DOI: 10.2478/$auom$ − 2018 − 0016
An. Şt. Univ. Ovidius Constanţa

sciendo
Vol. 26(2),2018, 11-40

Optimal Control of a Stefan Problem Fully Coupled with Incompressible Navier–Stokes Equations and Mesh Movement

Björn Baran, Peter Benner, Jan Heiland, Jens Saak

Abstract

The optimal control of moving boundary problems receives growing attention in science and technology. We consider the so called two-phase Stefan problem that models a solid and a liquid phase separated by a moving interface. The Stefan problem is coupled with incompressible Navier–Stokes equations. We take a sharp interface model approach and define a quadratic tracking-type cost functional that penalizes the deviation of the interface from the desired state and the control costs. With the formal Lagrange approach and an adjoint system we derive the gradient of the cost functional. The derived formulations can be used to achieve a desired interface position. Among others, we address how to handle the weak discontinuity of the temperature along the interface with mesh movement methods in a finite element framework.

1 Introduction

Free boundary and moving boundary problems can be used to model crystal growth or the solidification and melting of pure materials. The optimal control of these problems is of great interest since certain desired shapes of the boundaries improve, e.g., the material quality in the case of crystal growth [14].

Key Words: Stefan problem, optimal control, moving interface, mesh movement, Navier–Stokes equations.
2010 Mathematics Subject Classification: Primary 49M29, 49-XX, 65-XX; Secondary 65Mxx, 65M60, 65Nxx, 65N50.
Received: March, 2017.
Revised: April, 2017.
Accepted: June, 2017.

Problems with moving boundaries feature a strong coupling between geometric and physical unknowns. As a consequence, these problems are non-linear and their numerical solution as well as optimal control require the characterization of the geometric unknowns. One instance of a free boundary problem is the two-dimensional two-phase Stefan problem, fully coupled to the incompressible Navier–Stokes equations. In this case, the geometrical unknown is an interface that separates the domain into a liquid and a solid phase. Physical unknowns are the velocity and pressure in the liquid phase and the temperature over the whole domain. One difficulty, especially for the numerical solution, is the discontinuity of the temperature gradient across the interface. The temperature and the inner boundary are coupled by the Stefan condition. This condition connects the jump of the temperature gradient across the interface with the normal velocity of the interface.

There exist several approaches to represent the interface and deal with the discontinuity of the temperature gradient. One possibility is to formulate the Stefan problem in enthalpy formulation, as done for example by White [18]. The interface can be treated implicitly with a mushy region of material and no explicit representation or tracking of the inner boundary is necessary. This is one advantage of the enthalpy formulation. Thereby, the implementation is relatively simple [17, p. 219]. Nevertheless, for the optimal control, interface tracking is required. Thus, a sharp interface representation is preferable. In the literature, there are different ways to treat the moving inner boundary explicitly. It can be represented as the zero level set of a time dependent, implicit function as done by Nochetto et al. [12, 13] and Zabaras et al. [19]. The numerical solution of this level set function is done by the finite element method (FEM). The temperature is approximated with the extended FEM (X-FEM), where the FEM functions are modified in a narrow band around the interface to deal with the discontinuity of the temperature gradient. Bernauer uses this technique in his PhD thesis [7], combined with an adjoint-based optimal control approach. An alternative approach is to use an adaptive mesh for the spatial discretization. The jump in the temperature gradient can be represented, if the edges of the mesh are aligned with the interface. To ensure this in every time step, the corresponding edges can be moved together with the moving interface. Ziegenbalg [10, 20] uses this technique combined with finite differences, a graph representation of the interface and an adjoint-based optimal control approach. Bänsch et al. [4, 3] use FEM and a variational form of the Stefan condition to solve for the interface velocity. Both works couple the Stefan problem with Navier–Stokes equations. A different name for these mesh movement methods is arbitrary Lagrangian-Eulerian (ALE) methods. A detailed overview of the existing literature can be found in [5].

In this work, we adapt the explicit representation of the interface as a graph from [20], use the heat equation for the temperature and Navier–Stokes equations for the velocity of the fluid and combine them with the numerical solution techniques from [4], which include FEM and mesh movement. Additionally, we add in- and outflow conditions on specific parts of the boundary. We use the adjoint-based optimal control approach from [7, 20] and define a quadratic tracking-type cost functional to steer the interface to a desired position. In contrast to the existing literature, we choose the pressure potential at the inlet as the control variable. Further, the mesh movement is fully integrated into the partial differential equation (PDE) systems. This results in a control which has a less direct influence on the interface than, for example, controlling the temperature directly. With the formal Lagrange approach, we derive an adjoint system of PDEs, which we use to compute the gradient of the cost functional. This first order optimality system follows the "optimize-then-discretize" paradigm. We plug the gradient into a gradient algorithm and compute a step size with a quadratic line minimization algorithm similar to the one in [20]. To solve the forward and adjoint PDE systems, we use FEniCS [1].

The paper is organized as follows. In Section 2, we describe the state equations which define the Stefan problem. This includes the heat equation for the temperature together with the Stefan condition for the interface velocity. Further, we describe the mesh movement equations in this section. The velocity and pressure in the liquid phase are characterized by the Navier–Stokes equations. The control variable appears in the boundary conditions of the Navier–Stokes equations. In Section 3, we define a cost functional and the optimization problem. We use a Lagrange functional to derive the adjoint system and the gradient of the cost functional. At the end of this section, we formulate the gradient and line minimization algorithms that we use to approximate an optimal control. Section 4 contains a brief summary of the time discretization with an implicit Euler scheme and the spatial discretization with FEM techniques. The behavior of the mesh movement is illustrated in this section. In Section 5, we illustrate the performance of the presented approach with two numerical experiments.

2 Two-Phase Stefan Problem

In this section, we present the arrangement of the domain and its boundary regions. Moreover, we define the graph representation of the interface and the equations characterizing the temperature. Further, we describe the interface movement in detail together with the corresponding boundary conditions. Analogously, we do the same for the mesh movement and for the velocity and pressure in the liquid phase. The definitions in this section closely follow [5].

We define the domain as $\Omega(t) \subset \mathbb{R}^2$. It is split into the solid phase $\Omega_s(t)$ and the liquid phase $\Omega_l(t)$. By $\Gamma_{\text{int}}(t)$ we denote the interface which separates the two phases as in Figure 1. The inflow and the heating with the temperature $T_{\text{heat}}(t)$ are located at $\Gamma_{\text{in}}(t)$, the outflow at $\Gamma_{\text{out}}(t)$. At the bottom, there is the cooling boundary $\Gamma_{\text{cool}}(t) = [a, b] \subset \mathbb{R}$ with the cooling temperature $T_{\text{cool}}(t)$. The remaining parts of the outer boundary are denoted $\Gamma_N(t)$. We denote the boundary part at the top $\Gamma_{\tilde{N}}(t) \subset \Gamma_N(t)$. The inner boundary $\Gamma_{\text{int}}(t)$ moves so that its position is time-dependent. Thus, the solid and liquid phases are time-dependent as are their boundaries. For the sake of brevity, the explicit mentioning of the time-dependence "(t)" is avoided in most places throughout this work.

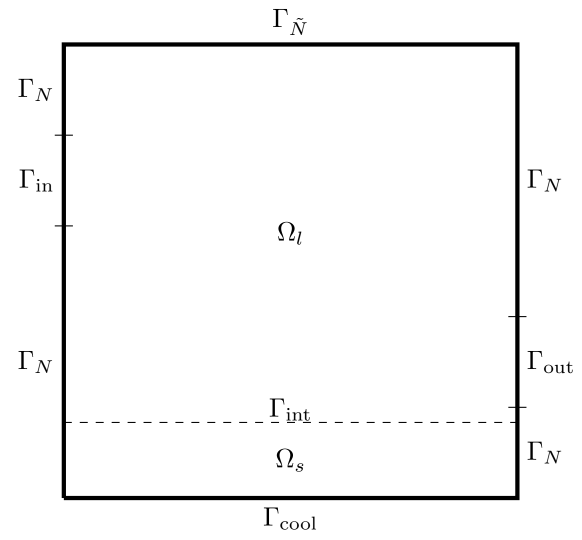


Figure 1: The domain $\Omega \subset \mathbb{R}^2$ for the Stefan problem.

As in [20], we assume that the interface can be represented as a graph

$$\Gamma_{\text{int}}(t) = \left\{ \begin{bmatrix} x_1 \\ f(t, x_1) \end{bmatrix} : x_1 \in \Gamma_{\text{cool}} \right\}, \qquad \text{with } f\colon [0, t_f] \times \Gamma_{\text{cool}} \to \mathbb{R},$$

where $[0, t_f], t_f > 0$ is the time interval. We abbreviate the derivatives of f with

$$f_{x_1} := \frac{df}{dx_1}, \qquad f_t := \partial_t f.$$

To map from Γ_{cool} to the interface Γ_{int}, the function $\Phi\colon [0,t_f]\times\Gamma_{\text{cool}} \to [0,t_f]\times\Gamma_{\text{int}}$ is used. It is defined as

$$\Phi(t,x_1) := \left(t, \begin{bmatrix} x_1 \\ f(t,x_1) \end{bmatrix}\right).$$

The unit normal vector $\boldsymbol{n}_{\text{int}}$ along the interface Γ_{int} is pointing from the solid to the liquid phase. It can be expressed as (see [20, Sec. 2.1])

$$\boldsymbol{n}_{\text{int}}(t,x_1) = \frac{1}{\sqrt{1+f_{x_1}(t,x_1)^2}} \begin{bmatrix} -f_{x_1}(t,x_1) \\ 1 \end{bmatrix}. \tag{1}$$

2.1 Heat Equation, Mesh Movement, Navier–Stokes Equations

We denote the temperature in the solid and liquid phases by T. It is modeled by the heat equation:

$$\partial_t T + (v - V_{\text{mesh}})\cdot\nabla T - \alpha\Delta T = 0, \qquad \text{on } (0,t_f]\times\Omega, \tag{2}$$

where v is the velocity of the fluid and V_{mesh} the mesh movement (for details see equation (5)). We define the boundary conditions for equation (2) and any further equations in (7). Whenever v is used over the whole domain Ω, it is extended with 0 on Ω_s. Additionally, the Stefan condition at the moving interface Γ_{int} couples the temperature with the velocity of the interface in normal direction:

$$[k_s(\nabla T)_s - k_l(\nabla T)_l] =: [k(\nabla T)]_l^s = L\cdot V_{\text{int}}, \qquad \text{on } \Gamma_{\text{int}}, \tag{3}$$

with k_s and k_l denoting the heat conductivities in the solid and liquid phases and L denoting the latent heat, and with $(\nabla T)_s := \partial_{\boldsymbol{n}_{\text{int}}} T\big|_{\Omega_s}$, $(\nabla T)_l := \partial_{-\boldsymbol{n}_{\text{int}}} T\big|_{\Omega_l}$. This equation can be used to determine V_{int} if T is known.

With (1), we can express the velocity V_{int} of the interface Γ_{int} in normal direction $\boldsymbol{n}_{\text{int}}$ as

$$\begin{aligned} V_{\text{int}}(t,x_1) &= \partial_t \begin{bmatrix} x_1 \\ f(t,x_1) \end{bmatrix} \cdot \boldsymbol{n}_{\text{int}}(t,x_1) = \begin{bmatrix} 0 \\ f_t(t,x_1) \end{bmatrix} \cdot \boldsymbol{n}_{\text{int}}(t,x_1) \\ &= \frac{f_t(t,x_1)}{\sqrt{1+f_{x_1}(t,x_1)^2}} = f_t(t,x_1)\boldsymbol{n}_{\text{int}}(t,x_1)\cdot\boldsymbol{e_2}. \end{aligned} \tag{4}$$

We denote the unit vector in vertical direction as $\boldsymbol{e_2} = [0,1]^\mathsf{T}$. Using equation (4), the Stefan condition (3) can be reformulated to

$$\sqrt{1+f_{x_1}^2}\cdot[k(\nabla T)]_l^s \circ \Phi = L\cdot f_t, \qquad \text{on } \Gamma_{\text{cool}},$$

$$\sqrt{1+f_{x_1}^2} \cdot V_{\text{int}} \circ \Phi = f_t, \qquad \text{on } \Gamma_{\text{cool}}.$$

We will need this reformulation to couple the whole system with the cost functional. The mesh movement V_{mesh} and the velocity of the liquid v will be discussed in the remaining part of this section.

In the initial partition, the edges of the mesh are aligned with the interface Γ_{int}. To keep this for the next time step, we move the vertices on the interface with V_{int} in normal direction. In order to prevent the mesh from degrading, i.e. avoid extreme cell deformations, or mesh tangling, V_{int} is smoothly extended to V_{mesh} on the whole domain Ω. For this, the following Laplace equation is solved:

$$\Delta V_{\text{mesh}} = 0, \qquad \text{on } (0, t_f] \times \Omega, \tag{5a}$$

$$V_{\text{mesh}} = V_{\text{int}} \cdot \boldsymbol{n}_{\text{int}}, \qquad \text{on } (0, t_f] \times \Gamma_{\text{int}}. \tag{5b}$$

The second equation (5b) is a Dirichlet condition on the inner boundary Γ_{int}, which ensures $V_{\text{mesh}} = V_{\text{int}} \cdot \boldsymbol{n}_{\text{int}}$ on Γ_{int}.

The interface Γ_{int} is a non-material surface. The movement V_{int} of the interface and the mesh movement V_{mesh} are not related to the movement of any physical material points. As pointed out in [4], the non-material movement V_{mesh} needs to be separated from the material movement in T and v with advection terms

$$-V_{\text{mesh}} \cdot \nabla T$$

for the heat equation (2) and

$$-(V_{\text{mesh}} \cdot \nabla) v$$

for the Navier–Stokes equations, which are denoted in the following. The velocity v and the pressure p in the liquid phase are described with the incompressible Navier–Stokes equations for Newtonian fluids [8]:

$$\partial_t v + ((v - V_{\text{mesh}}) \cdot \nabla) v - \eta \Delta v + \nabla p = 0, \qquad \text{on } (0, t_f] \times \Omega_l, \tag{6a}$$

$$\nabla \cdot v = 0, \qquad \text{on } (0, t_f] \times \Omega_l, \tag{6b}$$

$$p \cdot \boldsymbol{n} - \eta \partial_{\boldsymbol{n}} v = u \cdot \boldsymbol{n}, \qquad \text{on } (0, t_f] \times \Gamma_{\text{in}}. \tag{6c}$$

The constant η is the kinematic viscosity. In addition to the momentum and mass balance equations (6a)–(6b), equation (6c) defines an inflow boundary condition on Γ_{in}. It is influenced by the control variable u which is constant in space on Γ_{in}.

We use the control u to influence the pressure and velocity gradient in normal direction at the inflow boundary. It is used to steer the system to a desired state.

The whole system reads as follows:

$$\begin{aligned}
\partial_t T + (v - V_{\text{mesh}}) \cdot \nabla T - \alpha \Delta T &= 0, && \text{on } (0, t_f] \times \Omega, \\
\sqrt{1 + f_{x_1}^2} \cdot [k(\nabla T)]_l^s \circ \Phi &= L \cdot f_t, && \text{on } (0, t_f] \times \Gamma_{\text{cool}}, \\
\sqrt{1 + f_{x_1}^2} \cdot V_{\text{int}} \circ \Phi &= f_t, && \text{on } (0, t_f] \times \Gamma_{\text{cool}}, \\
T &= T_{\text{heat}}, && \text{on } (0, t_f] \times \Gamma_{\text{in}}, \\
T &= T_{\text{cool}}, && \text{on } (0, t_f] \times \Gamma_{\text{cool}}, \\
T &= T_{\text{melt}}, && \text{on } (0, t_f] \times \Gamma_{\text{int}}, \\
\partial_{\boldsymbol{n}} T &= 0, && \text{on } (0, t_f] \times (\Gamma_N \cup \Gamma_{\text{out}}), \\
T(0) &= T_0, && \text{on } \Omega, \\
V_{\text{int}}(0) &= 0, && \text{on } \Gamma_{\text{int}}, \\
f(0) &= f_0, && \text{on } \Gamma_{\text{cool}}, \\
\Delta V_{\text{mesh}} &= 0, && \text{on } (0, t_f] \times \Omega, \\
V_{\text{mesh}} &= V_{\text{int}} \cdot \boldsymbol{n}_{\text{int}}, && \text{on } (0, t_f] \times \Gamma_{\text{int}}, \\
V_{\text{mesh}} &= 0, && \text{on } (0, t_f] \times (\Gamma_{\text{cool}} \cup \Gamma_{\tilde{N}}), \\
V_{\text{mesh}} \cdot \boldsymbol{n} &= 0, && \text{on } (0, t_f] \times \partial\Omega, \\
V_{\text{mesh}}(0) &= 0, && \text{on } \Omega, \\
\partial_t v + ((v - V_{\text{mesh}}) \cdot \nabla) v - \eta \Delta v + \nabla p &= 0, && \text{on } (0, t_f] \times \Omega_l, \\
\nabla \cdot v &= 0, && \text{on } (0, t_f] \times \Omega_l, \\
v &= 0, && \text{on } (0, t_f] \times (\Gamma_{\text{int}} \cup (\Gamma_N \cap \partial\Omega_l)), \\
p \cdot \boldsymbol{n} - \eta \partial_{\boldsymbol{n}} v &= u \cdot \boldsymbol{n}, && \text{on } (0, t_f] \times \Gamma_{\text{in}}, \\
p \cdot \boldsymbol{n} - \eta \partial_{\boldsymbol{n}} v &= 0, && \text{on } (0, t_f] \times (\Gamma_{\text{out}} \cap \partial\Omega_l), \\
v(0) &= 0, && \text{on } \Omega_l, \\
p(0) &= 0, && \text{on } \Omega_l.
\end{aligned} \tag{7}$$

In this PDE system, the control u is given and the functions T (temperature), f (interface graph), V_{int} (interface velocity), V_{mesh} (mesh movement), v (velocity) and p (pressure) are unknowns. Throughout this work, the system (7) is called the forward system. A detailed description of the equations can be found in [5]. The next section describes the optimal control approach to steer the interface to a desired position.

3 Optimization

In this section, we introduce the control problem and the derivation of the adjoint system via the Lagrange formalism. The concrete control problem is defined in terms of the underlying PDE system (7) together with a cost functional. To derive the adjoint system, a Lagrange functional is required. We formulate a projected gradient algorithm combined with a quadratic line minimization algorithm to compute a control, which steers the interface to a desired position.

What follows is closely orientated towards [7, 20] and can be found with additional details in [5]. More details on the Lagrange formalism for the optimal control of PDEs can be found in [16].

For the state y from the state space $\mathcal{Y}$ and the control u, which is an element of the control space $\mathcal{U}$, the optimal control problem is defined as

$$\begin{aligned} &\min_{y\in\mathcal{Y},u\in\mathcal{U}} J(y,u) \\ \text{subject to}& \\ &e(y,u)=0, \\ &u\in\mathcal{U}_{\text{ad}}\subset\mathcal{U}. \end{aligned} \tag{8}$$

In the present Stefan problem, the state is defined as the tuple $y=[f,T,V_{\text{int}},V_{\text{mesh}},v,p]$. The control functions $u(x,t)=\tilde{u}(t)\cdot I_{in}(x)$ are chosen constant in space on Γ_{in} and will be identified with the scalar function $\tilde{u}\colon[0,t_f]\to\mathbb{R}$ in the remainder of this work. Further, the control constraint $u\in\mathcal{U}_{\text{ad}}$ defines restrictions on the control. The set of admissible controls $\mathcal{U}_{\text{ad}}$ is usually a convex subset of $\mathcal{U}$. In the case that $\mathcal{U}_{\text{ad}}=\mathcal{U}$, the problem is unrestricted. The state equation $e(y,u)=0$ connects the state and the control. It represents the PDE-constraints of the Stefan problem, which are defined in the forward system (7).

3.1 Definition of the Cost Functional and the Lagrange Functional

We define the cost functional to steer the position of the interface to a desired one. The graph f_d describes the desired position of the interface Γ_{int}. The scalars Λ, $\bar{\Lambda}$, and λ are weight factors for the cost functional J:

$$\begin{aligned} J(y,u):=&\frac{\Lambda}{2}\int\limits_{\Gamma_{\text{cool}}}\left(f(t_f,x_1)-f_d(t_f,x_1)\right)^2\,dx_1+\frac{\bar{\Lambda}}{2}\int\limits_0^{t_f}\int\limits_{\Gamma_{\text{cool}}}\left(f(t,x_1)-f_d(t,x_1)\right)^2\,dx_1dt \\ &+\frac{\lambda}{2}\int\limits_0^{t_f}\int\limits_{\Gamma_{\text{in}}}\left(u(t)\right)^2\,dx_2dt. \end{aligned} \tag{9}$$

The first term aims to steer the interface position to the desired position at terminal time t_f, while the second term penalizes the interface deviation over the complete time horizon $(0,t_f]$. The third term models control costs and has a regularizing effect [16, p. 3]. Since we have no proof of the existence and uniqueness of a solution to the forward system (7), the optimal control techniques, which we apply here, are only formal. It is assumed that for every $u \in \mathcal{U}_{\text{ad}}$, unique states $T(u)$, $f(u)$, $V_{\text{int}}(u)$, $V_{\text{mesh}}(u)$, $v(u)$, and $p(u)$ exist that solve the forward system (7) and thus, the state equation $e(y,u) = 0$.

As a consequence, we can define the *reduced* cost functional $\mathcal{K}(u) := J(y(u),u)$ together with an optimal control problem (equivalent to (8)):

$$\min_{u \in \mathcal{U}_{\text{ad}}} \mathcal{K}(u). \tag{10}$$

To find a solution $u^* \in \mathcal{U}_{\text{ad}}$ for (10), we use first-order necessary optimality conditions. These can be derived formally by applying the Lagrange formalism. For this, we define the Lagrange multiplier as the tuple of adjoint states $\zeta = [\omega, \omega_{\text{int}}, \psi, \psi_{\text{cool}}, \psi_{\text{mesh}}, \psi_{\text{int}}, \gamma, \pi, \varphi, \gamma_{\text{out}}]$.

For the sake of brevity dx, ds, dt are omitted in what follows. We define the Lagrange functional as

$$\begin{aligned}
\mathcal{L}(y,u,\zeta) :=& \, J(y,u) - e(y,u) \cdot \zeta \\
=& \frac{\Lambda}{2} \int_{\Gamma_{\text{cool}}} (f(t_f,x_1) - f_d(t_f,x_1))^2 + \frac{\bar{\Lambda}}{2} \int_0^{t_f} \int_{\Gamma_{\text{cool}}} (f(t,x_1) - f_d(t,x_1))^2 \\
&+ \frac{\lambda}{2} \int_0^{t_f} \int_{\Gamma_{\text{in}}} (u(t))^2 - \int_0^{t_f} \int_{\Omega} (\partial_t T + (v - V_{\text{mesh}}) \cdot \nabla T - \alpha \Delta T) \cdot \omega \\
&- \int_0^{t_f} \int_{\Gamma_{\text{cool}}} (\sqrt{1 + f_{x_1}^2} \cdot [k(\nabla T)]_l^s \circ \Phi - L \cdot f_t) \cdot \psi \\
&- \int_0^{t_f} \int_{\Gamma_{\text{cool}}} (\sqrt{1 + f_{x_1}^2} \cdot V_{\text{int}} \circ \Phi - f_t) \cdot \psi_{\text{cool}} - \int_0^{t_f} \int_{\Gamma_{\text{int}}} (T - T_{\text{melt}}) \cdot \omega_{\text{int}} \\
&- \int_0^{t_f} \int_{\Omega} (\Delta V_{\text{mesh}}) \cdot \psi_{\text{mesh}} - \int_0^{t_f} \int_{\Gamma_{\text{int}}} (V_{\text{mesh}} - V_{\text{int}} \cdot \boldsymbol{n}_{\text{int}}) \cdot \psi_{\text{int}} \\
&- \int_0^{t_f} \int_{\Omega_l} (\partial_t v + ((v - V_{\text{mesh}}) \cdot \nabla) v - \eta \Delta v + \nabla p) \cdot \gamma - \int_0^{t_f} \int_{\Omega_l} (\nabla \cdot v) \cdot \pi \\
&- \int_0^{t_f} \int_{\Gamma_{\text{in}}} (p \cdot \boldsymbol{n} - \eta \partial_{\boldsymbol{n}} v - u \cdot \boldsymbol{n}) \cdot \varphi - \int_0^{t_f} \int_{\Gamma_{\text{out}} \cap \partial \Omega_l} (p \cdot \boldsymbol{n} - \eta \partial_{\boldsymbol{n}} v) \cdot \gamma_{\text{out}}.
\end{aligned} \tag{11}$$

The Lagrange multiplier ζ is also called adjoint state. As laid out in [5] formally, the derivatives of $\mathcal{L}$ with respect to the states $y = [f, T, V_{\text{int}}, V_{\text{mesh}}, v, p]$ can be used to derive the adjoint system.

3.2 Derivation of the Adjoint System

The adjoint equation

$$e^*(y, u, \zeta) = 0 \tag{12}$$

is defined through the requirement that the first variation of the Lagrange functional vanishes in all admissible directions $\delta y \in \mathcal{Y}$, i.e.

$$e^*(y, u, \zeta) = 0 \Leftrightarrow \mathcal{L}_y(y, u, \zeta)\delta y = 0.$$

All terms from (7), which do not appear in $e(y, u)$ and thereby in the Lagrange functional, are treated explicitly as conditions to the directions of variation δy. For the Stefan problem, equation (12) has the form

$$D_{[f,T,V_{\text{int}},V_{\text{mesh}},v,p]}\mathcal{L}[\delta f, \delta T, \delta V_{\text{int}}, \delta V_{\text{mesh}}, \delta v, \delta p] = 0.$$

In the variation of the Lagrange functional with respect to the temperature, the jump across the interface must be treated and additional jump terms occur. Since this requires additional attention, we show this in detail. The variation of the Lagrange functional with respect to the other states is rather standard and can be found in [5].

The Variation with Respect to the Temperature T

The explicit conditions to the direction of variation δT are

$$\begin{aligned} \delta T &= 0, && \text{on } (0, t_f] \times (\Gamma_{\text{cool}} \cup \Gamma_{\text{in}}), \\ \partial_{\boldsymbol{n}} \delta T &= 0, && \text{on } (0, t_f] \times (\Gamma_N \cup \Gamma_{\text{out}}), \\ \delta T(0) &= 0, && \text{on } \Omega. \end{aligned} \tag{13}$$

The variation of the Lagrange functional with respect to T reads

$$\begin{aligned} 0 &= D_T\mathcal{L}\delta T \\ &= D_T\left(- \int_0^{t_f} \int_\Omega (\partial_t T + (v - V_{\text{mesh}}) \cdot \nabla T - \alpha \Delta T) \cdot \omega \right) \cdot \delta T \end{aligned}$$

$$+ D_T\Bigg(- \int_0^{t_f} \int_{\Gamma_{\mathrm{cool}}} \big(\sqrt{1+f_{x_1}^2} \cdot [k(\nabla T)]_l^s \circ \Phi - L \cdot f_t\big) \cdot \psi \Bigg) \cdot \delta T$$

$$+ D_T\Bigg(- \int_0^{t_f} \int_{\Gamma_{\mathrm{int}}} (T - T_{\mathrm{melt}}) \cdot \omega_{\mathrm{int}} \Bigg) \cdot \delta T.$$

We apply integration by parts as well as (13) to the variation of the first integral in the equation above with respect to the temperature. This leads to

$$\begin{aligned}
&D_T\Bigg(- \int_0^{t_f} \int_{\Omega} (\partial_t T + (v - V_{\mathrm{mesh}}) \cdot \nabla T - \alpha \Delta T) \cdot \omega \Bigg) \cdot \delta T \\
&= - \int_0^{t_f} \int_{\Omega} \partial_t \delta T \cdot \omega - \int_0^{t_f} \int_{\Omega} (v - V_{\mathrm{mesh}}) \cdot \nabla \delta T \cdot \omega + \int_0^{t_f} \int_{\Omega} \alpha \Delta \delta T \cdot \omega \\
&= - \int_0^{t_f} \int_{\Omega_s} \partial_t \delta T \cdot \omega - \int_0^{t_f} \int_{\Omega_l} \partial_t \delta T \cdot \omega - \int_0^{t_f} \int_{\Omega_s} (v - V_{\mathrm{mesh}}) \cdot \nabla \delta T \cdot \omega \\
&\quad - \int_0^{t_f} \int_{\Omega_l} (v - V_{\mathrm{mesh}}) \cdot \nabla \delta T \cdot \omega + \int_0^{t_f} \int_{\Omega_s} k_s \Delta \delta T \cdot \omega + \int_0^{t_f} \int_{\Omega_l} k_l \Delta \delta T \cdot \omega \\
&= - \int_{\Omega_s} \omega(t_f) \delta T(t_f) + \int_{\Omega_s} \omega(0) \underbrace{\delta T(0)}_{\overset{(13)}{=}\, 0} + \int_0^{t_f} \int_{\Omega_s} \partial_t \omega \cdot \delta T \\
&\quad - \int_{\Omega_l} \omega(t_f) \delta T(t_f) + \int_{\Omega_l} \omega(0) \underbrace{\delta T(0)}_{\overset{(13)}{=}\, 0} + \int_0^{t_f} \int_{\Omega_l} \partial_t \omega \cdot \delta T \\
&\quad - \int_0^{t_f} \int_{\partial \Omega_s} (v - V_{\mathrm{mesh}}) \cdot (\omega \cdot \boldsymbol{n}) \cdot \delta T + \int_0^{t_f} \int_{\Omega_s} (v - V_{\mathrm{mesh}}) \cdot \nabla \omega \cdot \delta T \\
&\quad - \int_0^{t_f} \int_{\partial \Omega_l} (v - V_{\mathrm{mesh}}) \cdot (\omega \cdot \boldsymbol{n}) \cdot \delta T + \int_0^{t_f} \int_{\Omega_l} (v - V_{\mathrm{mesh}}) \cdot \nabla \omega \cdot \delta T
\end{aligned}$$

$$
\begin{aligned}
&+\int_0^{t_f}\int_{\partial\Omega_s} k_s\omega\partial_{\boldsymbol{n}}\delta T-\int_0^{t_f}\int_{\Omega_s} k_s\nabla\omega\cdot\nabla\delta T\\
&+\int_0^{t_f}\int_{\partial\Omega_l} k_l\omega\partial_{\boldsymbol{n}}\delta T-\int_0^{t_f}\int_{\Omega_l} k_l\nabla\omega\cdot\nabla\delta T\\
=&-\int_\Omega \omega(t_f)\delta T(t_f)+\int_0^{t_f}\int_\Omega \partial_t\omega\cdot\delta T\\
&+\int_0^{t_f}\int_\Omega (v-V_{\text{mesh}})\cdot\nabla\omega\cdot\delta T-\int_0^{t_f}\int_{\Gamma_{\text{out}}\cap\partial\Omega_l} v\cdot(\omega\cdot\boldsymbol{n})\cdot\delta T\\
&+\int_0^{t_f}\int_{\Gamma_{\text{int}}} \omega k_s(\partial_{\boldsymbol{n}_{\text{int}}}\delta T)_s-\int_0^{t_f}\int_{\Gamma_{\text{int}}} \omega k_l(\partial_{\boldsymbol{n}_{\text{int}}}\delta T)_l\\
&-\int_0^{t_f}\int_{\Gamma_{\text{int}}} k_s(\partial_{\boldsymbol{n}}\omega)_s\delta T-\int_0^{t_f}\int_{\Gamma_{\text{int}}} k_l(\partial_{\boldsymbol{n}}\omega)_l\delta T\\
&+\int_0^{t_f}\int_\Omega \alpha\Delta\omega\cdot\delta T-\int_0^{t_f}\int_{\Gamma_N\cup\Gamma_{\text{out}}} \alpha\partial_{\boldsymbol{n}}\omega\delta T+\int_0^{t_f}\int_{\Gamma_{\text{in}}\cup\Gamma_{\text{cool}}} \alpha\omega\partial_{\boldsymbol{n}}\delta T\\
=&-\int_\Omega \omega(t_f)\delta T(t_f)+\int_0^{t_f}\int_\Omega (\partial_t\omega+(v-V_{\text{mesh}})\cdot\nabla\omega+\alpha\Delta\omega)\cdot\delta T\\
&-\int_0^{t_f}\int_{\Gamma_{\text{out}}\cap\partial\Omega_l} v\cdot(\omega\cdot\boldsymbol{n})\cdot\delta T+\int_0^{t_f}\int_{\Gamma_{\text{int}}} \omega[k_s(\partial_{\boldsymbol{n}_{\text{int}}}\delta T)_s-k_l(\partial_{\boldsymbol{n}_{\text{int}}}\delta T)_l]\\
&-\int_0^{t_f}\int_{\Gamma_{\text{int}}} [k_s(\partial_{\boldsymbol{n}_{\text{int}}}\omega)_s-k_l(\partial_{\boldsymbol{n}_{\text{int}}}\omega)_l]\delta T-\int_0^{t_f}\int_{\Gamma_N\cup\Gamma_{\text{out}}} \alpha\partial_{\boldsymbol{n}}\omega\delta T\\
&+\int_0^{t_f}\int_{\Gamma_{\text{in}}\cup\Gamma_{\text{cool}}} \alpha\omega\partial_{\boldsymbol{n}}\delta T.
\end{aligned}
$$

Further, inserting this into the variation of the Lagrange functional with respect to T, gives

$$
\begin{aligned}
0 &= D_T\mathcal{L}\delta T \\
&= \int_0^{t_f}\int_\Omega (\partial_t\omega + (v - V_{\text{mesh}})\cdot\nabla\omega + \alpha\Delta\omega)\cdot\delta T - \int_\Omega \omega(t_f)\delta T(t_f) \\
&\quad - \int_0^{t_f}\int_{\Gamma_{\text{out}}\cap\partial\Omega_l} (\alpha\partial_{\boldsymbol{n}}\omega + v\cdot(\omega\cdot\boldsymbol{n}))\cdot\delta T - \int_0^{t_f}\int_{\Gamma_N\cup(\Gamma_{\text{out}}\cap\partial\Omega_s)} \alpha\partial_{\boldsymbol{n}}\omega\delta T \\
&\quad + \int_0^{t_f}\int_{\Gamma_{\text{cool}}} \left(\omega\circ\Phi - \sqrt{1+f_{x_1}^2}\cdot\psi\right)\cdot[k(\nabla\delta T)]_l^s\circ\Phi \\
&\quad - \int_0^{t_f}\int_{\Gamma_{\text{int}}} (\omega_{\text{int}} + [k(\nabla\omega)]_l^s)\cdot\delta T + \int_0^{t_f}\int_{\Gamma_{\text{in}}\cup\Gamma_{\text{cool}}} \alpha\omega\partial_{\boldsymbol{n}}\delta T.
\end{aligned}
$$

By proper variation of δT, certain terms can be eliminated from the equation above. Thereby, terms which are integrated over the same domain and have the same multiplier on the right can be consolidated into one equation so that the following adjoint equations arise

$$
\begin{aligned}
\partial_t\omega + (v - V_{\text{mesh}})\cdot\nabla\omega + \alpha\Delta\omega &= 0, && \text{on } [0,t_f)\times\Omega && \text{(14a)}\\
\alpha\partial_{\boldsymbol{n}}\omega + v\cdot(\omega\cdot\boldsymbol{n}) &= 0, && \text{on } [0,t_f)\times(\Gamma_{\text{out}}\cap\partial\Omega_l) && \text{(14b)}\\
\partial_{\boldsymbol{n}}\omega &= 0, && \text{on } [0,t_f)\times(\Gamma_N\cup(\Gamma_{\text{out}}\cap\partial\Omega_s)) && \text{(14c)}\\
\omega &= 0, && \text{on } [0,t_f)\times(\Gamma_{\text{cool}}\cup\Gamma_{\text{in}}) && \text{(14d)}\\
\omega\circ\Phi - \sqrt{1+f_{x_1}^2}\cdot\psi &= 0, && \text{on } [0,t_f)\times\Gamma_{\text{cool}} && \text{(14e)}\\
\omega_{\text{int}} + [k(\nabla\omega)]_l^s &= 0, && \text{on } [0,t_f)\times\Gamma_{\text{int}} && \text{(14f)}\\
\omega(t_f) &= 0, && \text{on } \Omega. && \text{(14g)}
\end{aligned}
$$

The latter equations are the adjoint system for the adjoint state ω which can be interpreted as the adjoint temperature variable. The first equation (14a) is similar to the heat equation, while the equation (14f) is analogue to the Stefan condition. The other equations can be understood as boundary conditions and the initial condition at time $t = t_f$. The sole source term in these equations is $\sqrt{1+f_{x_1}^2}\cdot\psi$ in equation (14e), which realizes the coupling to the adjoint state ψ and through this to the distance terms in the cost functional (9).

The variation of the Lagrange functional with respect to $V_{\text{int}}, V_{\text{mesh}}, v, p$ and f are performed analogously.

The Adjoint System

Similar to the state equation in (8), the adjoint equation (12) for the present optimal control problem is a PDE system, which is called the adjoint system.

This formal approach leads to

$$
\begin{aligned}
\partial_t\omega + (v - V_{\text{mesh}})\cdot\nabla\omega + \alpha\Delta\omega &= 0, && \text{on } [0,t_f)\times\Omega,\\
\alpha\partial_{\boldsymbol{n}}\omega + v\cdot(\omega\cdot\boldsymbol{n}) &= 0, && \text{on } [0,t_f)\times(\Gamma_{\text{out}}\cap\partial\Omega_l),\\
\partial_{\boldsymbol{n}}\omega &= 0, && \text{on } [0,t_f)\times(\Gamma_N\cup(\Gamma_{\text{out}}\cap\partial\Omega_s)),\\
\omega\circ\Phi &= \sqrt{1+f_{x_1}^2}\cdot\psi, && \text{on } [0,t_f)\times\Gamma_{\text{cool}},\\
\omega &= 0, && \text{on } [0,t_f)\times(\Gamma_{\text{cool}}\cup\Gamma_{\text{in}}),\\
\omega(t_f) &= 0, && \text{on } \Omega,\\
\partial_t\gamma + ((v-V_{\text{mesh}})\cdot\nabla)\gamma \qquad &&&\\
-(\nabla v)^{\mathsf{T}}\cdot\gamma + \eta\Delta\gamma + \nabla\pi &= \omega\nabla T, && \text{on } [0,t_f)\times\Omega_l,\\
\nabla\cdot\gamma &= 0, && \text{on } [0,t_f)\times\Omega_l, \qquad (15)\\
(\gamma\cdot\boldsymbol{n})\cdot(v-V_{\text{mesh}}) \qquad &&&\\
+\eta\partial_{\boldsymbol{n}}\gamma + \pi\cdot\boldsymbol{n} &= 0, && \text{on } [0,t_f)\times(\Gamma_{\text{in}}\cup(\Gamma_{\text{out}}\cap\partial\Omega_l)),\\
\gamma &= 0, && \text{on } [0,t_f)\times(\Gamma_{\text{int}}\cup(\Gamma_N\cap\partial\Omega_l)),\\
\varphi &= -\gamma, && \text{on } [0,t_f)\times\Gamma_{\text{in}},\\
\gamma(t_f) &= 0, && \text{on } \Omega_l,\\
L\cdot\partial_t\psi \qquad &&&\\
+(1+f_{x_1}^2)\cdot[k(\partial_{x_2}^2 T)]_l^s\circ\Phi\cdot\psi \qquad &&&\\
-\partial_{x_1}(2f_{x_1}\cdot[k(\partial_{x_2}T)]_l^s\circ\Phi\cdot\psi) &= \bar\Lambda(f-f_d), && \text{on } [0,t_f)\times\Gamma_{\text{cool}},\\
\psi &= 0, && \text{on } [0,t_f)\times\partial\Gamma_{\text{cool}},\\
\psi(t_f) + \frac{\Lambda}{L}(f(t_f)-f_d(t_f)) &= 0, && \text{on } \Gamma_{\text{cool}}.
\end{aligned}
$$

In this PDE system, the states T, V_{mesh}, v and f are given and the functions ω, γ, π, ψ and φ are unknowns. Initial values for the adjoint states ω, γ and ψ are given for the end time t_f. Thus, in contrast to the forward system (7), the equations in (15) have to be solved backwards in time.

3.3 Projected Gradient Method and Line Minimization Algorithm

The optimal control problem can be solved with a gradient method [16]. For this, in addition to evaluations of the forward and backward systems, also the gradient of the cost functional $\nabla\mathcal{K}$ with respect to the control u is required.

As shown in [5], $\nabla\mathcal{K}$ can be expressed in terms of the Lagrange functional:

$$\mathcal{K}_u(u)\delta u = \mathcal{L}_u(y,u,\zeta)\delta u = \int_0^{t_f}\int_{\Gamma_{\text{in}}} (\lambda u + (\boldsymbol{n}\cdot\varphi))\delta u.$$

With this, we can formulate the *gradient condition*:

$$\langle \mathcal{L}_u(y,u,\zeta), \tilde{u}-u\rangle = \int_0^{t_f}\int_{\Gamma_{\text{in}}} (\lambda u + (\boldsymbol{n}\cdot\varphi))(\tilde{u}-u) \geq 0, \qquad \text{for all } \tilde{u}\in\mathcal{U}_{\text{ad}}. \quad (16)$$

We employ box constraints for the control $\mathcal{U}_{\text{ad}} = \{u\in\mathcal{U} : \underline{u}\leq u(t)\leq\overline{u}, t\in[0,t_f]\}$ with lower and upper bounds $\underline{u}<\overline{u}$. The unrestricted case $\mathcal{U}_{\text{ad}}=\mathcal{U}$ can be expressed with $\underline{u}=-\infty$, $\overline{u}=\infty$. In this case, (16) simplifies to the *gradient equation*

$$0 = \lambda u + \frac{1}{|\Gamma_{\text{in}}|}\int_{\Gamma_{\text{in}}} \boldsymbol{n}\cdot\varphi, \qquad t\in(0,t_f]. \quad (17)$$

As a consequence, the required gradient of the cost functional can be expressed as

$$\nabla\mathcal{K} = \lambda u + \frac{1}{|\Gamma_{\text{in}}|}\int_{\Gamma_{\text{in}}} \boldsymbol{n}\cdot\varphi, \quad (18)$$

and is now available to be plugged into a gradient method.

Consider the optimal control problem

$$\min_{u\in\mathcal{U}_{\text{ad}}} \mathcal{K}(u).$$

Given a control $u^{k-1}\in\mathcal{U}_{\text{ad}}$, the projected gradient method [16], described in Algorithm 1, uses the negative gradient $-\nabla\mathcal{K}(u^{k-1})$ as the descent direction (Step 6).

To proceed, a step size s^k is computed in Step 5 with Algorithm 2. To ensure that the computed control is admissible, the projection $\mathbf{P}_{[\underline{u},\,\overline{u}]}\colon\mathcal{U}\to\mathcal{U}_{\text{ad}}$ is applied pointwise in time (Step 7).

$$\mathbf{P}_{[\underline{u},\,\overline{u}]}(u) := \max\{\underline{u}, \min\{u,\overline{u}\}\}.$$

Possible stopping criteria*, evaluated in Step 2 of Algorithm 1, are the norm of the step $||s^k\cdot d^k|| < \delta_1$ and the relative change of the cost functional

$$\frac{|\mathcal{K}(u^{k-1})-\mathcal{K}(u^k)|}{|\mathcal{K}(u^{k-1})|} < \delta_2, \quad (19)$$

*Details can be found in the source code (Section 7, gradient_method.py: line 100 – 114) with certain tolerances $\delta_1, \delta_2 > 0$. Besides that, we use a maximum iteration number $k_{\max}$. The choice of the step size s^k is of great significance for the performance of the projected gradient method.

Algorithm 1: Projected Gradient Method

```
Input: initial control $u^0$
Output: control $u^{k_{end}}$
1 $k = 1$
2 while not converged do
3     solve forward problem (7)
4     solve backward problem (15)
5     compute step size $s^k$
6     $d^k = \lambda u^{k-1} + \frac{1}{|\Gamma_{\text{in}}|} \int_{\Gamma_{\text{in}}} \boldsymbol{n} \cdot \varphi$
7     $u^k = \mathbf{P}_{[\underline{u},\, \overline{u}]}(u^{k-1} - s^k \cdot d^k)$
8     $k = k + 1$
9 end
```

Algorithm 2: Quadratic Line Minimization

```
Input: The step direction $d^k$
Output: step size $s$
1  $i = 1$
2  choose $s_0 = 0 < s_1 < s_2,\ \epsilon_1$
3  $k_j = \mathcal{K}(\mathbf{P}_{[\underline{u},\, \overline{u}]}(u^{k-1} - s_j \cdot d^k)), \quad j = 0, 1, 2$ // Needs 2 evaluations of (7)
       on $(0, t_f]$
4  while not converged do
5      $q \in \mathbb{P}_2 : q(s_j) = k_j, \quad j = 0, 1, 2$
6      $s = \operatorname{argmin}_{\tilde{s} \in [s_0, s_2]} q(\tilde{s})$
7      if $|s - s_2| < \epsilon_1$ then
8          $s_0 = s_1,\ k_0 = k_1,\ s_1 = s_2,\ k_1 = k_2$
9          $s_2 = 2 \cdot s_2$                // Alternative $s_2 = s_2 + s_1 - s_0$
10         $k_2 = \mathcal{K}(\mathbf{P}_{[\underline{u},\, \overline{u}]}(u^{k-1} - s_2 \cdot d^k))$   // Needs 1 evaluation of (7) on
               $(0, t_f]$
11     else if $s > s_1$ then
12         $s_0 = s_1,\ k_0 = k_1,\ s_1 = s$
13         $k_1 = \mathcal{K}(\mathbf{P}_{[\underline{u},\, \overline{u}]}(u^{k-1} - s_1 \cdot d^k))$   // Needs 1 evaluation of (7) on
               $(0, t_f]$
14         $i = i + 1$
15     else
16         $s_2 = s_1,\ k_2 = k_1,\ s_1 = s$
17         $k_1 = \mathcal{K}(\mathbf{P}_{[\underline{u},\, \overline{u}]}(u^{k-1} - s_1 \cdot d^k))$   // Needs 1 evaluation of (7) on
               $(0, t_f]$
18         $i = i + 1$
19     end
20 end
```

The algorithm to compute the step size is a modification of the method used in [20]. Three sampling points are evaluated to approximate $q(s) \approx \mathcal{K}(\mathbf{P}_{[\underline{u},\, \overline{u}]}(u^{k-1} - s \cdot d^k))$ with a quadratic polynomial $q \in \mathbb{P}_2$. The local minimum of q is used as the next sampling point to refine the approximation.

In every iteration of the Algorithm 2, the cost functional $\mathcal{K}(\mathbf{P}_{[\underline{u},\, \overline{u}]}(u^{k-1} - s_j \cdot d^k))$ must be evaluated at least once. These evaluations require the solution of the forward problem (7) and are computationally expensive. To avoid excessive computational costs in the *Quadratic Line Minimization*, we added a maximum iteration number $i_{\max}$ in Step 4 of Algorithm 2. If $i > i_{\max}$, the sampling point s_j with the smallest cost value $k_j, j = 0, 1, 2$ is returned to Algorithm 1. Otherwise, with tolerances $\epsilon_2, \epsilon_3 > 0$, the algorithm stops if the newly computed minimum s of the polynomial q is close to an already existing sampling point

$$|s - s_j| < \epsilon_2, \qquad \text{for any } j = 0, 1, 2,$$

or if the relative change of the value of $\mathcal{K}$ at the new sampling point s is small

$$\frac{|\mathcal{K}(\mathbf{P}_{[\underline{u},\, \overline{u}]}(u^{k-1} - s \cdot d^k)) - k_j|}{|k_1|} < \epsilon_3, \qquad \text{for any } j = 0, 1, 2.$$

A more detailed discussion of Algorithms 1 and 2 can be found in [5].

Complexity and Convergence

Each iteration step of the Algorithm 1 requires the solution of the forward system (Step 3), the adjoint system (Step 4) and the step size computation (Step 5).

For the forward system, linear PDEs for the temperature, the interface velocity, and the mesh movement needs to be solved in every time step. Because of the implicit time-integration, the solution for the velocity and pressure fields amounts in the solution of a nonlinear system. Since the interface graph can be updated explicitly with the interface velocity, no additional equation system needs to be solved for the interface graph. The computational costs for the solution of the forward system is dominated by the costs for the nonlinear parts, which amounts to one linear solve per step of the applied Newton method.

The numerical integration of the adjoint system, requires the solution of linear PDEs for the adjoint temperature, the adjoint velocity and the adjoint interface graph in every time step.

The step size computation with Algorithm 2 requires two evaluations of the forward system at the initial step (Step 3) and one evaluation in every further iteration step (Steps 10, 13, 17). Thus, since the evaluation of the backward system is comparatively cheap, the step size computation significantly contributes to the overall complexity of every iteration step of Algorithm 1. However, it seems to be crucial to have a fast (superlinear) growth of the search interval (Step 9) for the step size in order to make the additional costs pay off.

The convergence of Algorithm 1 strongly depends on the problem settings and the initial guess for the control. Also, different choices of the weights in the cost functional influence the convergence behavior as well as the choice of the desired interface position. Several choices for the weights, the desired interface position and the initial guess are discussed in Section 5.

4 Implementation and Discretization

In this section, we explain our discretization of the systems and the implementation in FEniCS.

Step 3 of Algorithm 1 requires the forward system to be solved numerically. The same holds for the backward system in Step 4 and the evaluation of the cost functional in Algorithm 2. The domain $\Omega \subset \mathbb{R}^2$ is partitioned with a mesh of triangles. The mesh used in our experiments in Section 5 for $t = 0$ can be found in Figure 2. The interface Γ_{int} is respected by the triangulation. It is represented explicitly by edges of the mesh ($-$). These edges move in direction $V_{\text{int}} \cdot \boldsymbol{n}_{\text{int}}$ together with Γ_{int} as illustrated in Figure 3. In order to prevent the triangulation from extreme deformation, $V_{\text{int}} \cdot \boldsymbol{n}_{\text{int}}$ is extended smoothly to V_{mesh} over the whole domain and the whole mesh is moved with V_{mesh}. Thus, the domain is discretized with a varying mesh for each time step.

The PDE systems (7) and (15) are discretized with finite elements and an implicit Euler scheme. A detailed derivation of the weak formulations of the equations in (7) and (15) can be found in [5].

For the numerical implementation, the software FEniCS 1.5.0 [1] is used in Python 2.7.6 [15] together with the Python package SciPy 0.15.1 [11].

5 Numerical Experiments

In this section, two experiments are presented to illustrate the performance of the presented optimal control approach of a Stefan problem. The experiments aim to stabilize the interface to a flat position. They demonstrate that not all desirable interface positions are reachable due to the model chosen in this work. Further, the influence of the selection of an initial guess and the importance of well-chosen weights $\lambda, \Lambda, \bar{\Lambda}$ in the cost functional are highlighted. For the two experiments, we choose the following setting, which is the same as in [5].

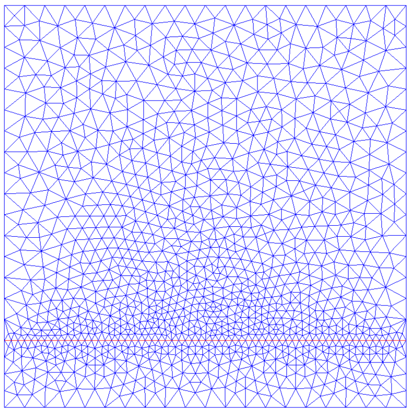


Figure 2: Triangulation of the domain $\Omega(0)$ respecting the interface position (−).

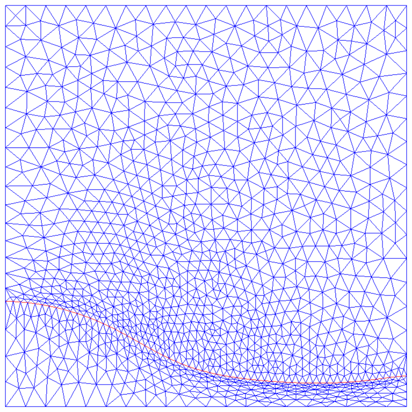


Figure 3: Triangulation of the domain $\Omega(t_f)$ respecting the moved interface position.

The domain described in Section 2 is a unit square $\Omega = [0,1] \times [0,1]$. The boundary regions are

$$\begin{aligned}
\Gamma_{\text{in}} &= \{0\} \times [0.6, 0.8], \\
\Gamma_{\text{cool}} &= [0,1] \times \{0\}, \\
\Gamma_{\text{out}} &= \{1\} \times [0.2, 0.4], \\
\Gamma_{\tilde{N}} &= [0,1] \times \{1\}, \\
\Gamma_{N} &= (\{0\} \times ([0,0.6] \cup [0.8,1])) \cup (\{1\} \times ([0,0.2] \cup [0.4,1])) \cup ([0,1] \times \{1\}),
\end{aligned}$$

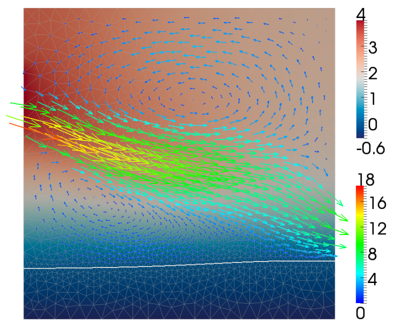
and the initial interface position is $\Gamma_{\text{int}} = [0,1] \times \{\frac{1}{6}\}$. The constants for the two-phase Stefan problem are

$$T_{\text{cool}} = -0.6,\ T_{\text{heat}} = 4,\ T_{\text{melt}} = 0,\ \eta = 0.05,\ k_s = 1,\ k_l = 0.6,\ L = 150,\ t_f = 1.$$

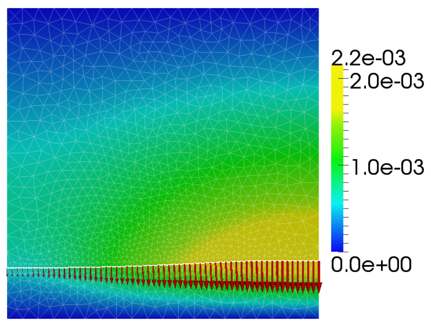
We have chosen $T_0 = 4x_2 - \frac{2}{3}$ as the initial temperature distribution. The tolerances and maximum iteration numbers of the gradient algorithm and the line minimization are

$$\delta_1 = 10^{-8},\ \delta_2 = 10^{-4},\ k_{\max} = 100,$$
$$\epsilon_1 = 10^{-12},\ \epsilon_2 = 10^{-4},\ \epsilon_3 = 10^{-4},\ \delta = 0.05,\ i_{\max} = 5.$$

Figure 4 illustrates the numerical solution of (7).



Velocity in the liquid (arrows, lower color bar) with temperature distribution (background color, upper color bar) and interface position (white line).



Interface velocity (red arrows) which is extended to the mesh movement on the whole domain (background color, magnitude plot).

Figure 4: Numerical solution of the forward problem.

The control constraints are set to

$$\underline{u} := 0 \leq u(t) \leq 20 =: \overline{u}, \qquad \text{for all } t \in [0,1].$$

In the majority of cases, the control constraints are inactive for the computed control. Nevertheless, the control constraints can become active for the sample points within the line minimization algorithm if the step size is overestimated. This behavior mainly depends on the choice of the weight parameters in the cost functional.

5.1 Experiment 1: Stabilizing to a Flat Position

The desired interface position is a straight line moving from the start position at $x_2 = \frac{1}{6}$ to $x_2 = 0.166$:

$$f_d(x_1, t) = \frac{1}{6} - t \cdot (\frac{1}{6} - 0.166), \qquad t \in [0,1].$$

The weight parameters in the cost functional are set to

$$\Lambda = 100, \ \bar{\Lambda} = 0, \ \lambda = 10^{-10}.$$

So, the cost functional primarily measures the distance of the interface to the desired interface at the end of the time interval and does not track the interface position for all points in time. Since the two-phase Stefan problem is non-linear, the cost functional must be assumed non-convex [7]. Consequently,

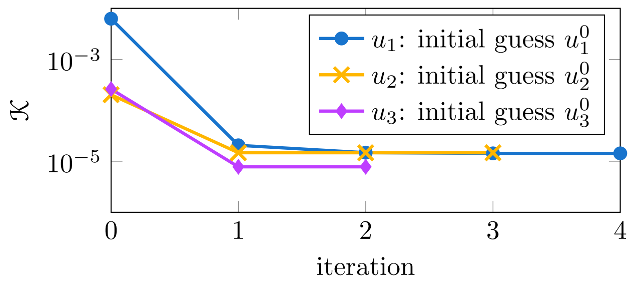


Figure 5: Cost functional for different initial guesses.

the projected gradient algorithm can only approximate stationary points of the cost functional. To which stationary point the algorithm converges, primarily depends on the initial guess. The following functions u_1^0, u_2^0, and u_3^0 are taken as initial guesses for the projected gradient algorithm to compute the controls u_1, u_2, and u_3.

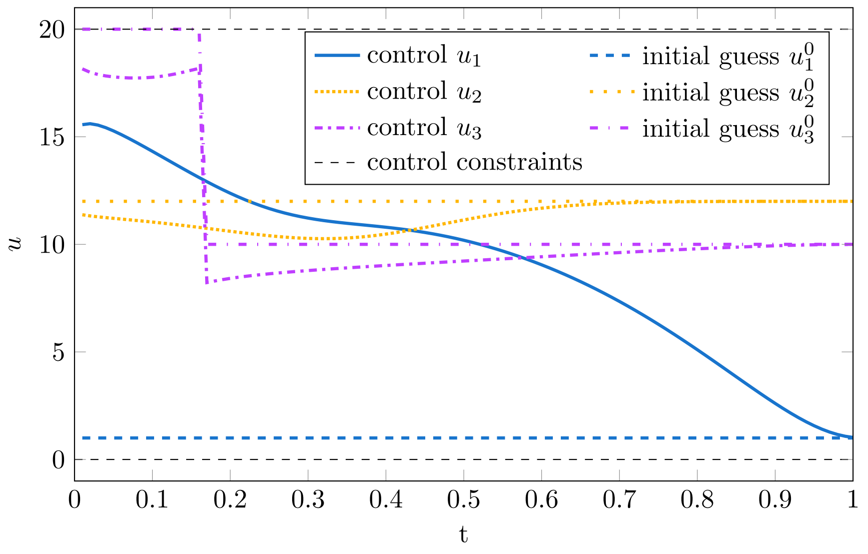


Figure 6: Computed controls u_1, u_2, u_3 with different initial guesses.

$$\begin{aligned} u_1^0 &\equiv 1, \\ u_2^0 &\equiv 12, \\ u_3^0(t) &= \begin{cases} 20, & t \in [0, 0.17], \\ 10, & t \in (0.17, 1]. \end{cases} \end{aligned}$$

Due to the initial values, where the interface is located close to the cooling boundary, the interface always moves upwards in the beginning of a forward simulation. The initial guesses u_2^0 and u_3^0 tend to induce higher velocities of the fluid at the beginning of the time interval to prevent the interface from moving upwards. Through this, the algorithm is expected to show better convergence behavior.

Since the described problem domain is not symmetric, the interface movement is not symmetric, as illustrated by the uncontrolled interface graph in Figure 7. This implies that the controlled interface can not be expected to be completely flat and to match the desired interface perfectly.

The presented algorithm is able to compute a control u_1 after 4 iteration steps, which keeps the interface close to the desired interface. It mainly acts at the beginning of the time interval (see Figure 6) to stop the interface from moving upwards and moves it back downwards to the desired position.

The control constraints are inactive at all points. We introduce the quantities

$$\mathfrak{d} := \int\limits_{\Gamma_{\text{cool}}} (f(t_f, x_1) - f_d(t_f, x_1))^2, \qquad \mathfrak{d}_{\text{all}} := \int\limits_0^{t_f} \int\limits_{\Gamma_{\text{cool}}} (f(t, x_1) - f_d(t, x_1))^2,$$

$$\mathfrak{p} := \int\limits_0^{t_f} \int\limits_{\Gamma_{\text{in}}} (u(t))^2, \qquad \#\mathbf{it} := \text{number of iterations},$$

that quantify the distance and control cost terms from the cost functional and the number of iterations for the outcomes of the optimizations using Algorithm 1. The results for the different controls are listed in Table 1.

The algorithm stops after 3 iterations with the control u_2. As expected, it converges slightly faster with the initial guess u_2^0 than with u_1^0 (see Figure 5) but does not reach a considerable smaller cost value. On the contrary, the algorithm converges clearly faster towards u_3, which also has a notable smaller cost value. In this case, it stopped after 2 iterations. Looking at the computed controls in Figure 6, the algorithm appears to converge to completely different stationary points which result in different interface graphs (see Figure 7). Again the control constraints are inactive for all controls and all points in time.

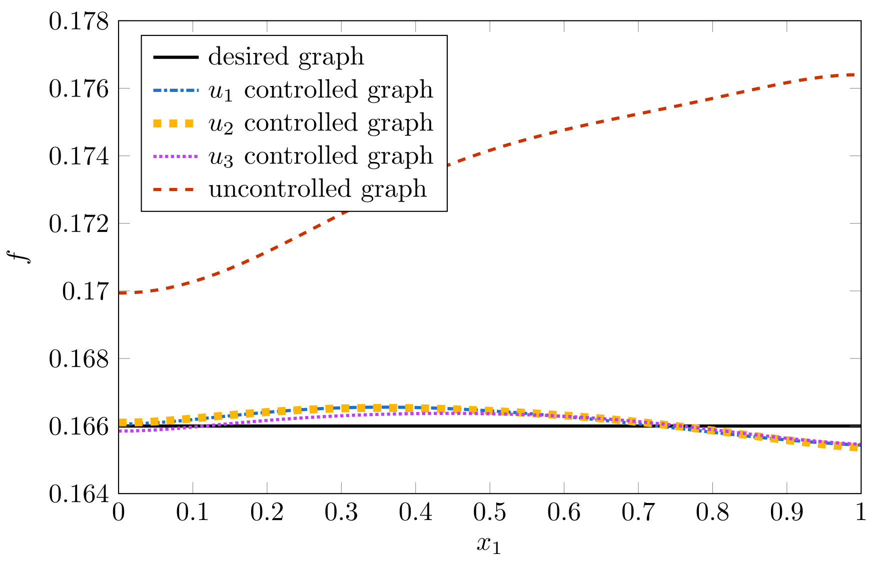


Figure 7: Interface graphs for the controls u_1, u_2, u_3.

Overall, the stationary points and thus the interface positions to which Algorithm 1 converges strongly depend on the initial guess.

Due to the already mentioned asymmetry of the problem domain, the interface positions for the computed controls clearly deviate from the desired interface and do not match it perfectly. To improve this situation, the performance of Algorithm 1 is analyzed for an actually reachable desired interface position in the next experiment.

5.2 Experiment 2: Stabilizing to a Reachable Flat Position

We run the forward simulation with the control $\tilde{u}_d$:

$$\tilde{u}_d := \begin{cases} 20, & t \in [0, 0.17], \\ 8.5, & t \in (0.17, 0.25], \\ 7.5, & t \in (0.25, 0.63), \\ 8.5, & t \in [0.63, 1]. \end{cases}$$

The resulting interface position and corresponding interface graph f_d are clearly reachable. We use this graph f_d as the desired graph for the optimiza-

Table 1: Distance and control cost terms for the controls u_1, u_2, u_3.

	u_1	u_2	u_3
$\mathfrak{d}$	$1.4317 \cdot 10^{-7}$	$1.4710 \cdot 10^{-7}$	$7.7936 \cdot 10^{-8}$
$\mathfrak{d}_{\text{all}}$	$7.0578 \cdot 10^{-7}$	$1.7631 \cdot 10^{-6}$	$6.0702 \cdot 10^{-7}$
$\mathfrak{p}$	15.3800	19.9472	19.4810
#it	4	3	2

tion problem and set the weight parameters and initial guess as

$$\Lambda = 10^5, \ \bar{\Lambda} = 0, \ \lambda = 10^{-10}, \\ \tilde{u}^0 \equiv 1.$$

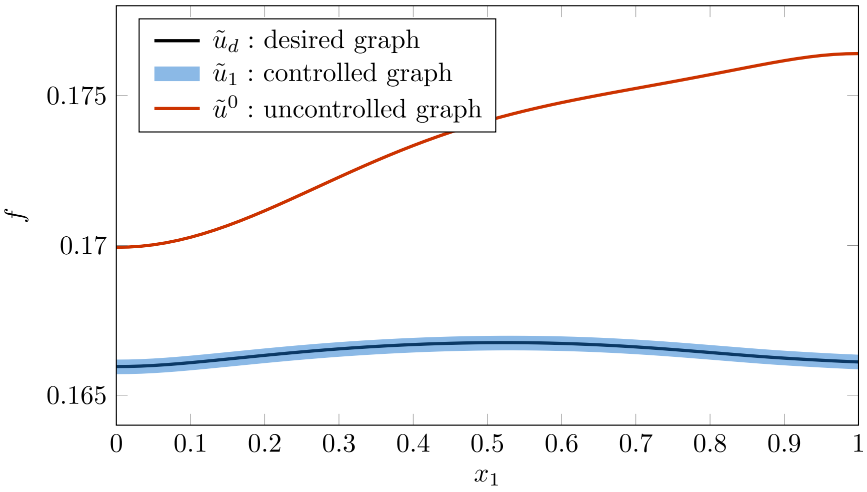


Figure 8: Interface graphs with a reachable interface position $\tilde{u}_d$.

The control $\tilde{u}_1$, computed by the projected gradient algorithm, is able to approximate the desired interface position closely. It can be seen in Figure 8, that it is almost indistinguishable from the desired interface graph. The projected gradient algorithm converges after 37 iteration steps with a significantly more accurate interface approximation than in the first experiment. The higher iteration count compared to the first experiment is due to the higher accuracy of this experiment. The stopping criterion (19) is not satisfied in the first iteration steps as for the first experiment.

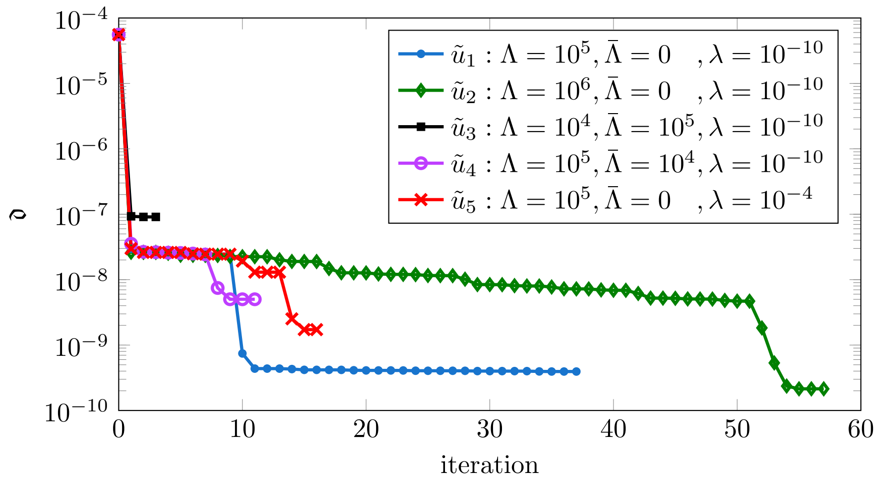


Figure 9: Interface distance $\mathfrak{d}$ at $t = t_f$ for different parameter sets.

The convergence behavior is influenced by the choice of the weight factors in the cost functional. Running the algorithm with different sets of weights

$$\begin{aligned}
&\tilde{u}_1 : \Lambda = 10^5, \bar{\Lambda} = 0 \quad , \lambda = 10^{-10},\\
&\tilde{u}_2 : \Lambda = 10^6, \bar{\Lambda} = 0 \quad , \lambda = 10^{-10},\\
&\tilde{u}_3 : \Lambda = 10^4, \bar{\Lambda} = 10^5, \lambda = 10^{-10},\\
&\tilde{u}_4 : \Lambda = 10^5, \bar{\Lambda} = 10^4, \lambda = 10^{-10},\\
&\tilde{u}_5 : \Lambda = 10^5, \bar{\Lambda} = 0 \quad , \lambda = 10^{-4},
\end{aligned}$$

changes the convergence speed and quality of the computed control. The algorithm shows a different convergence behavior if only the weight factor Λ is changed as for the control $\tilde{u}_2$ (see Table 2). Weight factors that produce good results for this experiment do not necessarily produce good results for Experiment 1 and vice versa. With $\bar{\Lambda} \neq 0$, the interface position is tracked over the whole time interval by the cost functional. Since the cost functional is not comparable among these parameter sets, instead of the cost functional, the two distances $\mathfrak{d}$ and $\mathfrak{d}_{\text{all}}$ are displayed in Figures 9 and 10. In case of the controls $\tilde{u}_3$ and $\tilde{u}_4$, the interface can be moved closer to the desired position over the whole time interval at the expense of a larger distance at $t = t_f$.

The factor λ, which penalizes the control cost $\mathfrak{p}$ in the cost functional, can be used to reduce the control cost and acts as a regularization. This could

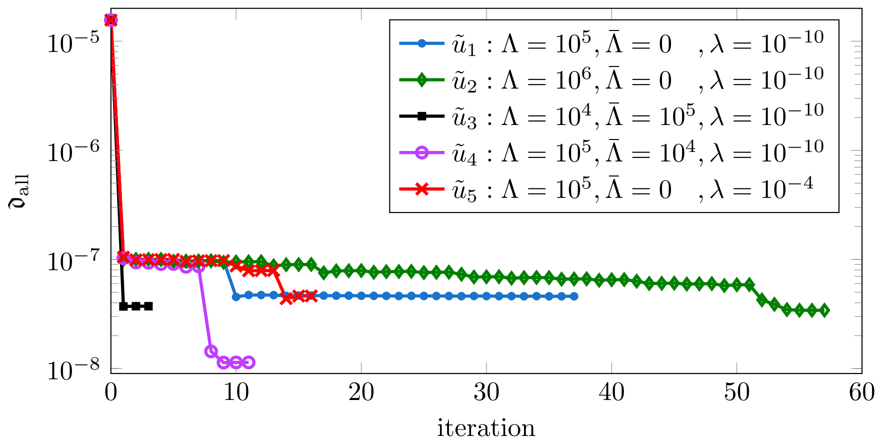


Figure 10: Interface distance $\mathfrak{d}_{\text{all}}$ over the whole time interval for different parameter sets.

make the control constraints dispensable. For the control $\tilde{u}_5$ with $\lambda = 10^{-4}$, the control cost $\mathfrak{p}$ can be reduced slightly (see Table 2), but this also leads to higher distances $\mathfrak{d}$ and $\mathfrak{d}_{\text{all}}$. By further increasing λ, the control cost $\mathfrak{p}$ can also be reduced further, but again at the expense of higher distances $\mathfrak{d}$ and $\mathfrak{d}_{\text{all}}$ [5].

Additional combinations of weight factors and experiments where the desired interface moves upwards can be found in [5].

Table 2: Distance and control cost terms for the controls $\tilde{u}_1 - \tilde{u}_5$.

	$\tilde{u}_1$	$\tilde{u}_2$	$\tilde{u}_3$	$\tilde{u}_4$	$\tilde{u}_5$
$\mathfrak{d}$	$3.9422 \cdot 10^{-10}$	$2.1424 \cdot 10^{-10}$	$9.1034 \cdot 10^{-8}$	$5.0140 \cdot 10^{-9}$	$1.7234 \cdot 10^{-9}$
$\mathfrak{d}_{\text{all}}$	$4.5865 \cdot 10^{-8}$	$3.4164 \cdot 10^{-8}$	$3.7233 \cdot 10^{-8}$	$1.1374 \cdot 10^{-8}$	$4.6233 \cdot 10^{-8}$
$\mathfrak{p}$	15.6606	15.7936	13.8367	15.3050	15.2697
#it	37	57	3	11	16

6 Conclusions and Perspectives

We introduced an approach for the optimal control of the interface position in a Stefan problem fully coupled to the Navier–Stokes equations. Compared to existing research, the new problem setting has increased in complexity. The mesh movement method used in this work is able to track the moving boundary and is fully included into the PDE systems.

To our extent of knowledge, this is the first attempt to combine mesh movement methods and finite elements for the optimal control of a two-dimensional two-phase Stefan problem. The control of the inflow pressure acts relatively indirect on the interface position, which makes the control of this non-linear problem a challenging task.

We developed the formulation of an adjoint system and, as a result of this, the first-order necessary optimality conditions using a formal Lagrange approach. Revealed by this, the gradient of the quadratic tracking-type cost functional can be used for a projected gradient algorithm. As illustrated with two numerical experiments, this algorithm can approach the desired state accurately. A powerful quadratic line minimization algorithm is integrated into the gradient method. Moreover, the experiments have demonstrated that such a method of steepest descent is limited to approximate stationary points and is heavily dependent on the choice of the cost functional. Weight factors that lead to good results in one setting can be an inappropriate selection for another setting and vice versa. Thus, a general purpose selection strategy is not close at hand. Further, we provide the numerical implementation of the PDE systems, mesh movement techniques, and algorithms in Python with the usage of FEniCS and SciPy.

The approach we propose showed its potential in several numerical experiments. Testing the algorithms for other settings would be interesting. Some more simple possibilities are to change the initial interface position to something else than a straight line or to change the shape of the domain. Extending the model to three spatial dimensions or to an m-phase Stefan problem should be realizable. Certainly, this would cause additional work, not only on the theoretical, but especially on the implementation side.

From the mathematical point of view, higher-order optimality conditions and consequently higher-order methods [9], are desirable. They might lead to faster convergence saving some of the computationally intensive approximations. Neither the existence nor the uniqueness of solutions of the two-phase Stefan problem, as formulated here, are known. A first step would be the derivation of a rigorous functional analytical framework for the problem [12]. Besides the quadratic cost functional used in our work, others, for example of L^1 or L^∞ type, might be of interest. Additionally, the curvature of the interface graph could be used as a measure of the flatness of the interface in the cost functional.

If the optimal control approach in this work is investigated satisfactorily, the next major step is to develop a closed loop optimal control system for the two-phase Stefan problem. For this boundary feedback stabilization approach, recent developments in the LQR/LQG controller design for incompressible flows [2, 6] might be applicable.

7 Code Availability

The source code of the implementations used to compute the presented results can be obtained from:

`https://gitlab.mpi-magdeburg.mpg.de/baran/Stefan_Problem_in_FEniCS.git`
with the tag publication_2017 and is authored by: Björn Baran

Please contact Björn Baran for licensing information.

References

[1] M. S. Alnæs, J. Blechta, J. Hake, A. Johansson, B. Kehlet, A. Logg, C. Richardson, J. Ring, M. E. Rognes, and G. N. Wells. The FEniCS project version 1.5. *Archive of Numerical Software*, 3(100), 2015.

[2] E. Bänsch, P. Benner, J. Saak, and H. K. Weichelt. Riccati-based boundary feedback stabilization of incompressible Navier–Stokes flows. *SIAM J. Sci. Comput.*, 37(2):A832–A858, 2015.

[3] E. Bänsch, J. Paul, and A. Schmidt. An ALE FEM for solid-liquid phase transitions with free melt surface. *Berichte aus der Technomathematik 10-07*, 2010.

[4] E. Bänsch, J. Paul, and A. Schmidt. An ALE finite element method for a coupled Stefan problem and Navier-Stokes equations with free capillary surface. *International Journal for Numerical Methods in Fluids*, 71(10):1282–1296, 2013.

[5] B. Baran. Optimal control of a Stefan problem with gradient-based methods in FEniCS. Master's thesis, Otto-von-Guericke-Universität, Magdeburg, Germany, 2016. `http://nbn-resolving.de/urn:nbn:de:gbv:ma9:1-7871`.

[6] P. Benner and J. Heiland. LQG-balanced truncation low-order controller for stabilization of laminar flows. In R. King, editor, *Active Flow and Combustion Control 2014*, volume 127 of *Notes on Numerical Fluid Mechanics and Multidisciplinary Design*, pages 365–379. Springer International Publishing, 2015.

[7] M. Bernauer. *Motion Planning for the Two-Phase Stefan Problem in Level Set Formulation.* PhD thesis, Technische Universität Chemmitz, Chemnitz, Germany, 2010.

[8] R. Glowinski. *Finite element methods for the numerical simulation of incompressible viscous flow. Introduction to the control of the Navier–Stokes equations.* Lectures in Applied Mathematics 28 (1991): 219-301, 1991.

[9] M. Hinze and K. Kunisch. Second order methods for optimal control of time-dependent fluid flow. *SIAM J. Cont. Optim.*, 40(3):925–946, 2001.

[10] M. Hinze and S. Ziegenbalg. Optimal control of the free boundary in a two-phase Stefan problem with flow driven by convection. *ZAMM - Journal of Applied Mathematics and Mechanics / Zeitschrift für Angewandte Mathematik und Mechanik*, 87(6):430–448, 2007.

[11] E. Jones, T. Oliphant, P. Peterson, et al. SciPy: Open source scientific tools for Python. `http://www.scipy.org/`, 2001–.

[12] R. H. Nochetto, M. Paolini, and C. Verdi. An adaptive finite element method for two-phase Stefan problems in two space dimensions. I: Stability and error estimates. *Mathematics of Computation*, 57(195):73–108, 1991.

[13] R. H. Nochetto, M. Paolini, and C. Verdi. An adaptive finite element method for two-phase Stefan problems in two space dimensions. II: Implementation and numerical experiments. *SIAM J. Sci. Statist. Comput.*, 12(5):1207–1244, 1991.

[14] P. Rudolph and M. Jurisch. Bulk growth of GaAs an overview. *Journal of Crystal Growth*, 198–199, Part 1:325–335, 1999.

[15] Stichting Mathematisch Centrum, Amsterdam, The Netherlands. Python. `http://www.python.org`, 1991–1995.

[16] F. Tröltzsch. *Optimal Control of Partial Differential Equations - Theory, Methods and Applications.* Graduate Studies in Mathematics, Vol. 112. American Mathematical Society, Providence, Rhode Island, 2010. (Translation of the second German edition below by J. Sprekels).

[17] P. N. Vabishchevich. *Computational Technologies. Advanced Topics.* Walter de Gruyter GmbH & Co. KG, 2014.

[18] R. E. White. An enthalpty formulation of the Stefan problem. *SIAM J. Numer. Anal.*, 19(6):1129–1157, 1982.

[19] N. Zabaras, B. Ganapathysubramanian, and L. Tan. Modelling dendritic solidification with melt convection using the extended finite element method. *J. Comput. Phys.*, 218(1):200 – 227, 2006.

[20] S. Ziegenbalg. *Kontrolle freier Ränder bei der Erstarrung von Kristallschmelzen.* PhD thesis, Technische Universität Dresden, Dresden, Germany, 2008. In German.

Björn BARAN,
Computational Methods in Systems and Control Theory,
Max Planck Institute for Dynamics of Complex Technical Systems,
Sandtorstr. 1, D-39106 Magdeburg, Germany.
Email: baran@mpi-magdeburg.mpg.de, orcid.org/0000-0001-6570-3653

Peter BENNER,
Computational Methods in Systems and Control Theory,
Max Planck Institute for Dynamics of Complex Technical Systems,
Sandtorstr. 1, D-39106 Magdeburg, Germany.

Jan HEILAND,
Computational Methods in Systems and Control Theory,
Max Planck Institute for Dynamics of Complex Technical Systems,
Sandtorstr. 1, D-39106 Magdeburg, Germany.

Jens SAAK,
Computational Methods in Systems and Control Theory,
Max Planck Institute for Dynamics of Complex Technical Systems,
Sandtorstr. 1, D-39106 Magdeburg, Germany.