# IMPLEMENTATION OF ISP

ALI HAQVERDIYEV AND JONATHAN GOLDFARB

Contents



## 1. Introduction

sec:inverse stefan problem

*Date*: May 7, 2019.
2010 *Mathematics Subject Classification.* TBD.

1.1. **Inverse Stefan Problem**. Consider the general one-phase Stefan problem:

eq:intro-pde
$$Lu \equiv (a(t)b(x)u_x)_x - u_t = f, \text{ in } \Omega \tag{1}$$

eq:intro-init
$$u(x,0) = \phi(x),\ 0 \le x \le s(0) = s_0 \tag{2}$$

eq:pde-bound
$$a(t)b(0)u_x(0,t) = g(t),\ 0 \le t \le T \tag{3}$$

eq:pde-stefan
$$a(t)b(s(t))u_x(s(t),t) + \gamma(s(t),t)s'(t) - \chi(s(t),t) = 0,\ 0 \le t \le T \tag{4}$$

eq:pde-freebound
$$u(s(t),t) = \mu(t),\ 0 \le t \le T, \tag{5}$$

where

eq:pde-domain-defn
$$\Omega = \{(x,t) : 0 < x < s(t),\ 0 < t \le T\} \tag{6}$$

where $a$, $b$, $f$, $\phi$, $g$, $\gamma$, $\chi$, $\mu$ are given functions. Assume now that $a$ is unknown; in order to find $a$, along with $u$ and $s$, we must have additional information. Assume that we are able to measure the temperature on our domain and the position of the free boundary at the final moment $T$.

eq:pde-finaltemp
$$u(x,T) = w(x),\ 0 \le x \le s(T) = s_*. \tag{7}$$

Under these conditions, we are required to solve an *inverse* Stefan problem (ISP): find a tuple

$$\{u(x,t), s(t), a(t)\}$$

that satisfy conditions (**??**)–(**??**). ISP is not well posed in the sense of Hadamard: the solution may not exist; if it exists, it may not be unique, and in general it does not exhibit continuous dependence on the data. The main methods available for ISP are based on a variational formulation, Frechet differentiability, and iterative gradient methods. We cite recent papers [**?**, **?**] and the monograph [**?**] for a list of references. The established variational methods in earlier works fail in general to address two issues:

- The solution of ISP does not depend continuously on the phase transition temperature $\mu(t)$ from (**??**). A small perturbation of the phase transition temperature may imply significant change of the solution to the inverse Stefan problem.
- In the existing formulation, at each step of the iterative method a Stefan problem must be solved (that is, for instance, the unknown heat flux $g$ is given, and the corresponding $u$ and $s$ are calculated) which incurs a high computational cost.

A new method developed in [**?**, **?**] addresses both issues with a new variational formulation. The key insight is that the free boundary problem has a similar nature to an inverse problem, so putting them into the same framework gives a conceptually clear formulation of the problem; proving existence, approximation, and differentiability is a resulting challenge. Existence of the optimal control and the convergence of the sequence of discrete optimal control problems to the continuous optimal control problem was proved in [**?**, **?**]. In [**?**, **?**], Frechet differentiability of the new variational formulation was developed, and a full result showing Frechet differentiability and the form of the Frechet differential with respect to the free boundary, sources, and coefficients were proven. Lastly, the gradient method was implemented in [**?**] and numerical results were proven. Our goal in this work is to extend the implementation to the identification of the diffusion coefficient $a(t)$ and document the implementation in MATLAB using the built-in PDEPE solver.

The rest of this note is as follows:

## 2. Variational Formulation

sec:variational-formulation

We will consider the problem (??)-(??) where $(s,a) \in V_R$ is fixed, and

$$V_R = \Big\{ v = (s,a) \in H, |v|_H \le R;\ s(0) = s_0,\ g(0) = a(0)b(0)\phi'(0),\ a_0 \le a(t)$$

eq:control-set

$$\chi(s_0,0) = \phi'(s_0)a(0)b(s_0) + \gamma(s_0,0)s'(0),\ 0 < \delta \le s(t) \Big\}, \tag{8}$$

where $\beta_0, \beta_1, \beta_2 \ge 0$ and $a_0, \delta, R > 0$ are given, and

$$H := W_2^2(0,T) \times W_2^1(0,T)$$
$$\|v\|_H := \max\left( \|s\|_{W_2^2(0,T)}, \|a\|_{W_2^1(0,T)} \right)$$

Define

$$D := \{(x,t) : 0 \le x \le \ell,\ 0 \le t \le T\},$$

where $l = l(R) > 0$ is chosen such that for any control $v \in V_R$, its component $s$ satisfies $s(t) \le l$.

Consider the minimization of the functional

eq:functional

$$\mathcal{J}(v) = \beta_0 \int_0^{s(T)} |u(x,T;v) - w(x)|^2\ dx + \beta_1 \int_0^T |u(s(t),t;v) - \mu(t)|^2\ dt + \beta_2 |s(T) - s_*|^2 \tag{9}$$

on the control set (??). The formulated optimal control problem (??), (??)-(??), (??) will be called Problem $\mathcal{I}$.

sec:frechet-gradient

### 2.1. **Gradient Result**.

defn:adjoint

**Definition 1**. For given $v$ and $u = u(x,t;v)$, $\psi$ is a solution to the adjoint problem if

eq:adj-pde

$$L^*\psi := \big(ab\psi_x\big)_x + \psi_t = 0, \quad \text{in } \Omega \tag{10}$$

eq:adj-finalmoment

$$\psi(x,T) = 2\beta_0(u(x,T) - w(x)),\ 0 \le x \le s(T) \tag{11}$$

eq:adj-robin-fixed

$$b(0)a(t)\psi_x(0,t) = 0,\ 0 \le t \le T \tag{12}$$

eq:adj-robin-free

$$\Big[ab\psi_x - s'\psi - 2\beta_1(u-\mu)\Big]_{x=s(t)} = 0,\ 0 \le t \le T \tag{13}$$

thm:gradient-result

**Theorem 1**. *Problem $\mathcal{I}$ has a solution, and the functional $\mathcal{J}(v)$ is differentiable in the sense of Frechet, and the first variation is*

$$\langle \mathcal{J}'(v), \Delta v \rangle_H = \int_0^T \Big[ 2\beta_1 \big(u-\mu\big) u_x + \psi\big(\chi_x - \gamma_x s' - a(bu_x)_x\big) \Big]_{x=s(t)} \Delta s(t)\, dt$$
$$+ \Big[ \beta_0 |u(s(T),T) - w(s(T))|^2 + 2\beta_2(s(T) - s_*) \Big] \Delta s(T) - \int_0^T \big[\psi\gamma\big]_{x=s(t)} \Delta s'(t)\, dt$$

eq:gradient-full

$$+ \int_0^T \Delta a \left[ \int_0^{s(t)} (bu_x)_x \psi\, dx - [bu_x\psi]_{x=s(t)} - [bu_x\psi]_{x=0} \right] dt \tag{14}$$

*where $\mathcal{J}'(v) \in H'$ is the Frechet derivative, $\langle \cdot, \cdot \rangle_H$ is a pairing between $H$ and its dual $H'$, $\psi$ is a solution to the adjoint problem in the sense of definition ??, and $\delta v = (\Delta s, \Delta a)$ is a variation of the control vector $v \in V_R$ such that $v + \delta v \in V_R$.*

2.2. **General Form of PDE Problem**. Both the forward problem (??)-(??) and the adjoint problem (??)-(??) have a common structure that can be exploited for implementation in code. Consider the problem

eq:general-pde
$$(a(t)b(x)u_x)_x - u_t = f(x,t),\ 0 < x < s(t),\ 0 < t < T \tag{15}$$

eq:general-init
$$u(x,0) = \phi(x),\ 0 < x < s(0) \tag{16}$$

eq:general-fixedbdy
$$a(t)b(0)u_x(0,t) = g(t),\ 0 < t < T \tag{17}$$

eq:general-movingbdy
$$a(t)b(s(t))u_x(s(t),t) + r(t)u(s(t),t) - p(t) = 0,\ 0 < t < T \tag{18}$$

For the forward problem, we have $r(t) \equiv 0$ and $p(t) = \gamma(s(t),t) - \chi(s(t),t)s'(t)$. Under the change of variables $\bar{t} = T - t$, the problem (??)-(??) is of the same form after taking $s(t) = s(T-t)$, $a(t) = a(T-t)$, etc. The initial condition becomes $\phi(x) = 2\beta_0(u(x,T) - w(x))$, the heat flux on the fixed boundary is $g(t) \equiv 0$, the coefficient $r(t) = -s'(t)$, and the function $p(t) = -2\beta_1(u - \mu)$.

In this work, we consider a model with reduced complexity in order to test the main algorithm in isolation; in particular, we take $\beta_i \equiv 1$, $b \equiv 1$, $\gamma \equiv 1$, $\chi \equiv 0$.

To solve the problem (??)-(??) for a fixed boundary curve $x = s(t)$ corresponding to either of the two situaions described above, we transform the domain $\Omega$ to the cylindrical domain $Q_T = (0,1) \times (0,T)$ by the change of variables $y = x/s(t)$. Let $d = d(x,t)$, $(x,t) \in \Omega$ stand for any of $u$, $f$, $\gamma$, $\chi$, define the function $\tilde{d}$ by

$$\tilde{d}(y,t) = d\big(ys(t),t\big),\ \tilde{b}(y) = b(ys(t)),\ \text{and } \tilde{\phi}(y) = \phi\big(ys_0\big)$$

The transformed function $\tilde{u}$ is a ***pointwise a.e.*** solution of the Neumann problem

$$\frac{a(t)}{s^2(t)}(b(ys(t))u_y)_y - \left(u_t - \frac{ys'(t)}{s(t)}u_y\right) = f(ys(t),t),\ \text{in } Q_T$$

$$u(y,0) = \phi(ys(t)),\ 0 < y < 1$$

$$\frac{a(t)b(0)}{s(t)}u_y(0,t) = g(t),\ 0 < t < T$$

$$\frac{a(t)b(s(t))}{s(t)}u_y(1,t) + r(t)u(1,t) - p(t) = 0,\ 0 < t < T$$

which can be written in the form

eq:tform-pde
$$su_t = \frac{a}{s}(\tilde{b}u_y)_y + ys'u_y - s\tilde{f}(y,t),\ \text{in } Q_T \tag{19}$$

eq:tform-iv
$$\tilde{u} = \tilde{\phi},\ 0 \le y \le 1 \tag{20}$$

eq:tform-lbdy
$$\frac{a\tilde{b}}{s}\tilde{u}_y - g|_{y=0} = 0,\ 0 \le t \le T \tag{21}$$

eq:tform-rbdy
$$\frac{a\tilde{b}}{s}u_y + ru - p|_{y=1} = 0,\ 0 \le t \le T \tag{22}$$

## 3. Details of MATLAB Implementation

The built-in MATLAB PDE solver, `pdepe`, solves parabolic-elliptic problems in one space dimension. In particular, it uses a method-of-lines technique and a differential-algebraic equation solver applied to problems of the form

eq:matlab-pde
$$c\left(y,t,u,\frac{\partial u}{\partial y}\right)\frac{\partial u}{\partial t} = y^{-m}\frac{\partial}{\partial y}\left(y^m F\left(y,t,u,\frac{\partial u}{\partial x}\right)\right) + d\left(y,t,u,\frac{\partial u}{\partial y}\right) \tag{23}$$

for

$$a \le y \le b, \quad t_0 \le t \le t_f \tag{24}$$

where $m = 0, 1, 2$ is fixed, $c$ is a diagonal matrix of size $n \times n$, where there are $n$ components in $u$. There must be at least one parabolic equation, which corresponds to the condition of at least one component of $c$ being positive. The solution components satisfy

eq:matlab_ic
$$u(y, t_0) = u_0(y), \ a \le y \le b \tag{25}$$

and boundary conditions of the form

eq:matlab_bc
$$p(y, t, u) + q(y, t) F\left(y, t, u, \frac{\partial u}{\partial y}\right)\Big|_{y=a,b} = 0 \tag{26}$$

In particular, the function $F$ appearing in the boundary condition is the same as the flux term in the PDE. Choose $m = 0$, $a = 0$, $b = 1$, $t_0 = 0$, $t_f = T$ and compare (??)–(??) to (??)–(??) to find

$$\begin{aligned} c(y, t, u, u_y) &:= s(t) \\ F(y, t, u, u_y) &:= \frac{\tilde{b}(y)a(t)}{s(t)} u_y \\ c(y, t, u, u_y) &:= y s'(t) u_y - s\tilde{f}(y, t) \\ q(y, t) &:= 1 \\ p(y, t, u) &:= \begin{cases} -g, & y = 0 \\ ru - p, & y = 1 \end{cases} \end{aligned}$$

The function `pdeSolver`, included in Section ??, solves problems in the previous formulation with $f \equiv 0$ and $\tilde{b} \equiv 1$. The necessary setup and calculation of traces for the forward problem are completed in the function `Forward`, included in Section ??. The model problem currently implemented in `test_Forward`, included in Section ?? can be found in Section ??.

The functional $J$ and the calculation of the state vector is completed in the function `Functional`, included in Section ??.

The corresponding routines for the adjoint problem are implemented in `Adjoint`, included in Section ??. The same problem is used in `test_Forward` and in `test_Adjoint` (taking the appropriate boundary measurements; see Section ??) and we simply check that the adjoint problem vanishes as synthetic error introduced in the measurements from the forward problem vanishes.

3.1. **Implementation of Gradient Formulae in MATLAB**. The last step in the implementation is to write the gradient update formula in MATLAB notation. For the differential with respect to $x = s(t)$, we have

$$\begin{aligned} \delta J_s := \int_0^T \Big[2\beta_1\big(u - \mu\big)u_x + \psi\big(\chi_x - \gamma_x s' - a(bu_x)_x\big)\Big]_{x=s(t)} \Delta s(t)\, dt \\ + \Big[\beta_0 \left|u(s(T), T) - w(s(T))\right|^2 + 2\beta_2(s(T) - s_*)\Big] \Delta s(T) - \int_0^T \big[\psi\gamma\big]_{x=s(t)} \Delta s'(t)\, dt \end{aligned}$$

Under the assumptions used to formulate "reduced" models above ($\gamma \equiv 1$, $\chi \equiv 0$, $b \equiv 1$, $\beta_i \equiv 1$) we have

$$\delta J_s = \int_0^T \Big[2\big(u-\mu\big)u_x - \psi a u_{xx}\Big]_{x=s(t)} \Delta s(t)\, dt$$
$$+ \Big[|u(s(T),T) - w(s(T))|^2 + 2(s(T)-s_*)\Big]\Delta s(T) - \int_0^T \psi(s(t),t)\Delta s'(t)\, dt$$

Pursuing integration by parts with respect to time, it follows that the corresponding differential term is

$$\delta J_s = \int_0^T \Big[2\big(u-\mu\big)u_x - \psi a u_{xx} + \psi_x s'(t) + \psi_t\Big]_{x=s(t)} \Delta s(t)\, dt$$
eq:gradient-wrt-s
$$+ \big[\big[u-w\big]\big[u-w-2\big] + 2(s(T)-s_*)\big]\,|_{(x,t)=(s(T),T)}\Delta s(T) \tag{27}$$

Interpreting the integrand in the first term as a pointwise function is one way to write the gradient calculation in discrete form; this is the routine implemented in `grad_s`, included in Section **??**.

The differential with respect to $a(t)$ is given by

$$\delta J_a := \int_0^T \Delta a \left[\int_0^{s(t)} u_{xx}\psi\, dx - [u_x\psi]_{x=0} - [u_x\psi]_{x=s(t)}\right] dt$$

Note that

$$\int_0^{s(t)} u_{xx}\psi\, dx = -\int_0^{s(t)} u_x\psi\, dx + u_x(s(t),t)\psi(s(t),t) - u_x(0,t)\psi(0,t) \tag{28}$$

so the gradient with respect to $a$ can be written in the form

eq:gradient-wrt-a
$$J_a(t) = -\int_0^{s(t)} u_x\psi_x\, dx - 2[u_x\psi]_{x=0} \tag{29}$$

This final form of the gradient with respect to $a$ is implemented in `grad_a.m`, included below in Section **??**.

3.2. **Sobolev Preconditioning**. Note that the gradient formulae (**??**), (**??**) do not, in general, agree precisely with the definition of the Frechet gradient of $s$ and $a$, respectively; for example, since $s \in W_2^2$, we should require the gradient $J_s$ be in the form

$$\mathcal{J}(s+\Delta s) - \mathcal{J}(s) = \langle J_s, \Delta s\rangle_{W_2^2(0,T)} + o(\Delta s)$$
$$= \int_0^T \left[J_s(t)\Delta s(t) + J_s'(t)\Delta s'(t) + J_s''(t)\Delta s''(t)\right]\, dt + o(\Delta s)$$

but the differential of eq. (**??**) has the form

$$\delta J_s = \int_0^T \tilde{J}_s(t)\Delta s(t)\, dt + \tilde{\tilde{J}}\Delta s(T)$$

Note that the above functional *is* formally a linear functional over $W_2^2$; functions in $W_2^2(0,T)$ are continuous, so the delta function is a member of the dual space. By Reisz representation formula, any such functional over $W_2^2$ can be written in the required form; what remains is to numerically approximate the Reisz element $J_s$ of the gradient, which we refer to as ***preconditioning*** (in particular, Sobolev preconditioning) the gradient.

This process is shown to be essential for the numerical results; without preconditioning, the original set of "$L_2$" gradients do not give a clear enough search direction for the gradient descent process. The

geometric interpretation of this can be thought of as a mathematically motivated preconditioning or smoothing operation.

Since the gradient is used to derive the search direction in the gradient descent algorithm, one should require that the action of $J_s$ on the increment $\Delta s$ be unchanged after this projection. We will for now project $s$ and $a$ into $W_2^1$; we should have

$$\int_0^T \left[\tilde{J}_s(t)\Delta s(t) + \delta_T(t)\tilde{\tilde{J}}\Delta s(T)\right]\, dt = \int_0^T \left[J_s(t)\Delta s(t) + J_s'(t)\Delta s'(t)\right]\, dt$$
$$= \int_0^T \left[J_s(t)\Delta s(t) - J_s''(t)\Delta s(t) + J_s'(T)\Delta s(T)\delta_T(t) - J_s'(0)\Delta s(0)\delta_0(t)\right]\, dt$$

and hence

$$0 = \int_0^T \left[\left(J_s(t) - \tilde{J}_s(t)\right) - J_s''(t)\right]\Delta s(t)\, dt + \left(J_s'(T) - \tilde{\tilde{J}}\right)\Delta s(T) - J_s'(0)\Delta s(0)$$

By the arbitraricity of $\Delta s$, it follows that

$$J_s(t) - J_s''(t) = \tilde{J}_s(t)$$
$$J_s'(T) = \tilde{\tilde{J}}, \quad J_s'(0) = 0$$

In practice, we may choose to penalize large gradient values by including a parameter $\ell > 0$ in the definition of the $W_2^1$ norm, which requires the solution of the alternative equation

$$\int_0^T \left[\tilde{J}_s(t)\Delta s(t) + \delta_T(t)\tilde{\tilde{J}}\Delta s(T)\right]\, dt = \int_0^T \left[J_s(t)\Delta s(t) + \ell^2 J_s'(t)\Delta s'(t)\right]\, dt$$

which implies $J_s$ should satisfy

$$\begin{cases} J_s(t) - \ell^2 J_s''(t) = \tilde{J}_s(t) \\ J_s'(T) = \tilde{\tilde{J}}, \quad J_s'(0) = 0 \end{cases}$$

Note that the formulation used to approximate the Reisz element in [?] would read

$$\begin{cases} J_s(t) - \ell^2 J_s''(t) = \tilde{J}_s(t) + \tilde{\tilde{J}} \\ J_s'(T) = 0, \quad J_s'(0) = 0 \end{cases}$$

in our notation, which will give significantly different results whenever $\tilde{\tilde{J}} \neq 0$. Even this formulation does not take into account one condition from (??); for a fixed control $a(t)$, given any two controls $s$, $\tilde{s}$ should satisfy $s(0) = s_0 = \tilde{s}(0)$, so $\Delta s(0) = 0$. Further, both satisfy the compatibility condition and hence

$$\Delta s'(0) = \tilde{s}'(0) - s'(0) = 0$$

Since $\Delta s(0)$ is not actually arbitrary, we are ***not***, in general, required to that $J_s'(0) = 0$. Given that $s(0) = s_0$ is fixed, it is justified to instead solve

:gradient-s-precond
$$\begin{cases} J_s(t) - \ell^2 J_s''(t) = \tilde{J}_s(t) \\ J_s'(T) = \tilde{\tilde{J}}, \quad J_s(0) = 0 \end{cases} \tag{30}$$

Making a similar calculation for $a(t)$, we see that the relationship between the preconditioned $(H_1)$ gradient $J_a$ and the original $(L_2)$ gradient $\tilde{J}_a$ should be

$$\begin{cases} J_a(t) - \ell^2 J_a''(t) = \tilde{J}_a(t) \\ J_a'(T) = 0, \quad J_a(0) = 0 \end{cases}$$

As in the previous case, the Dirichlet condition at $x = 0$ is justified by the lack of arbitrarity in $a(0)$ due to (??).

Note that in order to solve these problems numerically, they are reformulated as a first order system; for example letting $y_1 = J_s(t)$, $y_2 = y_1'(t)$, eq. (??) becomes

$$\begin{cases} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}' = \begin{bmatrix} y_2 \\ \frac{y_1 - \tilde{J}_s(t)}{\ell^2} \end{bmatrix} \\ y_2(T) = \tilde{J} \\ y_1(0) = 0 \end{cases} \tag{31}$$

This is the formulation of the preconditioning problem that is currently output by our preconditioning code `precond.m`, included below in Section ??; the original problem is selected by choosing `mode` to be 1, and the updated problem is selected by choosing `mode` to be 2.

## 4. Model Problem #1

`sec:model-problem-1`

4.1. **Problem Setup and Generation of Test Data**. As a model problem, we will take (??)–(??) with $\beta_i \equiv 1$, $b \equiv 1$, $\chi \equiv 0$, $\gamma \equiv 1$, and $f \equiv 0$:

$$Lu \equiv (a(t)u_x)_x - u_t = 0, \text{ in } \Omega \tag{32}$$

$$u(x,0) = \phi(x), \ 0 \le x \le s(0) = s_0 \tag{33}$$

$$a(t)u_x(0,t) = g(t), \ 0 \le t \le T \tag{34}$$

$$a(t)u_x(s(t),t) + s'(t) = 0, \ 0 \le t \le T \tag{35}$$

In this case, the adjoint problem (??)–(??) then takes the form

$$L^*\psi := \big(a\psi_x\big)_x + \psi_t = 0, \quad \text{in } \Omega \tag{36}$$

$$\psi(x,T) = 2(u(x,T) - w(x)), \ 0 \le x \le s(T) \tag{37}$$

$$a(t)\psi_x(0,t) = 0, \ 0 \le t \le T \tag{38}$$

$$\Big[a\psi_x - s'\psi - 2(u-\mu)\Big]_{x=s(t)} = 0, \ 0 \le t \le T \tag{39}$$

and the gradient is

$$\begin{aligned} \langle \mathcal{J}'(v), \Delta v \rangle_H == &\int_0^T \Big[2\big(u-\mu\big)u_x - \psi a u_{xx} + \psi_x s'(t) + \psi_t\Big]_{x=s(t)} \Delta s(t)\, dt \\ &+ \big[\big[u-w\big]\big[u-w-2\big] + 2(s(T) - s_*)\big]\,|_{(x,t)=(s(T),T)} \Delta s(T) \\ &+ \int_0^T \Delta a \left[\int_0^{s(t)} u_{xx}\psi\, dx - [u_x\psi]_{x=0} - [u_x\psi]_{x=s(t)}\right] dt \end{aligned}$$

To create synthetic problem data to test our algorithm, we introduce a time grid `tmesh` and evaluate the functions $a(t)$ and $s(t)$ on the given time grid to form vectors `svals` and `avals`. With all of the other data known, the forward problem can be solved to find an estimate for $u(x,t)$. The traces $u(s(t),t)$ and $u(x,T)$ are then set to $\mu$ and $w$ within the problem initialization routine, and the rest of the problem data is discarded before starting the gradient iteration process. In particular, the analytic solution is not subsequently used when considering the problem with synthetic data.

As an initial approach for $s(t)$ and $a(t)$, and given the analytic solutions $s_{\text{true}}(t)$, $a_{\text{true}}(t)$, we define

$$s_{\text{linear}}(t) = s_0 + \frac{t}{T}\left(s_{\text{true}}(T) - s_0\right) \tag{40}$$

where it is assumed that a measurement (or an approximation) of the free boundary is available at the final moment. For any fixed parameter $\lambda_s$ we set

$$s_{\text{initial}}(t) = \lambda_s s_{\text{linear}}(t) + \big(1 - \lambda_s\big) s_{\text{true}}(t) \tag{41}$$

Since the gradient descent method is local in nature, this choice of the initial approach allows the empirical determination of the convergence region. Since our example has a constant "true" function $a(t)$, we give a quadratic perturbation to this function in order to deduce similar information for this control; hence, for any $\lambda_a$ we define

$$a_{\text{initial}}(t) = a_{\text{true}} + \frac{4}{T}\lambda_a t(T-t) \tag{42}$$

Note that

$$\max_t |a_{\text{initial}} - a_{\text{true}}| = \frac{4}{T}|\lambda_a| t(T-t)|_{t=T/2} = |\lambda_a|$$

In particular, $\lambda_a$ should be bounded from below in order to ensure the ellipticity condition $a(t) \geq a_0$ is preserved, and $\lambda_s$ cannot be chosen arbitrarily, since we require $s(t) \geq \delta > 0$.

The process of deriving synthetic data and choosing the initial approach along with any other parameters is encapsulated in the function `initial_setup`, included below in Section **??**.

4.2. **Tikhonov & Samarskii Example**. A classical model problem in the form considered can be found in the text by Tikhonov & Samarskii [**?**, Ch. II, App. IV, pp. 283–288]. The problem originates in the following: find $\big(u, \xi\big)$ satisfying

`nov-samarskii-pde-1`
$$\frac{\partial u_1}{\partial t} - k_1 \frac{\partial^2 u_1}{\partial x^2} = 0,\ 0 < x < \xi(t),\ t > 0 \tag{43}$$

`nov-samarskii-pde-2`
$$\frac{\partial u_2}{\partial t} - k_2 \frac{\partial^2 u_2}{\partial x^2} = 0,\ \xi(t) < x < \infty,\ t > 0 \tag{44}$$

$$u_1(0,t) = c_1,\ t > 0 \tag{45}$$

$$u_2(x,0) = c_2,\ x > \xi(0) \tag{46}$$

$$u_1(x,0) = c_1,\ x < \xi(0) \tag{47}$$

$$u_1(\xi(t),t) = u_2(\xi(t),t) = 0,\ t > 0 \tag{48}$$

`amarskii-stefancond`
$$k_1 \frac{\partial u_1}{\partial x}(\xi(t),t) - k_2 \frac{\partial u_2}{\partial x}(\xi(t),t) = \gamma \frac{d\xi}{dt}(t) \tag{49}$$

where $k_1, k_2, \gamma > 0$ are given, and without loss of generality $c_1 < 0$, $c_2 \geq 0$. Direct calculation verifies that the exact solution to (**??**)–(**??**) is

$$u_1(x,t) = c_1 + B_1 \text{erf}\left(\frac{x}{\sqrt{4k_1 t}}\right) \tag{50}$$

$$u_2(x,t) = A_2 + B_2 \text{erf}\left(\frac{x}{\sqrt{4k_2 t}}\right) \tag{51}$$

$$\xi(t) = \alpha\sqrt{t} \tag{52}$$

where

$$\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-z^2}\, dz, \tag{53}$$

$$B_1 = -\frac{c_1}{\operatorname{erf}\left(\alpha/\sqrt{4k_1}\right)}, \quad B_2 = \frac{c_2}{1 - \operatorname{erf}\left(\alpha/\sqrt{4k_2}\right)} \tag{54}$$

$$A_2 = -\operatorname{erf}\left(\alpha/\sqrt{4k_2}\right) B_2 \tag{55}$$

and $\alpha$ is a solution of the transcendental equation

$$\frac{\sqrt{k_1} c_1 e^{-\frac{\alpha^2}{4k_1}}}{\operatorname{erf}\left(\alpha/\sqrt{4k_1}\right)} + \frac{\sqrt{k_2} c_2 e^{-\frac{\alpha^2}{4k_2}}}{\left[1 - \operatorname{erf}\left(\alpha/\sqrt{4k_2}\right)\right]} = -\frac{\gamma\sqrt{\pi}}{2}\alpha \tag{56}$$

The equation above has a unique solution $\alpha > 0$.

To write the problem as a one-phase Stefan problem, we simply set $c_2 = 0$, so $u_2 \equiv 0$, and the function $u = u_1$ satisfies

$$\frac{\partial u}{\partial t} - k_1 \frac{\partial^2 u}{\partial x^2} = 0,\ 0 < x < \xi(t),\ t > 0 \tag{57}$$

$$u(0,t) = c_1,\ t > 0 \tag{58}$$

$$u(x,0) = c_1,\ x < \xi(0) \tag{59}$$

$$u(\xi(t),t) = 0,\ t > 0 \tag{60}$$

$$k_1 \frac{\partial u}{\partial x}(\xi(t),t) = \gamma \frac{d\xi}{dt}(t) \tag{61}$$

where $k_1, k_2, \gamma > 0$ are given. The analytic solution is

$$u(x,t) = c_1 + B_1 \operatorname{erf}\left(\frac{x}{\sqrt{4k_1 t}}\right) \tag{62}$$

$$\xi(t) = \alpha\sqrt{t} \tag{63}$$

where

$$\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-z^2}\, dz, \tag{64}$$

$$B_1 = -\frac{c_1}{\operatorname{erf}\left(\alpha/\sqrt{4k_1}\right)}, \tag{65}$$

and $\alpha$ is a solution of the transcendental equation

$$\frac{\sqrt{k_1} c_1 e^{-\frac{\alpha^2}{4k_1}}}{\operatorname{erf}\left(\alpha/\sqrt{4k_1}\right)} = -\frac{\gamma\sqrt{\pi}}{2}\alpha \tag{66}$$

We calculate

$$u_x(x,t) = \frac{B_1}{\sqrt{\pi k_1 t}} \exp\left(-\frac{x^2}{4k_1 t}\right) \tag{67}$$

so

$$k_1 u_x(s(t),t) = B_1 \frac{k_1}{\sqrt{\pi k_1 t}} \exp\left(-\frac{\alpha^2}{4k_1}\right) \tag{68}$$

$$= \frac{c_1\sqrt{k_1}\exp\left(-\frac{\alpha^2}{4k_1}\right)}{\operatorname{erf}\left(\alpha/\sqrt{4k_1}\right)}\left(\frac{-1}{\sqrt{\pi t}}\right) \tag{69}$$

By virtue of $\alpha$ satisfying the above transcendental equation and $\frac{d\xi}{dt} \equiv \frac{\alpha}{2\sqrt{t}}$, it follows that

$$u_x(s(t),t) = \gamma\frac{\sqrt{\pi}}{2}\alpha\left(\frac{1}{\sqrt{\pi t}}\right) \tag{70}$$

$$= \gamma\frac{\alpha}{2\sqrt{t}} = \gamma\frac{d\xi}{dt} \tag{71}$$

That is, the Neumann boundary condition is satisfied at $x = s(t)$. Similarly,

$$k_1 u_x(0,t) = \frac{k_1 B_1}{\sqrt{\pi k_1 t}} = \frac{\sqrt{k_1}B_1}{\sqrt{\pi t}}$$

is the condition on the fixed boundary.

Note that the domain degenerates at $t = 0$ (and the problem data becomes singular), which is not supported by our theoretical framework; therefore, the data is shifted in time by a fixed amount `tShift=`: $\delta$ with $0 < \delta \ll 1$. We also give only the data necessary to solve the Neumann problem, and choose $k_1 \equiv 1$; the solution may still depend on $c_1 < 0$ and the latent heat $\gamma$, which is denoted by `latentHeat` in the code. Tor any trial boundary curve $\tilde{\xi}$ satisfying $\tilde{\xi}(0) = \xi(\delta)$, the problem formulated is

$$\frac{\partial u}{\partial t} - \frac{\partial^2 u}{\partial x^2} = 0,\ 0 < x < \tilde{\xi}(t),\ t > 0 \tag{72}$$

$$u_x(0,t) = \frac{B_1}{\sqrt{\pi(t+\delta)}},\ t > 0 \tag{73}$$

$$\frac{\partial u}{\partial x}(\tilde{\xi}(t),t) = \gamma\tilde{\xi}'(t) \tag{74}$$

$$u(x,0) = c_1 + B_1 \operatorname{erf}\left(\frac{x}{2\sqrt{\delta}}\right),\ x < \tilde{\xi}(0) \tag{75}$$

The `fzero` rootfinder is used to solve the necessary transcendental equation to find $\alpha$ (denoted by `boundary_constant` in the code.) This solution is implemented in `true_solution`, included in Section **??**.

The test code in `test_true_solution`, included in Section **??**, simply verified that the output shape and size of the parameters is correct.

## Code Listings

### Appendix A. pdeSolver

`sec:code-listing-pdeSolver`

### Appendix B. Forward

`sec:code-listing-forward`

### Appendix C. test_Forward

`sec:code-listing-test-forward`

Department of Mathematical Sciences, Florida Institute of Technology, Melbourne, FL 32901