Immersed Finite Element Methods for Parabolic Equations with Moving Interface

Xiaoming He,[1] **Tao Lin,**[2] **Yanping Lin,**[3,4] **Xu Zhang**[2]

[1] *Department of Mathematics and Statistics, Missouri University of Science and Technology, Rolla, Missouri 65409*

[2] *Department of Mathematics, Virginia Tech, Blacksburg, Virginia 24061*

[3] *Department of Applied Mathematics, Hong Kong Polytechnic University, Hung Hom, Hong Kong*

[4] *Department of Mathematical and Statistics Science, University of Alberta, Edmonton AB, Canada T6G 2G1*

Received 16 February 2011; accepted 5 March 2012
Published online in Wiley Online Library (wileyonlinelibrary.com).
DOI 10.1002/num.21722

This article presents three Crank-Nicolson-type immersed finite element (IFE) methods for solving parabolic equations whose diffusion coefficient is discontinuous across a time dependent interface. These methods can use a fixed mesh because IFEs can handle interface jump conditions without requiring the mesh to be aligned with the interface. These methods will be compared analytically in the sense of accuracy and computational cost. Numerical examples are provided to demonstrate features of these three IFE methods. © 2012 Wiley Periodicals, Inc. Numer Methods Partial Differential Eq 000: 000–000, 2012

Keywords: immersed finite element; moving interface; Crank-Nicolson scheme; Cartesian mesh

I. INTRODUCTION

In this article, we consider the following parabolic equation:

$$\begin{cases} u_t - \nabla \cdot (\beta \nabla u) = f(t, X), & X \in \Omega,\ t \in (0, T_{\text{end}}], \\ u(t, X) = g(t, X), & X \in \partial\Omega,\ t \in (0, T_{\text{end}}], \\ u(0, X) = u_0(X), & X \in \overline{\Omega}, \end{cases} \tag{1.1}$$

Correspondence to: Tao Lin, Department of Mathematics, Virginia Tech, Blacksburg, Virginia 24061 (e-mail: tlin@vt.edu)
Contract grant sponsor: NSF grant; contract grant number: DMS-1016313
Contract grant sponsor: GRF grant of Hong Kong; contract grant number: PolyU 501709
Contract grant sponsor: AMSS-PolyU Joint Research Institute for Engineering and Management of PolyU, NSERC (Canada)
Contract grant sponsor: National Science Foundation of China (10875034)

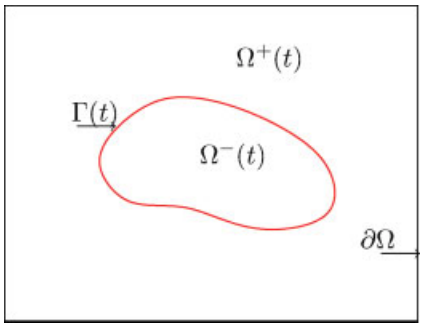

FIG. 1. A sketch of the domain for the moving interface problem. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

with the jump condition on a moving interface $\Gamma(t)$:

$$[u]|_{\Gamma(t)} = 0, \tag{1.2}$$

$$\left[\beta \frac{\partial u}{\partial \mathbf{n}}\right]\Big|_{\Gamma(t)} = 0. \tag{1.3}$$

Without loss of generality, we assume $\Omega \subset \mathbb{R}^r$, $r = 1, 2$, is an open interval or an open rectangular domain, and the interface curve $\Gamma(t)$ is defined by a smooth function $\Gamma : [0, T_{\text{end}}] \to \Omega$. At any time $t \in [0, T_{\text{end}}]$, the interface $\Gamma(t)$ separates Ω into two subdomains $\Omega^+(t)$ and $\Omega^-(t)$ such that $\Omega = \Omega^+(t) \cup \Omega^-(t) \cup \Gamma(t)$, see Fig. 1 for an illustration. The coefficient function $\beta(t, X), X = x$ or $(x, y)^t$, is discontinuous across the interface $\Gamma(t)$. For simplicity, we assume $\beta(t, X)$ is a piece-wise constant function as follows:

$$\beta(t, X) = \begin{cases} \beta^-, & X \in \Omega^-(t), \\ \beta^+, & X \in \Omega^+(t). \end{cases} \tag{1.4}$$

The moving interface problem (1.1)–(1.3) appears in many applications, such as the Stefan problem [1, 2] and the field injection problem [3, 4]. For instance, the Stefan problem is formed by this model problem and additional equations based on physics for tracing the interface. Here, we discuss the model problem in order to focus on the main difficulties arising from the interaction between different time levels in the discretization.

If the interface does not change with respect to time, then conventional finite element methods [5] can solve parabolic interface problems satisfactorily provided that body-fitting meshes are used [6–8]. A body-fitting mesh has to be constructed according to the interface such that each element is essentially on one side of the interface and only touches the interface on its vertices, see the illustration in Fig. 2. Otherwise suboptimal convergence will occur [9]. This restriction can cause the difficulties when applying traditional finite elements to solve moving interface problems, some of them are:

- Whenever the interface changes in the computation, a new mesh has to be generated according to the new location of the moving interface in order to satisfy the body-fitting restriction, which is a time-consuming task in many applications.
- Finite element spaces based body-fitting meshes generated at two time levels often have different degree of freedoms unless extra procedures are employed to keep them the same.

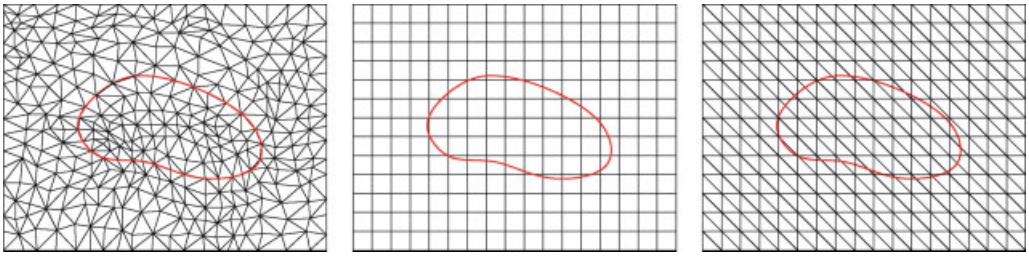

FIG. 2. Left plot: a body-fitting mesh showing how elements are placed along an interface in the standard finite element method. Right plots: Non-body-fitting (rectangular and triangular) Cartesian meshes that allow interface to cut through some elements. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

If the trial and test spaces in a bilinear form of a finite element method are on different time levels, they will have different dimensions; hence, the related matrix in the algebraic system of this finite element method cannot be square, which demands for extra efforts to solve the algebraic system.

- One main feature of finite element computation is the so called "local assembling" idea, by which a matrix in the algebraic system of a finite element method is generated by computing the related quantities locally on each element of a mesh and then assembling these local quantities globally into the matrix. However, when assembling the matrix defined by a bilinear form whose trial and test spaces are on different meshes due to the body-fitting restriction for a moving interface, an element of one mesh may or may not be that of another mesh which essentially makes the "local assembling" idea not applicable. This lack of "local" feature will lead to more complicated programming and can increase computational cost even further by a significant amount.
- Moreover, some of the traditional numerical techniques may become inefficient or even obsolete when body-fitting meshes are used for moving interface problems. Recall that semidiscretization methods solve an initial-boundary value problem of a parabolic equation by forming an ordinary differential equation (ODE) system through the discretization in the spatial variable. However, such an ODE system is not well defined when body-fitting mesh has to be used because of the indefiniteness of the dimension of the finite element space and the indefiniteness of the locations of global degree of freedoms. In particular, this limitation makes popular methods such as the method of lines [10–12], which semidiscretizes the original PDE into a system of ODEs and then solves them by any desired ODE solver, inapplicable due to the lack of "lines."

On the other hand, the advantage of a Cartesian mesh is clearer when the simulations or physical models require structured meshes for interface problems, such as particle-in-cell method for plasma particle simulations [13–16]. It is therefore desirable to develop numerical methods for moving interface problems that can be carried out on a mesh independent of the interface and allow the interface to cut through some elements. Many efforts have been attempted to develop such solvers for interface problems. In the finite difference formulation, the immersed boundary method [17, 18], immersed interface method [19, 20], cut-cell method [21, 22], matched interface and boundary method [23–25] and embedded boundary method [26, 27] have been developed. In the finite element formulation, the newly developed immersed finite element (IFE) methods [28–45] form local basis functions according to the interface jump conditions while their meshes

do not have to be aligned with interfaces. Hence structured (either rectangular or triangular) Cartesian meshes, illustrated by the right plots in Fig. 2, can be used to solve interface problems with nontrivial geometry.

However, most of previous works about IFE methods are for stationary problems. Recently IFE methods were utilized together with the Laplacian transform to treat parabolic equation with a fixed interface [46]. The authors in [47] developed an IFE method in the Eulerian-Lagrangian localized adjoint formulation for solving 1D transient advection-diffusion equations with interfaces. Our goal is to use IFE methods together with the Crank-Nicolson (CN) type discretizations to develop a set of new algorithms for solving parabolic problems with structured meshes independent of moving interfaces so that these new algorithms do not have any of the limitations mentioned above. The rest of the article is organized as follows. In Section II, we derive three CN-IFE algorithms from different types of discretization. In Section III, we compare these algorithms. In Section IV, we discuss implementation issues of these algorithms. In Section V, we use numerical examples to show the optimal convergence of these new CN-IFE algorithms. Brief conclusions will be presented in Section VI.

II. CRANK-NICOLSON-IFE ALGORITHMS

Assume $\mathcal{T}_h = \{T\}$ is a Cartesian mesh (triangular or rectangular) of Ω. Let $\mathcal{T}_h^{i,t}$ and $\mathcal{T}_h^{n,t}$ denote the union of interface elements and non-interface elements at the time t. In the discussion from now on, we assume that $\mathcal{T}_h = \{T\} = \mathcal{T}_h^{i,t} \cup \mathcal{T}_h^{n,t}$ does not change with respect to t while $\mathcal{T}_h^{i,t}$ and $\mathcal{T}_h^{n,t}$ may vary with a moving interface. Define $\mathcal{N}_h$ to be the set of nodes of $\mathcal{T}_h$. Let $\mathcal{N}_h^0$ and $\mathcal{N}_h^b$ be the sets of interior nodes and boundary nodes, respectively. Also, we define $\mathcal{N}_h^{i,t}$ to be the set of nodes of all interface elements at the time t and let $\mathcal{N}_h^{n,t} = \mathcal{N}_h / \mathcal{N}_h^{i,t}$ denote the rest of the nodes. Denote the finite element space $S_{h,t} = \text{span}\{\phi_j^t : X_j \in \mathcal{N}_h\}$ and the test function space $S_{h,t}^0 = \text{span}\{\phi_j^t : X_j \in \mathcal{N}_h^0\}$. Here, $\phi_j^t(X)$ is a global linear or bilinear IFE basis function [28, 32, 37–39, 41] associated with the node $X_j \in \mathcal{N}_h^{i,t}$ at the time t while $\phi_j^t(X)$ is a standard global linear or bilinear finite element basis function for $X_j \in \mathcal{N}_h^{n,t}$. Then we look for an IFE approximate solution to the parabolic interface problem (1.1) – (1.3) in the following form:

$$u_h(t, X) = \sum_{X_j \in \mathcal{N}_h} u_j(t) \phi_j^t(X). \tag{2.1}$$

We will consider two basic discretization procedures for a time-dependent problem. One of them is to discretize the space variable and then the time variable, and the other one carries out the discretization in the opposite order.

A. CN-IFE Algorithm 1

We consider an algorithm in which the spatial discretization is followed by time discretization. If $X_j \in \mathcal{N}_h^{i,t}$, then $\phi_j^t(X)$ depends on the interface location, hence depends on the time t. In contrast, $\phi_j^t(X)$ is independent of the time t for $X_j \in \mathcal{N}_h^{n,t}$. Therefore, we formally have

$$\frac{\partial u_h(t, X)}{\partial t} = \sum_{X_j \in \mathcal{N}_h} \frac{\partial u_j(t)}{\partial t} \phi_j^t(X) + \sum_{X_j \in \mathcal{N}_h^{i,t}} u_j(t) \frac{\partial \phi_j^t(X)}{\partial t}. \tag{2.2}$$

Now we discuss the discretization for the model problem starting from the following standard weak form at a given time t:

$$\int_\Omega v\frac{\partial u}{\partial t}\mathrm{d}X + \int_\Omega \nabla v\cdot(\beta\nabla u)\mathrm{d}X = \int_\Omega vf\,\mathrm{d}X, \forall v\in H_0^1(\Omega),$$

which is equivalent to

$$\sum_{T\in\mathcal{T}_h}\int_T v\frac{\partial u}{\partial t}\mathrm{d}X + \sum_{T\in\mathcal{T}_h}\int_T \nabla v\cdot(\beta\nabla u)\mathrm{d}X = \int_\Omega vf\,\mathrm{d}X, \forall v\in H_0^1(\Omega).$$

Consequently, we can introduce the following spatial discretization: Find $u_h\in S_{h,t}$, such that

$$\sum_{T\in\mathcal{T}_h}\int_T v_h\frac{\partial u_h}{\partial t}\mathrm{d}X + \sum_{T\in\mathcal{T}_h}\int_T \nabla v_h\cdot(\beta\nabla u_h)\mathrm{d}X = \int_\Omega v_h f\,\mathrm{d}X, \forall v_h\in S_{h,t}^0. \tag{2.3}$$

Plugging (2.1) and (2.2) into (2.3), and substituting $\phi_i^t\in S_{h,t}^0$ for v_h, the above formulation becomes: Find the coefficients $u_j(t)$ in $u_h(t,X)=\sum_{X_j\in\mathcal{N}_h}u_j(t)\phi_j^t(X)$ such that

$$\sum_{X_j\in\mathcal{N}_h}u_j'(t)\int_\Omega\phi_i^t\phi_j^t\mathrm{d}X + \sum_{X_j\in\mathcal{N}_{h,t}^i}u_j(t)\int_\Omega\phi_i^t\left(\frac{\partial}{\partial t}\phi_j^t\right)\mathrm{d}X$$

$$+\sum_{X_j\in\mathcal{N}_h}u_j(t)\int_\Omega\beta^t\nabla\phi_i^t\cdot\nabla\phi_j^t\mathrm{d}X = \int_\Omega f\phi_i^t\mathrm{d}X,\qquad \forall\,\phi_i^t\in S_{h,t}^0.$$

In matrix notation, this can be written as

$$M_h(t)\mathbf{u}'(t) + K_h(t)\mathbf{u}(t) + A_h(t)\mathbf{u}(t) = \mathbf{f}(t), \tag{2.4}$$

where

- $M_h(t)=(m_{ij}(t))$ is the mass matrix with $m_{ij}=\int_\Omega\phi_i^t\phi_j^t\mathrm{d}X$.
- $K_h(t)=(k_{ij}(t))$ with $k_{ij}=\int_\Omega\phi_i^t\frac{\partial\phi_j^t}{\partial t}\mathrm{d}X$.
- $A_h(t)=(a_{ij}(t))$ is the stiffness matrix with $a_{ij}=\int_\Omega\nabla\phi_i^t\cdot(\beta\nabla\phi_j^t)\mathrm{d}X$.
- $\mathbf{f}(t)=(f_i(t))$ is the source vector with $f_i(t)=\int_\Omega\phi_i^t f\mathrm{d}X$.
- $\mathbf{u}(t)$ is the vector whose entries are $u_j(t)$, i.e. $\mathbf{u}(t)=(u_j(t))$.

Then, we consider the time discretization. Without loss of generality, we use a uniform partition $0=t_0<t_1<\cdots<t_N=T_{\text{end}}$ in time, where $t_n=n\tau$ with $\tau=T_{\text{end}}/N$. Then we look for approximations such that

$$u_h^n(X)=\sum_{X_j\in\mathcal{N}_h}u_j^n\phi_j^{t_n}(X)\approx u_h(t_n,X).$$

In effect, we initialize the computation by letting $\mathbf{u}^0=(u_0(X_j))$ where $X_j\in\mathcal{N}_h$, and look for solutions $\mathbf{u}^n=(u_j^n)\approx\mathbf{u}(t_n)$, for $n=1,2,\ldots,N$. Then applying the idea of Crank-Nicolson type discretization to (2.4) leads to

$$M_h\left(t_{n+\frac{1}{2}}\right)\frac{\mathbf{u}^{n+1}-\mathbf{u}^n}{\tau} + \left(A_h\left(t_{n+\frac{1}{2}}\right)+K_h\left(t_{n+\frac{1}{2}}\right)\right)\frac{\mathbf{u}^{n+1}+\mathbf{u}^n}{2} = \mathbf{f}\left(t_{n+\frac{1}{2}}\right). \tag{2.5}$$

We introduce the following notations with multiple scripts to describe the evaluations of different functions at different time levels. Let n_v, n_u, n_β, and n_f denote the time levels for the test function v, trial function u, coefficient function β, source function f, respectively. Then, we define the matrices and vectors to be used in the algorithms as follows:

- $M_h^{n_v,n_u} = (m_{ij}^{n_v,n_u})$ is the mass matrix, where $m_{ij}^{n_v,n_u} = \int_\Omega \phi_i^{t_{n_v}} \phi_j^{t_{n_u}} \mathrm{d}X$.
- $A_h^{n_\beta,n_v,n_u} = (a_{ij}^{n_\beta,n_v,n_u})$ is the stiffness matrix, where $a_{ij}^{n_\beta,n_v,n_u} = \int_\Omega \nabla\phi_i^{t_{n_v}} \cdot (\beta^{t_{n_\beta}} \nabla\phi_j^{t_{n_u}})\mathrm{d}X$.
- $K_h^{n_v,n_u} = (k_{ij}^{n_v,n_u})$, where $k_{ij}^{n_v,n_u} = \int_\Omega \phi_i^{t_{n_v}} (\frac{\partial}{\partial t}\phi_j^{t_{n_u}})\mathrm{d}X$.
- $\mathbf{f}^{n_v,n_f} = (f_i^{n_v,n_f})$ is right hand side vector, where $f_i^{n_v,n_f} = \int_\Omega \phi_i^{t_{n_v}} f^{t_{n_f}} \mathrm{d}X$.

With these notations, we can write (2.5) as follows:

$$M_h^{n+\frac{1}{2},n+\frac{1}{2}} \frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\tau} + \left(A_h^{n+\frac{1}{2},n+\frac{1}{2},n+\frac{1}{2}} + K_h^{n+\frac{1}{2},n+\frac{1}{2}} \right) \frac{\mathbf{u}^{n+1} + \mathbf{u}^n}{2} = \mathbf{f}^{n+\frac{1}{2},n+\frac{1}{2}}, \tag{2.6}$$

which leads to the first algorithm: For $\mathbf{u}^n$ given, we find $\mathbf{u}^{n+1}$ by

- CN-IFE Algorithm 1

$$\begin{aligned} &\left(M_h^{n+\frac{1}{2},n+\frac{1}{2}} + \frac{\tau}{2} A_h^{n+\frac{1}{2},n+\frac{1}{2},n+\frac{1}{2}} + \frac{\tau}{2} K_h^{n+\frac{1}{2},n+\frac{1}{2}} \right) \mathbf{u}^{n+1} \\ &\quad = \left(M_h^{n+\frac{1}{2},n+\frac{1}{2}} - \frac{\tau}{2} A_h^{n+\frac{1}{2},n+\frac{1}{2},n+\frac{1}{2}} - \frac{\tau}{2} K_h^{n+\frac{1}{2},n+\frac{1}{2}} \right) \mathbf{u}^n + \tau \mathbf{f}^{n+\frac{1}{2},n+\frac{1}{2}}. \end{aligned}$$

Remark 2.1. Compared with the traditional Crank-Nicolson method for noninterface parabolic problems, CN-IFE Algorithm 1 contains extra terms involving the matrix $K_h^{n+\frac{1}{2},n+\frac{1}{2}}$ due to the moving interface. In Section III, we will analyze this matrix and show that terms involving $K_h^{n+\frac{1}{2},n+\frac{1}{2}}$ are higher order than the others, hence could be dropped for a simpler implementation in some cases.

B. CN-IFE Algorithm 2

Another approach to obtain the fully discretized formulation of a time dependent problem is to discretize in time first, and then in space. In this way, we can avoid terms involving matrix $K_h^{n+\frac{1}{2},n+\frac{1}{2}}$ caused by the time derivative of IFE basis functions.

This procedure can be carried out as follows. Evaluate (1.1) at time $t_{n+\frac{1}{2}}$ to obtain:

$$u_t(t_{n+\frac{1}{2}}, X) - \nabla \cdot \left(\beta \left(t_{n+\frac{1}{2}}, X \right) \nabla u \left(t_{n+\frac{1}{2}}, X \right) \right) = f \left(t_{n+\frac{1}{2}}, X \right). \tag{2.7}$$

Multiplying a test function $v \in H_0^1(\Omega)$ on both side of (2.7), integrating over Ω, applying Green's formula, and using the Crank-Nicolson scheme for time discretization, we obtain

$$\begin{aligned}
&\sum_{T\in\mathcal{T}_h}\int_T v(X)\frac{u(t_{n+1},X)-u(t_n,X)}{\tau}\mathrm{d}X\\
&\quad+\sum_{T\in\mathcal{T}_h}\int_T \nabla v(X)\cdot\left(\beta\left(t_{n+\frac{1}{2}},X\right)\frac{\nabla u(t_{n+1},X)+\nabla u(t_n,X)}{2}\right)\mathrm{d}X\\
&\approx\int_\Omega v(X)f\left(t_{n+\frac{1}{2}},X\right)\mathrm{d}X,\quad \forall v\in H_0^1(\Omega).
\end{aligned}$$

Then, we can introduce the discretization in space to obtain the following full discretization: Assuming $u_h^n \in S_{h,t_n}$ is known, find $u_h^{n+1} \in S_{h,t_{n+1}}$ such that

$$\begin{aligned}
&\sum_{T\in\mathcal{T}_h}\int_T v_h(X)\frac{u_h^{n+1}(X)-u_h^n(X)}{\tau}\mathrm{d}X\\
&\quad+\sum_{T\in\mathcal{T}_h}\int_T \nabla v_h(X)\cdot\left(\beta\left(t_{n+\frac{1}{2}},X\right)\frac{\nabla u_h^{n+1}(X)+\nabla u_h^n(X)}{2}\right)\mathrm{d}X\\
&=\int_\Omega v_h(X)f\left(t_{n+\frac{1}{2}},X\right)\mathrm{d}X,\quad \forall v_h\in S^0_{h,t_{n+\frac{1}{2}}}.
\end{aligned}$$

Thus, in particular by choosing $v_h = \phi_i^{t_{n+\frac{1}{2}}}$, we have

$$\begin{aligned}
&\left(\frac{1}{\tau}\sum_{X_j\in\mathcal{N}_h}\int_\Omega \phi_i^{t_{n+\frac{1}{2}}}\phi_j^{t_{n+1}}\mathrm{d}X+\frac{1}{2}\sum_{X_j\in\mathcal{N}_h}\int_\Omega \nabla\phi_i^{t_{n+\frac{1}{2}}}\cdot\left(\beta^{t_{n+\frac{1}{2}}}\nabla\phi_j^{t_{n+1}}\right)\mathrm{d}X\right)\mathbf{u}^{n+1}\\
&\quad+\left(-\frac{1}{\tau}\sum_{X_j\in\mathcal{N}_h}\int_\Omega \phi_i^{t_{n+\frac{1}{2}}}\phi_j^{t_n}\mathrm{d}X+\frac{1}{2}\sum_{X_j\in\mathcal{N}_h}\int_\Omega \nabla\phi_i^{t_{n+\frac{1}{2}}}\cdot\left(\beta^{t_{n+\frac{1}{2}}}\nabla\phi_j^{t_n}\right)\mathrm{d}X\right)\mathbf{u}^n\\
&=\int_\Omega \phi_i^{t_{n+\frac{1}{2}}}f^{t_{n+\frac{1}{2}}}\mathrm{d}X,\qquad \forall\,\phi_i^{t_{n+\frac{1}{2}}}\in S^0_{h,t_{n+\frac{1}{2}}},
\end{aligned}$$

and this leads to our second algorithm using the matrix notations defined above:

- CN-IFE Algorithm 2

$$\boxed{\left(M_h^{n+\frac{1}{2},n+1}+\frac{\tau}{2}A_h^{n+\frac{1}{2},n+\frac{1}{2},n+1}\right)\mathbf{u}^{n+1}=\left(M_h^{n+\frac{1}{2},n}-\frac{\tau}{2}A_h^{n+\frac{1}{2},n+\frac{1}{2},n}\right)\mathbf{u}^n+\tau\mathbf{f}^{n+\frac{1}{2},n+\frac{1}{2}}.}$$

Remark 2.2. The CN-IFE Algorithm 2 is a natural extension of the classic Crank-Nicolson Algorithm for treating the moving interface. However, the matrices in this algorithm are defined by function values at different time levels.

C. CN-IFE Algorithm 3

Note that IFE functions preserve the continuity of flux on the interface [28, 32, 35, 38, 39, 41]. Therefore we may average the flux instead of the gradient of u to discretize the parabolic equation in the time variable:

$$\frac{u(t_{n+1},X)-u(t_n,X)}{\tau}-\nabla\cdot\left(\frac{\beta(t_{n+1},X)\nabla u(t_{n+1},X)+\beta(t_n,X)\nabla u(t_n,X)}{2}\right)\approx f\left(t_{n+\frac{1}{2}},X\right).$$

Then, we can further discretize the space variable in a similar way to obtain another fully discretization: Find $u_h^{n+1}\in S_{h,t_{n+1}}$ such that

$$\begin{aligned}&\sum_{T\in\mathcal{T}_h}\int_T v_h(X)\frac{u_h^{n+1}(X)-u_h^n(X)}{\tau}\mathrm{d}X\\&\quad+\sum_{T\in\mathcal{T}_h}\int_T\nabla v_h(X)\cdot\left(\frac{\beta(t_{n+1},X)\nabla u_h^{n+1}(X)+\beta(t_n,X)\nabla u_h^n(X)}{2}\right)\mathrm{d}X\\&=\int_\Omega v_h(X)f\left(t_{n+\frac{1}{2}},X\right)\mathrm{d}X,\quad\forall v_h\in S^0_{h,t_{n+\frac{1}{2}}}.\end{aligned}$$

Because $u_h^n(X)=\sum_{X_j\in\mathcal{N}_h}u_j^n\phi_j^{t_n}(X)$, replacing v_h by $\phi_i^{t_{n+\frac{1}{2}}}\in S^0_{h,t_{n+\frac{1}{2}}}$ in the above leads to

$$\begin{aligned}&\left(\frac{1}{\tau}\sum_{X_j\in\mathcal{N}_h}\int_\Omega\phi_i^{t_{n+\frac{1}{2}}}\phi_j^{t_{n+1}}\mathrm{d}X+\frac{1}{2}\sum_{X_j\in\mathcal{N}_h}\int_\Omega\nabla\phi_i^{t_{n+\frac{1}{2}}}\cdot\left(\beta^{t_{n+1}}\nabla\phi_j^{t_{n+1}}\right)\mathrm{d}X\right)\mathbf{u}^{n+1}\\&\quad+\left(-\frac{1}{\tau}\sum_{X_j\in\mathcal{N}_h}\int_\Omega\phi_i^{t_{n+\frac{1}{2}}}\phi_j^{t_n}\mathrm{d}X+\frac{1}{2}\sum_{X_j\in\mathcal{N}_h}\int_\Omega\nabla\phi_i^{t_{n+\frac{1}{2}}}\cdot\left(\beta^{t_n}\nabla\phi_j^{t_n}\right)\mathrm{d}X\right)\mathbf{u}^n\\&=\int_\Omega\phi_i^{t_{n+\frac{1}{2}}}f^{t_{n+\frac{1}{2}}}\mathrm{d}X,\qquad\forall\,\phi_i^{t_{n+\frac{1}{2}}}\in S^0_{h,t_{n+\frac{1}{2}}}.\end{aligned}$$

Writing this in matrix form, we have our third algorithm:

- CN-IFE Algorithm 3

$$\boxed{\left(M_h^{n+\frac{1}{2},n+1}+\frac{\tau}{2}A_h^{n+1,n+\frac{1}{2},n+1}\right)\mathbf{u}^{n+1}=\left(M_h^{n+\frac{1}{2},n}-\frac{\tau}{2}A_h^{n,n+\frac{1}{2},n}\right)\mathbf{u}^n+\tau\mathbf{f}^{n+\frac{1}{2},n+\frac{1}{2}}.}$$

Remark 2.3. In the derivation of CN-IFE Algorithm 3, we have used the average of the flux $\beta\nabla u$ rather than the gradient of u in the discretization. In this configuration, the actual flux is preserved at each time level, which is consistent with the main idea of using IFEs. However, this algorithm is different from the classic Crank-Nicolson scheme since it replaces the exact coefficient $\beta^{n+\frac{1}{2}}$ by β^n and β^{n+1} at different time levels. We note that this coefficient replacement seems to cause the CN-IFE Algorithm 3 to be conditionally stable, i.e., this algorithm has to use a small time step in order to produce convergent numerical solutions especially when the diffusion coefficient has a large jump.

III. COMPARISON OF THE THREE ALGORITHMS

In this section, we derive estimates which can be used to compare the three algorithms derived in the previous section. In particular, we will show that the terms involving matrix $K_h^{n+\frac{1}{2},n+\frac{1}{2}}$ are relatively "small". Then we will discuss some computational features of these algorithms.

A. Estimates for Matrices

Compared with CN-IFE Algorithms 2 and 3, CN-IFE Algorithm 1 utilizes an additional matrix $K_h^{n+\frac{1}{2},n+\frac{1}{2}}$ due to the time derivative of IFE basis functions. Because this matrix does not appear in other two algorithms and the original CN scheme, it is desirable to know how significant this matrix is. For this purpose, we will derive estimates for matrices in the algorithms in 1D and 2D cases, respectively. In the following derivation, a generic constant notation C will be frequently used whose value might be different from place to place.

One-Dimensional Case. In 1D case, the interface curve $\Gamma(t)$ degenerates into a moving point $\alpha(t)$. Assume that the IFE space based on linear polynomials is used [28, 37]. Note that there is only one interface element at a time, and on the interface element T, there are two local basis functions $\phi_{1,T}^t(x)$ and $\phi_{2,T}^t(x)$ satisfying the jump conditions and the nodal restrictions. The restriction of a global basis $\phi^t(x)$ on an interface element T, i.e., $\phi_T^t(x) = \phi^t(x)|_T$ should be one of these two local basis functions. The following lemma gives estimates for these local basis functions.

Lemma 3.1. *Assume $\mathcal{T}_h$ is a uniform partition of $\Omega = (a, b)$, and $\phi_T^t(x)$ is one of the 1D local linear IFE basis functions on the interface element $T \in \mathcal{T}_h$ at the time t. If there exists a positive M such that $|\alpha'(t)| \le M$ for every $t \in [0, T_{\text{end}}]$, then there exists a constant C independent of h and interface such that*

$$\left|\phi_T^t(x)\right| \le C\,, \qquad \left|\frac{\partial \phi_T^t(x)}{\partial x}\right| \le Ch^{-1}\,, \qquad \left|\frac{\partial \phi_T^t(x)}{\partial t}\right| \le Ch^{-1}, \qquad \forall x \in T. \tag{3.1}$$

Proof. We verify the desired properties for the IFE local basis function $\phi_{1,T}^t$ and similar arguments can be applied for $\phi_{2,T}^t$; hence they are omitted. Without loss of generality, we assume $T = (0, h)$ is the interface element at time t, with the interface point $\alpha = \alpha(t)$ such that $0 < \alpha < h$. Denote $T^- = (0, \alpha)$, and $T^+ = (\alpha, h)$. From [28, 37], $\phi_{1,T}^t$ is defined piece-wisely as follows

$$\phi_{1,T}^t(x) = \phi_{1,T}(x, \alpha(t)) = \begin{cases} -\beta^+ \Lambda x + 1, & x \in T^-, \\ \beta^- \Lambda (h - x), & x \in T^+, \end{cases} \tag{3.2}$$

where

$$\Lambda = \frac{1}{\alpha\beta^+ + (h - \alpha)\beta^-} \le \frac{1}{\min\{\beta^-, \beta^+\}} h^{-1} \le Ch^{-1}. \tag{3.3}$$

Combining (3.2) and (3.3) gives the estimate $|\phi_{1,T}^t(x)| \le C$. Taking the x derivative of $\phi_{1,T}^t$, we have

$$\frac{\partial \phi_{1,T}^t(x)}{\partial x} = \begin{cases} -\beta^+ \Lambda, & x \in T^-, \\ -\beta^- \Lambda, & x \in T^+. \end{cases}$$

This confirms $|\frac{\partial \phi_{1,T}^t(x)}{\partial x}| \le Ch^{-1}$ because of (3.3). Note that the interface is moving, i.e. $\alpha = \alpha(t)$. Therefore,

$$\frac{\partial \phi_{1,T}^t(x)}{\partial t} = \frac{\partial \phi_{1,T}^t(x)}{\partial \alpha}\alpha'(t) = \begin{cases} \beta^+(\beta^+ - \beta^-)\Lambda^2 x\alpha'(t), & x \in T^-, \\ -\beta^-(\beta^+ - \beta^-)\Lambda^2(h-x)\alpha'(t), & x \in T^+. \end{cases} \tag{3.4}$$

Together with the assumption $|\alpha'(t)| \le M$, we obtain the last estimate in (3.1):

$$\left|\frac{\partial \phi_{1,T}^t(x)}{\partial t}\right| \le Ch^{-1}.$$

■

Remark 3.1. We recall that the local IFE basis functions are consistent with the standard FE basis function in the sense that the local IFE basis functions become the FE basis function when $\beta^- = \beta^+$. The time derivative of a local FE function should obviously be zero and this property is still preserved by the local IFE basis functions, and can be confirmed by simply set $\beta^- = \beta^+$ in (3.4).

The following theorem provides estimates for matrices in CN-IFE Algorithm 1 which confirms the fact that under certain assumptions the terms involving the matrix $K_h^{n+\frac{1}{2},n+\frac{1}{2}}$ in CN-IFE Algorithm 1 is of higher order than those involving the stiffness and mass matrices in one dimensional case.

Theorem 3.1. *Assume $\mathcal{T}_h$ is a uniform partition of (a,b), and $0 = t_0 < t_1 < \cdots < t_N = T_{\text{end}}$ is a uniform partition in time with $\tau = T_{\text{end}}/N$. If there exists a positive M such that $|\alpha'(t)| \le M$ for every $t \in [0, T_{\text{end}}]$, then there exists a constant C independent of h and interface such that for $n = 0, 1, \ldots, N-1$ we have*

$$\left\| A_h^{n+\frac{1}{2},n+\frac{1}{2},n+\frac{1}{2}} \right\|_F \le Ch^{-\frac{3}{2}}, \quad \left\| M_h^{n+\frac{1}{2},n+\frac{1}{2}} \right\|_F \le Ch^{\frac{1}{2}}, \quad \left\| K_h^{n+\frac{1}{2},n+\frac{1}{2}} \right\|_F \le C, \tag{3.5}$$

where $\|\cdot\|_F$ is the Frobenius norm of a matrix. Assume further $\tau = O(h)$. Then the coefficient matrices in Algorithm 1 satisfy

$$\left\| M_h^{n+\frac{1}{2},n+\frac{1}{2}} + \frac{\tau}{2} A_h^{n+\frac{1}{2},n+\frac{1}{2},n+\frac{1}{2}} \right\|_F \le Ch^{-\frac{1}{2}}, \qquad \left\| \frac{\tau}{2} K_h^{n+\frac{1}{2},n+\frac{1}{2}} \right\|_F \le Ch, \tag{3.6}$$

for all $n = 0, 1, \ldots, N-1$.

Proof. For the stiffness matrix $A_h^{n+\frac{1}{2},n+\frac{1}{2},n+\frac{1}{2}} = (a_{ij}^{n+\frac{1}{2},n+\frac{1}{2},n+\frac{1}{2}})$, assume a_{ij} is a non-zero entry, then by definition,

$$a_{ij}^{n+\frac{1}{2},n+\frac{1}{2},n+\frac{1}{2}} = \int_{\Omega'} \beta^{t_{n+\frac{1}{2}}} \frac{\partial \phi_i^{n+\frac{1}{2}}}{\partial x} \frac{\partial \phi_j^{n+\frac{1}{2}}}{\partial x} \mathrm{d}x,$$

where $\Omega' = \text{supp}(\phi_i^t) \cap \text{supp}(\phi_j^t) = O(h)$ since the IFE basis functions are locally supported. Note that the coefficient function β is a piecewise constant function such that it can be bounded by a positive constant. Using Lemma 3.1, we have

$$\left| a_{ij}^{n+\frac{1}{2},n+\frac{1}{2},n+\frac{1}{2}} \right| \le C \int_{\Omega'} h^{-1} h^{-1} \mathrm{d}x \le Ch^{-1}.$$

Note that the number of nonzero entries in the stiffness matrix $A_h^{n+\frac{1}{2},n+\frac{1}{2},n+\frac{1}{2}}$ is $O(h^{-1})$ because of the sparsity of the global finite element matrix. Then

$$\left\| A_h^{n+\frac{1}{2},n+\frac{1}{2},n+\frac{1}{2}} \right\|_F = \sqrt{\sum_{i,j} \left(a_{ij}^{n+\frac{1}{2},n+\frac{1}{2},n+\frac{1}{2}} \right)^2} \leq Ch^{-\frac{3}{2}}.$$

Similar arguments can be applied to the mass matrix $M_h^{n+\frac{1}{2},n+\frac{1}{2}}$ to derive following estimate:

$$\left\| M_h^{n+\frac{1}{2},n+\frac{1}{2}} \right\|_F \leq Ch^{\frac{1}{2}}.$$

For the matrix $K_h^{n+\frac{1}{2},n+\frac{1}{2}}$, note that the number of nonzero entries is only $O(1)$ because there is only one interface element at a time. Then applying the result of Lemma 3.1 and the same argument above, we can obtain

$$\left\| K_h^{n+\frac{1}{2},n+\frac{1}{2}} \right\|_F \leq C.$$

In addition, if $\tau = O(h)$, then using the estimates above, we have

$$\left\| M_h^{n+\frac{1}{2},n+\frac{1}{2}} + \frac{\tau}{2} A_h^{n+\frac{1}{2},n+\frac{1}{2},n+\frac{1}{2}} \right\|_F \leq Ch^{\frac{1}{2}} + Ch^{-\frac{1}{2}} \leq Ch^{-\frac{1}{2}},$$

when h is small enough. Similarly, we also have

$$\left\| \frac{\tau}{2} K_h^{n+\frac{1}{2},n+\frac{1}{2}} \right\|_F \leq Ch.$$

■

Two-Dimensional Case. Now, we derive estimates for the matrices appearing in the 2D CN-IFE Algorithm 1 on a Cartesian triangular mesh (as in the right plot of Fig. 2) for the linear IFE functions [38,39]. Similar results can be obtained for bilinear IFE functions on a rectangular mesh [32,41].

Assume $\Gamma(t,x,y) = 0$ is a moving curve in the domain $\Omega \subset \mathbb{R}^2$. Without loss generality, we consider an interface element $T = \triangle A_1A_2A_3$ with vertices

$$A_1(x_1,y_1), \quad A_2(x_2,y_2), \quad A_3(x_3,y_3),$$

cut through by the interface Γ at time t with the following two intersection points $D(t)$ and $E(t)$ (see Fig. 3):

$$D(x_D,y_D) \in \overline{A_1A_3}, \quad E(x_E,y_E) \in \overline{A_1A_2}.$$

Let $D(t) = A_1 + d(t)(A_3 - A_1)$ and $E(t) = A_1 + e(t)(A_2 - A_1)$ with $0 < d(t) < 1$, and $0 < e(t) < 1$. Note that a local IFE nodal basis function on T can be written in the following form:

$$\phi_T^t(X) = \begin{cases} \phi_T^{t-}(X) = v_1\psi_1(X) + c_2(t)\psi_2(X) + c_3(t)\psi_3(X), & X \in T^-(t), \\ \phi_T^{t+}(X) = c_1(t)\psi_1(X) + v_2\psi_2(X) + v_3\psi_3(X), & X \in T^+(t), \end{cases} \tag{3.7}$$

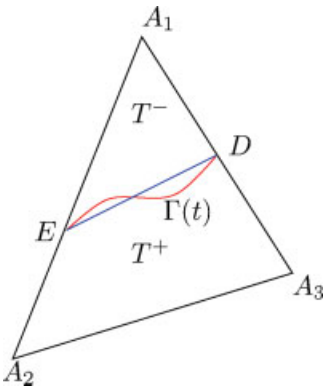

FIG. 3. A sketch of local interface triangle. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

where v_1, v_2, and v_3 are nodal values at the vertices A_1, A_2, and A_3 of T, respectively, and $\psi_i, i = 1, 2, 3$ are the standard linear finite element nodal basis functions satisfying

$$\psi_i(A_j) = \delta_{ij}, 1 \le i, j \le 3.$$

In a Cartesian triangular mesh, an interface element must be one of the six types displayed in Fig. 4.

Lemma 3.2. *Assume $T = \triangle A_1A_2A_3$ is an interface element at the time t with interface intersection points $D(x_D, y_D)$, $E(x_E, y_E)$. Let $\phi_T^t(x, y)$ be one of the linear IFE local basis functions defined by* (3.7)*. If there exists a constant M such that $\|D'(t)\| \le M$ and $\|E'(t)\| \le M$, then there exists a constant C, independent of h and interface, such that the following estimates hold for all $(x, y)^t \in T$:*

$$\left|\phi_T^t(x, y)\right| \le C\,, \qquad \left|\nabla\phi_T^t(x, y)\right| \le Ch^{-1}\,, \qquad \left|\frac{\partial\phi_T^t(x, y)}{\partial t}\right| \le Ch^{-1}.$$

Proof. Without loss of generality, we derive these estimates for local IFE basis functions on a type 1 interface element illustrated in Fig. 4. Estimates on interface elements of other types can be obtained using similar arguments. Also, we assume $x_1 = y_1 = 0$. Hence, the interface element T has the vertices $A_1(0, 0)$, $A_2(h, 0)$, $A_3(0, h)$. We consider the linear basis function $\phi_{1,T}^t$ in detail and the discussion for the other two basis functions can be carried out similarly. Note that $\phi_{1,T}^t$ satisfies the nodal value constrains

$$\phi_{1,T}^t(A_1) = 1, \quad \phi_{1,T}^t(A_2) = 0, \quad \phi_{1,T}^t(A_3) = 0.$$

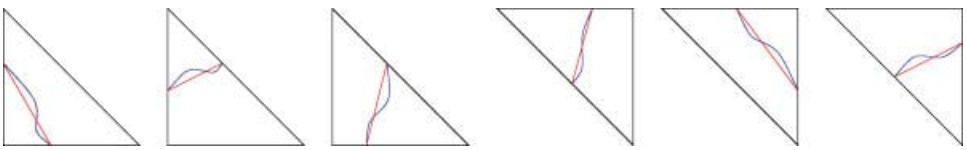

FIG. 4. Different types of interface triangles: curves represent actual interfaces and line segments are used to approximate actual interfaces. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

Together with the jump conditions, we have [38, 39]:

$$\phi_{1,T}^{t}(x,y)=\begin{cases}\phi_{1,T}^{t-}(x,y)=\psi_1(x,y)+c_2(t)\psi_2(x,y)+c_3(t)\psi_3(x,y), & (x,y)\in T^-(t),\\ \phi_{1,T}^{t+}(x,y)=c_1(t)\psi_1(x,y), & (x,y)\in T^+(t),\end{cases}$$

where

$$c_1=\frac{\beta^-(d^2+e^2)}{\Lambda},\quad c_2=\frac{(\beta^--\beta^+)d(d+e)(1-e)}{\Lambda},\quad c_3=\frac{(\beta^--\beta^+)e(d+e)(1-d)}{\Lambda},$$

with

$$\Lambda=\beta^+de(d+e)+\beta^-(d^2(1-e)+e^2(1-d)),$$

and ψ_i, $i=1,2,3$ are the standard finite element nodal basis functions, i.e.,

$$\psi_1=\frac{1}{h}(h-x-y),\quad \psi_2=\frac{x}{h},\quad \psi_3=\frac{y}{h}.$$

It is obvious that $x, y, x+y\in[0,h]$ given $(x,y)\in T$. Therefore,

$$|\psi_i(x,y)|\le 1,\left|\frac{\partial}{\partial x}\psi_i(x,y)\right|\le\frac{1}{h},\left|\frac{\partial}{\partial y}\psi_i(x,y)\right|\le\frac{1}{h},\quad i=1,2,3. \tag{3.8}$$

Also note that

$$|\Lambda|\ge\begin{cases}\beta^-d^2(1-e)\ge\frac{\beta^-}{8}, & \text{if}\quad 0\le e\le\frac{1}{2}\le d\le 1,\\ \beta^-e^2(1-d)\ge\frac{\beta^-}{8}, & \text{if}\quad 0\le d\le\frac{1}{2}\le e\le 1,\\ \beta^+de(d+e)\ge\frac{\beta^+}{4}, & \text{if}\quad \frac{1}{2}\le d\le 1,\frac{1}{2}\le e\le 1.\end{cases}$$

Hence $|\Lambda|\ge\frac{\min\{\beta^-,\beta^+\}}{8}$, if either d or e is greater than $\frac{1}{2}$. Therefore, we can estimate c_i, $i=1,2,3$ as follows:

- If $0\le d\le\frac{1}{2}$ and $0\le e\le\frac{1}{2}$

$$\begin{aligned}|c_1|&\le\frac{\beta^-(d^2+e^2)}{\beta^-(d^2(1-e)+e^2(1-d))}\le\frac{\beta^-(d^2+e^2)}{\frac{1}{2}\beta^-(d^2+e^2)}=2,\\ |c_2|&\le\frac{|\beta^--\beta^+|d(d+e)(1-e)}{\beta^-(d^2(1-e)+e^2(1-d))}\le\frac{|\beta^--\beta^+|d(d+e)}{\beta^-\frac{1}{2}(d^2+e^2)}\\ &\le\frac{2|\beta^--\beta^+|}{\beta^-}\left(\frac{d^2}{d^2+e^2}+\frac{de}{d^2+e^2}\right)\le\frac{3|\beta^--\beta^+|}{\beta^-},\\ |c_3|&\le\frac{|\beta^--\beta^+|e(d+e)(1-d)}{\beta^-(d^2(1-e)+e^2(1-d))}\le\frac{|\beta^--\beta^+|e(d+e)}{\beta^-\frac{1}{2}(d^2+e^2)}\\ &\le\frac{2|\beta^--\beta^+|}{\beta^-}\left(\frac{de}{d^2+e^2}+\frac{e^2}{d^2+e^2}\right)\le\frac{3|\beta^--\beta^+|}{\beta^-}.\end{aligned}$$

- Otherwise

$$|c_1| \le \frac{2\beta^-}{\frac{1}{8}\min\{\beta^-,\beta^+\}} = \frac{16\beta^-}{\min\{\beta^-,\beta^+\}},$$
$$|c_2| \le \frac{2|\beta^- - \beta^+|}{\frac{1}{8}\min\{\beta^-,\beta^+\}} = \frac{16|\beta^- - \beta^+|}{\min\{\beta^-,\beta^+\}},$$
$$|c_3| \le \frac{2|\beta^- - \beta^+|}{\frac{1}{8}\min\{\beta^-,\beta^+\}} = \frac{16|\beta^- - \beta^+|}{\min\{\beta^-,\beta^+\}}.$$

Combining these results with (3.8) we have

$$\left|\phi^t_{1,T}(x,y)\right| \le C, \quad \left|\nabla\phi^t_{1,T}(x,y)\right| \le Ch^{-1}.$$

For the time derivative of the IFE basis $\phi^t_{1,T}$, we have

$$\frac{\partial\phi^t_{1,T}(x,y)}{\partial t} = \frac{\partial\phi^t_{1,T}(x,y)}{\partial d}d'(t) + \frac{\partial\phi^t_{1,T}(x,y)}{\partial e}e'(t). \tag{3.9}$$

By direct calculations, we have

$$\begin{aligned}\frac{\partial\phi^{t-}_{1,T}}{\partial d} &= \frac{(\beta^+-\beta^-)e}{\Lambda^2 h}(\beta^+e(d+e)^2y \\ &\quad + \beta^-(2de((e-1)x+y) + e^2((e-1)x-y) + d^2((1-e)x+(1-2e)y))),\\ \frac{\partial\phi^{t+}_{1,T}}{\partial d} &= \frac{-\beta^-(\beta^+-\beta^-)e^2}{\Lambda^2 h}(-d^2+2de+e^2)(h-x-y).\end{aligned}$$

Because we have

$$\begin{aligned}\left|\frac{\partial\phi^{t-}_{1,T}}{\partial d}\right| &\le \frac{\left|\beta^+-\beta^-\right|e}{\Lambda^2 h}(\beta^+e(d+e)^2dh + \beta^-(2de(he+hd) + e^2(he+hd) + d^2(he+hd)))\\ &\le \frac{2|\beta^+-\beta^-|(\beta^++\beta^-)e}{\Lambda^2}(d+e)^3,\\ \left|\frac{\partial\phi^{t+}_{1,T}}{\partial d}\right| &\le \frac{\beta^-\left|\beta^+-\beta^-\right|e^2}{\Lambda^2}(d+e)^2,\end{aligned}$$

then

$$\left|\frac{\partial\phi^t_{1,T}}{\partial d}\right| \le \frac{2|\beta^+-\beta^-|(\beta^++\beta^-)}{\Lambda^2}(d+e)^4,$$

which leads to

- If $0 \le d \le \frac{1}{2}$ and $0 \le e \le \frac{1}{2}$

$$\left|\frac{\partial\phi^t_{1,T}}{\partial d}\right| \le \frac{2|\beta^+-\beta^-|(\beta^++\beta^-)}{\left(\frac{1}{2}\beta^-(d^2+e^2)\right)^2}(d+e)^4 \le \frac{16|\beta^{+2}-\beta^{-2}|}{\beta^{-2}}. \tag{3.10}$$

- Otherwise

$$\left|\frac{\partial \phi_{1,T}^{t}}{\partial d}\right| \leq \frac{2|\beta^{+}-\beta^{-}|(\beta^{+}+\beta^{-})16}{\min\{\beta^{-},\beta^{+}\}^{2}\frac{1}{64}} \leq \frac{2048|\beta^{+2}-\beta^{-2}|}{\min\{\beta^{-},\beta^{+}\}^{2}}. \tag{3.11}$$

Similar arguments can be used to show that the right hand sides of (3.10) and (3.11) can bound $\frac{\partial \phi_{1,T}^{t}}{\partial e}$ which has the following formulas:

$$\begin{aligned}\frac{\partial \phi_{1,T}^{t-}}{\partial e} &= \frac{(\beta^{+}-\beta^{-})d}{\Lambda^{2}h}((\beta^{+}d(d+e)^{2}x \\ &\quad + \beta^{-}(2de((d-1)y+x)+d^{2}((d-1)y-x)+e^{2}((1-d)y+(1-2d)x))),\\ \frac{\partial \phi_{1,T}^{t+}}{\partial e} &= \frac{-\beta^{-}(\beta^{+}-\beta^{-})d^{2}}{\Lambda^{2}h}(d^{2}+2de-e^{2})(h-x-y).\end{aligned}$$

The assumptions $\|D'(t)\| \leq M$ and $\|E'(t)\| \leq M$ imply

$$|hd'(t)| \leq M \quad \text{and} \quad |he'(t)| \leq M, \tag{3.12}$$

which lead to

$$|d'(t)| \leq Mh^{-1} \quad \text{and} \quad |e'(t)| \leq Mh^{-1}. \tag{3.13}$$

Combining (3.9), (3.10), and (3.11), and letting

$$C = 2M \max\left\{\frac{16|\beta^{+2}-\beta^{-2}|}{\beta^{-2}}, \frac{2048|\beta^{+2}-\beta^{-2}|}{\min\{\beta^{-},\beta^{+}\}^{2}}\right\},$$

we finally obtain the last estimate:

$$\left|\frac{\partial \phi_{1,T}^{t}(x,y)}{\partial t}\right| \leq Ch^{-1}.$$

■

Then, using the estimates in Lemma 3.2 and arguments similar to those in the proof of Theorem 3.1 we can obtain estimates for the matrices in the 2D CN-IFE Algorithm 1.

Theorem 3.2. *Assume $\mathcal{T}_h$ is a Cartesian mesh of a rectangular domain Ω, and $0 = t_0 < t_1 < \cdots < t_N = T_{\text{end}}$ is a uniform partition in time with $\tau = T_{\text{end}}/N$. If there exists a positive M such that $\|D'(t)\| \leq M$ and $\|E'(t)\| \leq M$ for every interface element at the time $t \in [0, T_{\text{end}}]$, then there exists a constant C independent of h and interface such that for all $n = 0, 1, \ldots, N-1$ we have*

$$\left\|A_h^{n+\frac{1}{2},n+\frac{1}{2},n+\frac{1}{2}}\right\|_F \leq Ch^{-1}, \quad \left\|M_h^{n+\frac{1}{2},n+\frac{1}{2}}\right\|_F \leq Ch, \quad \left\|K_h^{n+\frac{1}{2},n+\frac{1}{2}}\right\|_F \leq Ch^{\frac{1}{2}},$$

where $\|\cdot\|_F$ is the Frobenius norm of a matrix. Assume further $\tau = O(h)$, then the coefficient matrices in Algorithm 1 satisfy

$$\left\|M_h^{n+\frac{1}{2},n+\frac{1}{2}} + \frac{\tau}{2}A_h^{n+\frac{1}{2},n+\frac{1}{2},n+\frac{1}{2}}\right\|_F \leq C, \qquad \left\|\frac{\tau}{2}K_h^{n+\frac{1}{2},n+\frac{1}{2}}\right\|_F \leq Ch^{\frac{3}{2}},$$

for all $n = 0, 1, \ldots, N-1$.

TABLE I. Comparison of three algorithms in one time step.

Algorithms	CN-IFE A1	CN-IFE A2	CN-IFE A3
Time level for coefficient function	$n+\frac{1}{2}$	$n+\frac{1}{2}$	n and $n+1$
Time level for trial function	$n+\frac{1}{2}$	n and $n+1$	n and $n+1$
Time level for test function	$n+\frac{1}{2}$	$n+\frac{1}{2}$	$n+\frac{1}{2}$
Time level for RHS function	$n+\frac{1}{2}$	$n+\frac{1}{2}$	$n+\frac{1}{2}$
Matrices to be assembled	$2+\epsilon$	4	4
Works well for large time step	✓	✓	×
Works well for large coefficient jump	✓	✓	×

B. Comparison from the Computational Perspective

In this section, we discuss the three algorithms developed in the previous section from the viewpoints of their differences and similarities.

First of all, we would like to point out that CN-IFE Algorithm 1 has obvious advantages in the sense of easy implementation and efficiency. Because the matrices in this algorithm are to be assembled at the same time level, the standard IFE matrix assembler developed for time independent interface problems can be easily adopted for assembling all of them. Moreover, matrix $K_h^{n+\frac{1}{2},n+\frac{1}{2}}$ can be generated by looping over interface elements only. Because the number of interface elements is much less than that of noninterface elements, the cost to assemble $K_h^{n+\frac{1}{2},n+\frac{1}{2}}$ is significantly less than other matrices. Overall, we only need to generate "$2+\epsilon$" matrices in order to apply CN-IFE Algorithm 1 to solve a parabolic problem with a moving interface in contrast with four matrices required by CN-IFE Algorithm 2 or 3. This significantly reduces the cost of assembling matrices and makes CN-IFE Algorithm 1 more efficient than other two.

CN-IFE Algorithm 2 is a natural extension of the classic Crank-Nicolson scheme for parabolic problems with moving interface, in which all the given functions are evaluated exactly at the middle time level. Numerical experiments (see Section V) indicate that it has the usual rate of convergence and preserves the unconditional stability of the classic Crank-Nicolson scheme.

CN-IFE Algorithm 3 seems to have conditional stability due to the modification of the traditional Crank-Nicolson algorithm, which substitutes β^{n+1} and β^n for $\beta^{n+\frac{1}{2}}$ at t_{n+1} and t_n, respectively. Numerical results produced by this algorithm may deteriorate if the jump in coefficient or the time step is large since the related matrices may vary greatly. Hence a smaller time step size may be needed in order to obtain an optimal convergence rate.

The matrix $K_h^{n+\frac{1}{2},n+\frac{1}{2}}$ will disappear if we are solving a parabolic problem without interface, or if the interface is fixed since the IFE/FE basis functions will not depend on time in these cases. Moreover all of these three algorithms are also related with each other in the sense of the remark below.

Remark 3.2. All of the three algorithms become the same, i.e., the classic Crank-Nicolson Algorithm, if there is no interface, or if the interface is time independent.

Finally we summarize the major differences among the three algorithms in Table I based on the results in Sections II and V.

IV. SOME IMPLEMENTATION ISSUES

In this section, we will discuss implementation issues for the three CN-IFE Algorithms for solving parabolic moving interface problems.

A. Local Matrices Assembling

At each time level, the process of assembling local matrices into global matrices is identical to that of the standard finite element computation. A standard finite element assembler can be used to generate local matrices over all the noninterface elements. The crux of implementing these three algorithms is to generate local matrices on the interface elements at every time step.

Local Matrices for CN-IFE Algorithm 1. The implementation of CN-IFE Algorithm 1 is much simpler than that of Algorithms 2 and 3 because all test functions, trial functions and coefficient functions are evaluated on the same time level $t = t_{n+\frac{1}{2}}$. For the interface curve at a given time level, assembling the needed matrices follows the same procedure as assembling matrices of an IFE method for time independent interface problems. Hence, the local matrix assembler of an IFE method for time independent interface problems can be easily adopted for assembling matrices used in CN-IFE Algorithm 1.

Generating matrix $K_h^{n+\frac{1}{2},n+\frac{1}{2}}$ does not need much computational cost because it involves interface elements only. On the other hand, the matrix $K_h^{n+\frac{1}{2},n+\frac{1}{2}}$ in CN-IFE-Algorithm 1 requires the time derivative of IFE basis functions. We note that for the linear IFE basis function defined by (3.7) on an interface element $T = \triangle A_1A_2A_3$ (see Fig. 3) at time t, its time derivative is given in (3.9) which involves the computation of $\frac{\partial \phi_{1,T}^t(x,y)}{\partial d}$, $\frac{\partial \phi_{1,T}^t(x,y)}{\partial e}$, $d'(t)$ and $e'(t)$. The proof of Lemma 3.2 has provided formulas for $\frac{\partial \phi_{1,T}^t(x,y)}{\partial d}$ and $\frac{\partial \phi_{1,T}^t(x,y)}{\partial e}$. Here we discuss how to find $d'(t)$ and $e'(t)$ for a moving curve $\Gamma(t,x,y) = 0$ in the domain Ω. Note that $D(x_D, y_D)$ is on the moving curve, therefore

$$\Gamma(t, x_D, y_D) = 0. \tag{4.1}$$

Taking the derivative with respect to t, we have

$$\Gamma_t(t, x_D, y_D) + \Gamma_x(t, x_D, y_D)\frac{\partial x_D}{\partial t} + \Gamma_y(t, x_D, y_D)\frac{\partial y_D}{\partial t} = 0. \tag{4.2}$$

Since $x_D = x_1 + d(t)(x_3 - x_1)$, and $y_D = y_1 + d(t)(y_3 - y_1)$, we have

$$\Gamma_t(t, x_D, y_D) + \Gamma_x(t, x_D, y_D)(x_3 - x_1)d'(t) + \Gamma_y(t, x_D, y_D)(y_3 - y_1)d'(t) = 0. \tag{4.3}$$

Therefore,

$$d'(t) = \frac{-\Gamma_t(t, x_D, y_D)}{\Gamma_x(t, x_D, y_D)(x_3 - x_1) + \Gamma_y(t, x_D, y_D)(y_3 - y_1)}. \tag{4.4}$$

Similarly,

$$e'(t) = \frac{-\Gamma_t(t, x_E, y_E)}{\Gamma_x(t, x_E, y_E)(x_2 - x_1) + \Gamma_y(t, x_E, y_E)(y_2 - y_1)}. \tag{4.5}$$

We note that these formulas can be easily extended for the other two linear IFE basis functions on a triangular interface element T. Similar computation procedures can be readily developed for the bilinear IFE and 1D linear IFE spaces.

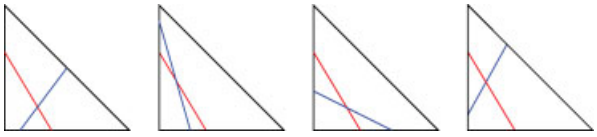

FIG. 5. Cases of the reference triangle cut by two interface segments: intersect inside the triangle. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

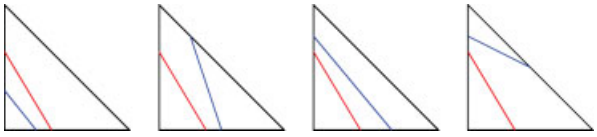

FIG. 6. Cases of the reference triangle cut by two interface segments: intersect outside the triangle. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

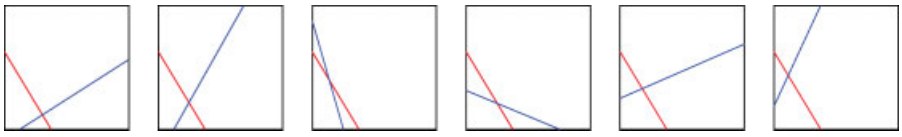

FIG. 7. Cases of the type I reference rectangle cut by two interface segments: intersect inside the rectangle. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

Local Matrices for CN-IFE Algorithm 2 and Algorithm 3. The implementation for CN-IFE Algorithms 2 and 3 is more complicated than that of CN-IFE Algorithm 1 because test functions, trial functions and coefficient functions in the matrices of these algorithms are not evaluated at the same time level. Essentially, we need a local matrix assembler that can handle multiple interface curves within one element and many configurations of interface locations have to be considered. We note that all the matrices are defined according to bilinear forms integrating the product of a test function and a trial function, and a coefficient function. For the CN-IFE Algorithm 2, the coefficient functions and test functions are evaluated at the same time $t_{n+\frac{1}{2}}$, but the trial functions are evaluated at different time t_n or t_{n+1}. For the CN-IFE Algorithm 3, the coefficient functions and trial functions are evaluated at the same time t_n or t_{n+1} while the test functions are evaluated at a different time $t_{n+\frac{1}{2}}$. Hence, for CN-IFE Algorithm 2 and 3, each interface element can contain up to two interfaces.

For the two-dimensional case, the interface curve $\Gamma(t)$ restricted in an interface element is approximated by a straight segment $\overline{\Gamma}(t)$. If an element T contains only one interface line segment, then the related computations for generating a local matrix should be carried out through two subelements of T. If T contains two interfaces at two consecutive time levels, then T is partitioned into 4 or 3 subelements by the interface line segments depending on whether these

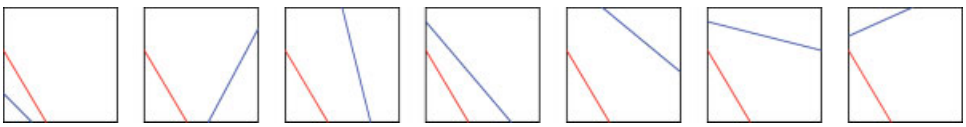

FIG. 8. Cases of the type I reference rectangle cut by two interface segments: intersect outside the rectangle. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

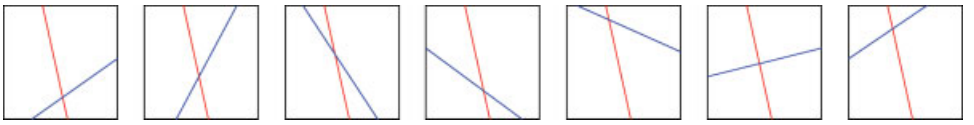

FIG. 9. Cases of the type II reference rectangle cut by two interface segments: intersect inside the rectangle. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

two line segments intersect within or outside T, see illustrations in Figs. 5–10. Accordingly, the computations for assembling a local matrix should be carried out over all of these subelements. Similar ideas apply to the corresponding 1D algorithms.

B. Efficient Construction of Global Matrices

Solving a parabolic problem with a moving interface is very time consuming because an algebraic system has to be formed and then solved at each time step. This issue becomes more severe if smaller time steps are needed for the stability of the algorithm or for a better solution resolution.

The conventional procedure for generating a global matrix is to go through each element in a mesh, generate the related local matrix, and then assemble it into the global matrix. In this way, generating all the global matrices in an algorithm at all the time steps inevitably costs a significant amount of computational time. However, using IFE spaces in the algorithms allows us to assemble the matrices at each time step through an alternative and more efficient procedure. If we use a time independent mesh in an IFE method, each element in the mesh is fixed. At any given time, an element will be considered as a noninterface element containing one material or another depending on whether the interface has moved across it, or as an interface element if the interface has moved into this element. We note that the majority of elements in the mesh are noninterface elements if the mesh is fine enough. The property of most of noninterface elements remain unchanged when the interface changes from one time level to another, see Fig. 11. Hence, the local matrices generated on these elements remain same, and their contributions to the global matrices at the old and new time levels are the same. Specifically, at a given time level, we say that an element has changed its type if its material configuration has changed with respect to the previous time level. This suggests that we can generate global matrices at the new time level by modifying their corresponding matrices at the previous time level through those elements whose types have changed. The assembling procedure based on this idea obviously costs much less computational time than the conventional procedure to assemble global matrices at each level, and the accumulated time reduction over all the time steps is significant. We would also like to note that the conventional FE methods based body-fitting meshes do not have this time-saving feature.

V. NUMERICAL EXPERIMENTS

In this section, we will present numerical examples in both 1D and 2D cases to illustrate the features of the proposed CN-IFE algorithms.

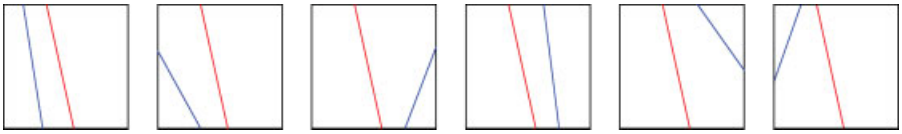

FIG. 10. Cases of the type II reference rectangle cut by two interface segments: intersect outside the rectangle. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

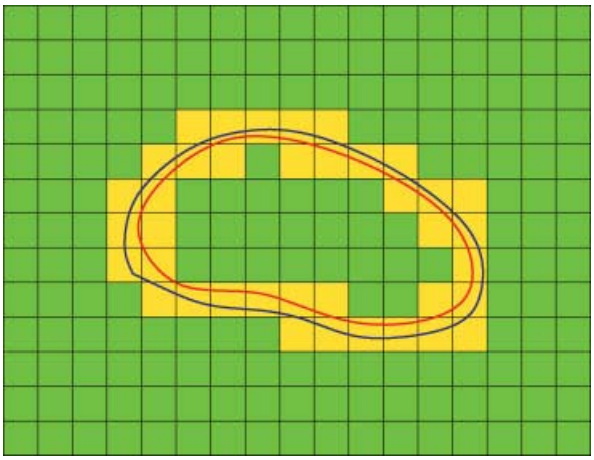

FIG. 11. The two curves represent the interfaces at two different time levels. The darker grey square elements remain unchanged and the grey square elements change their type. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

A. 1D Example

For 1D case, we consider the interface problem (1.1)–(1.4) defined on $\Omega \times [0, T_{\text{end}}]$, where $\Omega = (0, 1)$ and $T_{\text{end}} = 1$. The interface $\alpha(t)$ is a moving point in Ω separating Ω into two subdomains $\Omega^-(t) = (0, \alpha(t))$ and $\Omega^+ = (\alpha(t), 1)$. The initial and boundary value functions, and the source term $f(t, x)$ are chosen such that the following function $u(t, x)$ is the exact solution:

$$u(t,x) = \begin{cases} \left((x-\alpha(t))^2 + \frac{1}{\beta^-}\right) e^x, & x \in \Omega^-(t), \\ \left((x-\alpha(t))^2 + \frac{1}{\beta^+}\right) e^x + \left(\frac{1}{\beta^-} - \frac{1}{\beta^+}\right) e^{\alpha(t)}, & x \in \Omega^+(t). \end{cases} \tag{5.1}$$

In all the numerical examples, a uniform mesh $\mathcal{T}_h$ is used for the spatial discretization with mesh size $h = 1/N_s$. For the uniform time discretization, we denote its step size by τ and define $t_n = n\tau$, $n = 1, 2, \ldots, N$.

In the first example, we set $\alpha(t) = \frac{1}{2}t + \frac{1}{4}$, choose the time step $\tau = h$, and let $\beta^- = 1, \beta^+ = 2$ representing a moderate jump in the coefficient. Corresponding numerical results at the time level $t = 1$ are given in Table II. We observe that all the CN-IFE Algorithms behave similarly and

TABLE II. Errors of 1D linear IFE solution with $\beta^- = 1$, $\beta^+ = 2$ and $\tau = h$ at time $t = 1$.

	CN-IFE A1		CN-IFE A2		CN-IFE A3	
N_s	$\|\cdot\|_{L^2}$	$\|\cdot\|_{H^1}$	$\|\cdot\|_{L^2}$	$\|\cdot\|_{H^1}$	$\|\cdot\|_{L^2}$	$\|\cdot\|_{H^1}$
20	1.045E-3	6.680E-2	1.038E-3	6.680E-2	1.154E-3	6.689E-2
40	2.620E-4	3.343E-2	2.598E-4	3.343E-2	2.890E-4	3.345E-2
80	6.557E-5	1.672E-2	6.499E-5	1.672E-2	7.218E-5	1.673E-2
160	1.640E-5	8.359E-3	1.625E-5	8.358E-3	1.805E-5	8.361E-2
320	4.102E-6	4.179E-3	4.063E-6	4.179E-3	4.518E-6	4.180E-2
640	1.026E-6	2.090E-3	1.016E-6	2.090E-3	1.131E-6	2.090E-3
1280	2.564E-7	1.045E-3	2.540E-7	1.045E-3	2.831E-7	1.045E-3
2560	6.409E-8	5.224E-4	6.348E-8	5.224E-4	7.084E-8	5.224E-4

TABLE III. Errors of 1D linear IFE solution with $\beta^- = 1$, $\beta^+ = 100$ and $\tau = h$ at time $t = 1$.

	CN-IFE A1		CN-IFE A2		CN-IFE A3	
N_s	$\|\cdot\|_{L^2}$	$\|\cdot\|_{H^1}$	$\|\cdot\|_{L^2}$	$\|\cdot\|_{H^1}$	$\|\cdot\|_{L^2}$	$\|\cdot\|_{H^1}$
20	9.630E-4	6.022E-2	9.720E-4	6.022E-2	2.220E-2	4.936E-1
40	2.441E-4	3.013E-2	2.454E-4	3.013E-2	5.912E-3	1.951E-1
80	6.154E-5	1.507E-2	6.160E-5	1.507E-2	1.538E-3	7.661E-2
160	1.547E-5	7.535E-3	1.543E-5	7.535E-3	3.886E-4	2.990E-2
320	3.879E-6	3.768E-3	3.861E-6	3.768E-3	9.657E-5	1.175E-2
640	9.712E-7	1.884E-3	9.658E-7	1.884E-3	2.401E-5	4.697E-3
1280	2.430E-7	9.419E-4	2.415E-7	9.419E-4	5.984E-6	1.920E-3
2560	6.078E-8	4.710E-4	6.038E-8	4.710E-4	1.494E-6	8.054E-4

their numerical solutions converge to the exact solution with the optimal rates $O(h^2)$ and $O(h)$ in the L^2 and semi-H^1 norms, respectively. By linear regression, we can see these data have the following estimates:

- 1D small jump with $\tau = h, n = N$:

$$\text{CN-IFE A1:} \quad \left\|u_h^n - u(t_n,\cdot)\right\|_{L^2} \approx 0.4177h^{1.9992}, \quad \left|u_h^n - u(t_n,\cdot)\right|_{H^1} \approx 1.3362h^{0.9999}.$$
$$\text{CN-IFE A2:} \quad \left\|u_h^n - u(t_n,\cdot)\right\|_{L^2} \approx 0.4151h^{1.9996}, \quad \left|u_h^n - u(t_n,\cdot)\right|_{H^1} \approx 1.3362h^{0.9999}.$$
$$\text{CN-IFE A3:} \quad \left\|u_h^n - u(t_n,\cdot)\right\|_{L^2} \approx 0.4599h^{1.9989}, \quad \left|u_h^n - u(t_n,\cdot)\right|_{H^1} \approx 1.3385h^{1.0001}.$$

To see the performance of these algorithms for problems with large jumps in the coefficient, we test them with the problem whose exact solution is given in (5.1) with $\beta^- = 1$ and $\beta^+ = 100$. Corresponding numerical results are presented in Table III. Linear regression yields the following error estimates:

- 1D large jump with $\tau = h, n = N$:

$$\text{CN-IFE A1:} \quad \left\|u_h^n - u(t_n,\cdot)\right\|_{L^2} \approx 0.3815h^{1.9938}, \quad \left|u_h^n - u(t_n,\cdot)\right|_{H^1} \approx 1.2043h^{0.9998}.$$
$$\text{CN-IFE A2:} \quad \left\|u_h^n - u(t_n,\cdot)\right\|_{L^2} \approx 0.3878h^{1.9970}, \quad \left|u_h^n - u(t_n,\cdot)\right|_{H^1} \approx 1.2043h^{0.9998}.$$
$$\text{CN-IFE A3:} \quad \left\|u_h^n - u(t_n,\cdot)\right\|_{L^2} \approx 8.9355h^{1.9853}, \quad \left|u_h^n - u(t_n,\cdot)\right|_{H^1} \approx 25.830h^{1.3284}.$$

The data in Table III indicate that the Algorithm 1 and Algorithm 2 behave similarly and their numerical solutions converge optimally in both L^2 and semi-H^1 norms. Errors in the numerical solutions generated by Algorithm 3 still decrease with optimal convergence rates in L^2 norm and semi-H^1 norms, but the magnitudes of errors are much larger than those of Algorithms 1 and 2. If we reduce the time step τ in Algorithm 3, the magnitudes of errors become smaller and comparable to those of Algorithms 1 and 2. Table IV presents the corresponding data, and the related linear regressions are given below:

TABLE IV. Errors of 1D linear IFE solution in CN-IFE A3 with $\beta^- = 1$ and $\beta^+ = 100$ at time $t = 1$.

	$\tau = h/2$		$\tau = h/8$		$\tau = h/32$	
N_s	$\|\cdot\|_{L^2}$	$\|\cdot\|_{H^1}$	$\|\cdot\|_{L^2}$	$\|\cdot\|_{H^1}$	$\|\cdot\|_{L^2}$	$\|\cdot\|_{H^1}$
20	1.009E-2	2.923E-1	1.365E-3	7.696E-2	8.793E-4	6.022E-2
40	2.722E-3	1.218E-1	3.921E-4	4.345E-2	2.209E-4	3.014E-2
80	7.024E-4	4.963E-2	1.098E-4	2.178E-2	5.538E-5	1.557E-2
160	1.768E-4	2.017E-2	2.969E-5	1.030E-2	1.388E-5	7.540E-3
320	4.434E-5	8.308E-3	7.831E-6	4.798E-3	3.480E-6	3.777E-3
640	1.112E-5	3.486E-3	2.031E-6	2.248E-3	8.740E-7	1.892E-3

- 1D large jump for CN-IFE A3 with different $\tau, n = N$:

$$\tau = \frac{1}{2}h: \quad \left\|u_h^n - u(t_n,\cdot)\right\|_{L^2} \approx 3.8239h^{1.9696}, \quad \left|u_h^n - u(t_n,\cdot)\right|_{H^1} \approx 13.645h^{1.2820}.$$

$$\tau = \frac{1}{8}h: \quad \left\|u_h^n - u(t_n,\cdot)\right\|_{L^2} \approx 0.3986h^{1.8796}, \quad \left|u_h^n - u(t_n,\cdot)\right|_{H^1} \approx 1.8608h^{1.0315}.$$

$$\tau = \frac{1}{32}h: \quad \left\|u_h^n - u(t_n,\cdot)\right\|_{L^2} \approx 0.3470h^{1.9952}, \quad \left|u_h^n - u(t_n,\cdot)\right|_{H^1} \approx 1.2129h^{0.9999}.$$

B. 2D Example

We now consider the interface problem defined by (1.1)–(1.4) on $\Omega \times [0, T_{\text{end}}]$, where $\Omega = (-1,1)\times(-1,1)$ and $T_{\text{end}} = 1$. The interface $\Gamma(t)$ is a moving circle centered at origin with radius $r(t)$ which separates Ω into two subdomains $\Omega^-(t) = \{(x,y) \in \Omega : x^2 + y^2 < r(t)^2\}$, and $\Omega^+(t) = \{(x,y) \in \Omega : x^2 + y^2 > r(t)^2\}$. The exact solution is chosen as:

$$u(t,x,y) = \begin{cases} \dfrac{1}{\beta^-}(x^2+y^2)^{5/2}\cos(t), & (x,y) \in \Omega^-(t), \\ \dfrac{1}{\beta^+}(x^2+y^2)^{5/2}\cos(t) + \left(\dfrac{1}{\beta^-} - \dfrac{1}{\beta^+}\right) r(t)^5\cos(t), & (x,y) \in \Omega^+(t). \end{cases} \tag{5.2}$$

In all the numerical examples presented below, the radius change is governed by $r(t) = r_0(\frac{\sin(t)+3}{4})$ with $r_0 = \frac{\pi}{6.28}$, and we use triangular Cartesian meshes $\mathcal{T}_h$ which are formed by partitioning Ω with $N_s \times N_s$ rectangles of size $h = 2/N_s$ and then cutting each rectangle into two triangles along one of its diagonal line. For time discretization, we denote its step size by τ and define $t_n = n\tau$, with $n = 1, 2, \ldots, N$.

In the first example, we choose the diffusion coefficients to be $\beta^- = 1$, $\beta^+ = 2$, and we test the algorithms with $\tau = h$ and $\tau = \frac{1}{8}h$. Errors at the final time level in the L^2 and semi-H^1 norms are listed in Table V and Table VI, respectively.

From Tables V and VI, we can see both CN-IFE Algorithms 1 and 2 perform optimally with either large or small time steps. As we expected, CN-IFE Algorithm 3 have certain difficulties when a large time step τ is used. Data in Table V suggest that numerical solutions generated by CN-IFE Algorithm 3 with $\tau = h$ converge to the exact solution, but not optimally. However, when we shrink the time step size from $\tau = h$ to $\tau = \frac{1}{8}h$, the optimal convergence for CN-IFE Algorithm 3 is recovered. Linear regression for these data yields the following estimates:

TABLE V. Errors of 2D linear IFE solution with $\beta^- = 1$, $\beta^+ = 2$ and $\tau = h$ at time $t = 1$.

	CN-IFE A1		CN-IFE A2		CN-IFE A3	
N_s	$\|\cdot\|_{L^2}$	$\|\cdot\|_{H^1}$	$\|\cdot\|_{L^2}$	$\|\cdot\|_{H^1}$	$\|\cdot\|_{L^2}$	$\|\cdot\|_{H^1}$
20	1.020E-2	2.918E-1	1.012E-2	2.918E-1	1.021E-2	2.932E-1
40	2.551E-3	1.463E-1	2.550E-3	1.462E-1	2.611E-3	1.501E-1
60	1.134E-3	9.759E-2	1.133E-3	9.755E-2	1.247E-3	1.061E-1
80	6.380E-4	7.323E-2	6.377E-4	7.318E-2	6.950E-4	8.087E-2
100	4.087E-4	5.861E-2	4.084E-4	5.855E-2	5.069E-4	7.150E-2
120	2.840E-4	4.889E-2	2.837E-4	4.880E-2	3.419E-4	5.928E-2
140	2.087E-4	4.193E-2	2.084E-4	4.183E-2	2.925E-4	5.884E-2
160	1.599E-4	3.670E-2	1.596E-4	3.661E-2	2.336E-4	5.346E-2
180	1.264E-4	3.264E-2	1.261E-4	3.254E-2	2.070E-4	5.192E-2
200	1.025E-4	2.942E-2	1.022E-4	2.930E-2	1.764E-4	5.030E-2

- 2D small jump with $\tau = h$, $n = N$:

$$\text{CN-IFE A1:} \quad \left\|u_h^n - u(t_n,\cdot)\right\|_{L^2} \approx 4.0518h^{1.9980}, \quad \left|u_h^n - u(t_n,\cdot)\right|_{H^1} \approx 5.7820h^{0.9969}.$$

$$\text{CN-IFE A2 :} \quad \left\|u_h^n - u(t_n,\cdot)\right\|_{L^2} \approx 4.0189h^{1.9966}, \quad \left|u_h^n - u(t_n,\cdot)\right|_{H^1} \approx 5.8133h^{0.9985}.$$

$$\text{CN-IFE A3 :} \quad \left\|u_h^n - u(t_n,\cdot)\right\|_{L^2} \approx 1.7818h^{1.7634}, \quad \left|u_h^n - u(t_n,\cdot)\right|_{H^1} \approx 2.6062h^{0.7686}.$$

- 2D small jump with $\tau = \frac{1}{8}h$, $n = N$:

$$\text{CN-IFE A1 :} \quad \left\|u_h^n - u(t_n,\cdot)\right\|_{L^2} \approx 4.1100h^{1.9983}, \quad \left|u_h^n - u(t_n,\cdot)\right|_{H^1} \approx 5.8153h^{0.9986}.$$

$$\text{CN-IFE A2 :} \quad \left\|u_h^n - u(t_n,\cdot)\right\|_{L^2} \approx 4.1100h^{1.9983}, \quad \left|u_h^n - u(t_n,\cdot)\right|_{H^1} \approx 5.8153h^{0.9986}.$$

$$\text{CN-IFE A3 :} \quad \left\|u_h^n - u(t_n,\cdot)\right\|_{L^2} \approx 4.0540h^{1.9944}, \quad \left|u_h^n - u(t_n,\cdot)\right|_{H^1} \approx 5.6961h^{0.9928}.$$

The next example is for a large jump in coefficient. We test the same exact solution defined in (5.2) with $\beta^- = 1$, $\beta^+ = 100$. We choose $\tau = \frac{1}{8}h$ in our numerical experiments, and the related errors of numerical solutions at the final time level in the L^2 and semi-H^1 norms are listed in Table VII. Data in this table indicate that numerical solutions generated by CN-IFE Algorithms 1 and 2 converge optimally to the exact solution. In contrast, numerical solutions generated by

TABLE VI. Errors of 2D linear IFE solution with $\beta^- = 1$, $\beta^+ = 2$ and $\tau = \frac{1}{8}h$ at time $t = 1$.

	CN-IFE A1		CN-IFE A2		CN-IFE A3	
N_s	$\|\cdot\|_{L^2}$	$\|\cdot\|_{H^1}$	$\|\cdot\|_{L^2}$	$\|\cdot\|_{H^1}$	$\|\cdot\|_{L^2}$	$\|\cdot\|_{H^1}$
20	1.032E-2	2.917E-1	1.032E-2	2.917E-1	1.032E-2	2.917E-1
40	2.588E-3	1.462E-1	2.588E-3	1.462E-1	2.587E-3	1.463E-1
60	1.150E-3	9.754E-2	1.150E-3	9.754E-2	1.150E-3	9.766E-2
80	6.473E-4	7.317E-2	6.473E-4	7.317E-2	6.503E-4	7.347E-2
100	4.144E-4	5.854E-2	4.144E-4	5.854E-2	4.154E-4	5.866E-2
120	2.878E-4	4.879E-2	2.878E-4	4.879E-2	2.880E-4	4.900E-2
140	2.115E-4	4.182E-2	2.115E-4	4.182E-2	2.131E-4	4.219E-2
160	1.619E-4	3.659E-2	1.619E-4	3.659E-2	1.627E-4	3.688E-2
180	1.280E-4	3.253E-2	1.280E-4	3.253E-2	1.286E-4	3.278E-2
200	1.037E-4	2.928E-2	1.037E-4	2.928E-2	1.049E-4	2.983E-2

TABLE VII. Errors of 2D linear IFE solution with $\beta^- = 1$, $\beta^+ = 100$ and $\tau = \frac{1}{8}h$ at time $t = 1$.

	CN-IFE A1		CN-IFE A2		CN-IFE A3	
N_s	$\|\cdot\|_{L^2}$	$\|\cdot\|_{H^1}$	$\|\cdot\|_{L^2}$	$\|\cdot\|_{H^1}$	$\|\cdot\|_{L^2}$	$\|\cdot\|_{H^1}$
20	5.759E-4	1.448E-2	5.760E-4	1.449E-2	1.073E-2	4.894E-1
40	1.665E-4	8.055E-3	1.666E-4	8.055E-3	6.278E-1	5.292E+1
60	7.284E-5	5.564E-3	7.285E-5	5.565E-3	2.143E+3	2.659E+5
80	4.072E-5	4.271E-3	4.070E-5	4.265E-3	4.880E+6	4.374E+8
100	2.718E-5	3.461E-3	2.715E-5	3.457E-3	5.936E+7	1.246E+10
120	1.962E-5	2.914E-3	1.959E-5	2.902E-3	5.991E+10	1.432E+13
140	1.455E-5	2.530E-3	1.453E-5	2.519E-3	9.439E+10	1.639E+13
160	1.084E-5	2.247E-3	1.083E-5	2.244E-3	2.100E+17	6.150E+19
180	8.462E-6	2.027E-3	8.450E-6	2.019E-3	1.216E+17	3.926E+19
200	7.004E-6	1.815E-3	6.991E-6	1.803E-3	1.287E+25	4.480E+27

CN-IFE Algorithm 3 do not converge because of its instability. This further suggests that CN-IFE Algorithm 3 cannot handle large changes in the coefficient, either caused by a large jump or a large time step.

Our numerical experiments indicate that CN-IFE Algorithm 3 can still produce good numerical results provided that the time step is small enough. As we shrink the time step size τ further, the numerical solutions obtained by CN-IFE Algorithm 3 converge and the corresponding errors become comparable to those generated by Algorithms 1 and 2. Corresponding data can be found in Table VIII.

C. Algorithm 1*

The results of Theorem 3.1 and 3.2 show that terms involving the matrix $K_h^{n+\frac{1}{2},n+\frac{1}{2}}$ in Algorithm 1 are much "smaller" than those terms involving the mass matrix $M_h^{n+\frac{1}{2},n+\frac{1}{2}}$ and the stiffness matrix $A_h^{n+\frac{1}{2},n+\frac{1}{2},n+\frac{1}{2}}$. If we omit those terms involving $K_h^{n+\frac{1}{2},n+\frac{1}{2}}$, then we obtain the following algorithm in a form similar to Algorithm 2.

- CN-IFE Algorithm 1*

$$\left(M_h^{n+\frac{1}{2},n+\frac{1}{2}} + \frac{\tau}{2}A_h^{n+\frac{1}{2},n+\frac{1}{2},n+\frac{1}{2}}\right)\mathbf{u}^{n+1} = \left(M_h^{n+\frac{1}{2},n+\frac{1}{2}} - \frac{\tau}{2}A_h^{n+\frac{1}{2},n+\frac{1}{2},n+\frac{1}{2}}\right)\mathbf{u}^n + \tau\mathbf{f}^{n+\frac{1}{2},n+\frac{1}{2}}.$$

TABLE VIII. Errors of 2D linear IFE solution in CN-IFE Algorithm 3 with $\beta^- = 1$ and $\beta^+ = 100$.

	$\tau = h/16$		$\tau = h/64$		$\tau = h/256$	
N_s	$\|\cdot\|_{L^2}$	$\|\cdot\|_{H^1}$	$\|\cdot\|_{L^2}$	$\|\cdot\|_{H^1}$	$\|\cdot\|_{L^2}$	$\|\cdot\|_{H^1}$
20	6.263E-4	1.670E-2	5.768E-4	1.448E-2	5.760E-4	1.448E-2
40	1.146E-3	9.479E-2	1.723E-4	8.097E-3	1.674E-4	8.041E-3
60	1.140E-3	1.415E-1	1.606E-4	6.314E-3	8.201E-5	5.557E-3
80	6.639E-2	1.156E+1	9.318E-5	5.772E-3	4.377E-5	4.259E-3

TABLE IX. Errors of 1D linear IFE solution using CN-IFE-A1* with $\beta^- = 1$ and $\tau = h$ at time $t = 1$.

	$\beta^+ = 2$		$\beta^+ = 100$	
N_s	$\|\cdot\|_{L^2}$	$\|\cdot\|_{H^1}$	$\|\cdot\|_{L^2}$	$\|\cdot\|_{H^1}$
20	8.050E-4	6.688E-2	8.199E-4	6.025E-2
40	1.921E-4	3.347E-2	1.967E-4	3.015E-2
80	4.660E-5	1.674E-2	4.672E-5	1.508E-2
160	1.360E-5	8.368E-3	1.125E-5	7.538E-3
320	5.954E-6	4.184E-3	2.806E-6	3.769E-3
640	3.128E-6	2.092E-3	7.777E-7	1.884E-3
1280	1.649E-6	1.046E-3	2.855E-7	9.422E-4
2560	8.516E-7	5.230E-4	1.366E-7	4.711E-4

For the 1D examples above, errors of the numerical solutions generated by CN-IFE Algorithm 1* are given in Table IX. Applying linear regression to these data yields the following estimates:

- CN-IFE A1* : 1D small jump with $\tau = h, n = N$

$$\left\|u_h^n - u(t_n,\cdot)\right\|_{L^2} \approx 0.0275h^{1.3857}, \quad \left|u_h^n - u(t_n,\cdot)\right|_{H^1} \approx 1.3380h^{0.9999}.$$

- CN-IFE A1* : 1D large jump with $\tau = h, n = N$

$$\left\|u_h^n - u(t_n,\cdot)\right\|_{L^2} \approx 0.1565h^{1.8420}, \quad \left|u_h^n - u(t_n,\cdot)\right|_{H^1} \approx 1.2046h^{0.9997}.$$

We observe that these numerical solutions still maintain the optimal convergence rate in the semi-H^1 norm, but only a suboptimal rate in the L^2 norm. As suggested by Theorem 3.1, such a behavior is probably caused by the omitting terms of order $O(h)$ which will not impact the error measurement in the semi-H^1 norm which is expected to have an $O(h)$ order. But these terms seem not to be negligible for the error measurement in the L^2 norm.

For the same 2D example above, errors of the numerical solutions generated by CN-IFE Algorithm 1* are presented in Table X. By linear regression, we have

- CN-IFE A1* : 2D small jump with $\tau = \frac{h}{8}, n = N$

$$\left\|u_h^n - u(t_n,\cdot)\right\|_{L^2} \approx 4.1115h^{1.9983}, \quad \left|u_h^n - u(t_n,\cdot)\right|_{H^1} \approx 5.8153h^{0.9986}.$$

TABLE X. Errors of 2D linear IFE solution using CN-IFE-A1* with $\beta^- = 1$ and $\tau = h$ at time $t = 1$.

	$\beta^+ = 2, \tau = h$		$\beta^+ = 2, \tau = \frac{1}{8}h$		$\beta^+ = 100, \tau = \frac{1}{8}h$	
N_s	$\|\cdot\|_{L^2}$	$\|\cdot\|_{H^1}$	$\|\cdot\|_{L^2}$	$\|\cdot\|_{H^1}$	$\|\cdot\|_{L^2}$	$\|\cdot\|_{H^1}$
20	1.020E-2	2.918E-1	1.032E-2	2.917E-1	5.741E-4	1.448E-2
40	2.551E-3	1.463E-1	2.588E-3	1.462E-1	1.661E-4	8.055E-3
60	1.134E-3	9.759E-2	1.151E-3	9.754E-2	7.260E-5	5.564E-3
80	6.381E-4	7.323E-2	6.473E-4	7.317E-2	4.054E-5	4.271E-3
100	4.088E-4	5.861E-2	4.145E-4	5.854E-2	2.704E-5	3.461E-3
120	2.841E-4	4.889E-2	2.879E-4	4.879E-2	1.953E-5	2.914E-3
140	2.088E-4	4.193E-2	2.115E-4	4.182E-2	1.451E-5	2.530E-3
160	1.600E-4	3.670E-2	1.619E-4	3.659E-2	1.081E-5	2.247E-3
180	1.264E-4	3.264E-2	1.280E-4	3.253E-2	8.444E-6	2.027E-3
200	1.026E-4	2.942E-2	1.037E-4	2.928E-2	6.985E-6	1.815E-3

- CN-IFE A1* : 2D large jump with $\tau = \frac{h}{8}$, $n = N$

$$\left\|u_h^n - u(t_n, \cdot)\right\|_{L^2} \approx 0.2211 h^{1.9554}, \quad \left|u_h^n - u(t_n, \cdot)\right|_{H^1} \approx 0.2440 h^{0.9240}.$$

From these numerical results, we observe that even though CN-IFE Algorithm 1* is simpler than CN-IFE Algorithm 1, it can still perform optimally. As stated in Theorem 3.2, in 2D case, matrix $K_h^{n+\frac{1}{2},n+\frac{1}{2}}$ has a bound $\|K_h^{n+\frac{1}{2},n+\frac{1}{2}}\|_F \leq Ch^{\frac{1}{2}}$ that is of half order smaller than that in 1D case; hence, omitting those terms involving $K_h^{n+\frac{1}{2},n+\frac{1}{2}}$ should have a less impact on the numerical results. It is an interesting research topic to find out why the simpler CN-IFE Algorithm 1* can perform optimally for 2D parabolic moving problems.

VI. CONCLUSIONS

In this article, we propose three immersed finite element methods based on the Crank-Nicolson type discretization for solving parabolic moving interface problems with a fixed Cartesian mesh. Numerical examples are presented to illustrate their features. The first algorithm seems to be promising in the sense of its optimal convergence and efficient implementation nature. The second method is a natural extension of the original Crank-Nicolson method and it maintains the same accuracy as the first one, but its implementation is more complicated. The third method is based on the idea of maintaining the continuity of flux across the interface, but it has a conditional stability. All three methods reduces to the standard Crank-Nicolson method if the diffusion coefficient has no jump or if the interface is time independent.

References

1. Jr. J. R. Cannon, J. Douglas, and C. Denson Hill, A multi-boundary stefan problem and the disappearance of phases, J Math Mech 17 (1967), 21–33.
2. G. H. Meyer, the Multidimensional stefan problems, SIAM J Numer Anal 10 (1973), 522–538.
3. B. H. Gilding, Qualitative mathematical analysis of the Richards equation, Trans Porous Med 6 (1991), 651–666.
4. M. Ye, R. Khaleel, and T. J. Yeh, Stochastic analysis of moisture plume dynamics of a field injection experiment, Water Resour Res 41 (2005), W03013.
5. V. Thomée, Galerkin finite element methods for parabolic problems, Springer series in computational mathematics, 2nd Ed., Vol. 25, Springer-Verlag, Berlin, 2006.
6. I. Babuška, The finite element method for elliptic equations with discontinuous coefficients, Computing 5 (1970), 207–213.
7. J. H. Bramble and J. T. King, A finite element method for interface problems in domains with smooth boundary and interfaces, Adv Comput Math 6 (1996), 109–138.
8. Z. Chen and J. Zou, Finite element methods and their convergence for elliptic and parabolic interface problems, Numer Math 79 (1998), 175–202.
9. I. Babuška and J. E. Osborn, Can a finite element method perform arbitrarily badly? Math Comp 69 (2000), 443–462.
10. S. C. Reddy and L. N. Trefethen, Stability of the method of lines, Numer Math 62 (1992), 235–267.
11. W. E. Schiesser and G. W. Griffiths, A compendium of partial differential equation models, Method of lines analysis with Matlab, Cambridge University Press, Cmbridge, 2009.

12. A. Zafarullah, Application of the method of lines to parabolic partial differential equations with error estimates, J ACM 17 (1970), 294–302.

13. T. Lin and J. Wang, An immersed finite element electric field solver for ion optics modeling, In Proceedings of the AIAA Joint Propulsion Conference, Indianapolis, IN, July, 2002. AIAA, pp. 2002–4263.

14. T. Lin and J. Wang, The immersed finite element method for plasma particle simulation, In Proceedings of the AIAA Aerospace Sciences Meeting, Reno, NV, January, 2003, AIAA, pp. 2003–0842.

15. J. Wang, X.-M. He, and Y. Cao, Modeling spacecraft charging and charged dust particle interactions on lunar surface, Proceedings of the 10th Spacecraft Charging Technology Conference, Biarritz, France, 2007.

16. J. Wang, X.-M. He, and Y. Cao, Modeling electrostatic levitation of dusts on lunar surface, IEEE Trans Plasma Sci 36 (2008), 2459–2466.

17. C. S. Peskin, Flow patterns around heart valves, J Comput Phys 10 (1972), 252–271.

18. C. S. Peskin, Numerical analysis of blood flow in the heart, J Comput Phys 25 (1977), 220–252.

19. A. L. Fogelson and J. P. Keener, Immersed interface methods for Neumann and related problems in two and three dimensions, SIAM J Sci Comput 22 (2001), 1630–1654.

20. R. J. LeVeque and Z. Li, The immersed interface method for elliptic equations with discontinuous coefficients and singular sources, SIAM J Numer Anal 34 (1994), 1019–1044.

21. D. M. Ingram, D. M. Causon, and C. G. Mingham, Developments in Cartesian cut cell methods, Math Comput Simul 61 (2003), 561–572.

22. H. Ji, F.-S. Lien, and E. Yee, An efficient second-order accurate cut-cell method for solving the variable coeffcient Poisson equation with jump conditions on irregular domains, Int J Numer Methods Fluids 52 (2006), 723–748.

23. Y. C. Zhou, M. Feig, and G. W. Wei, Highly accurate biomolecular electrostatics in continuum dielectric environment, J Comput Chem 29 (2008), 87–97.

24. Y. C. Zhou and G. W. Wei, On the fictitious-domain and interpolation formulations of the matched interface and boundary (MIB) method, J Comput Phys 219 (2006), 228–246.

25. Y. C. Zhou, S. Zhao, M. Feig, and G. W. Wei, High order matched interface and boundary method for elliptic equations with discontinuous coefficients and singular sources, J Comput Phys 213 (2006), 1–30.

26. D. W. Hewitt, The embedded curved boundary method for orthogonal simulation meshes, J Comput Phys 138 (1997), 585–616.

27. H. Johansen and P. Colella, A Cartesian grid embedded boundary method for Poisson's equation on irregular domains, J Comput Phys 147 (1998), 60–85.

28. S. Adjerid and T. Lin, p-th degree immersed finite element for boundary value problems with discontinuous coefficients, Appl Numer Math 59 (2009), 1303–1321.

29. S. Chou, D. Y. Kwak, and K. T. Wee, Optimal convergence analysis of an immersed interface finite element method, Adv Comput Math 33 (2010), 149–168.

30. Y. Gong, B. Li, and Z. Li, Immersed-interface finite-element methods for elliptic interface problems with non-homogeneous jump conditions, SIAM J Numer Anal 46 (2008), 472–495.

31. X.-M. He, Bilinear immersed finite elements for interface problems, Ph.D. thesis, Virginia Polytechnic Institute and State University, 2009.

32. X.-M. He, T. Lin, and Y. Lin, Approximation capability of a bilinear immersed finite element space, Numer Methods Partial Differential Eq 24 (2008), 1265–1300.

33. X.-M. He, T. Lin, and Y. Lin, The convergence of the bilinear and linear immersed finite element solutions to interface problems, Numer Methods Partial Differential Eq 28 (2012), 312–330.

34. X.-M. He, T. Lin, and Y. Lin, Immersed finite element methods for elliptic interface problems with non-homogeneous jump conditions, Int J Numer Anal Model 8 (2012), 284–301.

35. R. Kafafy, T. Lin, Y. Lin, and J. Wang, Three-dimensional immersed finite element methods for electric field simulation in composite materials, Int J Numer Methods Eng 64 (2005), 940–972.

36. R. Kafafy, J. Wang, and T. Lin, A hybrid-grid immersed-finite-element particle-in-cell simulation model of ion optics plasma dynamics, Dyn Contin Discrete Impuls Syst Ser B Appl Algorithms 12 (2005), 1–16.

37. Z. Li, The immersed interface method using a finite element formulation, Appl Numer Math 27 (1997), 253–267.

38. Z. Li, T. Lin, Y. Lin, and R. C. Rogers, An immersed finite element space and its approximation capability, Numer Methods Partial Differential Eq 20 (2004), 338–367.

39. Z. Li, T. Lin, and X. Wu, New Cartesian grid methods for interface problems using the finite element formulation, Numer Math 96 (2003), 61–98.

40. Z. Li and X. Yang, An immersed finite element method for elasticity equations with interfaces. Recent advances in adaptive computation, Contemp Math 383 (2005), 285–298.

41. T. Lin, Y. Lin, R. C. Rogers, and L. M. Ryan, A rectangular immersed finite element method for interface problems, P. Minev and Y. Lin, editors, Advances in computation: theory and practice, Vol. 7, Nova Science Publishers, Huntington, New York, 2001, pp. 107–114.

42. T. Lin, Y. Lin, and W. Sun, Error estimation of a class of quadratic immersed finite element methods for elliptic interface problems, Discr Contin Dyn Syst Ser B 7 (2007), 807–823.

43. S. A. Sauter and R. Warnke, Composite finite elements for elliptic boundary value problems with discontinuous coefficients, Computing 77 (2006), 29–55.

44. S. Vallaghè and T. Papadopoulo, A trilinear immersed finite element method for solving the electroencephalography forward problem, SIAM J Sci Comput 32 (2010), 2379–2394.

45. T. S. Wang, A Hermite cubic immersed finite element space for beam design problems, Master's thesis, Virginia Polytechnic Institute and State University, 2005.

46. T. Lin and D. Sheen, The immersed finite element method for parabolic problems with the Laplace transformation in time discretization, Int J Numer Anal Model, to appear.

47. K. Wang, H. Wang, and X. Yu, An immersed eulerian-lagrangian localized adjoint method for transient advection-diffusion equations with interfaces, Int J Numer Anal Model 9 (2012), 29–42.