Received: 10 July 2018 | Revised: 25 October 2018 | Accepted: 21 December 2018 | Published on: 21 January 2019

DOI: 10.1002/num.22350

RESEARCH ARTICLE

WILEY

Adaptive operator splitting finite element method for Allen–Cahn equation

Yunqing Huang[1] | **Wei Yang**[1] | **Hao Wang**[1] | **Jintao Cui**[2,3]

[1]Hunan Key Laboratory for Computation and Simulation in Science and Engineering, Xiangtan University, Xiangtan, China
[2]Department of Applied Mathematics, The Hong Kong Polytechnic University, Hung Hom, Hong Kong
[3]Department of Applied Mathematics, The Hong Kong Polytechnic University Shenzhen Research Institute, Shenzhen, China

Correspondence
Jintao Cui, Department of Applied Mathematics, TU829, Block T, The Hong Kong Polytechnic University, 11 Yuk Choi Rd, Hung Hom, Hong Kong.
Email: jintao.cui@polyu.edu.hk

Funding information
National Natural Science Foundation of China, 11671340117713671177137191430213916 30205. Natural Science Foundation of Hunan Province, 2017jj3304. Research Grants Council, University Grants Committee, 15302518.

In this paper, a new numerical method is proposed and analyzed for the Allen–Cahn (AC) equation. We divide the AC equation into linear section and nonlinear section based on the idea of operator splitting. For the linear part, it is discretized by using the Crank–Nicolson scheme and solved by finite element method. The nonlinear part is solved accurately. In addition, *a posteriori* error estimator of AC equation is constructed in adaptive computation based on superconvergent cluster recovery. According to the proposed *a posteriori* error estimator, we design an adaptive algorithm for the AC equation. Numerical examples are also presented to illustrate the effectiveness of our adaptive procedure.

KEYWORDS
adaptive algorithm, Allen–Cahn equation, finite element method, operator splitting, SCR

1 | INTRODUCTION

In this paper, we adopt Strang operator splitting method for solving the Allen–Cahn (AC) equation

$$\begin{cases} u_t - \Delta u = -\frac{1}{\epsilon^2} f(u) & \text{in } \Omega \times T, \\ \frac{\partial u}{\partial n} = 0 & \text{on } \partial\Omega \times T, \\ u(x,0) = u_0 & \text{in } \Omega \times 0. \end{cases} \tag{1.1}$$

Here Ω is a convex polygonal domain, ϵ defines the thickness of interfaces separating different phases and u represents the concentration of one of the two metallic components of the alloy.

Without lose of generality, we consider the bistable nonlinearity

$$f(u) = F'(u),$$

where $F(u) = \frac{1}{4}(u^2-1)^2$ is a double equal well potential which satisfies the global minimum value 0 at $u = \pm 1$. The AC equation can also be regarded as the L^2-*gradient* flow of Liapunov energy functional

$$E(u) = \int_\Omega \left\{ \frac{1}{2}|\nabla u|^2 + \frac{1}{\varepsilon^2}F(u) \right\} dx. \tag{1.2}$$

By differentiating the energy $E(u)$ with respect of t, we obtain an intrinsic property of AC equation as follows:

$$\begin{aligned} \frac{d}{dt}E(u) &= \int_\Omega \left(\nabla u \cdot \nabla u_t + \frac{F'(u)}{\varepsilon^2} \cdot u_t \right) dx \\ &= \int_\Omega \left(-\Delta u \cdot u_t + \frac{F'(u)}{\varepsilon^2} \cdot u_t \right) dx \\ &= -\int_\Omega u_t^2 dx \\ &\leq 0. \end{aligned}$$

In other words, energy is decreasing in time.

In 1979, Allen and Cahn in [8] first applied the AC equation to describe the motion of antiphase boundaries in crystalline solids. Soon after, the AC equation was brought into play widely, such as image analysis [6], crystal growth [7], and mean curvature motion [9], and so forth. In [10], *a posteriori* error estimate of residual type was developed by Feng and Wu. They proved that the error relied on ε^{-1} only in some low polynomial order for AC equation.

Adaptive finite element was first proposed by Babuška et al. in 1977 [2]. Then in 1986, they presented accuracy estimates and adaptive refinements [3]. When the solution is singular, adaptive finite element can find the region of singular solution and refine the mesh locally so as to improve the accuracy of the solution. In this paper, we use time-splitting adaptive finite element method to solve the AC equation based on the idea of [11, 12]. The basic idea of adaptive algorithm has the following steps. Starting from an initial shape regular mesh, we apply the successive mesh refinement algorithm (in form of loops) consisting of *Solve* → *Estimate* → *Mark* → Refine to generate a sequence of meshes. The proposed numerical scheme can be applied in the *solve* step to obtain a numerical approximation on each level of meshes. Next we derive a reliable and efficient *a posteriori* error estimator in the *estimate* step. Then the error estimator is employed to mark the elements on which the error indicators are large. Finally we design rules to refine the marked elements such that the resulting mesh is still shape regular.

The basic conditions for the adaptive algorithm to work well are reliability and efficiency of the *a posteriori* error estimator, and the adjusted discrete space can capture enough high frequency information. The *a posteriori* error estimator is the basis for adjusting the discrete space, and adjustment of discrete space is the important part of adaptive method. Traditional operator-splitting methods include sequential operator-splitting [4], symmetrically weighted sequential operator-splitting [4], Strang-Marchuk (or Strang) operator-splitting [5], iterative operator-splitting method [4] and so forth. In this paper, we mainly use Strang operator-splitting to solve AC equation.

This article is organized as follows. In Section 2, we give some notations and preliminaries. Section 3 is devoted to deriving the discrete scheme of heat equation and constructing *a posteriori* error estimator for gradient reconstruction by superconvergent cluster recovery (SCR). Based on the *a posteriori* error estimator introduced in Section 3, we propose an adaptive algorithm in Section 4. Finally, in Section 5, several numerical experiments are presented to show the validity of the proposed *a posteriori* error estimator and adaptive algorithm.

2 | PRELIMINARIES AND FINITE ELEMENT FORMATS

In order to get our discretization, we will use the following notations. Let $\Omega \subseteq \mathbb{R}^2$ be a bounded region, defining $L^p(\Omega)$ space such as

$$L^p(\Omega) = \{f\{x\} | f\{x\} \text{ is measurable in } \Omega \text{ and } \|f\|_{L^p} < \infty\}.$$

And we use the notations

$$\begin{aligned} \|f\|_{0,p,\Omega} &= \left(\int_\Omega |f(x)|^p dx\right)^{\frac{1}{p}}, \quad 1 \le p < \infty, \\ \|f\|_{0,p,\Omega} &= \operatorname*{ess\,sup}_{x\in\Omega} |f(x)|, \quad p = \infty. \end{aligned} \tag{2.1}$$

Next, when $k \in N$, $p \ge 1$, the Sobolev space defined as follows

$$W^{m,p}(\Omega) = \{v \mid D^\alpha u \in L^p(\Omega), |\alpha| \le m\}. \tag{2.2}$$

Then we have the standard norm of Sobolev space such as

$$\begin{aligned} \|u\|_{m,p,\Omega} &= \left(\sum_{|\alpha|\le m} \|D^\alpha u\|^p_{0,p,\Omega}\right)^{\frac{1}{p}}, \quad 1 \le p < +\infty, \\ \|u\|_{m,\infty,\Omega} &= \max_{|\alpha|\le m} \operatorname*{ess\,sup}_{\mathbf{x}\in\Omega} |D^\alpha u(\mathbf{x})|, \quad p = \infty, \end{aligned} \tag{2.3}$$

and the Sobolev seminorm associated with $W^{m,p}(\Omega)$ is

$$\begin{aligned} |u|_{m,p,\Omega} &= \left(\sum_{|\alpha|= m} \|D^\alpha u\|^p_{0,p,\Omega}\right)^{\frac{1}{p}}, \quad 1 \le p < +\infty, \\ |u|_{m,\infty,\Omega} &= \max_{|\alpha|= m} \operatorname*{ess\,sup}_{\mathbf{x}\in\Omega} |D^\alpha u(\mathbf{x})|, \quad p = \infty. \end{aligned} \tag{2.4}$$

We describe the operator-splitting method for the general development equation as

$$\frac{\partial u}{\partial t} = g(u) = Au + Bu, t \in (0, T], u(0) = u_0, \tag{2.5}$$

where $g(u)$ is a nonlinear operator, the choice of the operator A and B is arbitrary.

For the general development equation (2.5), the operator splitting format can be described by the following algorithm:

Algorithm 1

Step 1: solve $\frac{dv}{dt} = Av$ with initial value $v(t_n) = u(t_n)$, $t_n \le t \le t_{n+1/2}$

Step 2: solve $\frac{dw}{dt} = Bw$ with initial value $w(t_n) = v(t_{n+1/2})$, $t_n \le t \le t_{n+1}$

Step 3: solve $\frac{du}{dt} = Au$ with initial value $u(t_{n+1/2}) = w(t_{n+1})$, $t_{n+1/2} \le t \le t_{n+1}$

Step 4: update $u(t_n) = u(t_{n+1})$, and go to Step 1.

According to Strang operator-splitting format, the AC equation has the following two algorithms:

Algorithm 2

Step 1: solve $\frac{\partial \bar{u}}{\partial t} = -\frac{1}{\varepsilon^2} f(\bar{u}),\ \bar{u}(t_m) = u(t_m),\ t_m \le t \le t_{m+1/2}$

Step 2: solve $\frac{\partial \tilde{u}}{\partial t} = \Delta \tilde{u},\ \tilde{u}(t_m) = \bar{u}(t_{m+1/2}),\ t_m \le t \le t_{m+1}$

Step 3: solve $\frac{\partial \bar{u}}{\partial t} = -\frac{1}{\varepsilon^2} f(\bar{u}),\ \bar{u}(t_{m+1/2}) = \tilde{u}(t_{m+1}),\ t_{m+1/2} \le t \le t_{m+1}$

Step 4: update $u(t_m) = u(t_{m+1})$, and go to Step 1.

Algorithm 3

Step 1: solve $\frac{\partial \bar{u}}{\partial t} = \Delta \bar{u},\ \bar{u}(t^m) = u(t_m),\ t_m \le t \le t_{m+1/2}$

Step 2: solve $\frac{\partial \tilde{u}}{\partial t} = -\frac{1}{\varepsilon^2} f(\tilde{u}),\ \tilde{u}(t_m) = \bar{u}(t_{m+1/2}),\ t_m \le t \le t_{m+1}$

Step 3: solve $\frac{\partial \bar{u}}{\partial t} = \Delta \bar{u},\ \bar{u}(t_{m+1/2}) = \tilde{u}(t_{m+1}),\ t_{m+1/2} \le t \le t_{m+1}$

Step 4: update $u(t_m) = u(t_{m+1})$, and go to Step 1.

Here we remark operator $A : \frac{\partial \overline{u}}{\partial t} = -\frac{1}{\varepsilon^2} f(\overline{u})$ and $B : \frac{\partial \widetilde{u}}{\partial t} = \Delta \widetilde{u}$.

For A, the exact solution is

$$\overline{u}_{m+\alpha} = \frac{u_m}{\sqrt{u_m^2 + (1 - u_m^2) e^{-\frac{2}{\varepsilon^2}\alpha}}}, \tag{2.6}$$

where α is the length of time interval. For B, we use finite element method to solve it, see the next subsection.

Next, we will give the derivation of exact solution of the nonlinear part

$$\frac{du}{dt} = -f(u), \tag{2.7}$$

where $f(u) = \frac{1}{\varepsilon^2} u(u^2 - 1)$.

First, by multiplying 2 at both sides of (2.7) and then separation of variables, we have

$$\int_\Omega \left(\frac{2}{u} - \frac{1}{u+1} - \frac{1}{u-1} \right) du = \int_\alpha^\beta \frac{2}{\varepsilon^2} d\tau, \tag{2.8}$$

then by integration we can obtain

$$\ln \left(\frac{u^2}{u^2 - 1} \right) = \frac{2}{\varepsilon^2} (\beta - \alpha) + C. \tag{2.9}$$

Noting when $\beta = \alpha$, and $u(x, \alpha) = u_0$, so we can easily get

$$C = \ln \left(\frac{u_0^2}{u_0^2 - 1} \right). \tag{2.10}$$

Hence, the solution of u can be written in the form as following

$$u = \frac{u_0}{\sqrt{u_0^2 + (1 - u_0^2) e^{-2(\beta - \alpha)/\varepsilon^2}}}. \tag{2.11}$$

We know the nonlinear part A can be solved accurately and the linear part B is solved numerically, the results achieved by Algorithm 2 will be better than Algorithm 3. The reasons are as follows. Firstly, by using classic Newton iterative method as a standard to compare two algorithms, and we conclude that Algorithm 2 can achieve the same effect as Newton iterative method. Secondly, Algorithm 2 solves only one nonlinear equation, while Algorithm 3 solves two nonlinear equations. From the perspective of calculations and errors, Algorithm 2 is better. Hence, we apply Algorithm 2 to solve AC equation in our paper.

We consider the two-dimensional Heat equation as follows

$$\begin{cases} u_t = \Delta u \ \text{ in } \ \Omega_T \coloneqq \Omega \times (0,T], & \text{(a)} \\ \frac{\partial u}{\partial n} = 0 \ \text{ on } \ \partial\Omega_T \coloneqq \partial\Omega \times (0,T], & \text{(b)} \\ u_0 = \psi \ \text{ in } \ \Omega \times \{\mathrm{T} = 0\}. & \text{(c)} \end{cases} \tag{2.12}$$

Let

$$H^1(\Omega) = \{v | v \in L^2(\Omega), \nabla v \in L^2(\Omega)\}. \tag{2.13}$$

Now, we introduce bilinear form and inner product separately,

$$a(u,v) = \int_\Omega \nabla u \cdot \nabla v dx, \tag{2.14}$$

$$(u_t, v) = \int_\Omega u_t \cdot v dx. \tag{2.15}$$

The weak form of (2.12) reads: find u such that for every fixed time t, $u \in H^1(\Omega)$ and

$$(u_t, v) + a(u,v) = 0, \quad \forall v \in H^1(\Omega). \tag{2.16}$$

Next, let τ be a triangle and let $P_1(\tau)$ be the space of linear functions on τ, defined by

$$P_1(\tau) = \{v : v = c_0 + c_1 x + c_2 y, (xy) \in \tau, c_0, c_1, c_2 \in \mathbb{R}\}. \tag{2.17}$$

Let the mesh $T_h = \{\tau\}$, we define

$$V_h = \{v_h \in C^0(\Omega), v|_\tau \in P_1(\tau), \forall \tau \in T_h\}. \tag{2.18}$$

Here, $C^0(\Omega)$ denotes the space of all continuous functions on Ω.

We make the space discrete ansatz

$$u_h = \sum_{j=1}^{N} \xi_j(t) \varphi_j, \tag{2.19}$$

where $\{\varphi_j\}_{j=0}^N$ is a basis for V_h associated with the nodes and such that

$$\varphi_j(x_i, y_i) = \begin{cases} 1, & \text{if } i = j \\ 0, & \text{if } i \neq j \end{cases}, \quad i,j = 0,1,\ldots,N. \tag{2.20}$$

Substituting (2.19) into (2.16), and taking $v_h = \varphi_i$, $i = 1, \ldots, N$, we obtain a system of N ordinary differential equations

$$\sum_{j=1}^{N} (\varphi_j, \varphi_i) \xi_j' + \sum_{j=1}^{N} a(\varphi_j, \varphi_i) \xi_j = 0. \tag{2.21}$$

In matrix form, we get

$$M\xi' + A\xi = 0, \tag{2.22}$$

where M is the mass matrix and A is the stiffness matrix.

For the time part, we use Crank–Nicolson (CN) scheme

$$M\frac{\xi_l-\xi_{l-1}}{k_l}+A\frac{\xi_l+\xi_{l-1}}{2}=0, \tag{2.23}$$

we obtain

$$\xi_l=(2M+k_lA)^{-1}(2M-k_lA)\xi_{l-1}. \tag{2.24}$$

The scheme of (2.23) can be also written as

$$\frac{1}{k_l}(\xi_l-\xi_{l-1},v_h)+\frac{1}{2}a(\xi_l+\xi_{l-1},v_h)=0. \tag{2.25}$$

Theorem 1 *The scheme of (2.25) is stable with respect to initial values.*

Proof. Taking $v_h=\xi_l-\xi_{l-1}$, we can obtain

$$\frac{1}{k_l}\|\xi_l-\xi_{l-1}\|^2+\frac{1}{2}a(\xi_l+\xi_{l-1},\xi_l-\xi_{l-1})=0. \tag{2.26}$$

Due to the symmetry of $a(\cdot,\cdot)$, we can get

$$a(\xi_l+\xi_{l-1},\xi_l-\xi_{l-1})=a(\xi_l,\xi_l)-a(\xi_{l-1},\xi_{l-1})=|\xi_l|_1^2-|\xi_{l-1}|_1^2. \tag{2.27}$$

From (2.26), (2.27), we obtain

$$|\xi_n|_1\le|\xi_{n-1}|_1\le\cdots\le|\xi_0|_1. \tag{2.28}$$

Finally, by Poincaré inequality, we complete the proof. ∎

3 | *A POSTERIORI* ERROR ESTIMATION AND ADAPTIVE ALGORITHM

3.1 | *A posteriori* error estimation based SCR

A new gradient recovery technique based on SCR is proposed in [11]. It can be used as *a posteriori* error estimator, which is relatively simple to implement, cheap in terms of storage, and computational cost for adaptive algorithms. Below we briefly describe the SCR method.

For an interior vertex $z=z_0=(x_0,y_0)\in\mathcal{N}_h$, we want to recover the gradients on it. Select some points $z_i=(x_i,y_i)$, $1\le i\le n$ (at least 4) so that they locate around z as symmetrically as possible. Generally, choosing the mesh nodes performs well.

The SCR operator at z is defined by

$$(G_hu_h)(z)=\nabla p_z(z),$$

where $p_z(x,y)$ is a linear polynomial satisfying $p_z(x,y)=\arg\min_{p\in P_1}\sum_{i=0}^{n}|(u_h-p)(z_i)|^2$. Without loss of generality, let $h=\max\{|x_i-x_0|,|y_i-y_0|:1\le i\le n\}$. To avoid the computational instability resulting from small h, we introduce coordinate transformation

$$F:(x,y)\to(\xi,\eta)=\frac{(x,y)-(x_0,y_0)}{h}.$$

Then we can rewrite the fitting polynomial as

$$p_z(x,y)=\mathbf{P}^T\mathbf{a}=\widehat{\mathbf{P}}^T\widehat{\mathbf{a}},$$

with

$$\mathbf{P}^T=(1,x,y),\quad \widehat{\mathbf{P}}^T=(1,\xi,\eta),$$

$$\mathbf{a}^T=(a_1,a_2,a_3),\quad \widehat{\mathbf{a}}^T=(\widehat{a}_1,\widehat{a}_2,\widehat{a}_3)=(a_1+a_2x_0+a_3y_0,ha_2,ha_3).$$

Algorithm 4

Given time error tolerance Γ_{t0} and Γ_{t1}, and space error tolerance Γ_s, and the discrete solution u_n on the mesh $\mathcal{T}_n$, the time step size k_n and the time t_n.

Step 1: Taken $\mathcal{T}_{n+1} := \mathcal{T}_n$.
$k_{n+1} := k_n$.
$t_{n+1} := t_n + k_{n+1}$.
Solve u_{n+1} using **Algorithm2** on the current mesh $\mathcal{T}_{n+1}$, then calculate the time error estimators η_t.

Step 2: **While** $\eta_t > \Gamma_{t0}$
$k_{n+1} := \delta_1 k_n$.
$t_{n+1} := t_n + k_{n+1}$.
Solve u_{n+1} using **Algorithm2** on the current mesh $\mathcal{T}_{n+1}$, then calculate the time error estimators η_t and the space error estimator η_s.
End While.

Step 3: **Do**
Mark the elements for refinement and coarsening, then adapt mesh $\mathcal{T}_{n+1}$.
Solve u_{n+1} using **Algorithm2** on the current mesh $\mathcal{T}_{n+1}$, the calculate the time error estimators η_t and the space error estimator η_s.
While $\eta_t > \Gamma_{t0}$
$k_{n+1} := \delta_1 k_n$.
$t_{n+1} := t_n + k_{n+1}$.
Solve u_{n+1} using **Algorithm2** on the current mesh $\mathcal{T}_{n+1}$, then calculate the time error estimators η_t and the space error estimator η_s.
End While.
While $\eta_s > \Gamma_s$.

Step 4: **If** $\eta_t <= \Gamma_{t1}$
$k_{n+1} := \delta_2 k_n$.
$t_{n+1} := t_n + k_{n+1}$.
Else
Go to Step 1.
End If.

The coefficient vector $\hat{a}$ can be solved by the following linear system

$$\mathbf{A}^T\mathbf{A}\hat{\mathbf{a}} = \mathbf{A}^T\mathbf{u},$$

where

$$A = \begin{pmatrix} 1 & \xi_0 & \eta_0 \\ 1 & \xi_1 & \eta_1 \\ \vdots & \vdots & \vdots \\ 1 & \xi_n & \eta_n \end{pmatrix}, \quad \text{and} \quad \mathbf{u} = \begin{pmatrix} u(z_0) \\ u(z_1) \\ \vdots \\ u(z_n) \end{pmatrix}.$$

By the linear system, we obtain the recovered gradient

$$G_h u(z) = \nabla p_z = \begin{pmatrix} a_2 \\ a_3 \end{pmatrix} = \frac{1}{h}\begin{pmatrix} \hat{a}_2 \\ \hat{a}_3 \end{pmatrix}.$$

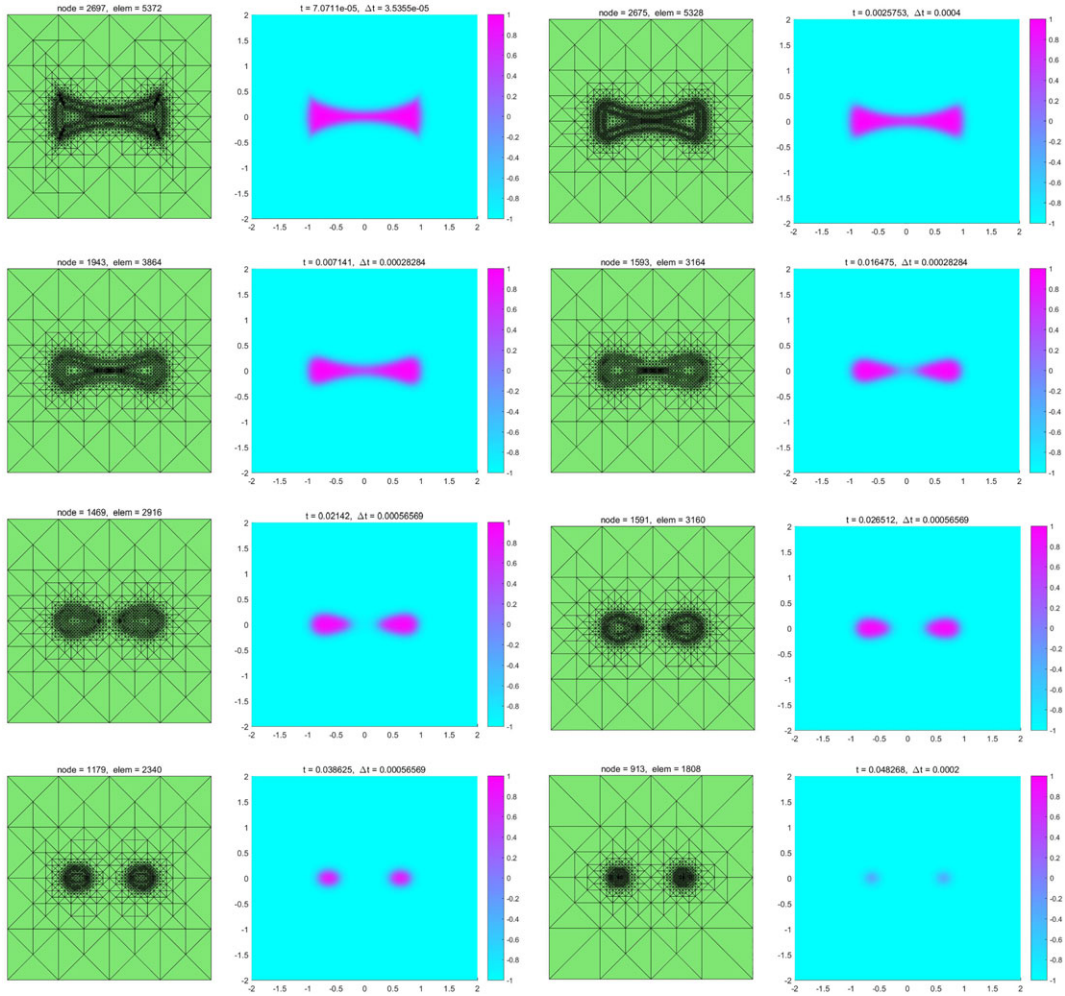

FIGURE 1 Snapshots of the computed solutions and adaptive meshes of Example 1. Here we take $\varepsilon = 1/16$, $\Gamma_{t0} = 0.12$, $\Gamma_{t1} = 0.04$, $\Gamma_s = 0.12$, $\delta_1 = \frac{1}{\sqrt{2}}$, $\delta_2 = \sqrt{2}$, the initial time step is $\Delta t = 10^{-4}$. We can see the initially connected interface splits into two curves; then the two component of the interface develop circular shapes and eventually the diameters of the two particles decrease to zero until they collapse [Color figure can be viewed at wileyonlinelibrary.com]

4 | ADAPTIVE ALGORITHM

Based on SCR recovery technique, we let space error indicator be η_s, defined by

$$\eta_s = \frac{\|\nabla u_n - G_h u_n\|_{0,\Omega}}{\|\nabla u_n\|_{0,\Omega}}, \tag{3.1}$$

and the time error indicator η_t defined as

$$\eta_t = \frac{\|\nabla u_{n+1} - \nabla u_n\|_{0,\Omega}}{\|\nabla u_n\|_{0,\Omega}}. \tag{3.2}$$

And based on maximum tagging strategy, we construct the following adaptive operator splitting algorithm.

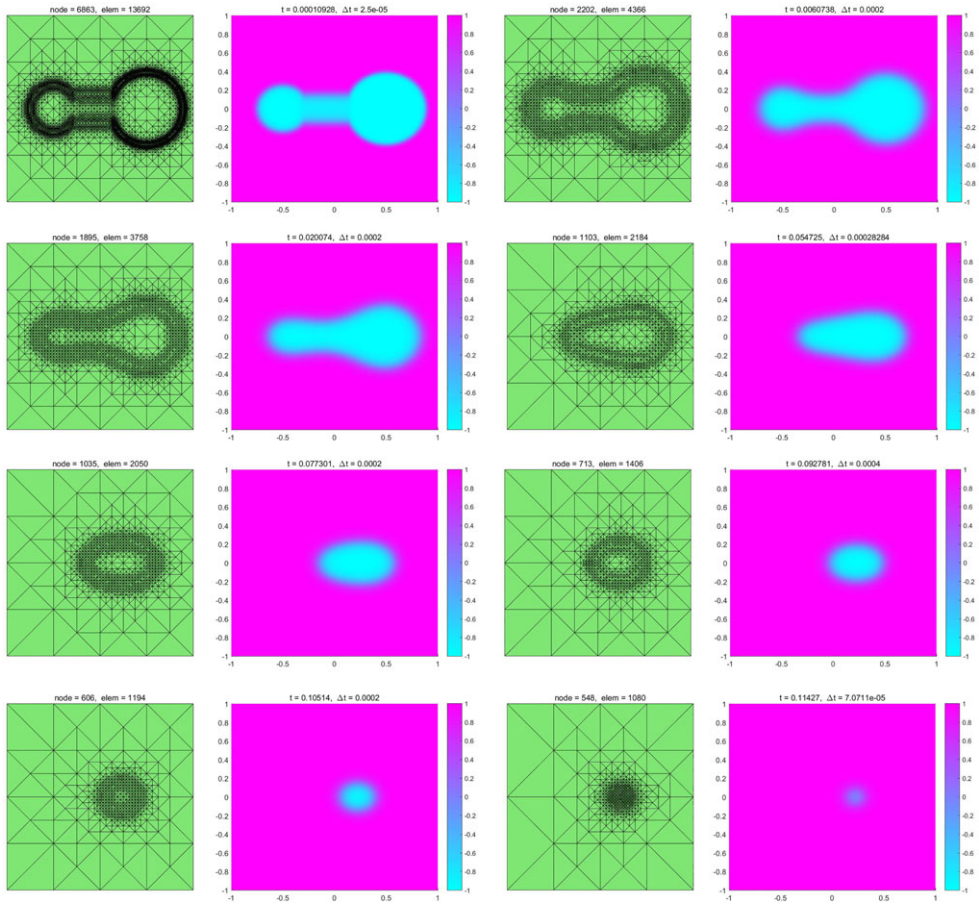

FIGURE 2 We display snapshots of the computed solutions and meshes of Example 2. Here we take $\epsilon = 0.05$, $\Gamma_{t0} = 0.12$, $\Gamma_{t1} = 0.03$, $\Gamma_s = 0.12$, $\delta_1 = \frac{1}{\sqrt{2}}$, $\delta_2 = \sqrt{2}$, the initial time step is $\Delta t = 10^{-4}$. A dumbbell shape with unequal bells are observed at the beginning; it then gradually becomes a similar ellipse and eventually disappears [Color figure can be viewed at wileyonlinelibrary.com]

5 | NUMERICAL RESULTS

In this section, we present several numerical experiments to verify the adaptive algorithm based on SCR *a posteriori* error estimator. These experiments indicate that the algorithm is reliable and efficient for solving the AC equation. For all experiments given in this section, the initial grid is taken to be the linear criss-cross. We implemented the schemes by using the MATLAB© (MathWorks, 1 Apple Hill Drive Natick, MA, USA) software package iFEM [1].

Example 1 Let $\Omega := [-2, 2]^2$, define $m_1 = [0, 2]$, $m_2 = [0, 0]$, $m_3 = [0, -2]$. For given $\epsilon > 0$, let $r_1 = r_3 = 2 - 3\epsilon/2$, $r_2 = 1$ and set $d_j(\mathbf{x}) = |\mathbf{x} - m_j| - r_j$ for $\mathbf{x} \in \Omega$ and $j = 1, 2, 3$, we consider the 2D AC equation with the initial condition

$$u_0(\mathbf{x}) = -\tanh\left(\frac{d(\mathbf{x})}{\sqrt{2}\epsilon}\right), \tag{4.1}$$

where $d(\mathbf{x}) := \max\{-d_1(\mathbf{x}), d_2(\mathbf{x}), -d_3(\mathbf{x})\}$.

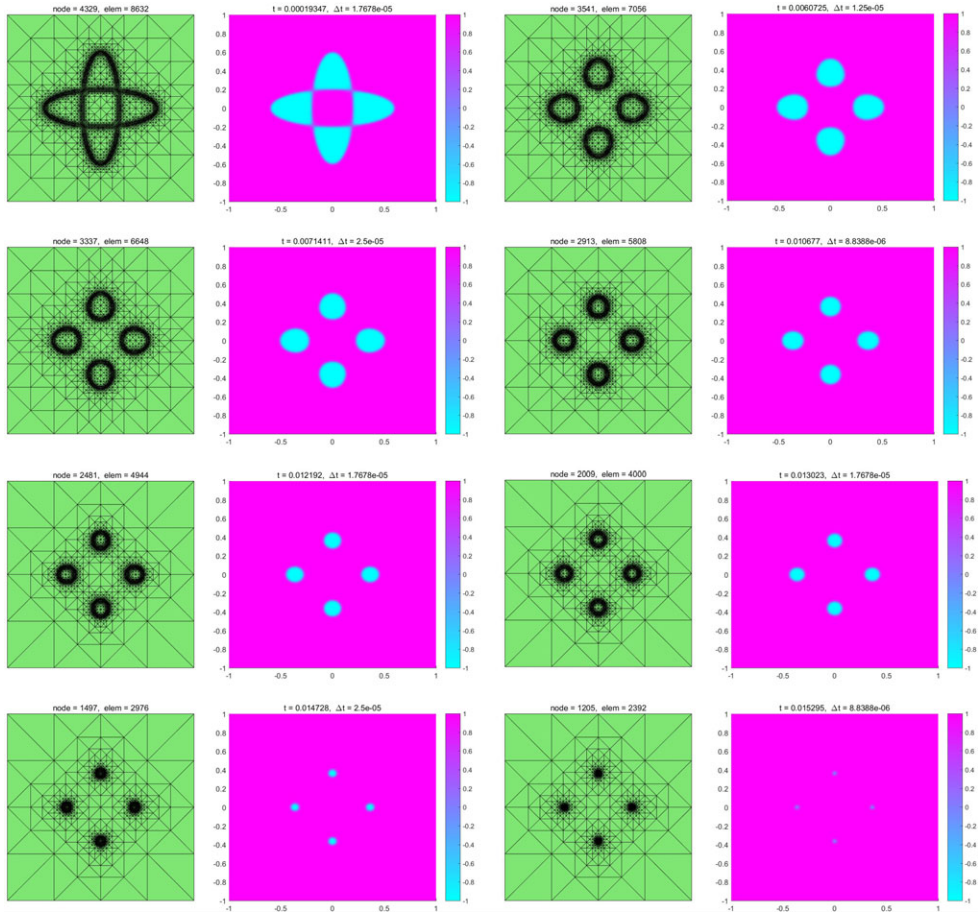

FIGURE 3 Snapshots of the computed solutions and meshes of Example 3. Here we take $\varepsilon = 0.01$, $\Gamma_{t0} = 0.2$, $\Gamma_{t1} = 0.05$, $\Gamma_s = 0.2$, $\delta_1 = \frac{1}{\sqrt{2}}$, $\delta_2 = \sqrt{2}$, the initial time step is $\Delta t = 10^{-4}$. Once again, it shows that the mesh changes as the numerical solution changes. The graph gradually split into four parts, such that the top and bottom are symmetrical, as well as the left and right. They eventually disappear with time evolution [Color figure can be viewed at wileyonlinelibrary.com]

Example 2 We consider as initial condition u_0 a function constituted of a dumbbell shape with unequal bells. For given $\varepsilon > 0$ and $(x, y) \in \Omega := [-1, 1]^2$, let

$$u_0(x,y) = \begin{cases} \tanh\left(\frac{3}{\varepsilon}((x-0.5)^2 + y^2 - (0.39)^2)\right), & x > 0.14, \\ \tanh\left(\frac{3}{\varepsilon}(y^2 - (0.15)^2)\right), & -0.3 \le x \le 0.14, \\ \tanh\left(\frac{3}{\varepsilon}((x+0.5)^2 + y^2 - (0.25)^2)\right), & x < -0.3. \end{cases} \tag{4.2}$$

Example 3 For given $\varepsilon > 0$ and $(x, y) \in \Omega := [-1, 1]^2$, we consider AC equations with the following initial condition

$$u_0(x,y) = \tanh\left(5\left(\frac{x^2}{0.04} + \frac{y^2}{0.36} - 1\right)\left(\frac{x^2}{0.36} + \frac{y^2}{0.04} - 1\right)\right), \quad x > 0.14. \tag{4.3}$$

The evolution of the computed solutions and meshes for three examples are presented in Figures 1–3.

ACKNOWLEDGMENTS

This research work was partially supported by NSFC Key Project 91430213 and 91630205, NSFC Projects 11671340, 11771367 and 11771371, Hunan NSF 2017jj3304, and Hong Kong RGC General Research Fund (GRF) grant 15302518.

ORCID

Jintao Cui https://orcid.org/0000-0001-9912-1889

REFERENCES

[1] L. Chen, iFEM: An integrated finite element methods package in MATLAB. Technical report, University of California at Irvine, 2009.
[2] I. Babuvška, W. C. Rheinboldt, *Error estimates for adaptive finite element computations*, SIAM J. Numer. Anal. vol. 15 (1978) pp. 736–754.
[3] I. Babuška, "*Accuracy estimates and adaptive refinements in finite element computations*," in *John Wiley & Sons*, Hoboken, NJ, 1986.
[4] I. Farago, J. Geiser, *Iterative operator-splitting methods for linear problems*, Int. J. Comput. Sci. Eng. vol. 3 (2005) pp. 255–263.
[5] G. Strang, *On the construction and comparison of difference schemes*, SIAM J. Numer. Anal. vol. 5 (1968) pp. 506–517.
[6] M. Beneš, V. Chalupecký, K. Mikula, *Geometrical image segmentation by the Allen–Cahn equation*, Appl. Numer. Math. vol. 51 (2004) pp. 187–205.
[7] X. F. Chen, C. M. Elliott, A. Cardiner, Z. Jing, *An efficient algorithm for solving the phase field crystal model*, J. Comput. Phys. vol. 227 (2008) pp. 6241–6248.
[8] S. M. Allen, J. W. Cahn, *A microscopic theory for antiphase boundary motion and its application to antiphase domain coarsening*, Acta Metall. vol. 27 (1979) pp. 1085–1095.
[9] X. Feng, A. Prohl, *Numerical analysis of the Allen–Cahn equation and approximation for mean curvature flows*, Numer. Math. vol. 94 (2003) pp. 33–65.
[10] X. Feng, H. Wu, *A posteriori error estimates and an adaptive finite element method for the Allen–Cahn equation and the mean curvature flow*, J. Sci. Comput. vol. 24 (2005) pp. 121–146.
[11] Y. Huang, N. Yi, *The superconvergent cluster recovery method*, J. Sci. Comput. vol. 44 (2010) pp. 301–322.
[12] Y. Li et al., *An unconditionally stable hybrid numerical method for solving the Allen–Cahn equation*, Comput. Math. Appl. vol. 60 (2010) pp. 1591–1606.

How to cite this article: Huang Y, Yang W, Wang H, Cui J. Adaptive operator splitting finite element method for Allen–Cahn equation. *Numer Methods Partial Differential Eq.* 2019;35:1290–1300. https://doi.org/10.1002/num.22350