A SPACE-TIME DISCONTINUOUS GALERKIN SPECTRAL ELEMENT METHOD FOR THE STEFAN PROBLEM

CHAOXU PEI, MARK SUSSMAN∗ AND M. YOUSUFF HUSSAINI

Department of Mathematics
Florida State University
Tallahassee, FL, 32306, USA

(Communicated by Jie Shen)

Abstract. A novel space-time discontinuous Galerkin (DG) spectral element method is presented to solve the one dimensional Stefan problem in an Eulerian coordinate system. This method employs the level set procedure to describe the time-evolving interface. To deal with the prior unknown interface, a backward transformation and a forward transformation are introduced in the space-time mesh. By combining an Eulerian description, i.e., a fixed frame of reference, with a Lagrangian description, i.e., a moving frame of reference, the issue of dealing with implicitly defined arbitrary shaped space-time elements is avoided. The backward transformation maps the unknown time-varying interface in the fixed frame of reference to a known stationary interface in the moving frame of reference. In the moving frame of reference, the transformed governing equations, written in the space-time framework, are discretized by a DG spectral element method in each space-time slab. The forward transformation is used to update the level set function and then to project the solution in each phase back from the moving frame of reference to the fixed Eulerian grid. Two options for calculating the interface velocity are presented, and both options exhibit spectral accuracy. Benchmark tests indicate that the method converges with spectral accuracy in both space and time for the temperature distribution and the interface velocity. A Picard iteration algorithm is introduced in order to solve the nonlinear algebraic system of equations and it is found that just a few iterations lead to convergence.

1. Introduction. The Stefan problem is a moving boundary problem that models phase change; e.g., the freezing and thawing process for a solid-liquid system [49, 38, 43, 7]. A key issue in the Stefan problem is that the interface between different phases is evolving in time. The heat equation for each phase is solved in a time-dependent domain. The velocity of the phase boundaries in turn depends on the jump in the normal heat flux at the interface separating the two phases. In other words, the problem requires one to find the solutions in a prior unknown domain and to compute the shape of the unknown domain as part of the solution, which makes the problem nonlinear. Such a problem arises in numerous industrial and technological applications, for example, metal solidification, food freezing, dendritic

2010 Mathematics Subject Classification. Primary: 65M70, 35R37.
Key words and phrases. Space-time, discontinuous Galerkin, spectral accuracy, transformations, the Stefan problem.

∗ Corresponding author: Mark Sussman (sussman@math.fsu.edu).

3595
crystal growth in a supercooled liquid and ice accretion on aircraft [45, 3, 51, 18, 24, 57, 60].

Various numerical methods have been proposed for solving the Stefan problem. Ronquist and Patera [48] presented a continuous Galerkin spectral element method combined with an interface-local transformation, in which the element boundaries coincide with the phase interface. They showed exponential convergence in space and first order in time in one dimensional numerical tests. In contrast to generating a conforming grid, some authors [32, 1, 2] proposed a hybrid method in an Eulerian coordinate system, in which the interface is tracked sharply while the treatment of the interface is done by the immersed boundary method [45]. One drawback for tracking the interface explicitly is that special care has to be exercised when topological changes, such as merging or breaking occur. To overcome this difficulty, interface capturing methods, such as the phase-field method [33, 20] or the level set method [44, 55], have been introduced to solve the Stefan problem. The phase field method is based on the concept of a diffuse interface, in which the interface is given a certain thickness (see [26, 27, 25]). The phase-field method for the Stefan problem has two drawbacks that is pointed out by Javierre et al. [29]. One is that the grid resolution requirements within the interfacial region is much more stringent with the phase-field method than with the level set method. The other drawback is that more intricate physical properties must be prescribed for the phase field model which makes the method a challenge to modify if new physics are added.

Examples of sharp interface methods are the immersed interface method [39, 40], the ghost fluid method [19], the XFEM method [12], the virtual node method [8, 6], the coupled level set and volume of fluid method [54], the coupled level set and moment of fluid method [30, 60], the cell integrated semi-Lagrangian with moment of fluid approach [31], and sharp interface Cartesian grid finite volume method [59]. The earliest level set method for solidification was presented by Sethian and Strain [51]. Their method combined the level set method with a boundary integral method. Chen et al. [11] introduced a simple approach in an Eulerian coordinate system for the Stefan problem, which involved the level set method and a finite difference approach for solving the heat equation in each phase and for evaluating the interface velocity. The interface boundary condition was imposed directly by constructing interpolating polynomials on both sides of the interface. Some improvements were proposed by Gibou and his colleagues [23, 22]. In contrast to Chen et al. [11], their suggested discretizations led to a symmetric linear system. In addition, they pointed out the necessity to redefine the grid points that are swept over by the interface, i.e., a liquid node crossed by the interface must be converted to a solid node with an appropriate temperature value. Later, a high order non-symmetric approach, a third order accurate method, was presented for solving the Stefan problem by Gibou and Fedkiw [21], in which a high order extrapolation approach [5] was applied to redefine the values at nodes crossed by the interface.

In the present paper, a novel space-time discontinuous Galerkin (DG) spectral element method in an Eulerian coordinate system, coupled with the level set method, is proposed. In order to redefine the values at nodes in the phase change region, we introduce two global transformations: a backward transformation and a forward transformation. The idea of the transformations is similar to a semi-Lagrangian method, which distinguishes our method from the interface-local transformation [48] and from a mixed Eulerian-Lagrangian framework [58]. In the space-time mesh, we
first apply the backward transformation to map the unknown time-dependent interface in the Eulerian coordinate system to a known stationary interface in a new coordinate system, which also avoids handling the implicitly defined arbitrary shaped space-time elements. Such a mapping transforms the heat equation, coupled with the interface condition, defined in a prior unknown time-dependent domain into a nonlinear convection-diffusion equation defined in a known stationary domain. To solve the transformed nonlinear equation, we employ a Picard iteration [9, 46]. At each iteration, we solve a linear advection-diffusion equation by a space-time DG spectral element method in each space-time slab locally, and the level set function is updated by the forward transformation associated with the previous interface velocity. After the interface position is updated in the Eulerian coordinate system, the solution is then projected on the updated time-dependent domain.

DG methods in space have been developed for solving a wide range of physical problems, especially for those dealing with flows with discontinuities computed on unstructured meshes. Cockburn et al.[13], Canuto et al. [10], Hesthaven et al. [28], Kopriva [36], and Karniadakis and Sherwin [34] provide detailed surveys. Recently, space-time DG finite element methods have been explored in [53, 35, 61, 52, 47] for problems with a moving domain. Such space-time DG schemes result in implicit time integration schemes and naturally allow for time adaptivity. Furthermore, the auxiliary variables introduced in a DG method for discretizing the diffusion term [15, 14] can be used directly to compute the interface velocity in the weak formulation, which enables the temperature distribution and the interface velocity to be computed simultaneously. Motivated by these properties of space-time DG finite element methods, we present a novel space-time DG spectral element method for solving the one dimensional Stefan problem, which leads to spectral accuracy in both space and time.

The organization of the paper is as follows. In section 2, we first present the governing equations of the Stefan problem and introduce the level set function for capturing the phase boundary. The space-time discretization is introduced in section 3. Transformation techniques are then introduced in section 4. In section 5, the DG spectral element method is presented for the transformed equations in the space-time framework. Then two new algorithms for determining the interface velocity are described in section 5.4. Numerical tests are presented in one spatial dimension in section 6 to show the accuracy and efficiency of the present numerical scheme.

2. Mathematical model. The Stefan problem describes the temperature distribution of a pure material undergoing a phase transition. Let us consider a square domain, \( \Omega \in \mathbb{R}^d \), of a pure material but at different states, such as ice and water. Let \( \mathbf{x} = (x_1, x_2, \ldots, x_d) \) be the spatial variable, where \( d \) is the spatial dimension. The domain \( \Omega \) is composed of a time-dependent solid region denoted as \( \Omega_s^t \), and a time-dependent liquid region \( \Omega_l^t \). The interface \( \Gamma(t) \), separating different phases, is a prior unknown free boundary. The description of the Stefan problem on a Cartesian domain, \( \Omega \in \mathbb{R}^2 \), is illustrated in Fig. 1.

2.1. Governing equations. We assume that convection, thermal expansion and buoyancy are not considered. Since there is no flow in the liquid region, the governing equation in each phase is written as follows,

\[
\frac{\partial (\rho^s c_v^s \theta^s)}{\partial t} = \nabla \cdot (k^s \nabla \theta^s), \quad \text{in } \Omega_s^t, \tag{1}
\]
Figure 1. The Stefan problem on a Cartesian domain $\Omega \in \mathbb{R}^2$. $\Omega^s_t$ and $\Omega^l_t$ are time-dependent domains at time $t$, while domain $\Omega$ is fixed. $\Gamma(t)$ is the moving interface at time $t$, and $n = (n_1, n_2)$ is the interface normal pointing in the direction of interface movement. $\phi$ is the level set function introduced in section 2.2.

\[
\frac{\partial (\rho c^l \theta^l)}{\partial t} = \nabla \cdot (k^l \nabla \theta^l), \quad \text{in } \Omega^l_t, \tag{2}
\]

where $\rho$ is the density, $c_v$ denotes the specific heat capacity, $k$ is the thermal conductivity, $\theta$ is the temperature distribution, and indices "$s$", "$l$" denote the solid and liquid phase, respectively. The jump condition, also referred to as the Stefan condition, at the interface is (with the assumption of $\rho^s = \rho^l = \rho$ and $c^s_v = c^l_v = c_v$),

\[
\rho L V_n = -(k^l \nabla \theta^l - k^s \nabla \theta^s) \cdot n, \tag{3}
\]

where $L$ is a positive constant representing the latent heat of fusion, $n = (n_1, n_2, ..., n_d)$ is the interface normal pointing in the direction of interface movement, and $V_n = V \cdot n$ is the normal velocity at the interface. On $\partial \Omega$, Dirichlet boundary conditions are specified. At the interface, we set

\[
\theta^s = \theta^l = \theta^l. \tag{4}
\]

For a classical Stefan problem, $\theta^l = \theta_m$ where $\theta_m$ denotes the melting temperature. However, for a realistic modeling, such as dendritic solidification, it is important to take surface tension effects and interface curvature into account, which leads to the Gibbs-Thomson relation \[11\]. We refer the reader to \[24, 23, 58, 57, 17\] and the references therein for information on freezing models and numerical methods that take into account more detailed physics than we do in this article.

2.2. Level set function. We represent the time evolving interface by a level set function. The level set function $\phi$ is a continuous function defined as follows,

\[
\begin{cases}
\phi(x, t) < 0, & x \in \Omega^s_t, \\
\phi(x, t) = 0, & x \in \Gamma(t), \\
\phi(x, t) > 0, & x \in \Omega^l_t.
\end{cases}
\]
The evolution of the level set function is determined by the following equation,

\[ \phi_t + W \cdot \nabla \phi = 0, \]  

(5)

where \( W \) is a continuous extension of the normal velocity \( V_n \) of the interface, such that,

\[ W(x, t) = V_n(x^*, t), \]  

(6)

where \( x^* \) is the closest point on \( \Gamma(t) \) to the coordinates of a point \((x, t)\). The outward unit normal pointing in the direction of interface movement, \( n \), can be determined by the level set function \( \phi \) as follows,

\[ n = \frac{\nabla \phi}{|\nabla \phi|}. \]  

(7)

The level function \( \phi \) is initialized as a signed distance function

\[ \phi(x, 0) = \begin{cases} 
-d_I, & x \in \Omega^s_0, \\
0, & x \in \Gamma(0), \\
+d_I, & x \in \Omega^l_0, 
\end{cases} \]

where \( d_I \) is the distance of a point \( x \) to the interface \( \Gamma \). Updating the level set function by Eq. (5), may change the level set function away from being a distance function. Thus, it is necessary to perform a reinitialization procedure to keep the level set function as a distance function, which satisfies \( |\nabla \phi| = 1 \) (cf. [56]).

3. Space-time discretization. In this section, we follow the notation of, e.g., [53, 46], to introduce the definitions of the space-time domain, space-time slabs and space-time elements.

In a space-time discretization, we introduce a space-time domain \( \mathcal{E} \) as \( \mathcal{E} = \Omega \times [t_0, T] \in \mathbb{R}^{d+1} \). The coordinates of a point \( \bar{x} \in \mathcal{E} \) are defined as \( \bar{x} = (x, t_{d+1}) \) with the spatial variables \( x = (x_1, x_2, ..., x_d) \) and the time variable \( t = t_{d+1} \). The space-time domain boundary \( \partial \mathcal{E} \) consists of \( \Omega_0 := \{ \bar{x} \in \partial \mathcal{E} | t_{d+1} = t_0 \} \), \( \Omega_T := \{ \bar{x} \in \partial \mathcal{E} | t_{d+1} = T \} \), and \( \mathcal{E} := \{ \bar{x} \in \partial \mathcal{E} | t_0 < t_{d+1} < T \} \). First, we partition the time interval \([t_0, T]\) by the time levels \( 0 = t_0 < t_1 < ... < t_{E(t)} = T \). The space-time domain \( \mathcal{E} \) is then divided into \( E(t) \) space-time slabs. The \( n \)-th space-time slab is denoted as \( E^n = \mathcal{E} \cap I_n \), where \( I_n = [t_n, t_{n+1}] \) is the \( n \)-th time interval with length \( \Delta t_n = t_{n+1} - t_n \). Now, we describe the construction of the space-time elements in the space-time slab \( E^n \). Divide the spatial domain \( \Omega_{t_n} \) and \( \Omega_{t_{n+1}} \) into \( E^{(x)} \) non-overlapping spatial elements \( K^n \) and \( K^{n+1} \) with a uniformed size \( h \). A space-time element \( \mathcal{K}^n \) is then obtained by connecting \( K^n \) and \( K^{n+1} \). The element boundary \( \partial \mathcal{K} \) contains \( K^n, K^{n+1} \) and \( \mathcal{Q}^n \). On \( \partial \mathcal{K} \), the outward unit space-time normal is denoted by \( \bar{n}_{\mathcal{K}} = (n, n_{d+1}) \).

The Stefan problem on a Cartesian domain \( \Omega \) involves two different phases, i.e., \( \Omega^s_t \) and \( \Omega^l_t \). Thus, the space-time elements are classified into two categories: regular cells and cut-cells.

- **Regular cells.** A space-time element is referred to as a regular cell if it contains only one phase. We denote a regular cell as \( \mathcal{K}^{n,s}_j \) if the space-time element \( \mathcal{K}^n_j \) is in the solid phase. Similarly, \( \mathcal{K}^{n,l}_j \) denotes a regular cell which contains only liquid phase.
Cut-cells. A cell is called a cut-cell if a regular cell intersects with the moving front. In particular, a cut-cell contains different phases, which are separated by the phase boundary. Here, we denote a cut-cell as $K_{j}^{n,sl}$ if the space-time element $K_{j}^{n}$ intersects with the front $\Gamma$. The subelements associated with the cut-cell $K_{j}^{n,sl}$ are denoted as $K_{j}^{n,s,\Gamma}$ and $K_{j}^{n,l,\Gamma}$.

To generate a cut-cell in a space-time slab $E$, one can partition the time interval $[t_{n},t_{n+1}]$ by a set of Gauss-Lobatto points $(\tau_{0},\tau_{1},...,\tau_{p})$, and then determine the zero level set crossing values $\Gamma(\tau_{i}) (i = 0,...,p)$. By connecting $\Gamma(\tau_{i})$ using high order interpolation in time, we introduce a curve $Q_{i}$ cutting a regular cell into two subelements, i.e., $K_{j}^{n,s,\Gamma}$ and $K_{j}^{n,l,\Gamma}$. (See Fig. 2).

The space-time slab discretization is a tessellation of a domain $E$, which implies a division into non-overlapping, space-time-filling subsets. In the space-time slab $E$, the tessellation of the solid (liquid) phase is defined as $T_{h}^{n,s}$ ($T_{h}^{n,l}$), which consists of all regular cells, $K_{j}^{n,s}$ ($K_{j}^{n,l}$), and subelements of cut-cells, $K_{j}^{n,s,\Gamma}$ ($K_{j}^{n,l,\Gamma}$). Fig. 2 illustrates a tessellation of the space-time slab $E$ in a space-time domain, which is composed of tiles [16]. As we see in Fig. 2, regular cell, $K_{j}^{n,s}$, is one tile, while cut cells, $K_{j}^{n,s,\Gamma}$ and $K_{j}^{n,l,\Gamma}$, make up two tiles. Then the tessellation of the space-time slab $E$ is defined as $T_{h}^{n}$, such that $T_{h}^{n} = \{ T_{h}^{n,s}, T_{h}^{n,l} \}$. In addition, the tessellation of the space-time domain $E$ is denoted as $T_{h} = \bigcup_{n} T_{h}^{n}$.

In a space-time domain $E \in \mathbb{R}^{2}$, a sketch of a space-time slab $E$ with the moving interface $\Gamma(t)$ is demonstrated in Fig. 2.

4. Fixed frame of reference versus moving frame of reference. In a space-time slab $E$, a cut-cell $K_{j}^{n,sl}$ is generated due to the moving front $Q_{i}$ intersecting or embedding within a space-time element. Some examples of a cut-cell are shown in Fig. 3. A subelement generated in a cut-cell could be a triangle, a quadrilateral or a pentagon. Note that for the Stefan problem, a key issue is that the phase boundary is transported with a velocity that is proportional to the jump of the normal heat flux at the evolving and prior unknown boundary. Instead of dealing with implicitly defined arbitrary shaped space-time elements, we transform a prior unknown time-varying front into a known stationary front by applying the idea of Arbitrary Lagrangian–Eulerian Methods [37, 41]. First, two coordinate systems are introduced.

- Fixed frame of reference. The fixed frame of reference is the frame that we use to define the physical problem. The coordinates of the fixed frame of reference are denoted as $(\bf{x},t)$, where $\bf{x} = (x_{1},x_{2},...,x_{d})$ is the spatial variables. (See Fig. 4 (a, c)).

- Moving frame of reference. The moving frame of reference is the coordinate system that we use to define the transformed physical problem. In the moving frame of reference, the coordinates are defined as $(\bf{x},t)$. (See Fig. 4 (b, d)).

Next, we construct a mapping $\bf{\mathcal{M}}(\bf{x},t)$ between the moving frame of reference and the fixed frame of reference. In particular, the mapping will take a point on the time-varying front in the fixed frame of reference and transform it to a point on the stationary front relative to the moving frame of reference. For example, $\Gamma(t_{n})$ and $\Gamma(t_{n+1})$ in Fig. 4 (a or c) denote the zero level set at time $t_{n}$ and $t_{n+1}$, respectively. The transformation $\bf{\mathcal{M}}(\bf{x},t)$ transforms $\Gamma(t_{n})$ and $\Gamma(t_{n+1})$ to $\tilde{\Gamma}(t_{n})$ and $\tilde{\Gamma}(t_{n+1})$ in Fig. 4 (b or d), such that $\tilde{\Gamma}(t_{n}) = \Gamma(t_{n})$ and $\tilde{\Gamma}(t_{n+1}) = \Gamma(t_{n})$. Thus,
Figure 2. A tessellation of the space-time slab $\mathcal{E}^n$ of a space-time domain: $\mathcal{E} = \Omega \times [t_0, T] \in \mathbb{R}^2$. The coordinates of a point $\bar{x}$ are denoted by $\bar{x} = (x_1, x_2)$, where $x = x_1$ is the spatial variable and $x_2$ denotes the time variable. The square with bold lines is an example of a space-time slab $\mathcal{E}^n$, such that $\mathcal{E}^n = \mathcal{E} \cap I_n$, where $I_n = [t_n, t_{n+1}]$. We assume that the solid phase is on the left hand side of the interface $Q^n_\Gamma$, while the liquid is on the other side. The space-time element $K_{j}^{n,s}$ is referred to as a regular cell in the solid phase. However, $K_{j+1}^{n,sl}$ and $K_{j+2}^{n,sl}$ are examples of a cut-cell due to the time evolving interface, $Q^n_\Gamma$, defined on a nonconforming computational grid. In this example, the curve $Q^n_\Gamma$ obtained by 4th order interpolation cuts $K_{j+1}^{n,sl}$ into two subelements, i.e., a pentagon $K_{j+1}^{n,s,\Gamma}$ and a triangle $K_{j+1}^{n,l,\Gamma}$ in a space-time slab $\mathcal{E}^n$, our transformation $\bar{x} = \mathcal{M}^b(x, t)$ is determined by

\[
\begin{cases}
\dot{\mathcal{M}}^b_t + (\nabla \mathcal{M}^b) W = 0, \\
\mathcal{M}^b(x, t_n) = x,
\end{cases}
\] (8)

where $\nabla$ denotes the gradient operator with respect to $x$, and $W$ is defined in Eq. (6). Since the zero level set of the transformed front is determined by the front at time $t_n$, we refer to this transformation as the backward mapping transformation. Note that matrix $\nabla \mathcal{M}^b$ is the Jacobian of the transformation, which is defined by

\[
\nabla \mathcal{M}^b = \nabla \bar{x} = \\
\begin{pmatrix}
\frac{\partial \bar{x}_1}{\partial x_1} & \frac{\partial \bar{x}_1}{\partial x_2} & \cdots & \frac{\partial \bar{x}_1}{\partial x_d} \\
\frac{\partial \bar{x}_2}{\partial x_1} & \frac{\partial \bar{x}_2}{\partial x_2} & \cdots & \frac{\partial \bar{x}_2}{\partial x_d} \\
\vdots & \vdots & \ddots & \vdots \\
\frac{\partial \bar{x}_d}{\partial x_1} & \frac{\partial \bar{x}_d}{\partial x_2} & \cdots & \frac{\partial \bar{x}_d}{\partial x_d}
\end{pmatrix}.
\] (9)
After updating the temperature distribution in the moving frame of reference and calculating the updated interface curve \( \mathscr{L}^n \), we need to project the solutions back onto the fixed frame of reference. This projection \( x = \mathcal{M}^f(\tilde{x}, t) \) in a space-time slab \( \mathcal{E}^n \) is called the forward mapping transformation and is defined by

\[
\begin{align*}
\mathcal{M}^f - (\tilde{\nabla} \mathcal{M}^f)W &= 0, \\
\mathcal{M}^f(\tilde{x}, t_n) &= \tilde{x},
\end{align*}
\]

(10)

where \( \tilde{\nabla} \) denotes the gradient operator with respect to \( \tilde{x} \).

In one spatial dimension, we apply the method of characteristics to solve the transformations (8) and (10), and obtain,

- the backward transformation,

\[
\tilde{x}(x, t) = x - \int_{t_n}^{t} W(s)ds,
\]

(11)

- the forward transformation,

\[
x(\tilde{x}, t) = \tilde{x} + \int_{t_n}^{t} W(s)ds.
\]

(12)

By applying a mapping technique, the phase front becomes a straight line in the time direction in the moving frame of reference (see Fig. 4), which reduces the cost of quadrature on the implicitly defined surfaces and volumes. In one spatial dimension, the tessellation of the space-time domain becomes rectangular meshes in the moving frame of reference.
Figure 4. Examples of a mapping technique applied to two kinds of a cut-cell in a space-time domain $\mathcal{E} \in \mathbb{R}^2$. In each case, the curve $\mathcal{L}^n$ is mapped to a straight bold line $\tilde{\mathcal{L}}^n$, which is either aligned with an inter-element boundary or cutting a regular cell into two subelements.

4.1. The transformed equations in the moving frame of reference. In this section, we describe a general approach to transform the heat equations in the fixed frame of reference, Eqs. (1) and (2), into the moving frame of reference. In the moving frame of reference, the coordinates of a point is denoted as $(\tilde{x}, t)$. As the mapping procedure is identical in both phases, we consider only the solid phase, Eq. (1), and drop the index “s”,

$$\frac{\partial (\rho c_v \theta)}{\partial t} = \nabla \cdot (k \nabla \theta). \quad (13)$$

Given a mapping $\tilde{x} = \mathcal{M}^b(x, t)$, we find the effect of the mapping on Eq. (13) by use of the chain rule. First, the time derivative of $\theta$ transforms as,

$$\frac{\partial (\rho c_v \tilde{\theta})}{\partial t} = \frac{\partial (\rho c_v \tilde{\theta})}{\partial t} + \frac{\partial \tilde{x}}{\partial t} \cdot \nabla (\rho c_v \tilde{\theta}), \quad (14)$$

where $\tilde{\theta}$ denotes the temperature distribution in the moving frame of reference. For the gradient operator $\nabla \theta$, we have

$$\nabla \theta = (\nabla \tilde{x})^\top \nabla \tilde{\theta}, \quad (15)$$
where superscript “$^T$” denotes the matrix transpose, and $\nabla \tilde{x}$ is the Jacobian of the transformation, which is defined in Eq. (9). Similarly, we have

\[
\nabla \tilde{\theta} = (\nabla x)^T \nabla \theta.
\]

(16)

It then follows that

\[
\nabla \tilde{x} = (\nabla x)^{-1}.
\]

(17)

From (15), (16) and (17), we obtain

\[
\nabla \theta = ((\nabla x)^{-1})^T \nabla \tilde{\theta}.
\]

(18)

The transformation for the divergence operator, $\nabla \cdot F$, is defined as

\[
\nabla \cdot F = \tilde{\nabla} \tilde{F} : \nabla \tilde{x} = \tilde{\nabla} \tilde{F} : (\nabla x)^{-1},
\]

(19)

where the symbol “:” is defined as

\[
A : B = \sum_{j} \sum_{i} a_{i,j} b_{j,i},
\]

where $A \in \mathbb{R}^{d \times d}$ and $B \in \mathbb{R}^{d \times d}$ are matrices. From the definition of the backward mapping in Eq. (8), we have

\[
\frac{\partial \tilde{x}}{\partial t} = -(\nabla x)^{-1} W.
\]

(20)

By applying the transformations Equations (14) and (18) to (20), the transformed heat equation is then written as

\[
\partial \left(\rho c v \tilde{\theta}\right) \partial t - (\nabla x)^{-1} W \partial \left(\rho c v \tilde{\theta}\right) = \tilde{\nabla} \left(k((\nabla x)^{-1})^T \nabla \tilde{\theta}\right),
\]

(21)

Hence, the space-time formulation of Eq. (22) is written as

\[
\nabla \cdot \left(\rho c v B \tilde{\theta} - (\nabla x)^{-1} A \nabla \tilde{\theta}\right) = 0,
\]

(23)

where $\nabla$ is the space-time gradient operator with respect to $\tilde{x} = (x,t)$, $B \in \mathbb{R}^{1+1}$, and $A \in \mathbb{R}^{(1+1) \times (1+1)}$; $B$ and $A$ are defined as,

\[
B = \begin{pmatrix} -(\nabla x)^{-1} W & 1 \end{pmatrix}, \quad A = \begin{pmatrix} k((\nabla x)^{-1})^T & 0 \\ 0 \end{pmatrix}.
\]

(24)

4.1.1. Proposed extension of space-time mapping algorithm to multiple dimensions.

In multiple dimensions, one would solve (10) by the method of characteristics where a Picard iteration technique is applied to find the arrival points of the characteristics. At each iteration, one solves

\[
\frac{\partial}{\partial t}(x^{(k+1)}(\tilde{x},t)) = W(x^{(k)}(\tilde{x},t),t)
\]

with the initial guess $x^{(0)}(\tilde{x},t) = \tilde{x}$, which is the initial condition of Eq. (10). It is possible that the transformations become singular in multiple dimensions. It would then be necessary to reduce the spatial and temporal order as well as reduce $\Delta t_n$ if the condition number of $\nabla x$ exceeds a threshold. One can also use the condition number of $\nabla x$ as an indicator for adaptive mesh refinement.
Note that using vector identities, the transformed heat equation (21) in multiple dimensions is written as
\[
\frac{\partial (\rho c_v \tilde{\theta})}{\partial t} - \nabla \cdot ((\nabla x)^{-1} W \rho c_v \tilde{\theta}) = \nabla \cdot ((\nabla x)^{-1} k((\nabla x)^{-1})^\top \nabla \tilde{\theta}) + \mathcal{L} \tilde{\theta},
\]
where \( \mathcal{L} \tilde{\theta} \) is a non-conservative term defined as follows
\[
\mathcal{L} \tilde{\theta} \equiv \rho_c \tilde{\theta} \nabla \cdot ((\nabla x)^{-1}) W - (k((\nabla x)^{-1})^\top \nabla \tilde{\theta}) \cdot \nabla \cdot ((\nabla x)^{-1})^\top.
\]
The term \( \mathcal{L} \tilde{\theta} \) is linearized so that the nonlinear parts are frozen for each separate Picard iterate.

5. Space-time DG spectral element method.

5.1. Outline of the space-time DG spectral element method. We present an outline of our space-time DG spectral element method for solving the Stefan problem in Algorithm 1. Note that the transformed heat equation (22) in the moving frame of reference is a nonlinear equation since \( W \) in (6) and \( B \) in (24) are functions of the interfacial temperature gradients evaluated from both sides of the interface. We use the following expression to indicate the nonlinear property,
\[
W = W(\nabla \tilde{\theta}), \quad B = B(\nabla \tilde{\theta}).
\]
To solve such a nonlinear equation, we use a Picard iteration scheme for which at each Picard iteration a linear advection-diffusion equation is solved in the space-time framework locally. More details of the nonlinear system solver is discussed in section 5.1.1. In multiple dimensions especially, the level set equation (5) needs to be solved with spectral accuracy, which can be done by the method proposed by Sussman and Hussaini [55].

5.1.1. Nonlinear system solver. We use a Picard iteration scheme to solve the nonlinear system. The matrix of the linearized system in step 4 of Algorithm 1 is nonsymmetric. The solver we applied in our numerical tests is the direct solver from the LAPACK library. LAPACK is short for Linear Algebra PACKage, which provides routines for solving systems of linear equations, eigenvalue problems and singular value problems [4]. We predict that many iterative solvers for non-symmetric matrices can be applied to solve the linearized system, i.e., GMRES and BiCGSTAB. However, an efficient preconditioner needs to be developed for those iterative solvers.

Instead of a Picard iteration method, a Newton type method can be applied to solve the nonlinear system,
\[
A(\theta) \theta = b.
\]
One obtains the following by applying the Newton method,
\[
(\frac{\partial f}{\partial \theta})^k(\theta^{k+1} - \theta^k) = -A(\theta^k) \theta^k + b,
\]
where \( \frac{\partial f}{\partial \theta} \) is defined as follows,
\[
\frac{\partial f}{\partial \theta} = \theta \frac{\partial A(\theta)}{\partial \theta} + A(\theta).
\]
This method will be the same as the Picard iteration when \( \frac{\partial A(\theta)}{\partial \theta} \) is zero. To avoid the evaluation of this derivative, we chose the Picard iteration rather than a Newton type method. Also, the nonlinearity in the transformed equations is not strong so that a few iterations in the Picard iteration scheme leads to convergence.
Algorithm 1: The space-time discontinuous Galerkin spectral element method for solving the Stefan problem in a space-time slab $\mathcal{E}^n$.

1: Given $\theta^{n,s}$, $\theta^{n,l}$ and $\phi^n$ at time $t_n$ in the space-time slab $\mathcal{E}^n$.
2: Compute the normal velocity $V_n^t$ at the interface from Eq. (3). Details are given in section 5.4.
3: Define a space-time slab $\mathcal{E}^n$ in the moving frame of reference by introducing the backward transformation $\tilde{x} = \mathcal{M}^b(\tilde{x}, t)$ (Eq. 8) and forward transformation $x = \mathcal{M}^f(\tilde{x}, t)$ (Eq. (10)). Here, the temperature values in the solid and liquid phases are denoted as $\theta^{n,s}$ and $\theta^{n,l}$, respectively.
4: In $\mathcal{E}^n$, apply a Picard iteration scheme to solve (23) in each phase, and let the initial guess of the normal velocity to be $V_n^{j,(k)} = V_n^t$, where $k = 0$ and $j = 1, ..., p^t$. Note that $V_n^{j,(k)} = V_n^{n+1,(k)}$. In each phase, we solve

$$\nabla \cdot \left( \rho c_p B(\tilde{\theta}^{(k)}) \tilde{\theta}^{(k+1)} - (\tilde{\nabla} \tilde{x})^{-1} \mathbf{A} \nabla \tilde{\theta}^{(k+1)} \right) = 0. \quad (27)$$

5: while $(\|V_n^{n+1,(k+1)} - V_n^{n+1,(k)}\| > tol)$ do
   i. Use the DG spectral element method to solve the linear equation (27) in each phase. The discretization of (27) will be discussed in section 5.3.
   ii. Applying the forward transformation $\mathcal{M}^f(\tilde{x}, t)$ defined in (10), we obtain the level set function $\phi^{j,(k)}$ at time $t_j$, where $j = 1, ..., p^t$.
   iii. The backward transformation $\mathcal{M}^b(\tilde{x}, t)$ is applied to obtain $\theta^{j,s,(k+1)}$ and $\theta^{j,l,(k+1)}$, where $j = 1, ..., p^t$.
   iv. Compute $V_n^{j,(k+1)} (j = 1, ..., p^t)$, $\mathcal{M}^b(\tilde{x}, t)$ and $\mathcal{M}^f(\tilde{x}, t)$.
6: end while
7: Update the level set function, $\phi^t (j = 1, ..., p^t)$, using the forward transformation.
8: return $\theta^{n+1,s}$, $\theta^{n+1,l}$, $V_n^{n+1}$ and $\phi^{n+1}$.

5.2 Function spaces and notations. First, we introduce a transformation, $\tilde{x} = X(\xi, \eta)$, which defines a mapping that connects a one spatial dimension space-time element $\tilde{\mathcal{E}}$ ($\tilde{x} = (\tilde{x}, t) \in \tilde{\mathcal{E}}$) in the moving frame of reference and the reference square $\mathcal{R} = [-1, 1] \times [-1, 1] (\xi, \eta) \in \mathcal{R}$). In the space-time domain $\mathcal{E} \in \mathbb{R}^2$, each space-time element in the moving frame of reference is a square. So the transformation $(\tilde{x}, t) = X(\xi, \eta)$ [36] is linear in each coordinate direction and is written as

$$X(\xi, \eta) =\frac{1}{4}\left\{ x^1(1 - \xi)(1 - \eta) + x^2(1 + \xi)(1 - \eta) \right. \\
\left. x^3(1 + \xi)(1 + \eta) + x^4(1 - \xi)(1 + \eta) \right\}, \quad (31)$$

where $\{x^j\}_{j=1,...,4}$ are four corners of a space-time element $\tilde{\mathcal{E}}$, which are numbered counter clockwise.

Let’s introduce two function spaces, $V_h$ and $\Sigma_h$, associated with the tessellation $\tilde{\mathcal{H}}$ defined in the moving frame of reference,

$$V_h = \{ v \in L^2(\tilde{\mathcal{E}}) : v|_{\tilde{x}} \circ X_{\tilde{x}} \in \mathcal{P}^p(\mathcal{R}), \forall \tilde{x} \in \tilde{\mathcal{H}} \}, \quad (32)$$

$$\Sigma_h = \{ \sigma \in (L^2(\tilde{\mathcal{E}}))^{1+1} : \sigma|_{\tilde{x}} \circ X_{\tilde{x}} \in (\mathcal{P}^p(\mathcal{R}))^{1+1}, \forall \tilde{x} \in \tilde{\mathcal{H}} \}, \quad (33)$$

where $\mathcal{P}^p(\mathcal{R})$ is the set of all polynomials of degree at most $p = (p^{(x)}, p^{(t)})$ on $\mathcal{R}$, with $p^{(x)}$ in the spatial direction and $p^{(t)}$ in the time direction.
Spectral element methods with non-periodic boundaries use orthogonal polynomial approximations of the solution [10, 36]. In the space-time domain $\hat{E} \in \mathbb{R}^3$, we introduce a tensor product basis, $\ell_i(\xi)\ell_j(\eta) \in \mathcal{P}^p(\hat{E})$, such that

$$
\ell_i(\xi) = \prod_{k=0}^{p(s)} \frac{\xi - \xi_k}{\xi_i - \xi_k}, \quad \ell_j(\eta) = \prod_{k=0}^{p(t)} \frac{\eta - \eta_k}{\eta_j - \eta_k},
$$

where $\{\xi_i\}_{i=0,\ldots,p(s)}$ and $\{\eta_j\}_{j=0,\ldots,p(t)}$ are the sets of Legendre Gauss-Lobatto points.

Due to the discontinuous approximation spaces, the temperature and the temperature gradient approximations are double valued across element boundaries. Thus, we introduce the average $\{\cdot\}$ and jump $[\cdot]$ operators. Considering two adjacent elements $\mathcal{K}_j^n$ and $\mathcal{K}_{j+1}^n$ such that $\partial_{E,j} = \mathcal{K}_j^n \cap \mathcal{K}_{j+1}^n$. Let $\bar{n}_{\mathcal{K}_j^n}$ and $\bar{n}_{\mathcal{K}_{j+1}^n}$ denote the corresponding outward unit normal of $\mathcal{K}_j^n$ and $\mathcal{K}_{j+1}^n$ on $\partial_{E,j}$. Then the average $\{\nu\}$ and jump $[\nu]$ operators are defined as,

$$
\{\nu\} = \left( \nu^+_\mathcal{K}_j^n + \nu^-_{\mathcal{K}_{j+1}^n} \right)/2, \quad \{\tau\} = \left( \tau^+_\mathcal{K}_j^n + \tau^-_{\mathcal{K}_{j+1}^n} \right)/2, \quad [\nu] = \nu^+_\mathcal{K}_j^n \cdot \bar{n}_{\mathcal{K}_j^n} + \nu^-_{\mathcal{K}_{j+1}^n} \cdot \bar{n}_{\mathcal{K}_{j+1}^n}, \quad [\tau] = \tau^+_\mathcal{K}_j^n \cdot \bar{n}_{\mathcal{K}_j^n} + \tau^-_{\mathcal{K}_{j+1}^n} \cdot \bar{n}_{\mathcal{K}_{j+1}^n},
$$

where $\nu^+_\mathcal{K}_j^n$ denotes the left value at the element interface, and $\nu^-_{\mathcal{K}_{j+1}^n}$ denotes the right value at the element interface.

5.3. Discretization of DG spectral element method. To derive the space-time DG weak formulation for Eq. (23), we follow the same approach presented in [15]. By introducing an auxiliary variable $\tilde{\sigma}$, we rewrite Eq. (23) into a first order system

$$
\tilde{\sigma} = A \nabla \tilde{\theta},
$$

$$
\nabla \cdot (\rho_c B \tilde{\theta} - (\nabla \tilde{\theta})^{-1} \tilde{\sigma}) = 0.
$$

Multiply (37) with a test function $\tau \in \mathcal{S}_h$, and then integrate over a space-time element $\mathcal{K}^s$, we obtain

$$
\int_{\mathcal{K}^s} \tilde{\sigma} \cdot \tau d\mathcal{K}^s = \int_{\mathcal{K}^s} A \nabla \tilde{\theta} \cdot \tau d\mathcal{K}^s.
$$

After integration by parts twice, we have the following formulation,

$$
\int_{\mathcal{K}^s} \tilde{\sigma} \cdot \tau d\mathcal{K}^s = \int_{\mathcal{K}^s} A \nabla \tilde{\theta} \cdot \tau d\mathcal{K}^s + \int_{\partial \mathcal{K}^s} A(\tilde{\theta}^D - \tilde{\theta}) \bar{n} \cdot \tau d\mathcal{K}^s,
$$

where $\bar{n}$ is the unit normal at the interface pointing in the direction of interface movement in a space-time slab $\tilde{E}^n$. $\tilde{\theta}^D$ denotes the numerical flux, which should be carefully defined to ensure stability of the method.

Similarly, the weak formulation of (38) for a space-time element $\mathcal{K}^s$ is obtained by first multiplying (38) with a test function $\nu \in V_h$, and then performing integration by parts once.

$$
\int_{\mathcal{K}^s} \left( \rho_c B \tilde{\theta} - (\nabla \tilde{\theta})^{-1} \tilde{\sigma} \right) \cdot \nabla \nu d\mathcal{K}^s = \int_{\partial \mathcal{K}^s} \left( \rho_c B \tilde{\theta}^A - (\nabla \tilde{\theta})^{-1} \tilde{\sigma}^A \right) \cdot \bar{n} \nu d\mathcal{K}^s = 0.
$$

(41)
where \( \hat{\theta}^A \) and \( \hat{\sigma}^D \) are the numerical fluxes. We separate the numerical fluxes into an advection flux \( \hat{\theta}^A \) and the diffusion fluxes \( (\hat{\theta}^D, \hat{\sigma}^D) \). For the advection flux, we apply the upwind flux, as described in [53].

\[
B \hat{\theta}^A = \{\{B \hat{\theta}\}\} + \frac{1}{2} |B \cdot n| \parallel\hat{\theta}\parallel.
\]  
(42)

For the diffusion fluxes, we choose the LDG [15] fluxes with the stabilization parameter \( \alpha_j = 0 \),

\[
\hat{\theta}^D = \{\{\hat{\theta}\}\} - \beta \cdot J \hat{\theta},
\]  
(43)

\[
\hat{\sigma}^D = \{\{\hat{\sigma}\}\} + \beta \cdot J \hat{\sigma} - \alpha_j \parallel\hat{\theta}\parallel,
\]  
(44)

where \( \beta \cdot n = \text{sign}(v \cdot n)/2 \) [14], with \( v = (1, 1)^\top \).

Note that special care is needed when imposing the boundary conditions through \( \hat{\theta}^D \). The reason is that the boundary conditions must be the exact values imposed in the moving frame of reference, which are obtained by applying the backward transformation on the boundary conditions imposed in the fixed frame of reference. Since the backward transformation (Eq. (8)) depends on the interface velocity, we have

\[
(\hat{\theta}^D)_{(\partial E_{n,s})} = (\hat{\theta}^D(\nabla \hat{\theta}))_{(\partial E_{n,s})}.
\]  
(45)

5.4. Calculating the interface velocity. In the Stefan problem, the phase boundary is evolving in time with the velocity defined in Eq. (3). The accuracy of the interface velocity directly affects the accuracy of the interface position. The interface velocity is a function of the temperature gradient evaluated at both sides of the interface. The order of accuracy for approximating the interface velocity must be commensurate with the order for approximating the solutions of the heat equations, otherwise the overall order of accuracy will reduce to the minimum of the velocity or temperature order.

In order to obtain overall spectral accuracy, the interface velocity must be computed by a spectrally accurate method. We propose two options for computing the interface velocity. We find that each option leads to a spectrally accurate method for computing the interface velocity.

Since the velocity is determined by the temperature gradient evaluated at the interface in both phases, one option is to approximate the interface normal velocity using the following expression

\[
V_n = \frac{1}{\rho L} (\hat{\sigma}^s - \hat{\sigma}^l) \cdot \bar{n},
\]  
(46)

where \( \bar{n} \) is the unit normal at the interface pointing in the direction of interface movement in a space-time slab \( \hat{E}^n \). Note that \( \hat{\sigma} \) is the auxiliary variable introduced in (37). Now, we show that Eq. (46) is equivalent to Eq. (3) defined in the fixed frame of reference by,

\[
\hat{\sigma} \cdot \bar{n} = A \nabla \hat{\theta} \cdot \bar{n},
\]

\[
= k((\nabla x)^{-1})^\top \nabla \hat{\theta} \cdot \bar{n},
\]

\[
= k \nabla \hat{\theta} \cdot n,
\]  
(47)
where \( \mathbf{n} \) is the unit normal at the interface pointing in the direction of interface movement in the fixed frame of reference. We refer to Eq. (46) as the weak form method.

Another option is to apply the normal probe method in the fixed frame of reference. In a space-time slab \( \mathcal{S}^n \in \mathbb{R}^2 \), the interface curve \( \mathcal{P}^n \) is updated by the forward transformation, while the temperature values \( \theta(x,t) \) in the fixed frame of reference are obtained by the backward transformation.

Note that the transformations (11) and (12) are discretized by a \( (p^t + 1) \)-point Gaussian quadrature rule, where \( p^t \) is the polynomial order in the time direction. In order to obtain the temperature values \( \theta(x,t) \) in the fixed frame of reference, we first find the departure points (i.e., \( \tilde{x} \)) of the Legendre Gauss-Lobatto nodes (i.e., \( x \)) in the fixed frame of reference by the backward transformation (11), then the temperature values \( \theta(x,t) \) are obtained via a \( p^x \)-th order accurate interpolation, such that

\[
\theta(x,t) = I_m(\tilde{\theta}(\tilde{x},t)),
\]

where \( I_m \) is the interpolation operator due to the mapping, and \( p^x \) is the polynomial order in the spatial direction. According to the zero level set at time \( t_j \) (\( j = 1, ..., p^t \)), we construct a fictitious element in each phase with the size \( h \) that is the length of a regular space-time element in the spatial direction, i.e., Fig. 5 is an example of the normal probe method applied at time \( t_{p^t} \). In each fictitious element, the temperature values, \( \theta^{fic}(x,t) \), at its Gauss-Lobatto nodes are obtained by

\[
\theta^{fic}(x,t) = I_n(\theta(x,t)),
\]

where \( I_n \) is the interpolation operator due to the normal probe method. The normal interface velocity is then computed by

\[
V_n = \frac{1}{\rho L} \left\{ \nabla \left( I_n(I_m(\tilde{\theta}^{n+1,s})) \right) - \nabla \left( I_n(I_m(\tilde{\theta}^{n+1,l})) \right) \right\} \cdot \mathbf{n}.
\]

The aim of using fictitious elements is to avoid computing the temperature gradient in a small cell, which can eliminate the instability caused by computing the temperature gradient directly (i.e., spectral collocation discretization) in a small cell. The description of the normal probe method is illustrated in Fig. 5.

6. Numerical experiments. In this section, we test the space-time DG spectral element method for solving the Stefan problem on two benchmark problems in one spatial dimension. The numerical results are compared with the exact solutions in order to examine the convergence rate both with respect to space and time. The time step size is restricted by the following CFL criterion

\[
\Delta t = C \frac{h}{V_n},
\]

where \( C \) is a constant and set to be less than 1.0, and \( h \) is the length of a regular space-time element. This CFL condition (51) makes sure that the interface does not move across two regular space-time elements in one space-time slab.

As we mentioned in section 5.1, we apply a direct solver from the LAPACK library in order to invert the resulting linear system in each space-time slab. In
Figure 5. Description of the normal probe method applied in a space-time slab $E^n \in \mathbb{R}^2$ in the fixed frame of reference. The polynomial order, $p = (p(x), p(t))$, in each space-time element is chosen to be $(3,2)$. We assume that the solid phase is on the left hand side of the interface $Q^n$ while the liquid is on the other side. The fictitious element in the solid phase at time $t_{n+1}$ is denoted as $e_{fic}^{n+1,s}$, and $e_{fic}^{n+1,l}$ is the fictitious element for the liquid phase at time $t_{n+1}$. The length of both fictitious elements is $h$, which is the same as the length of a regular space-time element, i.e., the length of $K_{j-1}$. The number of Gauss-Lobatto nodes in each fictitious element is $p+1$, where $p = p(x)$. The temperature gradient is evaluated at the node with symbol “□” in each fictitious element.

The last space-time slab $E^{(t)-1}$, the errors are measured in the $L^\infty$ norms at time $t_{E(t)}$, i.e., $t_{E(t)} = T$, where $T$ is the final computational time.

$$
\|Err\theta\|_{\infty} = \max_{i=0,\ldots,p(x)} \|\theta_{E(t)} - (\theta_h)_{E(t)}\|_i,
$$

(52)

where $\theta_{E(t)}$ denotes the exact temperature solution evaluated at node $x_i$, and $(\theta_h)_{E(t)}$ is the approximation temperature evaluated at the same node.

6.1. Test 1 – constant velocity. Consider a computational domain $\Omega = [-0.5, 1.0]$ with an exact solution

$$
\theta(x,t) = \begin{cases} 
-1 + e^{-V_n(x-V_n t)}, & x > V_n t, \\
0, & x \leq V_n t,
\end{cases}
$$

(53)

where the normal velocity $V_n$ is a constant, and we take $V_n = 1.0$. The interface is governed by $\Gamma(t) = V_n t$.

In this test, the ice phase is on the left and motionless, existing at the melting temperature, i.e., $\theta^s = \theta_m = 0$, while the liquid phase is on the right and initialized as a supercooled liquid. The interface between the ice and liquid phase is initialized at the origin, $\Gamma(0)$, and moves to the right with the normal velocity $V_n$ defined in Eq. (3). We take $\rho^l = \rho^s = 1.0$, $k^l = k^s = 1.0$, $c^l_v = c^s_v = 1.0$ and $L = 1.0$. Dirichlet boundary conditions for temperature distributions are enforced on the domain’s boundary $\partial \Omega$ using the exact solution. At the interface, we choose $\theta_i = \theta_m$. 

First, we demonstrate the spectral accuracy of the approximation by plotting the maximum errors in the temperature and the interface velocity as a function of polynomial order, \( p = (p(x), p(t)) \), respectively. The simulation is computed over the time \( t = 0 \) to \( t = 0.5 \) with the time step \( \Delta t = 3.84 \times 10^{-2} \). The polynomial orders in space and in time are chosen to be the same, i.e., \( p(x) = p(t) \). The computational domain is divided into \( E(x) = 5 \) and \( E(x) = 20 \) elements. Two options for approximating the interface velocity, i.e., the weak form method (46) and the normal probe method (50) are compared in Fig. 6. Through the comparison of the temperature (left) and the interface velocity (right), we see that both options show exponential convergence of the computed temperature to the exact solution, as well as the interface velocity.

The overall accuracy of our method is spectrally accurate. However, the convergence rates of temperature and the interface velocity are different as shown in Fig. 6. This is because the velocity is determined by the temperature gradient evaluated at the interface in both phases. The convergence rate of interface velocity is then the same as the one of temperature gradient, which is one order less than the convergence rate of temperature.

The errors stagnate after polynomial order 6 for the test cases with \( E(x) = 20 \) (Fig. 6) because the error hits the asymptotic region where the truncation error dominates the computations. The error stagnates around \( 10^{-13} \); we hypothesize that this is due to the accumulation of round-off errors which interact with the error in the position and weights for the Legendre-Gauss points. The Legendre-Gauss points are computed by the method described by Kopriva [36]. The errors in both Fig. 4.8 on page 120 and Fig. 4.12 on page 135 of Kopriva’s book ([36]) stagnate around \( 10^{-13} \).

Note that the DG spectral element method is applied on the transformed heat equations in both phases in the moving frame of reference (Algorithm 1). If we transform the exact solution (53) into the moving frame of reference by the backward transformation (11), we have

\[
\tilde{\theta}(\tilde{x}, t) = \begin{cases} 
-1 + e^{-V_n(\tilde{x})}, & \tilde{x} > 0, \\
0, & \tilde{x} \leq 0.
\end{cases}
\]  

(54)

So the convergence rate does not depend on the polynomial order \( p(t) \) in this test problem, which is shown in Figs. 7 and 8. We choose three values of the polynomial order in time, i.e., \( p(t) = p(x) \), \( p(t) = 2 \) and \( p(t) = 10 \). All cases lead to the same error behavior, which is consistent with the exact solution (54) in the moving frame of reference. Fig. 7 applies the weak form method (46) to calculate the interface velocity, while Fig. 8 applies the normal probe method (50).

In Table 1 and Table 2, we list the maximum errors of the temperature, interface position and interface velocity. We also report the number of required Picard iterations in the last space-time slab as a function of the polynomial order in Table 1 and Table 2. The computational domain is divided into \( E(x) = 5 \) elements in the spatial direction, and the time step is chosen to be \( \Delta t = 3.84 \times 10^{-2} \). We choose the tolerance of the Picard iteration to be \( 10^{-13} \) (Algorithm 1). The errors in Table 1 are obtained using the weak form velocity algorithm (46), while Table 2 uses the normal probe velocity algorithm (50). From Table 1 and Table 2, we see that only a few (i.e., less than 15) iterations lead to convergence no matter which option we applied to compute the interface velocity.
Figure 6. Errors in the temperature (left) and the interface velocity (right) as a function of polynomial order \( p(x) \) in the liquid phase. The polynomial order in time direction, \( p(t) \), is chosen to be the same as \( p(x) \). The comparison is made between the weak form method (46) and the normal probe method (50). The simulation is computed over the time \( t = 0 \) to \( t = 0.5 \) with the time step \( \Delta t = 3.84 \times 10^{-2} \).

Figure 7. Errors in the temperature (left) and the interface velocity (right) as a function of polynomial order \( p(x) \) in the liquid phase. The simulation is computed over the time \( t = 0 \) to \( t = 0.5 \). The computational domain is divided into \( E(x) = 5 \) in spatial direction, with \( E(x)_l = 4 \) in the liquid in the beginning and \( E(x)_l = 2 \) in the liquid at the end. The interface velocity is computed by the weak form method (46). The time step is chosen to be \( \Delta t = 3.84 \times 10^{-2} \).
Figure 8. Errors in the temperature (left) and the interface velocity (right) as a function of polynomial order $p(x)$ in the liquid phase. The simulation is computed over the time $t = 0$ to $t = 0.5$. The computational domain is divided into $E^{(x)} = 5$ in spatial direction, with $E^{(x),l} = 4$ in the liquid in the beginning and $E^{(x),l} = 2$ in the liquid at the end. The interface velocity is computed by the normal probe method (50). The time step is chosen to be $\Delta t = 3.84 \times 10^{-2}$.

Table 1. Errors of the temperature ($\|\text{Err}_\theta\|_\infty$), interface position ($\|\text{Err}_\Gamma\|_\infty$) and interface velocity ($\|\text{Err}_V\|_\infty$) in the last space-time slab. The interface velocity is computed by applying the weak form method (46). The number of the Picard iterations is listed in the last column with $tol = 10^{-13}$.

| $p = (p^{(x)}, p^{(t)})$ | $\|\text{Err}_\theta\|_\infty$ | $\|\text{Err}_\Gamma\|_\infty$ | $\|\text{Err}_V\|_\infty$ | iter |
|--------------------------|-----------------|-----------------|-----------------|------|
| (2,1)                    | 1.24E-003       | 1.75E-003       | 6.12E-003       | 11   |
| (3,1)                    | 5.16E-005       | 6.40E-005       | 2.68E-004       | 11   |
| (4,1)                    | 1.28E-006       | 1.47E-006       | 7.20E-006       | 9    |
| (5,1)                    | 2.40E-008       | 2.63E-008       | 1.44E-007       | 8    |
| (6,1)                    | 3.77E-010       | 4.02E-010       | 2.31E-009       | 5    |

Table 2. Errors of the temperature ($\|\text{Err}_\theta\|_\infty$), interface position ($\|\text{Err}_\Gamma\|_\infty$) and interface velocity ($\|\text{Err}_V\|_\infty$) in the last space-time slab. The interface velocity is computed by applying the normal probe method (50). The number of the Picard iterations is listed in the last column with $tol = 10^{-13}$.

| $p = (p^{(x)}, p^{(t)})$ | $\|\text{Err}_\theta\|_\infty$ | $\|\text{Err}_\Gamma\|_\infty$ | $\|\text{Err}_V\|_\infty$ | iter |
|--------------------------|-----------------|-----------------|-----------------|------|
| (2,1)                    | 3.89E-003       | 5.29E-003       | 1.34E-002       | 12   |
| (3,1)                    | 1.55E-004       | 1.85E-004       | 5.08E-004       | 12   |
| (4,1)                    | 3.73E-006       | 4.16E-006       | 1.25E-005       | 10   |
| (5,1)                    | 6.80E-008       | 7.32E-008       | 2.29E-007       | 6    |
| (6,1)                    | 9.97E-010       | 1.05E-009       | 3.47E-009       | 5    |
Now, we fix the polynomial order $p$ but vary the grid size $h$. The results are shown in Table 3. From Table 3, we observe that the $L^\infty$ norm of the temperature distribution is of order $p$ when polynomials of degree $(p, p)$ are used. This is referred to as “sub-optimal” convergence since the convergence rate is not the classical rate for a $p$th order approximation which is $(p + 1, p + 1)$. The reason for sub-optimal convergence is that the accuracy of the temperature distribution is impacted by the accuracy of the interface velocity, which in turn is determined by the temperature gradient evaluated at the interface separating the two phases. In other words, the interface velocity, which is computed by (3), is a polynomial of degree $p - 1$ if the temperature is approximated by a polynomial of degree $p$. Thus, the overall accuracy is of order $p$ when polynomials of degree $(p, p)$ are used.

**Table 3.** Order of convergence of the temperature ($\|\text{Err}_\theta\|_\infty$) for both options in the last space-time slab.

| Order of convergence | $E^{(x)}$ | Weak form (46) | Normal prob (50) |
|----------------------|-----------|----------------|-----------------|
| $p = (p^{(x)}, p^{(t)})$ |           |                |                 |
| (4,4)                | 5         | 3.58           | 3.79            |
|                      | 10        | 4.04           | 3.85            |
|                      | 20        |                |                 |
| (5,5)                | 5         |                |                 |
|                      | 10        | 4.65           | 4.82            |
|                      | 20        | 5.38           | 4.89            |

6.2. **Test 2 – variable velocity.** Consider a computational domain $\Omega = [0, l]$ with an exact solution [3, 62],

$$\theta = \theta_{\text{wall}} + \frac{\theta_m - \theta_{\text{wall}}}{\text{erf}(\lambda)} \text{erf}\left(\frac{x}{2\sqrt{\lambda}t}\right),$$

$$\Gamma(t) = 2\lambda\sqrt{\alpha t},$$

where $\alpha = k_s/(\rho_s c_s v)$, $\theta_{\text{wall}}$ is the Dirichlet boundary condition at $x = 0$, $\text{erf}(-)$ is the error function and $\lambda$ is a solution to the transcendental equation.

$$\lambda \exp(\lambda^2)\text{erf}(\lambda) = \frac{c_v^s(\theta_{\text{wall}} - \theta_m)}{L\sqrt{\pi}}. \quad (57)$$

In this test, the ice phase is on the left, while the liquid phase is on the right and motionless, existing at the melting temperature, i.e., $\theta_l = \theta_m$. The interface between the ice and liquid phase is initialized at $\Gamma(0) = 0.1$ and moves to the right with the normal velocity $V_n$ defined in Eq. (3). Dirichlet boundary conditions for temperature distributions are enforced on the domain’s boundary $\partial \Omega$ using the exact solution.

We set $\rho^l = \rho^s$ and $c_v^l = c_v^s$. Now, we derive the following non-dimensional form,

$$\frac{\partial \Theta_s}{\partial \tau} = \frac{\partial^2 \Theta_s}{\partial \zeta^2}, \quad 0 < \zeta < \zeta_r \quad (58)$$

$$\frac{\partial \Theta_l}{\partial \tau} = K_{ls} \frac{\partial^2 \Theta_l}{\partial \zeta^2}, \quad \zeta_r < \zeta < 1 \quad (59)$$

$$V_n = \text{St} \left( \frac{\partial \Theta_s}{\partial \zeta} - K^{st} \frac{\partial \Theta_l}{\partial \zeta} \right), \quad \zeta = \zeta_r. \quad (60)$$
with Dirichlet boundary conditions
\[ \Theta(0, \tau) = 1.0, \quad \zeta = 0, \quad (61) \]
\[ \Theta(\zeta, \tau) = 0.0, \quad \zeta = \zeta. \quad (62) \]
Here \( \zeta = x/l, \tau = \alpha(t/l^2) \), \( \Theta = (\theta - \theta_m) / \Delta \theta \), \( \zeta = \Gamma/l \), \( K_{ls} = k_l / k_s \) and \( St = (c_v \Delta \theta) / L \), where \( l \) is the length of the domain and set to be 1, \( \Delta \theta = \theta_{wall} - \theta_m \) and \( \Gamma \) is the phase boundary position. Note that \( St \) is called the Stefan number.

First, we demonstrate the spectral accuracy of our method by plotting the maximum errors in the temperature and the interface velocity as a function of polynomial order, \( p = (p(x), p(t)) \), respectively. The computational domain is divided into \( E(x) = 6 \), and \( E(x) = 24 \) elements. The Stefan number \( St \) is set to be 0.02, and the interface moves from \( x = 0.1 \) to \( x = 0.3 \). Two options for approximating the interface velocity, i.e., the weak form method (46) and the normal probe method (50) are compared in Fig. 9. In Fig. 9, the left part reveals the exponential convergence of the temperature, while the exponential convergence of the interface velocity is observed from the right part. We see that both options exhibit spectral accuracy.

![Figure 9](image)

**Figure 9.** Errors in the temperature (left) and the interface velocity (right) as a function of polynomial order \( p(x) \) in the ice phase. The comparison is made between the weak form method (46) and the normal probe method (50). The interface starts at \( x = 0.1 \) and moves to \( x = 0.3 \). The polynomial order in time direction, \( p(t) \), is chosen to be 10 so that the temporal errors are negligible. The time step is chosen to be \( \Delta t = 4.9 \times 10^{-2} \). The Stefan number \( St \) is set to be 0.02.

Now, we examine maximum errors in the temperature and the interface velocity as a function of temporal order \( p(t) \). Fig. 10 applies the weak form method (46) to compute the interface velocity, while Fig. 11 uses the normal probe method (50). The number of the space-time slab is denoted by \( E(t) \), which is chosen to \( E(t) = 41 \), \( E(t) = 81 \) and \( E(t) = 162 \). The interface starts at \( x = 0.1 \) and moves to \( x = 0.3 \). Fig. 10 and Fig. 11 are shown the exponential convergence in time by fixing the polynomial order in space, \( p(x) \), to be 10 so that the spatial errors are negligible.
Figure 10. Errors in the temperature (left) and the interface velocity (right) as a function of polynomial order $p(t)$ in the ice phase. The interface velocity is computed by the weak form method (46). The interface starts at $x = 0.1$ and moves to $x = 0.3$. The comparison is made among different numbers of the space-time slab, i.e., $E(t) = 41$, $E(t) = 81$ and $E(t) = 162$. The corresponding time step is $\Delta t = 4.9 \times 10^{-2}$, $\Delta t = 2.5 \times 10^{-2}$ and $\Delta t = 1.2 \times 10^{-2}$. The polynomial order in spatial direction, $p(x)$, is chosen to be 10 so that the spatial errors are negligible. The computational domain is divided into $E(x) = 5$ in the spatial direction. The Stefan number $St$ is set to be 0.02.

In Table 4 and Table 5, we show the maximum errors of the temperature, interface position and interface velocity. We also report the number of required Picard iterations in the last space-time slab as a function of the polynomial order in Table 4 and Table 5. The computational domain is divided into $E(x) = 6$ elements in the spatial direction, and the time step is chosen to be $\Delta t = 4.9 \times 10^{-2}$. We choose the tolerance for the Picard iteration to be $10^{-15}$ (Algorithm 1). The errors in Table 4 are obtained using the weak form method (46) for computing the interface velocity, while Table 5 uses the normal probe method (50). From Table 1 and Table 2, it shows that only a few (i.e., less than 10) iterations lead to convergence for different combinations of the polynomial order no matter which option we applied to compute the interface velocity.

Note that the solution to the Stefan problem is quasi-steady. For example, the interface moves relatively rapidly if the Stefan number $St$ is large (see [48]). We vary the Stefan number, i.e., $St = 0.001$, $St = 0.02$, $St = 0.05$ and $St = 0.08$. Fig. 12 applies the weak form method (46) to compute the interface velocity, while Fig. 13 uses the normal probe method (50). The interface starts at $x = 0.1$ and moves to $x = 0.4$. Fig. 12 and Fig. 13 are shown the exponential convergence for different values of $St$.

7. Summary and conclusions. A space-time DG spectral element method for solving the Stefan problem has been presented in one spatial dimension. The
A SPACE-TIME DGSEM FOR THE STEFAN PROBLEM

Figure 11. Errors in the temperature (left) and the interface velocity (right) as a function of polynomial order $p(t)$ in the ice phase. The interface velocity is computed by the normal probe method (50). The interface starts at $x = 0.1$ and moves to $x = 0.3$. The comparison is made among different numbers of the space-time slab, i.e., $E(t) = 41$, $E(t) = 81$ and $E(t) = 162$. The corresponding time step is $\Delta t = 4.9 \times 10^{-2}$, $\Delta t = 2.5 \times 10^{-2}$ and $\Delta t = 1.2 \times 10^{-2}$. The polynomial order in spatial direction, $p(x)$, is chosen to be 10 so that the spatial errors are negligible. The computational domain is divided into $E(x) = 5$ in the spatial direction. The Stefan number $St$ is set to be 0.02.

Table 4. Errors of the temperature ($\|\text{Err}_\theta\|_\infty$), interface position ($\|\text{Err}_\Gamma\|_\infty$) and interface velocity ($\|\text{Err}_{V_n}\|_\infty$) in the last space-time slab. The interface velocity is computed by applying the weak form method (46). The computational domain is divided into $E(x) = 6$ in the spatial direction. The interface starts at $x = 0.1$ and moves to $x = 0.3$. The time step is chosen to be $\Delta t = 4.9 \times 10^{-2}$. The number of the Picard iteration is listed in the last column with $tol = 10^{-15}$.

| $p = (p^{(x)}, p^{(t)})$ | $\|\text{Err}_\theta\|_\infty$ | $\|\text{Err}_\Gamma\|_\infty$ | $\|\text{Err}_{V_n}\|_\infty$ | iter |
|--------------------------|---------------------|---------------------|---------------------|-----|
| (3,3)                    | 3.86E-006           | 8.44E-007           | 1.87E-007           | 7   |
| (3,10)                   | 2.46E-006           | 7.67E-008           | 1.67E-008           | 7   |
| (4,4)                    | 3.82E-009           | 9.46E-010           | 2.11E-010           | 7   |
| (4,10)                   | 2.41E-009           | 1.74E-010           | 3.98E-011           | 7   |
| (5,5)                    | 1.48E-009           | 4.72E-010           | 1.05E-010           | 7   |
| (5,10)                   | 4.18E-010           | 1.28E-010           | 2.84E-011           | 7   |
| (6,6)                    | 3.07E-012           | 9.52E-013           | 2.12E-013           | 7   |
| (6,10)                   | 1.14E-012           | 3.47E-013           | 7.79E-014           | 7   |
| (7,7)                    | 3.19E-013           | 9.91E-014           | 2.19E-014           | 7   |
| (7,10)                   | 1.46E-013           | 4.55E-014           | 1.09E-014           | 7   |

TABLE 4. Errors of the temperature ($\|\text{Err}_\theta\|_\infty$), interface position ($\|\text{Err}_\Gamma\|_\infty$) and interface velocity ($\|\text{Err}_{V_n}\|_\infty$) in the last space-time slab. The interface velocity is computed by applying the weak form method (46). The computational domain is divided into $E(x) = 6$ in the spatial direction. The interface starts at $x = 0.1$ and moves to $x = 0.3$. The time step is chosen to be $\Delta t = 4.9 \times 10^{-2}$. The number of the Picard iteration is listed in the last column with $tol = 10^{-15}$. 
Table 5. Errors of the temperature ($||\text{Err}_\theta||_\infty$), interface position ($||\text{Err}_\Gamma||_\infty$) and interface velocity ($||\text{Err}V_n||_\infty$) in the last space-time slab. The interface velocity is computed by applying the normal probe method (50). The computational domain is divided into $E(x) = 6$ in the spatial direction. The interface starts at $x = 0.1$ and moves to $x = 0.3$. The time step is chosen to be $\Delta t = 4.9 \times 10^{-2}$. The number of the Picard iteration is listed in the last column with $tol = 10^{-15}$.

| $p = (p(x), p(t))$ | $||\text{Err}_\theta||_\infty$ | $||\text{Err}_\Gamma||_\infty$ | $||\text{Err}V_n||_\infty$ | iter |
|---------------------|---------------------------------|---------------------------------|-----------------------------|------|
| (3,3)               | 1.36E-005                       | 4.65E-006                       | 2.34E-007                   | 7    |
| (3,10)              | 1.29E-005                       | 4.42E-006                       | 1.84E-007                   | 8    |
| (4,4)               | 8.27E-008                       | 2.67E-008                       | 1.02E-009                   | 7    |
| (4,10)              | 8.50E-008                       | 2.75E-008                       | 1.01E-009                   | 7    |
| (5,5)               | 1.44E-008                       | 4.58E-009                       | 1.62E-009                   | 7    |
| (5,10)              | 1.42E-008                       | 4.52E-009                       | 1.60E-009                   | 7    |
| (6,6)               | 1.43E-012                       | 1.62E-014                       | 4.03E-013                   | 10   |
| (6,10)              | 1.20E-012                       | 3.86E-013                       | 4.81E-013                   | 8    |
| (7,7)               | 4.08E-012                       | 1.27E-012                       | 2.80E-013                   | 10   |
| (7,10)              | 4.20E-012                       | 1.31E-012                       | 2.87E-013                   | 9    |

Figure 12. Errors in the temperature (left) and the interface velocity (right) as a function of polynomial order $p(x)$ in the ice phase. The interface velocity is computed by the weak form method (46). The polynomial order in time direction, $p(t)$, is chosen to be the same as $p(x)$. The interface starts at $x = 0.1$ and moves to $x = 0.4$. The comparison is made among different values of $St$, i.e., $St = 0.001$, $St = 0.005$, $St = 0.01$ and $St = 0.05$. The computational domain is divided into $E(x) = 6$ in the spatial direction. The time step is chosen to be $\Delta t = 0.9 \times 10^{-3}$.

Stefan problem is a nonlinear problem due to the coupling between the prior unknown time-varying domain and the temperature distribution in this domain. To address this complexity, transformation techniques are introduced. The space-time...
curved subelements in a cut-cell are handled by mapping the fixed frame of reference to a moving frame of reference. In the moving frame of reference, we have a rectangular mesh in each space-time slab. In each space-time slab, a DG spectral element method is applied to solve the transformed heat equations in the space-time formulation in each phase independently. We present two options for computing the interface velocity: the weak form option and the normal probe option. Numerical experiments on standard one-dimensional Stefan problems demonstrate the spectral convergence in both space and time for the temperature and the interface velocity. Both options for calculating the interface velocity exhibit spectral convergence in both space and time. Moreover, the Picard iteration convergences only in a just iterations (less than 15) for both options. We have also analyzed the present method for different values of the Stefan number, $St$, and found that the space-time spectral accuracy property of the present method is not sensitive to the values of $St$.

The present method is not restricted to one spatial dimension, and its extension to multiple space dimensions is the subject of future activity. In multiple space dimensions, a high-order accurate numerical quadrature scheme [50] needs to be developed for computing the integrals over curved and implicitly defined surfaces and volumes in the spatial element, generated by the phase interface, in the weak formulations. Future work will also explore an alternative direct solver methodology [42] for the sparse linear systems resulting from each Picard iteration.

**Acknowledgments.** This work was partially supported by the National Science Foundation under contract DMS 1418983.
REFERENCES

[1] N. Al-Rawahi and G. Tryggvason, Numerical simulation of dendritic solidification with convection: Two-dimensional geometry, *J. Comput. Phys.*, **180** (2002), 471–496.

[2] N. Al-Rawahi and G. Tryggvason, Numerical simulation of dendritic solidification with convection: Three-dimensional flow, *J. Comput. Phys.*, **194** (2004), 677–696.

[3] V. Alexiades and A. D. Solomon, *Mathematical Modelling of Melting and Freezing Processes*, Hemisphere Publishing Corporation, Washington, 1981.

[4] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney and D. Sorensen, *LAPACK Users’ Guide*, 3rd edition, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1999.

[5] T. D. Aslam, A partial differential equation approach to multidimensional extrapolation, *J. Comput. Phys.*, **193** (2004), 349–355.

[6] D. C. Assencio and J. M. Teran, A second order virtual node algorithm for Stokes flow problems with interfacial forces, discontinuous material properties and irregular domains, *J. Comput. Phys.*, **250** (2013), 77–105.

[7] M. Azaiez, F. Jelassi, M. Mint Brahim and J. Shen, Two-phase Stefan problem with smoothed enthalpy, *Commun. Math. Sci.*, **14** (2016), 1625–1641.

[8] J. Bedrossian, J. H. von Brecht, S. Zhu, E. Sifakis and J. M. Teran, A second order virtual node method for elliptic problems with interfaces and irregular domains, *J. Comput. Phys.*, **229** (2010), 6405–6426.

[9] M. Benzi and M. A. Olshanskii, An augmented Lagrangian-based approach to the Oseen problem, *SIAM J. Sci. Comput.*, **28** (2006), 2095–2113.

[10] C. Canuto, M. Y. Hussaini, A. Quarteroni and T. A. Zang, *Spectral Methods: Fundamentals in Single Domains*, Berlin, Springer, 2006.

[11] S. Chen, B. Merriman, S. Osher and P. Smereka, A simple level set method for solving Stefan problems, *J. Comput. Phys.*, **135** (1997), 8–29.

[12] J. Chessa, P. Smolinski and T. Belytschko, The extended finite element method (XFEM) for solidification problems, *Int. J. Numer. Meth. Eng.*, **53** (2002), 1959–1977.

[13] B. Bernardo Cockburn, G. Karniadakis and C.-W. Shu (eds.), *Discontinuous Galerkin Methods: Theory, Computation, and Applications*, Lecture notes in computational science and engineering, Springer, Berlin, New York, 2000.

[14] B. Cockburn, G. Kanschat, I. Perugia and D. Schötzau, Superconvergence of the local discontinuous Galerkin method for elliptic problems on Cartesian grids, *SIAM J. Numer. Anal.*, **39** (2001), 264–285.

[15] B. Cockburn and C.-W. Shu, The local discontinuous Galerkin method for time-dependent convection-diffusion systems, *SIAM J. Numer. Anal.*, **35** (1998), 2440–2463.

[16] H. Coxeter, *Regular Polytopes*, Dover Publications, Inc., New York, 1973.

[17] A. Criscione, D. Kintea, Z. Tuković, S. Jakirlić, I. Roisman and C. Tropea, Crystallization of supercooled water: A level-set-based modeling of the dendrite tip velocity, *Int. J. Heat Mass Transfer*, **66** (2013), 830–837.

[18] M. Farid, The moving boundary problems from melting and freezing to drying and frying of food, *Chemical Engineering and Processing: Process Intensification*, **41** (2002), 1–10.

[19] R. P. Fedkiw, T. Aslam, B. Merriman and S. Osher, A Non-oscillatory Eulerian approach to interfaces in multimaterial flows (the Ghost Fluid Method), *J. Comput. Phys.*, **152** (1999), 457–492.

[20] W. L. George and J. A. Warren, A parallel 3D dendritic growth simulator using the phase-field method, *J. Comput. Phys.*, **177** (2002), 264–283.

[21] F. Gibou and R. Fedkiw, A fourth order accurate discretization for the Laplace and heat equations on arbitrary domains, with applications to the Stefan problem, *J. Comput. Phys.*, **202** (2005), 577–601.

[22] F. Gibou, R. Fedkiw, R. Caflisch and S. Osher, A level set approach for the numerical simulation of dendritic growth, *J. Sci. Comput.*, **19** (2003), 183–199.

[23] F. Gibou, R. P. Fedkiw, L.-T. Cheng and M. Kang, A second-order-accurate symmetric discretization of the Poisson equation on irregular domains, *J. Comput. Phys.*, **176** (2002), 205–227.

[24] S. C. Gupta, *The Classical Stefan Problem: Basic Concepts, Modelling and Analysis*, vol. 45 of North-Holland Series in Applied Mathematics and Mechanics, Elsevier Science B.V., Amsterdam, 2003.
[25] D. Han, A decoupled unconditionally stable numerical scheme for the Cahn-Hilliard-Hele-Shaw system, *J. Sci. Comput.*, 66 (2016), 1102–1121.

[26] D. Han and X. Wang, A second order in time, uniquely solvable, unconditionally stable numerical scheme for Cahn-Hilliard-Navier-Stokes equation, *J. Comput. Phys.*, 290 (2015), 139–156.

[27] D. Han and X. Wang, Decoupled energy-law preserving numerical schemes for the Cahn-Hilliard-Darcy system, *Numer. Methods Partial Differential Equations*, 32 (2016), 936–954.

[28] J. S. Hesthaven and T. Warburton, *Nodal Discontinuous Galerkin Methods: Algorithms, Analysis, and Applications*, vol. 54 of Texts in Applied Mathematics, Springer, New York, 2008.

[29] E. Javierre, C. Vuik, F. J. Vermolen and S. van der Zwaag, A comparison of numerical models for one-dimensional Stefan problems, *J. Comput. Appl. Math.*, 192 (2006), 445–459.

[30] M. Jemison, E. Loch, M. Sussman, M. Shashkov, M. Arienti, M. Ohta and Y. Wang, A coupled level-set-moment of fluid method for incompressible two-phase flows, *J. Sci. Comput.*, 54 (2013), 454–491.

[31] M. Jemison, M. Sussman and M. Arienti, Compressible, multiphase semi-implicit method with moment of fluid interface representation, *J. Comput. Phys.*, 279 (1996), 127–148.

[32] A. Karma and W.-J. Rappel, Quantitative phase-field modeling of dendritic growth in two and three dimensions, *Phys. Rev. E*, 57 (1998), 4323–4349.

[33] G. E. Karniadakis and S. J. Sherwin, *Spectral/hp Element Methods for CFD*, 2nd edition, Numerical Mathematics and Scientific Computation, Oxford University Press, Oxford, 2013.

[34] C. M. Klaij, J. J. W. van der Vegt and H. van der Ven, Space-time discontinuous Galerkin method for the compressible Navier-Stokes equations, *J. Comput. Phys.*, 217 (2006), 589–611.

[35] D. Juric and G. Tryggvason, A front-tracking method for dendritic solidification, *J. Comput. Phys.*, 123 (1996), 127–148.

[36] D. A. Kopriva, *Implementing Spectral Methods for Partial Differential Equations: Algorithms for Scientists and Engineers*, Scientific Computation, Springer, Berlin, 2009.

[37] M. Kucharik, R. V. Garimella, S. P. Schofield and M. J. Shashkov, A comparative study of interface reconstruction methods for multi-material ALE simulations, *J. Comput. Phys.*, 229 (2010), 2432–2452.

[38] B. Li and S. Da-Wen, Novel methods for rapid freezing and thawing of foods - a review, *Journal of Food Engineering*, 54 (2002), 175–182.

[39] Z. Li, Immersed interface methods for moving interface problems, Numer. Algorithms, 14 (1997), 269–293.

[40] Z. Li and M.-C. Lai, The immersed interface method for the Navier-Stokes equations with singular forces, *J. Comput. Phys.*, 171 (2001), 822–842.

[41] R. Loubère, P.-H. Maire, M. Shashkov, J. Breil and S. Galera, ReALE: a reconnection-based arbitrary-Lagrangian-Eulerian method, *J. Comput. Phys.*, 229 (2010), 4724–4761.

[42] P. G. Martinsson, A direct solver for variable coefficient elliptic PDEs discretized via a composite spectral collocation method, *J. Comput. Phys.*, 242 (2013), 460–479.

[43] M. N. J. Moore, Riemann-hilbert problems for the shapes formed by bodies dissolving, melting, and eroding in fluid flows, 2016, Accepted Communications in Pure and Applied Mathematics.

[44] S. Osher and R. Fedkiw, *Level Set Methods and Dynamic Implicit Surfaces*, vol. 153 of Applied Mathematical Sciences, Springer, New York, N.Y., 2003.

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

[46] S. Rhebergen and B. Cockburn, A space-time hybridizable discontinuous Galerkin method for incompressible flows on deforming domains, *J. Comput. Phys.*, 231 (2012), 4185–4204.

[47] S. Rhebergen, B. Cockburn and J. J. W. van der Vegt, A space-time discontinuous Galerkin method for the incompressible Navier-Stokes equations, *J. Comput. Phys.*, 233 (2013), 339–358.

[48] E. M. Rønquist and A. T. Patera, A Legendre spectral element method for the Stefan problem, *Int. J. Numer. Meth. Eng.*, 24 (1987), 2273–2299.

[49] B. Šarler, Stefan’s work on solid-liquid phase changes, *Engineering Analysis with Boundary Elements*, 16 (1995), 83–92.

[50] R. I. Saye, High-order quadrature methods for implicitly defined surfaces and volumes in hyperrectangles, *SIAM J. Sci. Comput.*, 37 (2015), A993–A1019.
[51] J. A. Sethian and J. Strain, Crystal growth and dendritic solidification, J. Comput. Phys., 98 (1992), 231–253.
[52] W. E. H. Sollie, O. Bokhove and J. J. W. van der Vegt, Space-time discontinuous Galerkin finite element method for two-fluid flows, J. Comput. Phys., 230 (2011), 789–817.
[53] J. J. Sudirham, J. J. W. van der Vegt and R. M. J. van Damme, Space-time discontinuous Galerkin method for advection-diffusion problems on time-dependent domains, Appl. Numer. Math., 56 (2006), 1491–1518.
[54] M. Sussman, K. M. Smith, M. Y. Hussaini, M. Ohta and R. Zhi-Wei, A sharp interface method for incompressible two-phase flows, J. Comput. Phys., 221 (2007), 469–505.
[55] M. Sussman and M. Y. Hussaini, A discontinuous spectral element method for the level set equation, Special issue in honor of the sixtieth birthday of Stanley Osher J. Sci. Comput., 19 (2003), 479–500.
[56] M. Sussman, P. Smereka and S. Osher, A level set approach for computing solutions to incompressible two-phase flow, Journal of Computational Physics, 114 (1994), 146–159.
[57] L. Tan and N. Zabaras, A level set simulation of dendritic solidification of multi-component alloys, J. Comput. Phys., 221 (2007), 9–40.
[58] H. Udaykumar, S. Morella and S. Krishnan, Sharp-interface simulation of dendritic growth with convection: Benchmarks, International Journal of Heat and Mass Transfer, 46 (2003), 2615–2627.
[59] M. Vahab and G. Miller, A front-tracking shock-capturing method for two gases, Communications in Applied Mathematics and Computational Science, 11 (2016), 1–35.
[60] M. Vahab, C. Pei, M. Y. Hussaini, M. Sussman and Y. Lian, An adaptive coupled level set and moment-of-fluid method for simulating droplet impact and solidification on solid surfaces with application to aircraft icing, in 54th AIAA Aerospace Sciences Meeting, 2016, p1340.
[61] J. J. W. van der Vegt and J. J. Sudirham, A space-time discontinuous Galerkin method for the time-dependent Oseen equations, Appl. Numer. Math., 58 (2008), 1892–1917.
[62] S. W. Welch and J. Wilson, A volume of fluid based method for fluid flows with phase change, J. Comput. Phys., 160 (2000), 662–682.

Received May 2017; revised June 2017.

E-mail address: cpei@math.fsu.edu
E-mail address: sussman@math.fsu.edu
E-mail address: yhussain@fsu.edu