Marcus Meyer

Parameter identification problems for elastic large deformations – Part II: numerical solution and results

CSC/09-06

Impressum:

Chemnitz Scientific Computing Preprints — ISSN 1864-0087
(1995–2005: Preprintreihe des Chemnitzer SFB393)

Herausgeber:
Professuren für
Numerische und Angewandte Mathematik
an der Fakultät für Mathematik
der Technischen Universität Chemnitz

Postanschrift:
TU Chemnitz, Fakultät für Mathematik
09107 Chemnitz
Sitz:
Reichenhainer Str. 41, 09126 Chemnitz

`http://www.tu-chemnitz.de/mathematik/csc/`

TECHNISCHE UNIVERSITÄT CHEMNITZ

Chemnitz Scientific Computing Preprints

Marcus Meyer

Parameter identification problems for elastic large deformations – Part II: numerical solution and results

CSC/09-06

Abstract

In this paper we continue the considerations of [5]. A numerical study for the parameter identification problem with linear elastic material and large deformations is presented. We discuss the numerical implementation and illustrate some results for a 2D test problem.

CSC/09-06 ISSN 1864-0087 September 2009

Contents

Author's addresses:

Marcus Meyer
TU Chemnitz
Fakultät für Mathematik
D-09107 Chemnitz

`http://www.tu-chemnitz.de/mathematik/`
`marcus.meyer@mathematik.tu-chemnitz.de`

1 Introduction

With this paper we proceed the considerations of [5], where the identification of material parameters in a large deformation framework was discussed. Now, we focus on the numerical solution of the inverse identification problem for linear elastic material. For simplicity, the identification problem was restricted to a 2D framework.

In this study FEM with uniform meshes was applied for solving the underlying nonlinear PDE. Due to the large number of necessary forward operator calls during the identification process, this leads to immense numerical costs. Thus, the aim of our work is, to present a concept for implementing adaptive FE methods, which promises a considerable increase of efficiency.

The paper is organized as follows. A survey on the identification problem and the solution algorithm is presented in section 2. The numerical implementation is discussed in section 3, where in the first part of section 3 we focus on some basics in tensor calculus, concerning the calculation of tensor products. The 4th section is devoted to numerical results for the Cook membrane test problem. Finally, in the last section an outlook on future work is given. In this context a survey on the implementation of adaptive FEM within the identification procedure is presented and emerging problems and ideas in the concept of adaptivity are discussed.

2 Model and solution of the inverse problem

We shortly sketch the model of the identification problem [IP] with linear elastic material. See [5] for an extensive discussion of details. We also refer to [5] for an explanation of the used notation.

The forward problem consist of finding a displacement U as a solution of the variational problem

$$\int\limits_{\Omega_0} \overset{2}{T} : E(U;V)\mathrm{d}\Omega_0 = \int\limits_{\Omega_0} \rho_0 \vec{f} \cdot V \mathrm{d}\Omega_0 + \int\limits_{\Gamma_{N_0}} \vec{g} \cdot V \mathrm{d}S_0 \quad \forall V \in (H_0^1(\Omega_0))^3 \,. \tag{1}$$

For the linear elastic material law, the 2nd Piola-Kirchhoff stress tensor is definded as

$$\overset{2}{T} = 2\mu E(U) + \lambda((\mathrm{tr}\ E(U))I) = \mathbb{C} : E(U) \tag{2}$$

with a 4th-order material tensor $\mathbb{C} = \mathbb{C}(\lambda, \mu)$. Hence, the material parameter to be identified is

$$p(X) := (\lambda(X), \mu(X)), \quad X \in \Omega_0 \,.$$

For simplicity we omit volume forces ($\vec{f} = 0$) and rewrite (1) as

$$a(U;V|p) = f(V) \quad \forall V \in Z = (H_0^1(\Omega_0))^3 \tag{3}$$

with the functionals

$$a(U;V|p) := \int\limits_{\Omega_0} \overset{2}{T} : E(U;V) \mathrm{d}\Omega_0 \tag{4}$$

and

$$f(V) = \int\limits_{\Gamma_{N_0}} \vec{g} \cdot V \mathrm{d}S_0 \ . \tag{5}$$

Due to the nonlinearity of $a(U;V|p)$ in U, the problem (3) is nonlinear and has to be solved with a Newton linearization (see [5, algorithm 2.1]). The linearization of $a(U;V|p)$ for linear elasticity holds

$$a_0(U;W,V|p) := \int\limits_{\Omega_0} \left[E(U;W) : \mathbb{C}(p) : E(U;V) + \left(\overset{2}{T}(U) \cdot \mathrm{Grad}W \right) : \mathrm{Grad}V \right] \mathrm{d}\Omega_0 \ . \tag{6}$$

Note, that $a(U;V|p)$ is linear in p. The inverse problem [IP] of finding p for given displacement data U_{data} can be solved via an SQP iteration [5, algorithm 3.1]. At this, in each iteration the SQP update for a given iterate (U_k, p_k, ξ_k) is defined as

$$U_{k+1} = U_k + \Delta U, \ p_{k+1} = p_k + \Delta p, \ \xi_{k+1} = \xi$$

with an iteration update $(\Delta U, \Delta p)$ solving the linearized system

$$\begin{array}{rcll} \alpha_k b_p(\Delta p, q) + a_0(U_k; \Delta U, \xi_k|q) + a(U_k; \xi|q) & = & \alpha_k b_p(q, p_k^*) & \forall q \in Q \\ a_0(U_k; \xi_k, V|\Delta p) + b_U(\Delta U, V) & & & \forall V \in Z \\ +a_1(U_k; \Delta U, \xi_k, V|p_k) + a_0(U_k; \xi, V|p_k) & = & \langle U_{data} - \mathcal{P}U_k, \mathcal{P}V \rangle_{L^2(\Omega_0)} & \\ a(U_k; W|\Delta p) + a_0(U_k; \Delta U, W|p_k) & = & f(W) - a(U_k; W|p_k) & \forall W \in Z \end{array} \tag{7}$$

In this context the second order linearization of $a(U;V|p)$ is introduced as

$$\begin{aligned} a_1(U;\Delta U;W,V|p) \ := \ & \int\limits_{\Omega_0} \Big[\big((E(U;\Delta U) : \mathbb{C}) \cdot \mathrm{Grad}W \big) : \mathrm{Grad}V \\ & + \big((E(U;W) : \mathbb{C}) \cdot \mathrm{Grad}V \big) : \mathrm{Grad}\Delta U \\ & + \big((E(U;V) : \mathbb{C}) \cdot \mathrm{Grad}\Delta U \big) : \mathrm{Grad}W \Big] \mathrm{d}\Omega_0 \ . \end{aligned} \tag{8}$$

3 Numerical implementation

3.1 Basics in tensor calculus

In the following we sketch the calculation of tensor products in orthonormal bases. See [6] for details on tensor analysis. Note, that for a FEM discretization of (1) the tensor coordinates have to be implemented w.r.t. a fixed basis in the domain Ω_0. Due to the considerations of [4], it is allowed w.l.o.g. to introduce an orthonormal tensor basis in Ω_0.

For a clear notation, we sign in this section an nth order tensor by n times underlining. In the further sections the underlining is omitted. Additionally we assume, that Einstein's summation convention holds

Definition 3.1 (Einstein's summation convention)**.** *If an index variable appears twice in a monomial (single term), then we have to sum over all possible values for this index.*

E. g. let $i = 1, \ldots, d$ with d denoting the dimension of the Euclidian space $\mathbb{X} = \mathbb{R}^d$. Then the standard scalar product of two vectors in $\mathbb{X}$ can be written as

$$a_i b_i := \sum_{i=1}^{d} a_i b_i = a_1 b_1 + \ldots + a_d b_d = \langle \underline{a}, \underline{b} \rangle_{\mathbb{R}^d} \ .$$

A Tensor A of k-th order is defined as

$$A(X) = a_{i_1,\ldots,i_k}(X) \underline{e}_{i_1} \ldots \underline{e}_{i_k}$$

with coordinates $a_{i_1,\ldots,i_k}(X) \in \mathbb{R}$, $X \in \mathbb{X}$ and the k-fold tensor product of basis vectors $\underline{e}_{i_1} \ldots \underline{e}_{i_k}$, $i_1, \ldots, i_k \in \{1, \ldots, \dim \mathbb{X}\}$, denoting a tensor basis for tensors of k-th order. According to (1) we assume in the following an underlying 3-dimensional Euclidian space $\mathbb{X} = \mathbb{R}^3 \supset \Omega_0$ with an orthonormal basis $\underline{e}_1, \underline{e}_2, \underline{e}_3$.

We define tensor products, in particular the contraction of tensors. Let $\underline{\underline{A}}, \underline{\underline{B}}$ denote tensors of second order. The double contraction of these tensors is defined as

$$\underline{\underline{A}} : \underline{\underline{B}} = D \quad \text{with} \quad D = a_{ij} b_{ji} = \sum_{i,j=1}^{3} a_{ij} b_{ji} \ .$$

The tensor D is a tensor of zero order, i.e. scalar. Using the coordinates of $\underline{\underline{A}}$ and $\underline{\underline{B}}$, the double contraction of two second order tensors can be written in matrix form as

$$a_{ij} b_{ji} = \begin{pmatrix} a_{11} \\ a_{12} \\ a_{13} \end{pmatrix}^T \begin{pmatrix} b_{11} \\ b_{21} \\ b_{31} \end{pmatrix} + \begin{pmatrix} a_{21} \\ a_{22} \\ a_{23} \end{pmatrix}^T \begin{pmatrix} b_{12} \\ b_{22} \\ b_{32} \end{pmatrix} + \begin{pmatrix} a_{31} \\ a_{32} \\ a_{33} \end{pmatrix}^T \begin{pmatrix} b_{13} \\ b_{23} \\ b_{33} \end{pmatrix} \ .$$

Analogously, the double contraction of a second order and a fourth order tensor holds

$$\underline{\underline{A}} : \underline{\underline{\underline{\underline{C}}}} = \underline{\underline{D}} \quad \text{with} \quad d_{kl} = a_{ij}c_{jikl} = \sum_{i,j=1}^{3} a_{ij}c_{jikl} \ ,$$

where $\underline{\underline{D}}$ is a second order tensor with coordinates d_{kl}. Finally we consider two cases of single contraction of tensors. At one hand the contraction of tensors of second and first order is defined via

$$\underline{\underline{A}} \cdot \underline{B} = \underline{D} \quad \text{with} \quad d_i = a_{ij}b_j = \sum_{j=1}^{3} a_{ij}b_j$$

with a first order tensor $\underline{D}$ with coordinates

$$\begin{pmatrix} d_1 \\ d_2 \\ d_3 \end{pmatrix} := \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} .$$

At the other hand the contraction of two first order tensors holds

$$\underline{A} \cdot \underline{B} = D \quad \text{with} \quad D = a_j b_j = \sum_{j=1}^{3} a_j b_j \ .$$

From now on we omit the underlining of tensors. Note, that in the following considerations underlined variables will denote vectors and matrices.

We discuss specific tensors arising in [IP]. The strain tensor $E(U)$ and its Fréchet derivative $E(U;V)$ were introduced as

$$\begin{aligned} 2E(U) &= \text{Grad}U + \text{Grad}U^T + \text{Grad}U \cdot \text{Grad}U^T \\ 2E(U;V) &= \text{Grad}V + \text{Grad}V^T + \text{Grad}U \cdot \text{Grad}V^T + \text{Grad}V \cdot \text{Grad}U^T \end{aligned}$$

with the second order tensors

$$\text{Grad } U(X) = \frac{\partial U_j(X)}{\partial X_i} \vec{e}_i \vec{e}_j$$

and

$$\text{Grad } U(X) \ \cdot \ \text{Grad } V(X)^T = \frac{\partial U_j(X)}{\partial X_i} \frac{\partial V_j(X)}{\partial X_k} \vec{e}_i \vec{e}_k \ .$$

Thus, $E(U)$ and $E(U;V)$ are symmetric second order tensors and for the coordinates of $D = \text{Grad } U \cdot \text{Grad } V^T$ holds the relation

$$d_{ij} = \langle U_{,i}, V_{,j} \rangle_{\mathbb{R}^3} = U_{,i}^T V_{,j}, \quad i,j = 1,2,3 \ ,$$

where

$$U_{,i} := \left(\frac{\partial U_1}{\partial X_i}, \frac{\partial U_2}{\partial X_i}, \frac{\partial U_3}{\partial X_i}\right)^T$$

denotes the partial derivative of U w.r.t. X_i. The coordinates of $E(U)$ and $E(U;V)$ are calculated via

$$2e(U)_{ij} = \frac{\partial U_j}{\partial X_i} + \frac{\partial U_i}{\partial X_j} + \frac{\partial U_k}{\partial X_i}\frac{\partial U_k}{\partial X_j} = U_{j,i} + U_{i,j} + U_{,i}^T U_{,j} \tag{9}$$

and

$$\begin{aligned} 2e(U;V)_{ij} &= \frac{\partial V_j}{\partial X_i} + \frac{\partial V_i}{\partial X_j} + \frac{\partial U_k}{\partial X_i}\frac{\partial V_k}{\partial X_j} + \frac{\partial V_k}{\partial X_i}\frac{\partial U_k}{\partial X_j} \\ &= V_{j,i} + V_{i,j} + U_{,i}^T V_{,j} + V_{,i}^T U_{,j} \ . \end{aligned} \tag{10}$$

We shortly discuss, how to define the fourth order material tensor $\mathbb{C} := \mathbb{C}(\lambda, \mu)$ appropriately. We derive from (2), that at one hand the coordinates of $\overset{2}{T}$ are expressed in terms of $\mathbb{C}$ as

$$\overset{2}{t}_{ij} = c_{ijkl} e(U)_{lk}$$

and otherwise they have to fulfill

$$\begin{aligned} \overset{2}{t}_{11} &= 2\mu e(U)_{11} + \lambda \left[e(U)_{11} + e(U)_{22} + e(U)_{33}\right] = \sum_{k,l=1}^{3} c_{11kl} e(U)_{lk} \\ &\Rightarrow \quad c_{1111} = 2\mu + \lambda, \quad c_{1122} = c_{1133} = \lambda, \quad c_{11kl} = 0 \ \forall k \neq l \\ \overset{2}{t}_{12} &= 2\mu e(U)_{12} = \sum_{k,l=1}^{3} c_{12kl} e(U)_{lk} \\ &\Rightarrow \quad c_{1221} = 2\mu, \quad c_{12kl} = 0 \text{ otherwise} \\ &\cdots \quad , \end{aligned}$$

which defines $\mathbb{C}$. Additionally we mention, that in [5] the definition (2) of the 2nd Piola Kirchhoff tensor was derived as

$$\overset{2}{T} = 2\frac{\partial \Psi}{\partial G} \ .$$

If an orthonormal tensor basis is introduced, this is equivalent to

$$\overset{2}{T} = 2\frac{\partial \Psi}{\partial G} = 2\left(\frac{\partial \Psi(G)}{\partial g_{ij}}\right)\vec{e}_i\vec{e}_j \ ,$$

which means, that the coordinates of $\overset{2}{T}$ are derivatives of Ψ w.r.t. the coordinates of G.

Concluding this section, we discuss, how the the variational problem (1) can be formulated in terms of the tensor coordinates. We introduce the vectors

$$\underline{E}(U) := \begin{pmatrix} e(U)_{11} \\ e(U)_{22} \\ e(U)_{33} \\ 2e(U)_{12} \\ 2e(U)_{13} \\ 2e(U)_{23} \end{pmatrix} = \begin{pmatrix} U_{1,1} + \frac{1}{2} U_{,1}^T U_{,1} \\ U_{2,2} + \frac{1}{2} U_{,2}^T U_{,2} \\ U_{3,3} + \frac{1}{2} U_{,3}^T U_{,3} \\ U_{1,2} + U_{2,1} + U_{,1}^T U_{,2} \\ U_{1,3} + U_{3,1} + U_{,1}^T U_{,3} \\ U_{2,3} + U_{3,2} + U_{,2}^T U_{,3} \end{pmatrix} \tag{11}$$

and

$$\underline{E}(U;V) := \begin{pmatrix} e(U;V)_{11} \\ e(U;V)_{22} \\ e(U;V)_{33} \\ 2e(U;V)_{12} \\ 2e(U;V)_{13} \\ 2e(U;V)_{23} \end{pmatrix} = \begin{pmatrix} V_{1,1} + \frac{1}{2} U_{,1}^T V_{,1} \\ V_{2,2} + \frac{1}{2} U_{,2}^T V_{,2} \\ V_{3,3} + \frac{1}{2} U_{,3}^T V_{,3} \\ V_{1,2} + V_{2,1} + U_{,1}^T V_{,2} + U_{,2}^T V_{,1} \\ V_{1,3} + V_{3,1} + U_{,1}^T V_{,3} + U_{,3}^T V_{,1} \\ V_{2,3} + V_{3,2} + U_{,2}^T V_{,3} + U_{,3}^T V_{,2} \end{pmatrix} , \tag{12}$$

referring to the coordinates of $E(U)$ and $E(U;V)$. Additionally we define the matrix

$$\underline{\mathbb{C}} := \begin{pmatrix} 2\mu+\lambda & \lambda & \lambda & 0 & 0 & 0 \\ \lambda & 2\mu+\lambda & \lambda & 0 & 0 & 0 \\ \lambda & \lambda & 2\mu+\lambda & 0 & 0 & 0 \\ 0 & 0 & 0 & \mu & 0 & 0 \\ 0 & 0 & 0 & 0 & \mu & 0 \\ 0 & 0 & 0 & 0 & 0 & \mu \end{pmatrix} , \tag{13}$$

which contains the coordinates of the tensor $\mathbb{C}$. Then the semilinear functional $a(U;V|p)$ holds

$$a(U;V|p) = \int\limits_{\Omega_0} \underline{E}(U)^T \underline{\mathbb{C}}\, \underline{E}(U;V) \mathrm{d}\Omega_0 \,, \tag{14}$$

if we assume, that an underlying orthonormal tensor basis is fixed. Note, that analogously the functionals $a_0(U;W,V|p)$ and $a_1(U;\Delta U,W,V|p)$ can be written in terms of the tensor coordinates. We omit details here, due to the fact, that this results in extensive but straight forward calculations with tensor contractions in coordinate form as introduced at the beginning of this section.

3.2 Implementation of a two-dimensional model problem

In order to simplify the numerical calculations, we want to introduce a model problem, which can be reduced from a three-dimensional domain $\Omega_0 \subset \mathbb{R}^3$ to a two-dimensional domain $\Omega_0 \subset \mathbb{R}^2$.

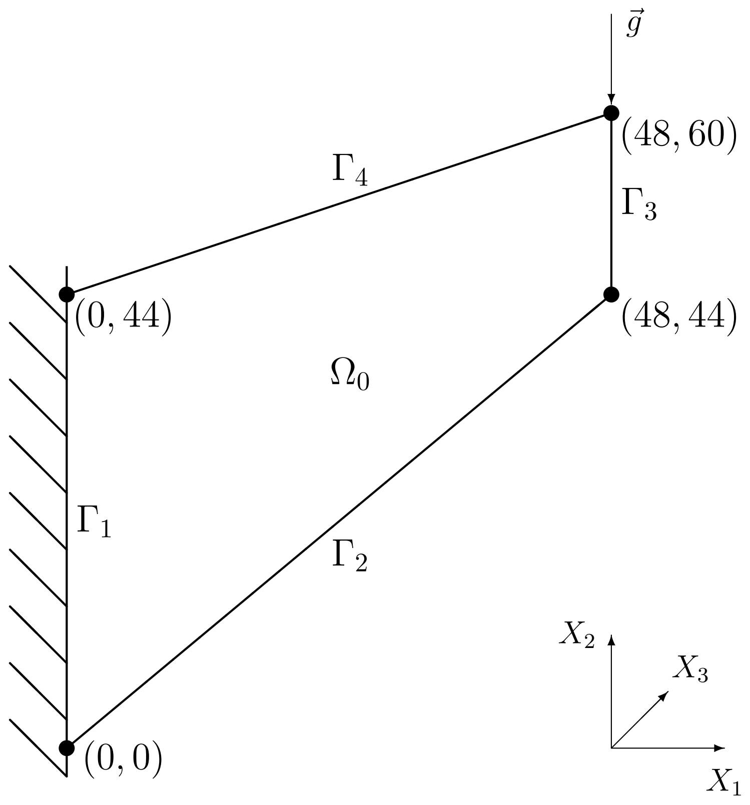

Figure 1: 2D Cook membrane

In our study we focus on the Cook membrane test problem which is widely used in numerical studies on structural mechanics (see e.g [2]). As displayed in figure 1, the geometry of the Cook membrane is given as a trapezoidal profile which is constant in X_3-direction. Thus

$$\Omega_0^{3D} = \Omega_0 \times [0, d], \quad \Omega_0 \subset \mathbb{R}^2$$

holds with a thickness d of the membrane. We assume, that the load $\vec{g}$ is constant w.r.t. X_3 and hence the deformation of the membrane is constant over X_3 as well, i. e.

$$U = U(X_1, X_2) \quad \Rightarrow \quad U_3 \equiv 0, \quad U_{i,j} \equiv 0 \text{ if } i = 3 \text{ or } j = 3\,. \tag{15}$$

Consequently, if the test functions V fulfill (15), the vectors $\underline{E}(U)$ and $\underline{E}(U;V)$ and the matrix $\underline{\mathbb{C}}$ can be simplified by removing rows and columns, which refer to X_3. Due to the fact, that $\underline{E}(U)^T \underline{\mathbb{C}}\, \underline{E}(U;V)$ is not depending on X_3 we derive under the assumption (15) that

$$a(U;V|p) = d \int\limits_{\Omega_0} \underline{E}(U)^T \underline{\mathbb{C}}\, \underline{E}(U;V) \mathrm{d}\Omega_0 \quad \Omega_0 \subset \mathbb{R}^2$$

holds. Finally, in (1) the thickness d can be canceled and the variational problem is two-dimensional. In an analogous way, the functionals $a_0(U;W,V|p)$ and $a_1(U;\Delta U,W,V|p)$ can be simplified. Note, that despite the independence of X_3 the stress tensor $\mathbb{C}:E(U)$ denotes a three-dimensional stress tensor.

3.3 FE discretization

We consider a discretization of [IP]. For simplicity we assume the two-dimensional situation with (15). Let a triangulation $\mathcal{T}$ of $\Omega_0 \subset \mathbb{R}^2$ be given with n_U nodes τ_i , $i = 1,\ldots,n_U$, and FE ansatz functions $\varphi_1,\ldots,\varphi_{n_U}$, such that $\varphi(\tau_j) = \delta_{ij}$, $1 \leq i,j \leq n_U$. Here, δ_{ij} denotes the Kronecker symbol

$$\delta_{ij} = \begin{cases} 1, & i = j; \\ 0, & i \neq j. \end{cases}$$

We define the FE space

$$\mathbb{V}^{(2n_U)} := [\mathrm{span}\{\varphi_1,\ldots,\varphi_{n_U}\}]^2$$

as a finite dimensional subspace of $Z = (H^1(\Omega_0))^2$. Additionally we introduce subspaces $\mathbb{V}_D^{(2n_U)}$ and $\mathbb{V}_0^{(2n_U)}$ of $\mathbb{V}^{(2n_U)}$, which fulfill the corresponding inhomogeneous and homogeneous Dirichlet boundary conditions at Γ_{D_0}. Then an approximation of U is given as

$$U \approx \left(\sum_{i=1}^{n_U} U_1^{(i)}\varphi_i \, , \, \sum_{i=1}^{n_U} U_2^{(i)}\varphi_i \right)^T \in \mathbb{V}_D^{(2n_U)} \, ,$$

Thus, in the discretized context we identify U with the vector

$$\underline{U} := (U_1^{(1)},\ldots,U_1^{(n_U)},U_2^{(1)},\ldots,U_2^{(n_U)})^T \in \mathbb{R}^{2n_U} \, .$$

Analogously we discretize the parameter p with ansatz functions ψ_i, , $i = 1,\ldots,n_p$

$$p \approx \left(\sum_{i=1}^{n_p} \lambda^{(i)}\psi_i \, , \, \sum_{i=1}^{n_p} \mu^{(i)}\psi_i \right)^T \in \mathbb{Q}^{(2n_p)}$$

with $\mathbb{Q}^{(2n_p)} = [\mathrm{span}\{\psi_1,\ldots,\psi_{n_p}\}]^2$ as a subspace of $Q = [L^\infty(\Omega_0)]^2$. Let e. g. the ψ_i refer to a triangulation $\mathcal{T}_2$, where $\mathcal{T}$ is a refinement of $\mathcal{T}_2$ or vice versa. In a simple ansatz we may choose $\mathcal{T} = \mathcal{T}_2$. Finally we identify p with

$$\underline{p} := (\lambda^{(1)},\ldots,\lambda^{(n_p)},\mu^{(1)},\ldots,\mu^{(n_p)})^T \in \mathbb{R}^{2n_p} \, .$$

According to (7), we introduce matrices and vectors referring to the functionals $b_p(p,q)$, $b_U(U,V)$, $a(U;V|p)$, $a_0(U;W,V|p)$, $a_1(U;\Delta U,W,V|p)$, and $f(V)$. Let

$$\tilde{\varphi}_i := \begin{pmatrix} \varphi_i \\ 0 \end{pmatrix}, \quad \tilde{\varphi}_{i+n_U} := \begin{pmatrix} 0 \\ \varphi_i \end{pmatrix}, \qquad 1 \le i \le n_U$$

as well as

$$\tilde{\psi}_i := \psi_i, \quad \tilde{\psi}_{i+n_p} := \psi_i, \qquad 1 \le i \le n_p \ .$$

We define the matrices

$$\begin{aligned}
\underline{Q} = (q_{ij}) \in \mathbb{R}^{2n_p \times 2n_p} \quad &: \quad q_{ij} := \int\limits_{\Omega_0} \tilde{\psi}_i \tilde{\psi}_j \, \mathrm{d}\Omega_0 \\
\underline{K} = (k_{ij}) \in \mathbb{R}^{2n_U \times 2n_U} \quad &: \quad k_{ij} := \int\limits_{\Omega_0} \tilde{\varphi}_i \tilde{\varphi}_j \, \mathrm{d}\Omega_0 \\
\underline{A}(\underline{U}) = (a_{ij}) \in \mathbb{R}^{2n_U \times 2n_p} \quad &: \quad a_{ij} := a(U;\tilde{\varphi}_i|\tilde{\psi}_j) \\
\underline{A}_0^k(\underline{U}) = (a_{ij}^k) \in \mathbb{R}^{2n_U \times 2n_U} \quad &: \quad a_{ij}^k := a_0(U;\tilde{\varphi}_i,\tilde{\varphi}_j|\tilde{\psi}_k), \quad k = 1,\dots,2n_p \\
\underline{A}_1^k(\underline{U},\underline{p}) = (\hat{a}_{ij}^k) \in \mathbb{R}^{2n_U \times 2n_U} \quad &: \quad \hat{a}_{ij}^k := a_1(U;\tilde{\varphi}_i,\tilde{\varphi}_k,\tilde{\varphi}_j|p), \quad k = 1,\dots,2n_U
\end{aligned}$$

and the vector

$$\underline{f} := (f(\tilde{\varphi}_1),\dots,f(\tilde{\varphi}_{2n_U}))^T \in \mathbb{R}^{2n_U} \ .$$

Additionally, in the discrete sense, the linear projection operator $\mathcal{P}$ picks the displacement data $\underline{U}_{data} \in \mathbb{R}^{n_{data}}$ from $\underline{U}$, where we bear in mind, that the n_{data} measuring points denote nodes of the mesh $\mathcal{T}$. Hence, it refers to the matrix

$$\underline{\mathcal{P}} := (p_{ij}) \in \mathbb{R}^{2n_{data} \times 2n_U} \quad \text{with}$$

$$\begin{cases} p_{ij} = p_{(i+n_{data})(j+n_U)} := 1, & \text{if the } i\text{th measuring point is the } j\text{th node in } \mathcal{T} \\ p_{ij} := 0, & \text{otherwise .} \end{cases}$$

Consequently, we derive for the terms arising in (7) the following discretizations

$$\begin{aligned}
b_p(\Delta p, q) \ \forall q \quad &\Rightarrow \quad \underline{Q}\Delta\underline{p} \\
b_U(\Delta U, V) \ \forall V \quad &\Rightarrow \quad \underline{P}^T \underline{P}\, \underline{K} \Delta \underline{U}
\end{aligned}$$

as well as

$$\begin{aligned}
a(U_k; W|\Delta p) \ \forall W \quad &\Rightarrow \quad \underline{A}(\underline{U}_k)\Delta\underline{p} \\
a(U_k; \xi|q) \ \forall q \quad &\Rightarrow \quad \underline{A}(\underline{U}_k)^T \underline{\xi} \ .
\end{aligned}$$

For the terms including first and second order linearizations of $a(U;V|p)$ we deduce

$$
\begin{aligned}
a_0(U_k;\Delta U,W|p_k)\;\forall W \quad &\Rightarrow \quad \underbrace{\left(\sum_{l=1}^{2n_p}(\underline{p}_k)_l\,\underline{A}_0^l(\underline{U}_k)\right)}_{=:\,\underline{\tilde{A}}_0(\underline{U}_k,\underline{p}_k)}\Delta\underline{U} \\
a_0(U_k;\xi,V|p_k)\;\forall V \quad &\Rightarrow \quad \underline{\tilde{A}}_0(\underline{U}_k,\underline{p}_k)\underline{\xi} = \underline{\tilde{A}}_0(\underline{U}_k,\underline{p}_k)^T\underline{\xi} \quad (\underline{\tilde{A}}_0(\underline{U}_k,\underline{p}_k) \text{ is symm.}) \\
a_0(U_k;\xi_k,V|\Delta p)\;\forall V \quad &\Rightarrow \quad \sum_{l=1}^{2n_p}(\Delta\underline{p})_l a_0(U_k;\xi_k,V|\tilde{\psi}_l)\;\forall V \\
&\Rightarrow \quad \sum_{l=1}^{2n_p}(\Delta\underline{p})_l\,\underline{A}_0^l(\underline{U}_k)\,\underline{\xi}_k = \underbrace{\left(\underline{A}_0^1(\underline{U}_k)\underline{\xi}_k,\ldots,\underline{A}_0^{2n_p}(\underline{U}_k)\underline{\xi}_k\right)}_{=:\,\underline{A}_0(\underline{U}_k,\underline{\xi}_k)}\Delta\underline{p} \\
a_0(U_k;\Delta U,\xi_k|q)\;\forall q \quad &\Rightarrow \quad \begin{pmatrix} \sum\limits_{i,j=1}^{2n_U}(\Delta\underline{U})_i(\underline{\xi}_k)_j\,a_0(U_k;\tilde{\varphi}_i,\tilde{\varphi}_j|\tilde{\psi}_1) \\ \vdots \\ \sum\limits_{i,j=1}^{2n_U}(\Delta\underline{U})_i(\underline{\xi}_k)_j\,a_0(U_k;\tilde{\varphi}_i,\tilde{\varphi}_j|\tilde{\psi}_{2n_p}) \end{pmatrix} \\
&\Rightarrow \quad \underbrace{\begin{pmatrix} (\underline{A}_0^1(\underline{U}_k)\underline{\xi}_k)^T \\ \vdots \\ (\underline{A}_0^{2n_p}(\underline{U}_k)\underline{\xi}_k)^T \end{pmatrix}}_{=\,\underline{A}_0(\underline{U}_k,\underline{\xi}_k)^T}\Delta\underline{U}
\end{aligned}
$$

and

$$
a_1(U_k;\Delta U,\xi_k,V|p_k)\;\forall V \quad \Rightarrow \quad \underbrace{\left(\sum_{l=1}^{2n_U}(\underline{\xi}_k)_l\,\underline{A}_1^l(\underline{U}_k,\underline{p}_k)\right)}_{=:\,\underline{\tilde{A}}_1(\underline{U}_k,\underline{\xi}_k,\underline{p}_k)}\Delta\underline{U}\;.
$$

Remark 3.1. The matrix $\underline{\tilde{A}}_0(\underline{U}_k,\underline{p}_k)$ can be calculated directly without any summation of $\underline{A}_0^l(\underline{U}_k)$ via

$$
\underline{\tilde{A}}_0(\underline{U}_k,\underline{p}_k) = (\bar{a}_{ij}) \in \mathbb{R}^{2n_U\times 2n_U} \;:\; \bar{a}_{ij}^k := a_0(U;\tilde{\varphi}_i,\tilde{\varphi}_j|p_k)\;,
$$

which may be more efficient than the summed definition. But despite this, it might be practicable to apply the summed definition due to the fact, that the $\underline{A}_0^l(\underline{U}_k)$ have to be calculated anyway. The same holds for $\underline{\tilde{A}}_1(\underline{U}_k,\underline{\xi}_k,\underline{p}_k)$.

Remark 3.2. The solution of the nonlinear forward problem (1) with a Newton linearization was introduced in [5] as algorithm 2.1. Applying the above definitions to algorithm 2.1, in each iteration we have to solve the discretized linear system

$$\underline{\tilde{A}}_0(\underline{U},\underline{p})\Delta\underline{U} = t\underline{f} - \underline{A}(\underline{U})\underline{p}, \qquad \Delta\underline{U} \in \mathbb{V}_0^{2n_U} \tag{16}$$

with $\Delta\underline{U} \in \mathbb{V}_0^{2n_U}$ fulfilling homogeneous Dirichlet boundary conditions at Γ_{D_0}.

Remark 3.3. In (16), the update vector $\Delta\underline{U}$ has to satisfy Dirichlet boundary conditions, which means, that (16) has to be solved under the additional constraint

$$\underline{H}\Delta\underline{U} = \underline{R}\,, \tag{17}$$

where $\underline{H}$ and $\underline{R}$ are appropriately chosen matrices and vectors. Two possibilities for solving (16) under (17) exist. First, the equations (17) may be eliminated in (16) or, in a second approach, (16) can be formulated as a minimization problem, where the constraint (17) is added with a Lagrange multiplier. See [9] for details.

Finally, for a given iterate $\underline{U}_k, \underline{\xi}_k, \underline{p}_k$ a discretization of the linear system (7) reads as

$$\begin{pmatrix} \alpha_k\underline{Q} & \underline{A}_0(\underline{U}_k,\underline{\xi}_k)^T & \underline{A}(\underline{U}_k)^T \\ \underline{A}_0(\underline{U}_k,\underline{\xi}_k) & \underline{P}^T\underline{P}\,\underline{K} + \underline{\tilde{A}}_1(\underline{U}_k,\underline{\xi}_k,\underline{p}_k) & \underline{\tilde{A}}_0(\underline{U}_k,\underline{p}_k)^T \\ \underline{A}(\underline{U}_k) & \underline{\tilde{A}}_0(\underline{U}_k,\underline{p}_k) & \underline{0} \end{pmatrix} \begin{pmatrix} \Delta\underline{p} \\ \Delta\underline{U} \\ \underline{\xi} \end{pmatrix} \tag{18}$$

$$= \begin{pmatrix} \alpha_k\underline{Q}\,\underline{p}_k^* \\ \underline{P}^T\underline{P}\,\underline{K}\,\underline{P}^T(\underline{U}_{data} - \underline{P}\,\underline{U}_k) \\ \underline{f} - \underline{A}(\underline{U}_k)\underline{p}_k \end{pmatrix} \qquad \underline{\xi}, \Delta\underline{U} \in \mathbb{V}_0^{2n_U}$$

and therefore the SQP iteration update is defined as

$$\underline{U}_{k+1} := \underline{U}_k + \Delta\underline{U}, \underline{p}_{k+1} := \underline{p}_k + \Delta\underline{p} \quad \text{and} \quad \underline{\xi}_{k+1} := \underline{\xi}\,.$$

Note, that $\underline{U}_k \in \mathbb{V}_D^{2n_U}$ and $\underline{\xi}, \Delta\underline{U} \in \mathbb{V}_0^{2n_U}$ have to fulfill Dirichlet boundary conditions at Γ_{D_0} coinciding with the definition of the forward problem. The statements of remark 3.3 can be applied analogously.

For the sake of completeness we shortly discuss the implementation of box conditions and the additional force measuring operator $\mathcal{G}$ (see [5, section 3.3] for details).

If the additional force measurement via $\mathcal{G}$ is included, we have to add several terms refering to [5, equations (37) and (38)] in the linear system (18). In this context we define a data vector

$$\begin{aligned} \underline{h}_{data} &:= \Big(\langle \vec{h}_{data}, \tilde{\varphi}_{i_1}\rangle_{L^2(\tilde{\Gamma}_{D_0})}, \dots, \langle \vec{h}_{data}, \tilde{\varphi}_{i_{n_h}}\rangle_{L^2(\tilde{\Gamma}_{D_0})}, \\ &= \langle \vec{h}_{data}, \tilde{\varphi}_{i_1+n_U}\rangle_{L^2(\tilde{\Gamma}_{D_0})}, \dots, \langle \vec{h}_{data}, \tilde{\varphi}_{i_{n_h}+n_U}\rangle_{L^2(\tilde{\Gamma}_{D_0})} \Big)^T, \end{aligned}$$

where the ansatz functions $\tilde{\varphi}_{i_j}, \tilde{\varphi}_{i_j+n_U}$, $j = 1, \ldots, n_h$, refer to n_h nodes located at $\tilde{\Gamma}_{D_0}$. Additionally we introduce the matrices

$$\underline{P}_{\mathcal{G}} := (\tilde{p}_{ij}) \in \mathbb{R}^{2n_h \times 2n_U} \quad : \quad \begin{cases} \tilde{p}_{ij} = 1, & \text{if } (\underline{h}_{data})_i \text{ refers to } \tilde{\varphi}_j \\ \tilde{p}_{ij} = 0, & \text{else .} \end{cases}$$

and

$$\underline{B}_U := (\underline{B}_U^1, \ldots, \underline{B}_U^{2n_U}) \in \mathbb{R}^{2n_U \times 2n_U}$$

defined via

$$\underline{B}_U^i = \underline{A}_1^i(\underline{U}_k, \underline{p}_k)^T \underline{P}_{\mathcal{G}}^T \left[\underline{P}_{\mathcal{G}}(\underline{A}(\underline{U}_k)\underline{p}_k - \underline{f}) - \underline{h}_{data} \right], \quad i = 1, \ldots, 2n_U$$

as well as

$$\underline{B}_p := (\underline{B}_p^1, \ldots, \underline{B}_U^{2n_p}) \in \mathbb{R}^{2n_U \times 2n_p}$$

given via

$$\underline{B}_p^i = \underline{A}_0^i(\underline{U}_k)^T \underline{P}_{\mathcal{G}}^T \left[\underline{P}_{\mathcal{G}}(\underline{A}(\underline{U}_k)\underline{p}_k - \underline{f}) - \underline{h}_{data} \right], \quad i = 1, \ldots, 2n_p \; .$$

Consequently, with a weight factor ϑ the added terms are

- 1st equation, left hand side

$$\vartheta \left[\underline{A}(\underline{U}_k)^T \underline{P}_{\mathcal{G}}^T \underline{P}_{\mathcal{G}} \left[\underline{\tilde{A}}_0(\underline{U}_k, \underline{p}_k)\Delta\underline{U} + \underline{A}(\underline{U}_k)\Delta\underline{p} \right] + \underline{B}_p^T \Delta\underline{U} \right]$$

- 1st equation, right hand side

$$\vartheta \left[A(\underline{U}_k)^T \underline{P}_{\mathcal{G}}^T \left[\underline{h}_{data} + \underline{P}_{\mathcal{G}}(\underline{f} - \underline{A}(\underline{U}_k)\underline{p}_k) \right] \right]$$

- 2nd equation, left hand side

$$\vartheta \left[\underline{\tilde{A}}_0(\underline{U}_k, \underline{p}_k)^T \underline{P}_{\mathcal{G}}^T \underline{P}_{\mathcal{G}} \left[\underline{\tilde{A}}_0(\underline{U}_k, \underline{p}_k)\Delta\underline{U} + \underline{A}(\underline{U}_k)\Delta\underline{p} \right] + \underline{B}_U \Delta\underline{U} + \underline{B}_p \Delta\underline{p} \right]$$

- 2nd equation, right hand side

$$\vartheta \left[\tilde{A}_0(\underline{U}_k, \underline{p}_k)^T \underline{P}_{\mathcal{G}}^T \left[\underline{h}_{data} + \underline{P}_{\mathcal{G}}(\underline{f} - \underline{A}(\underline{U}_k)\underline{p}_k) \right] \right]$$

Finally we sketch, how to implement the box conditions [5, equation (26)] for the parameter $\underline{p}$, which here means, that vectors $\underline{C}_l$ and $\underline{C}_u$ are given with

$$\underline{C}_l \leq \underline{p} \leq \underline{C}_u \; .$$

Thus, for a given iterate $\underline{p}_k$ we define the active sets referring to the lower and the upper bound as

$$\begin{aligned}\mathbb{A}_l &= \{i \ : \ (\underline{p}_k)_i < (\underline{C}_l)_i, \ i = 1, \dots, 2n_p\} \\ \mathbb{A}_u &= \{i \ : \ (\underline{p}_k)_i > (\underline{C}_u)_i, \ i = 1, \dots, 2n_p\} \ .\end{aligned}$$

If $\mathbb{A}_l$ and $\mathbb{A}_u$ are empty sets, the box conditions are not active and the linear system (18) needs not to be modified. Otherwise, we introduce the additional variables

$$\underline{\xi}_l \in \mathbb{R}^{|\mathbb{A}_l|} \quad \text{and} \quad \underline{\xi}_u \in \mathbb{R}^{|\mathbb{A}_u|}$$

and referring to the considerations in [5, section 3.3], we define the matrices

$$\begin{aligned}\underline{Q}_l := (q^l_{ij}) \in \mathbb{R}^{2n_p \times |\mathbb{A}_l|} \quad &: \int\limits_{\Omega_0} \tilde{\psi}_i \tilde{\psi}_j \, \mathrm{d}\Omega_0, \quad i \in \mathbb{A}_l, j = 1, \dots, 2n_p \\ \underline{Q}_u := (q^u_{ij}) \in \mathbb{R}^{2n_p \times |\mathbb{A}_u|} \quad &: \int\limits_{\Omega_0} \tilde{\psi}_i \tilde{\psi}_j \, \mathrm{d}\Omega_0, \quad i \in \mathbb{A}_u, j = 1, \dots, 2n_p\end{aligned}$$

and modify (18) as follows

- add in the first equation at the left hand side

$$-\underline{Q}_l \underline{\xi}_l + \underline{Q}_u \underline{\xi}_u$$

- add an equation referring to the lower bound

$$-\underline{Q}_l^T \Delta \underline{p} = \underline{Q}_l^T (\underline{p}_k - \underline{C}_l)$$

- add an equation referring to the upper bound

$$\underline{Q}_u^T \Delta \underline{p} = \underline{Q}_u^T (\underline{C}_u - \underline{p}_k) \ .$$

3.4 Implementation of [IP] with MATLAB

For the numerical studies presented in this paper, the SQP identification algorithm and therefore the solution of (18) was implemented in MATLAB including the PDE toolbox [9]. We mention, that the PDE toolbox provides limited skills for the solution of PDEs, as e.g. only linear ansatz functions φ_i are available, which increases numerical errors in the considered identification problem. Additionally, the applicability of MATLAB is restricted to 2D problems. In general, a variety of sophisticated FEM software exists and could be used as well. See in this context section 5.2 for notes on the use of extern FE code. Details on the appropriate choice of ansatz functions and calculation of stiffness matrices are presented e.g. in [1], [4], or [9] .

We discuss some facts of the MATLAB implementation. The matrices assembled for (18) are in general sparse matrices and therefor MATLAB provides the *sparse* function, which creates sparse matrices. Hence, an efficient storage of the system matrix in (18) is possible, if all submatrices are defined as sparse.

The PDE or the variational problem (14), resp., refer to an elliptic system (of order 2 in the 2D case). Those systems are implemented in MATLAB with a fourth order tensor $\mathfrak{C} = \mathfrak{C}(U, \lambda, \mu)$ such that the main part of the elliptic system is written as

$$-\nabla \cdot (\mathfrak{C} \otimes \nabla U) \; .$$

Consequently, for $\Omega_0 \subset \mathbb{R}^2$ from integrating by parts of

$$-\int\limits_{\Omega_0} (\nabla \cdot (\mathfrak{C} \otimes \nabla U)) V \mathrm{d}\Omega_0$$

follows the equivalence

$$a(U; V|p) \;\; = \;\; \int\limits_{\Omega_0} \underline{E}(U)^T \mathbb{C} \, \underline{E}(U; V) \mathrm{d}\Omega_0$$

$$= \int\limits_{\Omega_0} \begin{pmatrix} V_{1,1} \\ V_{1,2} \end{pmatrix}^T \left[\begin{pmatrix} \hat{c}_{1111} & \hat{c}_{1112} \\ \hat{c}_{1121} & \hat{c}_{1122} \end{pmatrix} \begin{pmatrix} U_{1,1} \\ U_{1,2} \end{pmatrix} + \begin{pmatrix} \hat{c}_{1211} & \hat{c}_{1212} \\ \hat{c}_{1221} & \hat{c}_{1222} \end{pmatrix} \begin{pmatrix} U_{2,1} \\ U_{2,2} \end{pmatrix} \right]$$
$$+ \begin{pmatrix} V_{2,1} \\ V_{2,2} \end{pmatrix}^T \left[\begin{pmatrix} \hat{c}_{2111} & \hat{c}_{2112} \\ \hat{c}_{2121} & \hat{c}_{2122} \end{pmatrix} \begin{pmatrix} U_{1,1} \\ U_{1,2} \end{pmatrix} + \begin{pmatrix} \hat{c}_{2211} & \hat{c}_{2212} \\ \hat{c}_{2221} & \hat{c}_{2222} \end{pmatrix} \begin{pmatrix} U_{2,1} \\ U_{2,2} \end{pmatrix} \right] \mathrm{d}\Omega_0 \; .$$

Here, the scalars $\hat{c}_{ijkl}$ denote the coordinates of $\mathfrak{C}$ and thus the tensor $\mathfrak{C}$ is defined by the last equation. See [9, chapter 3] for details.

Furthermore, the following MATLAB functions were used:

- *decsg* for defining the geometry Ω_0
- boundary conditions defined as explained in *assemb*
- assemble stiffness matrices with *assema*
- calculate a solution of a PDE with *assempde*

For details we refer to [9].

4 Numerical results

Throughout the numerical study presented in this section we assume the following framework:

- discretization with uniform meshes
- dicretization of U and p referring to equivalent triangulations $\mathcal{T} = \mathcal{T}_2$
- [IP] is restricted to the estimation of λ for a known μ (remove rows and columns referring to μ in the linear system (18))
- noiseless data U_{data}
- single measurements $n_{data} = 1$

Referring to the domain $\Omega_0 \subset \mathbb{R}^2$ as defined in figure 1 the boundary conditions are applied as

$$\begin{cases} \underline{U} = \underline{0}, & \text{on } \Gamma_1 \\ \vec{n}_0 \cdot \overset{1}{T}(\underline{U}) = \underline{0}, & \text{on } \Gamma_2 \cup \Gamma_4 \\ \vec{n}_0 \cdot \overset{1}{T}(\underline{U}) = \vec{g} = (0, -\frac{1}{16}F_{load})^T, & \text{on } \Gamma_3 \end{cases} \tag{19}$$

with $\Gamma_1 = \Gamma_{D_0}$, $\Gamma_2 \cup \Gamma_3 \cup \Gamma_4 = \Gamma_{N_0}$ and

$$\begin{cases} \underline{U} = \underline{0}, & \text{on } \Gamma_1 \\ \vec{n}_0 \cdot \overset{1}{T}(\underline{U}) = \underline{0}, & \text{on } \Gamma_2 \cup \Gamma_4 \\ \underline{U} = (-U_{load}, 0)^T, & \text{on } \Gamma_3 \end{cases} \tag{20}$$

with $\Gamma_1 = \Gamma_{D_0}$, $\Gamma_4 = \tilde{\Gamma}_{D_0}$, $\Gamma_2 \cup \Gamma_3 = \Gamma_{N_0}$. The exact parameter functions are set as

$$\lambda^\dagger := 100\left(1 + \frac{X_1}{50}\frac{X_2}{60}\right) \quad \text{and} \quad \mu^\dagger := 80\left(2 - \frac{X_1^2}{2500} + \frac{X_2}{60}\right) .$$

If all lengths are given in mm and the units of F_{load}, λ, μ are scaled as kN/mm^2, then the above defined parameters are similar to a steel material. In this context, integrating the load $\vec{g}$ in the Neumann boundary condition over Γ_3 will result in a force load F_{load} with unit kN.

The displacement data U_{data} is simulated as the solution of the forward problem (16) with a triangulation $\mathcal{T}$ consisting of 30976 elements ($n_U = n_p = 15737$). Here, for the incremental applying of loads, the time t is subdivided in 10 load steps $\Delta t = 0.1$. In order to avoid inverse crime, [IP] is solved at coarser discretization levels, namely with $\mathcal{T}$ consisting of 484 or 1936 elements. See figure 2 for an example of the solution of the direct problem with boundary condition (20).

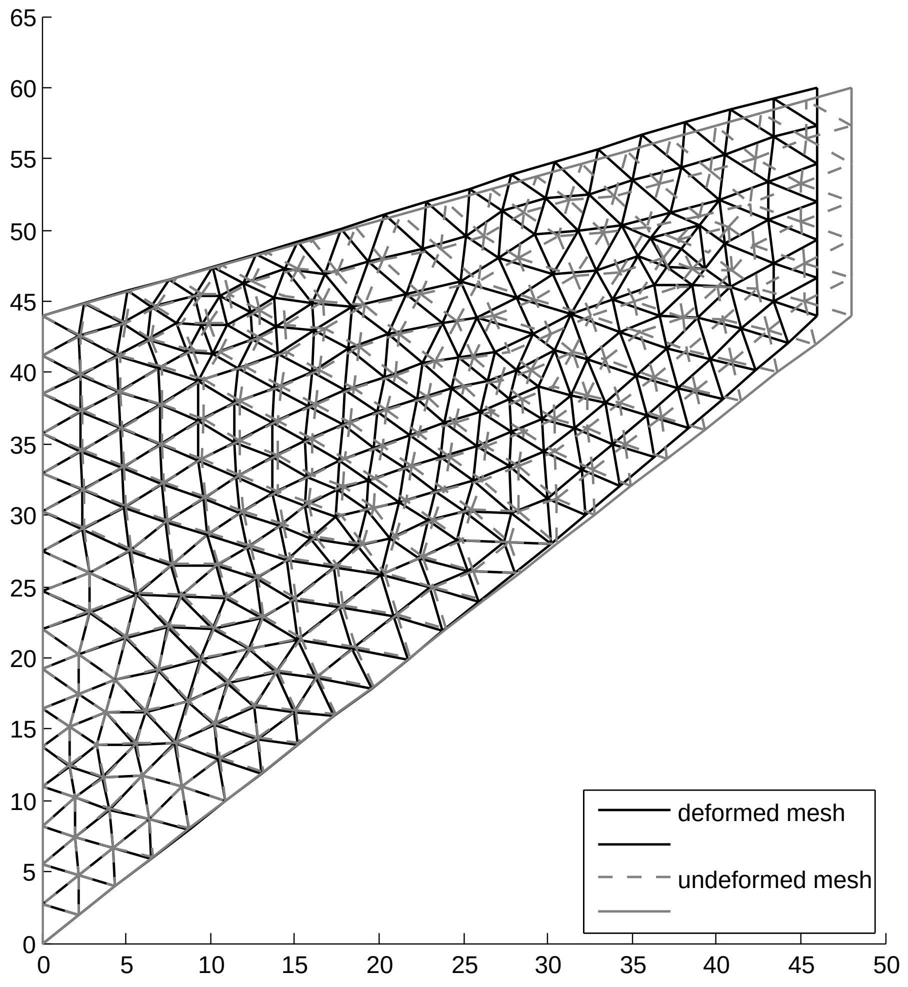

Figure 2: Solution of the direct problem: deformed and undeformed mesh of 2D test problem Cook-Membrane with boundary condition (20)

For solving [IP] via the SQP iteration algorithm (18), we introduce lower and upper bounds for the box conditions as

$$C_l = 100 = const \quad \text{and} \quad C_u = 200 = const$$

and set the initial guess

$$p_0 = \lambda_0 = const \quad \text{with} \quad 100 \leq \lambda_0 \leq 200 \; .$$

The SQP iteration is terminated at iteration K_0, if the relative residual norm and the relative norm of the parameter update $\Delta\underline{\lambda}$ are smaller than given tolerances, in particular i.e.

$$\frac{\|\underline{P}\,\underline{U}_{K_0} - \underline{U}_{data}\|}{\|\underline{U}_{data}\|} \leq tol_{res} = 10^{-3} \quad \text{and} \quad \frac{\|\Delta\underline{\lambda}\|}{\|\underline{\lambda}_{K_0}\|} \leq tol_p = 10^{-5} \; .$$

The relative identification error is calculated as

$$\frac{\left\|\underline{\lambda}_{K_0} - \underline{\lambda}^{\dagger}\right\|}{\left\|\underline{\lambda}^{\dagger}\right\|}\,.$$

Here we used the standard Euclidian vector norms.

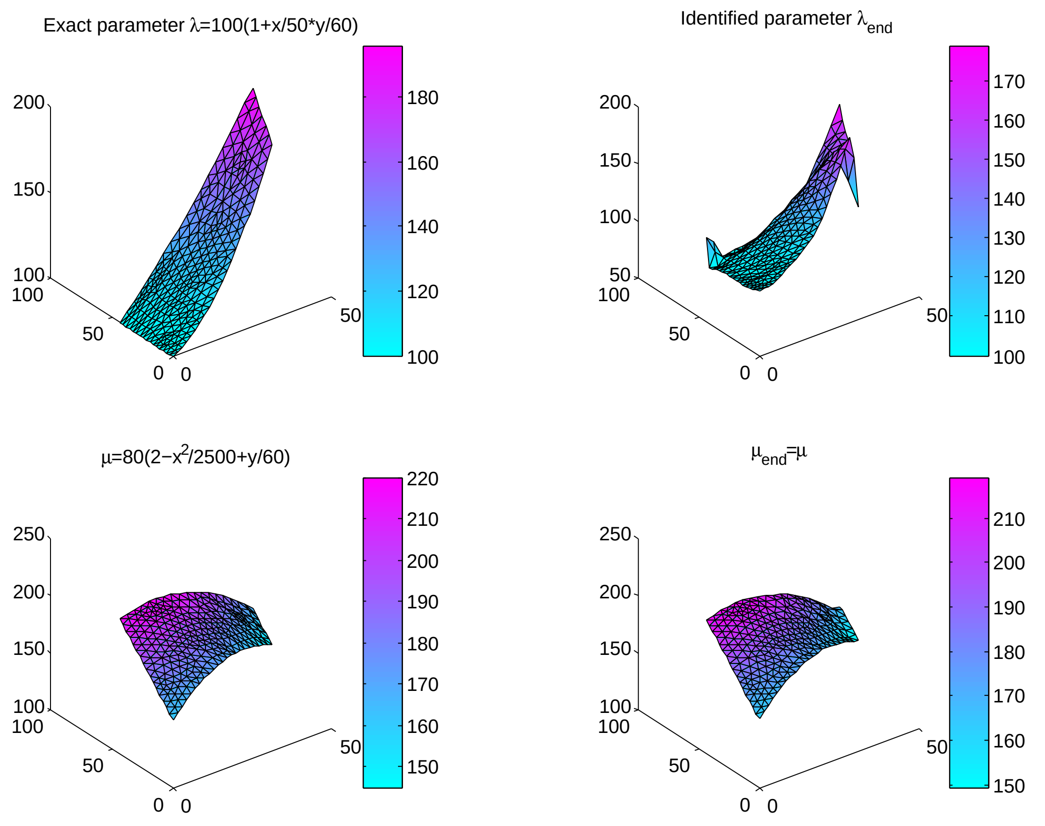

Figure 3: Identification of parameter λ with Tikhonov regularization, identification error $\approx 11.6\%$

An example for the solution of [IP] is given in figure 3. See table 4 for a concluding survey of detailed results. Here, the displacement data was given as the whole displacement field, i.e. $\underline{P}$ was set as the identity matrix. In table 4 results for the identification of λ with restricted data are presented.

As table 4 shows, the experimental design influences the quality of the results considerably. The identification error for boundary condition (19) is almost two times larger than for boundary condition (20). In the case of boundary condition (20) the additional force measurement can be applied by choosing $\vartheta > 0$, which improves the results slightly. As usual for identification problems, the choice of the initial guess influences the results as well. We mention, that all results are corrupted by numerical errors, due to the limited skills of the MATLAB PDE

$\sharp$elem.	boundary condition	ω	ϑ	regul. method	α	λ_0	$\sharp$iter.	ident. error
484	(20), $U_{load} = 2$	1	10^{-5}	Tikh.	10^{-7}	110	5	11.6411%
1936	(20), $U_{load} = 2$	1	10^{-5}	Tikh.	10^{-7}	110	5	10.4609%
1936	(20), $U_{load} = 2$	1	0	Tikh.	10^{-7}	110	5	10.5917%
484	(20), $U_{load} = 2$	1	0	Tikh.	10^{-7}	150	4	17.1042%
484	(20), $U_{load} = 2$	1	0	Lev.-Mar.	10^{-7}	110	15	12.5976%
484	(20), $U_{load} = 2$	1	10^{-4}	Lev.-Mar.	10^{-7}	110	15	15.0586%
484	(19), $F_{load} = 16$	1	0	Tikh.	10^{-7}	110	4	21.2502%
484	(19), $F_{load} = 16$	1	0	Tikh.	10^{-7}	150	5	18.7010%
1936	(19), $F_{load} = 16$	1	0	Tikh.	10^{-7}	150	5	19.2354%

Table 1: Identification of λ, full displacement data measurement $\underline{U}_{data}$

toolbox. Thus, a more detailed study requires an improved and more accurate solution of the forward problem. See section 5.2 for a discussion of adaptive FE software.

$\sharp$elem.	boundary condition	ω	ϑ	regul. method	α	λ_0	$\sharp$iter.	ident. error
484	(20), $U_{load} = 2$	1	0	Tikh.	10^{-7}	110	7	19.0781%
484	(20), $U_{load} = 2$	1	10^{-5}	Tikh.	10^{-7}	110	7	18.6737%
484	(20), $U_{load} = 2$	1	10^{-4}	Tikh.	10^{-7}	110	7	17.0270%
1936	(20), $U_{load} = 2$	1	10^{-4}	Tikh.	10^{-7}	110	6	17.8963%

Table 2: Identification of λ, displacement data $\underline{U}_{data}$ measured at Γ_4

Concluding this study, we verify by the results of table 4, that a restriction of available data increases the identification error.

5 Future work, ideas and problems

5.1 Ideas and open problems

Following from the studies above, there are a lot of open questions:

- identification of parameters with multiple displacement measurements, e.g. the load-displacement curve w.r.t. the stepwise applied load is given, which refers to knowledge of the Dirichlet-to-Neumann map and should improve the solution
- numerical studies on the influence of noisy data and experimental design on the identification results
- simultaneous identification of heterogenous material parameters
- studies on the identifiability of parameters for nonlinear material laws (for 'real' large deformations it is expected, that the linear elasticity model may not be applicable and will lead to useless numerical results)
- identification of discontinuous parameter functions
- optimal choice of the weights ϑ_i, ω_i and the regularization parameter α_i
- solution of 3D identification problems
- identification of material parameters with experimental data

Furthermore, the question, if an implementation of different discretizations for $\underline{U}$ and $\underline{p}$ (such that $\mathcal{T} \neq \mathcal{T}_2$) may be appropriate, could be discussed.

5.2 Concept for an application of adaptive FEM

The solution of [IP] via (18) involves the numerical costs of a number of forward operator calculations. From this reason the repeated calls of a PDE solver should be as efficient as possible, e.g. by using adaptive FEM. The basic principle of adaptive FEM for the forward operator calculation with given p is, that an error estimator is implemented in the FEM solver, which estimates for each triangle in $\mathcal{T}$ an appropriately chosen error functional

$$e(U_{\mathcal{T}} - U) \ .$$

Here $U_{\mathcal{T}} \in \mathbb{V}_D^{(2n_U)}$ denotes the discrete solution of (1) and U corresponds to the (unknown) exact solution.

For the solution of the inverse problem [IP] the adaptivity has to be modified slightly, due to the fact that now the error in p is of interest. E.g. in [10] a strategy is suggested, where [IP] is solved for a coarse discretization and then for the corresponding discrete solution $p_{\mathcal{T}} \in \mathbb{Q}^{(2n_p)}$ an error functional

$$e(p_{\mathcal{T}} - p)$$

is estimated, which measures the discrepancy between $p_{\mathcal{T}}$ and the (unknown) exact p. Then these error estimation w.r.t. p is used for adaptive mesh refinement and the solution of [IP] is reiterated until a stopping criterion for $e(p_{\mathcal{T}} - p)$ is fulfilled. See [10] for details on the implementation of such error estimators. Note, that the author additionally suggests error estimators, which measure simultaneously the discretization errors in U and p.

Consequently, an existing FE code for the solution of the forward problem can also be used for adaptive mesh refinement in the inverse problem [IP], if the error estimator is modified appropriately. Depending on the 'quantity of interest' in the underlying inverse problem, several error estimating strategies may be applicable. E.g. in [3] the authors mention a couple of so called goal-oriented error estimators. Another concept of adaptive parameter identification is given by the adaptive grid regularization discussed in [7]. In this context we also mention multilevel methods as presented e.g. in [8]. Such methods start the parameter identification at a coarse discretization level and during an iteration the mesh is refined. Contrary to classical adaptive FEM strategies, here the discretization level is used as a regularization parameter and consequently the discretization is chosen such that optimal regularizing properties are reached.

Referring to the discussion above we present an algorithmic scheme for the solution of [IP] with adaptive mesh refinement. In this context we assume, that an adaptive PDE solver **FEtool** is available with the following properties:

- **FEtool** knows geometry, boundary conditions, material law, $a(U;V|p)$, $a_0(U;W,V|p)$, and $a_1(U;\Delta U,W,V|p)$
- for given initial mesh and parameter p, **FEtool** calculates $U = U(p)$, where adaptivity is controlled with an error estimator err_U for $e(U_{\mathcal{T}} - U)$
- for given mesh, U, and p, **FEtool** is able to estimate the error $e(p_{\mathcal{T}} - p)$ with an error estimator err_p and to refine the mesh w.r.t. err_p
- furthermore, for given mesh, U, and p, all matrices and vectors needed for the implementation of (18) should be available as output of **FEtool**

Under this assumption we derive an algorithmic scheme for the adptive solution of [IP].
Algorithm 5.1. Adaptive FEM identification algorithm according to [5, algorithm 3.1]:

```
START   set $k = l := 0$ and define a coarse initial mesh $\mathcal{T}^0$
        choose initial guess $\underline{p}_0^0$ and set $\underline{\xi}_0^0 := 0$
        calculate values $\underline{U}_0^0 = \underline{U}(\underline{p}_0^0)$ at nodes of $\mathcal{T}^0$:
                $[\underline{U}_0^0, \underline{f}^0, \underline{P}^0, \mathcal{T}^0] :=$ FEtool$(\underline{p}_0, \mathcal{T}^0, err_U)$
REPEAT
        REPEAT
            SQP ITERATION
            assemble matrices:
            $[\underline{A}(\underline{U}_k), \underline{A}_0(\underline{U}_k, \underline{\xi}_k), \underline{\tilde{A}}_0(\underline{U}_k, \underline{p}_k), \underline{\tilde{A}}_1(\underline{U}_k, \underline{\xi}_k, \underline{p}_k), \underline{Q}^l, \underline{K}^l, \underline{P}^l, \mathcal{T}^l]$
                    $:=$ FEtool$(\underline{U}_k^l, \underline{\xi}_k^l, \underline{p}_k^l, \mathcal{T}^l)$
            calculate a solution $\Delta\underline{U}, \Delta\underline{p}, \underline{\xi}^l$ of (18)
            update $\underline{U}_{k+1}^l := \underline{U}_k^l + \Delta\underline{U}$, $\underline{p}_{k+1}^l := \underline{p}_k^l + \Delta\underline{p}$, $\underline{\xi}_{k+1}^l := \underline{\xi}^l$
            $k := k + 1$
        UNTIL stopping rule of SQP iteration fulfilled
        estimate error of $\underline{p}_k^l$ and create refined mesh$\mathcal{T}^{l+1}$:
        $[\underline{U}_k^{l+1}, \underline{\xi}_k^{l+1}, \underline{p}_k^{l+1}, \underline{f}^{l+1}, \underline{P}^{l+1}, \mathcal{T}^{l+1}] :=$ FEtool$(\underline{U}_0^l, \underline{\xi}_0^l, \underline{p}_0^l, \mathcal{T}^l, err_p)$
        $l := l + 1$ and $k := 0$
UNTIL   $\mathcal{T}^l = \mathcal{T}^{l-1}$
```

Note, that the solution of (18) is calculated w.r.t. the homogeneous Dirichlet boundary conditions. Additionally we mention, that in algorithm 5.1 the complete SQP iteration is executed before some mesh refinement is done, which seems to be inefficient. A quite 'natural' modification would be, to refine the mesh w.r.t. err_p within the SQP iteration, which might influence the convergence of the SQP iteration.

The algorithm 5.1 is easy to implement in MATLAB, if an arbitrary (extern) **FEtool** is available with the explained properties. Applicable FE packages and options for the communication with MATLAB are e.g.:

- adaptive FE code SPC (FORTRAN based): link of MATLAB and SPC via MATLAB meshfiles or a master-slave communication

- COMSOL FE software: the package is basing on MATLAB and assessable from MATLAB

Besides an improvement of efficiency, we expect a higher accuracy of the identification results, if a well-developed extern adaptive FE code is used for the identification algorithm. The results in this study are disturbed by the limited skills of the MATLAB PDE-toolbox and therefore the presented results with more than 10% error even for noiseless data may be improved in future.

5.3 Application of Gradient methods

The introduced SQP algorithm for the solution of [IP] converges fast, but one drawback is, that each SQP iteration step is expensive, in particular due to the explicit computation of the (large dimensioned) system matrix in (18). Thus, for nontrivial FEM problems (with usually $> 10^5$ elements) troubles could arise because of memory overflow or exploding CPU times for solving (18). One idea to overcome such problems is the application of gradient methods (e.g. Landweber iteration) with an appropriate step size control. Compared with SQP iterations such methods have a slower convergence but cheaper iteration steps.

We discuss an ansatz for a gradient method solution of [IP]. Let in this context [IP] be given as finding a solution of the operator equation

$$F(U,p) := \begin{pmatrix} F_1(U,p) \\ F_2(U,p) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \tag{21}$$

where the operator $F : (H^1(\Omega_0))^3 \times Q \to Z^* \times Z^*$ is given in weak form as

$$\begin{aligned} \langle F_1(U,p), V\rangle_{(L^2(\Omega_0))^3} &= \langle \mathcal{P}U - U_{data}, \mathcal{P}V\rangle_{(L^2(\Omega_0))^3} && \forall V \in Z \\ \langle F_2(U,p), V\rangle_{Z^*,Z} &= \langle \mathcal{R_Z} F_2(U,p), V\rangle_Z = a(U;V|p) - f(V) && \forall V \in Z\,. \end{aligned}$$

Note, that in the last equation we have to consider $F_1(U,p)$ and $F_2(U,p)$ as elements of the dual space Z^*. Despite this, in the case of $F_1(U,p)$ we are able to identify $F_1(U,p) = \mathcal{P}^*(\mathcal{P}U - U_{data}) \in (L^2(\Omega_0))^3$ due to the above definition, and therefore the standard scalar product in $(L^2(\Omega_0))^3$ can be applied. This will not work for $F_2(U,p)$, which is not an element of $(L^2(\Omega_0))^3$. Thus, we have to introduce the corresponding Riesz isomorphism

$$\mathcal{R}_Z : Z^* \to Z\,.$$

Then $\langle\cdot,\cdot\rangle_Z$ denotes the standard scalar product in the Hilbert space $Z = (H_0^1(\Omega_0))^3$ and

$$\|F_2(U,p)\|_{Z^*} = \|\mathcal{R}_Z F_2(U,p)\|_Z$$

holds. A solution of (21) can be found by a least squares minimization

$$\frac{1}{2}\left(\beta_1\|F_1(U,p) - 0\|^2_{(L^2(\Omega_0))^3} + \beta_2\|\mathcal{R}_Z F_2(U,p) - 0\|^2_Z\right) \to \min_{U,p} \tag{22}$$

with weights β_1 and β_2. Consequently, a gradient method for finding a minimizer of (22) is e.g. the Landweber iteration

$$\begin{aligned} \begin{pmatrix} U_{k+1} \\ p_{k+1} \end{pmatrix} - \begin{pmatrix} U_k \\ p_k \end{pmatrix} &= -\beta_1[F_1'(U_k,p_k)]^*(F_1(U_k,p_k)) \\ &\quad -\beta_2[(\mathcal{R}_Z F_2(U_k,p_k))']^*(\mathcal{R}_Z F_2(U_k,p_k)) \end{aligned} \tag{23}$$

with the operators

$$[F_1'(U_k,p_k)]^* : (L^2(\Omega_0))^3 \to Z \times Q$$

and

$$[(\mathcal{R}_Z F_2(U_k,p_k))']^* : Z \to Z \times Q$$

denoting the (Hilbert space) adjoint derivatives of $F_1(U,p)$ and $\mathcal{R}_Z F_2(U,p)$ at the iterate (U_k,p_k). The directional derivatives

$$[F_1'(U,p)](\Delta U,\Delta p) \quad \text{and} \quad [(\mathcal{R}_Z F_2(U,p))'](\Delta U,\Delta p)$$

are given in weak form as

$$\begin{aligned}
\langle [F_1'(U,p)](\Delta U,\Delta p), V\rangle_{(L^2(\Omega_0))^3} &= \langle \mathcal{P}\Delta U, \mathcal{P}V\rangle_{(L^2(\Omega_0))^3} \qquad \forall V \in Z \\
\langle [(\mathcal{R}_Z F_2(U,p))'](\Delta U,\Delta p), V\rangle_{Z} &= a_0(U;\Delta U, V|p) + a(U;V|\Delta p) \qquad \forall V \in Z\ .
\end{aligned}$$

Hence we derive, that the adjoint derivatives hold

$$\begin{aligned}
\langle [F_1'(U,p)](W,q), V\rangle_{(L^2(\Omega_0))^3} &= \langle (W,q), [F_1'(U,p)]^* V\rangle_{(L^2(\Omega_0))^3 \times (L^2(\Omega_0))^3} \\
&= \langle W, \mathcal{P}^*\mathcal{P}V\rangle_{(L^2(\Omega_0))^3} + \langle q, 0V\rangle_{(L^2(\Omega_0))^3} \\
\langle [(\mathcal{R}_Z F_2(U,p))'](W,q), V\rangle_{Z} &= \langle (W,q), [(\mathcal{R}_Z F_2(U,p))']^* V\rangle_{Z \times (L^2(\Omega_0))^3} \\
&= a_0(U;W,V|p) + a(U;V|q)
\end{aligned}$$

for all $V, W \in Z$, $q \in Q$. Thus, by setting $V := \mathcal{R}_Z F_2(U_k,p_k)$ and renaming W as V in the last equation, the Landweber iteration (23) is written in weak formulation as

$$\begin{aligned}
\begin{pmatrix} \langle U_{k+1}, V\rangle_{(L^2(\Omega_0))^3} \\ \langle p_{k+1}, q\rangle_{(L^2(\Omega_0))^3} \end{pmatrix} &= \begin{pmatrix} \langle U_k, V\rangle_{(L^2(\Omega_0))^3} \\ \langle p_k, q\rangle_{(L^2(\Omega_0))^3} \end{pmatrix} \\
&\quad -\beta_1 \begin{pmatrix} \langle \mathcal{P}^*\mathcal{P}\mathcal{P}^*(\mathcal{P}(U_k) - U_{data}), V\rangle_{(L^2(\Omega_0))^3} \\ 0 \end{pmatrix} \\
&\quad -\beta_2 \begin{pmatrix} a_0(U_k; \mathcal{R}_Z F_2(U_k,p_k), V|p_k) \\ a(U_k; \mathcal{R}_Z F_2(U_k,p_k)|q)) \end{pmatrix} \qquad \forall V \in Z,\ q \in Q\ .
\end{aligned}$$

Note, that the convergence of the iteration can be accelerated by an appropriate step size control, which we omitted here.

Due to the above results, we have to calculate $f_{\mathcal{R}} := \mathcal{R}_Z F_2(U_k,p_k) \in Z$. With

$$\langle F_2(U_k,p_k), V\rangle_{Z^*,Z} = \langle f_{\mathcal{R}}, V\rangle_Z = \int_{\Omega_0} (f_{\mathcal{R}} \cdot V + \operatorname{Grad} f_{\mathcal{R}} \cdot \operatorname{Grad} V) \mathrm{d}\Omega_0$$

we derive, that $f_{\mathcal{R}}$ denotes the solution of the variational problem

$$\int_{\Omega_0} (f_{\mathcal{R}} \cdot V + \operatorname{Grad} f_{\mathcal{R}} \cdot \operatorname{Grad} V) \mathrm{d}\Omega_0 = a(U_k; V|p_k) - f(V) \quad \forall V \in Z\ ,$$

which means, that an additional elliptic PDE has to be solved.

For the discrete problem, the Landweber iteration is formulated as

$$\begin{aligned}\begin{pmatrix} \underline{U}_{k+1} \\ \underline{p}_{k+1} \end{pmatrix} &= \begin{pmatrix} \underline{U}_{k} \\ \underline{p}_{k} \end{pmatrix} - \beta_1 \begin{pmatrix} \underline{P}^T(\underline{P}\,\underline{U}_k - \underline{U}_{data}) \\ \underline{0} \end{pmatrix} \\ &\quad - \beta_2 \begin{pmatrix} \underline{\tilde{A}}_0(\underline{U}_k, \underline{p}_k)\underline{f}_{\mathcal{R}} \\ \underline{A}(\underline{U}_k)^T \underline{f}_{\mathcal{R}} \end{pmatrix} .\end{aligned}$$

with $\underline{f}_{\mathcal{R}}$ solving the linear system

$$\underline{K}_{\mathcal{R}} \underline{f}_{\mathcal{R}} = \underline{A}(\underline{U}_k) - \underline{f} ,$$

where $\underline{K}_{\mathcal{R}}$ denotes the stiffness matrix corresponding to the PDE for calculating $f_{\mathcal{R}}$.

For the sake of completeness, we mention that regularization may be introduced by an appropriate choice of the termination index for the Landweber iteration or additional regularization terms in (22).

Remark 5.1. Instead of using the Riesz isomorphism $\mathcal{R}_Z$, we could apply any operator $\mathcal{R} : Z^* \to Z$, that transforms F_2 from the dual space Z^* into the function space Z as a preconditioner, such that the search direction $(\Delta U, \Delta p)^T$ depends continuously on U and p. Applicable are e.g preconditioners as implemented in a conjugate gradient method. In our context, the above mentioned Riesz isomorphism $\mathcal{R}_Z$ can be seen as a special case of such preconditioners.

References

[1] Braess, D.: *Finite elements. Theory, fast solvers and applications in solid mechanics.* Cambridge: Cambridge University Press, 2007.

[2] Görke, U. J.; Bucher, A.; Kreißig, R.: *Zur Numerik der inversen Aufgabe für gemischte (u/p) Formulierungen am Beispiel der nahezu inkompressiblen Elastizität bei großen Verzerrungen.* (German), TU Chemnitz, CSC-preprint 07-07, 2007.

[3] Griesbaum, A.; Kaltenbacher, B.; Vexler, B.: *Efficient computation of the Tikhonov regularization parameter by goal-oriented adaptive discretization.* Inverse Probl., 24(2):Article ID 025025, 20 p, 2008.

[4] Meyer, A.: *Grundgleichungen und adaptive Finite-Elemente-Simulation bei Großen Deformationen.* (German), TU Chemnitz, CSC-preprint 07-02, 2007.

[5] Meyer, M.: *Parameter identification problems for elastic large deformations – Part I: model and solution the inverse problem.* TU Chemnitz, CSC-preprint 09-05, 2009.

[6] Schade, H.: *Tensoranalysis.* (German), Walter de Gruyter, Berlin, New York, 1997.

[7] Neubauer, A.: *Computation of discontinuous solutions of 2D linear ill-posed integral equations via adaptive grid regularization.* J. Inverse Ill-Posed Probl., 15(1):99-106, 2007.

[8] Scherzer, O.: *An iterative multi level algorithm for solving nonlinear ill-posed problems.* Numer. Math. 80, No.4, 579-600, 1998.

[9] The MathWorks, Inc.: *Partial Differential Equation Toolbox*^{TM} *User's Guide.* MATLAB software documentation, www.mathworks.com, 2009.

[10] Vexler, B.: *Adaptive finite element methods for parameter identification problems.* Heidelberg: Universität Heidelberg, Naturwissenschaftlich- Mathematische Gesamtfakultät (Dissertation 2004)., 2004.

Chemnitz Scientific Computing Preprints – ISSN 1864-0087