5-6-2016

On Identification of Nonlinear Parameters in PDEs

Raphael Kahler
rak9698@rit.edu

On Identification of Nonlinear Parameters in PDEs

by

Raphael Kahler

A Thesis Submitted in Partial Fulfillment for the
Degree of Master of Science in Applied and Computational Mathematics

School of Mathematical Sciences
College of Science

Rochester Institute of Technology
Rochester, NY
May 6, 2016

Committee Approval:

Baasansuren Jadamba — Date
School of Mathematical Sciences
Thesis Advisor

Akhtar A. Khan — Date
School of Mathematical Sciences
Thesis Co-advisor

Patricia Clark — Date
School of Mathematical Sciences
Committee Member

Joshua Faber — Date
School of Mathematical Sciences
Committee Member

Elizabeth Cherry — Date
School of Mathematical Sciences
Director of Graduate Programs

Abstract

Inverse problems have been studied in great detail and optimization methods using objective functionals such as output least-squares (OLS) and modified output least-squares (MOLS) are well understood. However, the existing literature has only dealt with identifying parameters that appear linearly in systems of partial differential equations. We investigate the changes that occur in the identification process if the parameter appears nonlinearly. We extend the OLS and MOLS functionals to this nonlinear case and give first and second derivative formulas. We further show that the typical convexity of the MOLS functional can not be guaranteed when identifying nonlinear parameters. To numerically verify our findings we employ a C++ based computational framework. Discretization is done via the finite element method, and details are given for the new results of the functionals and their derivatives. Since we consider nonlinear parameters, gradient methods such as adjoint stiffness are not applicable to the OLS functional and we instead show computation methods using the adjoint approach.

Contents

Chapter 1

The Lost Convexity of the MOLS Functional

We begin by taking a look at the modified output-least squares (MOLS) functional that emerged as an alternative to the generally non-convex output least-squares (OLS) functional. The MOLS has the desirable property of being a convex functional, which was shown in [1]. However, the convexity of the MOLS has only been established for parameters appearing linearly in the PDEs. The primary objective of this chapter is to introduce and analyze a variant of the MOLS for the inverse problem of identifying parameters that appear nonlinearly in general variational problems. We are interested in understanding what geometric properties of the original MOLS can be retained for the nonlinear case. Besides giving an existence result for the optimization formulation of the inverse problem, we give a thorough derivation of the first-order and second-order derivative formulas for the new objective functional. The derivative formulae suggest that the convexity of the MOLS cannot be retained for the parameters appearing nonlinearly without imposing additional assumptions on the data involved.

1.1 Introduction

Applied models frequently lead to partial differential equations involving parameters attributed to physical characteristics of the model. The direct problem in this setting is to solve the partial differential equation. By contrast, an inverse problem seeks for the identification of the parameters when a certain measurement of a

solution of the partial differential equation is available.

For clarification, consider the following elliptic boundary value problem (BVP)

$$-\nabla \cdot (q\nabla u) = f \ \text{ in } \Omega, \ \ u = 0 \ \text{ on } \partial\Omega,$$

where Ω is a sufficiently smooth domain in $\mathbb{R}^2$ or $\mathbb{R}^3$ and $\partial\Omega$ is its boundary. The above BVP models interesting real-world problems and has been studied in great detail. For instance, here $u = u(x)$ may represent the steady-state temperature at a given point x of a body; then q would be a variable thermal conductivity coefficient, and f the external heat source. This system also models underground steady state aquifers in which the parameter q is the aquifer transmissivity coefficient, u is the hydraulic head, and f is the recharge. The inverse problem in the context of the above BVP is to estimate the coefficient q from a measurement z of the solution u. This inverse problem has been the subject of numerous papers, see [2, 3, 4]. Various other inverse problems occur with complicated boundary problems and with diverse applications, see [5, 6, 7, 8, 9, 10, 11, 12, 13].

It is convenient to investigate the inverse problem of parameter identification in an abstract setting allowing for more general PDEs. Let B be a Banach space and let A be a nonempty, closed, and convex subset of B. Given a Hilbert space V, let $T : B \times V \times V \to \mathbb{R}$ be a trilinear form with $T(a, u, v)$ that is symmetric in u and v, and let m be a bounded linear functional on V. Assume there are constants $\alpha > 0$ and $\beta > 0$ such that for all $u, v \in V$ the following conditions hold:

$$T(a, u, v) \leq \beta \|a\|_B \|u\|_V \|v\|_V , \qquad \forall\, a \in B, \tag{1.1}$$

$$T(a, u, u) \geq \alpha \|u\|_V^2 , \qquad \forall\, a \in A \tag{1.2}$$

Consider the following variational problem: Given $a \in A$, find $u = u(a) \in V$ such that

$$T(a, u, v) = m(v), \ \ \text{for every } v \in V. \tag{1.3}$$

Due to the imposed conditions, it follows from the Riesz representation theorem that for every $a \in A$, the variational problem (1.3) admits a unique solution $u(a)$. In this abstract setting, the inverse problem of identifying parameter now seeks a in (1.3) from a measurement z of u.

A common approach for solving the inverse problem is to pose it as a minimization problem through the output least-squares (OLS) functional given by

$$\widehat{J}(a) = \frac{1}{2}\|u(a) - z\|_Z^2, \tag{1.4}$$

where $\|\cdot\|_Z$ is the norm in a suitable observation space, z is the data (the measurement of u) and $u(a)$ solves the variational problem (1.3).

The functional (1.4) has a serious deficiency of being non-convex for nonlinear inverse problems. To circumvent this drawback, in [1], the following modified OLS functional (MOLS) was introduced

$$J(a) = \frac{1}{2}T(a, u - z, u - z) \tag{1.5}$$

where z is the data (the measurement of u) and $u(a)$ solves (1.3). In [1], the author established that (1.5) is convex and used it to estimate the Lamé moduli in the equations of isotropic elasticity. Studies related to MOLS functional and its extensions can be found in [14, 15, 16, 17].

The first observation necessary for the convexity of the MOLS is that for each a in the interior of A, the first derivative $\delta a = Du(a)\delta a$ is the unique solution of the variational equation (see [1]):

$$T(a, \delta u, v) = -T(\delta a, u, v), \quad \text{for every } v \in V, \tag{1.6}$$

The authors in [1] proved the following derivative formulae:

$$DJ(a)\delta a = -\frac{1}{2}T(\delta a, u(a) + z, u(a) - z), \tag{1.7}$$

$$D^2J(a)(\delta a, \delta a) = T(a, Du(a)\delta a, Du(a)\delta a). \tag{1.8}$$

Due to the coercivity (1.2) of the trilinear form, the above formula for the second-order derivative ensures that $D^2J(a)(\delta a, \delta a) \geq \alpha\|Du(a)\delta a\|_V^2$, and hence the convexity of the modified output least-squares functional in the interior of A follows.

A careful look at the proof of the above mentioned results reveal that for the convexity of the MOLS, it is essential that the first argument of T be the parameter to be identified. On the other hand, interesting applications lead to cases when the first argument of T in fact contains a nonlinear function of the sought parameter (see [18]). The objective of this chapter is to introduce and analyze a variant of the MOLS for the inverse problem of identifying parameter that appears nonlinearly in general variational problems. We are interested in understanding what geometric properties of the MOLS can be retained for such a case.

1.2 Main Results

Let $S \subset X$ be an open and convex subset of X. We say that a map $f : S \subset X \to Y$ is directionally differentiable at $x \in S$ in a direction $\delta x \in X$ if the following limit exists

$$f'(x; \delta x) = \lim_{t \downarrow 0} \frac{f(x + t\delta x) - f(x)}{t}.$$

The map f is said to be directionally differentiable at x if f is directionally differentiable at $x \in S$ in every direction $u \in X$. Given that f is directionally differentiable, the second-order directional derivative along

directions $(\delta x_1, \delta x_2) \in X \times X$ is given by the following expression, provided that the limit exists:

$$f''(x; \delta x_1, \delta x_2) = \lim_{t \downarrow 0} \frac{f'(x + t\delta x_2; \delta x_1) - f'(x; \delta x_1)}{t}$$

Note that $Df(x)(\delta x) = f'(x; \delta x)$, $D^2 f(x)(\delta x_1, \delta x_2) = f''(x; \delta x_1, \delta x_2)$ if f is differentiable, twice differentiable at x, respectively.

Given an open S open and convex subset of B, let $g : S \subset B \to B$ be a map such that range$(g) \subset A$, that is, $g(x) \in A$ for every $x \in \text{dom}(g)$. Moreover, assume that g is continuous and directionally differentiable. Given $a \in A$, consider the variational problem of finding $u(a) \in V$ such that

$$T(g(a), u(a), v) = m(v), \quad \text{for every } v \in V. \tag{1.9}$$

Our objective is to identify the variable parameter a from a measurement z of u. Of course, a natural idea is to define $b(x) = g(a(x))$ and consider the problem of identifying $b(x)$ in (1.9) and then solve the nonlinear equation $b(x) = g(a(x))$ to find a. Clearly, such an approach is, at best, heuristic and fails to give any insight into the geometric properties of the associated MOLS.

We give the following continuity result for its later use.

Lemma 1.2.1. *The following bounds are all valid:*

$$\begin{aligned}
\|u(a) - u(b)\|_V &\leq \frac{\beta}{\alpha} \|u(a)\|_V \|g(b) - g(a)\|_B, \\
\|u(a) - u(b)\|_V &\leq \frac{\beta}{\alpha} \|u(b)\|_V \|g(b) - g(a)\|_B, \\
\|u(a) - u(b)\|_V &\leq \frac{\beta}{\alpha^2} \|m\|_{V^*} \|g(b) - g(a)\|_B.
\end{aligned}$$

Proof. For every $v \in V$, we have $T(g(a), u(a), v) = m(v)$ and $T(g(b), u(b), v) = m(v)$ implying that $T(g(a), u(a), v) - T(g(b), u(b), v) = 0$ or

$$T(g(a), u(a) - u(b), u(a) - u(b)) = -T(g(a) - g(b), u(b), u(a) - u(b)).$$

Using (1.1) and (1.2), we get

$$\|u(a) - u(b)\|_V \leq \frac{\beta}{\alpha} \|u(b)\|_V \|g(a) - g(b)\|_B,$$

proving the second of the desired bounds; the first is obtained by interchanging the roles of a and b. The fact $\|u(b)\|_V \leq \alpha^{-1} \|m\|_{V^*}$ which is easy to prove, yields the third bound. □

We introduce the following new modified output least squares

$$J(a) = \frac{1}{2} T(g(a), u - z, u - z), \tag{1.10}$$

where z is the data (the measurement of u) and $u(a)$ solves (1.9).

We can now formulate the inverse problem as an optimization problem using (1.10). However, due to the known ill-posedness of inverse problems, we need some kind of regularization for developing a stable computational framework. Therefore, instead of (1.10), we will use its regularized analogue and consider the following regularized optimization problem: Find $a \in A$ by solving

$$\min_{a\in A} J_\kappa(a) := \frac{1}{2}T(g(a), u(a) - z, u(a) - z) + \kappa R(a), \tag{1.11}$$

where, given a Hilbert space H, $R : H \to \mathbb{R}$ is a regularizer, $\kappa > 0$ is a regularization parameter, $u(a)$ is the unique solution of (1.9) that corresponds to the coefficient a, and z is the measured data.

In the following, we give an existence result for the regularized optimization problem (1.11).

Theorem 1.2.1. *Assume that the Hilbert space H is compactly embedded into the space B, $A \subset H$ is nonempty, closed, and convex, the map R is convex, lower-semicontinuous and there exists $\alpha > 0$ such that $R(\ell) \geq \alpha\|\ell\|_H^2$, for every $\ell \in A$. Then the optimization problem* (1.11) *has a nonempty solution set.*

Proof. Since $J_\kappa(a)$ is nonnegative for all $a \in A$, there exists a minimizing sequence $\{a_n\}$ in A such that $\lim_{n\to\infty} J_\kappa(a_n) = \inf\{J_\kappa(a)|\, a \in A\}$. Therefore, $\{J_\kappa(a_n)\}$ is bounded from above, and by the definition of J_κ, there exists a constant $c > 0$ such that $R(a_n) \leq c$ which implies that $\{a_n\}$ is bounded in H. Due to the compact embedding of H into B, there exists a subsequence converging strongly in B. By keeping the same notations for subsequences as well, we assume that a_n converges strongly some $\bar{a} \in A$. Moreover, due to the continuity of g, we have $g(a_n) \to g(\bar{a})$. By the definition of u_n, for every $v \in V$, we have $T(g(a_n), u_n, v) = m(v)$, which for $v = u_n$ yields $T(g(a_n), u_n, u_n) = m(u_n)$. Using (1.2), we get

$$\alpha\|u_n\|_V^2 \leq \|m\|_{V^*}\|u_n\|_V$$

which ensures the boundedness of $u_n = u(a_n)$. Therefore, there exists a subsequence of $\{u_n\}$ that converges weakly to some $\bar{u} \in V$. We claim that $\bar{u} = \bar{u}(\bar{a})$. By the definition of u_n, for every $v \in V$, we have $T(g(a_n), u_n, v) = m(v)$. This, after a simple rearrangements of terms, implies that $T(g(\bar{a}), \bar{u}, v) - m(v) = -T(g(a_n) - g(\bar{a}), u_n, v) - T(g(\bar{a}), u_n - u, v)$, which, when passed to the limit $n \to \infty$, implies that $T(g(\bar{a}), \bar{u}, v) = m(v)$ as all the terms on the right-hand side go to zero. Since $v \in V$ is arbitrary and since (1.3) is uniquely solvable, we deduce that $\bar{u} = \bar{u}(\bar{a})$.

We claim that $J(a_n) \to J(\bar{a})$. The identities $T(g(a_n), u_n - z, u_n - z) = m(u_n - z) - T(g(a_n), z, u_n - z)$ and $T(g(\bar{a}), \bar{u} - z, \bar{u} - z) = m(\bar{u} - z) - T(g(\bar{a}), z, \bar{u} - z)$, in view of the rearrangement

$$T(g(a_n), z, u_n - z) - T(g(\bar{a}), z, \bar{u} - z) = T(g(a_n) - g(\bar{a}), z, u_n - z) - T(g(\bar{a}), z, u_n - \bar{u}),$$

ensure that $T(g(a_n), u_n - z, u_n - z) \to T(g(\bar{a}), \bar{u} - z, \bar{u} - z)$, and consequently,

$$\begin{aligned} J_\kappa(\bar{a}) &= T(g(\bar{a}), \bar{u} - z, \bar{u} - z) + \kappa R(\bar{a}) \\ &\leq \lim_{n\to\infty} T(g(a_n), u(a_n) - z, u(a_n) - z) + \liminf_{n\to\infty} \kappa R(a_n) \\ &\leq \liminf_{n\to\infty} \{T(g(a_n), u(a_n) - z, u(a_n) - z) + \kappa R(a_n)\} = \inf \{J_\kappa(a) \,:\, a \in A\}, \end{aligned}$$

confirming that $\bar{a}$ is a solution of (1.11). The proof is complete. □

In the following, we assume that g and the parameter-to-solution map $u : a \to u(a)$ are directionally differentiable at $a \in A$. Recall that u is directionally differentiable if the following limit exists:

$$u'(a, b) := \lim_{t\to 0^+} \frac{u(a + tb) - u(a)}{t}.$$

The following theorem gives a derivative characterization.

Theorem 1.2.2. *The parameter-to-solution map $u : A \subset B \to V$ is directionally differentiable. Moreover, given any $a \in A$ for each direction $\delta a \in B$ the directional derivative $u'(a; \delta a)$ is the unique solution of the following variational equation*

$$T\left(g(a), u'(a; \delta a), v\right) = -T\left(g'(a; \delta a), u(a), v\right), \quad \textit{for every } v \in V. \tag{1.12}$$

Furthermore, if g is differentiable, then u is differentiable.

Proof. The unique solvability of the variational equation is a direct consequence of the Lax-Milgram lemma. Let $a \in A$, $\delta a \in B$ be arbitrary. For any $t > 0$ and for every $v \in V$, we have

$$\begin{aligned} T(g(a + t\delta a), u(a + t\delta a), v) &= m(v), \\ T(g(a), u(a), v) &= m(v). \end{aligned}$$

The above equations after a simple rearrangement of terms, yield

$$\begin{aligned} 0 &= \frac{1}{t}\left[T(g(a + t\delta a), u(a + t\delta a), v) - T(g(a), u(a), v)\right] \\ &= \frac{1}{t}\left[T(g(a + t\delta a), u(a + t\delta a), v) - T(g(a), u(a + t\delta a), v)\right] \\ &\quad + \frac{1}{t}\left[T(g(a), u(a + t\delta a), v) - T(g(a), u(a), v)\right] \\ &= T\left(\frac{g(a + t\delta a) - g(a)}{t}, u(a + t\delta a), v\right) + T\left(g(a), \frac{u(a + t\delta a) - u(a)}{t}, v\right), \end{aligned}$$

and therefore, for an arbitrary $v \in V$, we have

$$T\left(g(a), \frac{u(a + t\delta a) - u(a)}{t}, v\right) = -T\left(\frac{g(a + t\delta a) - g(a)}{t}, u(a + t\delta a), v\right). \tag{1.13}$$

Let $w \in V$ be the unique solution to the following variational equation

$$T\left(g(a), w, v\right) = -T\left(g'(a;\delta a), u(a), v\right), \quad \text{for every } v \in V. \tag{1.14}$$

Clearly, the element w is well defined. Furthermore, combining (1.13) and (1.14), we have

$$\begin{aligned} T\left(g(a), \delta u_t - w, v\right) = &-T\left(\frac{g(a+t\delta a) - g(a)}{t} - g'(a;\delta a), u(a+t\delta a), v\right) \\ &- T\left(g'(a;\delta a), u(a+t\delta a) - u(a), v\right) \end{aligned} \tag{1.15}$$

where we are denoting $\delta u_t = t^{-1}(u(a+t\delta a) - u(a))$. Taking $v = \delta u_t - w$ in this expression, we get

$$\begin{aligned} \alpha \left\|\delta u_t - w\right\|_V^2 &\leq T\left(g(a), \delta u_t - w, \delta u_t - w\right) \\ &= -T\left(\frac{g(a+t\delta a) - g(a)}{t} - g'(a;\delta a), u(a+t\delta a), \delta u_t - w\right) \\ &\quad - T\left(g'(a;\delta a), u(a+t\delta a) - u(a), \delta u_t - w\right) \end{aligned}$$

and hence

$$\alpha \left\|\delta u_t - w\right\|_V \leq \left\|\frac{g(a+t\delta a) - g(a)}{t} - g'(a;\delta a)\right\|_B \left\|u(a+t\delta a)\right\|_V + \left\|g'(a;\delta a)\right\|_B \left\|u(a+t\delta a) - u(a)\right\|_V.$$

By taking the limit $t \to 0$, the right-hand side tends to zero since u is continuous and g directionally differentiable. We get

$$\left\|\delta u_t - w\right\|_V \to 0,$$

which implies that

$$t^{-1}(u(a+t\delta a) - u(a)) \to w \text{ in } V,$$

and hence u is directionally differentiable at a in the direction δa with $u'(a;\delta a) = w$.

To prove the differentiability, we first take a fixed $a \in A$. Define the linear operator $T : B \to V$ such that for every $\delta a \in B$, $T(\delta)$ gives the unique solution to the following variational equation:

$$T\left(g(a), T(\delta a), v\right) = -T\left(Dg(a)(\delta a), u(a), v\right), \quad \text{for every } v \in V.$$

Since $-T\left(g'(a;\delta a), u(a), \cdot\right) \in V^*$, T is well defined by the Riesz representation theorem. Furthermore, T is bounded by applying the basic properties of the trilinear form:

$$\begin{aligned} \alpha \left\|T(\delta a)\right\|_V^2 &\leq T\left(g(a), T(\delta a), T(\delta a)\right) \\ &= -T\left(Dg(a)(\delta a), u(a), T(\delta a)\right) \\ &\leq \beta \left\|Dg(a)(\delta a)\right\|_B \left\|T(\delta a)\right\|_V \left\|u(a)\right\|_V \end{aligned}$$

Since g is differentiable $\|Dg(a)(\delta a)\|_B \leq C \|\delta a\|_B$, and hence

$$\|T(\delta a)\|_V \leq (\beta C \|u(a)\|_V) \|u(a)\|_V .$$

On the other hand, following the previous calculation, we have

$$T\left(g(a), \frac{u(a+\delta a)-u(a)}{\|\delta a\|_B}, v\right) = -T\left(\frac{g(a+\delta a)-g(a)}{\|\delta a\|_B}, u(a+\delta a), v\right),$$

and therefore

$$\begin{aligned} T\left(g(a), \frac{u(a+\delta a)-u(a)}{\|\delta a\|_B}, v\right) - T\left(g(a), T\left(\frac{\delta a}{\|\delta a\|_B}\right), v\right) = & - T\left(\frac{g(a+\delta a)-g(a)}{\|\delta a\|_B}, u(a+\delta a), v\right) \\ & - T\left(Dg(a)\left(\frac{\delta a}{\|\delta a\|_B}\right), u(a), v\right), \end{aligned}$$

or equivalently,

$$\begin{aligned} T\left(g(a), \frac{u(a+\delta a)-u(a)-T(\delta a)}{\|\delta a\|_B}, v\right) = & - T\left(\frac{g(a+\delta a)-g(a)-Dg(a)(\delta a)}{\|\delta a\|_B}, u(a+\delta a), v\right) \\ & + T\left(Dg(a)\left(\frac{\delta a}{\|\delta a\|_B}\right), u(a+\delta a)-u(a), v\right). \end{aligned}$$

If we denote $\Delta u = \|\delta a\|_B^{-1}(u(a+\delta a)-u(a)-T(\delta a))$, then by following the same reasoning as for the previous case, we have

$$\begin{aligned} \alpha \|\Delta u\|_V^2 &\leq T(g(a), \Delta u, \Delta u) \\ &= - T\left(\frac{g(a+\delta a)-g(a)-Dg(a)(\delta a)}{\|\delta a\|_B}, u(a+\delta a), \Delta u\right) \\ &\quad + T\left(Dg(a)\left(\frac{\delta a}{\|\delta a\|_B}\right), u(a+\delta a)-u(a), \Delta u\right), \end{aligned}$$

implying

$$\begin{aligned} \alpha \|\Delta u\|_V^2 \leq & \left(\frac{\|g(a+\delta a)-g(a)-Dg(a)(\delta a)\|_B}{\|\delta a\|_B} \|u(a+\delta a)\|_V \right. \\ & \left. + \left\|Dg(a)\left(\frac{\delta a}{\|\delta a\|_B}\right)\right\|_B \|u(a+\delta a)-u(a)\|_V\right) \|\Delta u\|_V \end{aligned}$$

and hence

$$\begin{aligned} \alpha \|\Delta u\|_V \leq & \left(\frac{\|g(a+\delta a)-g(a)-Dg(a)(\delta a)\|_B}{\|\delta a\|_B} \|u(a+\delta a)\|_V \right. \\ & \left. + \left\|Dg(a)\left(\frac{\delta a}{\|\delta a\|_B}\right)\right\|_B \|u(a+\delta a)-u(a)\|_V\right). \end{aligned}$$

Since u is continuous by hypothesis and g is differentiable, we have

$$\frac{\|g(a+\delta a)-g(a)-Dg(a)\,(\delta a)\|_B}{\|\delta a\|_B}\,\|u(a+\delta a)\|_V+\left\|Dg(a)\left(\frac{\delta a}{\|\delta a\|_B}\right)\right\|_B\|u(a+\delta a)-u(a)\|_V\to 0,$$

for $\|\delta a\|_B \to 0$. Finally, for $\|\delta a\|_B \to 0$ and for any $v \in V$, we have

$$\|\Delta u\|_V=\frac{\|u(a+\delta a)-u(a)-T(\delta a)\|_V}{\|\delta a\|_B}\to 0.$$

and this ensures differentiability. The proof is complete. □

Remark 1.2.1. The derivative characterization (1.12) is a natural extension of (1.6).

We have the following derivative formulas for the MOLS:

Theorem 1.2.3. *The first-order derivative of the MOLS functional* (1.10) *is given by:*

$$J'(a;\delta a)=-\frac{1}{2}T\left(g'(a;\delta a),u(a)+z,u(a)-z\right). \tag{1.16}$$

Proof. Denoting

$$\Delta J=2\frac{J(a+t\delta a)-J(a)}{t},$$

we have

$$\begin{aligned}
\frac{\Delta J}{2}&=\frac{1}{t}\left[T(g(a+t\delta a),u(a+t\delta a)-z,u(a+t\delta a)-z)-T(g(a),u(a)-z,u(a)-z)\right]\\
&=\frac{1}{t}\left[T(g(a+t\delta a),u(a+t\delta a)-z,u(a+t\delta a)-z)-T(g(a),u(a+t\delta a)-z,u(a+t\delta a)-z)\right]\\
&\quad+\frac{1}{t}\left[T(g(a),u(a+t\delta a)-z,u(a+t\delta a)-z)-T(g(a),u(a)-z,u(a)-z)\right]\\
&=T\left(\frac{g(a+t\delta a)-g(a)}{t},u(a+t\delta a)-z,u(a+t\delta a)-z\right)\\
&\quad+\frac{1}{t}\left[T(g(a),u(a+t\delta a)-z,u(a+t\delta a)-z)-T(g(a),u(a)-z,u(a)-z)\right]\\
&=T\left(\frac{g(a+t\delta a)-g(a)}{t},u(a+t\delta a)-z,u(a+t\delta a)-z\right)\\
&\quad+\frac{1}{t}\left[T(g(a),u(a+t\delta a)-z,u(a+t\delta a)-z)-T(g(a),u(a)-z,u(a+t\delta a)-z)\right]\\
&\quad+\frac{1}{t}\left[T(g(a),u(a)-z,u(a+t\delta a)-z)-T(g(a),u(a)-z,u(a)-z)\right].
\end{aligned}$$

Thus we get

$$\begin{aligned}\frac{\Delta J}{2} &= T\left(\frac{g(a+t\delta a)-g(a)}{t}, u(a+t\delta a)-z, u(a+t\delta a)-z\right)\\ &\quad + T\left(g(a), \frac{u(a+t\delta a)-u(a)}{t}, u(a+t\delta a)-z\right) + T\left(g(a), u(a+t\delta a)-z, \frac{u(a+t\delta a)-u(a)}{t}\right)\\ &= T\left(g(a), \frac{u(a+t\delta a)-u(a)}{t}, u(a+t\delta a)-z\right) + 2T\left(g(a), u(a+t\delta a)-z, \frac{u(a+t\delta a)-u(a)}{t}\right)\\ &= A_1(t) + A_2(t),\end{aligned}$$

where

$$\begin{aligned}A_1(t) &= T\left(\frac{g(a+t\delta a)-g(a)}{t}, u(a+t\delta a)-z, u(a+t\delta a)-z\right),\\ A_2(t) &= 2T\left(g(a), \frac{u(a+t\delta a)-u(a)}{t}, u(a+t\delta a)-z\right).\end{aligned}$$

In the limit as t goes to zero, the terms become the following:

$$\begin{aligned}\lim_{t\to 0} A_1(t) &= T\left(g'(a;\delta a), u(a)-z, u(a)-z\right),\\ \lim_{t\to 0} A_2(t) &= 2T\left(g(a), u'(a;\delta a), u(a)-z\right).\end{aligned}$$

We therefore end up with

$$\begin{aligned}J'(a,\delta a) &= \frac{1}{2}\lim_{t\to 0}\Delta J = \frac{1}{2}\lim_{t\to 0}\left[A_1(t)+A_2(t)\right]\\ &= \frac{1}{2}T\left(g'(a;\delta a), u(a)-z, u(a)-z\right) + T\left(g(a), u'(a;\delta a), u(a)-z\right)\\ &= \frac{1}{2}T\left(g'(a;\delta a), u(a)-z, u(a)-z\right) - T(g'(a;\delta a), u(a), u(a)-z)\\ &= -\frac{1}{2}T\left(g'(a;\delta a), u(a)+z, u(a)-z\right),\end{aligned}$$

where Theorem 1.2.2 was used. The proof is complete. □

Remark 1.2.2. The derivative characterization (1.16) is a natural extension of (1.7).

We now proceed to give compute the second-order derivative for the MOLS:

Theorem 1.2.4. *The second-order derivative of the MOLS functional* (1.10) *is given by:*

$$J''(a;\delta a_1,\delta a_2) = -\frac{1}{2}T\left(g''(a;\delta a_1,\delta a_2), u(a)+z, u(a)-z\right) + T(g(a), u'(a,\delta a_1), u'(a,\delta a_2)). \quad (1.17)$$

Proof. Recall that the second-order directional derivative is given by:

$$J''(a;\delta a_1,\delta a_2)=\lim_{t\to 0^+}\frac{J'(a+t\delta a_2;\delta a_1)-J'(a,\delta a_1)}{t}.$$

Setting

$$\Delta J=\frac{J'(a+t\delta a_2;\delta a_1)-J'(a;\delta a_1)}{t},$$

by applying Theorem 1.2.3, we have

$$\begin{aligned}
\Delta J &= \frac{-\frac{1}{2}T\left(g'(a+t\delta a_2;\delta a_1),u(a+t\delta a_2)+z,u(a+t\delta a_2)-z\right)-\left(-\frac{1}{2}T\left(g'(a;\delta a_1),u(a)+z,u(a)-z\right)\right)}{t}\\
&=-\frac{1}{2}\left[t^{-1}\left(T\left(g'(a+t\delta a_2;\delta a_1),u(a+t\delta a_2)+z,u(a+t\delta a_2)-z\right)-T\left(g'(a;\delta a_1),u(a)+z,u(a)-z\right)\right)\right]\\
&=-\frac{1}{2}\left[t^{-1}\left(T\left(g'(a+t\delta a_2;\delta a_1),u(a+t\delta a_2)+z,u(a+t\delta a_2)-z\right)\right.\right.\\
&\quad\left.\left.-T(g'(a;\delta a_1),u(a+t\delta a_2)+z,u(a+t\delta a_2)-z)\right)\right]\\
&\quad-\frac{1}{2}\left[t^{-1}\left(T(g'(a;\delta a_1),u(a+t\delta a_2)+z,u(a+t\delta a_2)-z)-T\left(g'(a;\delta a_1),u(a)+z,u(a)-z\right)\right)\right]\\
&=-\frac{1}{2}\left[T\left(t^{-1}\left(g'(a+t\delta a_2;\delta a_1)-g'(a;\delta a_1)\right),u(a+t\delta a_2)+z,u(a+t\delta a_2)-z\right)\right]\\
&\quad-\frac{1}{2}\left[t^{-1}\left(T(g'(a;\delta a_1),u(a+t\delta a_2)-z,u(a+t\delta a_2)-z)-T\left(g'(a;\delta a_1),u(a)+z,u(a+t\delta a_2)-z\right)\right)\right]\\
&\quad-\frac{1}{2}\left[t^{-1}\left(T(g'(a;\delta a_1),u(a)+z,u(a+t\delta a_2)-z)-T\left(g'(a;\delta a_1),u(a)+z,u(a)-z\right)\right)\right]\\
&=-\frac{1}{2}\left[T\left(t^{-1}\left(g'(a+t\delta a_2;\delta a_1)-g'(a;\delta a_1)\right),u(a+t\delta a_2)+z,u(a+t\delta a_2)-z\right)\right]\\
&\quad-\frac{1}{2}T\left[g'(a;\delta a_1),t^{-1}\left(u(a+t\delta a_2)-u(a)\right),u(a+t\delta a_2)-z\right]\\
&\quad-\frac{1}{2}T\left[g'(a;\delta a_1),u(a)+z,t^{-1}\left(u(a+t\delta a_2)-u(a)\right)\right]\\
&=-\frac{1}{2}B_1(t)-\frac{1}{2}B_2(t)-\frac{1}{2}B_3(t).
\end{aligned}$$

where

$$\begin{aligned}
B_1(t)&=T\left(t^{-1}\left(g'(a+t\delta a_2;\delta a_1)-g'(a;\delta a_1)\right),u(a+t\delta a_2)+z,u(a+t\delta a_2)-z\right)\\
B_2(t)&=T\left[g'(a;\delta a_1),t^{-1}\left(u(a+t\delta a_2)-u(a)\right),u(a+t\delta a_2)-z\right]\\
B_3(t)&=T\left[g'(a;\delta a_1),u(a)+z,t^{-1}\left(u(a+t\delta a_2)-u(a)\right)\right].
\end{aligned}$$

Since

$$\lim_{t\to 0} B_1(t) = T\left(g''(a;\delta a_1,\delta a_2), u(a)+z, u(a)-z\right)$$
$$\lim_{t\to 0} B_2(t) = T(g'(a;\delta a_1), u'(a;\delta a_2), u(a)-z)$$
$$\lim_{t\to 0} B_3(t) = T(g'(a;\delta a_1), u(a)+z, u'(a;\delta a_2))$$

we have

$$\begin{aligned}
J''(a,\delta a_1,\delta a_2) &= \lim_{t\to 0}\left[-\frac{1}{2}B_1(t) - \frac{1}{2}B_2(t) - \frac{1}{2}B_3(t)\right] \\
&= -\frac{1}{2}T\left(g''(a;\delta a_1,\delta a_2), u(a)+z, u(a)-z\right) - \frac{1}{2}T(g'(a;\delta a_1), u'(a;\delta a_2), u(a)-z) \\
&\quad - \frac{1}{2}T(g'(a;\delta a_1), u(a)+z, u'(a;\delta a_2)) \\
&= -\frac{1}{2}T\left(g''(a;\delta a_1,\delta a_2), u(a)+z, u(a)-z\right) - T(g'(a;\delta a_1), u'(a;\delta a_2), u(a)).
\end{aligned}$$

By applying Theorem 1.2.2, we obtain

$$\begin{aligned}
J''(a,\delta a_1,\delta a_2) &= -\frac{1}{2}T\left(g''(a;\delta a_1,\delta a_2), u(a)+z, u(a)-z\right) - T(g'(a;\delta a_1), u'(a;\delta a_2), u(a)) \\
&= -\frac{1}{2}T\left(g''(a;\delta a_1,\delta a_2), u(a)+z, u(a)-z\right) + T(g(a), u'(a;\delta a_1), u'(a;\delta a_2)),
\end{aligned}$$

and the proof is complete. □

Remark 1.2.3. Note that the sign of the first term in $-\frac{1}{2}T\left(g''(a;\delta a_1,\delta a_2), u(a)+z, u(a)-z\right)$ in the formula for the second order derivative is undetermined. Therefore, for the positivity of the second-order derivative, it is necessary to assume that

$$\alpha\|u'(a,\delta a_2)\|_V^2 - \frac{1}{2}T\left(g''(a;\delta a_1,\delta a_2), u(a)+z, u(a)-z\right) \geq 0, \tag{1.18}$$

uniformly. If g coincides with the identity map, then $g''(a,\cdot,\cdot) = 0$ and we recover the formula corresponding to (1.8).

1.3 Conclusions

We have developed direct expressions for the MOLS functional that give the first derivative (1.16) and second derivative (1.17). Moreover, we note that the first derivative of the MOLS functional has a direct expression that does not depend on the derivative of u. This is a remarkable property that allows for a simple and direct computation of the MOLS derivative. In comparison, the computation of the derivative $Du(a)(\delta a)$ is the

main difficulty when using the alternative OLS functional. In the next chapters we discuss ways to overcome this difficulty and compute the OLS derivatives using the adjoint method. However, the efforts for the OLS functional to get past the derivative computation of u can be quite significant. The MOLS functional does not suffer from this need and has the advantage of having a very simple and easily computable expression for its first derivative. Thus it presents a good alternative to the more common OLS functional.

As remarked, the MOLS functional does not retain convexity for nonlinear parameters unless (1.18) can be assumed. Convexity for linear parameters is a major strength of the MOLS functional. Therefore, for nonlinear parameters more care is needed for the MOLS, just as for the generally non-convex OLS functional.

Chapter 2

A Scalar Problem with a Nonlinear Parameter

We begin this chapter by introducing an exemplary scalar boundary value problem where the unknown parameter appears in the form of a nonlinear map. Out goal is to develop expressions for the output-least squares (OLS) functional together with its derivatives for this case of variational problems with nonlinear parameters. We derive the formulas first in the continuous setting and then give discretization details.

2.1 Problem Statement

Consider the following problem

$$-\nabla \cdot (g(a)\nabla u) = f \text{ in } \Omega, \quad u = 0 \text{ on } \partial\Omega. \tag{2.1}$$

Define $V = \{u \in H^1(\Omega) \mid u = 0 \text{ on } \partial\Omega\}$ as the space of solution functions u and A as the space of admissible parameters a. We take A as a nonempty, closed, and convex subset of a Banach space B. The parameter $a \in A$ appears nonlinearly as governed by the parameter map $g : A \to A$. Let g be a continuous and differential map that satisfies $range(g) \subset A$. To solve this system using finite element method, we first bring the equation into its weak form. Taking equation (2.1) we can multiply both sides with a test function $v \in V$ as well as integrate both sides over the domain Ω without destroying equality. Doing so gives us

$$-\int_\Omega \nabla \cdot (g(a)\nabla u) \cdot v = \int_\Omega f \cdot v \qquad \forall\, v \in V.$$

We can apply integration by parts to the left hand side and use Green's identity to obtain

$$\int_\Omega g(a)\nabla u \cdot \nabla v - \int_{\partial\Omega} g(a)v \cdot \frac{\partial u}{\partial n} = \int_\Omega f \cdot v \qquad \forall\, v \in V.$$

Now, the test function v lives in the space V as the solution variable u. Thus it also fulfills the same boundary conditions as u, and in this case we have enforced homogeneous Dirichlet boundary conditions. This means that the boundary integral on $\partial\Omega$ falls away since the test function v is zero along the entire boundary. We are then left with the other two terms that together make up the weak form

$$\int_\Omega g(a)\nabla u \cdot \nabla v = \int_\Omega f \cdot v \qquad \forall\, v \in V. \tag{2.2}$$

Notice that the solution variable u now only needs to be once differentiable, as opposed to (2.1) where we needed u to be twice differentiable. To look at the inverse problems in a more abstract framework, we can define $T(g(a), u, v) = \int_\Omega g(a)\nabla u \cdot \nabla v$ and $m(v) = \int_\Omega f \cdot v$. Then (2.2) is identical to the variational problem

$$T(g(a), u, v) = m(v) \qquad \forall\, v \in V. \tag{2.3}$$

This formulation allows us to talk about general inverse problems that lead to a structure like this, and are not limited to the specific equation (2.2). We require that the same assumptions hold as described for the identical variational problem in Section 1.1, including the continuity and coercivity of T. Theorems 1.2.1 and 1.2.2 hold as well.

2.2 Inverse Problem Functionals

Given a variational problem such as (2.2), the forward problem consists of finding the solution function u given that the parameter a and the load function f are known. The inverse problem of parameter identification, on the other hand, aims at finding the parameter a given that the load function f is known and that we have a measurement z of u. More descriptively, we want to find the parameter a such that the resulting solution u from the variational problem (2.2) matches the measured solution z as closely as possible. Thus, we need a way to quantify how well any given choice of a matches this requirement. Let u_a be the solution of the variational problem given a choice of a. Call $J(a)$ the objective functional that tells us how large the error between the solution u_a and the measured solution z is. The process of solving the inverse problem then becomes finding the coefficient a that minimizes $J(a)$, that is finding

$$\arg\min_{a \in A} J(a).$$

This is usually done with the help of iterative minimization methods such as Newton or Quasi-Newton methods. The challenge then lies in finding the derivatives of the functional $J(a)$ which are needed for the

minimization methods. Using a good iterative solver and having efficient ways of computing the derivatives of $J(a)$ are the main difficulties in solving an inverse problem.

2.2.1 Objective Functional for the Inverse Problem

The output least-squares (OLS) functional is given as

$$J_{OLS}(a) = \frac{1}{2} \langle u(a) - z, u(a) - z \rangle = \frac{1}{2} \|u(a) - z\|_V^2 .$$

As the name suggests, it quantifies the least-squares type of error between the output $u(a)$ of the variational problem and the measured solution z of u. The goal of the inverse problem is to determine the parameter a for which $u(a)$ most closely matches the measurement z.

Inverse problems are very ill-posed and, arising from that, are innately instable. Therefore it is necessary to add a regularizer $R(a)$ to the functional that can alleviate some the instabilities of the inverse problem. The regularized OLS then becomes

$$J_{OLS}(a) = \frac{1}{2} \|u(a) - z\|_V^2 + \kappa R(a), \tag{2.4}$$

where the regularization parameter $\kappa \in \mathbb{R}$ is small positive constant. The key is to have κ small enough so that the regularizer $R(a)$ does not dominate the minimization process, yet large enough to reduce the ill-posed nature of the inverse problem. We will use Tikhonov regularization of the form

$$R(a) = \|a\|_D^2 ,$$

where the choice of D can be the L_2 norm, the H_1 norm, or the H_1 semi-norm. These norms are given as

$$\begin{aligned} \|a\|_{L_2}^2 &= \int_\Omega (a \cdot a), \\ \|a\|_{H_1\text{-semi}}^2 &= \int_\Omega (\nabla a \cdot \nabla a) \qquad\qquad = \|\nabla a\|_{L_2}^2 , \\ \|a\|_{H_1}^2 &= \int_\Omega (a \cdot a) + (\nabla a \cdot \nabla a) = \|a\|_{L_2}^2 + \|\nabla a\|_{L_2}^2 . \end{aligned} \tag{2.5}$$

2.2.2 Derivate Formulas for the OLS Functional

Now that we have defined the functionals $J(a)$ we need a way to find the minimum of these functions. To that end, we use iterative minimization methods that require us to compute the first (and possibly the second) derivatives of $J(a)$. Recall that the regularized OLS functional was given as

$$\begin{aligned} J_{OLS}(a) &= \frac{1}{2} \|u(a) - z\|_V^2 + \kappa R(a) \\ &= \frac{1}{2} \langle u(a) - z, u(a) - z \rangle + \kappa R(a). \end{aligned}$$

The derivative of J at some $a \in A$ in the direction δa can be found by use of the chain rule.

$$\begin{aligned} DJ_{OLS}(a)(\delta a) &= \frac{1}{2}\langle Du(a)(\delta a), u(a) - z\rangle + \frac{1}{2}\langle u(a) - z, Du(a)(\delta a)\rangle + \kappa DR(a)(\delta a) \\ &= \langle Du(a)(\delta a), u(a) - z\rangle + \kappa DR(a)(\delta a) \end{aligned} \tag{2.6}$$

Here, $DR(a)(\delta a)$ is the derivative of the regularizer $R(a)$ in the direction δa. By definition the regularizer is simply the norm of the parameter, and as such, the derivative can be found directly and very easily. The difficulty lies in finding $Du(a)(\delta a)$, the derivative of the parameter-to-solution map $u(a)$.

Several methods exists to find this derivative, including the direct—or classical—method, as well as the popular adjoint method and adjoint stiffness method. The classical method is the most natural in that it finds $Du(a)(\delta a)$ by solving the variational problem from Theorem 1.2.2 for the direction δa. This leads to the need of solving a forward problem once for each derivative direction δa which becomes quickly intractable for large problems. A much more efficient alternative is the adjoint method which requires only the solution of a single additional forward problem. Recent uses of the adjoint method can be found in [18, 19, 20, 21, 22, 23, 24, 25], and a survey article of first and second-order adjoint methods is given by Tortorelli and Michaleris [26]. The adjoint stiffness method builds on the adjoint equation but linearizes out the parameter a. The derivative can then be found directly with a single matrix equations, versus the adjoint method which has to evaluate $Du(a)(\delta a)$ for each direction δa. The drawback of the adjoint stiffness method is that by its nature it is only applicable to linear parameters a. We are dealing with nonlinearly appearing parameters, and as such we can't make use of the adjoint stiffness method. Instead we will utilize the adjoint method and develop expressions for the nonlinear parameter case.

2.2.3 Adjoint Method

The idea behind the adjoint method is to avoid having to compute $Du(a)(\delta a)$ directly. In order to achieve this, an adjoint variable $w \in V$ is introduced in such a way that its properties will allow us remove the undesired term $Du(a)(\delta a)$ from the calculation. To begin deriving the adjoint method, define a new functional $L : A \times V \to \mathbb{R}$ as

$$L(a, v) = J_{OLS}(a) + T(g(a), u, v) - m(v), \tag{2.7}$$

where the terms $T(,,)$ amd $m()$ are identical to the variational problem (2.3). Since equation (2.3) holds for any $u \in V$, we have by construction that

$$L(a, v) = J_{OLS}(a) \qquad \forall\, v \in V.$$

Thus, we also have for any derivative direction δa that

$$\partial_a L(a, v)(\delta a) = DJ_{OLS}(a)(\delta a) \qquad \forall\, v \in V. \tag{2.8}$$

However, we can also differentiate (2.7) directly with respect to a to get

$$\begin{aligned}\partial_a L(a,v)(\delta a) = &\langle Du(a)(\delta a), u(a) - z\rangle + \kappa DR(a)(\delta a) \\ &+ T(Dg(a)(\delta a), u(a), v) + T(g(a), Du(a)(\delta a), v).\end{aligned} \tag{2.9}$$

Now, consider some $a \in A$ and introduce the adjoint variable $w(a)$ as the solution to the variational problem

$$T(g(a), w, v) = \langle z - u(a), v\rangle \qquad \forall\, v \in V, \tag{2.10}$$

where $u(a)$ is the solution of the variational problem (2.3). Notice that the following holds by the symmetry of T in the second and third arguments when we take $v = Du(a)(\delta a)$ in (2.10).

$$\begin{aligned}T(g(a), Du(a)(\delta a), w) &= T(g(a), w, Du(a)(\delta a)) \\ &= \langle Du(a)(\delta a), z - u(a)\rangle \\ &= -\langle Du(a)(\delta a), u(a) - z\rangle\end{aligned}$$

Using this fact it is easy to see that, if we let v in (2.9) be the adjoint variable w, we get

$$\begin{aligned}\partial_a L(a,w)(\delta a) &= \langle Du(a)(\delta a), u(a) - z\rangle + \kappa DR(a)(\delta a) + T(Dg(a)(\delta a), u(a), w) + T(g(a), Du(a)(\delta a), w) \\ &= \langle Du(a)(\delta a), u(a) - z\rangle + \kappa DR(a)(\delta a) + T(Dg(a)(\delta a), u(a), w) - \langle Du(a)(\delta a), u(a) - z\rangle \\ &= \kappa DR(a)(\delta a) + T(Dg(a)(\delta a), u(a), w)\end{aligned}$$

Now we can use fact (2.8) to get an expression for the derivative of J_{OLS} that does not rely on $Du(a)(\delta a)$

$$DJ_{OLS}(a)(\delta a) = \kappa DR(a)(\delta a) + T(Dg(a)(\delta a), u(a), w) \tag{2.11}$$

Here we only have to compute the derivatives of R and g with respect to a. Both of these are functions that depend directly on a and as such the derivatives can be computed easily. The process of finding the OLS derivative $DJ_{OLS}(a)(\delta a)$ using the adjoint method can thus be summarized in the following way:

Step 1. Find $u(a)$ by solving (2.3).

Step 2. Find $w(a)$ by solving (2.10).

Step 3. Compute $DJ_{OLS}(a)(\delta a)$ from (2.11).

2.2.4 Direct Method for the Second Derivative

Taking the derivative expression (2.11) for $DJ_{OLS}(a)(\delta a_1)$ and applying another derivative in direction δa_2 gives

$$\begin{aligned}D^2 J_{OLS}(a)(\delta a_1, \delta a_2) = &\kappa D^2 R(a)(\delta a_1, \delta a_2) + T(D^2 g(a)(\delta a_1, \delta a_2), u, w) \\ &+ T(Dg(a)(\delta a_1), Du(a)(\delta a_2), w) + T(Dg(a)(\delta a_1), u, Dw(a)(\delta a_2)).\end{aligned}$$

This expression allows us to compute the second derivative $D^2 J_{OLS}(\delta a, \delta a)$, but it has the drawback that it requires both derivatives $\delta u = Du(a)(\delta a)$ and $\delta w = Dw(a)(\delta a)$. The derivative δu is found through direct computation using (1.12), whereas δw is found by taking the derivative of the adjoint variational problem (2.10), giving

$$T(Dg(a)(\delta a), w, v) + T(g(a), Dw(a)(\delta a), v) = -\langle Du(a)(\delta a), v\rangle .$$

The direct method therefore requires us to solve two variational problems for each derivative direction δa, one for δu and one for δw. This becomes the dominating factor in the computation process and makes the method overall quite inefficient. A much better method is presented next which removes the need to calculate δw.

2.2.5 Second-Order Adjoint Method

The second-order adjoint method uses a direct computation method for the first derivative of $u(a)$ but utilizes the adjoint method to avoid direct computation of the second derivative of $u(a)$. The approach is very similar to that of the first order adjoint method. We introduce the a new functional as

$$\begin{aligned} L(a, v) &= DJ_{OLS}(a)(\delta a_2) + T(g(a), Du(a)(\delta a_2), v) + T(Dg(a)(\delta a_2), u, v) \\ &= \langle Du(a)(\delta a_2), u - z\rangle + \kappa DR(a)(\delta a_2) \\ &\quad + T(g(a), Du(a)(\delta a_2), v) + T(Dg(a)(\delta a_2), u, v), \end{aligned} \tag{2.12}$$

where we used the result (2.6) about the derivative $DJ_{OLS}(a)(\delta a)$. It is clear from (1.12) that the last two terms in L cancel out and we have

$$L(a, v) = DJ_{OLS}(a)(\delta a_2) \qquad \forall\, v \in V \tag{2.13}$$

$$\partial_a L(a, v)(\delta a_1) = D^2 J_{OLS}(a)(\delta a_1, \delta a_2) \quad \forall\, v \in V, \quad \forall\, \delta a_1 \in A \tag{2.14}$$

A direct computation of the partial derivative of L with respect to a yields

$$\begin{aligned} \partial_a L(a, v)(\delta a_1) = &\left\langle D^2 u(a)(\delta a_1, \delta a_2), u - z\right\rangle + \langle Du(a)(\delta a_2), Du(a)(\delta a_1)\rangle \\ &+ \kappa D^2 R(a)(\delta a_1, \delta a_2) \\ &+ T(Dg(a)(\delta a_1), Du(a)(\delta a_2), v) + T(g(a), D^2 u(a)(\delta a_1, \delta a_2), v) \\ &+ T(D^2 g(a)(\delta a_1, \delta a_2), u, v) + T(Dg(a)(\delta a_2), Du(a)(\delta a_1), v). \end{aligned} \tag{2.15}$$

Note the two occurrences of $D^2 u(a)(\delta a_1, \delta a_2)$. We wish to eliminate these terms with the second order derivatives of u. In order to do so, we again choose the adjoint variable $w(a)$ as the solution of the variational problem (2.10), that is

$$T(g(a), w, v) = \langle z - u, v\rangle \qquad \forall\, v \in V.$$

It follows again from symmetry of T in the second and third argument that

$$T(g(a), D^2u(a)(\delta a_1, \delta a_2), w) = -\left\langle D^2u(a)(\delta a_1, \delta a_2), u - z \right\rangle.$$

This equation lets us cancel exactly the two unwanted terms in (2.15) if we let $v = w$. Keeping the remaining terms gives us

$$\begin{aligned} \partial_a L(a, v)(\delta a_1) = &\langle Du(a)(\delta a_2), Du(a)(\delta a_1)\rangle + \kappa D^2R(a)(\delta a_1, \delta a_2) \\ &+ T(Dg(a)(\delta a_1), Du(a)(\delta a_2), w) + T(Dg(a)(\delta a_2), Du(a)(\delta a_1), w) \\ &+ T(D^2g(a)(\delta a_1, \delta a_2), u, w). \end{aligned} \tag{2.16}$$

Using fact (2.14) gives us the formulation for the second order derivative of the OLS functional. Let $\delta u = Du(a)(\delta a)$, then we have in particular that

$$\begin{aligned} D^2J_{OLS}(a)(\delta a, \delta a) = &\langle \delta u, \delta u\rangle + \kappa D^2R(a)(\delta a, \delta a) \\ &+ 2T(Dg(a)(\delta a), \delta u, w) \\ &+ T(D^2g(a)(\delta a, \delta a), u, w). \end{aligned} \tag{2.17}$$

To compute the second-order derivative of the OLS functional $D^2J_{OLS}(a)(\delta a, \delta a)$ in a direction δa using the second-order adjoint method we therefore have the following steps.

Step 1. Find $u(a)$ by solving (2.3).

Step 2. Find $Du(a)(\delta a)$ from (1.12).

Step 3. Find $w(a)$ by solving (2.10).

Step 4. Compute $D^2J_{OLS}(a)(\delta a, \delta a)$ from (2.17).

2.3 Finite Element Discretization

In the variational problem (2.2) we have two function spaces to deal with. The space V for the solution u and the space A for the parameter a. We will have to discretize both in order to be able to solve the problem computationally.

2.3.1 Discretization of the Solution Space

Let $\mathcal{T}$ be a triangulation that covers the domain Ω. Triangulation is the historical term for the mesh or grid on which the inverse problem will be solved. Traditionally, in two dimensional problems triangular grid

elements (cells) were used to build a mesh—hence the name triangulation—whereas we will be working with quadrilateral cells instead. Now, define the finite dimensional counterpart of V to be the space V_h of piecewise continuous polynomials of degree d_u relative to the triangulation $\mathcal{T}$. Consider for a while the case where $d_u = 1$ where we have piecewise linear polynomials. Then the grid points $x_i \in \Omega$ on the vertices of the cells correspond to the n degrees of freedom for the problem and they will be the locations on which we discretize the system. Take the polynomials $\varphi_i \in V_h$ given as

$$\varphi_i(x) = \begin{cases} 1, & x = x_i \\ 0, & x = x_j, j \neq i \end{cases}, \qquad 1 \leq i, j \leq n. \tag{2.18}$$

That is, φ_i is a piecewise linear polynomial that has a value of 1 on the triangulation grid point x_i and has the value 0 on all other triangulation grid points. The values $\varphi_i(x)$ for all other points are then uniquely defined through the constraint of piecewise linearity and are in effect linearly interpolated values of the adjacent grid points x_j. The polynomials φ_i are called basis functions and x_j are called the support points of the basis functions. The support of φ_i is the space on which $\varphi_i(x) > 0$ and covers the cells of $\mathcal{T}$ that are adjacent to grid point x_i.

In the case where degree $d_u > 1$ we need more information to uniquely define the polynomials of degree d_u and this information comes in the form of using more support points x_j. Having more support points increases the number of degrees of freedom n and makes computations more expensive, but it allows for a more accurate discretization. In practice there is a trade-off between accuracy and speed, and one often chooses the lowest degree d_u that can represent the solution sufficiently well. In our case, we will be able to use $d_u = 1$ most of the time.

It can be shown that the set $\{\varphi_1, \varphi_2, \ldots, \varphi_n\}$ forms a basis for V_h. We can therefore uniquely define any function in V_h as a linear combination of basis functions φ_i. Thus, take $u_h \in V_h$ given as

$$u_h(x) = \sum_{i=1}^{n} U_i \varphi_i(x),$$

where $U_i \in \mathbb{R}$ are the coefficients for the linear combination of basis functions. We define u_h to match the solution function $u \in V$ on the support points x_i by requiring

$$u_h(x_i) = u(x_i), \qquad 1 \leq i \leq n.$$

With this, u_h is the finite element approximation of u that is uniquely defined by the coefficients U_i. By construction of the basis functions in (2.18) we then have

$$u(x_j) = u_h(x_j) = \sum_{i=1}^{n} U_i \varphi_i(x_j) = U_j \underbrace{\varphi_j(x_j)}_{=1} + \sum_{i \neq j} U_i \underbrace{\varphi_i(x_j)}_{=0} = U_j.$$

The coefficients U_j are thus the representation of u on the finite element mesh points x_j. We have hereby transformed the problem of finding u in the infinite dimensional function space V to finding the vector $U \in \mathbb{R}^n$ whose elements U_j describe the finite element discretization of u.

2.3.2 Discretization of the Parameter Space

We can discretize the parameter function a in exactly the same way as we dicretized the solution u. Define $\mathcal{T}_A$ as another triangulation over Ω to be used for the discretization of a. Let A_h be the space of piecewise continuous polynomials of degree d_a with basis $\{\chi_1, \chi_2, \ldots, \chi_k\}$. Then let the finite element discretization of $a \in A$ be $a_h \in A_h$ where

$$a_h(x) = \sum_{i=1}^{k} A_i \chi_i(x). \tag{2.19}$$

Then finding a is reduced to the task of finding the vector $A \in \mathbb{R}^k$ whose elements are the coefficients A_i.

Considerations for the meshes

It is possible to take $\mathcal{T}_A = \mathcal{T}$ and use the same mesh for solution and parameter. This makes computations much simpler because the basis functions of the different spaces will be defined on the same regions. That is, for every cell $c \in \mathcal{T}$ we also have $c \in \mathcal{T}_A$, and both basis functions φ_i and χ_i are defined for all support points of the cell c. When computing the matrix system of the discretized variational problem the contribution from each mesh cell can easily be computed since basis functions will be defined for each cell. On the other hand, it can be argued as in [27] that the parameter a does not require as fine of a discretization as the solution u. Having a coarser coefficient mesh $\mathcal{T}_A$ reduces the overall size of the system and thereby the total number of computations while also acting as a form of additional regularization. In this case when $\mathcal{T}_A$ is coarser, there will be cells $c_u \in \mathcal{T}$ that do not exist on $\mathcal{T}_A$. To compute the contribution from a cell c_u all φ_i are defined on c, but some χ_i may be undefined on the cell $c_u \notin \mathcal{T}_A$. Thus, it is necessary to map χ_i to the cell c_u which is itself a costly process. While having separate meshes also incurs this overhead from making the computations themselves more complicated, the benefit from reducing the system size can outweigh these problems, especially in a three dimensional setting. Separate meshes tie in very well with adaptive mesh refinement, because the solution and parameter meshes can be refined independently. We will consider the results for the case of $\mathcal{T}_A = \mathcal{T}$.

2.3.3 Discretized Varitational Form

Consider a variational problem of the type (2.3) which was given as

$$T(g(a), u, v) = m(v) \qquad \forall\, v \in V,$$

where $u, v \in V$ and $a \in A$. Using our finite element approximation in the spaces V_h and A_h, we can replace u with $u_h \in V_h$ and a with $a_h \in A_h$. Furthermore, since the equation in the discretized space holds for all $v_h \in V_h$, it also holds for all basis functions $\varphi_j \in V_h$. We therefore have

$$\begin{aligned} T(g(a_h), u_h, \varphi_j) &= m(\varphi_j), \qquad 1 \leq j \leq n, \\ T(g(a_h), \sum_{i=1}^{n} U_i \varphi_i, \varphi_j) &= m(\varphi_j), \qquad 1 \leq j \leq n. \end{aligned}$$

Using the linearity of T in the second argument, we get

$$\sum_{i=1}^{n} U_i T(g(a_h), \varphi_i, \varphi_j) = m(\varphi_j), \qquad 1 \leq j \leq n,$$

which is equivalent to the matrix formulation

$$K(a_h)U = F, \tag{2.20}$$

where $U \in \mathbb{R}^n$ is the vector of coefficients U_i and $K(a_h) \in \mathbb{R}^{n \times n}$ and $F \in \mathbb{R}^n$ are given by

$$[K(a_h)]_{i,j} = T(g(a_h), \varphi_i, \varphi_j), \tag{2.21}$$

$$F_j = m(\varphi_j). \tag{2.22}$$

Solving the discrete variational problem to find u_h now consists of constructing the matrices K and F and then solving the linear system (2.20) for U.

The particular form for the scalar problem Laplace problem (2.2) gives us

$$\begin{aligned} [K(a_h)]_{i,j} &= \int_\Omega g(a_h) \nabla \varphi_i \cdot \nabla \varphi_j, \\ F_j &= \int_\Omega f \cdot \varphi_j. \end{aligned}$$

These integrals are usually computed using a quadrature rule and can thus be replaced by a summation over a set of quadrature points x_q. This is straightforward in general, but we have to consider how to discretize $g(a_h)$ in (2.21). Because of the nonlinearity of the map, we have to precompute the values of $g(a_h(x_q))$ for all x_q. In essence we have

$$g(a_h(x)) = g\left(\sum_{i=1}^{k} A_i \chi_i(x) \right).$$

To approximate the term $g(a_h)$, do the following for all quadrature points x_q:

Step 1. Evaluate $y_q := a_h(x_q) = \sum_{i=1}^{k} A_i \chi_i(x_q)$.

Step 2. Evaluate $z_q := g(y_q)$.

Then use z_q as the precomputed values of $g(a_h(x_q))$. We will use this approach for all following occurrences of $g()$ or $g'()$ in the discretized formulas.

2.4 Discretized OLS

Recall the definition (2.4) of the regularized output least-squares (OLS) functional. Let the discretized version of $z \in V$ on the triangulation $\mathcal{T}$ be given by $\sum_{j=1}^{n} Z_j \varphi_j$, with $Z \in \mathbb{R}^n$ as the vector of coefficients Z_j. We have

$$\begin{aligned} J_{OLS} = \frac{1}{2} \langle u - z, u - z \rangle + \kappa R(a_h) &= \frac{1}{2} \left\langle \sum_{i=1}^{n} (U_i - Z_i)\varphi_i, \sum_{j=1}^{n} (U_j - Z_j)\varphi_j \right\rangle + \kappa R(a_h) \\ &= \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} (U_i - Z_i) \langle \varphi_i, \varphi_j \rangle (U_j - Z_j) + \kappa R(a_h) \\ &= (U - Z)^T M (U - Z) + \kappa R(a_h) \end{aligned}$$

with $M_{i,j} = \langle \varphi_i, \varphi_j \rangle = \int_\Omega \varphi_i \cdot \varphi_j$. Note that $M \in \mathbb{R}^{n \times n}$ matches the dimensions of U and Z.

To discretize $R(a_h)$, consider the matrices $M_a, K_a \in \mathbb{R}^{k \times k}$ given as

$$\begin{aligned} [M_a]_{i,j} &= \langle \chi_i, \chi_j \rangle , \\ [K_a]_{i,j} &= \langle \nabla \chi_i, \nabla \chi_j \rangle . \end{aligned}$$

Then from (2.5) it follows that for a given choice of regularizer norm we get

$$\begin{aligned} R(a_h)_{L_2} &= A^T M_a A, \\ R(a_h)_{H_1\text{-semi}} &= A^T K_a A, \\ R(a_h)_{H_1} &= A^T (K_a + M_a) A. \end{aligned} \tag{2.23}$$

2.4.1 OLS Derivative

When computing the gradient $DJ_{OLS}(a_h)(\delta a_h)$, we necessarily have to compute the derivative of the parameter a_h. For this we want to consider $a_h = \sum_{i=1}^{k} A_i \chi_i$ as a function of the coefficient vector A, that is, we consider $a_h(A)$. For a linearly appearing coefficient the j-th partial derivative $\partial_j a_h$ for $j = 1, \ldots, k$

becomes simply

$$\begin{aligned}\partial_j a_h &= \frac{\partial}{\partial A_j} a_h = \frac{\partial}{\partial A_j} \sum_{i=1}^{k} (A_i \chi_i) = \sum_{i=1}^{k} \frac{\partial}{\partial A_j} (A_i \chi_i) \\ &= \sum_{i=1}^{k} \frac{\partial}{\partial A_j} (A_i) \chi_i + \frac{\partial}{\partial A_j} (\chi_i) A_i = \frac{\partial A_j}{\partial A_j} \chi_j = \chi_j.\end{aligned} \tag{2.24}$$

Because the coefficients appear linearly, the partial derivatives are either zero or one and we recover simply the basis function χ_j as the j-th directional derivative of $\partial_j a_h$. This makes it very simple to calculate the derivative vector. Alternatively we can think of the derivative as

$$\partial_j a_h = \sum_{i=1}^{k} X_i \chi_i, \text{ where } X_i = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}. \tag{2.25}$$

In the case of nonlinear parameters with mapping $g(a_h)$ we have to now also consider the derivative of g. Using the result from above we have

$$\partial_j g(a_h) = \frac{\partial}{\partial A_j} g(a_h) = \frac{\partial g(a_h)}{\partial a_h} \cdot \frac{\partial a_h}{\partial A_j} = g'(a_h) \cdot \chi_j. \tag{2.26}$$

In effect, we now have

$$\partial_j g(a_h) = \sum_{i=1}^{k} X_i \chi_i, \text{ where } X_i = \begin{cases} g'(a_h), & i = j \\ 0, & i \neq j \end{cases}. \tag{2.27}$$

If we take the linear map $g(a_h) = a_h$, then $g'(a_h) = 1$ and we recover equation (2.25).

The derivative $DR(a_h)(\delta a_h)$ is found by using (2.24). Call $\partial_j R(a_h)$ the j-th partial derivative of the regularizer. Taking the L_2 norm as an example, we have for any one of the derivative directions $1 \leq j \leq k$ that

$$\partial_j R(a_h) = \partial_j \left(\int_\Omega a_h \cdot a_h \right) = 2 \int_\Omega \partial_j a_h \cdot a_h = 2 \int_\Omega \chi_j \cdot a_h = 2 \sum_{i=1}^{k} A_i \int_\Omega \chi_j \cdot \chi_i.$$

Taking all derivative directions immediately gives us $\nabla R(a_h) = 2 M_a A$ for the L_2 norm. The other norms work analogously and we have

$$\begin{aligned}\nabla R(a_h)_{L_2} &= 2 M_a A, \\ \nabla R(a_h)_{H_1\text{-semi}} &= 2 K_a A, \\ \nabla R(a_h)_{H_1} &= 2 (K_a + M_a) A.\end{aligned} \tag{2.28}$$

The gradient formulation of the OLS functional via the adjoint method was given in (2.11). In the discrete setting we have

$$DJ_{OLS}(a_h)(\delta a_h) = \kappa DR(a_h)(\delta a_h) + T(Dg(a_h)(\delta a_h), u_h, w_h).$$

The derivative of $DR(a_h)$ was given just above, so we will now consider the derivative expression of T. For this we first need a way to compute the discrete adjoint variable w_h. Recall that w was given as the solution to the variational problem (2.10). In the discrete setting, this is equivalent to finding W, the coefficient vector of w_h, in the system

$$K(a_h)W = M(Z - U), \tag{2.29}$$

where Z is the discrete measured solution, U is the computed solution for a given a_h using (2.20), and $M_{i,j} = \langle \varphi_i, \varphi_j \rangle$ is the mass matrix for the solution variable. Take $\delta a_h = A_t$, that is the t-th derivative direction of a_h, then using (2.27) and the linearity of T in the second and third argument we get

$$\begin{aligned} T(Dg(a_h)(A_t), u_h, w_h) &= T(\partial_t g(a_h), u_h, w_h) \\ &= T(g'(a_h)\chi_t, u_h, w_h) \\ &= \sum_{i=1}^{n}\sum_{j=1}^{n} U_i T(g'(a_h)\chi_t, \varphi_i, \varphi_j) W_j \\ &= U^T K^t(a_h) W, \end{aligned} \tag{2.30}$$

where the values of the directional stiffness matrix $K^t(a_h)$ in the direction A_t are given as

$$[K^t(a_h)]_{i,j} = T(g'(a_h)\chi_t, \varphi_i, \varphi_j). \tag{2.31}$$

This directional stiffness matrix depends on the current value of a_h because of the occurrence of the map $g'(a_h)$. In the special case when we have the identity map $g(a_h) = a_h$, the derivative becomes $g'(a_h) = 1$ and does not depend on a_h. This case opens the way for algorithmic improvements in the computational implementation that lead to major speed improvements, as discussed in Section 4.2.2.

Overall, the OLS derivative $\partial_t J_{OLS}(a_h)$ in the t-th derivative direction A_t is given as

$$\partial_t J_{OLS}(a_h) = \kappa \partial_t R(a_h) + U^T K^t(a_h) W. \tag{2.32}$$

The gradient of the OLS derivative is given from the k partial derivatives as

$$\nabla J_{OLS}(a_h) = [\partial_1 J_{OLS}(a_h),\ \partial_2 J_{OLS}(a_h),\ \ldots,\ \partial_k J_{OLS}(a_h)]^T.$$

The steps to compute the OLS derivative are therefore:

Step 1. Compute U using (2.20).

Step 2. Compute W using (2.29).

Step 3. Compute $\nabla J_{OLS}(a_h)$ by finding $\partial_t J_{OLS}(a_h)$ for all $1 \leq t \leq k$ using (2.32).

2.4.2 Second-Order Derivative by the Second-Order Adjoint Method

Taking result (2.17) for the continuous second-order OLS derivative, we get that the discrete version for $D^2 J_{OLS}(a_h)(\delta a_h, \delta a_h)$ at a_h in the direction δa_h is given as:

$$\begin{aligned} D^2 J_{OLS}(a_h)(\delta a_h, \delta a_h) = \langle \delta u_h, \delta u_h \rangle + \kappa D^2 R(a_h)(\delta a_h, \delta a_h) \\ + 2T(Dg(a_h)(\delta a_h), \delta u_h, w_h) + T(D^2 g(a_h)(\delta a_h, \delta a_h), u_h, w_h). \end{aligned}$$

Beginning with the regularizer derivative, we saw that for the L_2 norm we have $\partial_j R(a_h) = 2\int_\Omega \chi_j \cdot a_h$. Thus, using (2.24) we get

$$\partial^2_{j,j} R(a_h) = 2\int_\Omega \chi_j \cdot \partial_j(a_h) = 2\int_\Omega \chi_j \cdot \chi_j.$$

That is $\nabla^2 R(a_h) = 2M$. The other norms follow identically and we have in total

$$\begin{aligned} \nabla^2 R(a_h)_{L_2} &= 2M_a, \\ \nabla^2 R(a_h)_{H_1\text{-semi}} &= 2K_a, \\ \nabla^2 R(a_h)_{H_1} &= 2(K_a + M_a). \end{aligned} \tag{2.33}$$

Next, consider the derivative of the parameter-to-solution map $Du_h(a_h)(\delta a_h)$. Take the t-th derivative direction of a_h and let $\partial_t u_h$ be the t-th partial derivative whose unique vector representation is

$$\nabla_t U = \sum_{i=1}^{n} U_i^t \varphi_i.$$

To find $\partial_t u_h = Du_h(a_h)(A_t)$ in the t-th derivative direction of a_h, we solve the discrete version of (1.12) which amounts to

$$T(g(a_h), \partial_t u_h, v_h) = -T(\partial_t g(a_h), u_h, v_h).$$

This is equivalent to solving the following matrix system for $\nabla_t U$.

$$K(a_h)(\nabla_t U) = -K^t(a_h)U \tag{2.34}$$

The solution $\nabla_t U$ is the discrete representation of the derivative of U in the direction A_t. It follows that for the t-th direction with $\delta u_h = \partial_t u_h$ we have

$$\langle \delta u_h, \delta u_h \rangle = (\nabla_t U) M (\nabla_t U). \tag{2.35}$$

For the first-order derivative of the OLS functional we saw in (2.26) that $\partial_j g(a_h) = g'(a_h)\chi_j$. The second-order derivative thus becomes

$$\begin{aligned} \partial^2_{j,j} g(a_h) = \partial_j(g'(a_h)\chi_j) = \frac{\partial}{\partial A_j} g'(a_h)\chi_j \\ = \frac{\partial g'(a_h)}{\partial a_h} \cdot \frac{\partial a_h}{\partial A_j} \cdot \chi_j + \frac{\partial \chi_j}{\partial A_j} \cdot g'(a_h) = g''(a_h) \cdot \chi_j \cdot \chi_j. \end{aligned} \tag{2.36}$$

We therefore have that the second-order term of the discretized second-order adjoint method can be computed as

$$T(\partial^2_{t,t} g(a_h), u_h, w_h) = \sum_{i=1}^{n}\sum_{j=1}^{n} U_i T(g''(a_h)\cdot\chi_t\cdot\chi_t, \varphi_i, \varphi_j) W_j = UK^{tt}(a_h)W, \tag{2.37}$$

where $K^{tt}(a_h)$ is the second-order directional stiffness matrix given as

$$[K^{tt}(a_h)]_{ij} = T(g''(a_h)\chi_t\cdot\chi_t, \varphi_i, \varphi_j). \tag{2.38}$$

Finally, the last term in the second-order adjoint method can be computed as

$$T(\partial_t g(a_h), \partial_t u_h, w_h) = \sum_{i=1}^{n}\sum_{j=1}^{n} (U_i^t) T(g'(a_h)\chi_t, \varphi_i, \varphi_j) W_j = (\nabla_t U) K^t(a_h) W, \tag{2.39}$$

where $\nabla_t U$ is the solution of (2.34) and $K^t(a_h)$ is the directional stiffness matrix as given in (2.31).

In total, the t-th second-order partial derivative $\partial^2_{t,t} J_{OLS}(a_h)$ without the regularizer term for $1 \leq t \leq k$ is given as

$$\partial^2_{t,t} J_{OLS}(a_h) = (\nabla_t U) M (\nabla_t U) + 2(\nabla_t U) K^t(a_h) W + U K^{tt}(a_h) W. \tag{2.40}$$

The Hessian is then formed from the partial derivatives as

$$[\nabla^2 J_{OLS}(a_h)]_{t,t} = \partial^2_{t,t} J_{OLS}(a_h).$$

For the sake of efficiency it makes sense to first compute the unregularized $\nabla^2 J_{OLS}$ and then add the regularizer $\nabla^2 R(a_h)$ afterwards since the regularizer can be computed in one step.

The process for the second-order adjoint method is thus:

Step 1. Compute U using (2.20).

Step 2. Compute W using (2.29).

Step 3. Compute $\nabla_t U$ by finding $\partial_t u_h$ for all directions $1 \leq t \leq k$ using (2.34).

Step 4. Compute the unregularized $\nabla^2 J_{OLS}$ by finding $\partial^2_{t,t} J_{OLS}(a_h)$ for all directions $1 \leq t \leq k$ using (2.40).

Step 5. Compute the regularizer term $\nabla^2 R(a_h)$ from (2.33) and add to $\nabla^2 J_{OLS}$.

Chapter 3

The Elastography Inverse Problem

A popular application for the elastography inverse problem is to model the deformation of human muscle tissue. Using, for example, elastography imaging, one can find measurements z of the tissue deformation $\bar{u}$ that occurs when applying a force to the tissue. Using these measurements z one would like to find the Lamé parameters λ and μ which determine the elastic properties of the tissue. This method is used to detect breast cancer in tissue and stems from the fact that healthy tissue has different elastic properties than tumorous tissue. In this context it is often assumed that muscle tissue is incompressible, isotropic, and continuous so that the elasticity equations hold. Strictly speaking, muscle tissue and most biological materials are anisotropic, but as a simplification one can assume them to be isotropic [28], which is what we will do. The Lamé parameters are assumed to be coupled by a constant τ through the equation $\lambda = \tau\mu$. This reduces the number of parameters to identify to one, but introduces a nonlinear term in the weak form of the system. Our goal is to extend the general results from the previous chapter for dealing with nonlinearly appearing parameters in general variational problems to the specific mixed variational problem of the elasticity system. We will first introduce the elasticity equations in Section 3.1 and derive the weak form of the system. Next we will discuss the output least-squares (OLS) functional in 3.2 followed by computation formulas for first and second order derivatives of the regularized OLS via the adjoint method in Section 3.3 Finally we give discretization details in 3.4 and show numerical results in 4.1.3.

3.1 Elasticity Equations

For the elastography inverse problem we will consider the following equations which describe the displacement of elastic tissue under force.

$$
\begin{aligned}
-\nabla \cdot \sigma &= f \qquad \text{in } \Omega, && (3.1a)\\
\sigma &= 2\mu\, \varepsilon(\bar{u}) + \lambda \operatorname{div} \bar{u}\, I, && (3.1b)\\
\bar{u} &= g \qquad \text{on } \Gamma_1, && (3.1c)\\
\sigma n &= h \qquad \text{on } \Gamma_2. && (3.1d)
\end{aligned}
$$

Here Ω is the domain of the system with the boundary given as $\partial\Omega = \Gamma_1 \cup \Gamma_2$. Usually, the equations are considered for $\Omega \in \mathbb{R}^d$ with $d = 2$ or 3. As such, the function $\bar{u}(x) \in \Omega$ describing the displacement of the tissue is vector valued, with each component accounting for the displacement in one space direction. That is, for $\Omega \subset \mathbb{R}^2$ we have $\bar{u}(x) = \begin{bmatrix} u_1(x) \\ u_2(x) \end{bmatrix}$ where $u_1(x)$ is the displacement along the x-axis and $u_2(x)$ is the displacement along the y-axis. Here, $\varepsilon(\bar{u}) = \frac{1}{2}\left(\nabla\bar{u} + \nabla\bar{u}^T\right)$ is the linearized strain tensor of $\bar{u}$ and $\operatorname{div} \bar{u} = \operatorname{Tr}(\varepsilon(\bar{u}))$. The stress tensor σ is defined by the stress-strain equation (3.1b). The equation is a form of Hooke's law for isotropic materials that holds when the displacement $\bar{u}$ remains small enough so that stress and strain have a linear relationship. In such a case we have a system of so-called linear elasticity.

The force acting on the elastic tissue is governed by the vector valued function f, and boundary conditions are set by the functions g and h. In detail, g describes the Dirichlet boundary conditions for $\bar{u}$ on the boundary Γ_1 and h corresponds to Neumann boundary conditions on the boundary Γ_2. Often times the case of homogeneous Dirichlet boundary conditions where $g = 0$ is used to make calculations simpler.

The parameters that are of interest for the elasticity inverse problem are the Lamé parameters $\lambda(x)$ and $\mu(x)$ which describe the elastic behavior of the tissue. Parameter $\mu(x)$ is also known as the shear modulus of a material. For a Young modulus E and a Poisson's ratio ν, the Lamé parameters are given as [29, 18]

$$
\lambda = \frac{E\nu}{(1+\nu)(1-2\nu)}, \qquad \mu = \frac{E}{2(1+\nu)}. \tag{3.2}
$$

It can be seen that λ does not necessarily have to be positive, but we will consider the case where it is. The elasticity inverse problem consists of finding the values of the Lamé parameters λ and μ, given a measurement z of the solution variable $\bar{u}$.

3.1.1 Near incompressibility

When the Poisson's ratio $\nu \to \frac{1}{2}$, the tissue nears incompressibility and the parameter λ tends to infinity, as can be seen from (3.2). In the case of near incompressibility where λ is still bounded, we nevertheless have

$\lambda \gg \mu$. This nearly incompressible state causes a locking effect [30] when solving the elasticity system via the finite element method that makes simple methods unable to find a solution. The locking effect can be overcome with different techniques, one of them being the mixed finite element approach [31]. The idea is to introduce a pressure variable p as

$$p = \lambda \operatorname{div} \bar{u}. \tag{3.3}$$

This pressure p now represents another unknown in the system, and has to be found together with $\bar{u}$. Note that unlike the displacement $\bar{u}$ which was vector valued, p is a scalar function.

Besides the complications of the locking effect, there are further difficulties involved in properly identifying more than one parameter at the same time. See [32] for a discussion on the identification of both Lamé parameters. Accurately recovering both λ and μ is quite hard to achieve, especially in the near incompressible case where the two parameters can differ by several orders of magnitude. One simplification to the problem is to simply take λ as a very large constant. This reduces the inverse problem to finding only μ. It can be argued that this method is quite artificial and does not have much physical meaning. A different approach [18] is to define a relation constant τ and assume that a linear relationship

$$\lambda(x) = \tau\mu(x) \tag{3.4}$$

holds for the Lamé parameters. We will apply this method and take the relation constant τ to be very large (e.g. 10^5) in order to match the near incompressible case where $\lambda \gg \tau$.

3.1.2 Deriving the Weak Formulation

To derive the weak form of (3.1) we will, for the time being, take $g = 0$ in (3.1c). With homogeneous Dirichlet boundary conditions the space of test functions is then

$$V = \{v \in (H^1(\Omega))^d \mid v = 0 \text{ on } \Gamma_1\} \tag{3.5}$$

where d is the space dimension of the problem. Taking equation (3.1a), we can multiply both sides by a test function $v \in V$ and integrate over the domain Ω.

$$\begin{aligned} -\int_\Omega (\nabla \cdot \sigma) \cdot v &= \int_\Omega f \cdot v \qquad \forall\, v \in V \\ -\int_\Omega (\nabla \cdot \sigma^T) \cdot v + \sigma : (\nabla v^T) &= \int_\Omega f \cdot v \qquad \forall\, v \in V \end{aligned} \tag{3.6}$$

By definition from (3.1b), σ is symmetric so that $\sigma = \sigma^T$. We can thus apply the divergence theorem

$$\int_\Omega \nabla \cdot \sigma = \int_{\partial\Omega} \sigma n. \tag{3.7}$$

Furthermore, note that because σ is symmetric we have

$$\begin{aligned}\sigma : (\nabla v^T) &= \frac{1}{2}\sigma : (\nabla v^T) + \frac{1}{2}\sigma^T : (\nabla v^T) \\ &= \frac{1}{2}\sigma : (\nabla v^T) + \frac{1}{2}\sigma : (\nabla v) \\ &= \sigma : \frac{1}{2}(\nabla v^T + \nabla v) \\ &= \sigma : \varepsilon(v).\end{aligned} \tag{3.8}$$

We can now apply (3.7) and (3.8) to (3.6). After that we can use $\Omega = \Gamma_1 \cup \Gamma_2$ and the boundary constraints (3.1c) and (3.1d) on the boundary integral. Recall that we have set $g = 0$ for now.

$$\begin{aligned}\int_\Omega \sigma : \varepsilon(v) - \int_{\partial\Omega} (\sigma n) \cdot v &= \int_\Omega f \cdot v \\ \int_\Omega \sigma : \varepsilon(v) - \int_{\Gamma_1} (\sigma n) \cdot \underbrace{v}_{=g=0 \text{ on } \Gamma_1} - \int_{\Gamma_2} \underbrace{(\sigma n)}_{=h \text{ on } \Gamma_2} \cdot v &= \int_\Omega f \cdot v \\ \int_\Omega \sigma : \varepsilon(v) - \int_{\Gamma_2} h \cdot v &= \int_\Omega f \cdot v\end{aligned}$$

Replacing σ with its definition from (3.1b) and rearranging we get

$$\begin{aligned}\int_\Omega (2\mu\, \varepsilon(\bar{u}) + \lambda \text{ div } \bar{u}\, I) : \varepsilon(v) &= \int_\Omega f \cdot v + \int_{\Gamma_2} h \cdot v \\ \int_\Omega 2\mu\, \varepsilon(\bar{u}) : \varepsilon(v) + \int_\Omega \lambda \text{ div } \bar{u}\, I : \varepsilon(v) &= \int_\Omega f \cdot v + \int_{\Gamma_2} h \cdot v\end{aligned}$$

Finally, if we use $I : \varepsilon(v) = \text{Tr}(\varepsilon(v)) = \text{ div } v$, we can get the weak form for the linear elasticity system:

$$\int_\Omega 2\mu\, \varepsilon(\bar{u}) : \varepsilon(v) + \int_\Omega \lambda \text{ div } \bar{u} \text{ div } v = \int_\Omega f \cdot v + \int_{\Gamma_2} h \cdot v \qquad \forall\, v \in V \tag{3.9}$$

This weak form by itself is not suited for the inverse problem because of the locking effect discussed earlier. We therefore introduce the pressure variable $p = \lambda \text{ div } \bar{u}$ as defined in (3.3). Let $Q = L^2(\Omega)$ be the space that p lives in. Rearranging the terms of (3.3) and multiplying by a test function $q \in Q$ gives the weak form for the pressure variable.

$$\int_\Omega \frac{1}{\lambda} p\, q = \int_\Omega \text{ div } \bar{u}\, q \qquad \forall\, q \in Q \tag{3.10}$$

The direct problem now consists of two unknowns $\bar{u}$ and p that have to be found jointly. To get the final weak form of the elasticity system we replace $\lambda \text{ div } \bar{u} = p$ in (3.9). Also, recall our assumption that $\lambda = \tau\mu$ for a

large constant τ. With this the weak form reads as follows: Find $(\bar{u}, p) \in V \times Q$ such that

$$\begin{aligned} \int_\Omega 2\mu\, \varepsilon(\bar{u}) : \varepsilon(v) + \int_\Omega p \ \text{div}\ v &= \int_\Omega f \cdot v + \int_{\Gamma_2} h \cdot v \qquad && \forall\, v \in V \\ \int_\Omega \text{div}\ \bar{u}\, q - \int_\Omega \frac{1}{\tau\mu} p\, q &= \ 0 && \forall\, q \in Q \end{aligned} \tag{3.11}$$

In notation of a general mixed variational problem this is equivalent to

$$a(\ell, \bar{u}, v) \ + \ b(v, p) \quad = \ m_1(v) \qquad \forall\, v \in V, \tag{3.12a}$$

$$b(\bar{u}, q) \ - \ c(\ell, p, q) \quad = \ m_2(q) \qquad \forall\, q \in Q, \tag{3.12b}$$

where ℓ is the parameter that corresponds to μ in the elasticity problem and we have the separate terms are given as

$$\begin{aligned} a(\mu, \bar{u}, v) &= \int_\Omega 2\mu\varepsilon(\bar{u}) : \varepsilon(v), \\ b(v, p) &= \int_\Omega p \ \text{div}\ v, \\ b(\bar{u}, q) &= \int_\Omega q \ \text{div}\ \bar{u}, \\ c(\mu, p, q) &= \int_\Omega \frac{1}{\tau\mu} pq, \\ m_1(v) &= \int_\Omega f \cdot v + \int_{\Gamma_2} h \cdot v, \\ m_2(q) &= 0. \end{aligned}$$

In the context of the general mixed variational problem (3.12), define the following spaces. Take V and Q as Hilbert spaces, B as a Banach space and A as a nonempty, closed and convex subset of B. The mappings are assumed to have the following properties. Let the map $a : B \times V \times V \to \mathbb{R}$ be a trilinear map, such that it is linear in each one of its three arguments. Similarly, let $b : V \times Q \to \mathbb{R}$ be a bilinear map. The map $c : B \times Q \times Q \to \mathbb{R}$ is nonlinear in its first argument ℓ but is bilinear and symmetric in the second and third arguments. The nonlinearity of c with respect to the unknown parameter ℓ is the novelty of this inverse problem formulation for the elasticity problem. The right hand side maps $m_1 : V \to \mathbb{R}$ and $m_2 : Q \to \mathbb{R}$ are assumed to be linear and continuous. It is further assumed that c is twice Fréchet differentiable with respect to the first argument. The partial derivative with respect to ℓ is written as $\partial_\ell c(\ell, p, q)$ and is assumed to be linear and symmetric with respect to the second and third arguments. Finally we assume the positive

constants κ_0, κ_1, κ_2, ς_1, and ς_2 exists such that the following coercivity and continuity statements hold.

$$
\begin{aligned}
a(\ell, \bar{v}, \bar{v}) &\geq \kappa_1 \|\bar{v}\|^2, && \forall\, \bar{v} \in V, \quad \forall \ell \in A, && \text{(3.13a)}\\
\|a(\ell, \bar{u}, \bar{v})\| &\leq \kappa_2 \|\ell\| \, \|\bar{u}\| \, \|\bar{v}\|, && \forall\, \bar{u}, \bar{v} \in V, \quad \forall \ell \in B, && \text{(3.13b)}\\
c(\ell, q, q) &\geq \varsigma_1 \|q\|^2, && \forall\, q \in Q, \quad \forall \ell \in A, && \text{(3.13c)}\\
\|c(\ell, p, q)\| &\leq \varsigma_2 \|p\| \, \|q\|, && \forall\, p, q \in Q, \quad \forall \ell \in B, && \text{(3.13d)}\\
\|b(\bar{v}, q)\| &\leq \kappa_0 \|\bar{v}\| \, \|q\|, && \forall\, \bar{v} \in V, \quad \forall q \in Q. && \text{(3.13e)}
\end{aligned}
$$

3.1.3 A Brief Literature Review

In the following we briefly review some of the related work done for simpler problems. In an interesting paper, Lewis [23] discusses the limitations and the applicability of the first-order adjoint method. Domingueza, Gibiatb, and Esquerrea [21] used the adjoint method for computing the topological gradient in work related to ultrasonic target detection. Pingen, Evgrafov and Maute [33] presented an adjoint parameter sensitivity analysis formulation and solution strategy for the lattice Boltzmann method. In another interesting work, Resendiz and Pinnau [24] investigated the optimal control of particle controls in low Reynolds number flows and used the adjoint approach for the derivative information. Knopoff, Fernández, Torres, and Turner [25] applied the adjoint method for a tumor growth PDE-constrained optimization problem. Wang, Yang and Zeng [34] applied the adjoint method for the inverse problem of option pricing. Ye, Li, and Liu [35] developed an exact time-domain second-order adjoint-sensitivity computation for linear circuit analysis. Recently Jensen, Nakshatrala, and Tortorelli [36] studied the consistency of adjoint sensitivity analysis for structural optimization of linear dynamic problems. Kourounis, Durlofsky, Jansen, Aziz [22] employed the adjoint approach for a gradient-based optimization of compositional reservoir flow. Volkov, Protas, Liao, and Glander [37] developed adjoint-based optimization framework for thermo-fluid phenomena in welding processes. Papadimitriou and Giannakoglou [38] used first- and second-order adjoint approach in the context of aerodynamic shape optimization. Boger and Paterson [39] used continuous adjoint approach to design optimization in cavitating flow using a barotropic model. Cioaca, Alexe, and Sandu [19] used second-order adjoint method for solving some PDE-constrained optimization problems (see also [20]). Arens, Rentropa, Stoll, and Wever [40] used the adjoint approach for an optimal design of turbine blades. Kennedy and Hansen [41] studied a hybrid-adjoint approach for a semi-analytic gradient evaluation technique which was applied to composite cure cycle optimization. Liu, Geier, Liu, Krafczyk, and Chen [42] devised a discrete adjoint sensitivity analysis for fluid flow topology optimization based on the generalized lattice Boltzmann method. Lozano [43] embarked on some issues related to discrete adjoint approach in inviscid flow problem. Zanganeh, Kraaijevanger, Buurman, Jansen, and Rossen [44] applied adjoint-based optimization to a surfactant-alternating gas foam process. Altaf, Gharamti, Heemink, and Hoteit [45] applied a reduced

adjoint approach to variational data assimilation. Oberai, Gokhale, and Feijoo [18] used first-order adjoint method for elasticity imaging inverse problem, which is quite similar to ours, but without an explicit use of the mixed variational formulation. An excellent survey article on adjoint methods of first-order and second-order is by Tortorelli and Michaleris [26].

3.2 Inverse Problem Functionals

The goal of the elastography inverse problem is to determine the parameter $\ell \in A$ for which the solution (u, p) of the variational problem (3.12) is as close as possible to a given measurement $(\bar{z}, \hat{z})$ of (u, p). Let $u = u(\ell) = (\bar{u}(\ell), p(\ell)) \in W := V \times Q$ be the computed solution for a given parameter ℓ. The output least-squares (OLS) functional for this problem takes the form

$$J_{OLS}(\ell) = \frac{1}{2} \|u(\ell) - z\|_W^2 = \frac{1}{2} \|\bar{u}(\ell) - \bar{z}\|_V^2 + \frac{1}{2} \|p(\ell) - \hat{z}\|_Q^2 . \tag{3.14}$$

Due to the ill-posed nature of inverse problems this functional will have to be regularized. Take $R(\ell) : H \to \mathbb{R}$ as the regularizer of ℓ given a Hilbert space H and take $\kappa > 0$ as a regularization value. Adding the regularizer to the OLS functional we can formulate the goal of the inverse problem as finding the minimizer of

$$\arg\min_{\ell \in A} J_\kappa(\ell) := \arg\min_{\ell \in A} \frac{1}{2} \|\bar{u}(\ell) - \bar{z}\|_V^2 + \frac{1}{2} \|p(\ell) - \hat{z}\|_Q^2 + \kappa R(\ell). \tag{3.15}$$

We have the following result concerning the solvability of the above optimization problem:

Theorem 3.2.1. *Assume that the Hilbert space H is compactly embedded into the space B, $A \subset H$ is nonempty, closed, and convex, the map R is convex, lower-semicontinuous and there exists $\alpha > 0$ such that $R(\ell) \geq \alpha \|\ell\|_H^2$, for every $\ell \in A$. Then* (3.15) *has a nonempty solution set.*

Proof. Since $J_\kappa(\ell) \geq 0$ for every $\ell \in A$, there exists a minimizing sequence $\{\ell_n\}$ in A such that we have $\lim_{n\to\infty} J_\kappa(\ell_n) = \inf\{J_\kappa(\ell) | \, \ell \in A\}$. This confirms that $\{\ell_n\}$ is bounded in H. Therefore, there exists a subsequence converging weakly in H, and due to the compact embedding of H in B, strongly converging in B. Retaining the same notation for subsequences as well, let ℓ_n converge to some $\hat{\ell} \in A$, where we used the fact that A is closed. For the corresponding $u_n = (\bar{u}_n, p_n)$, we have

$$\begin{aligned} a(\ell_n, \bar{u}_n, \bar{v}) + b(\bar{v}, p_n) &= m_1(\bar{v}), \quad \text{for every } \bar{v} \in V, \\ b(\bar{u}_n, q) - c(\ell_n, p_n, q) &= m_2(q), \quad \text{for every } q \in Q. \end{aligned}$$

The above mixed variational problem confirms that $\{u_n\}$ remains bounded in W and hence there is a subsequence converging weakly to some $\hat{u}$. By manipulating the above mixed variational problem, it can be

shown that $\hat{u} = \hat{u}(\hat{\ell})$. Furthermore, using the coercivity (3.13) of the system terms, it follows that in fact $\{u_n\}$ converges to $\hat{u} = \hat{u}(\hat{\ell})$ strongly.

Finally, using the continuity of the norm, we have

$$\begin{aligned} J_\kappa(\hat{\ell}) &= \frac{1}{2}\|\hat{u}(\hat{\ell}) - z\|^2 + \kappa R(\hat{\ell}) \\ &\leq \lim_{n\to\infty} \frac{1}{2}\|u_n(\ell) - z\|^2 + \liminf_{n\to\infty} \kappa R(\ell_n) \\ &\leq \liminf_{n\to\infty} \left\{ \frac{1}{2}\|u_n(\ell) - z\|^2 + \kappa R(\ell_n) \right\} = \inf \{J_\kappa(\ell) \,:\, \ell \in A\}, \end{aligned}$$

confirming that $\hat{\ell}$ is a solution of (1.11). The proof is complete. □

Remark 3.2.1. A natural choice of spaces and the regularizer involved in the above result is $H = H_2(\Omega)$, $B = L_\infty(\Omega)$ and $R(\ell) = \|\cdot\|^2_{H_2(\Omega)}$. Evidently, this choice is only satisfactory for smooth parameters. However, for discontinuous parameters total variation regularization can be employed and the framework given in [17] easily extends to the case of non-quadratic regularizers.

Theorem 3.2.2. *For each ℓ in the interior of A, $u = u(\ell) = (\bar{u}(\ell), p(\ell))$ is infinitely differentiable at ℓ. The first derivative of u at ℓ in the direction $\delta\ell$, denoted by $\delta u(\ell) = (D\bar{u}(\ell)\delta\ell, Dp(\ell)\delta\ell)$, is the unique solution of the mixed variational problem:*

$$a(\ell, \delta\bar{u}, \bar{v}) + b(\bar{v}, \delta p) = -a(\delta\ell, \bar{u}, \bar{v}), \qquad \forall\, \bar{v} \in V, \tag{3.16a}$$

$$b(\delta\bar{u}, q) - c(\ell, \delta p, q) = \partial_\ell c(\ell, p, q)(\delta\ell), \qquad \forall\, q \in Q. \tag{3.16b}$$

Proof. In [17] an equivalent theorem was given for the case of parameters appearing linearly. The proof for case of nonlinear parameters can be obtained by applying the results from Theorem 1.2.2 to the results in [17]. □

3.3 Derivative Formulae for the Regularized OLS

In this section, our objective is to derive a first-order adjoint method to compute the first-order derivative of the regularized OLS, and a second-order adjoint approach for the computation of its second-order derivative. We briefly mention a direct method for the second-order derivative but note that this method is inferior in efficiency to the second-order adjoint method.

3.3.1 First-Order Adjoint Method

Since the regularized output least-squares functional is given by

$$\begin{aligned} J_\kappa(\ell) &= \frac{1}{2}\left\|\bar{u}(\ell) - \bar{z}\right\|_V^2 + \frac{1}{2}\left\|p(\ell) - \hat{z}\right\|_Q^2 + \kappa R(\ell) \\ &= \frac{1}{2}\left\langle \bar{u} - \bar{z}, \bar{u} - \bar{z}\right\rangle + \frac{1}{2}\left\langle p - \hat{z}, p - \hat{z}\right\rangle + \kappa R(\ell), \end{aligned}$$

it follows, by using the chain rule, that the derivative of J_κ at $\ell \in A$ in any direction $\delta\ell$ is given by

$$DJ_\kappa(\ell)(\delta\ell) = \left\langle D\bar{u}(\ell)(\delta\ell), \bar{u} - \bar{z}\right\rangle + \left\langle Dp(\ell)(\delta\ell), p - \hat{z}\right\rangle + \kappa DR(\ell)(\delta\ell),$$

where $Du(\ell)(\delta\ell) = (D\bar{u}(\ell)(\delta\ell), Dp(\ell)(\delta\ell))$ is the derivative of the parameter-to-solution map u and $DR(\ell)(\delta\ell)$ is the derivative of the regularizer R, both computed at ℓ in the direction $\delta\ell$. Finding the derivative $Du(\ell)(\delta\ell)$ directly is a complicated and expensive process and we want to avoid having to do so. The adjoint method gives a way to compute $DJ_\kappa(\ell)(\delta\ell)$ without needing to find $Du(\ell)(\delta\ell)$.

For an arbitrary $v = (\bar{v}, q) \in W$, we define the functional $L_\kappa : B \times W \to \mathbb{R}$ by

$$L_\kappa(\ell, v) = J_\kappa(\ell) + a(\ell, \bar{u}, \bar{v}) + b(\bar{v}, p) - m_1(\bar{v}) + b(\bar{u}, q) - c(\ell, p, q) - m_2(q),$$

where we added the terms of the mixed variational problem (3.12). We therefore have the equalities

$$L_\kappa(\ell, v) = J_\kappa(\ell) \qquad \forall\, v \in W \tag{3.17}$$

$$\partial_\ell L_\kappa(\ell, v)(\delta\ell) = DJ_\kappa(\ell)(\delta\ell) \qquad \forall\, v \in W, \quad \forall\, \delta\ell. \tag{3.18}$$

The partial derivative of $L_\kappa(\ell, v)$ with respect to ℓ yields

$$\begin{aligned} \partial_\ell L_\kappa(\ell, v)(\delta\ell) = {} & \left\langle D\bar{u}(\ell)(\delta\ell), \bar{u} - \bar{z}\right\rangle + \left\langle Dp(\ell)(\delta\ell), p - \hat{z}\right\rangle + \kappa DR(\ell)(\delta\ell) \\ & + a(\delta\ell, \bar{u}, \bar{v}) + a(\ell, D\bar{u}(\ell)(\delta\ell), \bar{v}) + b(\bar{v}, Dp(\ell)(\delta\ell)) \\ & + b(D\bar{u}(\ell)(\delta\ell), q) - \partial_\ell c(\ell, p, q)(\delta\ell) - c(\ell, Dp(\ell)(\delta\ell), q). \end{aligned} \tag{3.19}$$

The key idea for the first-order adjoint method is to choose v to bypass a direct computation of $\delta u = Du(\ell)(\delta\ell)$. To achieve this, fix $\ell \in A$ and let $w(\ell) = (\bar{w}(\ell), p_w(\ell))$ be the unique solution of the mixed variational problem

$$a(\ell, \bar{w}, \bar{v}) + b(\bar{v}, p_w) = \left\langle \bar{z} - \bar{u}, \bar{v}\right\rangle, \qquad \forall\, \bar{v} \in V, \tag{3.20a}$$

$$b(\bar{w}, q) - c(\ell, p_w, q) = \left\langle \hat{z} - p, q\right\rangle, \qquad \forall\, q \in Q, \tag{3.20b}$$

where the right-hand sides of (3.20a) and (3.20b) involve the solution $u = (\bar{u}, p)$ of (3.12) and the measured data $z = (\bar{z}, \hat{z})$.

Note that if we let $v = (\bar{v}, q) = (D\bar{u}(\ell)(\delta\ell), Dp(\ell)(\delta\ell))$ in (3.20) we get

$$\begin{aligned} a(\ell, D\bar{u}(\ell)(\delta\ell), \bar{w}) + b(D\bar{u}(\ell)(\delta\ell), p_w) &= -\langle \bar{u} - \bar{z}, D\bar{u}(\ell)(\delta\ell)\rangle, \\ b(\bar{w}, Dp(\ell)(\delta\ell)) - c(\ell, Dp(\ell)(\delta\ell), p_w) &= -\langle p - \hat{z}, Dp(\ell)(\delta\ell)\rangle. \end{aligned}$$

It is easy to see that these are exactly the terms that appear and cancel out if we let $v = w$ in (3.19). The remaining terms then amount to

$$\partial_\ell L_\kappa(\ell, w)(\delta\ell) = \kappa DR(\ell)(\delta\ell) + a(\delta\ell, \bar{u}, \bar{w}) - \partial_\ell c(\ell, p, p_w)(\delta\ell).$$

By equality (3.18) of the functionals we thus have an expression for the OLS functional derivative that does not require the derivative of the parameter-to-solution map u:

$$DJ_\kappa(\ell)(\delta\ell) = \kappa DR(\ell)(\delta\ell) + a(\delta\ell, \bar{u}, \bar{w}) - \partial_\ell c(\ell, p, p_w)(\delta\ell). \tag{3.21}$$

Summarizing, the following scheme computes $DJ_\kappa(\ell)(\delta\ell)$ for the given direction $\delta\ell$:

Step 1. Compute $u(\ell) = (\bar{u}(\ell), p(\ell))$ by using (3.12).

Step 2. Compute $w(\ell) = (\bar{w}(\ell), p_w(\ell))$ by using (3.20).

Step 3. Compute $DJ_\kappa(\ell)(\delta\ell)$ by using (3.21).

3.3.2 A Direct Method for the Second-Order Derivative

The direct approach to find the second-order derivative $D^2J_\kappa(\ell)(\delta\ell_1, \delta\ell_2)$ is to take expression (3.21) for the first derivative in direction $\delta\ell_1$ and differentiate it in another direction $\delta\ell_2$. This procedure is straightforward, but leads to derivative expressions of both u and w.

$$\begin{aligned} D^2J_\kappa(\ell)(\delta\ell_1, \delta\ell_2) = {} & a(\delta\ell_1, D\bar{u}(\ell)(\delta\ell_2), \bar{w}) + a(\delta\ell_1, \bar{u}, D\bar{w}(\ell)(\delta\ell_2)) \\ & - \partial_\ell c(\ell, Dp(\ell)(\delta\ell_2), p_w)(\delta\ell_1) - \partial_\ell c(\ell, p, Dp_w(\ell)(\delta\ell_2))(\delta\ell_1) \\ & - \partial_\ell^2 c(\ell, p, p_w)(\delta\ell_1, \delta\ell_2) + \kappa D^2R(\ell)(\delta\ell_1, \delta\ell_2) \end{aligned} \tag{3.22}$$

For a given direction $\delta\ell$ the derivative $\delta u = (\delta\bar{u}, \delta p) = (D\bar{u}(\ell)(\delta\ell), Dp(\ell)(\delta\ell))$ can be computed by solving (3.16). The adjoint variable $w = (\bar{w}, p_w)$ is the solution to system (3.20). Differentiating in direction $\delta\ell$ gives a way to find $\delta w = (\delta\bar{w}, \delta p_w) = (D\bar{w}(\ell)(\delta\ell), Dp_w(\ell)(\delta\ell))$ as the solution of

$$\begin{aligned} a(\delta\ell, \bar{w}, \bar{v}) + a(\ell, \delta\bar{w}, \bar{v}) + b(\bar{v}, \delta p_w) &= -\langle \delta\bar{u}, \bar{v}\rangle, && \forall\ \bar{v} \in V \\ b(\delta\bar{w}, q) - \partial_\ell c(\ell, p_w, q) - c(\ell, \delta p_w, q) &= -\langle \delta p_w, q\rangle, && \forall\ q \in Q \end{aligned} \tag{3.23}$$

The steps for a direct computation of the second-order OLS derivative are therefore as follows.

Step 1. Compute $u(\ell) = (\bar{u}(\ell), p(\ell))$ by (3.12).

Step 2. Compute $\delta u = (\delta\bar{u}, \delta p)$ by (3.16).

Step 3. Compute $w(\ell) = (\bar{w}(\ell), p_w(\ell))$ by (3.20).

Step 4. Compute $\delta w = (\delta\bar{w}, \delta p_w)$ by (3.23).

Step 5. Compute $D^2 J_\kappa(\ell)(\delta\ell, \delta\ell)$ by (3.22).

It should be noted again that this method is very slow for large problems, due to the need of finding the derivative of both the solution u and the adjoint variable w. In practice, this method does not have much merit, and the second-order adjoint method presented next should be used instead.

3.3.3 Second-Order Adjoint Method

We now give a second-order adjoint method for the computation of the second-order derivative of the regularized OLS functional. The objective is to give a formula for the second-order derivative that does not involve the second-order derivative of the parameter-to-solution map u. The key idea is to compute δu directly by using Theorem 3.2.2 while the computation of $\delta^2 u$ is avoided by using an adjoint approach. Furthermore, this approach does not involve any derivative expressions of the adjoint variable w. Recall that

$$\begin{aligned} J_\kappa(\ell) &= \frac{1}{2}\langle \bar{u} - \bar{z}, \bar{u} - \bar{z}\rangle + \frac{1}{2}\langle p - \hat{z}, p - \hat{z}\rangle + \kappa R(\ell), \\ DJ_\kappa(\ell)(\delta\ell) &= \langle D\bar{u}(\ell)(\delta\ell), \bar{u} - \bar{z}\rangle + \langle Dp(\ell)(\delta\ell), p - \hat{z}\rangle + \kappa DR(\ell)(\delta\ell). \end{aligned}$$

Next, consider the derivative of the variational problem (3.12) with respect to ℓ in direction $\delta\ell_2$.

$$\begin{aligned} a(\delta\ell_2, \bar{u}, \bar{v}) + a(\ell, D\bar{u}(\ell)(\delta\ell_2), \bar{v}) + b(\bar{v}, Dp(\ell)(\delta\ell_2)) = 0, \qquad & \forall\, v \in V \\ b(D\bar{u}(\ell)(\delta\ell_2), q) - \partial_\ell c(\ell, p, q)(\delta\ell_2) - c(\ell, Dp(\ell)(\delta\ell_2), q) = 0, \qquad & \forall\, q \in Q \end{aligned}$$

Given a fixed direction $\delta\ell_2$ and an arbitrary $v = (\bar{v}, q) \in W$, we define a new functional by adding the above derivative terms to the derivative of the OLS functional.

$$\begin{aligned} L_\kappa(\ell, v) = DJ_\kappa(\ell)(\delta\ell_2) &+ a(\delta\ell_2, \bar{u}, \bar{v}) + a(\ell, D\bar{u}(\ell)(\delta\ell_2), \bar{v}) + b(\bar{v}, Dp(\ell)(\delta\ell_2)) \\ &+ b(D\bar{u}(\ell)(\delta\ell_2), q) - \partial_\ell c(\ell, p, q)(\delta\ell_2) - c(\ell, Dp(\ell)(\delta\ell_2), q) \end{aligned}$$

Evidently, by the definition of L_κ, for every $v \in W$, and any direction $\delta\ell_1$, we have

$$\partial_\ell L_\kappa(\ell, v)(\delta\ell_1) = D^2 J_\kappa(\ell)(\delta\ell_1, \delta\ell_2). \tag{3.24}$$

Computing the derivative of L_κ in the direction $\delta\ell_1$ directly, we have

$$\begin{aligned}\partial_\ell L_\kappa(\ell, v)(\delta\ell_1) = &\left\langle D^2\bar{u}(\ell)(\delta\ell_1, \delta\ell_2), \bar{u} - \bar{z}\right\rangle + \langle D\bar{u}(\ell)(\delta\ell_2), D\bar{u}(\ell)(\delta\ell_1)\rangle \\ &+ \left\langle D^2 p(\ell)(\delta\ell_1, \delta\ell_2), p - \hat{z}\right\rangle + \langle Dp(\ell)(\delta\ell_2), Dp(\ell)(\delta\ell_1)\rangle \\ &+ \kappa D^2 R(\ell)(\delta\ell_1, \delta\ell_2) + a(\delta\ell_2, D\bar{u}(\ell)(\delta\ell_1), \bar{v}) \\ &+ a(\delta\ell_1, D\bar{u}(\ell)(\delta\ell_2), \bar{v}) + a(\ell, D^2\bar{u}(\ell)(\delta\ell_1, \delta\ell_2), \bar{v}) \\ &+ b(\bar{v}, D^2 p(\ell)(\delta\ell_1, \delta\ell_2)) + b(D^2\bar{u}(\ell)(\delta\ell_1, \delta\ell_2), q) \\ &- \partial_\ell^2 c(\ell, p, q)(\delta\ell_1, \delta\ell_2) - \partial_\ell c(\ell, Dp(\ell)(\delta\ell_1), q)(\delta\ell_2) \\ &- \partial_\ell c(\ell, Dp(\ell)(\delta\ell_2), q)(\delta\ell_1) - c(\ell, D^2 p(\ell)(\delta\ell_1, \delta\ell_2), q).\end{aligned} \tag{3.25}$$

Introduce the adjoint variable $w(\ell) = (\bar{w}(\ell), p_w(\ell))$ and let it be the solution of the mixed variational problem (3.20), that is,

$$\begin{aligned} a(\ell, \bar{w}, \bar{v}) + b(\bar{v}, p_w) &= \langle \bar{z} - \bar{u}, \bar{v}\rangle, \qquad \forall\ \bar{v} \in V, \\ b(\bar{w}, q) - c(\ell, p_w, q) &= \langle \hat{z} - p, q\rangle, \qquad \forall\ q \in Q. \end{aligned}$$

Taking the choice $v = (\bar{v}, q) = (D^2\bar{u}(\ell)(\delta\ell_1, \delta\ell_2), D^2 p(\ell)(\delta\ell_1, \delta\ell_2))$ in (3.20) gives

$$\begin{aligned} a(\ell, D^2\bar{u}(\ell)(\delta\ell_1, \delta\ell_2), \bar{w}) + b(D^2\bar{u}(\ell)(\delta\ell_1, \delta\ell_2), p_w) &= -\left\langle \bar{u} - \bar{z}, D^2\bar{u}(\ell)(\delta\ell_1, \delta\ell_2)\right\rangle, \\ b(\bar{w}, D^2 p(\ell)(\delta\ell_1, \delta\ell_2)) - c(\ell, D^2 p(\ell)(\delta\ell_1, \delta\ell_2), p_w) &= -\left\langle p - \hat{z}, D^2 p(\ell)(\delta\ell_1, \delta\ell_2)\right\rangle. \end{aligned}$$

The above equalities let us cancel exactly all the unwanted second-order terms if we let $v = w$ in (3.25). The remaining terms are:

$$\begin{aligned}\partial_\ell L_\kappa(\ell, v)(\delta\ell_1) = &\kappa D^2 R(\ell)(\delta\ell_1, \delta\ell_2) + \langle D\bar{u}(\ell)(\delta\ell_2), D\bar{u}(\ell)(\delta\ell_1)\rangle + \langle Dp(\ell)(\delta\ell_2), Dp(\ell)(\delta\ell_1)\rangle \\ &+ a(\delta\ell_2, D\bar{u}(\ell)(\delta\ell_1), \bar{w}) + a(\delta\ell_1, D\bar{u}(\ell)(\delta\ell_2), \bar{w}) \\ &- \partial_\ell c(\ell, Dp(\ell)(\delta\ell_1), p_w)(\delta\ell_2) - \partial_\ell c(\ell, Dp(\ell)(\delta\ell_2), p_w)(\delta\ell_1) - \partial_\ell^2 c(\ell, p, p_w)(\delta\ell_1, \delta\ell_2).\end{aligned}$$

By (3.24) the above expression is directly equal to the second-order derivative of the OLS functional $D^2 J_\kappa(\ell)(\delta\ell_1, \delta\ell_2)$. Note that we have no appearance of second-order derivatives of u. Furthermore, there are no derivatives of w altogether, unlike in the direct method for the second-order OLS derivative.

Finally, consider a derivative direction $\delta\ell$ and call $\delta u = (\delta\bar{u}, \delta p) = (D\bar{u}(\ell)(\delta\ell), Dp(\ell)(\delta\ell))$ and similarly $\delta w = (\delta\bar{w}, \delta p_w) = (D\bar{w}(\ell)(\delta\ell), Dp_w(\ell)(\delta\ell))$. Then the second-order derivative $D^2 J_\kappa(\ell)(\delta\ell, \delta\ell)$ at a parameter ℓ in direction $\delta\ell$ is given by

$$\begin{aligned} D^2 J_\kappa(\ell)(\delta\ell, \delta\ell) = &\kappa D^2 R(\ell)(\delta\ell, \delta\ell) + \langle \delta\bar{u}, \delta\bar{u}\rangle + \langle \delta p, \delta p\rangle \\ &+ 2a(\delta\ell, \delta\bar{u}, \bar{w}) - 2\partial_\ell c(\ell, \delta p, p_w)(\delta\ell) - \partial_\ell^2 c(\ell, p, p_w)(\delta\ell, \delta\ell). \end{aligned} \tag{3.26}$$

Summarizing, the following scheme computes the derivative $D^2 J_\kappa(\ell)(\delta\ell, \delta\ell)$ for a given direction $\delta\ell$:

Step 1. Compute $u(\ell) = (\bar{u}(\ell), p(\ell))$ by (3.12).

Step 2. Compute $\delta u = (\delta\bar{u}, \delta p)$ by (3.16).

Step 3. Compute $w(\ell) = (\bar{w}(\ell), p_w(\ell))$ by (3.20).

Step 4. Compute $D^2 J_\kappa(\ell)(\delta\ell, \delta\ell)$ by (3.26).

3.4 Finite Element Discretization

The variational problem (3.12) has three function spaces that need to be discretized: the solution space V of displacements, the pressure space Q, and the parameter space A. Note also that the displacement function $\bar{u}$ is vector valued, which means that we will have to discretize the displacements in each spatial dimension. Let $\mathcal{T}$ and $\mathcal{T}_L$ be triangulations on the domain Ω. Let the V_h and Q_h be defined as the spaces of piecewise continuous polynomials on $\mathcal{T}$ of degree d_u and d_p respectively. Similarly, let W_h be the space of piecewise continuous polynomials of degree d_ℓ relative to $\mathcal{T}_L$. Take the basis for spaces V_h, Q_h, and W_h to be $\{\varphi_1, \varphi_2, \ldots, \varphi_n\}$, $\{\psi_1, \psi_2, \ldots, \psi_m\}$, and $\{\chi_1, \chi_2, \ldots, \chi_k\}$ respectively. For a given function ℓ_h define $L \in \mathbb{R}^k$ as the vector of coefficients L_i such that $\ell_h = \sum_{i=1}^{k} L_i\chi_i$ and $L_i = \ell_h(x_i)$ for $i \leq i \leq k$ where the points x_i are the support points for the basis functions χ_i. That is, ℓ_h is the unique representation of the discrete parameter a_h as a linear combination of basis functions χ_i. Likewise, let P be the unique representation of pressure function p_h by taking $P_i = p_h(x_i)$ so that $1 \leq i \leq m$ and $p_h = \sum_{i=1}^{m} P_i\psi_i$. For the vector valued displacement function $\bar{u}_h$, let $\bar{U}^d$ be the representation of the displacement in a single space dimension as $\bar{U}_i^d = \bar{u}_{hd}(x_i)$ and $\bar{u}_{hd} = \sum_{i=1}^{m} \bar{U}_i^d\varphi_i$. The discretization of the complete displacement is then the concatenation of the individual displacements. That is $\bar{U} = [\bar{U}^1, \bar{U}^2]^T$ in 2D or $\bar{U} = [\bar{U}^1, \bar{U}^2, \bar{U}^3]^T$ in 3D. Let n be the size of $\bar{U}$, then for d space dimensions we get $n = d\,m$. The above gives an ordering of $\bar{U}$ where all displacements in one space dimension follow all the displacements of anther space dimension. An alternative is to interleave the displacements at each support point x_i. The different orderings of the vector elements in $\bar{U}$ all result in the same system but give different sparsity pattern of the matrices in the system. Depending on the solution strategy, one ordering may be more appropriate than the other. For the discretization details we will not assume any specific ordering of $\bar{U}$ and treat it as a single representation of all displacements.

The discrete mixed variational problem seeks, for each ℓ_h, the unique $(\bar{u}_h, p_h) \in V_h \times Q_h$ such that

$$a(\ell_h, \bar{u}_h, \bar{v}) + b(\bar{v}, p_h) = m_1(\bar{v}), \quad \text{for every } \bar{v} \in V_h, \tag{3.27a}$$

$$b(\bar{u}_h, q) - c(\ell_h, p_h, q) = m_2(q), \quad \text{for every } q \in Q_h. \tag{3.27b}$$

We define $S : \mathbb{R}^k \to \mathbb{R}^{n+m}$ to be the finite element solution operator that assigns to each $\ell_h \in W$, the unique discrete solution $u_h = (\bar{u}_h, p_h) \in V_h \times Q_h$. Then $S(\ell_h) = U$, where $U = [\bar{U}, P]^T$ is given by

$$\begin{bmatrix} \widehat{K}_{n\times n}(\ell_h) & B^{\mathrm{T}}_{n\times m} \\ B_{m\times n} & -C_{m\times m}(\ell_h) \end{bmatrix} \begin{bmatrix} \bar{U} \\ P \end{bmatrix} = F, \tag{3.28}$$

where

$$\begin{aligned} \widehat{K}(\ell_h)_{i,j} &= a(\ell_h, \varphi_i, \varphi_j), && i,j = 1,2,\ldots,n, \\ B_{i,j} &= b(\varphi_j, \psi_i), && i = 1,2,\ldots,m, \quad j = 1,2,\ldots,n \\ C(\ell_h)_{i,j} &= c(\ell_h, \psi_i, \psi_j), && i,j = 1,2,\ldots,m, \\ F_i &= m_1(\varphi_i), && i = 1,2,\ldots,n, \\ F_j &= 0, && j = n+1, n+2, \ldots, n+m. \end{aligned}$$

Since a is linear in its first argument, we can compute the matrix $\widehat{K}$ as follows by splitting the parameter variable into its finite elements.

$$\widehat{K}(\ell_h)_{i,j} = a(\ell_h, \varphi_i, \varphi_j) = a(\sum_{t=1}^{k} L_t \chi_t, \varphi_i, \varphi_j). = \sum_{t=1}^{k} L_t a(\chi_t, \varphi_i, \varphi_j)$$

The map c is nonlinear in its first argument, so it is not possible to linearize out the summation of basis functions of ℓ_h. Instead we will have to precompute the values $\ell_h(x_q) = \sum_{t=1}^{k} L\chi(x_q)$ for each quadrature point x_q, and use the resulting $\ell_h(x_q)$ in the quadrature of $c(\ell_h, \psi_j, \psi_i)$. Compare Section 2.3.3 for the corresponding result of evaluating terms with nonlinear parameters in the scalar problem.

Computing the solution U for a given parameter ℓ_h then constitutes of building the matrices outlined above and solving system (3.28) for $[\bar{U}, P]^T$. While it is theoretically possible to solve the entire system at once, the large block matrix usually has bad properties that make it unsuitable for iterative solvers. Instead, we will solve the system using block elimination. In detail, our system has the two equations

$$\begin{aligned} \widehat{K}\bar{U} + B^T P &= F, \\ B\bar{U} - CP &= 0. \end{aligned}$$

Premultiplying the first equation by $B\widehat{K}^{-1}$ and subtracting the second equation gives

$$B\widehat{K}^{-1}B^T P + CP = B\widehat{K}^{-1}F.$$

This equation has P as the only unknown, so we can solve for it as a first step. In the second step we can solve the first equation for $\bar{U}$. Thus, the equations being solved are the following:

$$\left(B\widehat{K}^{-1}B^T + C\right) P = B\widehat{K}^{-1}F, \tag{3.29a}$$

$$\widehat{K}\bar{U} = F - B^T P \tag{3.29b}$$

Here, $B\widehat{K}^{-1}B^T + C$ is the Schur complement of the matrix block $\widehat{K}$.

Recall that the regularized partial OLS functional is given by

$$J_\kappa(\ell) = \frac{1}{2}\left\|\bar{u}(\ell) - \bar{z}\right\|_V^2 + \kappa R(\ell),$$

where $\bar{z}$ is the measured data and $u(\ell) = (\bar{u}(\ell), p(\ell))$ solves (3.12). The discrete analogue of the above functional is given by

$$J_\kappa(\ell_h) = \frac{1}{2}(\bar{U} - \bar{Z})^T M(\bar{U} - \bar{Z}) + \kappa R(\ell_h), \tag{3.30}$$

where U solves the linear system (3.28) and $M \in \mathbb{R}^{n\times n}$ is the mass matrix given by $M_{i,j} = \langle \varphi_i, \varphi_j \rangle$. The regularizer term is computed equivalently to (2.23). With this we now have the tools to solve the variational problem (forward problem) and evaluate the OLS functional for a given parameter choice ℓ_h in the discrete setting. The next sections will deal with the derivative computation methods for the OLS functional.

3.4.1 Discrete Derivative Forms

To compute the first and second derivatives of the OLS functional, we need to construct the discrete forms of terms that include derivatives of some variables. We will show discrete versions of these terms in this section and then apply them to find the OLS derivative expressions in the next sections.

Consider first the regular stiffness matrix $\widehat{K}(\ell_h)$ given as

$$\widehat{K}(\ell_h)_{i,j} = a(\ell_h, \varphi_i, \varphi_j).$$

Now, define the directional stiffness matrix $\widehat{K}^t(\ell_h)$ in direction L_t, that is in the t-th derivative direction of ℓ_h. Due to the fact that the map a, which leads to $\widehat{K}$, is linear in it first argument, we can use the analogous result from (2.24) to get $\partial_t \ell_h = \chi_t$ and therefore

$$\widehat{K}^t(\ell_h) = a(\partial_t \ell_h, \varphi_i, \varphi_j) = a(\chi_t, \varphi_i, \varphi_j). \tag{3.31}$$

Furthermore, the linearity allows us to consider the alternative representation via the so-called adjoint stiffness matrix $\mathbb{A}$ which is defined by the following condition

$$\widehat{K}(L)\bar{V} = \mathbb{A}(\bar{V})L, \quad \forall\, L \in \mathbb{R}^k,\ \forall\, \bar{V} \in \mathbb{R}^n. \tag{3.32}$$

The adjoint stiffness expression is only valid for the special case when the parameter appears linearly. In the elasticity problem this is given for the displacement term $\widehat{K}(L)$. The derivative result in derivative direction δL_t using the adjoint stiffness matrix then becomes

$$\widehat{K}^t(L)\bar{V} = \mathbb{A}(\bar{V})\delta L_t, \tag{3.33}$$

That is, if we use the adjoint stiffness matrix approach for the mapping a then we don't have to compute the matrix $\widehat{K}^t(\ell_h)$ at all and can instead just reuse the adjoint stiffness matrix $\mathbb{A}$. The adjoint stiffness method is thus very efficient, but the approach works only for linearly appearing parameters such as in the map a.

The mapping c, on the other hand, is nonlinear in ℓ since we have $c(\ell, p, p_w) = \int_\Omega \frac{1}{\tau\ell} pq$. The partial derivative of this map with respect to ℓ is then

$$\partial_\ell c(\ell, p, p_w) = \int_\Omega -\frac{1}{\tau\ell^2} pq.$$

We have in a sense an explicit case $g(\ell) = \frac{1}{\ell}$ of the parameter map used in Chapter 2. Hence we get a similar result to (2.25), so that the t-th partial derivative for our specific form becomes

$$\partial_t \frac{1}{\ell_h} = \frac{\partial}{\partial L_t}\left(\frac{1}{\ell_h}\right) = \frac{\partial}{\partial \ell_h}\left(\frac{1}{\ell_h}\right)\frac{\partial \ell_h}{\partial L_t} = -\frac{1}{\ell_h^2}\chi_t. \tag{3.34}$$

Thus, in the discrete setting we define $C^t(\ell_h)$ in the t-th direction as

$$[C^t(\ell_h)]_{i,j} = \partial_t c(\ell_h, \psi_i, \psi_j) = \int_\Omega -\frac{1}{\tau\ell_h^2}\psi_i\psi_j\chi_t, \tag{3.35}$$

where again we want to precompute the values for ℓ_h at all quadrature points since it appears nonlinearly. For the second derivative we have the analogous result to (2.36) and define $C^{tt}(\ell_h)$ in the t-th direction as

$$[C^{tt}(\ell_h)]_{i,j} = \partial^2_{t,t} c(\ell_h, \psi_i, \psi_j) = \int_\Omega \frac{2}{\tau\ell_h^3}\psi_i\psi_j\chi_t\chi_t, \tag{3.36}$$

Essentially, the matrices C and its derivative versions are just explicit instantiations of the general results presented in Chapter 2.

Consider next the derivative of the solution variable $u(\ell)$, that is $\delta u(\ell) = (\delta\bar{u}(\ell), \delta p(\ell))$, which by Theorem 3.2.2, solves the following mixed variational problem

$$\begin{aligned} a(\ell, \delta\bar{u}, \bar{v}) + b(\bar{v}, \delta p) &= -a(\delta\ell, \bar{u}, \bar{v}), \qquad \text{for every } \bar{v} \in V, \\ b(\delta\bar{u}, q) - c(\ell, \delta p, q) &= \partial_\ell c(\ell, p, q)(\delta\ell), \quad \text{for every } q \in Q. \end{aligned}$$

By standard arguments it can be shown that the discrete version of the above system to find $\nabla_t U$ for some ℓ_h in the t-th derivative direction has the form

$$\begin{bmatrix} \widehat{K}(\ell_h) & B^T \\ B & -C(\ell_h) \end{bmatrix} \begin{bmatrix} \nabla_t \bar{U} \\ \nabla_t P \end{bmatrix} = \begin{bmatrix} -\widehat{K}^t(\ell_h)\bar{U} \\ C^t(\ell_h)P \end{bmatrix}. \tag{3.37}$$

The block matrix on the left is identical to the one in (3.28). The matrix $C^t(\ell_h)$ is as defined in (3.35) and $\widehat{K}^t(\ell_h)$ is given in (3.31). The adjoint stiffness matrix approach (3.32) is also possible for $\widehat{K}^t(\ell_h)$. Solving the above system gives the discrete solution $\nabla_t U = (\nabla_t \bar{U}, \nabla_t P)$.

The discrete version of the adjoint variable w, given as $W = (\bar{W}, P_w)$, can be computed by solving the system

$$\begin{bmatrix} \widehat{K}(L) & B^{\mathrm{T}} \\ B & -C(L) \end{bmatrix} \begin{bmatrix} \bar{W} \\ P_w \end{bmatrix} = \begin{bmatrix} \bar{Z} - \bar{U} \\ \hat{Z} - P \end{bmatrix}. \tag{3.38}$$

that is the discrete formulation of the adjoint variational problem (3.20). This linear system is analogous to system (3.28) for finding U, but with a different right hand side. The system can therefore be solved using a similar block elimination method as (3.29).

We also want to compute the discrete derivative of the adjoint variable w. The variational problem to find $D\bar{w}(\ell)(\delta\ell)$ was given in (3.23) as

$$\begin{aligned} a(\delta\ell, \bar{w}, \bar{v}) + a(\ell, \delta\bar{w}, \bar{v}) + b(\bar{v}, \delta p_w) &= -\langle \delta\bar{u}, \bar{v} \rangle, \qquad \forall\, \bar{v} \in V \\ b(\delta\bar{w}, q) - \partial_\ell c(\ell, p_w, q) - c(\ell, \delta p_w, q) &= -\langle \delta p_w, q \rangle, \qquad \forall\, q \in Q \end{aligned}$$

The discrete counterpart $\nabla_t W = (\nabla_t \bar{W}, \nabla_t P_w)$ for a parameter ℓ_h in the t-th derivative direction is given by solving the following linear system

$$\begin{bmatrix} \widehat{K}(\ell_h) & B^T \\ B & -C(\ell_h) \end{bmatrix} \begin{bmatrix} \nabla_t \bar{W} \\ \nabla_t P_w \end{bmatrix} = \begin{bmatrix} -\widehat{K}^t(\ell_h)\bar{W} - M(\nabla_t \bar{U}) \\ C^t(\ell_h) P_w - M_P(\nabla_t P) \end{bmatrix}. \tag{3.39}$$

Here $M_{i,j} = \langle \varphi_i, \varphi_j \rangle$ and $[M_P]_{i,j} = \langle \psi_i, \psi_j \rangle$ while the remaining matrices are identical to those in (3.37). Note that we need to compute $\nabla_t U$ before we can find $\nabla_t W$.

Finally we need to compute the regularizer terms in its derivatives. This is analogous to the scalar problem in Chapter 2 and the gradient values $\nabla R(\ell_h)$ are identical to (2.28). The second derivatives $\nabla^2 R(\ell_h)$ are as in (2.33).

We now have all the necessary discrete forms and move on to the gradient computation methods.

3.4.2 Gradient Computation by a Direct Approach

Out of the different gradient computation methods, the direct approach is the most natural and straightforward, but also the least efficient. It requires the solution of variational problems for each derivative direction. Solving the forward problem is an expensive process and doing so once for all k derivative directions ℓ_h quickly becomes infeasible for larger problems. Other methods, such as the adjoint method which is presented next, avoid the need to solve a variational problem for each derivative direction and are therefore much faster. We cover the direct approach for completeness.

The first-order derivative of the regularized partial OLS functional

$$DJ_\kappa(\ell)(\delta\ell) = \langle \delta\bar{u}, \bar{u} - \bar{z} \rangle + DR(\ell)(\delta\ell), \tag{3.40}$$

involves $\delta u(\ell) = (\delta\bar{u}(\ell), \delta p(\ell))$. The construction of the discrete analogue $\nabla_t U$ is given in (3.37). A discretization of formula (3.40) is then given by

$$\partial_t J_\kappa(\ell_h) = \left\langle \nabla_t \bar{U}, \bar{U} - \bar{Z} \right\rangle + \partial_t R(\ell_h) = \left(\bar{U} - \bar{Z}\right)^T M(\nabla_t \bar{U}) + \partial_t R(\ell_h).$$

The gradient $\nabla U = [\nabla_1 U, \ldots, \nabla_k U]^T \in \mathbb{R}^{(n+m)\times k}$ is computed by solving k equations (3.37) in the directions $1 \leq t \leq k$. This leads to the the following expression for the OLS gradient:

$$\nabla J_\kappa(\ell_h) = \left(\bar{U} - \bar{Z}\right)^{\mathrm{T}} M \nabla \bar{U} + \nabla R(\ell_h). \tag{3.41}$$

Summarizing, the computation of the OLS gradient using the direct approach involves the following steps:

Step 1. Compute $U = (\bar{U}, P)$ by solving linear system (3.29).

Step 2. Compute $\nabla \bar{U}$ by solving k linear systems (3.37).

Step 3. Compute $\nabla J_\kappa(\ell_h)$ by using formula (3.41).

This direct method has a severe inefficiency from the need of computing $\nabla \bar{U}$, which requires the solution of system (3.37) in k directions. For large problem, both the linear system itself and k increase in size and the computational effort needed to solve these systems quickly becomes unfeasible.

3.4.3 Gradient Computation by the First-Order Adjoint Method

We shall now give a scheme for computing the OLS gradient using the first-order adjoint approach. Recall that the first-order adjoint approach led to the following formula for the first-order derivative (see (3.21))

$$DJ_{OLS}(\ell)(\delta\ell) = \kappa DR(\ell)(\delta\ell) + a(\delta\ell, \bar{u}, \bar{w}) - \partial_\ell c(\ell, p, p_w)(\delta\ell). \tag{3.42}$$

where $u = (\bar{u}, p)$ and $w = (\bar{w}, p_w)$ are the solutions of (3.12) and (3.20), respectively. The discrete counterparts of these elements are $U = (\bar{U}, P)$, which solves (3.28), and $W = (\bar{W}, P_w)$, which solves (3.38).

To discretized version of the OLS derivative (3.42) is hence

$$\partial_t J_\kappa(\ell_h)(\delta L_t) = \partial_t R(\ell_h) + \bar{U}^T \widehat{K}^t(\ell_h) \bar{W} - P^T C^t(\ell_h) P_w. \tag{3.43}$$

The derivative of the regularizer $\nabla R(L)$ is found identically as in the previous chapter; see (2.28) for the computations in different norms. Evaluating the above expression in all derivative directions $1 \leq t \leq k$ gives the elements of the gradient $\nabla J_\kappa(\ell_h)$ which is then formed as

$$\nabla J_\kappa(\ell_h) = \left[\partial_1 J_\kappa(\ell_h),\ \partial_2 J_\kappa(\ell_h),\ \ldots,\ \partial_k J_\kappa(\ell_h)\right]^T.$$

We therefore have the following scheme for the derivative computation:

Step 1. Compute $U = (\bar{U}, P)$ by solving equations (3.29).

Step 2. Compute $W = (\bar{W}, P_w)$ by solving linear system (3.38).

Step 3. Compute $\nabla J(\ell_h)$ by finding $\partial_t J_\kappa(\ell_h)$ in all directions $1 \leq t \leq k$ using formula (3.43).

3.4.4 Computation of the Hessian by the Direct Approach

The second-order derivative of the OLS by the direct approach is given in (3.22) as

$$\begin{aligned} D^2 J_\kappa(\ell)(\delta\ell_1, \delta\ell_2) = {} & a(\delta\ell_1, D\bar{u}(\ell)(\delta\ell_2), \bar{w}) + a(\delta\ell_1, \bar{u}, D\bar{w}(\ell)(\delta\ell_2)) \\ & - \partial_\ell c(\ell, Dp(\ell)(\delta\ell_2), p_w)(\delta\ell_1) - \partial_\ell c(\ell, p, Dp_w(\ell)(\delta\ell_2))(\delta\ell_1) \\ & - \partial_\ell^2 c(\ell, p, p_w)(\delta\ell_1, \delta\ell_2) + \kappa D^2 R(\ell)(\delta\ell_1, \delta\ell_2) \end{aligned}$$

This expression involves both derivatives of $u(\ell)$ and $w(\ell)$. In the discrete setting, the t-th second-order partial derivative therefore becomes

$$\begin{aligned} \partial_{t,t}^2 J_\kappa(\ell_h) = {} & (\nabla_t \bar{U})^T K^t(\ell_h) \bar{W} + (\nabla_t \bar{W})^T K^t(\ell_h) \bar{U} \\ & - (\nabla_t P)^T C^t(\ell_h) P_w - (\nabla_t P_w)^T C^t(\ell_h) P \\ & - P C^{tt}(\ell_h) P_w + \partial_{t,t}^2 R(\ell_h). \end{aligned} \tag{3.44}$$

The matrices are defined as $K^t(\ell_h)$ as given in (3.31), $C^t(\ell_h)$ as in (3.35), and $C^{tt}(\ell_h)$ as in (3.36).

Summarizing, the following scheme can be used to compute the Hessian for the second-order adjoint approach:

1. Compute $U = (\bar{U}, P)$ by solving linear system (3.29).

2. Compute $\nabla_t U = (\nabla_t \bar{U}, \nabla_t P)$ in all directions $1 \leq t \leq k$ using formula (3.37).

3. Compute $W = (\bar{W}, P_w)$ by solving linear system (3.38).

4. Compute $\nabla_t W = (\nabla_t \bar{W}, \nabla_t P_w)$ in all directions $1 \leq t \leq k$ using formula (3.39).

5. Compute $\nabla^2 J_\kappa(\ell_h)$ by finding $\partial_{t,t}^2 J_\kappa(\ell_h)$ in all directions $1 \leq t \leq k$ using formula (3.44).

As with the direct approach for the first OLS derivative, this method suffers from the need of solving too many linear systems. In this case we have to solve k systems to find ∇U and another k systems to find ∇W. The second-order adjoint method whose discretization is shown next does not need the derivative of the adjoint variable and thus saves itself a lot of computations. It is hence preferable to the direct second-order approach.

3.4.5 Computation of the Hessian by the Second-order Adjoint Method

We now consider the discrete formulation of the second derivatives for the OLS functional. The second-order adjoint approach has the benefit of not requiring the second derivative of the parameter-to-solution map $u(\ell)$, and further it does not require any derivatives of $w(\ell)$. We recall that the second-order adjoint approach led to the following formula

$$\begin{aligned} D^2 J_\kappa(\ell)(\delta\ell, \delta\ell) = \kappa D^2 R(\ell)(\delta\ell, \delta\ell) + \langle \delta\bar{u}, \delta\bar{u} \rangle + \langle \delta p, \delta p \rangle \\ + 2a(\delta\ell, \delta\bar{u}, \bar{w}) - 2\partial_\ell c(\ell, \delta p, p_w)(\delta\ell) - \partial_\ell^2 c(\ell, p, p_w)(\delta\ell, \delta\ell). \end{aligned}$$

The discrete version of the second-derivative in direction A_t is therefore given as

$$\begin{aligned} \partial_{t,t}^2 J_\kappa(\ell_h) = \kappa \partial_{t,t}^2 R(\ell_h) + (\nabla_t \bar{U})^T M (\nabla_t \bar{U}) + (\nabla_t P)^T M_P (\nabla_t P) \\ + 2(\nabla_t \bar{U})^T K^t(\ell_h) \bar{W} - 2(\nabla_t P)^T C^t(\ell_h) P_w - P C^{tt}(\ell_h) P_w. \end{aligned} \tag{3.45}$$

We have the matrix $K^t(\ell_h)$ as given in (3.31), $C^t(\ell_h)$ as in (3.35), and $C^{tt}(\ell_h)$ as in (3.36). The procedure for the computation of the Hessian of the regularized OLS is hence:

Step 1. Compute $U = (\bar{U}, P)$ by solving linear system (3.29).

Step 2. Compute $W = (\bar{W}, P_w)$ by solving linear system (3.38).

Step 3. Compute $\nabla_t U = (\nabla_t \bar{U}, \nabla_t P)$ in all directions $1 \leq t \leq k$ using formula (3.37).

Step 4. Compute $\nabla^2 J_\kappa(\ell_h)$ by finding $\partial_{t,t}^2 J_\kappa(\ell_h)$ in all directions $1 \leq t \leq k$ using formula (3.45).

Chapter 4

Computational Implementation

The computational implementation of the systems outlined in the previous was performed in C++ using the deal.II library [46, 47]. The library provides many mathematical tools necessary for the finite element method. An efficient modular framework was built to allow the the testing of different inverse problems. The resulting program was used to produce all numerical results in this thesis.

Unless otherwise stated, the measured solution z for each of the following inverse problems is generated computationally. To do so, a desired analytical solution a_{exact} of the parameter is chosen and the corresponding discrete parameter $a_{h,\,\text{exact}}$ computed exactly on the parameter mesh $\mathcal{T}_A$. Following that, the forward problem is solved once to produce a solution $u_{h,\,\text{exact}}$. To avoid committing the so-called inverse crime of taking $z = u_{h,\,\text{exact}}$ a certain amount of random noise is added to z. The noise is created from a uniform distribution on the interval $[-\alpha, \alpha]$, where α is the noise level. A new random noise value η_i is computed for each component of the solution vector. In essence, the coefficients Z_i of the measured solution z are given by

$$Z_i = U_{i,\,\text{exact}} + \eta_i, \qquad 1 \leq i \leq n.$$

Once the measured solution z is created in this fashion, the exact parameter $a_{h,\,\text{exact}}$ and solution $u_{h,\,\text{exact}}$ are set aside and treated as unknown. The inverse problem is then solved using z as the measured solution to produce a final parameter estimate a_h with corresponding solution u_h. These are then compared against their exact counterparts, $a_{h,\,\text{exact}}$ and $u_{h,\,\text{exact}}$, to judge the effectiveness of the reconstruction of the unknown parameter.

4.1 Numerical Results

In the following we present numerous example problems which consist of finding the parameter a in (2.1). Recall that we have imposed homogeneous Dirichlet boundary conditions along the entire boundary $\partial\Omega$.

4.1.1 Scalar Problem

Example 4.1.1. Consider equation (2.1) in a 2D setting with domain $\Omega = [0,1] \times [0,1]$. Take the points $p_1 = (0.6, 0.3)$ and $p_2 = (0.4, 0.75)$ and let the exact parameter function a and the load function f be given by

$$\begin{aligned} a(x) &= 1 + \frac{0.5}{1 + e^{50\|p_1 - x\|^2 - 3}} + \frac{0.3}{1 + e^{100\|p_2 - x\|^2 - 2}}, \\ f(x) &= 1 + 4\,\|x\|\,. \end{aligned}$$

The following options for the parameter map are investigated:

$$g_1(a) = a, \qquad g_2(a) = a^3, \qquad g_3(a) = \frac{1}{a}$$

The results of Example 4.1.1 are shown in Figures 4.1, 4.2, and 4.3. The first thing to note is that the parameter is identified well for all choices of parameter map g, both for linear maps and for nonlinear maps. This is a clear verification that the methods methods outlined in the earlier chapters for the identification of nonlinear parameters indeed work as intended. The effect of the regularizer can be seen by the smoothed out region between the two hills in $a_h(x)$, which is clearly visible in the results for the linear parameter map g_1 in Figure 4.1. The identified parameters for the two nonlinear maps g_2 and g_3 don't experience as much of an impact from the regularizer. As a result the final estimation is in a sense more accurate for the nonlinear parameters. Yet, the costs of solving the inverse problem and the resulting runtimes when using nonlinear maps such as g_2 or g_3 are significantly higher than for the linear map g_1. The computational impact of nonlinear parameters is discussed together with performance optimizations in Section 4.2. Note also, that while the actual parameter a_h (top left plots) is the same for all results, the resulting solution u_h (lower right plots) changes with each choice of the parameter map g.

Example 4.1.2. Consider equation (2.1) in a 2D setting with domain $\Omega = [0,1] \times [0,1]$. Let the exact parameter function a and the load function f be given by

$$\begin{aligned} a(x) &= 1 + e^{\|x\|}, \\ f(x) &= 0.5 + \|x\|^2 + 1.3\,[\sin(20\,\|x\|) + 1]\,, \end{aligned}$$

and let the parameter map options be:

$$g_1(a) = a, \qquad g_2(a) = a^3, \qquad g_3(a) = \frac{1}{a}$$

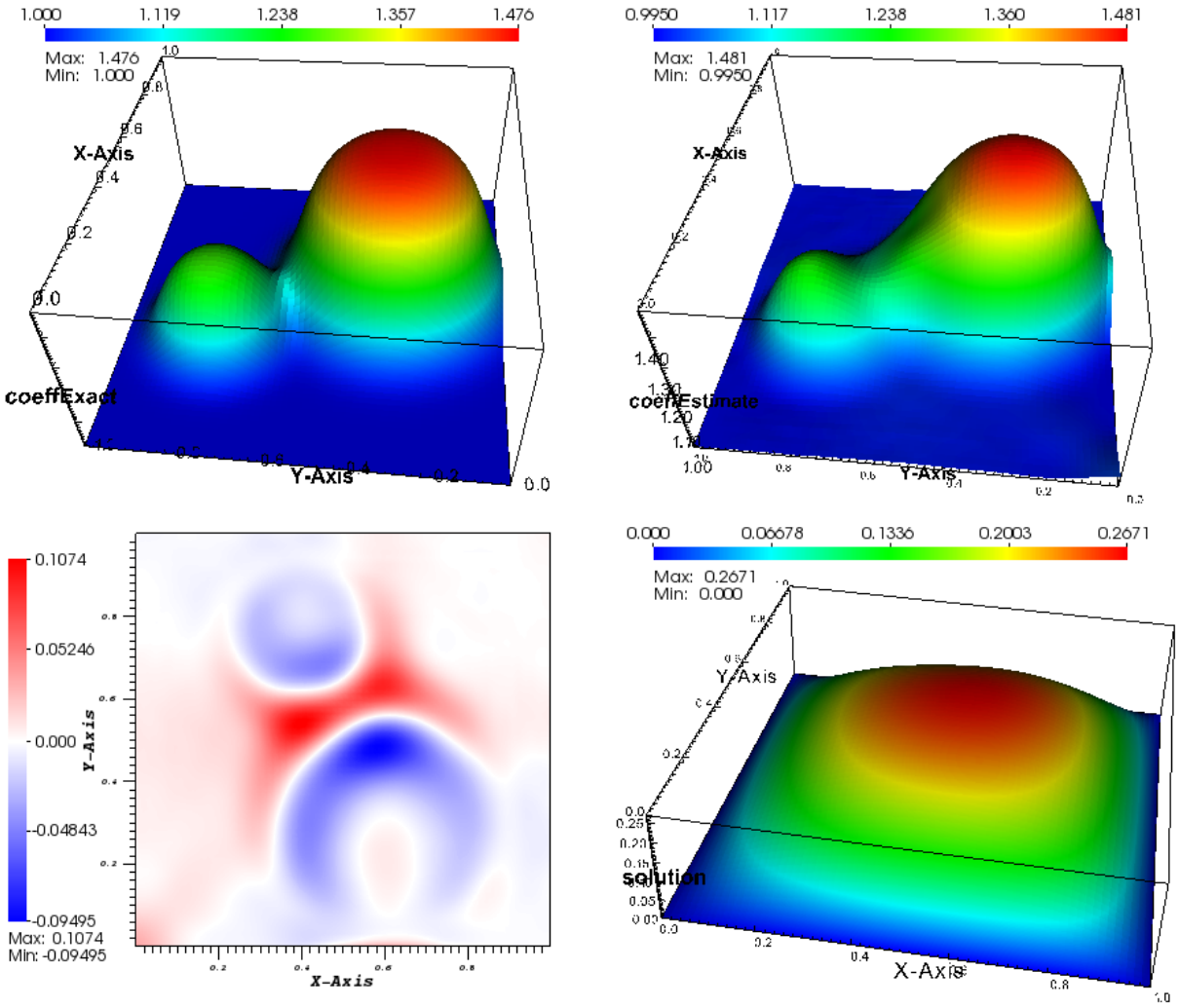

Figure 4.1: Exact and estimated parameter a_h, error in parameter a_h, and estimated solution u_h for Example 4.1.1 with parameter map g_1.

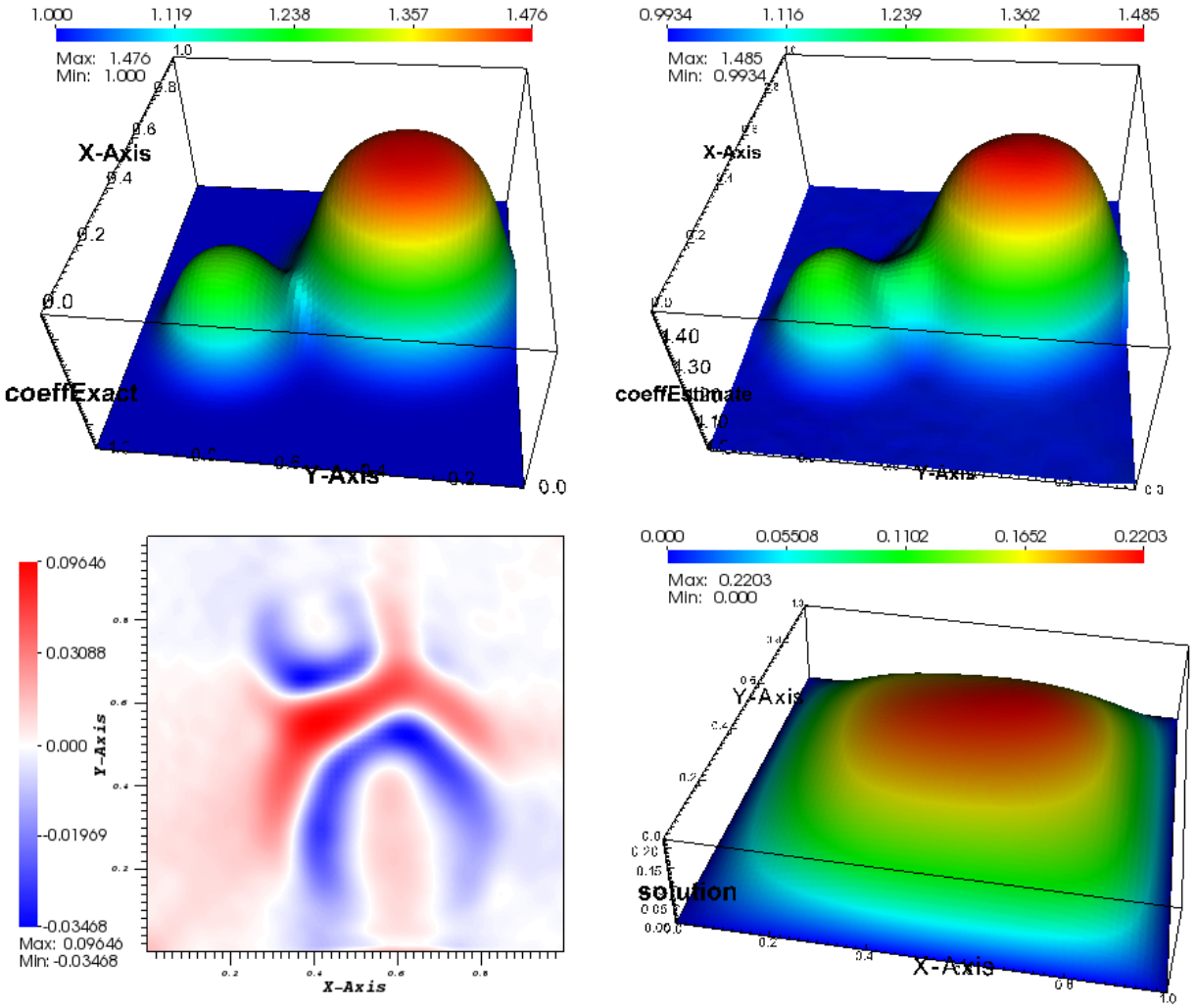

Figure 4.2: Exact and estimated parameter a_h, error in parameter a_h, and estimated solution u_h for Example 4.1.1 with parameter map g_2.

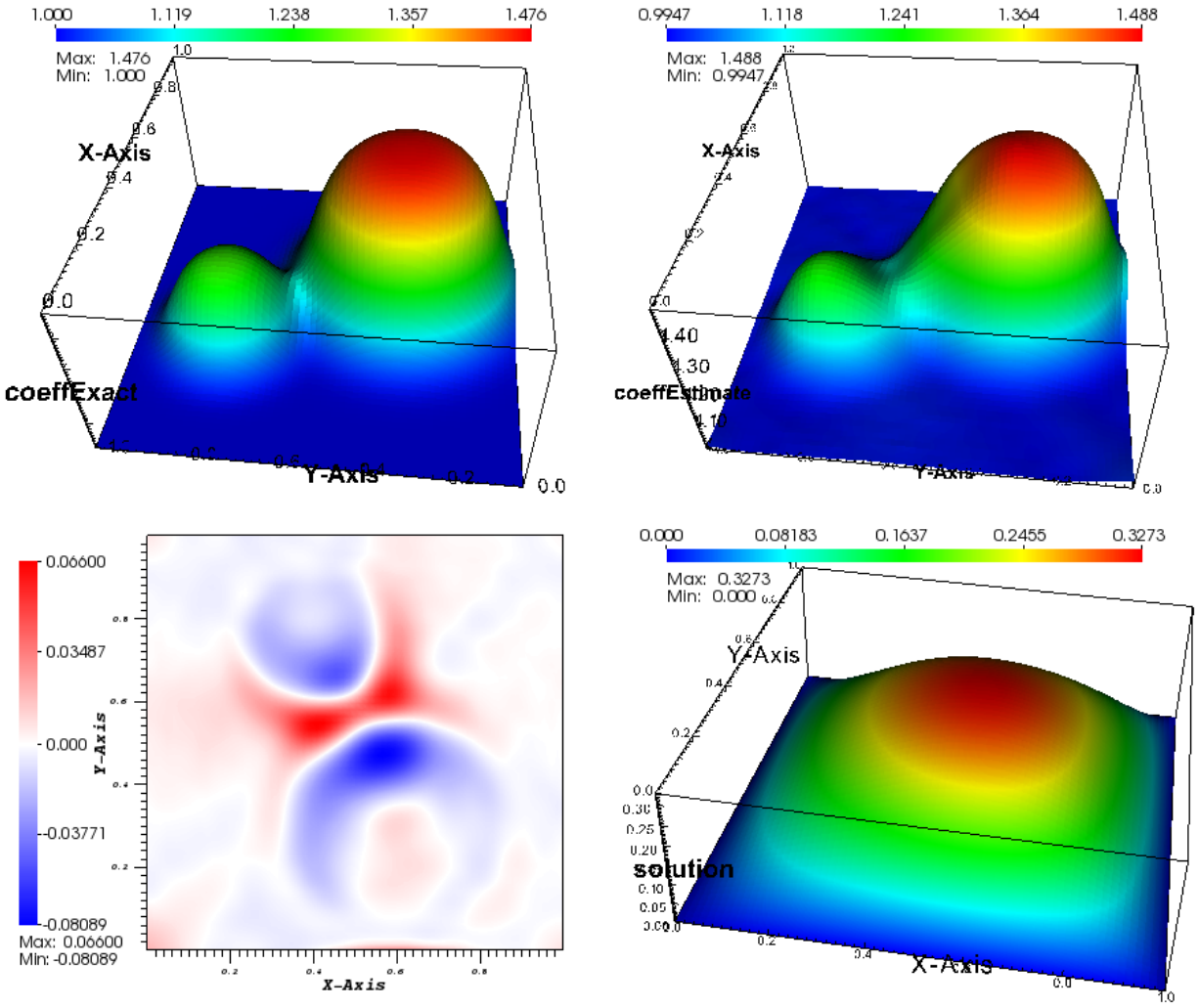

Figure 4.3: Exact and estimated parameter a_h, error in parameter a_h, and estimated solution u_h for Example 4.1.1 with parameter map g_3.

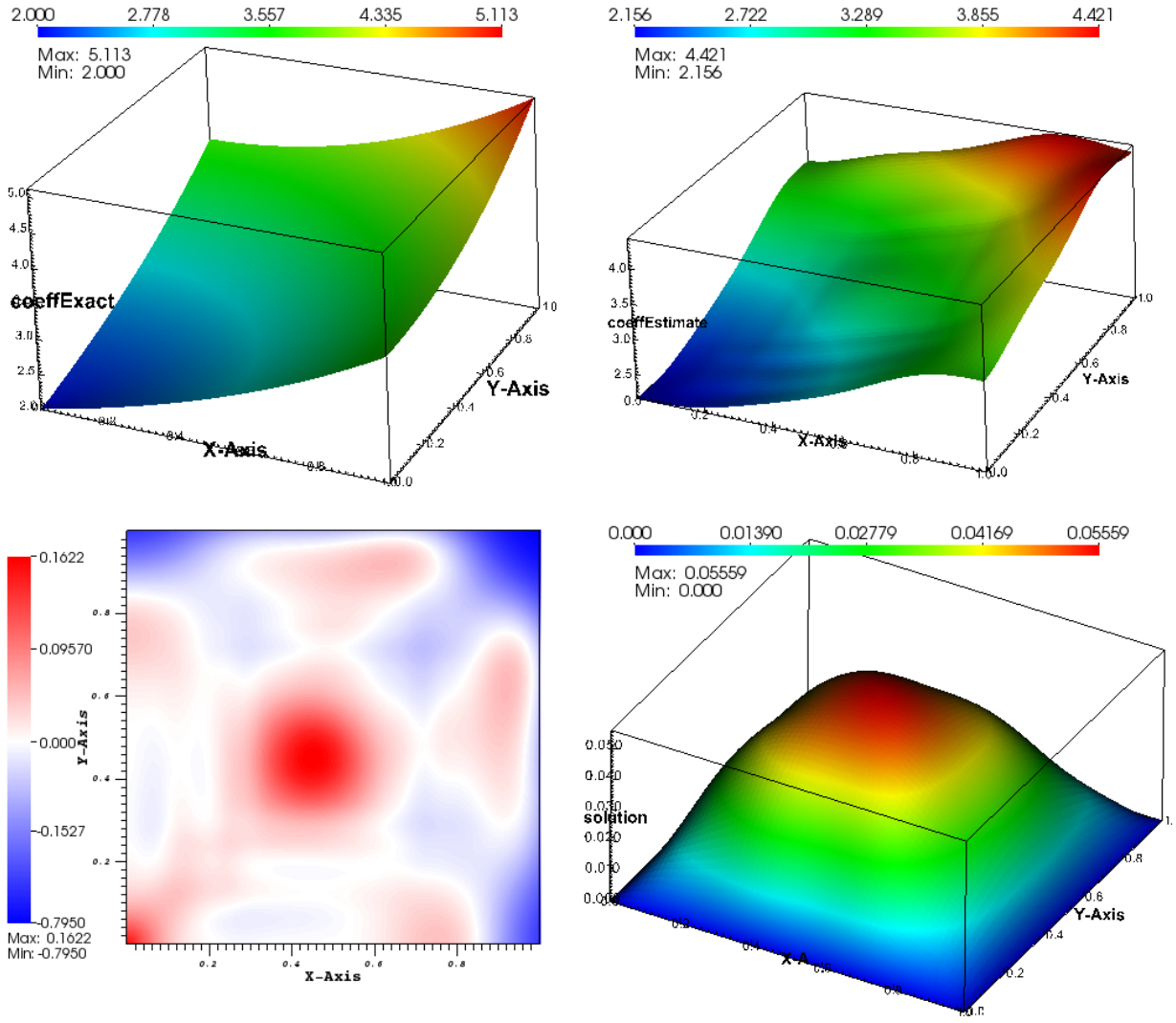

Figure 4.4: Exact (top left) and estimated (top right) parameter a_h, error in parameter a_h in bottom left, and estimated solution u_h in bottom right for Example 4.1.2 with parameter map g_1. Regularization is done using the H_1 semi-norm with $\kappa = 4 \cdot 10^{-8}$. The reconstruction is partially disturbed by the noise of level $\alpha = 0.001$.

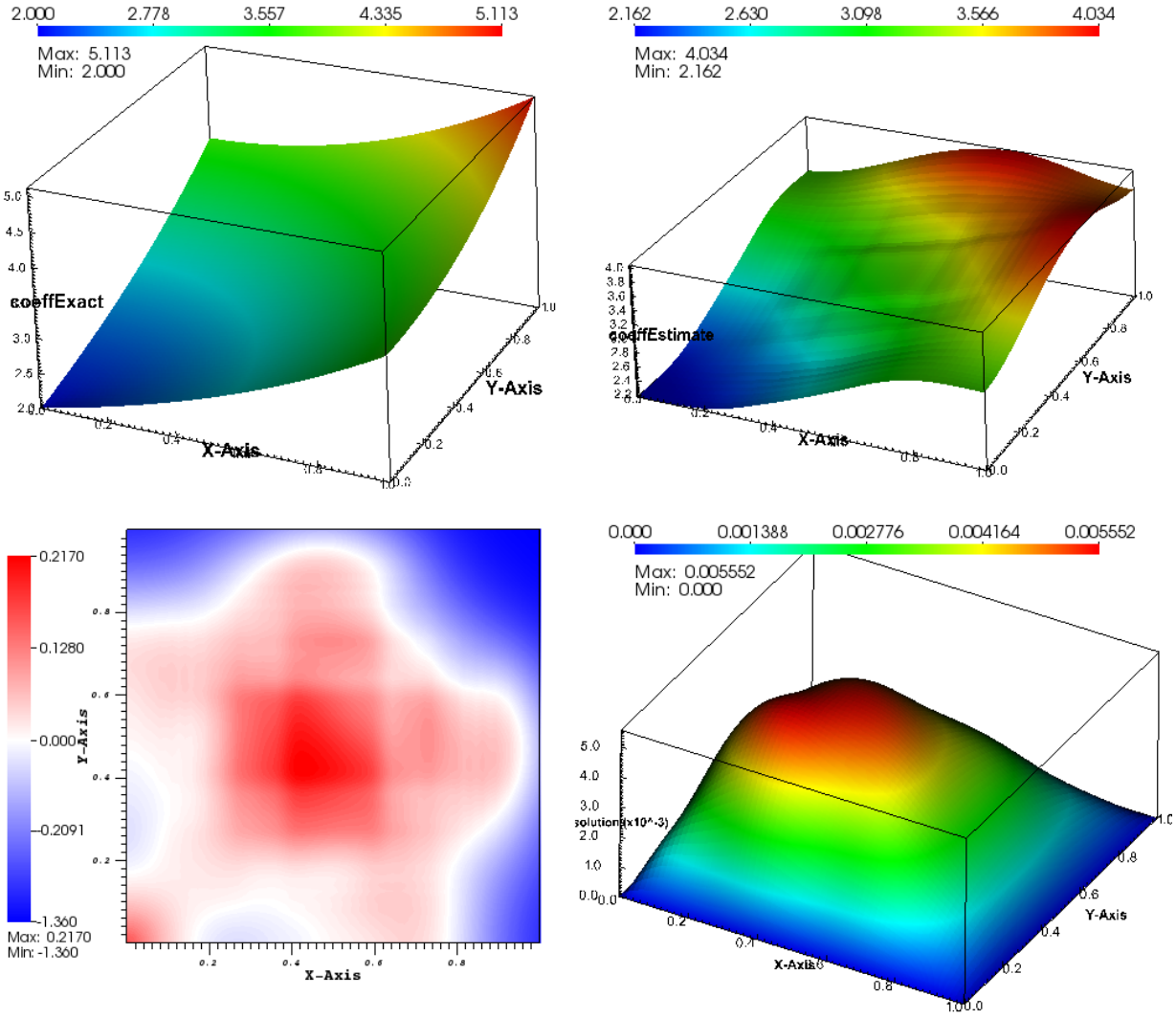

Figure 4.5: Exact (top left) and estimated (top right) parameter a_h, error in parameter a_h in bottom left, and estimated solution u_h in bottom right for Example 4.1.2 with parameter map g_2. Regularization is done using the H_1 semi-norm with $\kappa = 7 \cdot 10^{-9}$. The reconstruction is corrupted due to the noise of level $\alpha = 0.001$ which is of the same order of magnitude as the solution variable u.

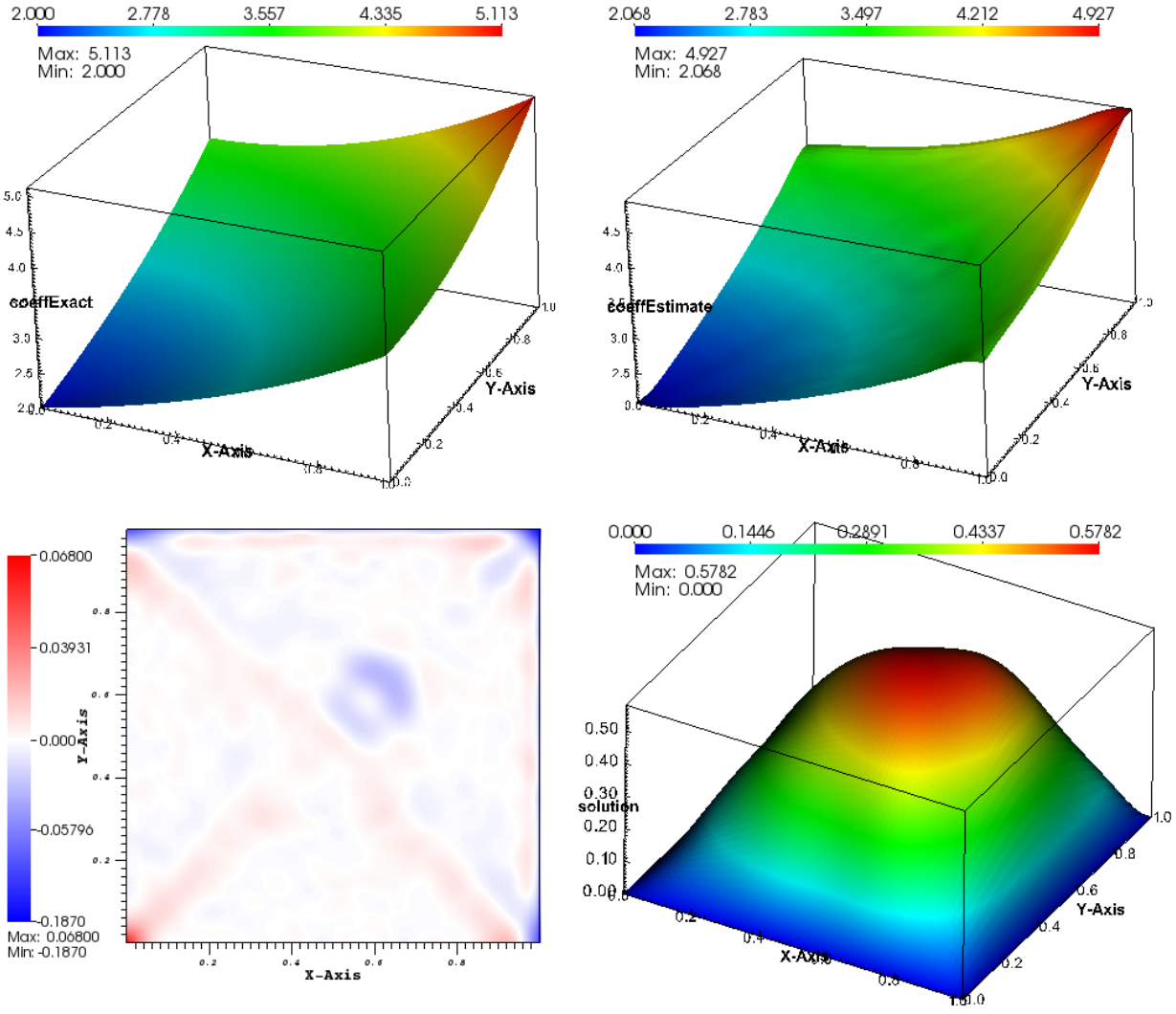

Figure 4.6: Exact (top left) and estimated (top right) parameter a_h, error in parameter a_h in bottom left, and estimated solution u_h in bottom right for Example 4.1.2 with parameter map g_3. Regularization is done using the H_1 semi-norm with $\kappa = 7 \cdot 10^{-9}$. The solution u is much larger in magnitude than the noise level $\alpha = 0.001$, and thus the reconstruction is mostly uncorrupted.

The results for Example 4.1.2 show nicely what effect the noise in the measured solution z has on the reconstruction of a—see figures 4.4, 4.5, and 4.6. The magnitude of the solution u is very different for each of the parameter maps g, so the effect of the added random noise, always with noise level $\alpha = 0.001$, changes. In the case of $g_3(a) = \frac{1}{a}$ the solution is very large in value and much larger than the add noise. The noise thus causes only little errors in the recovered parameter a. On the other hand, $g_2(a) = a^3$ causes the solution variable to be very small in magnitude, on the same level as the noise. The corruption through the random noise is therefore very pronounced and prevents a good reconstruction of a. The first case $g_1(a) = a$ lies in between the other cases. The noise adds some noticeable corruption to the identified parameter a, but not as strong as in the case of g_3.

Example 4.1.3. We will solve equation (2.1) in a 3D domain $\Omega = [0,1] \times [0,1] \times [0,1]$. This example is an extension of Example 4.1.1 for three dimensions. Take the points $p_1 = (0.6, 0.3, 0.3)$ and $p_2 = (0.4, 0.75, 0.6)$ and let the exact parameter function a and the load function f be given by

$$a(x) = 1 + \frac{0.5}{1 + e^{50\|p_1 - x\|^2 - 3}} + \frac{0.3}{1 + e^{100\|p_2 - x\|^2 - 2}},$$
$$f(x) = 1 + 4\,\|x\|\,.$$

The following options for the parameter map are investigated:

$$g_1(a) = a, \qquad g_2(a) = a^3, \qquad g_3(a) = \frac{1}{a}$$

4.1.2 MOLS Functional

Example 4.1.4. Consider equation (2.1) in a 2D setting with domain $\Omega = [0,1] \times [0,1]$. The position vector is thus $x = (x_1, x_2)$. The exact parameter a and the load function f are given by:

$$a(x) = 1.5 + 0.2x_2^2 + 0.1(\sin(20x_1) + 1)$$
$$f(x) = 4 + 0.02(x_2 - 0.5)^2$$

The following options for the parameter map are investigated:

$$g_1(a) = a, \qquad g_2(a) = a^3, \qquad g_3(a) = \frac{1}{a}$$

In figures 4.10, 4.11, and 4.12 the results of running Example 4.1.4 with the MOLS functional are presented. For the tests, a uniformly refined grid of 128×128 cells is used. The regularization value is set as $\kappa = 4 \cdot 10^{-6}$, and the noise level for the measured solution is taken to be $\alpha = 10^{-5}$. Figures 4.10, 4.11, and 4.12 show reconstructions of the parameter a using maps g_1 and g_2 respectively. The MOLS functional

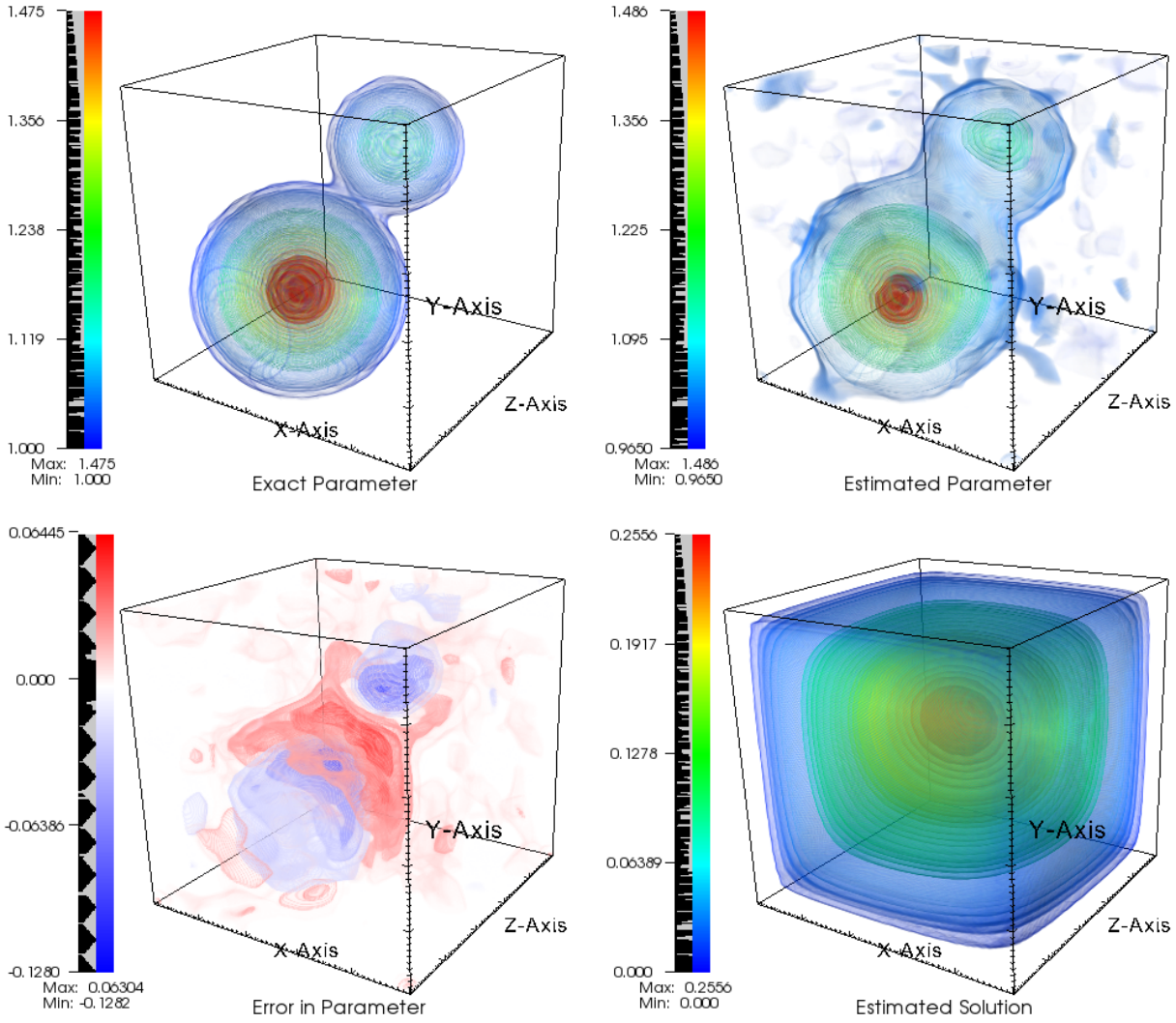

Figure 4.7: Exact (top left) and estimated (top right) parameter a_h, error in parameter a_h in bottom left, and estimated solution u_h in bottom right for Example 4.1.3 with parameter map g_1. The H_1 semi-norm was used with regularization value $\kappa = 4 \cdot 10^{-8}$ and a noise level $\alpha = 5 \cdot 10^{-4}$ was applied to the measured solution.

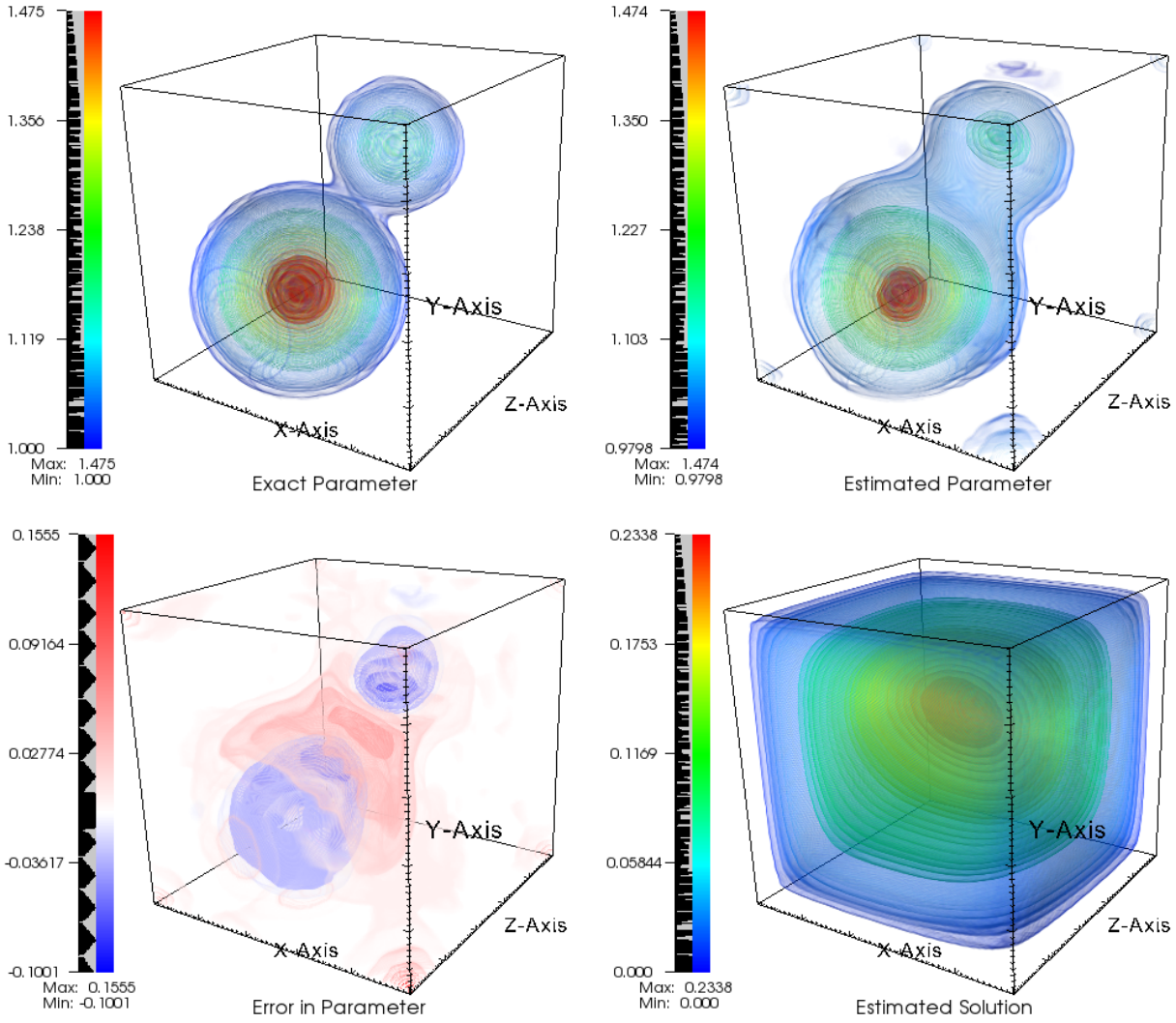

Figure 4.8: Exact (top left) and estimated (top right) parameter a_h, error in parameter a_h in bottom left, and estimated solution u_h in bottom right for Example 4.1.3 with parameter map g_2. The H_1 semi-norm was used with regularization value $\kappa = 4 \cdot 10^{-7}$ and a noise level $\alpha = 5 \cdot 10^{-4}$ was applied to the measured solution.

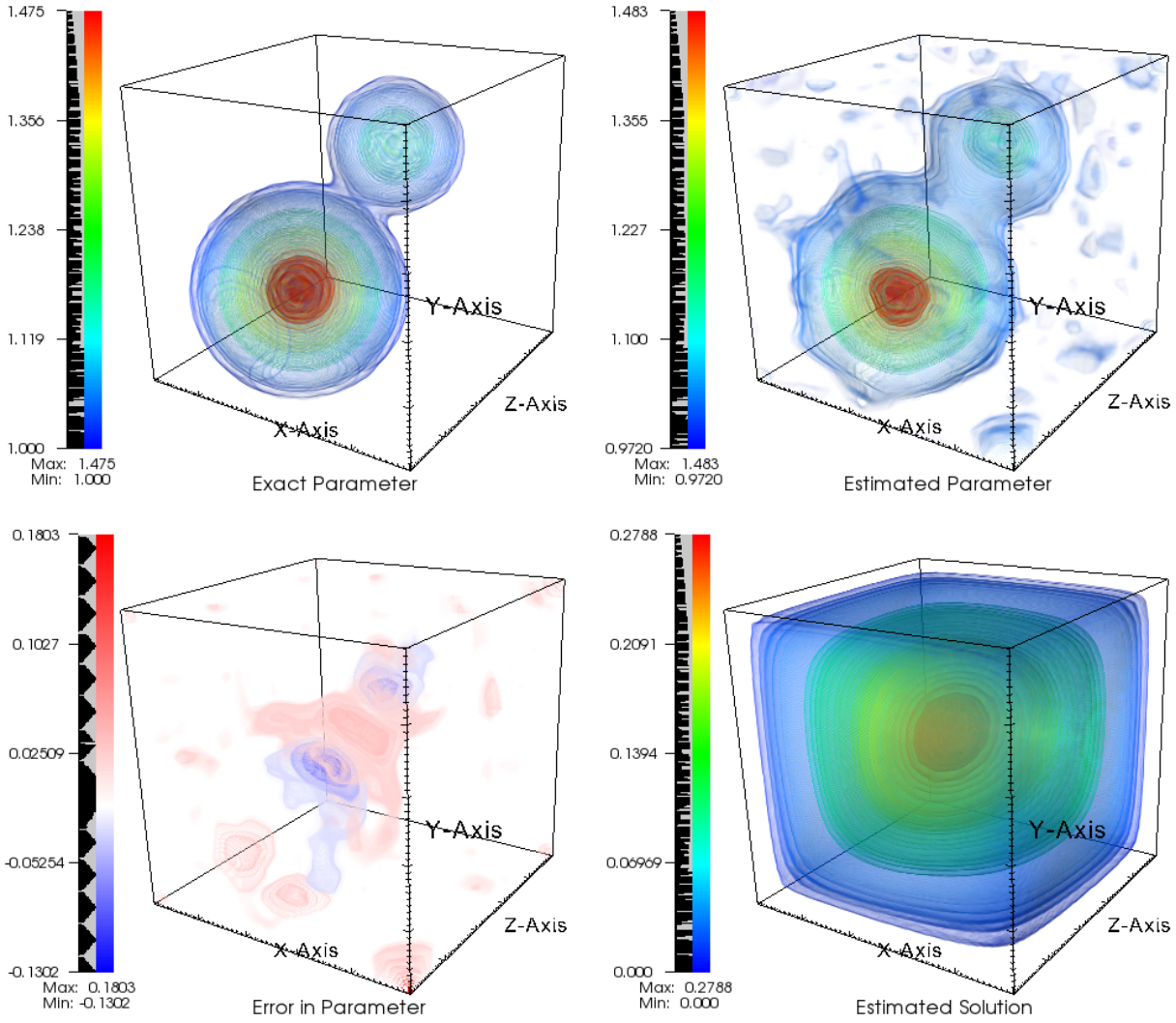

Figure 4.9: Exact (top left) and estimated (top right) parameter a_h, error in parameter a_h in bottom left, and estimated solution u_h in bottom right for Example 4.1.3 with parameter map g_3. The H_1 semi-norm was used with regularization value $\kappa = 4 \cdot 10^{-8}$ and a noise level $\alpha = 5 \cdot 10^{-4}$ was applied to the measured solution.

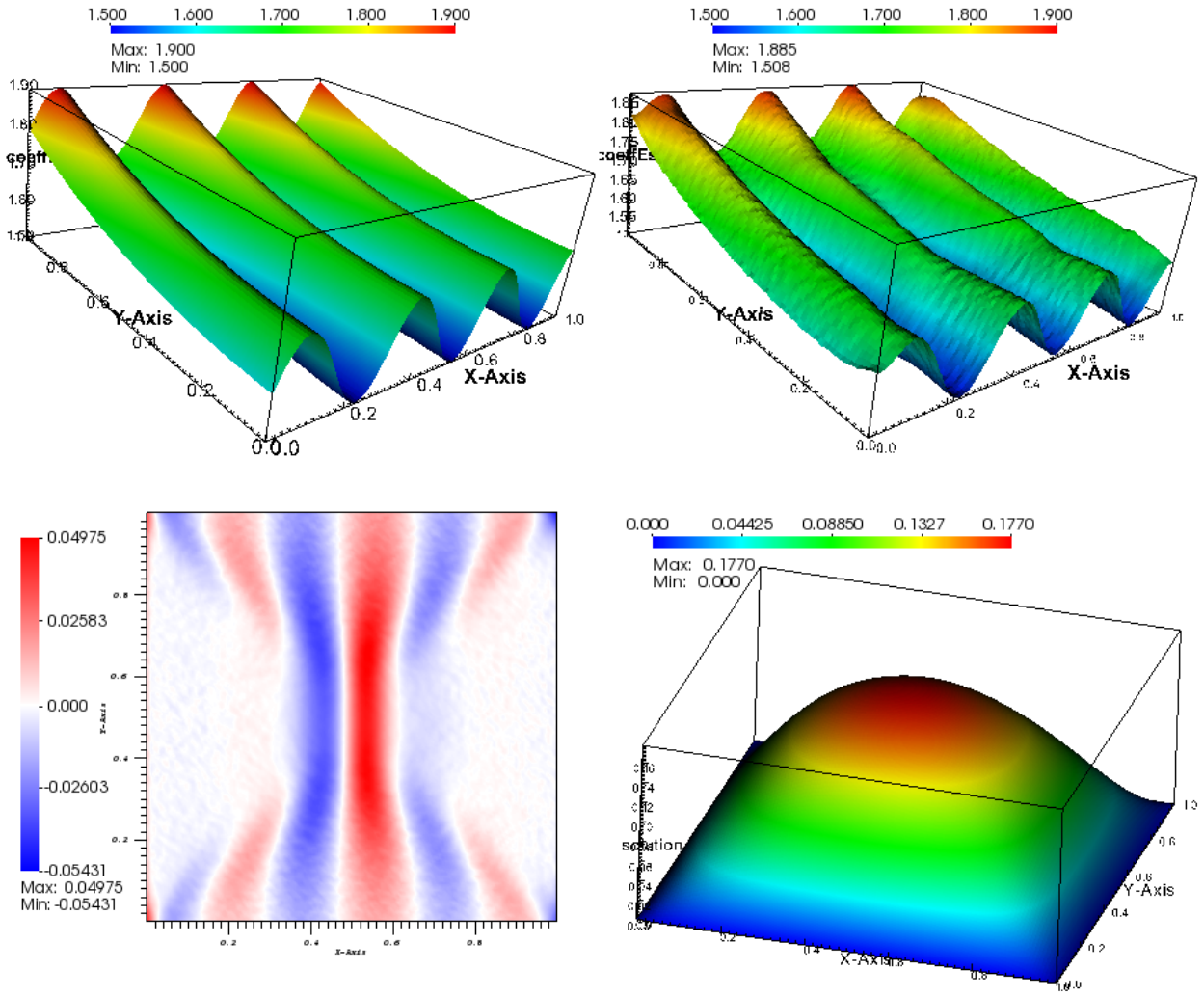

Figure 4.10: Exact (top left) and estimated (top right) parameter a_h, error in parameter a_h in bottom left, and estimated solution u_h in bottom right for Example 4.1.4 with parameter map g_1, regularization value $\kappa = 4 \cdot 10^{-6}$ and noise level $\alpha = 10^{-5}$, using the MOLS functional.

gives good results, for both linear maps (g_1) where the MOLS is convex and for nonlinear maps (g_2, g_3) where it is not convex. This noise level is large enough to produce a visible degradation of the identified parameter a, but still small enough to not render the inverse problem impossible. In the given examples, the noise in the measured solution is the limiting factor in the accuracy of the solution.

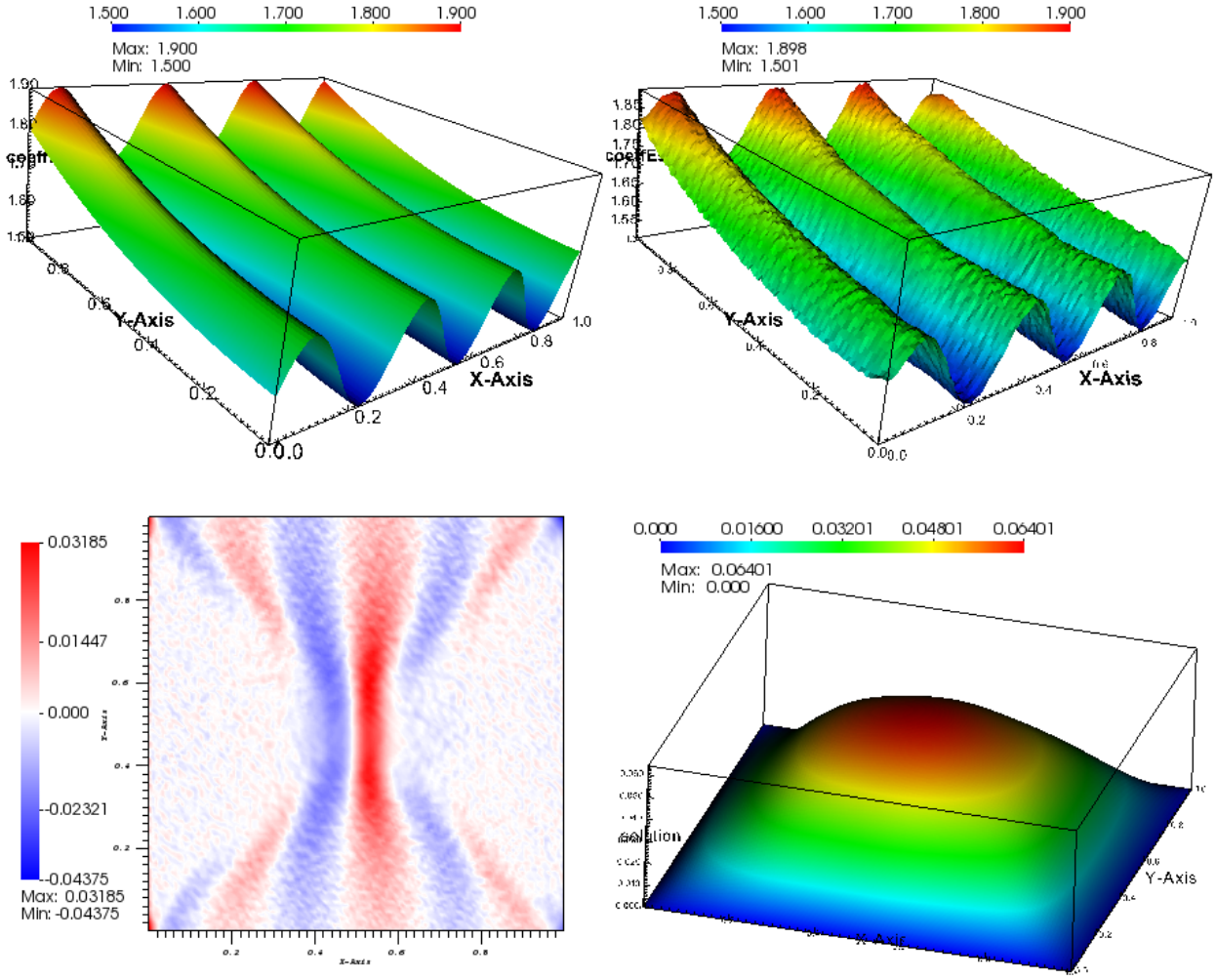

Figure 4.11: Exact (top left) and estimated (top right) parameter a_h, error in parameter a_h in bottom left, and estimated solution u_h in bottom right for Example 4.1.4 with parameter map g_2, regularization value $\kappa = 4 \cdot 10^{-6}$ and noise level $\alpha = 10^{-5}$, using the MOLS functional.

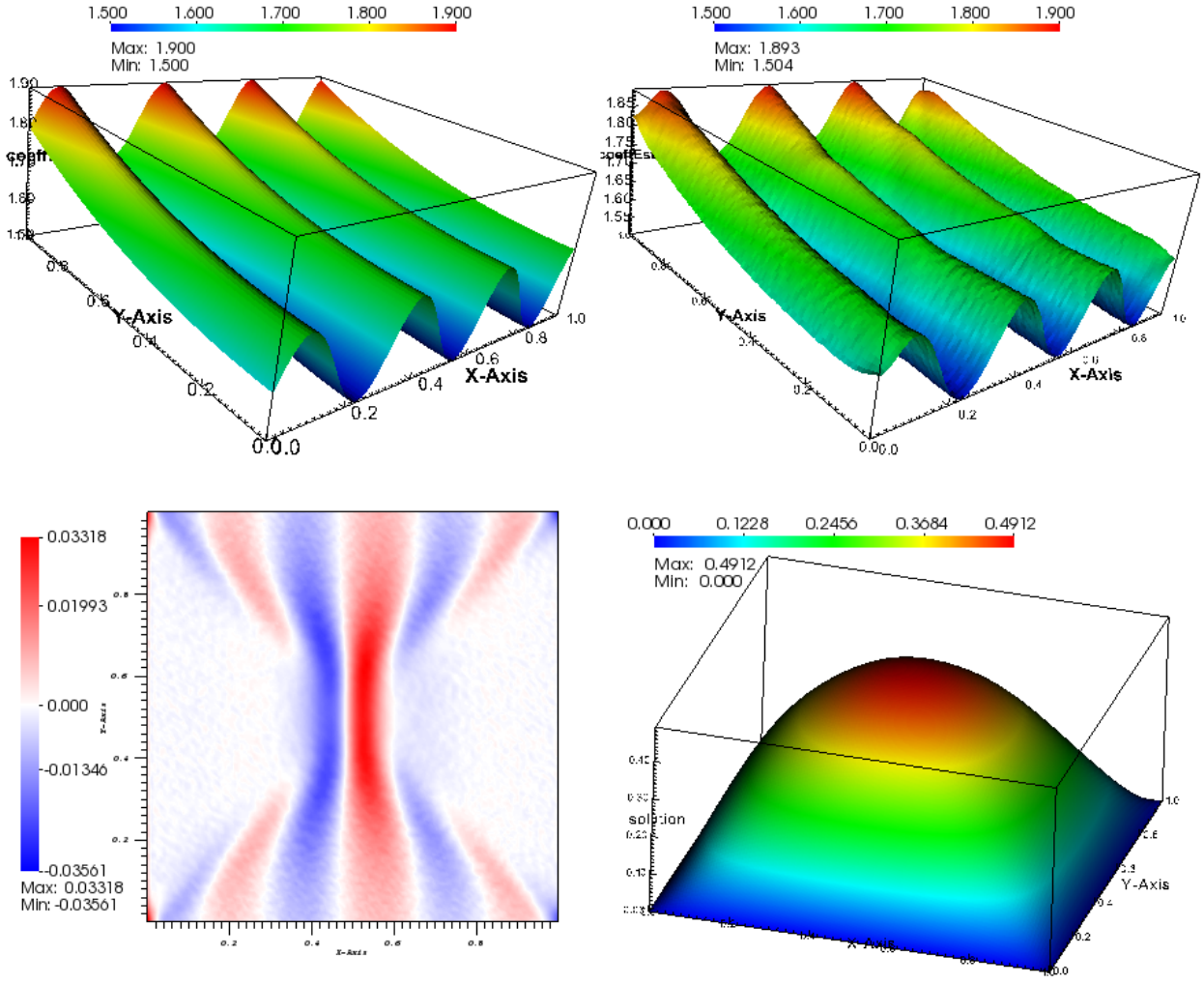

Figure 4.12: Exact (top left) and estimated (top right) parameter a_h, error in parameter a_h in bottom left, and estimated solution u_h in bottom right for Example 4.1.4 with parameter map g_3, regularization value $\kappa = 4 \cdot 10^{-6}$ and noise level $\alpha = 10^{-5}$, using the MOLS functional.

4.1.3 Elasticity Problem

Example 4.1.5. Consider the elasticity problem as defined in (3.1). Define the problem domain in $\mathbb{R}^2$ as the unit square $\Omega = [0,1] \times [0,1]$. The position vector is thus $x = (x_1, x_2)$. Let the domain boundaries Γ_1 be the right boundary where $x = 1$ and Γ_2 the remaining three sides. Take point $p = (0.4, 0.5)$ and range value $r = 0.2$. The exact parameter ℓ, the load function f, and the boundary functions g on Γ_1 and h on Γ_2 are given by:

$$\ell(x) = \begin{cases} 0.1, & \text{if } \|x - p\| < r \\ 0.1 + 10(r^2 - \|x - p\|^2), & \text{if } \|x - p\| \geq r \end{cases}$$

$$f(x) = \begin{bmatrix} f_1(x_1) \\ f_2(x_2) \end{bmatrix} = \begin{bmatrix} -0.1\sin(\pi x_1) \\ 0.1\cos(\pi x_1) \end{bmatrix}$$

$$g(x) = 0$$

$$h(x) = \begin{bmatrix} h_1(x_1) \\ h_2(x_2) \end{bmatrix} = \begin{bmatrix} 0.01 \\ 0 \end{bmatrix}$$

As discussed in Section 3.1.1, the inverse problem consists of finding the elasticity parameter $\mu = \ell$. The second parameter λ is described by $\lambda = \tau\mu$, where we take $\tau = 10^5$ to simulate the near incompressible case. The noise level added to the measured solution z is $\alpha = 0.001$ and the regularization value is taken as $\kappa = 10^{-7}$.

Example 4.1.6. The problem considered is again (3.1), with the domain in $\mathbb{R}^2$ being defined as a ring centered at the origin with inner radius $r_1 = 0.5$ and outer radius $r_2 = 1.0$. The inner boundary is defined as Γ_1 where the Dirichlet boundary conditions hold. The outer boundary is taken as Γ_2. The Lamé parameter μ that will be identified as ℓ in the equations is considered to be uniform throughout the domain except for an area with higher intensity value centered at $p = (0.725, 0.25)$ with radius $r = 0.2$. The load function is chosen to resemble a shearing motion along the domain. The exact parameter ℓ, the load function f, and the

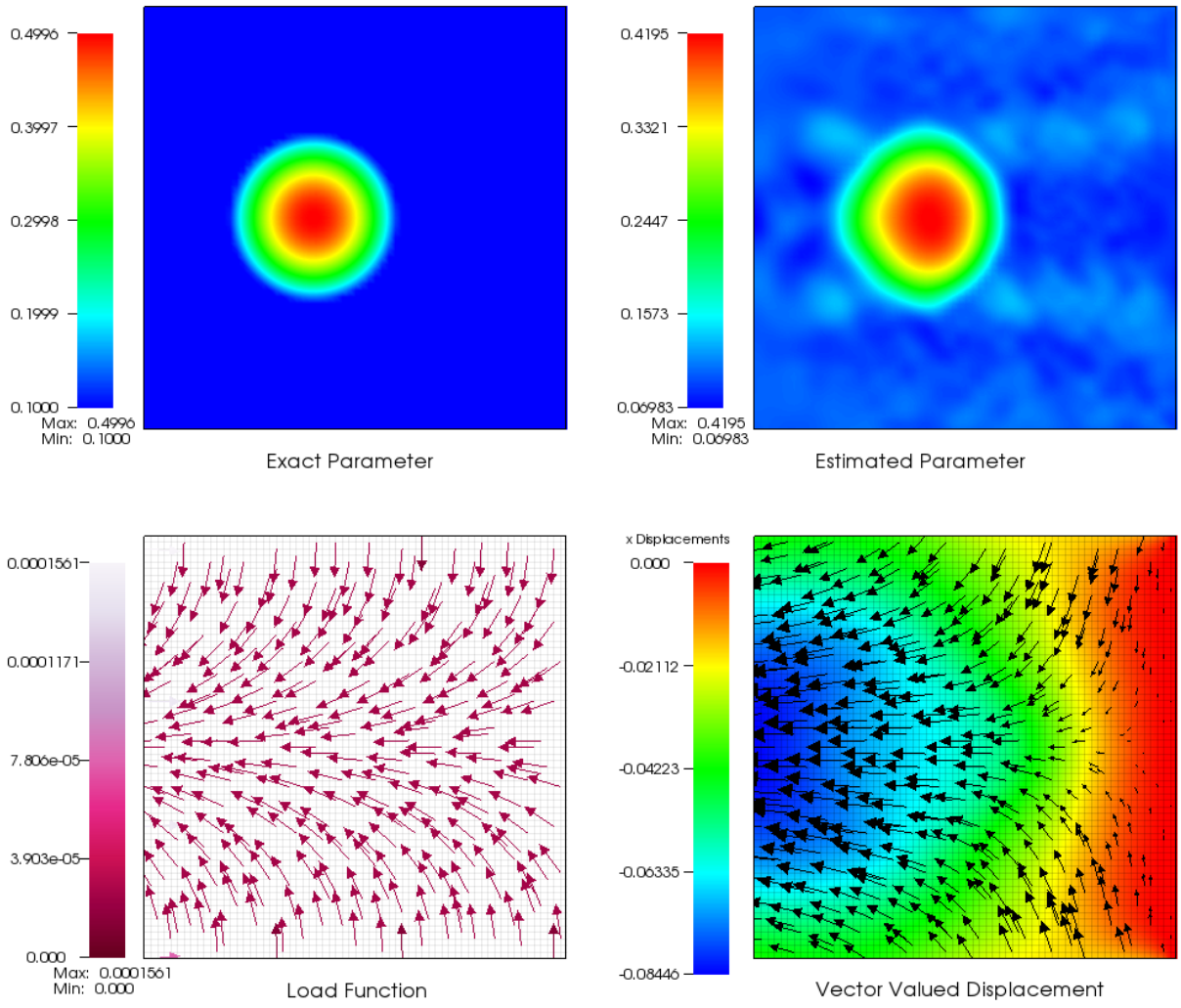

Figure 4.13: Exact (top left) and estimated (top right) parameter ℓ_h, vector valued load function f in the bottom left, and vector valued estimated displacement u_h together with a color plot of the x-axis displacements in the bottom right. Results come from Example 4.1.5 using a uniform mesh with 4096 cells.

boundary functions g on Γ_1 and h on Γ_2 are given by:

$$\ell(x) = \begin{cases} 1.0, & \text{if } \|x-p\| < r \\ 1.0 + 50(r^2 - \|x-p\|^2), & \text{if } \|x-p\| \geq r \end{cases}$$

$$f(x) = \begin{bmatrix} f_1(x_1) \\ f_2(x_2) \end{bmatrix} = \begin{bmatrix} -0.01x_1 \\ 0.0025\sin(0.8\pi x_1) \end{bmatrix}$$

$$g(x) = 0$$

$$h(x) = \begin{bmatrix} h_1(x_1) \\ h_2(x_2) \end{bmatrix} = \begin{bmatrix} 0.01 \\ 0 \end{bmatrix}$$

Again, we take $\lambda = \tau\mu$ with $\tau = 10^5$, and attempt to reconstruct the Lamé parameter μ. The noise level added to the measured solution z is $\alpha = 0.0001$ and the regularization value is taken as $\kappa = 8 \cdot 10^{-10}$ for the H_1 norm.

4.2 Performance Optimization Techniques

When solving the inverse problem on a computer, not only do should the program be mathematically correct and numerically accurate, but it should also run fast. Some factors that determine the speed of a program are the programming language and the resulting execution of the program in machine code, as well as the choice of mathematical algorithms. Yet, an additional factor is simply the efficiency of the actual implementation in the code. In the following, we present a few optimization methods that are purely implementation based, that is methods that do not change the numeric solution but only avoid doing redundant operations. These methods should be applicable to implementations in any language. In our case the time improvements from some of these methods were significant.

4.2.1 Directional Stiffness Matrix

Recall the formula (2.31) for the directional stiffness matrix $K^t(a_h)$ at a parameter a_h in the direction A_t was given by

$$[K^t(a_h)]_{i,j} = T(g'(a_h)\chi_t, \varphi_i, \varphi_j).$$

The computation follows exactly the same pattern as the regular stiffness matrix $K(a_h)$ which was seen in (2.21) to be

$$[K(a_h)]_{i,j} = T(g(a_h), \varphi_i, \varphi_j).$$

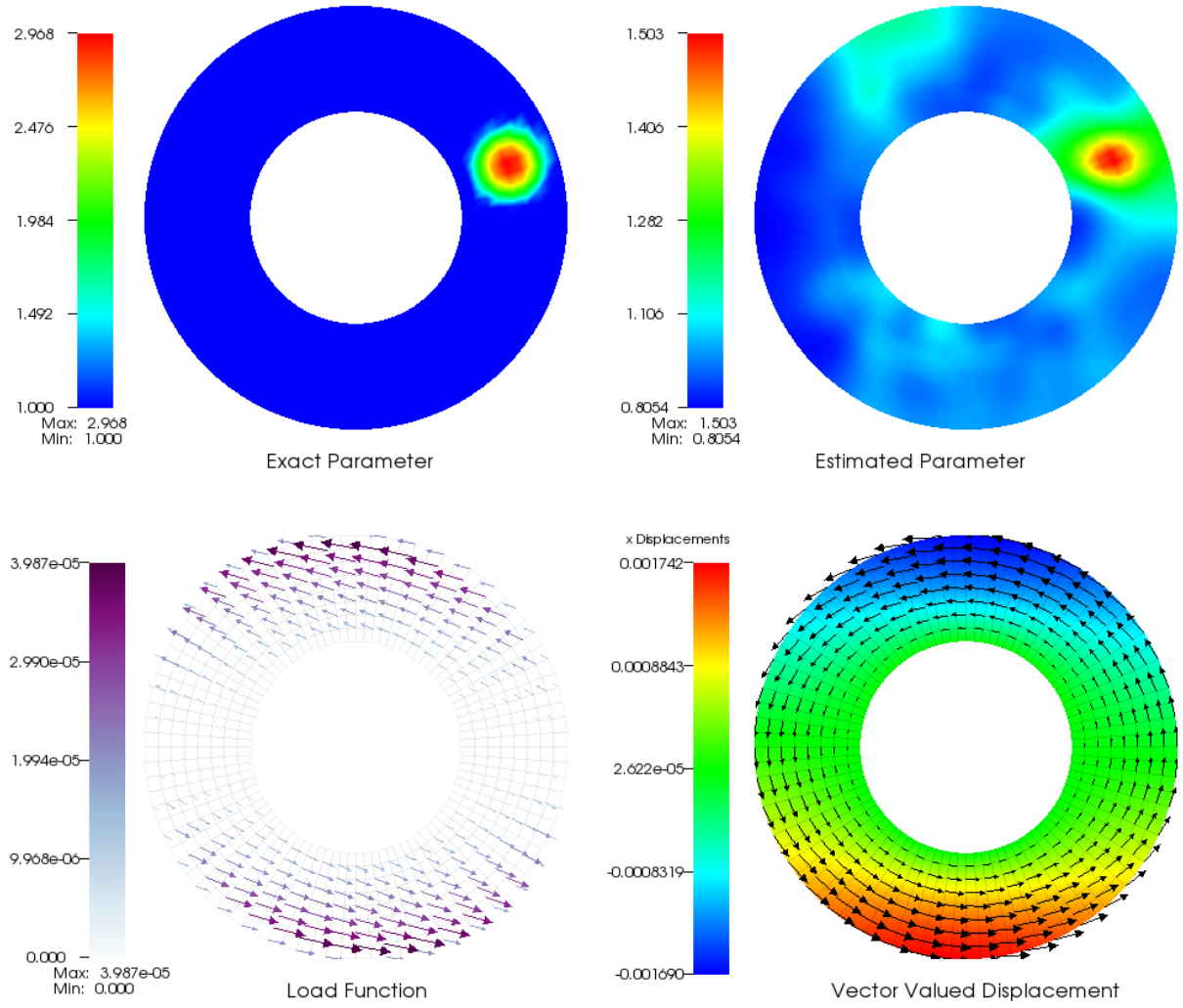

Figure 4.14: Exact (top left) and estimated (top right) parameter ℓ_h, vector valued load function f in the bottom left, and vector valued estimated displacement u_h together with a color plot of the x-axis displacements in the bottom right. Results come from Example 4.1.6 using a mesh with 704 cells.

In essence, the same method can be used to compute both the regular and the directional stiffness matrix, since all that changes is the first argument in T. However, the difference lies in the number of non-zero entries in each of the matrices. For $K(a_h)$, the sparsity pattern of potential non-zero values is determined by the mutual support of the basis functions φ_i and φ_j. In the case of $K^t(a_h)$, the non-zero values occur where the supports of all three basis functions φ_i, φ_j, and χ_t intersect. The support of χ_t extends only over the cells $c_l \in \mathcal{T}_A$ that are adjacent to the support point x_t that defines the center of the basis function χ_t. Since χ_t is fixed for a given direction A_t, the only non-zero values of $K^t(a_h)$ will be at entries i, j for which φ_i and φ_j are defined on the cells c_l. In other words, while the general sparsity patterns of $K(a_h)$ and $K^t(a_h)$ are the same, the actual number of non-zero entries in $K^t(a_h)$ is minuscule. For large scale problems the general sparsity pattern easily encompasses several million values, whereas the actual number of non-zero values in $K^t(a_h)$ stays close to constant regardless of the size of the mesh and usually lies well below 100. If the regular sparsity pattern gets used for $K^t(a_h)$, then the matrix multiplications have to go over all sparsity pattern entries regardless if they are zero or not. If instead a data structure is used that only stores the actual non-zero values of $K^t(a_h)$, call this data structure $\widetilde{K}^t(a_h)$, then that not only reduces the memory cost but also speeds up computations by avoiding the unnecessary calculations with zero values.

4.2.2 Special improvements for Linear Parameters

In the variational problem (2.3) we use the adjoint method to compute the OLS gradient, due to the nonlinear appearance of the parameter a in $g(a)$. The framework we developed can handle any form of $g(a)$, which includes the possibility that g is actually linear (we use this case as parameter map g_1 in the examples). In that special case we can tremendously improve the runtime of the program. Recall that in the gradient computation we have to compute the directional stiffness matrices in each derivative direction δa. In (2.31) we had the following discrete formulation for the directional stiffness matrix in derivative direction A_t:

$$[K^t(a_h)]_{i,j} = T(g'(a_h)\chi_t, \varphi_i, \varphi_j).$$

If g is linear, then its derivative is constant and does not depend on a_h. In other words, we know what the derivative g' looks like regardless of what a_h is. The direct consequence is that even though we update a_h in step of the optimization method, the directional stiffness matrices K^t will be the same every time. Thus, we can avoid redundant computations by computing each directional matrix K^t only once for each $1 \leq t \leq k$ and then storing the matrix for later. This does not come at a large memory cost if we use the optimized data structure $\widetilde{K}^t$ for the stiffness matrices that was discussed in the previous Section 4.2.1. In essence we have a map M of k elements given by

$$M(t) = \widetilde{K}^t, \qquad 1 \leq t \leq k$$

where each element $M(t) = \widetilde{K}^t$ should only store the actually nonzero entries of K^t to avoid unnecessary memory consumption. In the first step of the optimization method, we compute the matrices $\widetilde{K}^t$ and store

them in the map M. After that, whenever we need to use any matrix K^t, we simply look up the corresponding $\widetilde{K}^t$ as the map element $M(t)$. The benefit we get from an immediate matrix lookup as opposed to having to manually construct the matrices can be immense. Using the map M produces identical results to not using M (because we used a linear parameter map), but the time difference is quite drastic. In the results shown in Table 4.1 the versions of the code using the map M to look up matrices run up to 80 times as fast as the versions not using M. For larger problems, the matrix constructions become increasingly time consuming, and thus the benefits of the map M become more and more pronounced as the problem size increases.

Remark 4.2.1. This method is also applicable to mesh refinement in the optimization process, but the map M has to be cleared during the refinement step since the structure and the number of the stiffness matrices K^t changes. After a refinement occurs the map can be recomputed for the new mesh and then again used to look up the matrices until the mesh is refined once more. Filling in the matrix map is essentially for free except for the memory cost needed for storage, so even if there are only two optimization steps between a refinement, time will be saved. In a scheme such as described in [48] where the optimization process is run without refinements for a period of time, until a desired convergence is reached and a mesh refinement is triggered, the speedup gains from having a map M are still very good.

4.2.3 Heuristic Improvements for Nonlinear Parameters

The map M is not directly applicable to nonlinear maps g as $K^t(a_h)$ changes with a_h in each optimization step. It may be reasonable, however, to approximate $K^t(a_h)$ by the same matrix for a few iterations if a_h is known not to change too much during that time. In that case we could reinitialize the entries of M every z time steps with the exact matrices for the current parameter a_h, where z is a small integer. This would allow us to use the efficient matrix lookup at the cost of degrading the quality of the descent directions. While this is a heuristic approach, one can expect the performance gains to outweigh the potentially increased number of iteration steps necessary to achieve the desired accuracy of a_h. Preliminary results show that for certain problems the correct search direction is important and using the approximate matrices $K^t(a_h)$ prevents the solution from converging. On the other hand, some problems such as Example 4.1.4 work very well with the approximation and the run time improvement from using this heuristic method is immense. See Table 4.2 for numerical results. The mathematical validity of this heuristic method could be a part of future investigations.

exact optimizations for linear parameters

k	OLS runtime			MOLS runtime		
	without M	with M	speed increase	without M	with M	speed increase
1089	111.330s	3.927s	28.35 times	65.572s	2.567s	25.54 times
4225	1027.704s	17.120s	60.03 times	482.431s	10.658s	45.26 times
16641	9607.233s	116.157s	82.71 times	5550.554s	72.889s	76.15 times

Table 4.1: Runtimes for different mesh sizes, where k is the dimension of a_h. The results are given for running Example 4.1.4 with the linear parameter map g_1. The different columns show the runtimes with matrix lookup through M turned off where each stiffness matrix has to be recalculated each step and with map M turned on where the matrices have to calculated only once. The results using M and not using M are completely identical but their runtimes differ by the factor shown in the third columns.

different z values for map M for nonlinear parameters

z	runtime	reinitializations	final residual
1	125m46.250s	437	1.33916e-05
5	37m52.516s	131	1.33916e-05
10	15m55.748s	51	1.33916e-05
20	7m06.918s	20	1.33916e-05
50	3m32.901s	8	1.33916e-05
80	2m26.260s	4	1.33916e-05
100	2m10.902s	3	1.33916e-05

Table 4.2: Runtimes for different values of z that describes the maximum number of times the matrix map M can get reused for nonlinear parameters before it has to be reinitialized. The reinitialization count is shown in the third column, and final OLS values in the last column. Results are given for Example 4.1.4 with parameter map g_3 using the MOLS functional, a regularization value of $\kappa = 4 \cdot 10^{-6}$, a noise value of $\alpha = 5 \cdot 10^{-5}$, and the H_1 semi-norm. The dimension of a_h and therefore the number of gradient directions in the map M is $k = 16641$.
The final residual is limited by the noise and thus the final estimate is similar even when using large values of z. The extra iterations that may be necessary for the heuristic approach clearly get outweighed by the speed improvement of not having to reinitialize M at every step. However, using larger z values for the given example than listed in the table causes the approximated gradient directions to become non-descent directions and the inverse problem fails. The approximation thus works only up to a certain extent, but as can be seen the values of z and the resulting time benefits can be quite large.

Bibliography

[1] Mark S. Gockenbach and Akhtar A. Khan. An Abstract Framework for Elliptic Inverse Problems: Part 1. An Output Least-Squares Approach. *Mathematics and Mechanics of Solids*, 12(3):259–276, June 2007.

[2] Robert Acar. Identification of the Coefficient in Elliptic Equations. *SIAM Journal on Control and Optimization*, 31(5):24, September 1993.

[3] H. T. Banks and K. Kunisch. *Estimation Techniques for Distributed Parameter Systems*, volume 1 of *Systems & Control: Foundations & Applications*. Birkhäuser Boston, Boston, MA, 1989.

[4] Ian Knowles. Parameter identification for elliptic problems. *Journal of Computational and Applied Mathematics*, 131(1–2):175–194, June 2001.

[5] Giovanni Alessandrini. An identification problem for an elliptic equation in two variables. *Annali di Matematica Pura ed Applicata*, 145(1):265–295, December 1986.

[6] Habib Ammari, Pierre Garapon, and François Jouve. Separation of Scales in Elasticity Imaging: A Numerical Study. *Journal of Computational Mathematics*, 28(3):354–370, May 2010.

[7] H. W. Engl and P. Kugler. The influence of the equation type on iterative parameter identification problems which are elliptic or hyperbolic in the parameter. *European Journal of Applied Mathematics*, 14:129 – 163, April 2003.

[8] B. Jadamba, A. A. Khan, and M. Sama. Inverse problems of parameter identification in partial differential equations. pages 228–258. World Scientific, June 2011.

[9] Bangti Jin and Peter Maass. Sparsity regularization for parameter identification problems. *Inverse Problems*, 28(12):123001, December 2012.

[10] Jingzhi Li and Jun Zou. A multilevel model correction method for parameter identification. *Inverse Problems*, 23(5):1759–1786, 2007.

[11] Huipo Liu and Ningning Yan. Recovery type superconvergence and a posteriori error estimates for control problems governed by Stokes equations. *Journal of Computational and Applied Mathematics*, 209(2):187–207, December 2007.

[12] S. Manservisi and M. Gunzburger. A variational inequality formulation of an inverse elasticity problem. *Applied Numerical Mathematics*, 34(1):99–126, June 2000.

[13] L. W. White. Estimation of elastic parameters in beams and certain plates:H 1 regularization. *Journal of Optimization Theory and Applications*, 60(2):305–326, February 1989.

[14] M. S. Gockenbach, B. Jadamba, and A. A. Khan. Numerical estimation of discontinuous coefficients by the method of equation error. *ResearchGate*, 1:343–359, January 2006.

[15] Akhtar A. Khan and Mark S. Gockenbach. Identification of Lamé parameters in linear elasticity: a fixed point approach. *Journal of Industrial and Management Optimization*, 1(4):487–497, October 2005.

[16] Mark S. Gockenbach and Akhtar A. Khan. An Abstract Framework for Elliptic Inverse Problems: Part 2. An Augmented Lagrangian Approach. *Mathematics and Mechanics of Solids*, 14(6):517–539, August 2009.

[17] B. Jadamba, A. Khan, G. Rus, M. Sama, and B. Winkler. A New Convex Inversion Framework for Parameter Identification in Saddle Point Problems with an Application to the Elasticity Imaging Inverse Problem of Predicting Tumor Location. *SIAM Journal on Applied Mathematics*, 74(5):1486–1510, January 2014.

[18] Assad A. Oberai, Nachiket H. Gokhale, and Gonzalo R. Feijóo. Solution of inverse problems in elasticity imaging using the adjoint method. *Inverse Problems*, 19(2):297, 2003.

[19] Alexandru Cioaca, Mihai Alexe, and Adrian Sandu. Second-order Adjoints for Solving PDE-constrained Optimization Problems. *Optimization Methods Software*, 27(4-5):625–653, October 2012.

[20] Alexandru Cioaca and Adrian Sandu. An optimization framework to improve 4d-Var data assimilation system performance. *Journal of Computational Physics*, 275:377–389, October 2014.

[21] N. Dominguez, V. Gibiat, and Y. Esquerre. Time domain topological gradient and time reversal analogy: an inverse method for ultrasonic target detection. *Wave Motion*, 42(1):31–52, June 2005.

[22] Drosos Kourounis, Louis J. Durlofsky, Jan Dirk Jansen, and Khalid Aziz. Adjoint formulation and constraint handling for gradient-based optimization of compositional reservoir flow. *Computational Geosciences*, 18(2):117–137, January 2014.

[23] Robert Michael Lewis. Numerical Computation of Sensitivities and the Adjoint Approach. In Jeff Borggaard, John Burns, Eugene Cliff, and Scott Schreck, editors, *Computational Methods for Optimal Design and Control*, number 24 in Progress in Systems and Control Theory, pages 285–302. Birkhäuser Boston, 1998. DOI: 10.1007/978-1-4612-1780-0_16.

[24] Edgar Reséndiz and René Pinnau. Adjoint-based optimization of particle trajectories in laminar flows. *Applied Mathematics and Computation*, 248:567–583, December 2014.

[25] D. A. Knopoff, D. R. Fernández, G. A. Torres, and C. V. Turner. Adjoint method for a tumor growth PDE-constrained optimization problem. *Computers & Mathematics with Applications*, 66(6):1104–1119, October 2013.

[26] D. A. Tortorelli and P. Michaleris. Design sensitivity analysis: Overview and review. *Inverse Problems in Engineering*, 1(1):71–105, October 1994.

[27] Tao Feng, Ningning Yan, and Wenbin Liu. Adaptive finite element methods for the identification of distributed parameters in elliptic equation. *Advances in Computational Mathematics*, 29(1):27–53, July 2008.

[28] Yanning Zhu, T. J. Hall, and Jingfeng Jiang. A finite-element approach for Young's modulus reconstruction. *IEEE Transactions on Medical Imaging*, 22(7):890–901, July 2003.

[29] I. Bijelonja, I. Demirdžić, and S. Muzaferija. A finite volume method for incompressible linear elasticity. *Computer Methods in Applied Mechanics and Engineering*, 195(44–47):6378–6390, September 2006.

[30] Brian C. Winkler. *Iterative methods for the elasticity imaging inverse problem*. PhD thesis, Rochester Institute of Technology, Rochester, NY, USA, July 2014.

[31] Daniele Boffi, Franco Brezzi, and Michel Fortin. *Mixed Finite Element Methods and Applications*. Springer, Heidelberg ; New York, 2013 edition edition, July 2013.

[32] B. Jadamba, A. A. Khan, and F. Raciti. On the inverse problem of identifying Lamé coefficients in linear elasticity. *Computers & Mathematics with Applications*, 56(2):431–443, July 2008.

[33] Georg Pingen, Anton Evgrafov, and Kurt Maute. Adjoint parameter sensitivity analysis for the hydrodynamic lattice Boltzmann method with applications to design optimization. *Computers & Fluids*, 38(4):910–923, April 2009.

[34] Shou-Lei Wang, Yu-Fei Yang, and Yu-Hua Zeng. The Adjoint Method for the Inverse Problem of Option Pricing, The Adjoint Method for the Inverse Problem of Option Pricing. *Mathematical Problems in Engineering, Mathematical Problems in Engineering*, 2014, 2014:e314104, March 2014.

[35] X. Ye, P. Li, and F. Y. Liu. Exact Time-Domain Second-Order Adjoint-Sensitivity Computation for Linear Circuit Analysis and Optimization. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 57(1):236–248, January 2010.

[36] Jakob S. Jensen, Praveen B. Nakshatrala, and Daniel A. Tortorelli. On the consistency of adjoint sensitivity analysis for structural optimization of linear dynamic problems. *Structural and Multidisciplinary Optimization*, 49(5):831–837, November 2013.

[37] Oleg Volkov, Bartosz Protas, Wenyuan Liao, and Donn W. Glander. Adjoint-based optimization of thermo-fluid phenomena in welding processes. *Journal of Engineering Mathematics*, 65(3):201–220, April 2009.

[38] Dimitrios I. Papadimitriou and Kyriakos C. Giannakoglou. Aerodynamic Shape Optimization Using First and Second Order Adjoint and Direct Approaches. *Archives of Computational Methods in Engineering*, 15(4):447–488, December 2008.

[39] David A. Boger and Eric G. Paterson. A continuous adjoint approach to design optimization in cavitating flow using a barotropic model. *Computers & Fluids*, 101:155–169, September 2014.

[40] K. Arens, P. Rentrop, S. O. Stoll, and U. Wever. An adjoint approach to optimal design of turbine blades. *Applied Numerical Mathematics*, 53(2–4):93–105, May 2005.

[41] Graeme J. Kennedy and Jorn S. Hansen. The hybrid-adjoint method: a semi-analytic gradient evaluation technique applied to composite cure cycle optimization. *Optimization and Engineering*, 11(1):23–43, November 2008.

[42] Geng Liu, Martin Geier, Zhenyu Liu, Manfred Krafczyk, and Tao Chen. Discrete adjoint sensitivity analysis for fluid flow topology optimization based on the generalized lattice Boltzmann method. *Computers & Mathematics with Applications*, 68(10):1374–1392, November 2014.

[43] Carlos Lozano. Discrete surprises in the computation of sensitivities from boundary integrals in the continuous adjoint approach to inviscid aerodynamic shape optimization. *Computers & Fluids*, 56:118–127, March 2012.

[44] M. Namdar Zanganeh, J. F. B. M. Kraaijevanger, H. W. Buurman, J. D. Jansen, and W. R. Rossen. Challenges in adjoint-based optimization of a foam EOR process. *Computational Geosciences*, 18(3-4):563–577, May 2014.

[45] M. U. Altaf, M. El Gharamti, A. W. Heemink, and I. Hoteit. A reduced adjoint approach to variational data assimilation. *Computer Methods in Applied Mechanics and Engineering*, 254:1–13, February 2013.

[46] W. Bangerth, T. Heister, L. Heltai, G. Kanschat, M. Kronbichler, M. Maier, and B. Turcksin. The `deal.II` library, version 8.3. *Archive of Numerical Software*, 4(100):1–11, 2016.

[47] W. Bangerth, D. Davydov, T. Heister, L. Heltai, G. Kanschat, M. Kronbichler, M. Maier, B. Turcksin, and D. Wells. The `deal.II` library, version 8.4. *preprint*.

[48] Wolfgang Bangerth and Amit Joshi. Adaptive Finite Element Methods for Nonlinear Inverse Problems. In *Proceedings of the 2009 ACM Symposium on Applied Computing*, SAC '09, pages 1002–1006, New York, NY, USA, 2009. ACM.