algorithm for total variation image restoration. *UCLA CAM Report*, 34:8–34, 2008.