Commun. Comput. Phys.
doi: 10.4208/cicp.scpde14.45s

Vol. **19**, No. 5, pp. 1542-1563
May 2016

Parameter-Free Time Adaptivity Based on Energy Evolution for the Cahn-Hilliard Equation

Fuesheng Luo[1], Tao Tang[2,3,*] and Hehu Xie[4]

[1] *The Third Institute of Oceanography, SOA, Xiamen 361005, China.*
[2] *Department of Mathematics, Southern University of Science and Technology of China, Shenzhen, Guangdong 518055, China.*
[3] *Department of Mathematics, Hong Kong Baptist University, Kowloon Tong, Hong Kong.*
[4] *LSEC, ICMSEC, NCMIS, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, China.*

Received 1 May 2015; Accepted (in revised version) 29 December 2015

Abstract. It is known that large time-stepping method are useful for simulating phase field models. In this work, an adaptive time-stepping strategy is proposed based on numerical energy stability and equi-distribution principle. The main idea is to use the energy variation as an indicator to update the time step, so that the resulting algorithm is free of user-defined parameters, which is different from several existing approaches. Some numerical experiments are presented to illustrate the effectiveness of the algorithms.

AMS subject classifications: 65N10, 65N15, 35J25, 35Q99, 35R99, 65M12, 65M70

Key words: Adaptive time-stepping method, Cahn-Hilliard equation, Crank-Nicolson scheme, convex splitting method.

1 Introduction

This paper is concerned with the numerical method for the Cahn-Hilliard equation

$$\frac{\partial u}{\partial t}+\Delta(u-u^3+\kappa\Delta u)=0, \quad (x,t)\in\Omega\times(0,T], \tag{1.1a}$$

$$u(x,0)=u_0(x), \tag{1.1b}$$

where $\Omega=(0,L_1)\times(0,L_2)$ is a simple domain for the sake of simplicity, κ is a positive constant parameter and $u_0(x)$ is the initial data. Also for simplicity, the periodic boundary

*Corresponding author. *Email addresses:* luofusheng@tio.org.cn (F. Luo), tangt@sustc.edu.cn (T. Tang), hhxie@lsec.cc.ac.cn (H. Xie)

condition is imposed. The Cahn-Hilliard equation is first introduced in [1] to describe a continuum model for two mixture components separation and coarsening phenomena. In the model, u represents the concentration of one in the two components and the small parameter κ relates to the interfacial width.

There have been a large number of research work dealing with the Cahn-Hilliard problem. On the theoretical side, Elliot and Zheng [9] investigate the existence and uniqueness of the solution. On the numerical side, the finite element method [6–8], the finite difference method [14,17,19,25,27], the spectral method [11,16,21,23,30], the finite volume method [10] and the discontinuous Galerkin method [26] have been proposed and applied to solve the Cahn-Hilliard equation numerically. In particular, we mention here the conservation difference scheme with unconditional stability, see, e.g., [14,29].

As for the time discretization aspect, it is noted that the numerical simulation of Cahn-Hillard model needs a long time to reach the steady state. Therefore, the schemes with high stability that allows large time-stepping are very useful to reduce the total computation time. In [17], a semi-implicit difference scheme with an extra term is proposed which allows large time-stepping for long-term simulation. In [15,29], a time adaptivity strategy based on the energy evolution are presented. For more information, please refer to [20].

In this paper, we present a new type of adaptive time-stepping scheme which only requires to input the minimum time step, the maximum time step and the tolerance of the energy decay rate. An energy identity, which connects the energy variation and the space gradient of the solution, is used to predict the time-step. Using the energy identity for the adaptive time-stepping method can avoid the difficulty of choosing artificial parameters needed by some other existing methods. With the predicted time steps, we aim at equi-distributing the energy curve in each time step. By setting the minimal time step and by adjusting the quantity of the energy decay rate, the accuracy of the numerical solution is pre-determined. Moreover, we consider two kinds of difference schemes to demonstrate that they can be integrated into the adaptive time-stepping framework.

The outline of this paper goes as follows. In Section 2, we briefly review the finite difference scheme proposed in [29] as well as some of its important properties. Based on these results, Section 3 is devoted to deriving an adaptive time-stepping algorithm. In Section 4, the time-stepping method is applied to the linear scheme based on the convex splitting and a nonlinear scheme based on the Crank-Nicolson scheme. Some numerical experiments are carried out to show the effectiveness of the algorithm in Section 5. Some concluding remarks will be given in the final section.

2 Model and discretization

The Cahn-Hilliard equation is the gradient flow of the energy functional

$$E(u)=\frac{\kappa}{2}\|\nabla u\|^2+\frac{1}{4}\|u^2-1\|^2, \tag{2.1}$$

where and hereafter $\|\cdot\|$ denotes the L^2-norm. It is shown in [29] that the solution of the Cahn-Hilliard equation (1.1) satisfies the energy identity

$$\frac{d}{dt}E(u)+\|\nabla\mu\|^2=0, \tag{2.2}$$

where $\mu=u-u^3+\kappa\Delta u$.

A second order finite difference scheme proposed by Zhang and Qiao [29] is then introduced to solve the two-dimensional Cahn-Hilliard equation. Construct a partition for the domain Ω uniformly with $\Delta x=L_1/N_x$ and $\Delta y=L_2/N_y$. The mesh grid is $(x_i,y_j):=(i\Delta x,j\Delta y),0\le i\le N_x,0\le j\le N_y$. Denote the time step as Δt and we use the central discrete difference operators ∇_h and Δ_h

$$\nabla_h f_{j+\frac{1}{2},k+\frac{1}{2}}=\begin{pmatrix}\frac{f_{j+1,k}-f_{j,k}}{\Delta x}\\ \frac{f_{j,k+1}-f_{j,k}}{\Delta y}\end{pmatrix}, \tag{2.3}$$

$$\Delta_h f_{j,k}=\frac{f_{j+1,k}-2f_{j,k}+f_{j-1,k}}{(\Delta x)^2}+\frac{f_{j,k+1}-2f_{j,k}+f_{j,k-1}}{(\Delta y)^2}. \tag{2.4}$$

The finite difference scheme for problem (1.1) reads:

$$\frac{U_{j,k}^{n+1}-U_{j,k}^n}{\Delta t}=-\Delta_h\mu_{j,k}^{n+\frac{1}{2}},\quad 1\le j\le N_x,1\le k\le N_y, \tag{2.5a}$$

where

$$\mu_{j,k}^{n+\frac{1}{2}}=\frac{U_{j,k}^{n+1}+U_{j,k}^n}{2}-\frac{U_{j,k}^{n+1}+U_{j,k}^n}{2}\frac{(U_{j,k}^{n+1})^2+(U_{j,k}^n)^2}{2}+\kappa\Delta_h\frac{U_{j,k}^{n+1}+U_{j,k}^n}{2}. \tag{2.5b}$$

In this paper, we define the discrete L^2-norm as

$$\|f\|_h^2=\sum_{j=1}^{N_x}\sum_{k=1}^{N_y}\Delta x\Delta y f_{j,k}^2.$$

Then the discrete energy can be defined as

$$E_h(U)=\frac{\kappa}{2}\|\nabla_h U\|_h^2+\frac{1}{4}\|U^2-1\|_h. \tag{2.6}$$

For the scheme (2.5), we have a discrete energy identity which leads to the unconditional energy stability.

Lemma 2.1 ([29]). *Given any time step $\Delta t>0$, the unconditional energy stability holds:*

$$E_h(U^{n+1})\le E_h(U^n), \tag{2.7}$$

where U^n is defined by (2.5) *and E_h is defined by* (2.6). *Moreover, the following discrete energy identity holds*

$$\frac{E_h(U^{n+1})-E_h(U^n)}{\Delta t}+\|\nabla_h\mu^{n+\frac{1}{2}}\|_h^2=0. \tag{2.8}$$

Define the error by $e^n = U^n - u^n$ with the pointwise definition $e^n_{j,k} = U^n_{j,k} - u^n_{j,k}$. It can be demonstrated that the truncation error of the scheme (2.5) satisfies

$$\|e^n\|_h = \mathcal{O}\big((\Delta t)^2 + (\Delta x)^2 + (\Delta y)^2\big). \tag{2.9}$$

3 Adaptive time-stepping based on the energy variation

The purpose of the adaptive time-stepping method is to vary the time-step sizes. For example, when the phase change is drastic, the small time step is adopted; when it slows down, the large time step is used. With some underlying physical quantity change during the phase evolution, such mechanism can be provided.

Between the spatial adaptive method and the adaptive time-stepping method, there is something in common. The spatial adaptive method always aims at equi-distribution quantities, such as error, arch-length or mass, among the elements. The adaptive time-stepping method aims at equi-distributing similar physical quantity (denoted as M) in the time direction, like the energy change or the L^2 norm of the phase change from initial state to the steady state. If we divide M into N parts, and denote $\delta M = M/N$, we intend to make the quantity change for M in each step to be close to δM.

However, several things are different:

- In spatial adaptive methods, the global error at any iteration step is computed so that the relative error distribution among the elements will be used to decide the refined or coarsening elements. However, in the adaptive time-stepping method, the total change of the physical quantity M is unknown until the last time step.
- We need to compute δM by numerical method as precise as possible in the adaptive time-stepping method rather than using a relative error as used in the spatial adaptive method.
- In spatial adaptive method, the global mesh can be redistributed according to the error distribution, while in adaptive time-stepping methods, redistributing the time steps is neither necessary nor possible due to unaffordable computing time needed.

One advantage of the adaptive time-stepping method is that we only need to do the adaptivity in one direction. In each time level t_n, some method is used to predict the next time-step size δt. In each step, we need to ensure that

$$\int_{t_n}^{t_n+\delta t} M dt \le \delta M.$$

In practice, an approximate version may be chosen as ($\hat{M}$ is an approximation of M)

$$\int_{t_n}^{t_n+\delta t} \hat{M} dt \le \delta M. \tag{3.1}$$

If (3.1) is satisfied, the numerical solution is accepted and the time step δt will be added to t_n. So we can move to the time level t_{n+1}. Otherwise, we shorten the time step δt (bounded by $\Delta t_{\min}$ from below) and start the procedure from t_n again.

Based on the discussion above, we give a framework for the adaptive time-stepping method.

1. Find the physical quantities $\hat{M}$ and δM related to the phase change.
2. Predict the time-step size with the help of $\hat{M}$ and δM.
3. Based on (3.1) to accept or reject the time step. If (3.1) is not satisfied, then re-start the procedure with a smaller step.

Since the Cahn-Hilliard equation comes from the energy flow of (2.1), it is natural to explore the use of energy. To this end, we define the physical quantity M by using the energy change $-\partial_t E$. More precisely, we define

$$\hat{M} = -\frac{E_h(U^{n+1}) - E_h(U^n)}{\Delta t},$$

which satisfies

$$\int_{t_n}^{t_n+\Delta t} \hat{M} dt = E_h(U^n) - E_h(U^{n+1}).$$

Thus, δM can be defined as $\delta E_h^n := E_h(U^n) - E_h(U^{n+1})$ to control the energy change.

It follows from the discrete energy identity (2.8) that

$$\Delta t = -\frac{E_h(U^{n+1}) - E_h(U^n)}{\|\nabla_h \mu^{n+\frac{1}{2}}\|_h^2}. \tag{3.2}$$

Since the aim is to equi-distribute the energy variation, we set the numerator $|E_h(U^{n+1}) - E_h(U^n)|$ to be a constant δE in (3.2). This yields

$$\Delta t = \frac{\delta E}{\|\nabla_h \mu^{n+\frac{1}{2}}\|_h^2}. \tag{3.3}$$

There is still an unknown term U^{n+1} in the denominator of (3.3). Replacing it by U^n leads to a computable time predictor δt

$$\delta t = \frac{\delta E}{\|\nabla_h \mu^n\|_h^2}. \tag{3.4}$$

We now demonstrate that such a predictor is reasonable. Suppose there is an model with the energy evolution depicted in Fig. 1. A set of time steps $\{\Delta t_i\}$ divide the energy curve into N equal parts

$$\delta E = (\text{total energy variation})/N = |E_h(U^{n+1}) - E_h(U^n)|.$$

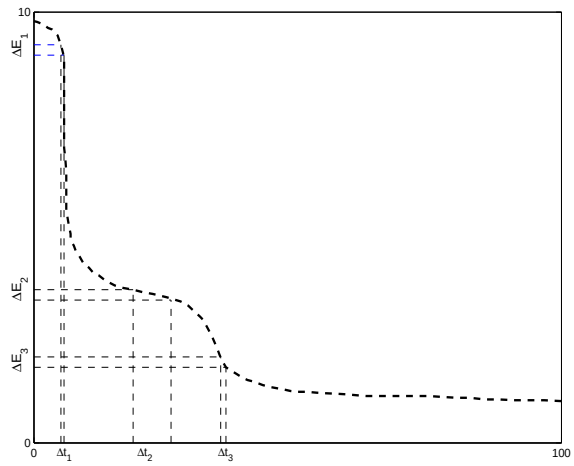


Figure 1: The energy evolution.

From (3.3), in order to equi-distribute the energy variation, the time steps Δt_i should vary according to the intensity of energy change. In case that the energy changes fast, the identity (3.3) can produce a small time step Δt_1. As the change slows down, the time step increases to Δt_2. If the energy changes fast again then the time step becomes smaller accordingly. All of these changes depend on the denominator in (3.3), which is small at first, latter becomes bigger, and then becomes small again. Comparing the equality (3.3) to (3.4), the only difference is that the latter uses U^n to approximate U^{n+1} in the former. So δt in (3.3) is regarded as a good approximation to Δt.

Such an adaptive time-stepping method is in fact an one-step fixed-point iteration for a nonlinear equation. Combing the Crank-Nicolson scheme (2.5) and the formula (3.2), and supposing $\delta E_h^{n+1} \neq 0$, we have

$$\frac{U^{n+1}-U^n}{E_h(U^{n+1})-E_h(U^n)} = \frac{\Delta_h \mu^{n+\frac{1}{2}}}{\|\nabla_h \mu^{n+\frac{1}{2}}\|_h^2}. \tag{3.5}$$

Our time adaptive method aims at equi-distributing the energy curve, i.e., a constant δE is used to control the energy decay in each time step. This requires that $E_h(U^n)-E_h(U^{n+1})=\delta E$, and (3.5) turns out to be

$$\frac{U^{n+1}-U^n}{\delta E} = -\frac{\Delta_h \mu^{n+\frac{1}{2}}}{\|\nabla_h \mu^{n+\frac{1}{2}}\|_h^2}. \tag{3.6}$$

Define the time step as

$$\Delta t = \frac{\delta E}{\|\nabla_h \mu^{n+\frac{1}{2}}\|_h^2}. \tag{3.7}$$

To apply a fixed-point iteration to solve the nonlinear problem (3.6), we replace the unknown term U^{n+1} with U^n to give the effective time step as

$$\widetilde{\Delta t} = \frac{\delta E}{\|\nabla_h \mu^n\|_h^2}. \tag{3.8}$$

It shows that the adaptive time-stepping method is of the form

$$\frac{U^{n+1}-U^n}{\widetilde{\Delta t}} = -\Delta_h \mu^{n+1},$$

which is exactly an one-step fixed-point iteration for solving the nonlinear equation (3.6).

Before proposing the algorithm, we would like to discuss three issues. The first one relates to the prescribed constant δE. The practical computational experience suggests that it does affect the time step significantly and thus it is critical. In general case, the energy changes extremely fast at initial stage. Thus, it is reasonable to use minimal time step at initial stages, i.e., the initial $[0,t_k]$ interval. We then use the average of the energy decrease values in the $[t_k/2,t_k]$ as δE to control the adaptive time-stepping method for the rest of the computational time. This way of determining δE can reflect the intensity of the initial phase change and avoid unreasonable effect to δE due to the large initial variation of the phase change.

The second issue is concerned with how to modify the time step when the computational results violate the criteria (3.1). Then we check the time-step size used: if it is the minimal size, we still accept it; otherwise, we do a local feedback procedure, i.e., shorten the time-step size and re-compute the model on this time level.

The third issue relates to the consideration of controlling accuracy by adjusting δE automatically. From (3.8), δE affects the time-step size Δt, and thus affects the accuracy. One case is that when the minimal time-step size is used and the energy variation $E_h(U^n)-E_h(U^{n+1})$ is smaller than δE, we can do an average such as

$$\delta E := \frac{m\times\delta E + \big(E_h(U^n)-E_h(U^{n+1})\big)}{m+1}, \tag{3.9}$$

to produce a smaller δE which leads to a smaller time step. Likewise, when the energy change $E_h(U^n)-E_h(U^{n+1})$ is larger than δE, an average procedure can enlarges δE which yields a larger time step.

Based on the discussion above, an adaptive time-stepping algorithm on the energy equi-distribution can be proposed as below:

Algorithm 3.1. (adaptive time-stepping algorithm based on energy decay)

1. Set $\Delta t_{\min}$ and $\Delta t_{\max}$, compute the problem (2.5) with the minimal time step $\Delta t_{\min}$ for initial t_k second(s), and set δE as the average of the energy decrease values in the time interval $[t_k/2,t_k]$. Then, set m, $n=1$, and $t_n=t_k$.

2. At the time level t_n, calculate the time step with the formula (3.8) and compute the problem (2.5) with the time step $\Delta t = \min(\Delta t_{\max}, \max(\widetilde{\Delta t}, \Delta t_{\min}))$.
3. Check the criteria, $E_h(U^n) - E_h(U^{n+1}) \le \delta E$ and $\Delta t = \Delta t_{\min}$ are true or not. If either one is true, accept the time step Δt and got to step 4; or else, return to step 2 with time step $\Delta t = \min(\Delta t_{\max}, \max(\frac{\Delta t}{2}, \Delta t_{\min}))$.
4. (Automatically adjust δE) If $\Delta t = \Delta t_{\min}$ in step 3, adjust δE with an arithmetic average (3.9) and set $m := m+1$.
5. Move to the next time level $t_{n+1} = t_n + \Delta t$, set $n := n+1$ and go to step 2.

4 Linear scheme with adaptive time steps

The adaptive time-stepping method does not limit its use to the Crank-Nicolson scheme. Combining it with other method, such as the linear scheme proposed by Eyre in [3], can also be an effective method for the Cahn-Hilliard model (1.1).

The scheme comes from the convex splitting to the energy of Cahn-Hilliard equation. Divide the energy into two parts:

$$\begin{aligned} &E(u) = E_c(u) - E_e(u), \\ &E_c(u) = \frac{\kappa}{2}\|\nabla u\|^2 + \frac{\alpha}{2}\|u\|^2 + |\Omega| \text{ is the contractive part}, \\ &E_e(u) = \frac{1}{4}\|u\|^4 - \frac{\alpha+1}{2}\|u\|^2 \text{ is expansive part}. \end{aligned}$$

Two conditions should be taken care in the convex splitting method: (1) the Jacobian matrix of ∇E satisfies $(J(\nabla E)(u)u,u) \ge \lambda(u,u)$; (2) the Jacobian matrix of ∇E_c satisfies $(J(\nabla E_c)(u)u,u) \ge \lambda_1(u,u)$ with $\lambda_1 \ge |\lambda| \ge 0$.

By direct calculation of the Jacobian matrix, we have

$$\begin{aligned} \big(J(\nabla E)(u)u,u\big) &= \int_\Omega \big(\kappa|\nabla u|^2 + u^2(3u^2-u-1)\big)dx, \\ \big(J(\nabla E_c)(u)u,u\big) &= \int_\Omega \big(\kappa|\nabla u|^2 + \alpha u^2\big)dx, \\ \big(J(\nabla E_e)(u)u,u\big) &= \int_\Omega u^2\big(-3u^2+1+\alpha+u\big)dx. \end{aligned}$$

The solution of Cahn-Hilliard equation is supposed to satisfy $u \in [-1,1]$ according to its physical meaning. A larger α leads to stronger convexity but also a larger truncation error in time. Considering these two factors, we use $\lambda = -2$, $\alpha = 1$ and $\lambda_1 = 2$. More details on the choice of α can be found in [28].

Then applying the gradient flow method to the splitting energy, we have

$$\frac{U^{n+1}-U^n}{\Delta t} = \nabla E_e(U^n) - \nabla E_c(U^{n+1}), \tag{4.1}$$

where the explicit scheme is used for the expansive part and the implicit scheme is used for the contractive part. So the linear scheme is:

$$\frac{U_{i,j}^{n+1}-U_{i,j}^{n}}{\Delta t}=\Big(-\kappa\Delta_h^2 U_{i,j}^{n+1}+\alpha\Delta_h U_{i,j}^{n+1}\Big)+\Big(\Delta_h(U_{i,j}^{n})^3-(\alpha+1)\Delta_h U_{i,j}^{n}\Big). \tag{4.2}$$

There are also other ways to divide the energy $E(u)$ to obtain some different schemes for the Cahn-Hilliard equation, which can be found in [3]. The linear scheme (4.2) satisfies the unique solvability listed below.

Lemma 4.1 ([3] unconditional energy stability)**.** *For any time-step size $\Delta t>0$, the unconditional energy stability holds for linear scheme* (4.2)

$$E_h(U^{n+1})\le E_h(U^n), \tag{4.3}$$

under the condition $U^n\in[-1,1]$ for any time level t^n.

Compared with the Crank-Nicolson scheme (2.5) which has second order accuracy in the time discretization, the linear scheme has only first order accuracy. More precisely, the truncation error of the linear scheme is

$$\text{err}=\frac{\Delta t}{2}\Big(J(\nabla E_e)(u)+J(\nabla E_e)(u)\Big)\nabla E(u(\xi)),$$

where $\xi\in(n\Delta t,(n+1)\Delta t)$.

It is easy to see such a linear scheme is uniquely solvable. Moreover, the linear scheme avoids the nonlinear iteration which is required in the nonlinear scheme (2.5).

The Crank-Nicolson scheme satisfying the discrete energy identity (2.6) which can be used to derive a time step predictor. For the linear scheme, such discrete version of the energy identity is unavailable. Thus we use the continuous one directly by replacing dE/dt with its approximation $\delta E/\delta t$ and replacing u with U^n to derive the time-step predictor

$$\delta t=\frac{\delta E}{\|\nabla_h\mu^n\|^2}. \tag{4.4}$$

Since the linear scheme is unconditionally stable with first order accuracy in time, we expect (4.4) works like (3.4) that varies the time-step size based on the intensity of the energy change.

The algorithm combining the linear scheme (4.2) and the adaptive time-stepping method (4.4) for Cahn-Hilliard equation is the same as Algorithm 3.1, except replacing (2.5) by (4.2).

The linear system in each time step can be solved efficiently if we follow Chapter 7 in [4]. Transforming (4.2) into linear system

$$\begin{aligned}&U_{i,j}^{n+1}-\Delta t\big(-\kappa\Delta_h^2 U_{i,j}^{n+1}+\alpha\Delta_h U_{i,j}^{n+1}\big)\\ =&\Delta t(\Delta_h(U_{i,j}^{n})^3-(\alpha+1)\Delta_h U_{i,j}^{n})+U_{i,j}^{n}.\end{aligned} \tag{4.5}$$

Denoted the items of (4.5) in matrix form, i.e. $U:=(U_{i,j})_{n\times n}$. The right hand side defined as $F:=(F_{i,j})_{n\times n}$, where

$$F_{i,j}:=\Delta t\left(\Delta_h((U_{i,j})^3)-(\alpha+1)\Delta_h U_{i,j}\right)+U_{i,j}.$$

Denote A as a $n\times n$ matrix:

$$A:=\frac{1}{h^2}\begin{pmatrix} 2 & -1 & 0 & \cdots & 0 & -1 \\ -1 & 2 & 1 & \cdots & 0 & 0 \\ 0 & -1 & 2 & \cdots & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ -1 & 0 & 0 & \cdots & -1 & 2 \end{pmatrix},$$

and M as a matrix with all elements equal to 1. Define the derivative operator Δ_h on matrix U as

$$\Delta_h U:=(\Delta_h U_{i,j})_{n\times n}.$$

It is easy to know

$$\Delta_h U=AU+UA.$$

The linear system can be rewritten as

$$M+\kappa(AAU+2AUA+UAA)-\alpha(AU+UA)=F. \tag{4.6}$$

Since A is symmetric, it has the decomposition $A=Z\Lambda Z^T$ with Z being orthogonal matrix and $\Lambda=\mathrm{diag}\{\lambda_i\}$. Replacing A in (4.6), we have

$$\begin{aligned} &M+\kappa(Z\Lambda Z^TZ\Lambda Z^TU+2Z\Lambda Z^TUZ\Lambda Z^T+UZ\Lambda Z^TZ\Lambda Z^T) \\ &-\alpha(Z\Lambda Z^TU+UZ\Lambda Z^T)=F. \end{aligned} \tag{4.7}$$

Pre-multiplying Z^T and post-multiplying Z on both sides, and denoting $U'=Z^TUZ$, $F'=Z^TFZ$ leads to the following equation

$$M+\kappa\Delta t(\Lambda^2U'+2\Lambda U'\Lambda+U'\Lambda^2)-\alpha\Delta t(\Lambda U'+U'\Lambda)=F'.$$

Solve this problem, we have

$$u'_{i,j}=\frac{F'_{i,j}}{1+\kappa\Delta(\lambda_i+\lambda_j)^2-\alpha\Delta t(\lambda_i+\lambda_j)}. \tag{4.8}$$

Finally, by inverse transformation, the solution is

$$U=ZU'Z^T.$$

So the algorithm for solving the linear system (4.5) (similar to Algorithm 6.13 in [4]) can be described as follows

Algorithm 4.1. [Solve the linear system]

1. Do the decomposition $A = Z\Lambda Z^T$.
2. Compute $F'_{i,j} = Z^T F Z$ and $u'_{i,j}$ with (4.8).
3. Do the inverse transformation $U = ZU'Z^T$ to get the solution.

Since the discrete FFT can be applied to decompose A and do the inverse transformation in step 3, such method is a fast algorithm. We denote the algorithm using linear scheme with adaptive time step and Algorithm 4.1 as **Algorithm 4.3**.

5 Numerical experiments

In this section, some numerical experiments are carried out to show the efficiency of the methods proposed in the previous sections.

5.1 Examples with Crank-Nicolson scheme

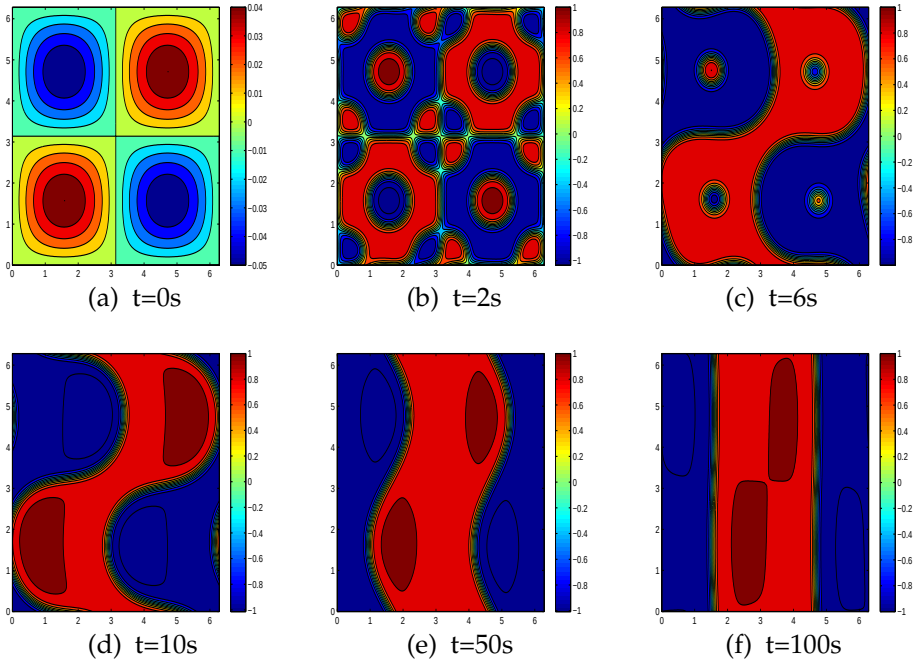
For the examples in this subsection, the Newton iteration method is adopted to solve the discrete nonlinear system (2.5). In each iteration step, the initial value is set to be the numerical solution of the previous step. The algebraic multigrid method is used to solve the linearized system at each Newton iterative step. The initial value in the algebraic multigrid method is set as the change of previous two iteration steps and the tolerance is set to be 10^{-8}.

Example 5.1. ([29]) We solve Eq. (1.1) with the initial condition:

$$u_0(x,y) = 0.05\sin x \sin y, \qquad (x,y) \in [0,2\pi]^2. \tag{5.1}$$

The parameter $\kappa = 0.01$, the periodic boundary condition is imposed.

This example was used in [29] and is used here to test the computation efficiency of Algorithm 3.1. We use a 200×200 grid to discretize the problem in space. First, we solve the problem with the uniform time step $\Delta t = 0.001$. Then the adaptive time-stepping method is also applied to solve this problem with $t_k = 0.2$, $\Delta t_{\min} = 0.001$, $\Delta t_{\max} = 0.1$, 0.5 or 1, respectively. In Fig. 2, we plot the solution contours at some selected time levels for $\Delta t = 0.001$. Also, the solution contours by adaptive time-stepping method with $\Delta t_{\min} = 0.001$ and $\Delta t_{\max} = 0.1$ are plotted in Fig. 3. From these two figures, we can find that the solutions obtained by the adaptive time-stepping method are almost the same as those obtained by the uniform time-stepping method.



(a) t=0s (b) t=2s (c) t=6s

(d) t=10s (e) t=50s (f) t=100s

Figure 2: Example 5.1: Phase contour lines on some time levels by the uniform time-stepping method with $\Delta t = 0.001$.

Table 1: Example 5.1: L^2-norm error (u_{const} denotes the solution by the uniform time step Δt=0.001, $u_{0.1\text{s}}$, $u_{0.5\text{s}}$ and $u_{1\text{s}}$ denote the adaptive time-stepping solution with $\Delta t_{\min}$=0.001 and $\Delta t_{\max}$=0.1, 0.5 and 1, respectively).

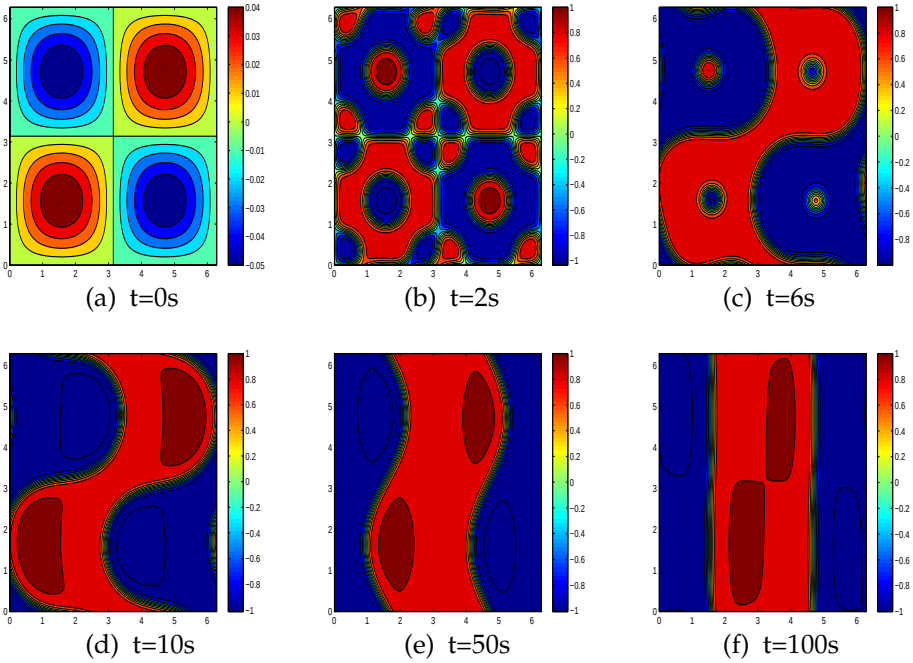
	$t=20$s	$t=40$s	$t=60$s	$t=80$s	$t=100$s
$\|u_{\text{const}}-u_{0.1\text{s}}\|_h$	9.77E-04	8.75E-04	7.91E-04	7.69E-04	7.66E-04
$\|u_{\text{const}}-u_{0.5\text{s}}\|_h$	9.80E-04	8.84E-04	8.60E-04	7.83E-04	7.76E-04
$\|u_{\text{const}}-u_{1\text{s}}\|_h$	9.80E-04	8.84E-04	1.10E-03	9.80E-04	9.18E-04

The corresponding energy curves are plotted in Fig. 4. It is observed that these energy curves obtained by the adaptive time-stepping method with different $\Delta t_{\max}$ are almost the same as those by the uniform time-stepping method with $\Delta t = \Delta t_{\min}$. The energy curves indicates that the solution has a quick motion in the early stage of the evolution, and it develops slowly latter until it reaches a steady state.

Fig. 5 presents the distribution of the time-step size. Corresponding to the fast development in the early stage, the time steps are small and then the time steps become larger when the solution reaches a steady state.

On some time levels, the L^2-norm of the difference between solutions by the adaptive time-stepping method and the uniform time step are listed in Table 1. The small error shows that the adaptive time-stepping method do enjoy high accuracy property.

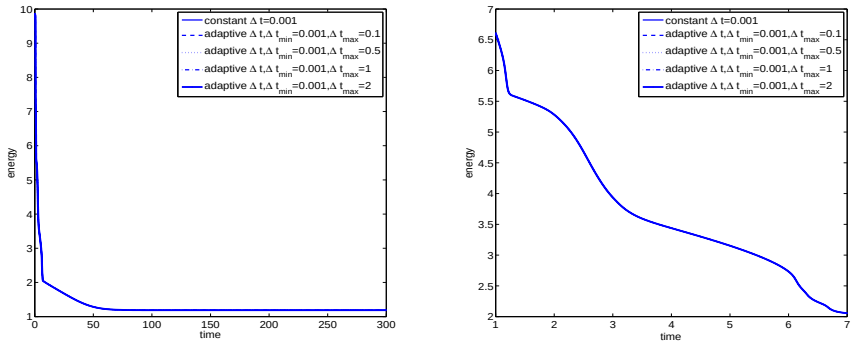
The CPU time (in seconds) consumed by the uniform time-stepping method and



(a) t=0s (b) t=2s (c) t=6s

(d) t=10s (e) t=50s (f) t=100s

Figure 3: Example 5.1: Phase contour lines on some time levels for the adaptive time-stepping method with $\Delta t_{\min}=0.001$ and $\Delta t_{\max}=0.1$.



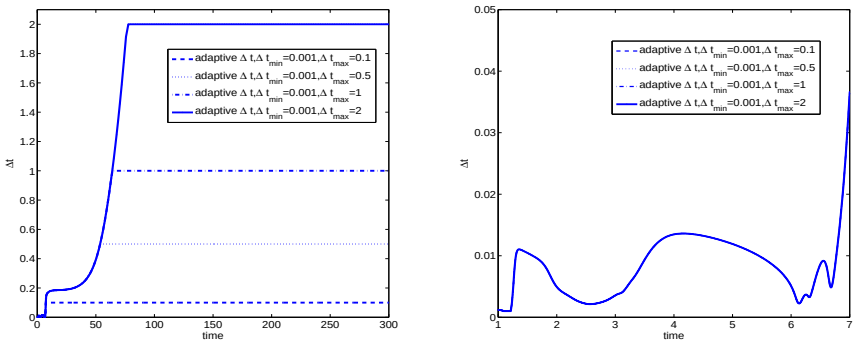
(a) $t\in[0,300]$ (b) $t\in[0,7]$

Figure 4: Example 5.1: energy evolution.

adaptive time-stepping methods with different $\Delta t_{\max}$ is depicted in Fig. 6. The adaptive time-stepping method saves the computation time for nearly 75%.

Example 5.2. In this example, the initial condition is taken as

$$u_0(x,y)=0.5\sin(x+\cos x)\cos(y+\sin 2x)$$



(a) $t\in[0,300]$ (b) $t\in[1,7]$

Figure 5: Example 5.1: the distribution of time steps.

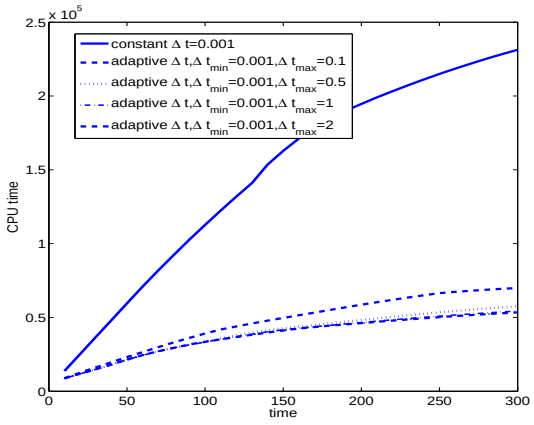


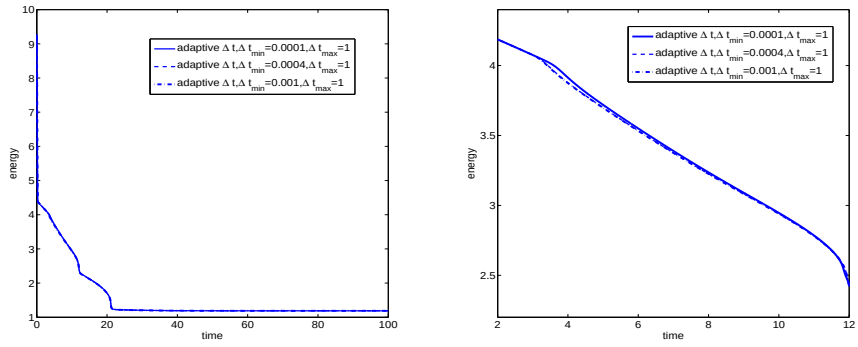
Figure 6: Example 5.1: CPU time (seconds) comparison.

and $\Omega=[0,2\pi]\times[0,2\pi]$, the periodic boundary condition and $\kappa=0.01$ are imposed.

This example is designed to show how the quantity of $\Delta t_{\min}$ affects the time-step evolution and the accuracy. The grid and t_k are the same as those in Example 5.1.

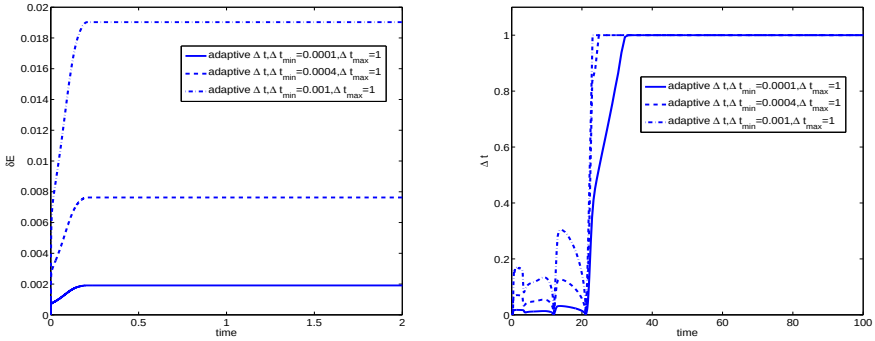
In Fig. 7, we present the energy evolution produced by the adaptive time-stepping with three different $\Delta t_{\min}$. The energy decrease quantities δE and the distribution of the time-step size Δt for the adaptive time-stepping with three different choices $\Delta t_{\min}=0.001,0.0004,0.0001$ are presented in Fig. 8. The left subfigure in Fig. 8 shows that $\delta E/\Delta t_{\min}$ are approximate to 20 for all the three choices. The right subfigure in Fig. 8 shows that anyone of the three choices have a similar pattern of reaction to the energy decay and they can eventually produce time steps large enough.

The corresponding CPU time (in seconds) for different choice of $\Delta t_{\min}$ are shown in Fig. 9, which is consistent with the distributions of time steps (the right one in Fig. 8).



(a) $t\in[0,100]$ (b) $t\in[2,12]$

Figure 7: Example 5.2: energy evolution.



(a) $t\in[0,2]$ (b) $t\in[0,100]$

Figure 8: Example 5.2: energy decrease and the distributions of time steps.

Example 5.3. Here, the initial condition is a random assignment with values in $[-0.1,0.1]$ on the grid point $(x,y)\in[0,2\pi]\times[0,2\pi]$. The periodic boundary condition and $\kappa=0.001$ is imposed.

We use the same spatial mesh and t_k as in Example 5.1 and set $\Delta t_{\min}=0.0001$ and $\Delta t_{\max}=1$. Fig. 10 presents the phase contour lines on some selected time levels.

In Fig. 11, the energy evolution curves and the distributions of time steps against time are presented in the left subfigure, from which we see that the growing up of the time step coincides with the slowing down of the energy decay. In the right of Fig. 11, the local zoom of the left figure in the time interval $[9.5,15]$ is presented, where we can see a sharp turning down of the time-step size happens where the energy decays quickly.

The CPU time (in seconds) is presented in Fig. 12. The adaptive time-stepping method saves the computation time and improves the computing efficiency.

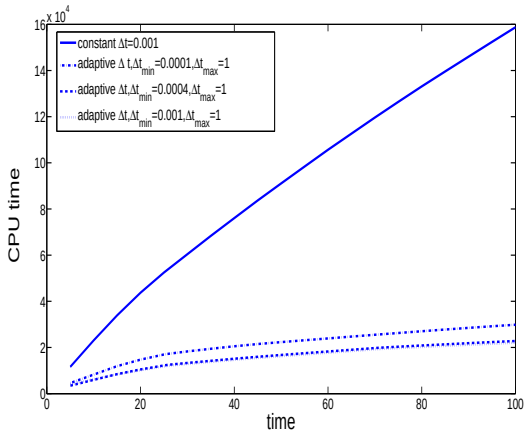
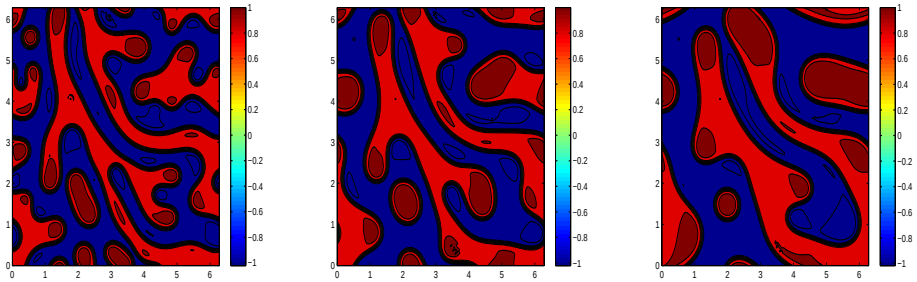
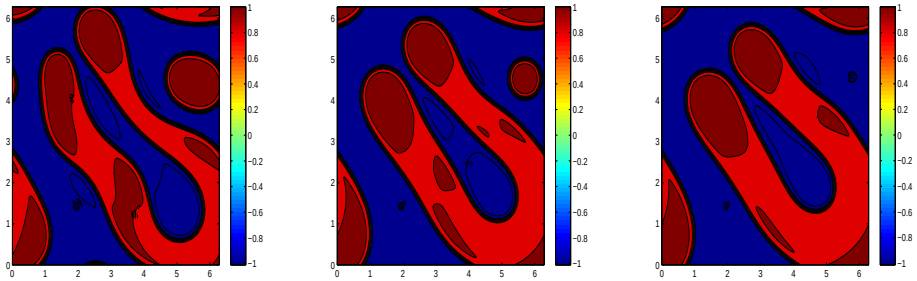


Figure 9: Example 5.2: CPU time (seconds) comparison.



(a) t=2.0039s (b) t=4.0189s (c) t=8.0568s

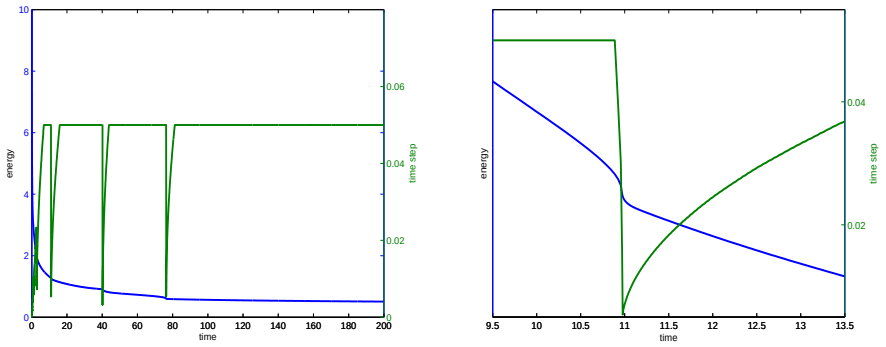


(d) t=16.0064s (e) t=32.0413s (f) t=40.0129s

Figure 10: Example 5.3: Phase contour lines on some time level.

5.2 Example with convex splitting method

In this subsection, the example 4 in [12] is used to check the efficiency of the adaptive time-stepping method with the linear scheme.



(a) $t\in[0,40]$ (b) $t\in[9.5,15]$

Figure 11: Example 5.3: the distributions of time steps according to energy evolution.

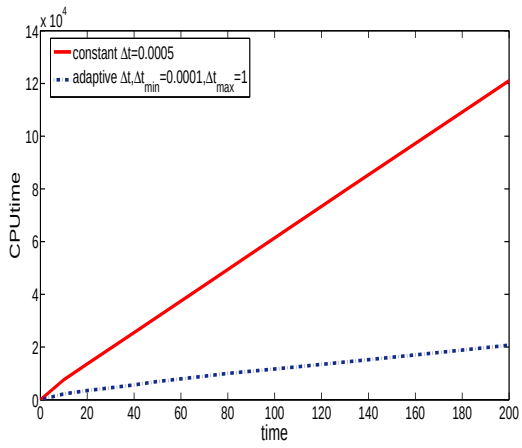


Figure 12: Example 5.3: CPU time (seconds).

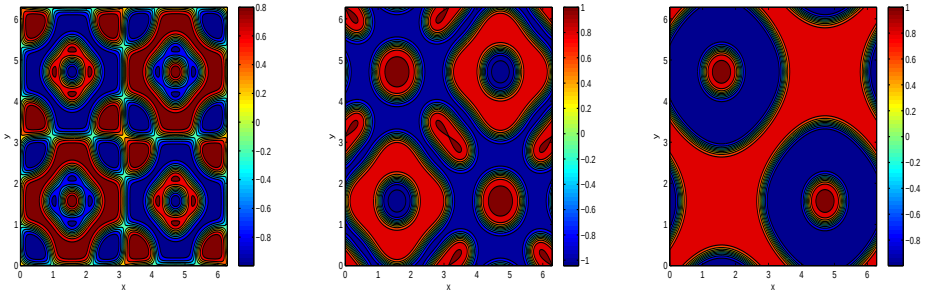
Example 5.4. The initial condition is set to be the trigonometric function

$$u_0(x,y)=0.05\sin x\sin y+0.01,\qquad (x,y)\in[0,2\pi]^2. \tag{5.2}$$

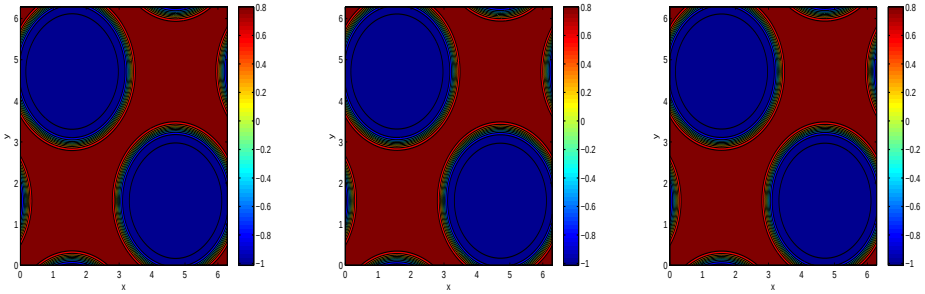
The parameter $\kappa=0.01$, the periodic boundary condition is imposed.

Due to the accuracy and convergence order consideration, we use a finer mesh with the meshgrid 400×400 and choose $\Delta t_{\min}=0.0001$, $\Delta t_{\max}=0.1$ and $t_k=0.05$ for the adaptive time-stepping method. First, we plot the solution contour lines as before at some selected time levels in Figs. 13 and 14 for the uniform time-stepping method and the adaptive time-stepping method, respectively.

The contours in Figs. 13 and 14 looks similar to Fig. 11 in [12]. Just from the contour figures, we can not see the difference between these two methods. And the energy figure

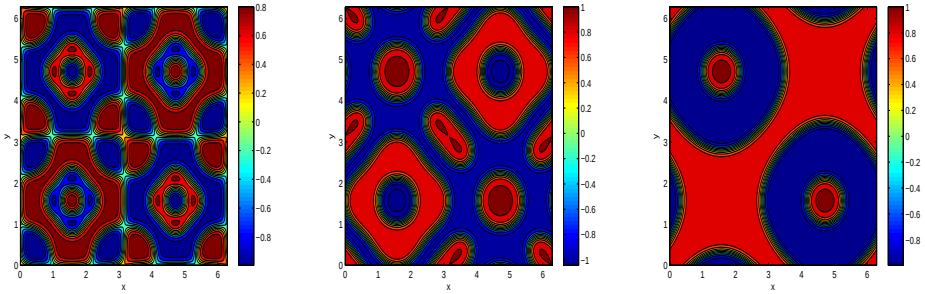


(a) t=1s (b) t=2s (c) t=5s

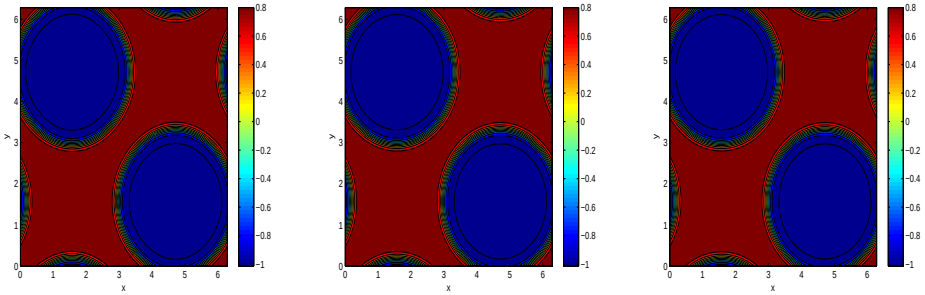


(d) t=10s (e) t=20s (f) t=50s

Figure 13: Example 5.4: Phase contour lines on some time levels by the uniform time-stepping method with $\Delta t = 0.0001$.

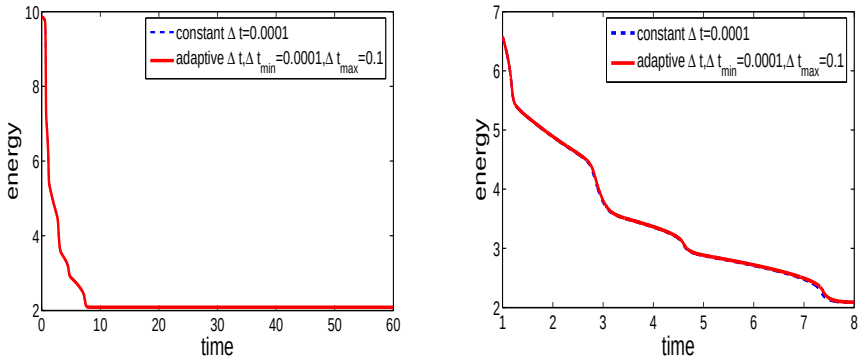


(a) t=1s (b) t=2s (c) t=5s



(d) t=10s (e) t=20s (f) t=50s

Figure 14: Example 5.4: Phase contour lines on some time levels for the adaptive time-stepping method with $\Delta t_{\min} = 0.0001$, $\Delta t_{\max} = 0.1$ and $K = 0.5$.



(a) $t \in [0,60]$ (b) $t \in [1,8]$

Figure 15: Example 5.4: energy evolution.

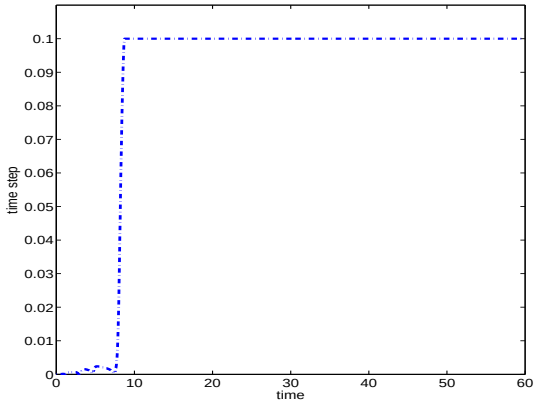


Figure 16: Example 5.4: the distributions of time steps.

from 0s to 60s of the two methods look the same. Only if we zoom the figure, we find there are slightly difference between them on the time interval $[1s,8s]$, see Fig. 15.

Fig. 16 shows the time step evolution. After some fluctuation in first few seconds, it gradually reaches to $\Delta t_{\max}$. Finally, the CPU time is showed in Fig. 17. The convex splitting method does save CPU time greatly by enlarging the time steps to reduce the computational times. It needs only 1/28 CPU time of the uniform time-stepping method.

We close this section by making the following observations. From the numerical examples presented in this section, by producing δE automatically, we reduce the artificial effect to the adaptive time-stepping scheme. The sensitivity of the time-step size to the intensity of the energy decay leads to the reduction in the number of the time steps. Also with such sensitivity, this method can produce enough large time step when the energy changes slowly, even starting from a very small time step $\Delta t_{\min}$. Moreover, it keeps the

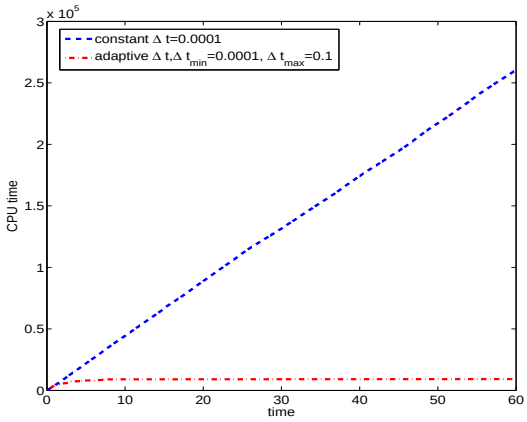


Figure 17: Example 5.4: CPU time (seconds).

accuracy quite well and saves computational time greatly. The adaptive time-stepping method can also be used in convex splitting method, as it is presented in Example 5.4. Together with the discrete FFT, the linear scheme saves CPU time much more greatly.

6 Concluding remarks

This work presents an adaptive time-stepping algorithms for simulating long time evolution of the Cahn-Hilliard problem. The time-adaptive algorithm is associated with two energy stable time-stepping schemes, i.e., the Crank-Nicolson scheme and the convex splitting method [3]. Since the algorithms follow the guide of the equality (2.8), the time-step size changes according to the phase (energy) variation. With the four numerical examples, the robustness, accuracy and the efficiency are tested. As the main purpose of the time adaptivity is to vary the time steps so larger time steps can be used whenever possible while preserving the overall accuracy, the numerical experiments indicate that the proposed algorithms serves the basic role. In future works, we may extend the present approach to more complicated phase field problems such as the molecular beam epitaxy model (see, e.g., [20,28]) and more general free energy such as the one in log form (see, e.g., [18,22]).

Acknowledgments

The work of the first author was supported in part by the National Science Foundation of China through NSFC 11401129. The second author gratefully acknowledge the financial support by the Hong Kong Research Grants Council CERG grants and Hong Kong Baptist University FRG grants. The work of the third author was supported in part by

the National Science Foundation of China through NSFC 11371026, 91330202, 11001259, 11031006, 2011CB309703, the National Center for Mathematics and Interdisciplinary Science and the President Foundation of AMSS-CAS.

References

[1] J. W. Cahn and J. E. Hilliard, Free energy of a non-uniform system I: Interfacial free energy, J. Chem. Phys., 28 (1958), 258-267.
[2] S. M. Choo, S. K. Chung and K. I. Kim, Conservative nonlinear difference scheme for the Cahn-Hilliard equation-II, Comput. Math. Appl., 39 (2000), 229-243.
[3] D. Eyre, An unconditionally stable one-step scheme for gradient systems, Preprint, 1998.
[4] J. W. Demmel, Applied Numerical Linear Algebra. Society for Industrial and Applied Mathematics, 1997.
[5] J. C. Doyle, B. A. Francis and A. R. Tannenbaum, Feedback Control Theory, Macmillan Publishing Co., 1990.
[6] Q. Du and R. A. Nicolaides, Numerical analysis of a continuum model of phase transition, SIAM J. Numer. Anal., 28 (1994), 1310-1322.
[7] Q. Du, L. Tian and L. Ju, Finite element approximation of the Cahn-Hilliard equation on surfaces, Comp. Meth. Appl. Mech. Engr, 200 (2011), 2458-2470.
[8] C. M. Elliott and D. A. French, Numerical studies of the Cahn-Hilliard equation for phase separation, IMA J. Appl. Math., 38 (1987), 97-128.
[9] C. M. Elliott and S. M. Zheng, On the Cahn-Hilliard equation, Arch. Rat. Meth. Anal., 96 (1986), 339-357.
[10] L. Cueto-Felgueroso and J. Peraire, A time-adaptive finite volume method for the Cahn-Hillard and Kuramoto-Sivashinsky equations, J. Comput. Phys., 227 (2008), 9985-10017.
[11] W. Feng, P. Yu, S. Hu, Z. Liu, Q. Du and L. Chen, A Fourier spectral moving mesh method for the Cahn-Hilliard equation with elasticity, Commun. Comput. Phys., 5 (2009), 582-599.
[12] X. Feng, T. Tang and J. Yang, Long time numerical simulations for phase-field problems using p-adaptive spectral deferred correction methods, SIAM J. Sci. Computing, 37 (2015), A271-A294.
[13] G. Franklin, J. D. Powell and A. Emami-Naeini, Feedback Control of Dynamic Systems, Prentice Hall, 2006.
[14] D. Furihata, A stable and conservative finite difference scheme for the Cahn-Hilliard equation, Numer. Math., 87 (2001), 675-699.
[15] H. Gomez, T.J.R. Hughes, Provably unconditionally stable, second-order time accurate, mixed variational methods for phase-field models, J. Comput. Phys., 230 (2011), 5310-5327.
[16] L. He and Y. Liu, A class of stable spectral methods for the Cahn-Hillard equation, J. Comput. Phys., 228 (2009), 5101-5110.
[17] Y. N. He, Y. X. Liu and T. Tang, On large time-stepping methods for the Cahn-Hilliard equation, Appl. Numer. Math., 57 (2007), 616-628.
[18] B. Li and J. G. Liu, Thin film epitaxy with or without slope selection, European J. Appl. Math., 14 (2003), 713-743.
[19] A. Novick-Cohen and L. A. Segel, Nonlinear aspects of the Cahn-Hilliard equation, Phys. D, 10 (1984), 277-298.
[20] Z. H. Qiao, Z. R. Zhang, and T. Tang, An adaptive time-stepping strategy for the molecular beam epitaxy models, SIAM. J. Sci. Comput., 33 (2011), 1395-1414.

[21] J. Shen, T. Tang and L. Wang, Spectral Methods: Algorithms, Analysis and Applications, Springer-Verlag, Berlin Heidelberg, 2011.
[22] J. Shen, C. Wang, X. Wang and S. Wise, Second-order convex splitting schemes for gradient flows with Ehrlich-Schwoebel type energy: Application to thin film epitaxy, SIAM J. Numer. Anal., 50 (2012), 105-125.
[23] J. Shen and X. Yang, Numerical approximations of Allen-Cahn and Cahn-Hilliard equations, DCDS, Series A, 28 (2010), 1669-1691.
[24] G. Söderlind, Automatic control and adaptive time-stepping, Numer. Algor., 31 (2001), 281-310.
[25] Z. Z. Sun, A second-order accurate linearized difference scheme for the two-dimensional Cahn-Hilliard equation, Math. Comput., 64 (1995), 1463-1471.
[26] G. Wells, E. Kuhl and K. Garikipati, A discontinuous Galerkin method for the Cahn-Hillard equation, J. Comput. Phys., 218 (2006), 860-877.
[27] S. M. Wise, Unconditionally stable finite difference, nonlinear multigrid simulation of the Cahn-Hillard-Hele-Shaw system of equations, J. Sci. Comput., 44 (2010), 38-68.
[28] C. J. Xu and T. Tang, Stability analysis of large time-stepping methods for epitaxial growth models, SIAM J. Numer. Anal., 44 (2006), 1759-1779.
[29] Z. R. Zhang and Z. H. Qiao, An adaptive time-stepping strategy for the Cahn-Hilliard equation, Commun. Comput. Phys., 11 (2012), 1261-1278.
[30] J. Zhu, L. Chen, J. Shen and V. Tikare, Coarsening kinetics from a variable mobility Cahn-Hilliard equation-application of semi-implicit Fourier spectral method, Phys. Review E, 60(4) (1999), 3564-3572.